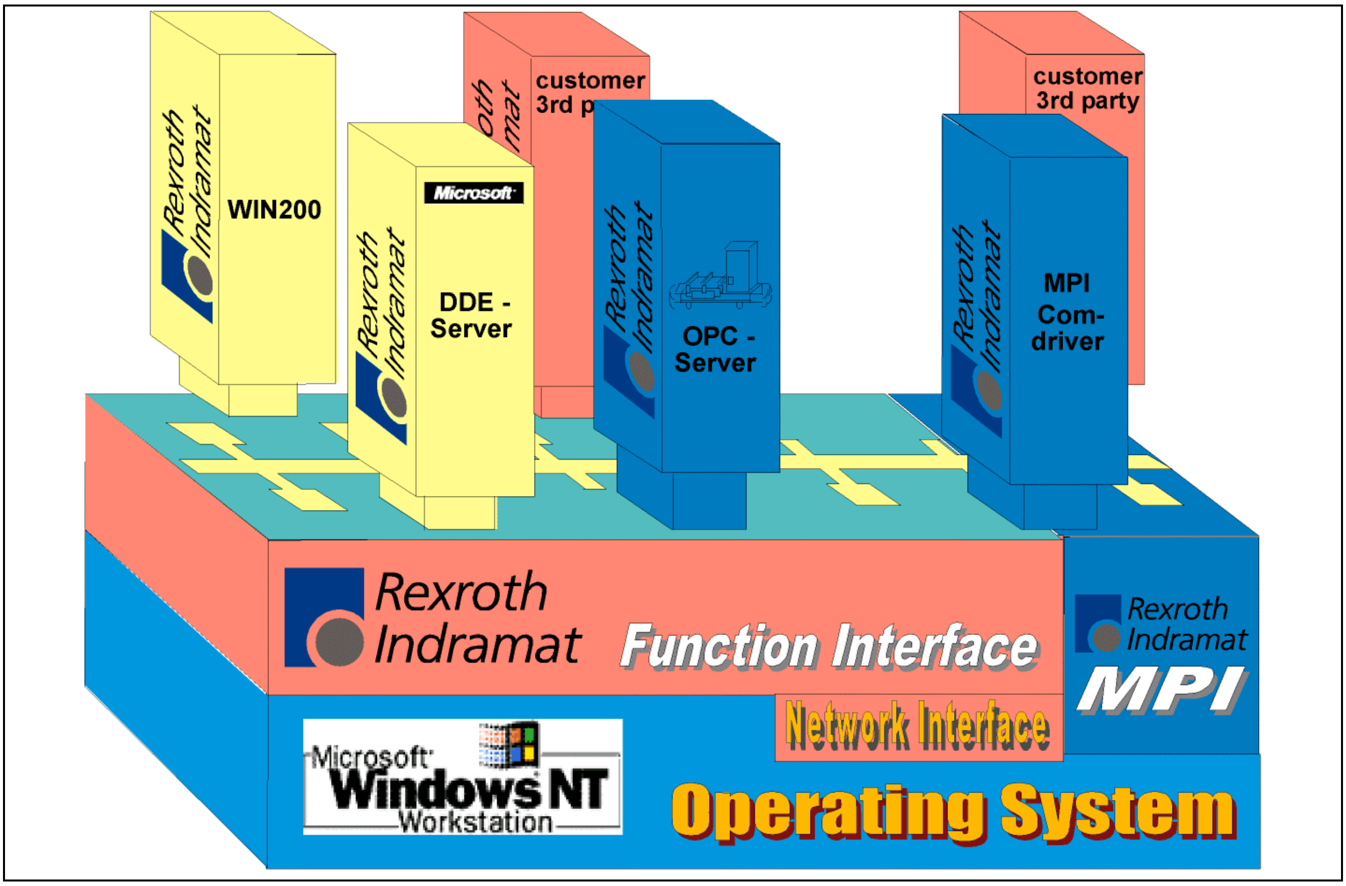
Rexroth



MTC200/ISP200/MTA200/TRANS200 Function Interface 07VRS

Application Manual

SYSTEM200

DOK-CONTRL-FUN*INT*V07-AW01-EN-P

Rexroth Indramat

Title MTC200/ISP200/MTA200/TRANS200
Function Interface
07VRS

Type of Documentation Application Manual

Document Typecode DOK-CONTRL-FUN*INT*V07-AW01-EN-P

Internal File Reference Document Number 120-0400-B375-01/EN

Purpose of Documentation This documentation is used:

- to give an overview of the function interface functionalities
- to define the application possibilities and
- for projecting and development of the user-defined user interfaces in C/C ++ and Visual Basic.

Record of Revisions

Description	Release Date	Notes
120-0400-B375-01/EN	10.02	Valid from version 22

Copyright © 2002 Rexroth Indramat GmbH

Copying this document, giving it to others and the use or communication of the contents thereof without express authority, are forbidden. Offenders are liable for the payment of damages. All rights are reserved in the event of the grant of a patent or the registration of a utility model or design (DIN 34-1).

Validity The specified data is for product description purposes only and may not be deemed to be guaranteed unless expressly confirmed in the contract. 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 +49 (0)93 52/40-0 • Tx 68 94 21 • Fax +49 (0)93 52/40-48 85

http://www.boschrexroth.de/

Dept. BRC/ESS1 (FW)

Dept. BRC/ESM6 (DiHa)

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

Contents

1 New in Versions 07VRS

1.1 Version 07V00

General Information

- The Visual Motion device group MWMX was included. For the commands, see the MWSX device group, as the WinPcl part is identical with this group. The Visual Motion component has been realized under SCP (Scalable Communication Platform).
- The MWYX device group has been included. The commands are a combination of the MSYX and MWSX device group, but united in a chapter of their own.
- The item „COM – Automation Interface" was included in the chapter "Programming".
- The FI command "PVS" was removed from the description as the existing FI command "PVF" possesses the efficiency of "PVS", plus the array handling. Accordingly, do not use "PVS" any more.
- From IF Version 07, the FI commands "DPN", "DPP" "IPP", "NPC", "NPD", "NPI", "PPD" and PPN" are re-activated and have been included in the description (MWCX device group).

New FI Commands

- FI command "ANM" supplies the size of the current NC magazine (MWCX device group).
- FI command "ASD" supplies the current spindle data (MWCX device group).
- FI command "CCA" causes an Upload of NC cycles by an Upload file (MWCX device group).
- FI command "CEI" displays the counts of the logged communication errors (MPCX device group).
- FI command "DCA" causes an Upload of NC D corrections by an Upload file (MWCX device group).
- FI command "DCT" sets the Timeout for a device or sets the time back to the default value (MWCX-, MWSX-, MWMX-, MWAX-, MSYX-, MWYX- and MSCX device group).
- FI command "DSF" deletes the FI command Stack management (MPCX device group).
- FI command "ICA" initializes a communication address with new parameters (MPCX device group).
- FI command "IFS" supplies the current occupancy state of FI command Stack management (MPCX device group).
- FI command „LDT" reads and writes the local PC date and the local PC time of the day (MPCX device group).
- FI command "MDA" has been extended by the command MDA4 with which all Machine Data Page definitions can be deleted in the selected device (MWCX device group).
- FI command "MSG" is used to read System Messages (MPCX-, MSCX-, MWCX-, MWSX-, MWMX-, MWAX- device group).
- FI command "MTC" is used to read the Firmware identification from the different control components (MWCX-, MWSX-, MWMX- and MWAX- device group).

- FI command "NCA" causes an Upload of NC programs by an Upload file (MWCX device group).
- FI command "NEA" causes an Upload of NC events by an Upload file (MWCX device group).
- FI command "NUA" causes an Upload of NC zero points by an Upload file (MWCX device group).
- FI command "NVA" causes an Upload of NC variables by an Upload file (MWCX device group).
- FI command "PAD" sets a parameter set inactive if the device is in Offline mode (MWCX device group).
- FI command "PAF" sorts a parameter download file (MPCX device group).
- FI command "PAS" sets a parameter set active if the device is in Offline mode (MWCX device group).
- FI command "PDD" supplies data for the ProVi criteria analysis (MWCX-, MWSX-, MWMX- and MWAX device group).
- FI command "POB" is used to write and read a PC port address (BYTE access) (MPCX device group).
- FI command "POI" supplies the current position information of all axes (MWAX device group).
- FI command "POW" is used to write and read a PC port address (WORD access) (MPCX device group).
- FI command "PVA" is used to write and read Provi Message files (MWCX-, MWSX-, MWYX- and MWAX device group).
- FI command "PVM" was extended by the command "PVM4" (MWCX-, MWSX-, MWMX- and MWAX device group).
- FI command "PVR" executes an Upload or Download of PLC retain variables (MWAX-, MWCX-, MWMX-, MWSX- and MWYX device group).
- FI command "REP" supplies data to return to the contour (MWCX device group).
- FI command "SDP" starts a FI device interrogation cycle (MPCX device group).
- FI command "SDS" sets the device status (ON/OFF) which is also entered in IND_DEV.ini (MSCX-, MWCX-, MWSX-, MWMX-, MSYX-, and MWAX device group).
- FI command "TPI" supplies information about grippers, spindles and tool magazine locations (MWCX device group).

Modifications of FI Commands

- FI command "AMM" reports the active mechanism errors and was increased by a file name for additional information for the message text or the extended text in the return value (MWCX-, and MWAX device group).
- FI command "API" supplies always 0 as index of the active parameter set (MWCX device group).
- FI command "APP" supplies the active NC block number and was extended in the reply by an output in the setup mode.
- FI command "ART" was extended by "**binary** reading of the current axis reference table for an Offline device" (DeviceStatus=OFF) (MWCX device group).
- FI command "ASM" reports the active system errors and was increased by a file name for additional information for the message text or the extended text in the return value (MWCX-, and MWAX device group).

- FI command "CCP" supplies the configuration settings from IND_DEV.ini and was extended by columns 14 (device protocol) and 15 (device simulation) in the reply (MPCX device group).
- FI command "DCD" supplies the value of D correction index and was extended by measuring unit [mm/inch] in the return (MWCX device group).
- FI command "DCR" reads and writes the values of a D correction set and was extended by measuring unit [mm/inch] in the return (MWCX device group).
- FI command "DIS" was revised and realized as B command (MWCX-, MWSX-, MWMX-, MWAX device group).
- FI command "DSI" supplies the most essential information about device status and was extended by columns 12 (device simulation switched on) and 13 (device status information) in the reply (MSCX-, MWCX-, MWSX-, MWMX-, and MWAX device group).
- FI command "DTY" was revised and realized as B command (MWCX-, MWSX-, MWMX-, MWAX-, MSCX-, MWSYX- and MWYX device group).
- FI command "DWD" supplies diagnosis messages and was extended in return value by the criteria analysis and a file name for additional information (MWCX-, MWSX-, MWMX-, and MWAX device group).
- FI command "ERI" now also supplies an error text with a WIN NT error code (MPCX device group).
- FI command "MKT" is used to write the GUI-SK 16 block in the PLC and was extended by command MKT2 (MWCX-, MWSX-, MWMX- and MWAX device group).
- FI command "NCM" supplies NC messages and was extended by a file name for additional information for the message text in the return value (MWCX device group).
- FI command "NVS" was set during writing NC variables to data type "LONG" or "Doublereal" (MWCX device group).
- FI command "PVM" supplies ProVi messages and was enlarged in return value by the criteria analysis and a file name for additional information for the message type or the extended text (MWCX-, MWSX-, MWMX-, and MWAX device group).
- FI command "ZOD" was extended by measuring unit [mm/inch] during reading an offset page (MWCX device group).

Authorized Errors

- FI command "DIS1" supplies "--" in all result columns if no valid parameter set is in the selected device (MWCX device group).

1.2 Version 06V00

General Information

- Chapter 1 has been extended to provide information on safety under the heading “Protection against dangerous movements”.
- The section in the chapter entitled “Programming” concerning SYS messages has been revised.
- A separate chapter entitled “Literature” has been appended.
- Documentation of previously undocumented and new commands for the software standard 05-21V00 WIN-HMI.
- Box 19 in the table on basic value range data is used to classify tools. The user can no longer edit (MWCX device group).
- The MTCX device group has been almost entirely converted to the MWCX device group. This has created a designation for the newly introduced WinPcl.
- Only the MTVNC remains in the MTCX device group. Its instruction set is a subset of the MWCX device group. Individual commands are listed only in table form with detailed reference to explanations in the MWCX device group.
- The MISX device group has been converted to the MWSX device group. This has created a designation for the newly introduced WinPcl.
- The MTAX device group has been converted to the MWAX device group. This has created a designation for the newly introduced WinPcl.
- From IF Version 06 the FI commands "IPP", "NPC", "NPD", "NPI", "PPD" and PPN" are no longer valid and have been removed from the description (MWCX device group).
- The list of error codes has been extended.

New FI Commands

- The FI command "ADM" supplies all messages from the Andron NC (MWAX device group).
- The FI command "ARF" indicates the reference flags of an axis for a process (MWCX device group).
- The FI command "ART" returns the complete axis reference tables of a system (MWCX device group).
- The FI command "ATR" returns the complete basic data and cutter data of the current processing tool (MWCX device group).
- The FI command "ATU" causes the NC to accept the data record of the current tool changed after editing (MWCX device group).
- The FI command "BFJ" interrupts FI jobs (MPCX device group).
- The FI command "CCA" triggers the downloading of NC cycles by means of the download file (MWCX device group).
- The FI command "CPR" starts a Win32 application (MPCX device group).
- FI command "DCA" triggers the downloading of NC D corrections by means of the download file (MWCX device group).
- The FI command "DPR" exits a Win32 application (MPCX device group).
- The FI command "DSI" returns the most important information on the status of the device (MSCX, MWCX, MWSX, and MWAX device groups).
- The FI command "DTC2" returns tool management data such as basic user data and cutter user data (MWCX device group).

- The FI command "DWD" outputs all diagnostic messages (MWCX, MWSX, and MWAX device groups).
- The FI command "EAD" returns which Andron diagnostic types are available in a module (MWAX device group).
- The FI command "EDE" returns whether or not there are errors present (MWCX, MWSX, and MWAX device groups).
- The FI command "EDW" returns all diagnostic window types (MWCX, MWSX, and MWAX device groups).
- The FI command "END" returns which NC diagnostic types are available in a module (MWCX device group).
- The FI command "EPD" returns which PLC diagnostic types are available on a control unit (MWCX and MWSX device groups).
- The FI command "EPT" returns all ProVi types (MWCX, MWSX, and MWAX device groups).
- The FI command "EST" queries the error state of a variable (MWCX, MWSX and MWAX device groups).
- The FI command "EXD" shows the extent to which a step or action can be executed (MWCX, MWSX and MWAX device groups).
- The FI command "GDB" writes/reads the data for the general FI data buffer (MPCX device group).
- The FI command "MDA" uploads or downloads machine data records (MWCX device group).
- The FI command "MDS" is for writing and reading machine data (MWCX device group).
- The FI command "MFD" loads the message text into the device indicated (MWCX, MWSX, and MWAX device groups).
- The FI command "MKS" returns information on the machine buttons (MWCX, MWSX, and MWAX device groups).
- The FI command "MKT" writes the GUI-SK16 block in the PLC (MWCX, MWSX and MWAX device groups).
- The FI command "NCA" triggers the downloading of NC programs by means of the download file (MWCX device group).
- The FI command "NCM" returns all messages from the Rexroth Indramat NC (MWCX device group).
- The FI command "NEA" triggers the downloading of NC events by means of the download file (MWCX device group).
- The FI command "NPA5" returns a list of a maximum of 10 parameters of the same parameter type (MWCX device groups).
- The FI command "NST" brings the operating system to a stop (MPCX device group).
- The FI command "NUA" triggers the downloading of NC zero points by means of the download file (MWCX device group).
- The FI command "NVA" triggers the downloading of NC variables by means of the download file (MWCX device group).
- The FI command "PAA" uploads or downloads parameter records (MWCX and MTAX device groups).
- The FI command "PDT" returns parameters from the definition table for the selected device in binary form (MWCX device group).
- The FI command "PHD" generates a physical directory with the assistance of the BDI (MPCX, MSCX, MWAX, MWSX, MWCX, and MSYX device groups).

- The FI command "PSM" initiates the most important SYS messages with regard to the PCL programming interface (MWCX, MWSX and MWAX device groups).
- The FI command "PVM" returns all ProVi messages (MWCX, MWSX, and MWAX device groups).
- The FI command "RPR" informs the caller that the program is now active (MPCX device group).
- The FI command "SDD" returns data for the step chain diagnosis (MWCX, MWSX, and MWAX device groups).
- The FI command "SFD" returns data for a step chain (MWCX, MWSX, and MWAX device groups).
- The FI command "SFE" returns all the step chain messages for a module (MWCX, MWSX, and MWAX device groups).
- The FI command "SFM" returns the operating mode of a step chain (MWCX, MWSX, and MWAX device groups).
- The FI command "SFW" focuses the screen (MPCX device group).
- The FI command "SSM" is for issuing SYS messages (MPCX device group).
- The FI command "WLA" requests free watch list allocations (MWCX-device group).
- The FI command "WLF" frees up requested watch list allocations (MWCX-device group).

Modifications of FI Commands

- The FI command "AAS" returns information about the current axis speed and, in return, has been extended by an additional unit of measurement [inch/min] (MWCX device group).
- The FI command "AFR" returns information about the current feed velocity and, in return, has been extended by an additional unit of measurement [inch/min] (MWCX device group).
- The FI command "APO" returns the current axis position value and, in return, has been extended by an additional unit of measurement [inch] (MWCX device group).
- The FI command "CPO" returns the current axis position command value and, in return, has been extended by an additional unit of measurement [inch] (MWCX device group).
- The FI command "DTG" returns the distance to go of an axis and, in return, has been extended by an additional unit of measurement [inch]" (MWCX device group).
- The FI command "EPO" returns the end point of an axis movement and, in return, has been extended by an additional unit of measurement [inch]" (MWCX device group).
- The FI command "MFR" returns the maximum feedrate and, in return, has been extended by an additional unit of measurement [inch/min]" (MWCX device group).
- The FI command "OPD" returns the optimum position distance of an axis and, in return, has been extended by an additional unit of measurement [inch]" (MWCX device group).
- The FI command "PFR" returns the value for the programmed feedrate and, in return, has been extended by an additional unit of measurement [inch/min]" (MWCX device group).
- The FI command "SLA" returns the current servo lag of an axis and, in return, has been extended by an additional unit of measurement [inch]" (MWCX device group).

- The FI command "TDR" returns the complete basic data and tool edge data of a tool and has been extended by information concerning optional data elements (MTCX device group).
- The FI command "TLD" returns elements of the basic data and cutter data of a tool and has been extended by information concerning optional data elements (MTCX device group).

1.3 Version 05V00

General Information

- Documentation of previously undocumented and new commands for the software standard 05-19V00 WIN-HMI.
- The chapter entitled "Practical Use of Tool Commands" has been included to better explain the tool commands (MTCX device group).

FI Commands Changes / Additions

- The FI command "ATP" returns information on the current tool location (MTCX device group).
- The FI command ”ERI1” returns the error text and the additional text of an FI error code or a NACK error number (MPCX device group).
- The FI command "MAR" reads the reference names of a PLC variable (MTCX device group):
- FI command "MTD1" for reading and writing machine user data has been expanded (MTCX device group). The FI command "MTD" is no longer used for reading!
- The FI command "PVF" is for the formatted reading and writing of PLC variables, arrays and structures (MTCX device group).
- The FI command "PVS" is for reading and writing PLC variables, arrays and structures (MTCX device group).
- The FI command "PVT" reads the declaration of PLC variables, including structures and arrays (MTCX device group).
- The FI command "TDR" returns the complete basic data and tool edge data of a tool (MTCX device group).
- The FI command "TLB" returns the basic data of the tool list (MTCX device group).
- The FI command "TLD" returns elements of the basic data or cutter data of a tool in the tool memory (MTCX device group).
- The FI command "TLE" returns the cutter data of the tool list (MTCX device group).
- The FI command "TII" initiates a tool replacement (MTCX device group).
- The FI command "TMV" moves an entire tool data record comprising the basic data and defined cutter data (MTCX device group).
- The FI command "TRS" resets the percentage life time of a tool to 100% (MTCX device group).
- The FI command "TIF" terminates a tool replacement (MTCX device group).

- The result [*X] of an FI command with invalid parameters has been changed into [--] (2 hyphens). This concerns the following commands of the MTCX device group:
  AAS1, AAS2, APO1, APO2, ASO1, ASS, CPO1, CPO2, DTG1, DTG2, EPO1, EPO2, MSS, OPD1, OPD2, PSS, SLA1, SLA2, TQE1, TQE2.
- Chapter 05 contains the table “Logical Linking of FI Commands” with direct links to the individual commands in the help file.
- The table “Availability of FI Commands” in Chapter 05 has been split into separate sections for each device group. It now contains direct links to the individual commands in the help file.
- The new device group MSYX (SYNAX200-P, SYNAX200-R) has been incorporated into the documentation.

1.4 Version 04V03

General Information

- Documents previously undocumented and new commands for the software standard 05-18V06 WIN-HMI.
- Includes a table with logical links for the FI commands in Chapter 5.
- Inclusion of a table above the command times in chapter 05.
- Includes the component types for the NC and PLC hardware in the file "IND_DEV.INI". Expansion of the FI command "DTY" by the output of the component types "Componenttype1=" and "Componenttype2=".

Identification of versions, Rexroth Indramat software components

Entries in the "C:\IND_BASE\INDRAMAT.INI" file:

- IfDllMode = 04.20
- IfVersion = 04V03

Software components contained normally within the function interface :

all Rexroth Indramat System 200 MT-CNC user interfaces for Version 18V06.

FI Commands Changes / Additions

- The FI command "AMM7" for outputting active mechanism messages and errors (MTAX device group).
- FI command “DCD1”: The values of a D-correction register are read out (MTCX device group).
- The FI command "DTC1" returns the most important system parameter data of the tool management (MTCX device group).
- The FI command ”FIT1” returns the additional text of an FI error code or a NACK error number (MPCX device group).
- The FI command "PTC1" returns the tool management data of all defined NC processes. The FI command "PTC2” returns the tool management data of a defined NC process. Only for the MTCX device group.
- The FI command "PPD" reads an NC program directory (MTCX device group).
- The FI command "PPN" converts an NC program from the NC program directory into an ASCII file and vice versa (MTCX device group).
- The FI command "PPP" changes the designation of an NC program package (MTCX device group).
- The FI command "PVF" handles the formatted reading and writing of PLC variables, arrays and structures (MTCX, MISX and MTAX device groups).

- The FI command "PVT" reads the type of PLC variables, including structures and arrays with their elements (MTCX, MISX and MTAX device groups).
- The FI command "SID1" returns information regarding the installation. This information includes installation paths, the software version used, DLL mode, plus service pack and release information. Valid for all device groups.
- The FI command "SLI" returns the single data from the SPS long ID such as the number, name and length of the program, the compiling date and more (MTCX, MISX and MTAX device groups).
- The FI command "TII" initiates a tool replacement (MTCX device group).
- The FI command "TIF" initiates the end of a tool replacement (MTCX device group).
- The FI command "TLB1", or "TLB2" returns basic data of the tool list such as tool memory, designation, correction type and more (MTCX device group).
- The FI command "TLE1", or "TLE2" returns cutter data of the tool list such as tool memory, location number, tool status and more (MTCX device group).
- The FI command "DCR1" has been implemented for reading and writing the D-correction register with the newly formatted output. (MTCX device group). The FI command "DCR" is no longer used!
- The FI command "DTY1" for outputting the device type has been expanded by the corresponding components. (MTCX, MSCX, MISX, and MTAX device groups). The FI command "DTY" is no longer used!
- The FI command "ZOD" for reading and writing data from the zero offset table has been expanded to include the FI commands "ZOD1" and "ZOD2" (MTCX and device group). The FI command "ZOD" is no longer used!
- New, speed-optimized FI command "GPP" for reading out the global process parameters. (MTCX device group).
- New, speed-optimized FI command "NPD3 and "NPD4" for the NC download of small NC part programs. (MTCX device group).
- "NPD1" and "NPD2" commands for the NC download expanded by the value to be written "Initialization" (MTCX device group).
- Expansion of the FI command "CCP" by the output of the component types "Componenttype1=" and "Componenttype2=" (MPCX device group).
- New FI command "CCP5" for outputting the configuration data of the device that is addressed via the indicated device address (MPCX device group).
- New FI commands "CMA, CMF and CMI" for reading and writing of CMOS RAM, ASCII, floating point and integer parameters. (MTAX Device Group)
- New FI command "CRT" for triggering a control reset for the selected device (MTAX device group).
- New FI command "NPS" for preselecting the NC program located in the NC memory for processing (MTCX device group).
- New FI command "NMM" for selecting the NC memory for the NC program processing (MTCX device group).
- Addition to the documentation of the FI command "SPA" of the Ident. Number formats. The error return in the event of a form error in the value to be written has also been improved (MTCX, and MSCX device group).

- New FI command "TDAx, TMV and TRM" for editing complete tool data records (MTCX device group).

1.5 Version 04V02

General Information

- New chapter in the documentation: "Answers to Frequently Asked Questions (FAQ)".
- Most resource wastage has been eliminated in Service Pack 2 of the MT-CNC user interface GUI 18V05.

Identification of versions, Rexroth Indramat software components

Entries in the "C:\IND_BASE\INDRAMAT.INI" file:

- IfDllMode = 04.10
- IfVersion = 04V02

Software components contained normally within the function interface :

- all Rexroth Indramat System 200 MT-CNC user interfaces for Version 18V05 with Service Pack 3.

FI Commands Changes / Additions

- Module commands MCD1, MCM1 and MCS1 enabled for the MISX device group.
- Module commands MCD1, MCM1, MCP1, MCS1, MAP1 enabled for the 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

- Waste of resources in logic process resolved.
- Expansion from 15 to a max. of 255 group requests during cyclic requests (see "Routines for Cyclic Reading via Pipes").

1.6 Version 04V01

General Information

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

Identification of versions, Rexroth Indramat software components

Entries in the "C:\IND_BASE\INDRAMAT.INI" file:

- IfDllMode = 04.10
- IfVersion = 04V01

Software components contained normally within the function interface :

- all Rexroth Indramat System 200 MT-CNC user interfaces for Version 18V05.

FI Commands Changes / Additions

- Expansions to the device-independent access functions.
- New FI command "CRT" for triggering a control reset. (MTCX and MISX device groups).

Basic processes Changes / Additions

- Error correction of the telegram optimizer (correction of timeout recognition).
- New SYS message "MSG_PC__ALIVE" in PC network.

1.7 Version 04V00

In contrast to the previous 03VRS versions, fundamental changes have been made in this version:

General Information

- Delivery of a Visual Basic example connection to the function interface (application including source codes).
- Delivery of printed documentation as online help in Windows NT/95 help format.
- Provision of an installation program for the function interface.
- New! FI commands for an NC download. (MTCX device group)

Identification of versions, Rexroth Indramat software components

Entries in the "C:\IND_BASE\INDRAMAT.INI" file:

- IfDllMode = 04.00
- IfVersion = 04V00

Software components contained normally within the function interface :

- all Rexroth Indramat System 200 MT-CNC user interfaces for Version 18V04 with Service Pack 2.

FI Commands Changes / Additions

- FI command "XYZ" are implemented as "XYZ1" with re-formatted output: AAC1, AAS1, ADN1, AFO1, APO1, ARO1, ASO1, AZB1, MFO1, MRO1, MSO1. The FI command "XYZ" should no longer be used!
- The FI command "ABN" has been replaced by the FI commands "ASM" and "AMM".
- 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 the BOF/GBO. Support for Rexroth Indramat TRANS200 device types.

Note: The Rexroth Indramat devices TRANS200-P and TRANS200-R are still at the development stage and therefore cannot yet be ordered.

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

Basic processes Changes / Additions

- Error correction of the telegram optimizer (correction of timeout recognition).
- Expansion in the PLC data optimizer.
- Enlarged input buffer for the telegram optimizer.
- Reworking of the internal interface.
- Error correction in data provision by means of the "ReadGroupItem" routine.
- Error correction of the communication channel.
- Error corrections in the internal DLL interfaces.
- Correction of the INDIF200.DLL (correction of the binary result for spindle data).
- Changes in LogOutIf(), with regard to the selective KILLTASK

- Reworking of the COMVIEW interface for WIN200.
- Moving of the new SYS-Message interface into the file "INDIF000.H".

1.8 The Data Interface Newsletter

We will be informing you by email of new developments and updates to the Rexroth Indramat Products MPI and Function Interface.

Please send an email request with the message **subscribe** to:

owner-ml_datainterface@proxy.indramat.de

To unsubscribe, please proceed identically, but instead write

Message: **unsubscribe.**

Note:	Your email address will be kept confidential and not passed on to third parties.

2 General

2.1 Introduction

The Rexroth Indramat Function Interface is a unified data interface produced by Indramat for application programs (sometimes referred to as clients) based on the Windows NT platform.

Requirements

To obtain free access to data on existing NC and PLC data, it is necessary to provide a data interface that is as open, reasonably priced and as simple to handle as possible. The main requirements are to be able to access CNC/PLC data with a large range of functions and rapid access and reaction speeds. Several clients can access the data.

Objective

The Rexroth Indramat Function Interface aims to do exactly that, i.e. it allows access to all required control data via a compact, functional interface. This therefore allows the customer to completely create his own user interface in the programming languages Visual C++ or Visual Basic. The user is thus provided with a powerful interface with which he can communicate with Indramat devices and user interfaces using mnemonic function calls. The Function Interface is therefore a universal solution for data communication.

Availability

This description is valid for the following versions:

WinHMI:	22Vxx
Function Interface:	07Vxx
Windows NT Workstation/Server:	4.0
Visual C++:	5.0 + 6.0
Visual Basic:	5.0 + 6.0

2.2 The Function Interface from the User's Point of View

The Function Interface is a client (service requester) – server (service provider) interface and provides the user with a library (DLL) for communication services. The services, i.e. the functions of the DLL, fulfill the communication tasks that are required for reading individual data, the cyclic reading of data, the cyclic reading of data groups, the writing of data and for processor communication with Rexroth Indramat user interfaces and devices.

The Function Interface can communicate with a maximum of ten independent user programs (clients). This means a user program can, for example, be a customized user interface, a Rexroth Indramat OPC-Server or a communication driver to another data interface.

Up to ten parallel communication channels, one for each client, are supported between the Function Interface and the device. One communication channel can connect with a maximum of 64 data terminal devices.

The physical communication address in this case can be a serial interface (RS232/RS485), a Dual-Port-RAM or a Shared Memory area.

2.3 Protection against dangerous movements

Dangerous movements can be caused by the faulty control of connected motors. The reasons can be extremely varied:

- careless or faulty wiring or cabling,
- errors in operating the components,
- faults in the measured-value and signal transmitters,
- defective components, and/or
- errors in the software.

These faults can occur immediately after switching on or at any time during operation.

The Rexroth Indramat Function Interface is communication software which can be used to change the values of variables in the control unit.

As far as possible, monitoring in the drive components precludes faults in the connected drives. Where personnel safety is concerned, particularly where there is a risk of physical injury and/or damage to property, this fact should not be relied on exclusively. Until the built-in monitoring systems become active faulty drive movement is always to be expected; the degree of movement depends on the type of control unit and the operating status.

DANGER

Dangerous movements! Risk of death, injury, severe physical injury or damage to property!

⇒ For the reasons given above, protection of personnel is to be guaranteed by means of monitoring or other higher-ranking measures within the system.

For this purpose risk and fault analysis are to be provided for by the system designer according to the specific conditions within the system. The safety regulations applicable for the system are also to be taken into consideration. Arbitrary movements in the machine or other erratic functions can occur if safety devices are switched off, bypassed or activated wrongly.

To avoid accidents, physical injury and/or damage to property:

⇒ Do not stay within the motional range of the machine or machine parts. Possible measures to prevent personnel accidentally accessing the machine:

- protective fencing
- protective grid
- protective cover
- light barrier

⇒ Fencing and covers must be adequately secured against the maximum possible force of movement.

⇒ Position emergency stop switches within the immediate vicinity and so that they are easily accessible. Check that the emergency stop equipment is functioning before start-up. Do not operate the device if the emergency stop switch is not functioning correctly.

⇒ Protect against the device starting unintentionally by providing safety isolation for the drive's power connection by means of an emergency stop circuit or by using a safe starting lockout function.

⇒ Before accessing or entering the danger area bring the drives safely to a standstill.

⇒ Secure vertical axes against falling or slipping after switching off the motor power by, for example:

- mechanically locking the vertical axis,
- providing external brake/catching/clamping mechanisms or
- adequately counterbalancing the axis.

The standard motor holding brake provided or an external motor holding brake controlled directly by the drive controller are not sufficient on their own to guarantee the safety of personnel!

⇒ De-energize electrical equipment by means of the main switch and secure against reconnection during:

- maintenance and repair work
- cleaning work
- lengthy breaks in operation

⇒ Avoid operating high-frequency, remote controlled and radio devices in the vicinity of the device electronics and their power supply cables. If the use of these devices cannot be avoided, check the system and installation for possible faults in all working areas before switching on the system. If necessary, the system will require a special EMC test.

3 Structure and Configuration Examples

3.1 The Structure of the Function Interface

Viewed as a complete component, the function interface consists of the following three basic processes:

- Logic process
- Communication process and
- Management process

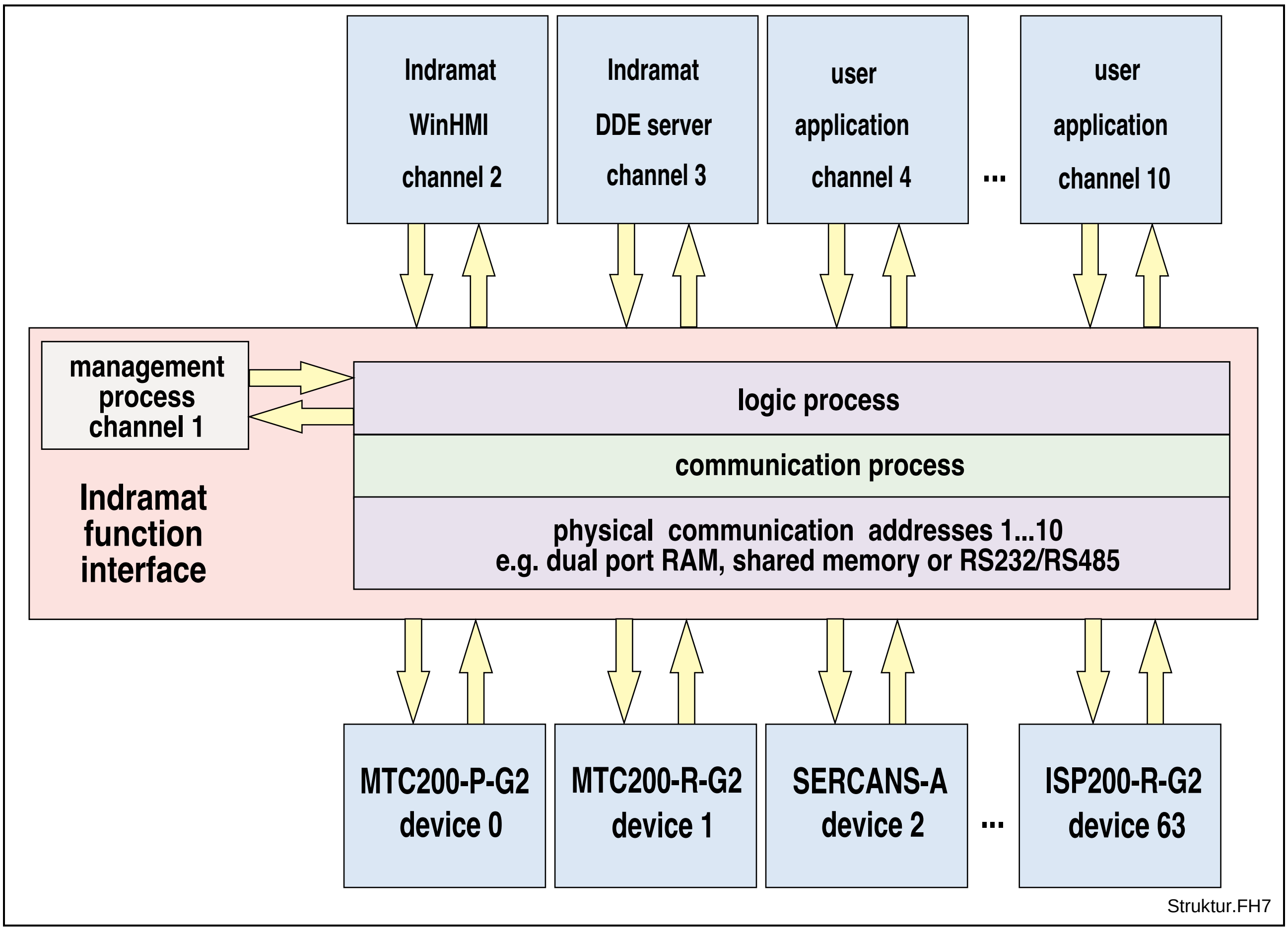


Fig. 3-1: Structural overview of the function interface

Logic process

The logic process provides the user program (client) with the actual data interface along with the services described in the previous chapter. To do this, it opens a logic channel (LOG channel) for every connected client. The number of active LOG channels therefore directly depends on the number of the connected clients. Furthermore, the logic process is a data interface to all defined devices and to the management and status terminal data that are monitored by the management process. As far as the user program (client) is concerned, the logic process is the server. On the other hand, the logic process provides the connection to the communication process via a shared memory. Data is distributed to the individual logic channels via this connection.

Note: The maximum number of LOG channels available to function interface applications is administered dynamically. If a function interface application exceeds this limit then an error message is issued. The chapter entitled “Programming” describes how the data interface to the logic process is to be handled and how to allow data access from the client to the function interface.

Communication process

The communication process executes the requirements of the various logic channels, generates communication to the devices together with the time allocations and initializes all devices configured on starting. The communication process and the logic process thereby allow data access to the respective Rexroth Indramat devices (MTC200-P-G2, ISP200-P-G2, SERCANS etc.).

On the one hand, it exchanges Rexroth Indramat telegrams with the logic process via the shared memory. On the other hand, it exchanges internal telegrams with the configured Rexroth Indramat devices 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 thereby allows simultaneous communication via various communication methods and via several parallel interfaces.

Note: The configuration of the communication addresses as well as the processing options of the individual Rexroth Indramat devices are carried out by the Rexroth Indramat system configurator and stored in the "IND_DEV.INI" file (see Chapter “Directory and File Structure of the Function Interface”).

Management process

The management process is designed as an internal user program and uses the first LOG channel for communication with the logic process. It provides static and dynamic configuration data, delivers the more valuable functions and creates the corresponding data structure for each configured device. The management process thereby collects, for example, MTC200-P-G2 control data together with data from the PC hard drive which a client can then access. The management process thereby fulfills administrative tasks.

Note: The "Function Interface Commands" chapter describes how to access data from the Rexroth Indramat devices and the PC hard drive.

3.2 Configuration Examples and Connection Options

MPI Connection with Profibus FMS

The following figure shows the connection of the Rexroth Indramat MPI (Multi-Protocol-Interface) with Profibus FMS design-type and additional clients to the function interface.

The first LOG channel (logic channel 1) of the function interface is used by a user program (client), e.g. a customized user interface. The Rexroth Indramat MT-CNC user interface (**WinHMI** = **Win**dows **H**uman **M**achine **I**nterface) runs under Windows NT. The MPI connection to the function interface is made via the second LOG channel (logic channel 2).

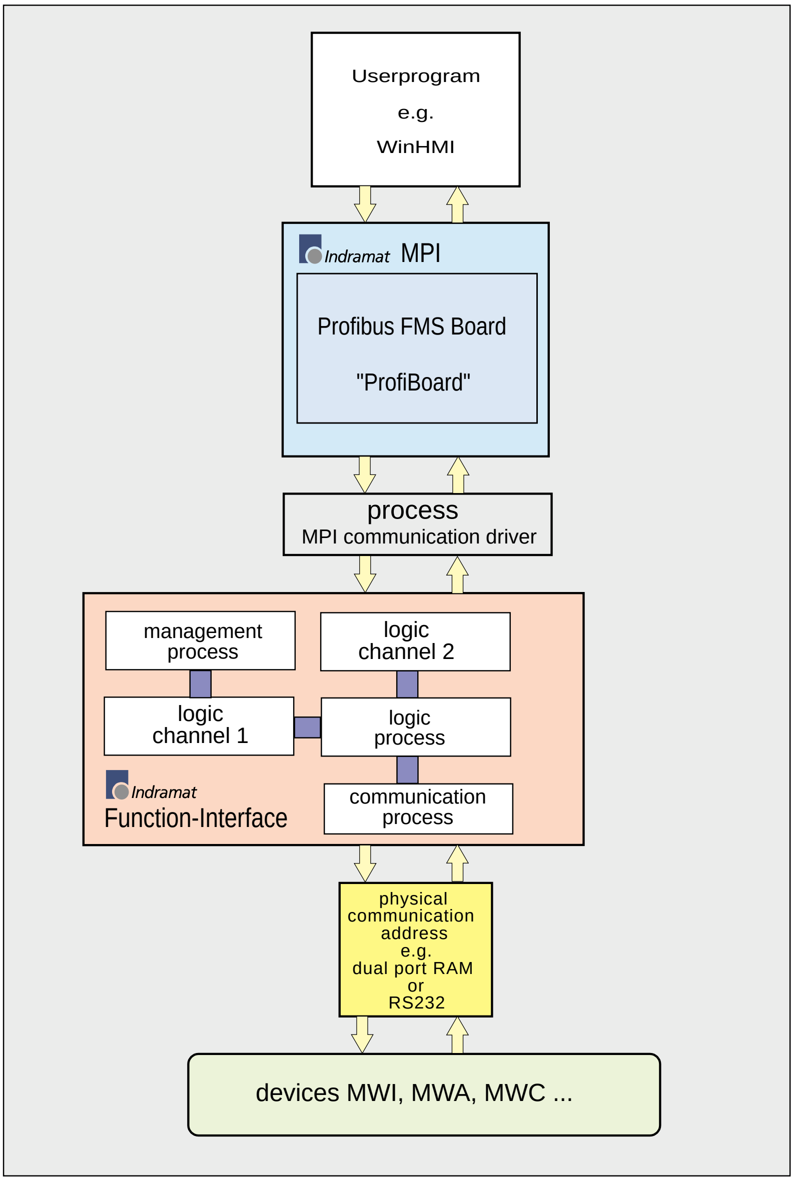


Fig. 3-2: MPI with Profibus FMS connection

Rexroth Indramat GUI and Server

The following figure shows the software structure with the 21VRS Rexroth Indramat GUI (WinHMI) as well as when using the Rexroth Indramat DDE server. It also shows the connection of an OPC server.

The components "WinMTC" and "WinHMI" are component parts of the Rexroth Indramat GUI WIN200. The DDE server allows connection via standard communication mechanisms to external program packages such as WONDERWARE "InTouch". Furthermore, using the NetDDE option, the DDE server allows a connection to be made via a network.

OPC™ stands for **O**LE for **P**rocess **C**ontrol. OLE (**O**bject **L**inking and **E**mbedding) was originally introduced by Microsoft for communication between software components. Today, we refer to the terms COM (**C**omponent **O**bject **M**odel) or DCOM.

The goal of OPC is to create a unified communication interface for process data from any sources such as PLC and NC controls.

The user (developer of OPC client programs) therefore has the following advantages:

- Only minimum knowledge of the controls is required in order to communicate with the control software.
- No adjustment has to be made if an application has to communicate with different makes of control.

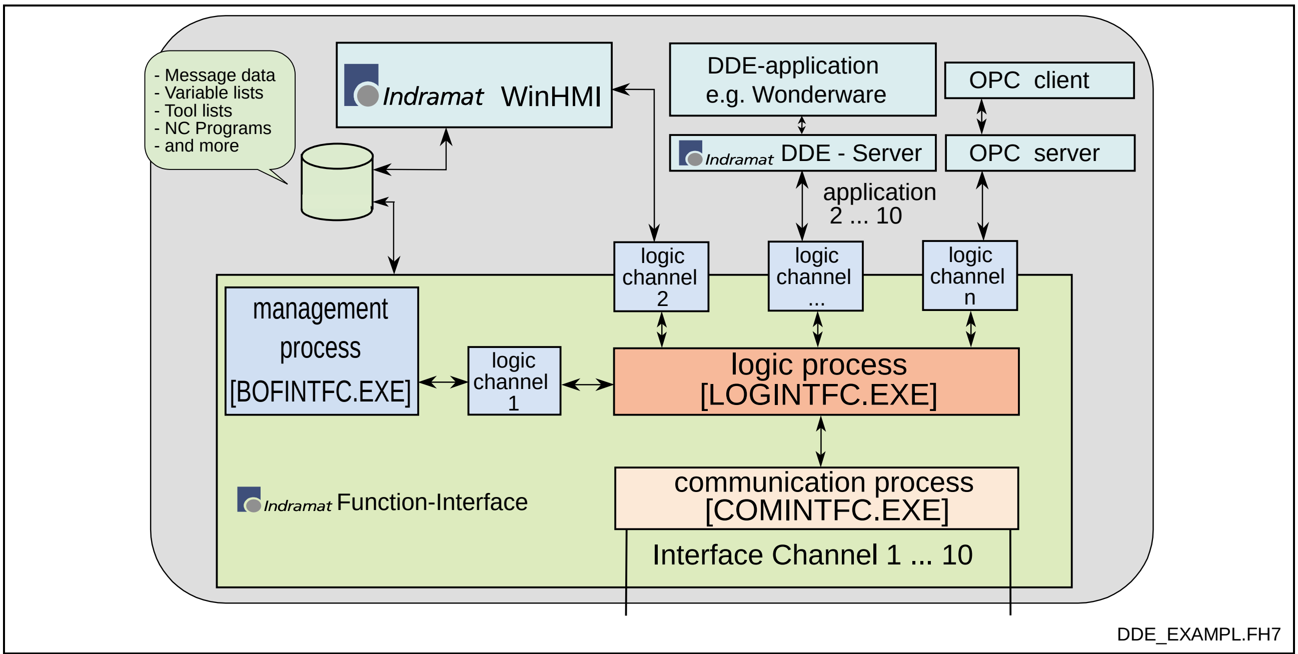


Fig. 3-3: Software structure: Rexroth Indramat user interface and DDE server

Connection to the function interface

The following illustration shows the various options for connecting an application to the function interface.

Direct connection can be achieved via:

- Rexroth Indramat's own GUI, WinHMI.
- programs written by the user in Visual C++ or Visual Basic (customer 3rd party).

The following are examples of indirect connection:

- DDE server,
- OPC server, and
- MPI Com Driver.

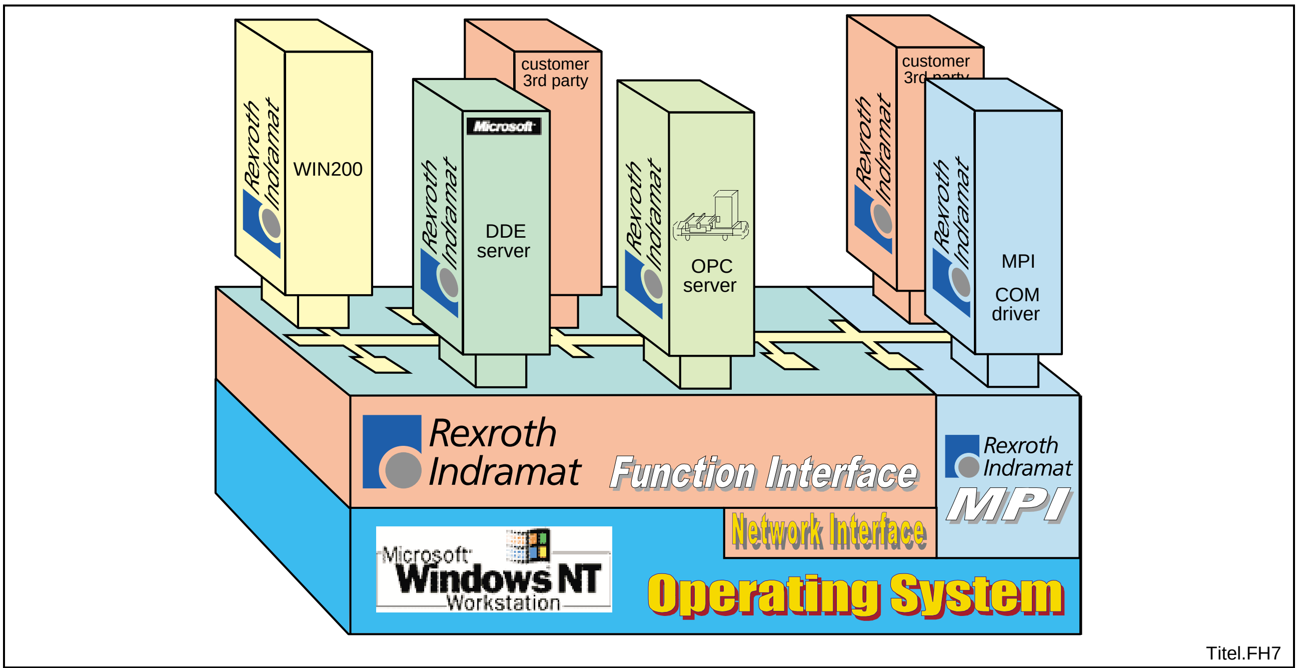


Fig. 3-4: Overview of the connection options

Communication between a Client and Rexroth Indramat Devices

The following figure shows the process of communication of a client on a Rexroth Indramat PC (BTV30) with two Rexroth Indramat devices (MTC200-R-G2 and MTC200-P-G2). On the one hand, the device 00 (MTC200-R-G2) communicates with the communication process via the serial interface (COM1), while device 01 (MTC200-P) communicates via a dual port RAM. The communication process opens a thread for each communication channel that has been configured. The client shown can access data from both devices. To do this, the appropriate device address is specified in the function interface command (FI command) (see Chapter "Design and Availability of the FI Command").

Note: Several cyclic requests (FI commands) can easily be combined at both devices. (See chapter entitled "Data Transfer and Result Evaluation Routines".)

During the initialization phase of the function interface, the configuration data of Rexroth Indramat devices is compared to the actual status. FI commands that have been requested are thereby checked as to their validity for the configured device group. Any errors in command mnemonics can then already be intercepted at the top level.

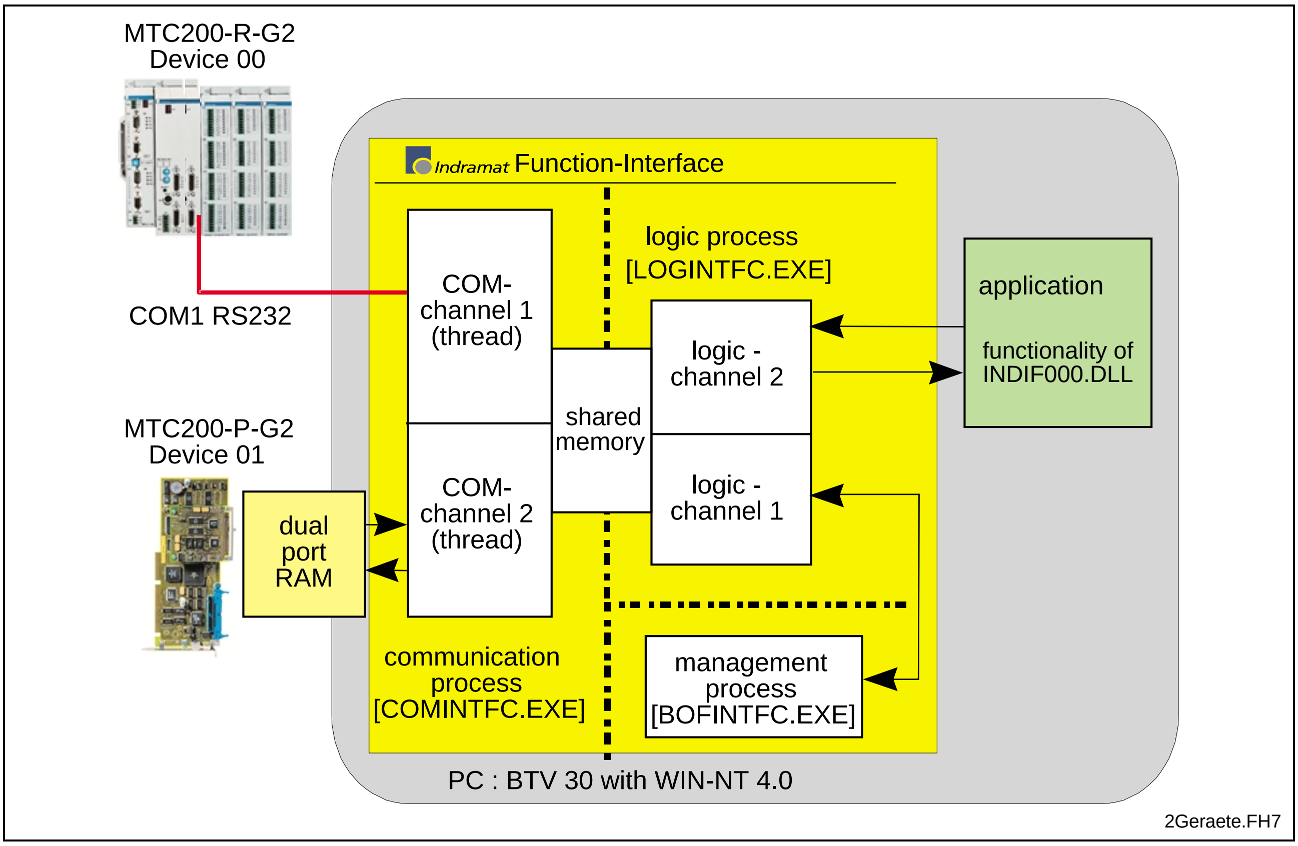


Fig. 3-5: Communication between a Client and Rexroth Indramat Devices

Communication between several clients and Rexroth Indramat devices

The following figure shows the software structure of the function interface when communicating with several devices during the operation of several clients.

Note: Combining the decentralized MTC200-R-G2 with the integrated MTC200-P-G2 is a practical configuration, for example, for a rotary transfer machine.

Here, the function interface allows parallel communication via various interfaces. In the following example, four programs are connected to the function interface in the direction of the clients. Every client can communicate with every device, independently of the other clients. When operating with several devices and several clients, the function interface works like a two-stage, buffered multiplexer. The communication process comprises a multiplexer in the direction of the device and the logic process comprises a multiplexer in the direction of the clients.

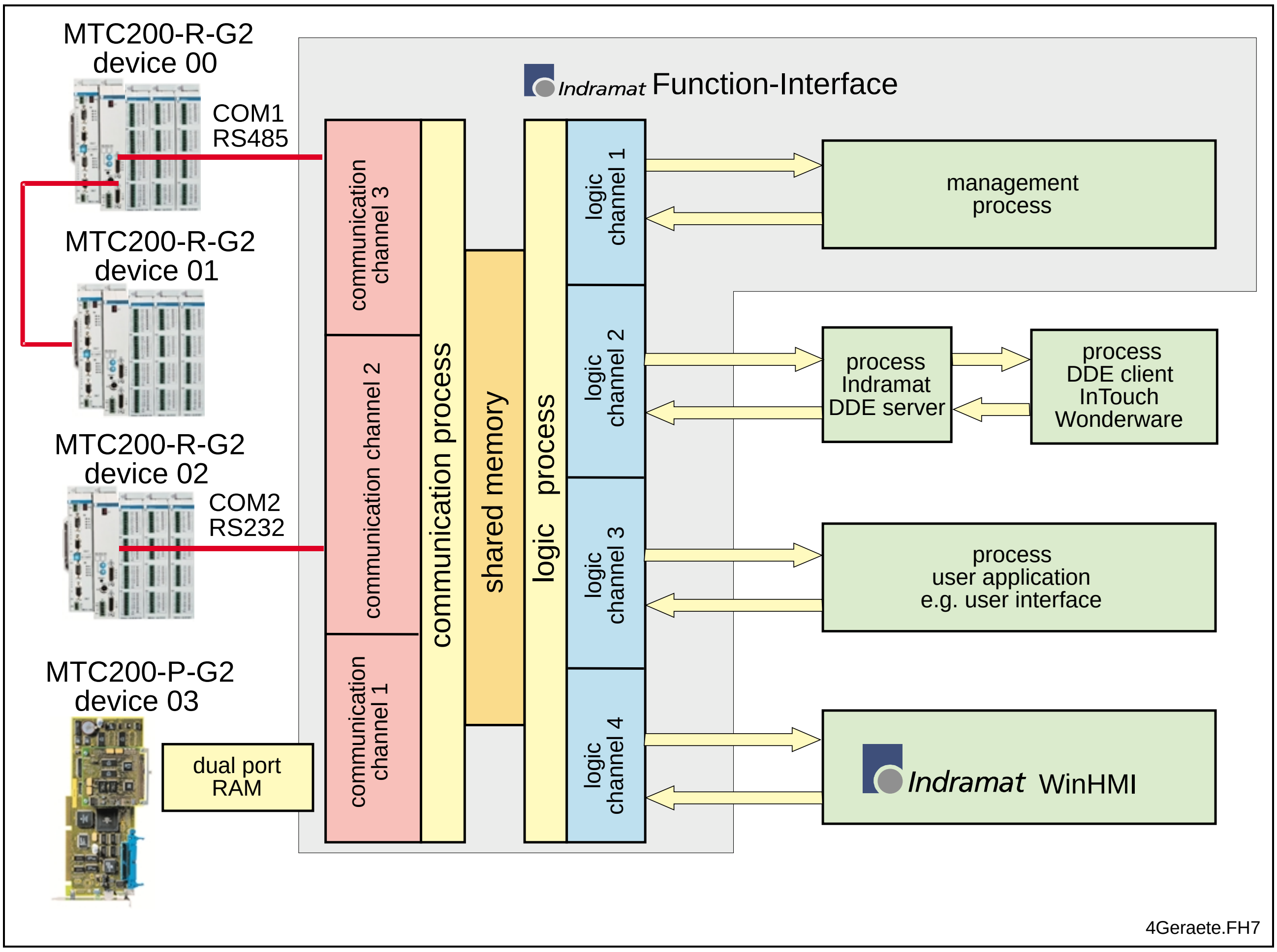


Fig. 3-6: Communication between several Client and Rexroth Indramat Devices

4 Programming

4.1 Guidelines

All user software (clients) that wants to access the function interface must be created in one of the following program languages:

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

The following should be observed when programming:

- the computer should be a Pentium Processor running at a min. of 200 MHz and with a RAM of at least 64 MB.

Note: Parts of the Rexroth Indramat function interface require the highest priority as a Windows NT process.

- Absolute paths should be avoided in the application as any later changes in the drive path (e.g. from C:\ to D:\) or in the directory structure are not supported.

Note: The system directory as well as the Windows NT disk drive can also be freely selected.

The following conditions and statuses of the controls or devices must be considered when programming:

- During a PLC program and/or parameter download from the Rexroth Indramat GUI, other applications must not read or write control data. The system messages (SYS-MSGs) from the call interface are used in evaluating this status. The system messages for the PLC program and/or parameter download are to be considered in the logic of the client.
- Reading and writing of PLC data is limited. Using the FI command “PVS” (see Chapter “Function Interface Commands”), PLC variables with a maximum length of 240 bytes can be read and written. PLC structures and arrays can have a dynamic length. Extremely precise planning is required for communication with the PLC.
- In principle, any PLC variable can be written using the function interface. However, only those PLC variables that are also found in the PLC program should be written in the application.

Note. Write-access to non-declared PLC variables should be avoided.

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

CAUTION

The control system can only be operated safely and correctly with the function interface when the guidelines are observed.

If the guidelines are not observed then all claims against Rexroth Indramat are excluded.

Software for Developing User Programs (Clients) (PRO VERSION)

Note: In Version 07, the function interface cannot be installed separately, but only within the context of the relevant Rexroth Indramat GUI.

Settings for the C++ Development Environment

In order to make the functions of the "INDIF000.DLL" library of the function interface globally available, the following header files:

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

are to be included in the client with the syntax "#include".

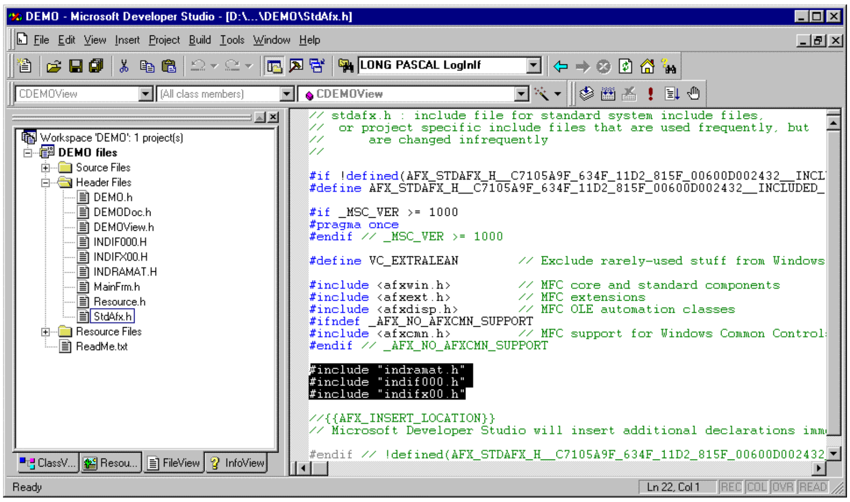
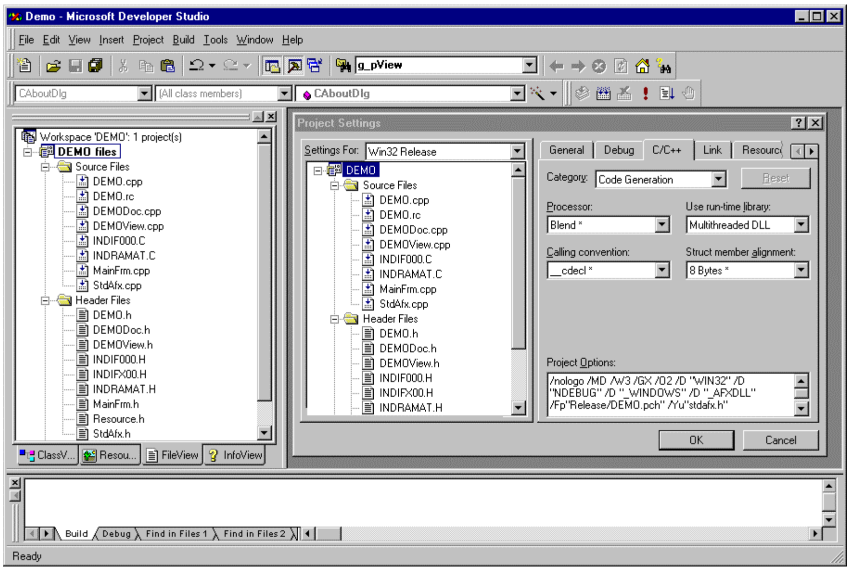


Fig. 4-1: Including the Rexroth Indramat Header Files in the Client

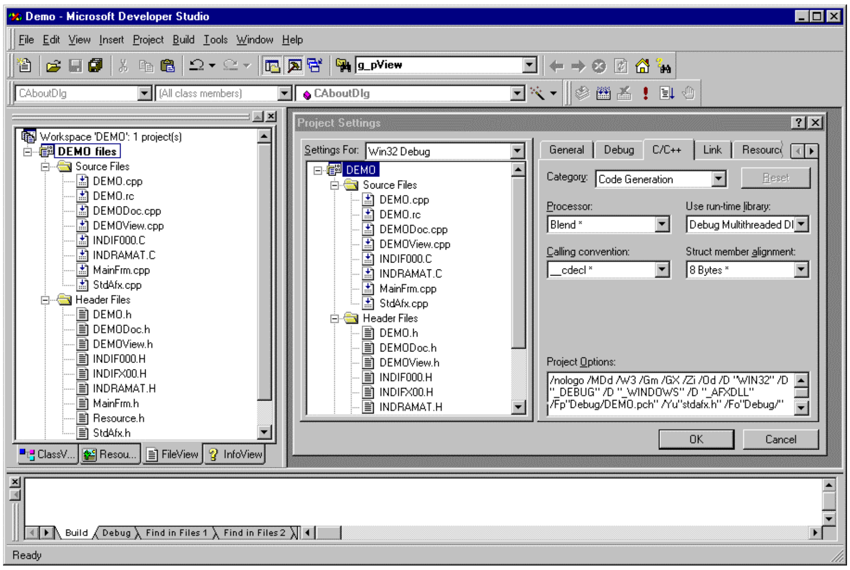
For Visual C++ 5.0, the entry "Multithreaded DLL" should be selected in the "For Win32 Release" project settings in the "C/C++" tab page under the category "Code-Generation" in the "Use run-time library" combo box.



ProjectSettings1.bmp

Fig. 4-2: Project settings "For Win32 Release": Multithreaded DLL

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

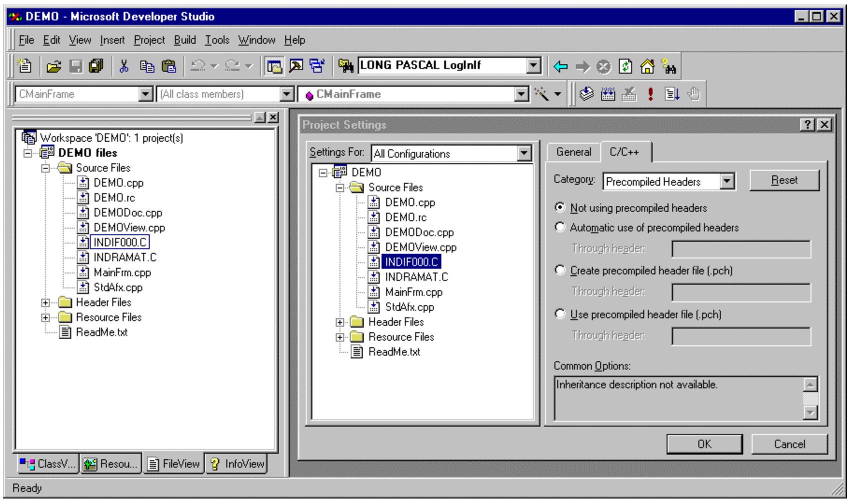


ProjectSettings2.bmp

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

In addition, select the setting "Not using precompiled Headers" in the "For All Configurations" project settings in the "C/C++" tab page under the category "Precompiled Headers" for the following C sources:

- INDIF000.C and
- INDRAMAT.C.



ProjectSettings3.bmp

Fig. 4-4: "For All Configurations" project settings

4.2 Routines for Logging in and Logging Out

Before being able to use the access functions described in the following chapter, the login routine “LogInIf” must always be called up first. Once work with the function interface has been completed, then the logout routine “LogOutIf” should be called.

"LogInIf" Login Routine

Explanation A client connects to the management structure of the function interface via the “LogInIf” routine.

Syntax

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

Pass Parameters

Parameters	**Explanation**
[IN] lpcTaskName	Pointer to the name of the client
[IN] lpcCommandLine	Pointer to the command row for the management and logic process. As a rule, the pass parameters are passed on to the client here. By this means the function interface can be switched to diagnostics mode via the command row of the client.
[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 (normal case)

Parameters	Explanation
[OUT] lhTerminateEvent	HANDLE to the termination event of a process.
[IN] lucIfChannel	Decides 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 (refer to entries in the "INDIFX00.H" file)
[OUT] lhSysMsgEvent	HANDLE on the SYS-Msg-Event.
[OUT] lucTaskId	TaskID, that is assigned to a client on logging in for management reasons [1..MAX_TASK_ANZAHL] (see entries in the "INDIFX00.H" file).
[OUT] ldwIFChannelId	Assigned ID of the Communication Channel [2 to 8]

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried with the "ReadGroupItem" routine in the form of a general error result line. For additional information, refer to the chapter “General Error Result Lines”.

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           'Function 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               'Function-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, IfChannelGrp, 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 equal to 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 = ZERO;
HANDLE ghSysMsgEv = ZERO;
UCHAR gucTaskld = 0;
DWORD gdwlFChannelld       = 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 on  TerminateEvent,
1,                        // Interface channel requested,
10,                       // Max. number of function requests in group,
&ghSysMsgEv,              // HANDLE on SYS-Msg-Event,
&gucTaskld,               // Task-ID,
```

```
&gdwlFChannelld);          // Communication channel - ID

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

"LogOutIf" Log out Routine

Explanation A client logs out from the management structure of the function interface via the “LogOutIf” routine.

Syntax **LONG PASCAL LogOutIf (   );**

Pass Parameters None

Return Values

0: Request successful.

1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

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();
// Error Handling
//******************
if (lRet)
{
     sprintf(acErrMsg,"Function-Interface LogOutIf ErrorCode:
```

4.3 Data Transfer and Result Evaluation Routines

The “DataTransfer” routine is used for reading and writing data to and from Rexroth Indramat devices. The data delivered in the result buffer is structured as follows:

Single Request

Group Element 1	Line 1	Column 1	...	Column j
	:	:	:	:
	Line m	Column 1	...	Column j

Notes: In case of an error, (return value <> 0), the result buffer contains a general error result line that may have to be evaluated in a separate routine (see Chapter 8, “Error Codes”).

As only one command row and **no** group of command rows (also called group request) can be processed via the “DataTransfer” routine, it only has one results group. The data in the result buffer can be evaluated using the "ReadGroupItem" routine.

The “GetNumberOfGroups” returns the number of groups in the result. The “GetNumberOfRows” routine determines the number of lines (rows) for a group, and the “GetNumberOfItems” routine determines the number of columns in the rows.

Group request (ONLY for cyclic requests)

During a group request, several command rows (single requests) are requested simultaneously. The command rows of a group request are separated by spaces. Exactly one group element is delivered in the result for each of these command rows in a group request. The data returned in the result buffer is therefore structured as follows:

Group Element 1	Line 1	Column 1	...	Column j
	:	:	:	:
	Line i	Column 1	...	Column j
:	:	:	:	:
Group Element n	Line 1	Column 1	...	Column j
	:	:	:	:
	Line i	Column 1	...	Column j

Example of a group request

During a group request (BR_NPS... BR_ABN... BR_AGF...), the single group elements can be accessed with *[bGroup]* The meaning of the elements is as follows:

1. Group Element *(bGroup* = 1): BR_NPS...
2. Group Element *(bGroup* = 2): BR_ABN...
3. Group Element *(bGroup* = 3): BR_AGF...

Note: A maximum of 256 command rows (FI commands) can be abstracted as a group request.

"DataTransfer" Routine

Explanation

Data is read or written in accordance to the configured functions using the "DataTransfer" routine (see chapter "Function Interface Commands”).

Syntax

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

Pass Parameters

Parameters	**Explanation**
[IN] pszFunction	Command row
[IN] acValue	Value to be written
[IN] ValLen	Length of value to be written
[IN] ValType	Data code of the value to be written (see chapter "Design and Availability of the F1 Command", Data Code)
[OUT] acResBuf	Result buffer
[IN] lMaxResLen	Length of the result buffer depending on the requested data. The RESULT_BUF_SIZE constant from the INDIF000.h file can be taken as the guide value.
[IN] lResBufType	Data code of results data (see Chapter "Design and Availability of the FI Command", Data code)

Note: The data delivered in the result buffer is coded. To access the single elements, the content of the result buffer must be processed using the "ReadGroupItem" routine:

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried in more detail with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

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 length
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;

// Starting Parameters of the DataTransfer-Routine
//*****************************************************
CHAR * szValue = "";
CHAR * szFunction = "02_CR_CCP4";
CHAR acResultbuf[RESULT_BUF_SIZE];

// Starting Parameters of the GetNumberOfRows-Routine
//************************************************************
LONG lNumOfRows;

// Starting Parameters of the GetNumberOfItems-Routine
//***********************************************************
LONG lNumOfItems;

// Starting Parameters of the ReadGroupItem-Routine
//*******************************************************
LONG lItemLen;
CHAR acItembuf[50];
BOOL boItemValid;

// Access to Function Interface
//***********************************

lRet = DataTransfer (szFunction, // Command row,
szValue,                         // Value,
strlen(szValue),                 // Length of value,
1,                               // Data code of value,
acResultbuf,                     // Result buffer,
RESULT_BUF_SIZE,                 // Length of result buffer,
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);
     }

// Get number of rowa
//*************************
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++)

{

     // Determine Number of Elements
     //******************************
     lRet = GetNumberOfItems(acResultbuf,

     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);
     }

     // Evaluate all Lines
     //**************************

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

     {

        // Evaluate all Results of a Line

       lRet=ReadGroupItem(acResultbuf, // Result data,
       1,                              // Group element,
       i,                              // Row,
       j,                              // Column,
       acItembuf,                      // Individual result,
       50,                             // Length of individual result buffer,
       &lItemLen,                      // Length of result,
       &boItemValid);                  // Valid value ?

       // 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 allows a single result, an entire row or a table of a single or group request to be read out. All results data must be evaluated using this routine.

Syntax

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

Pass Parameters

Parameters	Explanation and Value Range
[IN] CHAR acResBuf[ ]	Buffer for the entire result
[IN] BYTE bGroup	Details of group element [1 to n]
[IN] LONG lRow	-1: Output of a complete table, i.e. all rows of a request [1 to n]: the respective result line
[IN] LONG lItem	-1: Output of a row 0: Output of the requested command with management information [1...n]: Individual 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 *pboItemValid	TRUE: with valid value of the partial result

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Example of "ReadGroupItem" Routine

The following example assumes that a single request (*bGroup* = 1) has been requested:

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

Example of Syntax	Result
ReadGroupItem(acResBuf, **1**, **1**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E1
ReadGroupItem(acResBuf, **1**, **2**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E2
ReadGroupItem(acResBuf, **2**, **2**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E5
ReadGroupItem(acResBuf, **3**, **4**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E9
ReadGroupItem(acResBuf, **2**, **3**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	Error code
ReadGroupItem(acResBuf, **1**, **-1**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E1 E2 E3
ReadGroupItem(acResBuf, **2**, **-1**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	E4 E5
ReadGroupItem(acResBuf, **3**, **-1**, **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**, **0**, **1**, acItemBuf, lLenBuf, &lItemLen, &boItemValid)	supplies e.g.: 001234567800_CC_ ....

Example of Visual Basic/ C++ (see "DataTransfer" routine)

"GetNumberOfGroups" Routine

Explanation The “GetNumberOfGroups” routine returns the number of group elements.

Syntax

LONG PASCAL ReadGroupItem (CHAR *pszValBuf,
LONG *plGroupSize);

Pass Parameters

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

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried in more detail with the "ReadGroupItem" routine in the form of a general error result line. For more detailed information, please refer to the chapter "Error Code".

Example of Visual Basic/ C++ (see "DataTransfer" routine)

"GetNumberOfRows" Routine

Explanation The “GetNumberOfRows” routine determines the number of rows of the indicated group element.

Syntax

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

Pass Parameters

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

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried in more detail with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

Example of Visual Basic/ C++ (see "DataTransfer" routine)

"GetNumberOfItems" Routine

Explanation The “GetNumberOfItems” routine determines the number of partial results, depending on the row number as well as the number of the group element.

Syntax

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

Pass Parameters

Parameters	Explanation
[IN] CHAR *pszValBuf	Buffer for the entire result
[IN] BYTE bGroupIndex	Number of group elements
[IN] BYTE bRowIndex	Row index 0: number of all partial results
[OUT] LONG plNumberOfItems	Number of partial results for a particular row.

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried in more detail with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

Example of Visual Basic/ C++ (see "DataTransfer" routine)

4.4 Routine for Cyclical Reading via Pipes

The pipe access functions are used for cyclical reading of device data via the function interface. Several command rows can be passed simultaneously via a group request. The command rows of a group request are separated by a space (refer here also to the "ReadGroupItem" routine).

Note: A maximum of 256 command rows (FI commands) can be abstracted as a group request.

A pipe is started by the “StartCyclicPipe” routine and then provides itself continually with updated data. Asynchronous to this, access to this data is now made via the “ReadCyclicPipe” routine. The cyclical request is stopped by the "StopCyclicPipe" routine.

"StartCyclicPipe" Routine

Explanation The “StartCyclicPipe” routine starts a pipe for cyclical reading of the data.

Syntax

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

Pass Parameters

Parameters	Explanation
[IN] wPipe	Pipe number [1...1000]
[IN] *pszFunktion	Group of strings according to the defined function requests.
[IN] lBufSize	Size of result buffer [Byte]
[IN] lGroupSize	Number of group elements [1 to n]
[IN] dwSleep	Read delay time [ms]

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

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];

// Starting 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,
RESULT_BUF_SIZE,                // Size result buffer,
2,                              // Number group elements,
500);                           // Reading delay time [ms]

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

"ReadCyclicPipe" Routine

Explanation The “ReadCyclicPipe” routine reads the data of a pipe that has been started by “StartCyclicPipe”.

Syntax

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

Pass Parameters

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

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: The pass parameter [OUT] BYTE *pbGroupFault contains the number of the faulty group element. Further information on the meaning of the"ReadCyclicPipe" routine error code can be requested in the form of a general error result line using the “ReadGroupItem” routine. For additional information. refer to the chapter “General Error Result Lines”.

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[RESULT_BUF_SIZE];
UCHAR bIndexItemFault;
LONG lAttr;
// Input parameters of the GetNumberOfGroups routine
//***************************************************************
LONG lNumOfGroups;
// Read pipe
//*************
lRet = ReadCyclicPipe(wPipeNo,          // Pipe number,
acResultbuf,                            // Result buffer,
RESULT_BUF_SIZE,                        // Length result buffer,
&bIndexItemFault,                       // Index of the group
                                        // element with 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);
}
// Evaluation of results
//*************************
for (i=1; i<=lNumOfGroups; i++)
{
     // Results evaluation for each group result
     // e.g. LONG lItemLen;
     //       CHAR acItembuf[50];
     //       int iItemValid;
     //
     lRet=ReadGroupItem(acResultbuf,    // Result buffer,
     i,                           // Group element,
     1,                           // Line,
     1,                           // Element,
     acItembuf,                   // Individual result buffer,
     50,                          // Length of the individual result buffer,
     &lItemLen,                   // Length of the individual result,
     &iItemValid);                // Individual 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 that has been started by “StartCyclicPipe”.

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

Pass Parameters

Parameters	**Explanation**
[IN] WORD wPipe	Pipe number

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

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 stopped

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  ErrorCode:%ld ",lRet);
        MessageBox (GetFocus(),acErrMsg,"Function Interface Error", MB_OK);
        }
```

"SuspendCyclicPipe" Routine

Explanation

The “SuspendCyclicPipe” routine sets the data request of a pipe that has been started by “StartCyclicPipe” into standby mode. It is used to stop communication while at the same time maintaining the management structure of the function interface established by the "StartCyclicPipe" routine (see "ResumeCyclicPipe" routine).

Syntax

LONG PASCAL SuspendCyclicPipe (WORD wPipe);

Pass Parameters

Parameters	**Explanation**
[IN] WORD wPipe	Number of the pipe

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

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 that has been set to standby mode by the “SuspendCyclicPipe” routine.

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

Pass Parameters

Parameters	Explanation
[IN] WORD wPipe	Number of the pipe

Return Values

0: Request successful.
1 …n: Request unsuccessful (see chapter “Error Codes”).

Note: Additionally, an error can be queried with the "ReadGroupItem" routine in the form of a general error result line. For additional information. refer to the chapter “General Error Result Lines”.

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);
     }
```

4.5 System messages in the Network

SYS-Messages are used to report particular events in the system to the applications. Generally speaking the application is synchronized to the changed control system data.

Examples of this are PLC program download, parameter download, system error.

Example

PLC program download

Before the PLC GUI downloads a PLC program the SysMessage MSG_PCLUPDBEG is reported.

The PLC GUI reports the end of the download with the SYS-Message MSG_PCLUPDEND.

After that, the applications will re-synchronize access to the PLC data.

PLC user interface	SYS Message	Application status
Translate program		PLC available (1)
	→ MSG_PCLUPDBEG →	PLC available
		PLC blocked
	← MSG_PCLUPDBEG_Q ←	PLC blocked
Download		PLC blocked
	→ MSG_PCLUPDEND →	PLC blocked
		PLC blocked
	← MSG_PCLUPDEND_Q ←	PLC available
!! Download ended!!		PLC available

Fig. 4-5: Synchronization via SYS-Messages

- Transparent across the network.
- Messages logged in to a particular device.
- Multiple logins in one application to support different process packages running in a process.
- Dynamically expandable message type for messages relating to the application such as for updating global network data.
- Failure of the network connection: The failure of the connection to a device is acknowledged for each requested message.

Data Types

TyVMsgCallback

The following callback has been defined for receiving network messages:

Declaration

```
LONG          VOID              *pParam,
(TyVMsgCall   LONG              lMsgId,
back*)(       CONST CHAR        *psczDevice
              CONST CHAR        *pcData
              LONG              lDataLen);
```

Parameters

lMsgId Numerical identification of the SYS-Message.

Gives the system area in which the control component messages are written.

Gives the user area in which the applications can be assigned message IDs.

psczDevice The device for which a message is generated or received.

pcData User data of a message.

Programming interface

Constants

SYSMSG_ALLPC	Login for all PCs in the network
SYSMSG_ALLDEVICE	Login for all devices
SYSMSG_LOCALPC	Login for all devices of the local PC

Flags for SysMsgHookCreate

SYSMSG_MANUAL_ACK	Manual acknowledgement

This flag must be set if the acknowledgement of the system message is not to be carried out automatically after the callback function.

!! In this case, the application must call up

SYSMSG_LOGOUT_CALLBACK even when logging out.

TySysMsgCallbackInfo Data Type

This data type is used to provide the callback function with data from the system message.

```
typedef struct {
  VOID       *pUserParam;
  LONG        lDevice;
  CONST CHAR *pcData;
  LONG        lDataLen;
  LONG        lStatus;
  LONG        lHookId;
  LONG        lMsgId;
  LONG        lPc;
  LONG        lFarDevice;
} TySysMsgCallbackInfo;
```

Name	Description
pUserParam	User-defined parameter. Refer also to SysMsgHookCreate
lDevice	The local device address of the system message, or SYSMSG_ALLDEVICE, if the system message is not issued specifically for the device.
pcData	Message data
ldataLen	Length of data
lStatus	Callback status 0: OK Otherwise, logout or error
lHookId	ID of the connected hook
lMsgId	The message ID
lPc	The PC from which the system message originates. SYSMSG_LOCALPC, if the system message has been requested from the local PC.
lFarDevice	The device address in the network. If the network has not been activated then the local device address is accepted. SYSMSG_ALLDEVICE, if the system message has not been issued for a specific device.

TySysMsgCallback Data Type

```
typedef VOID ( __stdcall
*TySysMsgCallback)(TySysMsgCallbackInfo *pCbInfo);
```

The function pointer data type for the callback.

SysMsgHookCreate

The callback function is called up in a try catch block.

The callback is activated in a separate thread.

Note: No further system messages are handled while the function is active.

The transmitted data is generated on the stack and becomes invalid at the end of the return of the function.

Declaration	`LONG SysMsgHookCreate(`	`LONG *plHookId,`
		`LONG lDevice,`
		`LONG lPC,`
		`LONG lMsgId,`
		`TySysMsgCallback CallbackFunc,`
		`VOID *pParam,`
		`LONG lFlags);`

Parameters:	plHookId	Output Parameter: Handle on the hook This value is needed to deactivate the callback or to make a manual acknowledgement. Refer also to SysMsgHookDelete() and SysMsgHookAcknowledge()
	lDevice	Device number for which this callback is to be activated. A valid local or Far Device number. To log on the system message for all devices on a PC, SYSMSG_ALLDEVICE can also be transferred. In this case lPC defines the PC for which the callback is activated.
	lPC	Defines the PC for which the callback is to be activated. !! This parameter is only effective if lDevice is transferred with SYSMSG_ALLDEVICE. SysMsgHookCreateAll() must be used to activate the callback for all devices of all PCs in a network.

Declaration	`LONG SysMsgHookCreate(`	`LONG`
		`LONG`
		`LONG`
		`LONG`
		`TySysMsgCallback`
		`VOID`
		`LONG`
			`*plHookId,`
			`lDevice,`
			`lPC,`
			`lMsgId,`
			`CallbackFunc,`
			`*pParam,`
			`lFlags);`

	lMsgId	The message ID	
	CallbackFunc	Callback user function. !! This function is activated in a separate thread. !! No further system messages are handled while the function is active. The transmitted data is generated on the stack and becomes invalid at the end of the return of the function.	
	pParam	User Parameter for the callback function.	
	lFlags	The value of this parameter switches on certain hook options. The options can be abstracted by means of a logical OR operation '|'. The system message is not acknowledged automatically when the callback function is exited. The application must acknowledge receipt of the system message by means of the SysMsgHookAcknowledge() function. SYSMSG_LOGOUT_CALLBACK: On LogoutIF(), the callback is activated one last time. If necessary, the application can delete user data allocated to the hook.	
Return value:	0	OK	
	NET_ETIMEOUT:	Time for making a connection to a remote device has been exceeded.	
	NET_EINVPARAM	Invalid parameter	
	NET_EFALSE	LoginIf must be called up before system messages can be logged in.	

SysMsgHookAcknowledge

Acknowledgement of a system message in the network.

Declaration	`LONG SysMsgHookAcknowledge`	`(LONG lHookId);`
Parameters:	lHookId	HookId of the system message that is to be acknowledged.
Return value:	0	OK
	NET_EINVPARAM	Invalid parameter
	NET_EFALSE	LoginIf must be called up before this function can be called up.

SysMsgHookDelete

Deletion of a system message in the network.

Declaration	`LONG SysMsgHookDelete`	`(LONG lHookId);`
Parameters:	lHookId	HookId of the system message that is to be deleted.
Return0value:	0	OK
	NET_EINVPARAM	Invalid parameter
	NET_EFALSE	LoginIf must be called up before this function can be called up.

Rexroth Indramat

SysMsgHookCreateAll

Generates multiple hooks for system messages. A hook is generated for each PC in the network and is activated for each device.

A hook ID is returned for each PC. If plHookId is 0, the hook IDs are not returned.

Declaration	`LONG SysMsgHookCreateAll`	`(LONG *plHookId,`
		`LONG lHooktabMax,`
		`LONG *plHooktabCnt,`
		`LONG lMsgId,`
		`TySysMsgCallback CallbackFunc,`
		`VOID *pParam,`
		`LONG lFlags);`

Parameters:	plHookId	Output parameter: handles on the hooks to the various PCs These values are needed to deactivate the callback or to make a manual acknowledgement. Refer also to SysMsgHookDelete() and SysMsgHookAcknowledge()
	lHookTabMax	Size of the hook table the user is administering. → The function generates as many hooks as there are PCs declared in the network.
	plHooktabCnt	Transfer of number of hooks generated.
	lMsgId	The message ID
	CallbackFunc	Callback user function. !! This function is activated in a separate thread. !! No further system messages are handled while the function is active. The transmitted data is generated on the stack and becomes invalid at the end of the return of the function.
	pParam	User Parameter for the callback function.
	lFlags	The value of this parameter switches on certain hook options. The options can be OR operated '|'. The system message is not acknowledged automatically when the callback function is exited. The application must acknowledge receipt of the system message by means of the SysMsgHookAcknowledge() function. SYSMSG_LOGOUT_CALLBACK: On LogoutIF(), the callback is activated one last time. If necessary, the application can delete user data allocated to the hook.
Return value:	0	OK
	NET_ETIMEOUT:	Time for making a connection to a remote device has been exceeded.
	NET_EINVPARAM	Invalid parameter
	NET_EFALSE	LoginIf must be called up before system messages can be logged in.

Example of Programming

The following example describes an application of the system message mechanism. A class is declared that works with the functions described:

Declaration

```
#include "indif000.h"   // Declaration of the FI-Routines

//trigger up to maximum of 100 PCs in the PC network(more
than //100 are currently not permitted)
#define D_nSYSMSG_MAX_PC      100

//e.g., react to 28 messages
#define D_nSysMSG_COUNT       28


// Data structure to management (s.u.)
typedef struct _TyHookTable
{
  LONG  alHookId[D_nSYSMSG_MAX_PC];
  LONG  lCount;
} TyHookTable;

// Class declaration
/*##{class}############################################

------------------------------------------------------
  Description:
Sample Class for FI System message Handling

------------------------------------------------------
History:
#############################################{end}##*/
class KSysMsgSample
{
  //*
  //* Construction
  //*
  public:
       KSysMsgSample();
       ~KSysMsgSample();

// ...



  //*
  //* Attributes
  //*
  private:
       // Callback Routine
       static void __stdcall
       SysMsgCallback(TySysMsgCallbackInfo* pCbInfo);
```

```
        //Help Routines
        LONG HookSysMsg (void);
        LONG UnhookSysMsg (void);

        // Data structures to SysMsg-Handling
        static LONG s_alMsgList[D_nSysMSG_COUNT];
        static LONG s_lMsgCount;

TyHookTable  c_aoHookTable[D_nSysMSG_COUNT];

};
```

Implementation

```
// List of SYS messages to be handled (example)

LONG KSysMsgSample::s_alMsgList[] =
{
  MSG_PARUPDBEG,
  MSG_PCLUPDBEG,
  MSG_LAGCHABEG,
  MSG_MECERRGEN,
  MSG_SYSERRGEN,
  MSG_FWAUPDBEG,
  MSG_MEMUPDBEG,
  MSG_ACTERRBEG,
  MSG_DEVERRBEG,
  MSG_MDLERRBEG,

  MSG_PARUPDEND,
  MSG_PCLUPDEND,
  MSG_LAGCHAEND,
  MSG_MECERRDEL,
  MSG_SYSERRDEL,
  MSG_FWAUPDEND,
  MSG_MEMUPDEND,
  MSG_ACTERREND,
  MSG_DEVERREND,
  MSG_MDLERREND,

  MSG_STRTUPCHG,
  MSG_WARNINCHG,
  MSG_SETUP_CHG,
  MSG_MESSAGCHG,
  MSG_ERROR_CHG,
  MSG_SFCERRCHG,
  MSG_SFCMODCHG,
  MSG_DMPSELCHG
};

// Login of system messages
LONG KSysMsgSample::HookSysMsg (void)
{
  LONG lResult = 0;
```

```
  for (INT i=0; i< D_nSysMSG_COUNT; ++i)
  {
    // Login SysMessages for all PCs in network.
    lResult = :: SysMsgHookCreateAll(
       c_aoHookTable[i].alHookId,
       D_nSYSMSG_MAX_PC,
       &(c_aoHookTable[i].lCount),
      s_alMsgList[i],
      SysMsgCallback,
      static_cast<VOID*>(this),
      0);
  }
  return lResult;
}

// Logout of system messages
LONG KSysMsgSample::UnhookSysMsg (void)
{
  for (INT i=0; i< D_nSysMSG_COUNT; ++i)
  {
    for (INT j=0; j<c_aoHookTable[i].lCount; j++)
    {
      ::SysMsgHookDelete(
       c_aoHookTable[i].alHookId[j]);
    }
  }

  return 0;
}


// Callback function
void KSysMsgSample::SysMsgCallback(
  TySysMsgCallbackInfo* pCbInfo)
{
  KSysMsgSample * pInstance =
  static_cast<KSysMsgSample*>(pCbInfo->pUserParam);

  switch (pCbInfo->lMsgId)
  {
      case MSG_PARUPDBEG:
       case MSG_PCLUPDBEG:
       case MSG_FWAUPDBEG:
       // handle begin download
         break;

      case MSG_PCLUPDEND:
       case MSG_PARUPDEND:
       case MSG_FWAUPDEND:
         // handle end download
         break;
```

```
        //etc.
    }
}
```

What SYS Messages are there and how should I react to them?

The SYS messages that are most often used for a client 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 a SYS-MSG acknowledgement.

Note: The device address that has set the system message is returned as an ASCII character in the buffer "CHAR *pcSysMsgBuffer" of "GetIfMsg". The buffer may possibly contain additional SYS-MSG information such as the parameter identification string.

SYS Message	What happens?	Reaction from the Client
MSG_FWAUPDBEG	The firmware download commences, i.e. the firmware is loaded into the System200 components by the GUI.	Communication between all configured Indramat devices is interrupted.
MSG_FWAUPDEND	End of firmware download.	Communication recommences. PLEASE NOTE: Process and axis configuration data etc., may have changed.
MSG_PARUPDBEG	The parameter download commences, i.e. the parameter set is loaded into the System200 components by the user interface.	Communication between all configured Indramat devices is interrupted.
MSG_PARUPDEND	End of parameter download	Resumption of communicationPLEASE NOTE: Process and axis configuration data etc., may have changed.
MSG_PCLUPDBEG	The PLC program download commences, i.e. the PLC program is loaded into the System200 components by the GUI.	Communication between all configured Rexroth Indramat devices is interrupted.
MSG_PCLUPDEND	End of PLC program download	Communication recommences.
MSG_MEMUPDBEG	GUI begins to delete the data memory in the System200 components.	Communication between all configured Indramat devices is interrupted.
MSG_MEMUPDEND	GUI has deleted the data memory in the System200 components.	Communication recommences. PLEASE NOTE:Configuration data has been deleted.
MSG_SYSERRGEN	If there is a system error, this SYS-MSG is issued, i.e. the PLC cannot be accessed at the moment.	**No** interruption of communication to the Rexroth Indramat devices is necessary. (Used for presenting a system error from a particular Rexroth Indramat device in graphic form).
MSG_SYSERRDEL	A system error is deleted.	**No** interruption of communication to the Rexroth Indramat devices is necessary. (Information that the system error is no longer present at a particular Rexroth Indramat device.)

MSG_MECERRGEN MSG_MECERRDEL	This SYS-MSG is issued if there is a fault in the mechanism. A mechanism error is deleted.	**No** interruption of communication to the Rexroth Indramat devices is necessary. (Is used for presenting a system error from a particular Rexroth Indramat device in graphic form). **No** interruption of communication to the Rexroth Indramat devices is necessary. (Information that the system error is no longer present at a particular Rexroth Indramat device.)
MSG_LAGCHABEG MSG_LAGCHAEND	A language switch has been initialized at the Rexroth Indramat GUI. A language switch has been completed at the Rexroth Indramat GUI.	**No** interruption of communication to the Rexroth Indramat devices is necessary. (Information that the user interface language is being switched.) **No** interruption of communication to the Rexroth Indramat devices is necessary. (Information, that the user interface language has been switched.)
MSG_PC__ALIVE	A PC/device logs in/out of the PC network. The device address/FarDevice address that has set the system message is returned as an ASCII character in the buffer "CHAR *pcSysMsgBuffer" of the "GetIfMsg". The buffer also contains the information "0" for PC logged out or a "1" for PC logged in.	Depending on the application, this system message is used on the one hand purely as information regarding the logging in/logging out of PCs. On the other hand, communication to the PC/device or the re-establishment of communication with subsequent synchronization is required.

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
```

4.6 COM - Automation Interface

The function interface supports two simple COM automation interfaces:
IFIObject, and
IFIData.

IFIObject

Explanation IFIObject represents the following functions of a function interface (FI)
LogInIf,
LogOutIf, and
DataTransfer.

IFIObject::LogInIf

IDL description HRESULT LogInIf([in] BSTR bstrTaskName)

Explanation Compared tot he FI function, this method has been greatly simplified. For a more detailed description, refer to the "LogInIf" FI function.

Parameters bstrTaskName: Name of the user program

Example

Visual Basic Script

```
Dim oFIObject

On Error Resume Next
Set oFIObject = CreateObject("Indif000.FIObject")
if Not oFIObject is Nothing then
  oFIObject.LogInIf("fi.vbs")

end if

Set oFIObject = Nothing
```

Example

JavaScript

```
var oFIObject;
try{
  oFIObject = new ActiveXObject("Indif000.FIObject");
  if (oFIObject == null)
  {
    return;
  }

  oFIObject.LogInIf("fi.js");
  oFIObject = null;
}
catch(e)
{
}
```

IFIObject::LogOutIf

IDL description HRESULT LogOutIf();

Explanation Here, refer to the "LogOutIf" FI function description.

IFIObject::DataTransfer

IDL description HRESULT DataTransfer([in] BSTR bstrFunction,
[in, defaultvalue(32000)] long lResSize,
[in, defaultvalue(3)] long lResType,
[in, defaultvalue("")] BSTR bstrValue,
[in, defaultvalue(3)] long lValType,
[out, retval] IFIData **ppoData);

Explanation Compared to the "DataTransfer" FI function, the IFIObject DataTransfer was made easier to operate in view of the parameters. The sequence of the parameters has been changed, and some parameters have been pre-assigned default values.

Parameters

bstrFunction: Function interface command

lResSize: Length of the result buffer, default 32000 bytes

lResType: Data code of result data, default 3 (ANSI)

bstrValue: Writing value, default empty string

lValType: Data code of the value to be written, default 3 (ANSI)ppoData: A DataTransfer request results in an IFIData object. In case of error, a zero object is returned.

Example

Visual Basic Script

```
Dim oFIData
Set oFIData = oFIObject.DataTransfer("00_CR_PVF_bool0")
if Not oFIData is Nothing then
end if
```

The following source code corresponds exactly to the above example

```
Dim oFIData
Set
oFIData=oFIObject.DataTransfer("00_CR_PVF_bool0",32000,3,"",
3)
if Not oFIData is Nothing then
end if
```

Example

JavaScript

```
var oFIData;
oFIData = oFIObject.DataTransfer("00_CR_PVF_bool0");
if (oFIData == null)
{
  return;
}
```

The following source code corresponds exactly to the above example

```
var oFIData;
oFIData = oFIObject.DataTransfer("00_CR_PVF_bool0", 32000,
3, "", 3);
if (oFIData == null)
{
  return;
}
```

IFIData

Explanation IFIObject represents the following functions of a function interface (FI)
GetNumberOfRows,
GetNumberOfItems, and
ReadGroupItem.

IFIData::GetNumberOfRows

IDL description HRESULT GetNumberOfRows([out, retval] long *plRows)

Explanation Here, refer to the "GetNumberOfRows" FI function description.

Parameters pRows: Number of rows of an FI result.

Example

Visual Basic Script
Dim lRow
lRow = oFIData.GetNumberOfRows()

Example

JavaScript
var lRow;
lRow = oFIData.GetNumberOfRows()

IFIData::GetNumberOfItems

IDL description HRESULT GetNumberOfItems([in] long lRow, [out, retval] long *plItem)

Explanation Here, refer to the "GetNumberOItems" FI function description.

Parameters lRow: [1, 256]
plItem: Number of partial results

Example

Visual Basic Script
Dim lColumn
lColumn = oFIData.GetNumberOfItems(1)

Example

JavaScript

var lColumn;

lColumn = oFIData.GetNumberOfItems(1);

IFIData::ReadGroupItem

IDL description HRESULT ReadGroupItem([in] long lRow, [in] long lItem, [in, defaultvalue(32000)] long lBufSize, [out, retval] BSTR *pbstrVal)

Explanation Here, refer to the "ReadGroupItem" FI function description. The only difference is the default value assignment of the lBufSize parameter.

Parameters

lRow:	-1:	Output of a complete table, i.e. all rows of a request
	[1...256]:	the respective result line
lItem:	-1:	Output of a row
	0:	Output of the requested command with management information
	[1...n]:	Individual result (Element of a row)
lBufSize:		Length of the buffer for partial result, default 32000 bytes.
pbstrVal:		Requested partial result

Example

Visual Basic Script

Dim strText

strText = oFIData.ReadGroupItem(-1, -1)

Example

JavaScript

var strText;

strText = oFIData.ReadGroupItem(-1, -1);

Example for the Total Script

Visual Basic Script

```
Dim oFIObject
Dim oFIData
Dim lRow
Dim lColumn
Dim strText

On Error Resume Next
Set oFIObject = CreateObject("Indif000.FIObject")
if Not oFIObject is Nothing then

    oFIObject.LogInIf("fi.vbs")
    if Err = 0 then

        Set oFIData =
oFIObject.DataTransfer("00_CR_PVF_bool0")
        if Not oFIData is Nothing then
```

```
            lRow = oFIData.GetNumberOfRows()
            MsgBox "Rows: " & CStr(lRow)
            lColumn = oFIData.GetNumberOfItems(1)
            MsgBox "Columns: " & CStr(lColumn)
            MsgBox oFIData.ReadGroupItem(-1, -1)
        end if
        call oFIObject.LogOutIf()
    end if
end if

Set oFIObject = Nothing
```

JavaScript

```
var oFIObject;
var oFIData;
var lRow;
var lColumn;
var strText;

function FI()
{
  try{
    oFIObject = new ActiveXObject("Indif000.FIObject");
    if (oFIObject == null)
    {
      return;
    }

    oFIObject.LogInIf("fi.js");
    oFIData = oFIObject.DataTransfer("00_CR_PVF_bool0");
    if (oFIData == null)
{
  return;
}
    lRow = oFIData.GetNumberOfRows();
    strText = "Rows: " + lRow.toString();
    WScript.Echo ( strText );
    lColumn = oFIData.GetNumberOfItems(1);
    strText = "Columns: " + lColumn.toString();
    WScript.Echo ( strText );
    WScript.Echo (oFIData.ReadGroupItem(-1, -1));
    oFIData = null;
    oFIObject.LogOutIf();
    oFIObject = null;
  }
  catch(e)
{
}
}

FI();
```

4.7 Tips and Tricks when Working with the Interface

This chapter provides you with tips and tricks that are designed to help you to proceed faster when developing your user program (client) (see also Chapter “Installing Windows NT and the Function Interface”).

Furthermore, we have discovered that using Windows NT without a mouse always tends to present difficulties and we have therefore listed the most important Windows NT key combinations in a table.

Problem	**Remedy**
In your application, you issue an FI command and receive: - no response - **or** - - an unexpected response **- or -** - an error code (see Chapter "Error Codes")	Frequent cause:Device address has not been given **or** been incorrectly given! Check the correct details of the FI command (see Chapter “Installing Windows NT and the Function Interface” and Chapter “Function Interface Commands”). Issue the FI command that is causing problems using the VBDemo program (see Issuing FI Commands with the VBDemo Application).
Your client no longer reacts	See clearing the Memory using the "KILLTASK.EXE" Tool
Your client terminates "DR. WATSON" with a Windows access violation.	See clearing the Memory using the "KILLTASK.EXE" Tool Correct the programming error and re-start your application.
The entire system (Windows NT, client and Rexroth Indramat GUI) is reacting slowly.	Check the Windows NT settings for improved performance, idling activity, swapping the core-mode driver and idling activity in accordance with Chapter “Installing Windows NT and the Function Interface”.
Clearing the memory using the "KILLTASK.EXE" tool doesn’t work, i.e. neither the three base processes of the function interface nor the client is removed from memory.	Start Task manager, for example, using the key combination <Ctrl>+<Shift>+<Esc> (see chapter “Windows NT Task Manager”). Click on the ”Processes” tab page. Terminate the three basic processes of the function interface and your user program, if applicable: LOGINTFC.EXE (logic process) COMINTFC.EXE (communication process) BOFINTFC.EXE (management process) via the <Terminate process> button.
Your application terminates because: - required files are missing - **or** - - path entries do not exist or are incorrect.	Check to make sure the required files are located in their respective directories. Check the path entries. **Note!** Absolute paths should be avoided in the application as any later changes in the drive path (e.g. from C:\ to D:\) or in the directory structure are not supported. The system directory as well as the Windows NT disk drive can also be freely selected.

Clearing the Memory using the "KILLTASK.EXE" Tool

This tool can be used when creating software for clearing the memory. After a standard installation (see chapter “Installing Windows NT and the Function Interface”) it is located in the default directory "C:\Programme\Indramat\MTGUI\Bin\".

The tool provides you with the following two options for clearing the memory:

- complete reinitialization and
- selective reinitialization of the function interface.

complete reinitialization

On starting the "KILLTASK.EXE" application, the following three basic processes of the function interface:

- logic process (LOGINTFC.EXE),
- communication process (COMINTFC.EXE),
- management process (BOFINTFC.EXE)

are removed from the memory, as well as all applications connected to the function interface.

CAUTION

You should first terminate all other (stable) function interface applications correctly.

Only run Killtask after doing this!

If this stipulation is not observed then all claims against Rexroth Indramat are excluded.

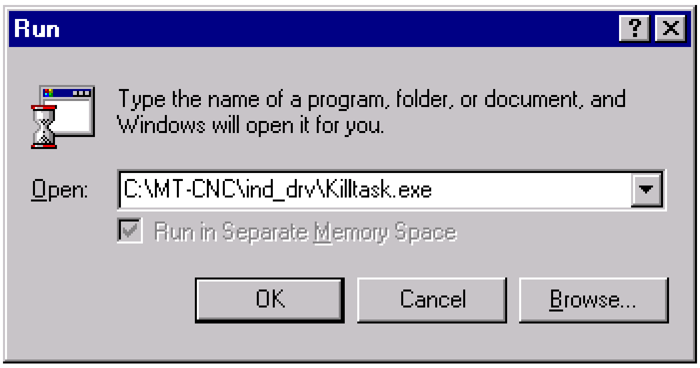
To completely reinitialize, proceed as follows:

⇒ Click on Start and then on the “Run” option.

⇒ Click on the <Find> button to search for the "KILLTASK.EXE" tool.

Note: After a standard installation (see Chapter “Installing Windows NT and the Function Interface”) the ”KILLTASK.EXE” application is located in the default directory "C:\Programme\Indramat\MTGUI\Bin".

⇒ Click on the <OK> button.
All applications connected to the function interface – and the basic processes of the function interface itself – are removed from the memory.



Killtask00.bmp

Fig. 4-6: "Run" dialog box of Windows NT: Complete re-initialization

Selective reinitialization

Only those applications that are connected to the function interface are removed from the memory and from the function interface management structure.

To selectively reinitialize, proceed as follows:

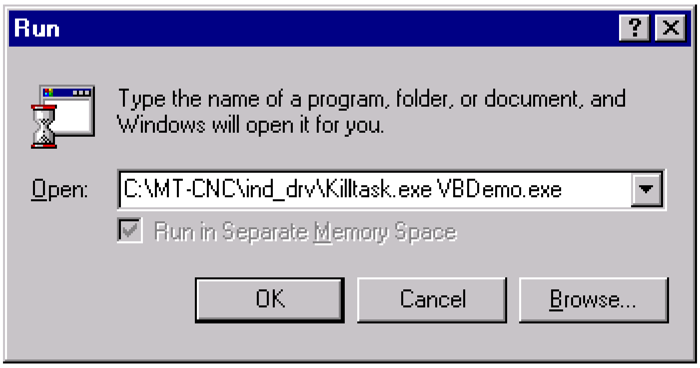
⇒ Click on Start and then on the “Run” option.

Note: You can search for the "KILLTASK.EXE" application by clicking on the “Find...” button. After a standard installation (see chapter “Installing Windows NT and the Function Interface”) this file is located in the default directory "C:\Programme\Indramat\MTGUI\Bin\".

⇒ In the text box, enter the name of the application that is to be removed from the memory and from the management structure of the function interface (here VBDemo.exe).

⇒ Then click on the <OK> button.

The client (here “VBDemo.exe”) is removed from the memory and from the management structure of the function interface.



Killtask01.bmp

Fig. 4-7: "Run" dialog box of Windows NT: Selective re-initialization

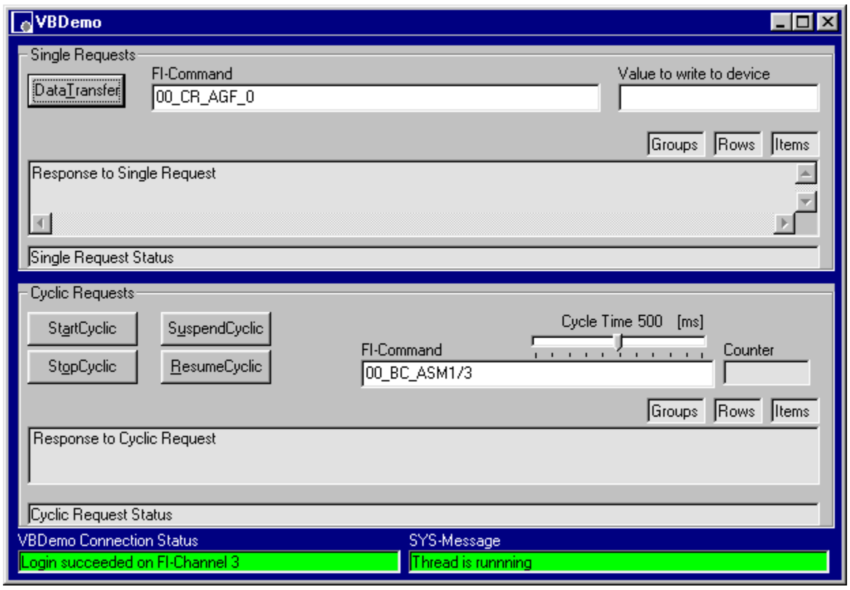
Issuing FI Commands using the “VBDemo” Application

Single FI commands and cyclical requests can be issued by the “VBDemo” application.

To start the application, proceed as follows:

⇒ Click on start, point to Programs, then to Rexroth Indramat and finally to FI.

⇒ Click on VBDemo.



VBDEMO.bmp

Fig. 4-8: The "VBDemo" user program

“Single Requests” Dialog Box

This dialog box allows single requests to be issued that both read and write using the “Data Transfer” routine.

To do this, enter the FI command in the “FI-Command” entry field. If a write request is made, then also enter the value that you wish to write in the "Value to write to device" box (see Chapter “Function Interface Commands”).

Then issue the FI command you have entered to the function interface by clicking on the <DataTransfer> button.

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

“Cyclic Requests” Dialog Box

This dialog box allows cyclic requests that write to be issued using the “StartCyclicPipe” routine.

To do this, enter the FI command in the entry field ”FI-Command” (see chapter Function Interface Commands).

Then issue the FI command entered cyclically to the function interface by clicking in the <StartCyclic> button.

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

Note: You can change the request time from between 10 to 100 ms using the “Cyclic Time” thumb switch.

To stop the cyclic request, click on the <StopCyclic> button. This will cause the “StopCyclicPipe” routine to be executed.

Note: Clicking on the <SuspendCyclic> button processes the “suspendCyclicPipe” routine and sets the cyclic request to standby mode. To reactivate the cyclic request, click on the <ResumeCyclic> button. This will cause the “ResumeCyclicPipe” routine to be executed.

"VBDemo Connection Status" Dialog Box

Displays the login status of the application at the function interface. There are two statuses:

- The dialog box is shaded green and shows the function interface channel (LOG channel) that has been assigned to the application.
- The dialog box is shaded in red and shows the error code that has been caused by logging in via the “LogInIf” login routine.

Starting "VBDemo" in Diagnostics Mode

To start the “VBDemo” program in diagnostics mode, proceed as follows:

⇒ Open the Windows NT Explorer. To do this, click on Start, point to Programs and then click on Windows NT Explorer.

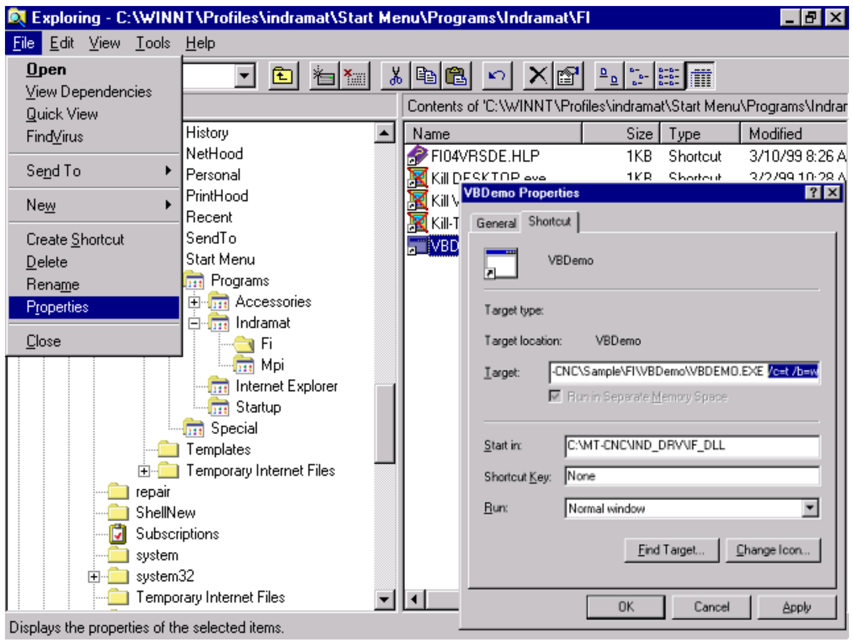
⇒ Via Winnt, go to Profiles and into the User Profile by which the function interface was installed.

⇒ Click on the Start Menu, point to Programs, then to Rexroth Indramat and finally click on “FI”.

⇒ Click on VBDemo and open the Properties dialog box via the Explorer menu file.

⇒ Click on the tab page link and enter the start parameter "/c=t /b=w" in the “Target” text field.

⇒ Click on the <Close> button and VBDemo will be started in diagnostics mode the next time it is called.



VBDemo_Explorer.bmp

Fig. 4-9: Starting VBDemo in the diagnostics mode of the function interface

Outputting Diagnostic Messages

By passing on the start parameters when starting the management process "BOFINTFC.EXE", various function interface diagnostic messages can be outputted to the screen.

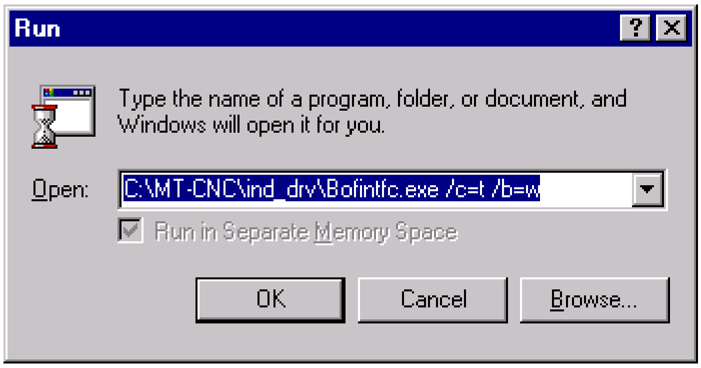
To start the function interface in diagnostics mode, proceed as follows:

⇒ Click on Start and then on the “Run” option.

Note: You can search for the management process "BOFINTFC.EXE" by clicking on the “Find” button. After a standard installation (see chapter “Installing Windows NT and the Function Interface Commands”) this file is located in the default directory "C:\Programme\Indramat\MTGUI\Bin\".

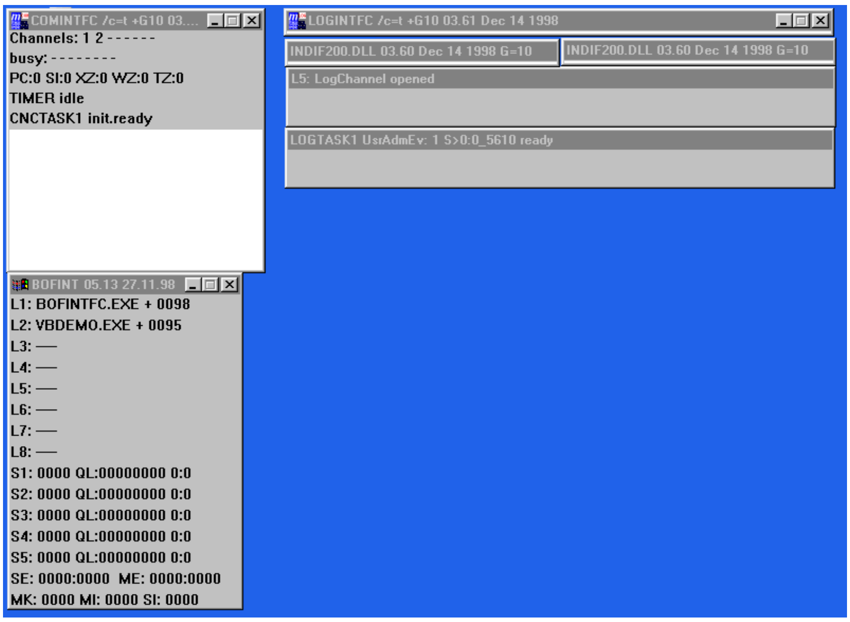
⇒ Enter the start parameter "/c=t /b=w" in lower case letters in the text box (observe spaces between entries).

⇒ Then click on the <OK> button.
The function interface is now started in diagnostics mode.



AusfuehrenDiagnose.bmp

Fig. 4-10: "Run" dialog box of Windows NT: BOFINTFC.EXE$



Diagnose.bmp

Fig. 4-11: Diagnose mode of the function interface

Meaning of the Counters

Five counters are shown in the 3rd line of the diagnostics window of the communication process (COMINTFC.EXE). The individual counters mean the following:

PC Number of communication errors that have occurred in the direction of transmission from device → PC.

SI Number of communication errors that have occurred in the direction of transmission from PC → device.

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

WZ The counter is increased if, in spite of five repetitions, it has not been possible to transmit a valid telegram to the device. The counter is increased by one if the "XZ" counter has been previously increased by five. In this case, the timeout counter is also increased by one.

TZ Timeout counter. The number of timeouts that occur are counted in this counter. A timeout is generated if, in spite of five repetitions, it has not been possible to transmit a valid telegram to the device.

The active control channels are displayed in the lower lines (CNC/DMA-Task).

Data accesses made by the individually connected applications are displayed in the diagnostics window of the LOG channel on the left side of the screen.

In the control window of the management process (BOFINT), the applications are shown that are known in the management mechanism of the BOF process.

Windows NT Key Combinations

The most important key combinations for using Windows NT without a mouse are displayed in the following table.

Action	Key combination
Open start bar	<Ctrl>+<Esc>
Navigate within the opened start bar and in the opened submenus in the start bar	<Arrow key left, right>, or <Arrow key up, down>
Select (start) the applications in the opened submenus in the start bar	<Enter>
Start Windows NT Task Manager	<Ctrl>+<Shift>+<Esc>
Move within the Windows NT menu	<Tab>
”Right mouse click” on button moved to	<Shift>+<F10>
Switch within a menu to other tab pages	<Ctrl>+<Tab>
Switch between opened applications	<Alt>+<Tab>

5 Installing Windows NT and the Function Interface

5.1 The Windows NT Operating System

Using the Windows NT operating system and the possibility of running various applications parallel to one another requires a powerful computer.

The hardware requirements depend directly on the number and nature of the applications running concurrently on the PC. This should be taken into account during the project planning phase. The network cards used and their drivers require a great deal of computing power which might then not be available for the rest of the system. Hardware must therefore be selected with great care and utmost precision.

Note: For the Windows NT Operating System, we recommend a PC with a Pentium processor and at least 32 MB RAM, as well as at least 500 MB available space on the hard drive.

Multitasking and Windows NT

Whereas under Windows 3.1x what is known as “cooperative” or "non-pre-emptive" multitasking controlled several applications running concurrently, genuine “pre-emptive” multitasking is integrated into Windows NT.

Non-pre-emptive Multitasking

Here, it is not the operating system that decides how much computing time is to be allocated, but the application itself; and the application decides when to surrender time back for a short while to the operating system. The disadvantage of this is that when several applications are running simultaneously, working with them in parallel is only possible to a limited degree.

Pre-emptive Multitasking

The operating system itself decides how much computing time is to be allocated to the individual applications. Switching between individual applications is now much more fluid a process as the operating system is able to distribute computing time faster and at shorter intervals, creating the impression that several instructions really can work “simultaneously and unrestrictedly”.

Windows NT Task Manager

The applications running can be monitored and controlled by the Task Manager i.e., applications that have been started can be overlaid on the desktop or can be terminated. Furthermore, it is possible to start applications or switch to other applications.

Calling the Task Manager

⇒ using the key combination <Ctrl>+<Shift>+<Esc>

⇒ clicking with the right mouse button on the taskbar

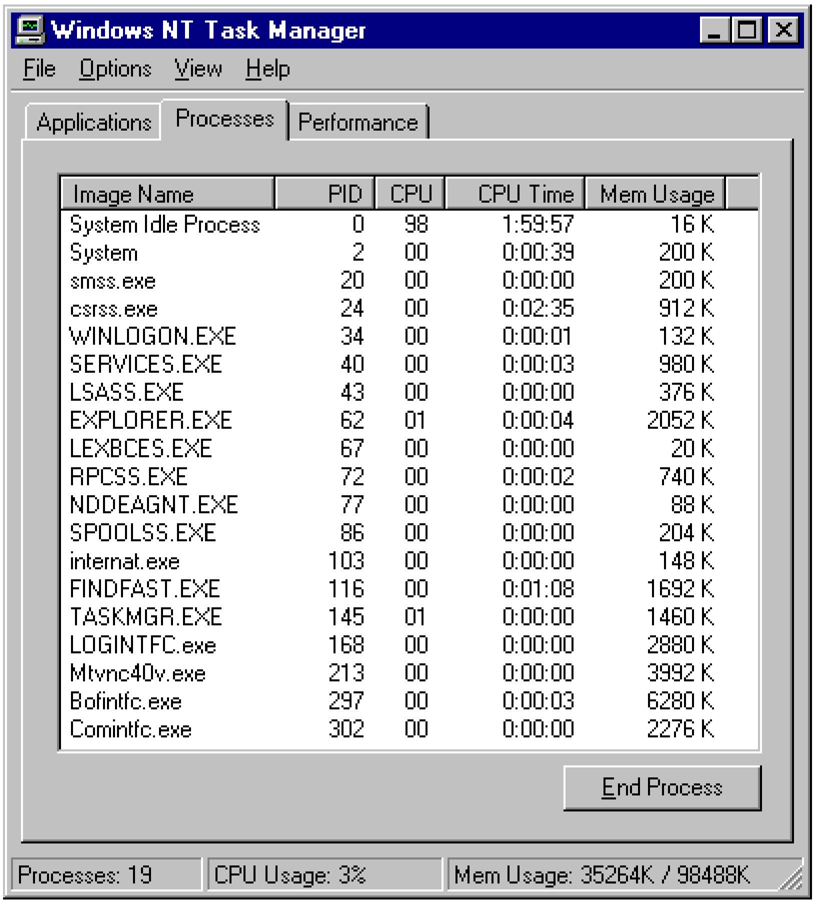


Image Name	PID	CPU	CPU Time	Mem Usage
System Idle Process	0	98	1:59:57	16 K
System	2	00	0:00:39	200 K
smss.exe	20	00	0:00:00	200 K
csrss.exe	24	00	0:02:35	912 K
WINLOGON.EXE	34	00	0:00:01	132 K
SERVICES.EXE	40	00	0:00:03	980 K
LSASS.EXE	43	00	0:00:00	376 K
EXPLORER.EXE	62	01	0:00:04	2052 K
LEXBCES.EXE	67	00	0:00:00	20 K
RPCSS.EXE	72	00	0:00:02	740 K
NDDEAGNT.EXE	77	00	0:00:00	88 K
SPOOLSS.EXE	86	00	0:00:00	204 K
internat.exe	103	00	0:00:00	148 K
FINDFAST.EXE	116	00	0:01:08	1692 K
TASKMGR.EXE	145	01	0:00:00	1460 K
LOGINTFC.exe	168	00	0:00:00	2880 K
Mtvnc40v.exe	213	00	0:00:00	3992 K
Bofintfc.exe	297	00	0:00:03	6280 K
Comintfc.exe	302	00	0:00:00	2276 K

Taskmanager.bmp

Fig. 5-1: Windows NT Task Manager

Note: You can bring applications consecutively up to the front of the screen (overlay them) using the key combination <Alt>+<Tab> without having to make your selection using Task Manager.

5.2 Setting the Windows NT System Properties

Performance Features

To guarantee an optimal reaction time for the function interface, the performance boost for the application in the foreground should be set to "none".

Note: Safe and error-free operation of the function interface is only ensured when the performance boost for the application in the foreground is set to "none".

To make this setting, proceed as follows:

⇒ Click on start, point to Settings, then to System Control and finally to System.

⇒ Click on the "System Properties" tab page and set the thumb to “none”.

⇒ Then click on the <OK> button.

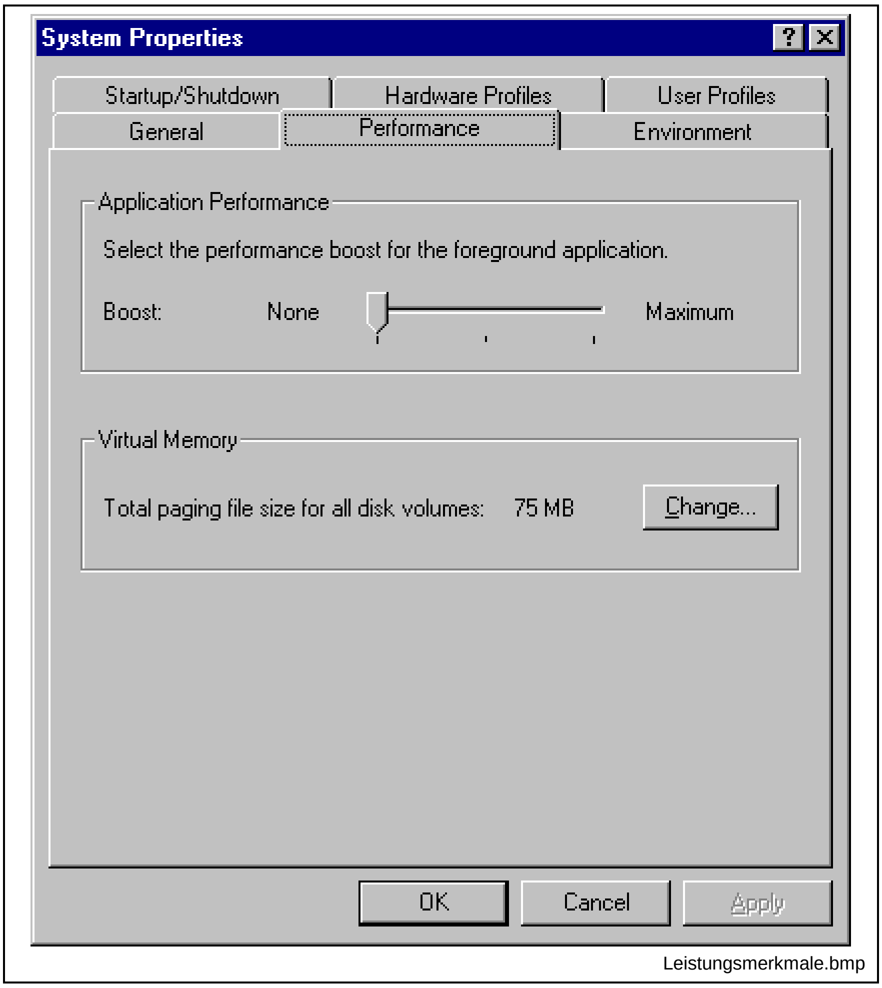


Fig. 5-2: Setting the WindowsNT system property "Performance"

Note: The setting for "Virtual Memory" may differ from the setting of your system.

Date/Time properties

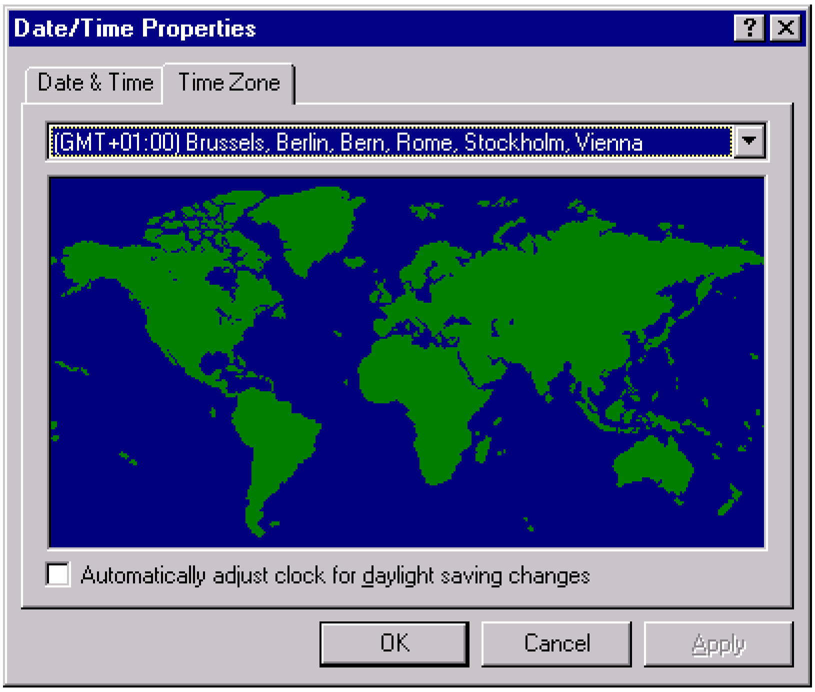
It is required for the exchange of Rexroth Indramat files between two PCs, to have an identical time zone setting. Furthermore, the automatic clock adjustment for daylight saving (switching between summer and winter times) must be deactivated.

To make this setting, proceed as follows:

⇒ Click on Start, point to Settings, then to System Control and finally to Date/Time Properties.

⇒ Click on the Time Zone tab page and deactivate the “Automatically adjust clock for daylight saving changes” toggle button.

⇒ Then click on the <OK> button.



Zeitzone.bmp

Fig. 5-3: Date and time settings

5.3 Installing the Function Interface

Note: In Version 07, the function interface cannot be installed separately, but only within the context of the relevant Rexroth Indramat GUI.

5.4 Directory and File Structure of the Function Interface

Contents of the "INDRAMAT.INI" File

The global settings for the function interface are stored in the "Indramat.ini" file. The function interface looks for the file in the "C:\Programme\Indramat\MTGUI\BasicData\Resource" directory.

However, the default directories as well as the drive [LW] can be freely selected. The "INDRAMAT.INI" file corresponds to the Microsoft Windows INI standard and is constructed as follows:

Identifier	Values	Explanation
[IfConfig]	This contains the configuration settings for the function interface	
IfInstDir=	z.B.: C:\Programme\ Indramat\MTGUI\Bin	Directory in which the three basic processes of the function interface are installed. This entry is set by the installation program.
AndInstDir=	z. B.: C:\MTA200	!Optional! Directory for MTA200 control software. Details refer to the "MTA200.EXE" application.
IfDllMode=	z.B.: 04.10 03.xx [00..70], 04.xx [00,10]	Here the mode is specified that is to be supported by the function interface. The IfDllMode of a more recent version of the function interface can, for example, be operated in the same mode as the previous version for troubleshooting.
IfVersion=	z.B.: 06V00	Current version of the function interface.
GBOVERSION=	z.B.: 005-21Vxx	Current version of the Rexroth Indramat GUI.
INDRAMAT_x=	x=1 to 9 Name of file	Reference to directory C:\Programme\Indramat\MTGUI\BasicData\Resource The existence of the files named here is checked on starting the function interface. The following applies: File identifier without an extension is a DLL. e.g. Indramat = INDRAMT.DLL Several file identifiers are separated by a "comma".
IND_DLL_x=	x=1 to 9 Name of file	Reference to directory [LW]:\....\MTGUI\Bin. The existence of the files named here is checked on starting the function interface. The following applies: A file identifier without an extension is a DLL, e.g. NDFS100 = INDFS100.DLL. Several file identifiers are separated by a "comma".
IF_DLL_x=	x=1 to 9 Name of file	Reference to directory [LW]:\....\MTGUI\Bin. The existence of the files named here is checked on starting the function interface. The following applies: File identifier without an extension is a DLL. e.g. INDIF000 = INDIF000.DLL Several file identifiers are separated by a "comma".
[Install]	This contains entries regarding the installed System200 software components.	
HMIVersion=	z.B.: 01V06	Version ID of the System200 software component WIN-HMI
TYP=	z.B.: HMI	System200 software component WIN-HMI
ServicePack=	z.B.: 2 [1,2,...]	Service Pack ID of the installed System200 software components
SP_Release=	[1, to F]	State of the Service Pack release ID (F = Final Version)

Example Entries in the "INDRAMAT.INI" File

```
[IfConfig]
IfInstDir=C:\Program Files\Indramat\MTGUI\Bin
AndInstDir=C:\MTA200
IfDllMode=04.10
IFVERSION=04V02
GBOVERSION=005-21V09
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=01V06
TYP=HMI
SP_Release=F
```

The DLL entries (If_DLL_1,...) can be expanded up to the ninth entry (If_DLL_9). A check for the existence of the DLLs is only made when the DLLs have been previously entered at the corresponding parameters. If the file name is given without an extension then the extension is automatically assumed to be "DLL". If the existence of another file is to be checked then the file extension of this file must also be entered, e.g., "Userprogram.dat".

Contents of the "IND_DEV.INI" File

The configuration of the individual communication addresses and the settings of the various Rexroth Indramat devices are determined in this file. The "IND_DEV.INI" file is edited by the system configurator and is located in the "[LW]\...\CONFIG\" directory.

Rexroth Indramat system configurator

The Rexroth Indramat System Configurator is an editor that sets and lists the devices connected to the control PC. The device addresses, the device type and the description of the communication path to the device are used for this. The goal is to create a 1:1 copy of the device structure connected to the control PC; this structure is termed the system configuration.

The system configuration is stored on the control PC. Furthermore, the devices can also be assigned basic properties, e.g. a Type MTVNC device (virtual MTC for the function "Offline Simulation") can be assigned to a device of Type MTC200-P or MTCNC, etc., in order to form a simulation pair. Here, the parameter records of the real device can be used by the virtual device allowing a simulation of NC programs to be started.

Note: An online help is also included in the system configurator. It can be called up by pressing the <F1> function key while the program is running.

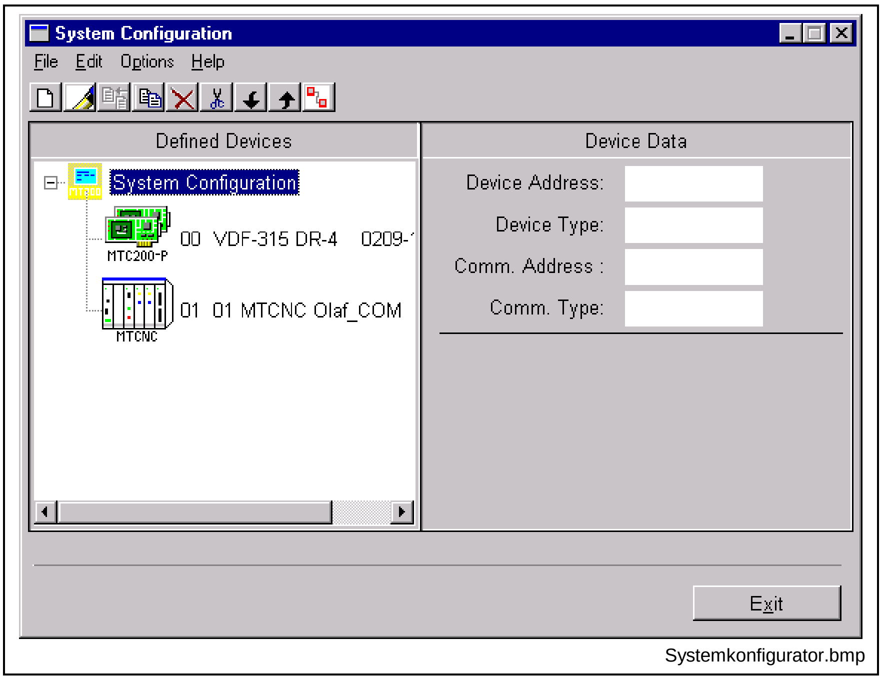


Fig. 5-4: Rexroth Indramat system configurator

The "IND_DEV.INI" file corresponds to the Microsoft Windows INI standard and is structured as follows:

Identifier	Values	Explanation
[CommAddrX]	X = 1...8	Assignment of the communication channel (thread) of the function interface.
CommStr=	V24, Port [COM1...4], baud rate, parity, type of interface, packet counter - or - DMA, address, offset, length - or - SHM, Channel No. [1...15]	Communication via RS232 serial interface, e.g. V24,COM1,19200,NONE,RS232,TCON. Communication via RS485 serial interface, e.g. V24,COM2,19200,NONE,R485H,TCON For communication via a dual port RAM, a DMA channel is also required for the MTC200-P, e.g., DMA,$D000,$0000,$2000. Communication channel to the MTVNC via a shared memory, e.g. SHM, 1.
Timeout=	>= 1000 [msec] Preset = 3500 [msec]	! OPTIONAL ! Time in which a response must be received from the device.
- **or** – only for dual port RAM (e.g. for MTC200-P)		
CommStr=	DPR, address [$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 according to the settings on the respective card.
PortVal=	$20, $21, ... $3F e.g., $28 at address [$D000,$0000]	Configuration byte for setting the physical memory address of the MTC/MTS card.
Timeout=	>= 1000 [msec] Preset = 3500 [msec]	! OPTIONAL ! Time in which a response must be received from the device.
[DeviceAddrX]	X = 0..0.15	device address
Component type1=	e.g., MTS-P01.2 NONE, MTS-P, MTS-P01.2, MTS-P02.2, MTS-R-M1, PPC-R	Name of the PLC component type
Component type2=	e.g., MTC-P-G2 NONE, MTC-P, MTC-R, PPC-R	Name of the NC component type
DeviceName=	Max. 32 ASCII characters	Device name; e.g., Processing Center 12T34

Identifier	Values	Explanation
DeviceType=	e.g. MTC200-P-G2 MTVNC MTC200-P-G2 MTC200-R-G2 ISP200-P-G2 ISP200-R-G2 TRA200-R ECODRIVE03 MTA200-P SERCANS-A SERCANS-P SYNAX-P SYNAX-R	Device type: MTC with PLC PC variant Virtual MTC MTC with PLC PC variant MTC with PLC RECO variant Standalone PLC PC variant Standalone PLC RECO variant TRANS200 RECO variant Ecodrive03 MTA200 control SERCANS-A card (via serial interface) SERCANS-P card (via serial interface) SYNAX PC variant SYNAX RECO variant
DeviceAssign=	0...15, NO	Assignment of a simulation pair. The MTVNC is, for example, hereby assigned to a real MTCNC.
DeviceStatus=	ON, OFF	Assignment of whether or not the device is incorporated into the management structure of the function interface.
MtvncMode=	OFF, RUN, STANDBY	! Only for virtual MTC (MTVNC) ! Status of the MTVNC with inactive utilization
MtvncMemory=	256, 257 .. 16383 [KB] Preset = 512 [KB]	! Only for virtual MTC (MTVNC) ! Size of the PC memory used by the MTVNC.
CommAddr=	1...8	Assignment of the communication address. Corresponds to the [CommAddr1...8] parameter.
PLC=	YES, NO	PLC support for the device. E.g. one MTVNC, TRANS200-R has no PLC, therefore the parameter PLC=NO is set.
[DeviceOrder]	This contains the configuration settings for the system configurator.	
Order=	0,1,2, ...15	Order in which the configured devices are displayed.
[NetManager]	This contains the configuration settings for the network driver "NETINTFC.EXE"	
NetManagerMode=	OFF, RUN	Starts 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,$2000	Communication address 2 Assignment of the DMA channel.
[CommAddr3] CommStr=V24,COM1,19200,NONE,RS232,TCON	Communication address 3 Settings for communication via RS232.
[CommAddr4] CommStr=DPR,$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.

Entry	Explanation
[DeviceAddr0] CommAddr=1 Componenttype1= MTS-P-G2 Componenttype2= MTC-P-G2 DeviceAssign=NO DeviceName=VDF-315 DR-4 0209-15 DeviceStatus=ON DeviceType=MTC200-P PLC=YES	Device address 00 Assigned communication channel PLC components MTS-P-G2 CNC components MTC-P-G2 No MTVNC assigned Device name Device is available and ready for operation Device type PLC support
[DeviceAddr1] CommAddr=4 Componenttype1= MTS-P02.02 Componenttype2= MTC-P-G2 DeviceAssign=NO DeviceName= Processing center 12T35 DeviceStatus=ON DeviceType=MTC200-P PLC=YES	Device address 01 Assigned communication channel PLC components MTS-P02.02 CNC components MTC-P-G2 No MTVNC assigned Device name Device is available and ready for operation Device type PLC support
[DeviceAddr2] CommAddr=5 Componenttype1= NONE Componenttype2= NONE DeviceAssign=1 DeviceName= V-Processing center 12T34 DeviceStatus=ON DeviceType=MTVNC MtvncMemory=512 MtvncMode=RUN PLC=NO	Device address 02 Assigned communication channel PLC component not available CNC component not available Assigned to device address 01 (simulation pair) Device name Device is available and ready for operation Device type Size of the PC memory Status during inactive use No PLC support
[DeviceAddr3] CommAddr=3 Componenttype1= NONE Componenttype2= PPC-R DeviceAssign=NO DeviceName= TRANS200 DeviceStatus=ON DeviceType=TRANS200-R PLC=NO	Device address 03 Assigned communication channel PLC component not available CNC component PPC-R No MTVNC assigned Device name Device is ready for operation Device type No PLC support
[DeviceOrder] Order=3,0,1,2	Order in which the configured devices are displayed in the system configurator
[NetManager] NetManagerMode=RUN	 Network driver is started
[BofManager] PollDeviceStatus=OFF PollDeviceStatusRate=4000 PollDeviceStatusCheckFactor=4	 On switching on (ON), the device status of the devices is requested cyclically. The cycle time of a device request is controlled by this value. If a device can not be addressed then a request is no longer made until a time has passed that is the product of PollDeviceStatusRate multiplied by the PollDeviceStatusCheckFactor.

Contents of the "[LW]:\Winnt\System32\" System Directory

The following Microsoft class libraries are stored in the system directory of Windows NT "[LW]:\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 "[LW]:\Winnt\System32\Drivers\" Driver Directory

The following files of the core-mode driver are stored in the driver directory of Windows NT "[LW]:\Winnt\System32\Drivers\":

File	Explanation
MTCNC00I.SYS	Windows NT core-mode driver
MTCNC00I.INI	Configuration file of the core-mode driver
REGINI.EXE	Application for registering the core-mode driver

Contents of the "[LW]:\...\MTGUI\BasicData\Help\" Directory

The drive as well as the path "[LW]:\...\" are pre-set during the standard installation routine to "C:\Programme\Indramat\MTGUI\". The following Windows 95/NT help files for the printed English and German manuals are stored in the "C:\...\MTGUI\BasicData\Help\[Language]\" directory:

File	Explanation
FIVRS_DE.HLP	Windows 95/NT help file in German
FIVRS_DE.CNT	Definition file of the Windows 95/NT help file
FIVRS_EN.HLP	Windows 95/NT help file in English (in preparation)
FIVRS_EN.CNT	Definition file of the Windows 95/NT help file

Contents of the "[LW]:\...\MTGUI\BasicData\Resource" Directory

The drive as well as the path "[LW]:\...\" are pre-set during the standard installation routine to "C:\Programme\Indramat\MTGUI\Bin". The following files are contained in the "C:\...\MTGUI\BasicData\Resource" directory:

File	Explanation
BOFINTFC.DAT	BOF process definition file
INDRAMAT.INI	File with global function interface settings
LOGINTFC.DAT	Definition file of the logic process
MECX.DAT	Definition file for the MECX device group
MISX.DAT	Definition file for the MWSX device group
MPCX.DAT	Definition file for the MPCX device group
MSCX.DAT	Definition file for the MSCX device group
MSYX.DAT	Definition file for the MSYX device group
MTAX.DAT	Definition file for the MWAX device group
MTCX.DAT	Definition file for the MWCX device group
VERSION.DAT	Definition file for the version ID

Example Entries in the "VERSION.DAT" File

The version ID of the individual parts of the program as well as the version of the function interface are entered in the "VERSION.DAT" file. This applies to all program parts (EXE, DLL) of the function interface. The following example shows the entries in this file:

Name	Build	Version	Date	Start Parameter
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 "[LW]:\...\MTGUI\Bin" Directory

The drive as well as the path "[LW]:\...\" are pre-set during the standard installation routine to "C:\Programme\Indramat\MTGUI\". The following function libraries of the function interface are contained in the C:\...\MTGUI\Bin directory:

File	Explanation
BOFINTFC.EXE	BOF process
COMINTFC.EXE	Communication process
INDFS100.DLL	Processing the file ID
INDIF000.DLL	General functions for the user.
INDIF120.DLL	Functions for outputting the trace file.
INDIF130.DLL	Functions for the BOF 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 BOF process.
INDIF310.DLL	Functions for the BOF process.
INDIF320.DLL	Functions for the DOS - Windows NT connections.
INDIF330.DLL	Functions for the BOF process.
INDIF340.DLL	Functions for the BOF process.
INDIF350.DLL	Functions for the DOS - Windows NT connections.
INDIF360.DLL	Functions for file access.
INDIF400.DLL	BOF process access to parameters.
INDIF500.DLL	Functions for access optimization.
INDIF510.DLL	Functions for access optimization.
INDIF520.DLL	Functions for access optimization.
INDIF530.DLL	Functions for access optimization.
INDIF540.DLL	Functions for access optimization.
INDIF550.DLL	Functions for access optimization.
INDIF560.DLL	Support for network connections and PLC and NC optimizer
INDIF570.DLL	Server for fast M-K communication
INDIF600.DLL	Functions for access to NC programs.
INDIF610.DLL	Functions for access to NC programs.
INDIF700.DLL	Functions for access to NC compiler.
INDIF800.DLL	Higher functions of all device groups.
INDIF810.DLL	Functions for access to the MWCX device groups.
INDIF820.DLL	Functions for access to the MSCX device groups.
INDIF830.DLL	Functions for access to the MECX device groups.
INDIF840.DLL	Functions for access to the MWAX device groups.
INDIF860.DLL	Functions for access to the MSYX device groups.
INDIF870.DLL	Functions for access to the MWSX device groups.

File	Explanation
INDIFA00.DLL	Functions for HMI support of the MWCX device groups.
INDIFZ00.DLL	Functions for access to the MWAX device groups.
INDMA900.DLL	Processing the MAP file
INDMA110.DLL	Connecting the MAP file
INDOF160.DLL	Using various system utilities
INDRAMAT.DLL	Access to global settings (GetInstPath, etc.)
INDUT140.DLL	Using various system utilities
KILLTASK.EXE	Application for terminating function interface applications (see Chapter "Programming")
LOGINTFC.EXE	Logic process
NETINTFC.EXE	Application for connection of client/server
VBDemo.exe	Test program in Visual Basic
IFDemo.exe	Test program in Visual C++

6 Construction and Availabilty of the FI Command

6.1 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.

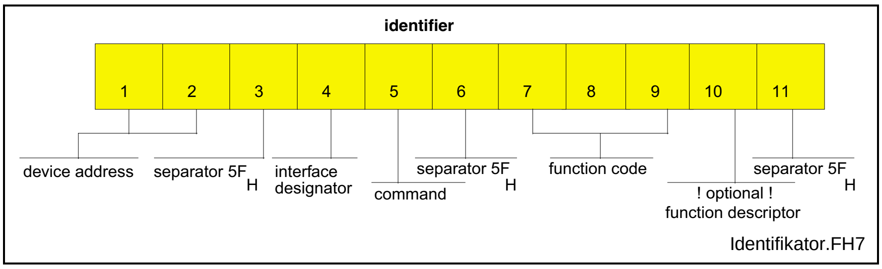


Fig. 6-1: Identifier

device address

The device address corresponds to the system address within the Rexroth Indramat GUI. This means, for example, that device 00 corresponds to system 0. Please observe, however, that the Rexroth Indramat GUI always requires a device 00. The addresses are listed specific to the device group in the following table as well as in the chapter "Function Interface Commands".

Address	Group	Affiliated device types
[xx]	MPCX	PCs
[00...63]	MWCX	MTC200-P-G2, MTC200-R-G2, MTVNC
[00]	MSCX	SERCANS-A, SERCANS-P
[00...63]	MWMX	VMISP200-P-G2, VMISP200-R-G2
[00...63]	MWSX	ISP200-P-G2, ISP200-R-G2
[00...63]	MWAX	MTA200-P (MTA200-controller)
[00...63]	MSYX	SYNAX200-P, SYNAX200-R
[00...63]	MWYX	SYNAXISP200-P-G2, SYNAXISP200-R-G2

Separator

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

Interface designator

<u>M</u>anagement process
<u>C</u>ontroller (logic process and communication process)

All data access via the interface identifier “M” are managed by the management process so as to ensure, for instance, that a user program can access the data from a controller via function calls. 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 management process ensures that a different user program cannot open the same file again.

Data access by way of the interface identifier “C” is managed by the logic process and by the communication process, thus enabling access to data of the relevant device groups (MWCX and MWSX, etc.).

Read and Write Commands

W = Single **W**rite	(Writing)
R = Single **R**ead	(Reading)
C = **C**yclic Read	(Cyclic reading)
B = **B**reak Cyclic Read	(Interrupt cyclic reading)

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 4, “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 with a 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

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

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

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 requests. This is why this access is 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.

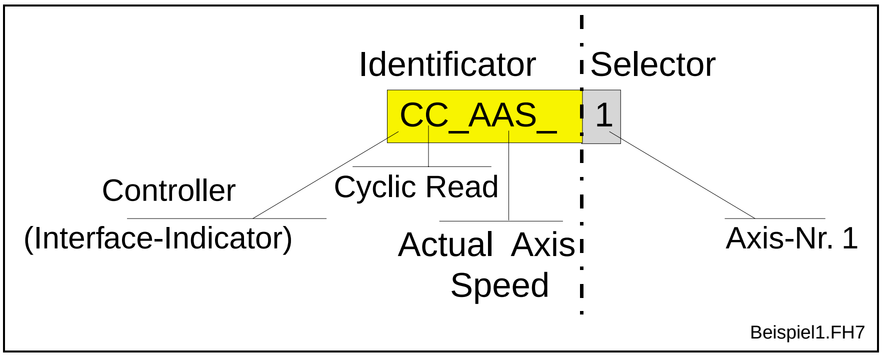


Fig. 6-2: Example 1: Cyclic reading in ASCII code

Example 2 (cyclic reading) Cyclic reading of the current feedrate in the NC process 3 of device address 02.

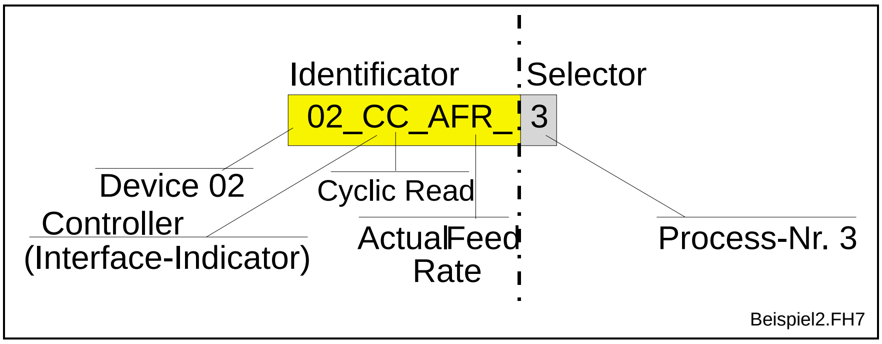


Fig. 6-3: Example 2: Cyclic reading in ASCII code

Data code

The data code is an optional identifier 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 response in the result buffer.

The following coding types are supported:

1 = ASCII Preset !

2 = Binary

3 = ANSI

4 = Unicode (not yet implemented).

Note: The data in the control is generally filed in ASCII.

Example 3 (cyclic reading)

Cyclic reading of the current spindle speed of the 2nd spindle in the NC process 1, of the device address 01 in the “Binary” data code.

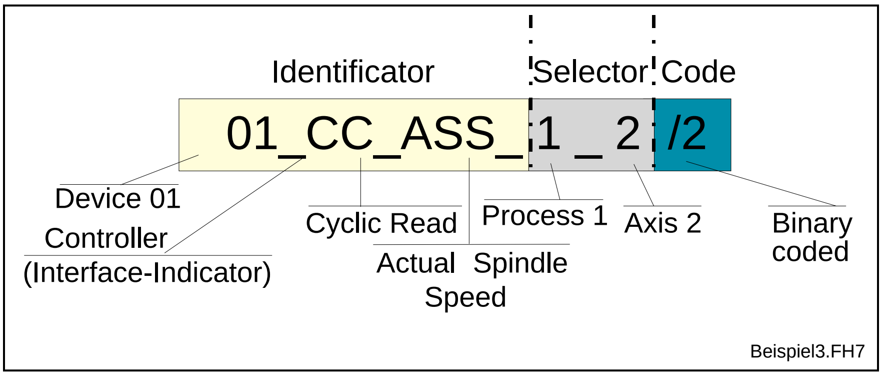


Fig. 6-4: Example 3: Cyclic reading in binary code

6.2 Data Tables

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

General Parameters for the MWCX Device Groups

Parameters	Value Range
Axis number	1...32
NC memory	1=A, 2=B
NC block No.	0...9999
NC program number	0...99
NC packet	1...99
Zero point database	0...9
Spindle number	S1, S2, S3
NC 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 = tool edge data
Data element	1...28 for basic tool data 1...40 for tool edge data
Memory	M = magazine/turret S = spindle G = gripper X = index data
Location	1...999 for M 1...4 for S,G 0...16770215 for X

Meanings of the Axes for the MWCX Device Group

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 for the MWCX Device Group

No.	Axis types	Comment
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	Digital C axis
87_H	DIGITAL_SERCOS_E_A	Digital Sercos I/O

Base Units

	Measurement System			
Base unit	**Linear in mm**	**Linear in inch**	**Rotatory in units**	**Specific to main spindle**
velocity	mm/min	inch/min	units/min	1/min
Feed constant	mm	inch	units	--
acceleration	mm/s^2	$inch/s^2$	$units/s^2$	rad/s^2
Distance	mm	inch	units	deg
Speed	rpm	rpm	rpm	rpm
Cutting speed	m/min	inch/min	units/min	--

6.3 Overview of FI Commands

The following table presents an overview of the available FI commands, arranged according to device groups.

Note: A detailed description is contained in the following Chapters “Function Interface Commands”.

Overview of the MPCX Device Group

Com.	Description	Process	Read	Write	Cyclic
BFJ1	**B**reak **F**unction Interface **J**obs	B	R		
BFJ2	**B**reak **F**unction Interface **J**obs	B	R		
CCP1	**C**ell **C**onfiguration **P**arameter	B	R		
CCP2	**C**ell **C**onfiguration **P**arameter	B	R		
CCP3	**C**ell **C**onfiguration **P**arameter	B	R		
CCP4	**C**ell **C**onfiguration **P**arameter	B	R		
CCP5	**C**ell **C**onfiguration **P**arameter	B	R		
CEI1	**C**ommunication **E**rror **I**nfo	B	R		
CPR1	**C**reate **PR**ocess	B		W	
CPR2	**C**reate **PR**ocess	B		W	
DFJ1	**D**elete **F**unction Interface **J**ob	B	R		
DFJ2	**D**elete **F**unction Interface **J**ob	B	R		
DFS1	**D**elete **IF** Command **S**tack	B		W	
DPR1	**D**elete **PR**ocess	B		W	
ERI1	**ER**ror **I**nformation	B	R		
FCP1	**F**ar Device **C**onfiguration **P**arameter	B	R		C
FCP2	**F**ar Device **C**onfiguration **P**arameter	B	R		C
FCP3	**F**ar Device **C**onfiguration **P**arameter	B	R		C
FDC1	**F**ar **D**evice **C**onfiguration	B	R		C
FIT1	**F**urther **I**nfo **T**ext	B	R		
FPC1	**F**ar **PC** **C**onfiguration	B	R		C
GDB1	**G**lobal **D**ata **B**uffer	B	R	W	
ICA1	**I**nitialisation **C**ommunication **A**ddress	B		W	C
IFJ1	**I**nformation about **F**unction Interface **J**obs	B	R		C
IFJ2	**I**nformation about **F**unction Interface **J**obs	B	R		C
IFS1	**IF** Command **S**tack Info	B	R		
LDT1	PC **L**ocal **D**ate **T**ime	B	R		C
LNG	Active **L**a**NG**uage	B	R	W	
MSG	**M**e**S**sa**G**e	C			C
NST1	**N**T-**S**hu**T**-Down	B		W	
NST2	**N**T-**S**hu**T**-Down	B		W	
PAF1	**PA**rameter **F**ile Converted	B		W	
PHD1	**Ph**ysical **D**irectory	B	R		

Com.	Description	Process	Read	Write	Cyclic
POB1	**PO**rt **B**yte Access	B	R	W	
POW1	**PO**rt **W**ord Access	B	R	W	
RPR1	**R**eady **PR**ocess	B		W	
SDP1	**S**tart **D**evice **P**olling	B		W	
SFW1	**S**et **F**ocus to **W**indow	B		W	
SFW2	**S**et **F**ocus to **W**indow	B		W	
SSM1	**S**et **S**ys **M**essage	B		W	
SSM2	**S**et **S**ys **M**essage	B		W	

Fig. 6-5: Overview of the MPCX device group

Overview of the MTCX Device Group

Com.	Description	Process	Read	Write	Cyclic
AAC1	**A**ctual **AC**celeration	C	R		C
AAD	**A**ctive **A**ngle **D**imension	C	R		C
AAS1	**A**ctual **A**xis **S**peed	C	R		C
AAS2	**A**ctual **A**xis **S**peed	C	R		C
ABI	**A**ctual NC-**B**lock **I**nformation	B	R		C
ACS	**A**ctual **C**utting **S**peed	C	R		C
ADN1	**A**ctive **D**-Correction **N**umber	C	R		C
AEM	**A**ctive **E**vent **M**onitoring	C	R		C
AEN	**A**ctive **E**dge-**N**umber	C	R		C
AFO1	**A**ctive **F**eedrate **O**verride	C	R		C
AFR	**A**ctive **F**eed**R**ate	C	R		C
AGF	**A**ctive **G**-**F**unction	C	R		C
AMF	**A**ctive **M**-**F**unction	C	R		C
AMM1	**A**ctive **M**echanism **M**essage	B	R		C
AMM2	**A**ctive **M**echanism **M**essage	B	R		C
AMM3	**A**ctive **M**echanism **M**essage	B	R		C
AMM4	**A**ctive **M**echanism **M**essage	B	R		C
AMM5	**A**ctive **M**echanism **M**essage	B	R		C
ANM	**A**ctive NC **M**emory Size	C	R		
API1	**A**ctual **P**arameter **I**ndex	B	R		C
API2	**A**ctual **P**arameter **I**ndex	B	R		C
APM	**A**ctive **P**art-Program **M**essage	C	R		C
APN	**A**ctive **P**art-Program Message Number	C	R		C
APO1	**A**ctual Machine **PO**sition	C	R		C
APO2	**A**ctual Machine **PO**sition	C	R		C
APP	**A**ctive **P**art-**P**rogram number	C	R		C
ARF	**A**xis **R**eference **F**lags	C	R		C
ARO1	**A**ctual **R**apid **O**verride	C	R		C

Com.	Description	Process	Read	Write	Cyclic
ART	**A**xis **R**eference **T**able	C	R		C
ART	**A**xis **R**eference **T**able	B	R		
ASD	**A**ctual **S**pindel **D**ata	C	R		C
ASF	**A**ctual **S**pindle **F**or Process	C	R		C
ASG	**A**ctual **S**pindle **G**ear	C	R		C
ASM1	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM2	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM3	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM4	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM5	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASN	**A**ctual **S**equence **N**umber	C	R		C
ASO1	**A**ctual **S**pindle **O**verride	C	R		C
ASS	**A**ctual **S**pindle **S**peed	C	R		C
ATN	**A**ctive **T**ool-**N**umber	C	R		C
ATP1	**A**ctual **T**ool **P**lace Information	C	R		C
ATP2	**A**ctual **T**ool **P**lace Information	C	R		C
ATP3	**A**ctual **T**ool **P**lace Information	C	R		C
ATR	**A**ctual **T**ool Data **R**ecord	C	R		C
ATU	**A**ctual **T**ool Data **U**pdate	C	R		
AZB1	**A**ctive **Z**ero Offset **B**ank	C	R		C
CCA1	N**C** **C**ycle **A**ccess	B	R	W	
CPO1	**C**ommand **PO**sition (COMMAND)	C	R		C
CPO2	**C**ommand **PO**sition by log. AxisNo	C	R		C
CRT	**C**ontrol **R**ese**T**	C		W	
DAC1	**D**evice **A**xis **C**onfiguration Parameter	B	R		C
DAC2	**D**evice **A**xis **C**onfiguration Parameter	B	R		C
DCA1	NC **D**-**C**orrection **A**ccess	B	R	W	
DCD1	**D**-**C**orrection **D**ata	C	R		C
DCP1	**D**evice **C**onfiguration **P**arameter	B	R		C
DCP2	**D**evice **C**onfiguration **P**arameter	B	R		C
DCR1	**D**-**C**orrection **R**ecord	C	R	W	C
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DIS1	**D**ata **I**dentification **S**tring Parameter	C	R		
DIS3	**D**ata **I**dentification **S**tring NC Packet	C	R		
DIS4	**D**ata **I**dentification **S**tring Tool List	C	R		
DIS5	**D**ata **I**dentification **S**tring Machine	C	R		
DIS6	**D**ata **I**dentification **S**tring NC Program	C	R		
DPN	**D**elete **N**C **P**rogram	B		W	
DPP	**D**elete **P**rogram **P**ackage	B		W	

Com.	Description	Process	Read	Write	Cyclic
DSI1	**D**evice **S**tatus **I**nformation	B	R		C
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTC1	**D**evice **T**ool Management **C**onfiguration	B	R		C
DTC2	**D**evice **T**ool Management **C**onfiguration	B	R		C
DTG1	**D**istance **T**o **G**o	C	R		C
DTG2	**D**istance **T**o **G**o by log. AxisNo.	C	R		C
DTY1	**D**evice **TY**pe	B	R		
DWD1	**D**iagnosis **W**indow **D**ata	B	R		C
DWD2	**D**iagnosis **W**indow **D**ata	B	R		C
EDE1	**E**xisting **D**iagnosis **E**rror	B	R		C
EDE2	**E**xisting **D**iagnosis **E**rror	B	R		C
EDW1	**E**xisting **D**iagnosis **W**indow	B	R		
EDW2	**E**xisting **D**iagnosis **W**indow	B	R		
EDW3	**E**xisting **D**iagnosis **W**indow	B	R		
END1	**E**xisting **N**C **D**iagnosis	B	R		
END2	**E**xisting **N**C **D**iagnosis	B	R		
EPD1	**E**xisting **P**LC **D**iagnosis	B	R		
EPD2	**E**xisting **P**LC **D**iagnosis	B	R		
EPD3	**E**xisting **P**LC **D**iagnosis	B	R		
EPO1	ProgrammEd **PO**sition (END)	C	R		C
EPO2	ProgrammEd **PO**sition (END)	C	R		C
EPT1	**E**xisting **P**roVi **T**ypes	B	R		
EST1	**E**rror **ST**ate	B	R		C
EXD1	**EX**ecution **D**isplay	B	R		C
EXD2	**EX**ecution **D**isplay	B	R		
GPC1	**G**lobal **P**rocess **C**onfiguration	B	R		C
GPC2	**G**lobal **P**rocess **C**onfiguration	B	R		C
GPP1	**G**lobal **P**rocess **P**arameter	B	R		C
GPP2	**G**lobal **P**rocess **P**arameter	B	R		C
IPP	**I**nsert NC-**P**rogram **P**ackage	B		W	
MAP1	**M**odule **A**ssign of **P**rocess	B	R		C
MCD1	**M**odule **C**onfiguration: **D**evice Information	B	R		C
MCM1	**M**odule **C**onfiguration: **M**odule Information	B	R		C
MCP1	**M**odule **C**onfiguration: **P**rocess Information	B	R		C
MCS1	**M**odule **C**onfiguration: **S**FC- Information	B	R		C
MDA1	**M**achine **D**ata **A**ccess	B	R	W	
MDA2	**M**achine **D**ata **A**ccess	B	R	W	
MDA4	**M**achine **D**ata **A**ccess	B		W	
MDI	**M**anual **D**ata **I**nput	C		W	
MDS1	**M**achine **D**ata **S**ingle	B	R	W	

Com.	Description	Process	Read	Write	Cyclic
MFD1	**M**essage **F**iles **D**ownload	B		W	
MFO1	**M**aximal **F**eedrate **O**verride	C	R		C
MFR	**M**aximal **F**eed**R**ate	C	R		C
MKS	**M**achine **K**ey **S**tatus	B	R		C
MKT1	**M**achine **K**ey **T**able	B		W	
MRO1	**M**aximal **R**apid **O**verride	C	R		C
MSG	**M**e**S**sa**G**e	C			C
MSO1	**M**aximal **S**pindle **O**verride	C	R		C
MSS	**M**aximal **S**pindle **S**peed	C	R		C
MTC	**MT**-**C**NC Slot Software Version	B	R		
MTD	**M**achine **T**able **D**ata	C	R	W	C
NCA1	**NC** Program **A**ccess	B	R	W	
NCA3	**NC** Program **A**ccess	B		W	
NCM1	**NC M**essages	B	R		C
NCM2	**NC M**essages	B	R		C
NEA1	**N**C **E**vent **A**ccess	B	R	W	
NEV	**N**C **Ev**ent	C	R	W	C
NMM	**N**C **M**e**M**ory selection	C		W	
NPA1	**N**C **PA**rameter	B	R		C
NPA2	**N**C **PA**rameter	B	R		C
NPA3	**N**C **PA**rameter	B	R		C
NPA4	**N**C **PA**rameter	B	R		C
NPA5	**N**C **PA**rameter	B	R		
NPC1	**N**C-**P**ackage **C**ompiling	B	R		
NPD1	**N**C-**P**ackage **D**ownload	B		W	
NPI	**N**C-**P**ackage D**I**rectory	B	R		
NPS	**N**C **P**rogram **S**election	C		W	
NTN	**N**ext **T**ool-**N**umber	C	R		C
NUA1	**N**C Offset Data **A**ccess	B	R	W	
NVA1	**N**C **V**ariable **A**ccess	B	R	W	
NVS	**N**C **V**ariable **S**ingle	C	R	W	C
OPD1	**O**ptimal **P**osition **D**istance by Axis sign.	C	R		C
OPD2	**O**ptimal **P**osition **D**istance by phys. AxisNo	C	R		C
PAA1	**PA**rameter **A**ccess	B	R	W	
PAA2	**PA**rameter **A**ccess	B	R	W	
PAC1	**P**rocess **A**xis **C**onfiguration Parameter	B	R		C
PAC2	**P**rocess **A**xis **C**onfiguration Parameter	B	R		C
PAD1	**PA**rameter **D**eactivate	B		W	
PAS1	**PA**rameter **S**et Active	B		W	
PDT	**P**arameter **D**efinition **T**able	B	R		

Com.	Description	Process	Read	Write	Cyclic
PFR	**P**rogrammed **F**eed**R**ate	C	R		C
PHD1	**Ph**ysical **D**irectory	B	R		
PPA	**P**art **P**rogram **A**ctive	B	R		C
PPD	**P**art **P**rogram **D**irectory	B	R		
PPN	**P**art.**P**rogram **N**C	B	R	W	A
PPP	**P**art **P**rogram **P**ackage	B			C
PPS	**P**art **P**rogram-**S**equence	C	R		
PSS	**P**rogrammed **S**pindle **S**peed	C	R		C
PTC1	**P**rocess **T**ool Management **C**onfiguration	B	R		C
PTC2	**P**rocess **T**ool Management **C**onfiguration	B	R		C
PVM1	**P**ro**V**i **M**essages	B	R		C
PVM2	**P**ro**V**i **M**essages	B	R		C
PVM3	**P**ro**V**i **M**essages	B	R		C
PVM4	**P**ro**V**i **M**essages	B	R		C
REP1	**REP**ositioning Data	C	R		
REP2	**REP**ositioning Data	C	R		C
SDD1	**S**fc **D**iagnosis **D**ata	B	R		
SDD2	**S**fc **D**iagnosis **D**ata	B	R		
SDD3	**S**fc **D**iagnosis **D**ata	B	R		
SDD4	**S**fc **D**iagnosis **D**ata	B	R		
SDD5	**S**fc **D**iagnosis **D**ata	B	R		
SDD6	**S**fc **D**iagnosis **D**ata	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SFD1	**SF**c **D**ata	B	R		
SFD2	**SF**c **D**ata	B	R		
SFD3	**SF**c **D**ata	B	R		
SFE1	**SF**c **E**rror	B	R		C
SFE2	**SF**c **E**rror	B	R		C
SFM1	**SF**c **M**ode	B	R		C
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SLA1	Actual **S**ervo **LA**g	C	R		C
SLA2	Actual **S**ervo **LA**g	C	R		C
SLI	**S**PS (PLC) **L**ong **I**dentification	B	R		C
SPP	**S**elected **P**art **P**rogram Number	C	R		C
TDA1	**T**ool **DA**ta	B	R	W	
TDA2	**T**ool **DA**ta	B	R		
TDD	**T**ool **D**ata **D**ownload	C		W	
TDR1	**T**ool **D**ata **R**ecord of Place	C	R		C
TDR2	**T**ool **D**ata **R**ecord	C	R		C

Com.	Description	Process	Read	Write	Cyclic
TIF	**T**ool **I**nsert **F**inish	C	R		
TII	**T**ool **I**nsert **I**nitiated	C	R		
TLB1	**T**oo**L** **B**asicdata List	B	R		C
TLB2	**T**oo**L** **B**asicdata List	B	R		C
TLD1	**T**oo**L** **D**ata of Place	C	R	W	C
TLD2	**T**oo**L** **D**ata of Tool	C	R	W	C
TLD3	**T**oo**L** **D**ata of Place	C	R	W	C
TLD4	**T**oo**L** **D**ata of Tool	C	R	W	C
TLE1	**T**oo**L** **E**dgedata List	B	R		C
TLE2	**T**oo**L** **E**dgedata List	B	R		C
TMV	**T**ool **M**o**V**e	C	R		
TPI1	**T**ool **P**osition **I**nformation	B	R		
TPI2	**T**ool **P**osition **I**nformation	B	R		
TQE1	Actual **T**or**Q**u**E**	C	R		C
TQE2	Actual **T**or**Q**u**E**	C	R		C
TRM	**T**ool **R**e**M**ove	C	R		
TRS	**T**ool **R**e**S**et	C	R		
ZOD	**Z**ero **O**ffset **D**ata	C		W	C
ZOD1	**Z**ero **O**ffset **D**ata	C	R		C
ZOD2	**Z**ero **O**ffset Data	C	R		C

Fig. 6-6: Overview of the MTCX device group

Overview of the MWCX Device Group

Com.	Description	Process	Read	Write	Cyclic
AAC1	**A**ctual **AC**celeration	C	R		C
AAD	**A**ctive **A**ngle **D**imension	C	R		C
AAS1	**A**ctual **A**xis **S**peed	C	R		C
AAS2	**A**ctual **A**xis **S**peed	C	R		C
ABI	**A**ctual NC-**B**lock **I**nformation	B	R		C
ACS	**A**ctual **C**utting **S**peed	C	R		C
ADN1	**A**ctive **D**-Correction **N**umber	C	R		C
AEM	**A**ctive **E**vent **M**onitoring	C	R		C
AEN	**A**ctive **E**dge-**N**umber	C	R		C
AFO1	**A**ctive **F**eedrate **O**verride	C	R		C
AFR	**A**ctive **F**eed**R**ate	C	R		C
AGF	**A**ctive **G**-**F**unction	C	R		C
AMF	**A**ctive **M**-**F**unction	C	R		C
AMM1	**A**ctive **M**echanism **M**essage	B	R		C
AMM2	**A**ctive **M**echanism **M**essage	B	R		C
AMM3	**A**ctive **M**echanism **M**essage	B	R		C
AMM4	**A**ctive **M**echanism **M**essage	B	R		C

Com.	Description	Process	Read	Write	Cyclic
AMM5	**A**ctive **M**echanism **M**essage	B	R		C
ANM	**A**ctive **N**C **M**emory Size	C	R		
API1	**A**ctual **P**arameter **I**ndex	B	R		C
API2	**A**ctual **P**arameter **I**ndex	B	R		C
APM	**A**ctive **P**art-Program **M**essage	C	R		C
APN	**A**ctive **P**art-Program Message Number	C	R		C
APO1	**A**ctual Machine **PO**sition	C	R		C
APO2	**A**ctual Machine **PO**sition	C	R		C
APP	**A**ctive **P**art-**P**rogram number	C	R		C
ARF	**A**xis **R**eference **F**lags	C	R		C
ARO1	**A**ctual **R**apid **O**verride	C	R		C
ART	**A**xis **R**eference **T**able	C	R		C
ART	**A**xis **R**eference **T**able	B	R		
ASD	**A**ctual **S**pindel **D**ata	C	R		C
ASF	**A**ctual **S**pindle **F**or Process	C	R		C
ASG	**A**ctual **S**pindle **G**ear	C	R		C
ASM1	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM2	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM3	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM4	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASM5	**A**ctive **S**ystem-Fault **M**essage	B	R		C
ASN	**A**ctual **S**equence **N**umber	C	R		C
ASO1	**A**ctual **S**pindle **O**verride	C	R		C
ASS	**A**ctual **S**pindle **S**peed	C	R		C
ATN	**A**ctive **T**ool-**N**umber	C	R		C
ATP1	**A**ctual **T**ool **P**lace Information	C	R		C
ATP2	**A**ctual **T**ool **P**lace Information	C	R		C
ATP3	**A**ctual **T**ool **P**lace Information	C	R		C
ATR	**A**ctual **T**ool Data **R**ecord	C	R		C
ATU	**A**ctual **T**ool Data **U**pdate	C	R		
AZB1	**A**ctive **Z**ero Offset **B**ank	C	R		C
CCA1	N**C** **C**ycle **A**ccess	B	R	W	
CPO1	**C**ommand **PO**sition (COMMAND)	C	R		C
CPO2	Command POsition by log.Axis No	C	R		C
CRT	**C**ontrol **R**ese**T**	C		W	
DAC1	**D**evice **A**xis **C**onfiguration Parameter	B	R		C
DAC2	**D**evice **A**xis **C**onfiguration Parameter	B	R		C
DCA1	NC **D**-**C**orrection **A**ccess	B	R	W	
DCD1	**D**-**C**orrection **D**ata	C	R		C
DCP1	**D**evice **C**onfiguration **P**arameter	B	R		C

Com.	Description	Process	Read	Write	Cyclic
DCP2	**D**evice **C**onfiguration **P**arameter	B	R		C
DCR1	**D**-**C**orrection **R**ecord	C	R	W	C
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DIS1	**D**ata **I**dentification **S**tring Parameter	C	R		
DIS2	**D**ata **I**dentification **S**tring PLC Program	C	R		
DIS3	**D**ata **I**dentification **S**tring NC Packet	C	R		
DIS4	**D**ata **I**dentification **S**tring Tool List	C	R		
DIS5	**D**ata **I**dentification **S**tring Machine	C	R		
DIS6	**D**ata **I**dentification **S**tring NC Program	C	R		
DPN	**D**elete **P**rogramm **N**C	B		W	
DPP	**D**elete **P**rogram **P**ackage	B		W	
DSI1	**D**evice **S**tatus **I**nformation	B	R		C
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTC1	**D**evice **T**ool Management **C**onfiguration	B	R		C
DTC2	**D**evice **T**ool Management **C**onfiguration	B	R		C
DTG1	**D**istance **T**o **G**o	C	R		C
DTG2	**D**istance **T**o **G**o by log. AxisNo.	C	R		C
DTY1	**D**evice **TY**pe	B	R		
DWD1	**D**iagnosis **W**indow **D**ata	B	R		C
DWD2	**D**iagnosis **W**indow **D**ata	B	R		C
EDE1	**E**xisting **D**iagnosis **E**rror	B	R		C
EDE2	**E**xisting **D**iagnosis **E**rror	B	R		C
EDW1	**E**xisting **D**iagnosis **W**indow	B	R		
EDW2	**E**xisting **D**iagnosis **W**indow	B	R		
EDW3	**E**xisting **D**iagnosis **W**indow	B	R		
END1	**E**xisting **N**C **D**iagnosis	B	R		
END2	**E**xisting **N**C **D**iagnosis	B	R		
EPD1	**E**xisting **P**LC **D**iagnosis	B	R		
EPD2	**E**xisting **P**LC **D**iagnosis	B	R		
EPD3	**E**xisting **P**LC **D**iagnosis	B	R		
EPO1	Programm**E**d **PO**sition (END)	C	R		C
EPO2	Programm**E**d **PO**sition (END)	C	R		C
EPT1	**E**xisting **P**roVi **T**ypes	B	R		
EST1	**E**rror **ST**ate	B	R		C
EXD1	**EX**ecution **D**isplay	B	R		C
EXD2	**EX**ecution **D**isplay	B	R		
GPC1	**G**lobal **P**rocess **C**onfiguration	B	R		C
GPC2	**G**lobal **P**rocess **C**onfiguration	B	R		C
GPP1	**G**lobal **P**rocess **P**arameter	B	R		C

Com.	Description	Process	Read	Write	Cyclic
GPP2	**G**lobal **P**rocess **P**arameter	B	R		C
IPP	**I**nsert NC-**P**rogram **P**ackage	B		W	
MAP1	**M**odule **A**ssign of **P**rocess	B	R		C
MAR	**M**ap **A**bsolute PCL **R**eference	B	R		
MCD1	**M**odule **C**onfiguration: **D**evice Information	B	R		C
MCM1	**M**odule **C**onfiguration: **M**odule Information	B	R		C
MCP1	**M**odule **C**onfiguration: **P**rocess Information	B	R		C
MCS1	**M**odule **C**onfiguration: **S**FC- Information	B	R		C
MDA1	**M**achine **D**ata **A**ccess	B	R	W	
MDA2	**M**achine **D**ata **A**ccess	B	R	W	
MDA4	**M**achine **D**ata **A**ccess	B		W	
MDI	**M**anual **D**ata **I**nput	C		W	
MDS1	**M**achine **D**ata **S**ingle	B	R	W	
MFD1	**M**essage **F**iles **D**ownload	B		W	
MFO1	**M**aximal **F**eedrate **O**verride	C	R		C
MFR	**M**aximal **F**eed**R**ate	C	R		C
MKS	**M**achine **K**ey **S**tatus	B	R		C
MKT1	**M**achine **K**ey **T**able	B		W	
MRO1	**M**aximal **R**apid **O**verride	C	R		C
MSG	**M**e**S**sa**G**e	C			C
MSO1	**M**aximal **S**pindle **O**verride	C	R		C
MSS	**M**aximal **S**pindle **S**peed	C	R		C
MTC	**MT**-**C**NC Slot Software Version	B	R		
MTD	**M**achine **T**able **D**ata	C	R	W	C
NCA1	**NC** Program **A**ccess	B	R	W	
NCA3	**NC** Program **A**ccess	B		W	
NCM1	**NC** **M**essages	B	R		C
NCM2	**NC** **M**essages	B	R		C
NEA1	**N**C **E**vent **A**ccess	B	R	W	
NEV	**N**C **Ev**ent	C	R	W	C
NMM	**N**C **M**e**M**ory selection	C		W	
NPA1	**N**C **PA**rameter	B	R		C
NPA2	**N**C **PA**rameter	B	R		C
NPA3	**N**C **PA**rameter	B	R		C
NPA4	**N**C **PA**rameter	B	R		C
NPA5	**N**C **PA**rameter	B	R		
NPC1	**N**C-**P**ackage **C**ompiling	B	R		
NPD1	**N**C-**P**ackage **D**ownload	B		W	
NPI	**N**C-**P**ackage D**I**rectory	B	R		
NPS	**N**C **P**rogram **S**election	C		W	

Com.	Description	Process	Read	Write	Cyclic
NTN	**N**ext Tool-**N**umber	C	R		C
NUA1	**N**C Offset Data **A**ccess	B	R	W	
NVA1	**N**C **V**ariable **A**ccess	B	R	W	
NVS	**N**C **V**ariable **S**ingle	C	R	W	C
OPD1	**O**ptimal **P**osition **D**istance by Axis sign.	C	R		C
OPD2	**O**ptimal **P**osition **D**istance by phys. AxisNo	C	R		C
PAA1	**PA**rameter **A**ccess	B	R	W	
PAA2	**PA**rameter **A**ccess	B	R	W	
PAC1	**P**rocess **A**xis **C**onfiguration Parameter	B	R		C
PAC2	**P**rocess **A**xis **C**onfiguration Parameter	B	R		C
PAD1	**PA**rameter **D**eactivate	B		W	
PAS1	**PA**rameter **S**et Active	B		W	
PDD1	**P**rovi **D**iagnosis **D**ata	B	R		
PDD2	**P**rovi **D**iagnosis **D**ata	B	R		
PDD3	**P**rovi **D**iagnosis **D**ata	B	R		
PDD4	**P**rovi **D**iagnosis **D**ata	B	R		
PDD5	**P**rovi **D**iagnosis **D**ata	B	R		
PDT	**P**arameter **D**efinition **T**able	B	R		
PFR	**P**rogrammed **F**eed**R**ate	C	R		C
PHD1	**Ph**ysical **D**irectory	B	R		
PPA	**P**art **P**rogram **A**ctive	B	R		C
PPD	**P**art **P**rogram **D**irectory	B	R		
PPN	**P**art.**P**rogram **N**C	B	R	W	A
PPP	**P**art **P**rogram **P**ackage	B			A
PPS	**P**art **P**rogram **S**equence	C	R		
PSM	**P**LC **S**ys **M**essage	B		W	
PSS	**P**rogrammed **S**pindle **S**peed	C	R		C
PTC1	**P**rocess **T**ool Management **C**onfiguration	B	R		C
PTC2	**P**rocess **T**ool Management **C**onfiguration	B	R		C
PVA1	**P**RO**V**I Messages **A**ccess	B	R	W	
PVA2	**P**RO**V**I Messages **A**ccess	B		W	
PVF	**P**LC **V**ariable **Formatted**	C		W	C
PVM1	**P**ro**V**i **M**essages	B	R		C
PVM2	**P**ro**V**i **M**essages	B	R		C
PVM3	**P**ro**V**i **M**essages	B	R		C
PVM4	**P**ro**V**i **M**essages	B	R		C
PVR1	**P**LC **V**ariable **R**etain Backup	B	R	W	
PVT	**P**LC **V**ariable **T**ype	B	R		
REP1	**REP**ositioning Data	C	R		
REP2	**REP**ositioning Data	C	R		C

Com.	Description	Process	Read	Write	Cyclic
SDD1	**S**fc **D**iagnosis **D**ata	B	R		
SDD2	**S**fc **D**iagnosis **D**ata	B	R		
SDD3	**S**fc **D**iagnosis **D**ata	B	R		
SDD4	**S**fc **D**iagnosis **D**ata	B	R		
SDD5	**S**fc **D**iagnosis **D**ata	B	R		
SDD6	**S**fc **D**iagnosis **D**ata	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SFD1	**SF**c **D**ata	B	R		
SFD2	**SF**c **D**ata	B	R		
SFD3	**SF**c **D**ata	B	R		
SFE1	**SF**c **E**rror	B	R		C
SFE2	**SF**c **E**rror	B	R		C
SFM1	**SF**c **M**ode	B	R		C
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SLA1	Actual **S**ervo **LA**g	C	R		C
SLA2	Actual **S**ervo **LA**g	C	R		C
SLI	**S**PS (PLC) **L**ong **I**dentification	B	R		C
SPA1	**S**ercos **PA**rameter	B	R	W	C
SPH	**S**ercos **PH**ase	C	R	W	
SPP	**S**elected **P**art **P**rogram Number	C	R		C
TDA1	**T**ool **DA**ta	B	R	W	
TDA2	**T**ool **DA**ta	B	R		
TDD	**T**ool **D**ata **D**ownload	C		W	
TDR1	**T**ool **D**ata **R**ecord of Place	C	R		C
TDR2	**T**ool **D**ata **R**ecord	C	R		C
TIF	**T**ool **I**nsert **F**inish	C	R		
TII	**T**ool **I**nsert **I**nitiated	C	R		
TLB1	**T**oo**L** **B**asicdata List	B	R		C
TLB2	**T**oo**L** **B**asicdata List	B	R		C
TLD1	**T**oo**L** **D**ata of Place	C	R	W	C
TLD2	**T**oo**L** **D**ata of Tool	C	R	W	C
TLD3	**T**oo**L** **D**ata of Place	C	R	W	C
TLD4	**T**oo**L** **D**ata of Tool	C	R	W	C
TLE1	**T**oo**L** **E**dgedata List	B	R		C
TLE2	**T**oo**L** **E**dgedata List	B	R		C
TMV	**T**ool **M**o**V**e	C	R		
TPI1	**T**ool **P**osition **I**nformation	B	R		
TPI2	**T**ool **P**osition **I**nformation	B	R		
TQE1	Actual **T**or**Q**u**E**	C	R		C

Com.	Description	Process	Read	Write	Cyclic
TQE2	Actual **T**or**Q**u**E**	C	R		C
TRM	**T**ool **R**e**M**ove	C	R		
TRS	**T**ool **R**e**S**et	C	R		
WLA1	**W**atch **L**ist **A**llocation	B	R		
WLF1	**W**atch **L**ist **F**ree	B	R		
WLF2	**W**atch **L**ist **F**ree	B	R		
ZOD	**Z**ero **O**ffset **D**ata	C		W	C
ZOD1	**Z**ero **O**ffset **D**ata	C	R		C
ZOD2	**Z**ero **O**ffset Data	C	R		C

Fig. 6-7: Overview of the MWCX device group

Overview of the MSCX Device Group

Com.	Description	Process	Read	Write	Cyclic
ASE	**A**ctual **S**ystem **E**rror	C	R		C
CSE	**C**lear **S**ystem **E**rror	C		W	
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DSI1	**D**evice **S**tatus **I**nformation	B	R		C
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTY1	**D**evice **TY**pe	B	R		
MSG	**M**e**S**sa**G**e	C			C
PHD1	**Ph**ysical **D**irectory	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SPA1	**S**ercos **PA**rameter	B	R	W	C
SPH	**S**ercos **PH**ase	C	R	W	C

Fig. 6-8: Overview of the MSCX device group

Overview of the MWMX Device Group

Com.	Description	Process	Read	Write	Cyclic
ASM1	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM2	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM3	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM4	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM5	**A**ctive **S**ystem Fault **M**essage	B	R		C
CRT	**C**ontrol **R**ese**T**	C		W	
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DIS2	**D**ata **I**dentification **S**tring PLC Program	C	R		
DSI1	**D**evice **S**tatus **I**nformation	B	R		C

Com.	**Description**	**Process**	**Read**	**Write**	**Cyclic**
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTY1	**D**evice **Ty**pe	B	R		
DWD1	**D**iagnosis **W**indow **D**ata	B	R		C
DWD2	**D**iagnosis **W**indow **D**ata	B	R		C
EDE1	**E**xisting **D**iagnosis **E**rror	B	R		C
EDE2	**E**xisting **D**iagnosis **E**rror	B	R		C
EDW1	**E**xisting **D**iagnosis **W**indow	B	R		
EDW2	**E**xisting **D**iagnosis **W**indow	B	R		
EDW3	**E**xisting **D**iagnosis **W**indow	B	R		
EPD1	**E**xisting **P**LC **D**iagnosis	B	R		
EPD2	**E**xisting **P**LC **D**iagnosis	B	R		
EPD3	**E**xisting **P**LC **D**iagnosis	B	R		
EPT1	**E**xisting **P**roVi **T**ypes	B	R		
EST1	**E**rror **ST**ate	B	R		C
EXD1	**EX**ecution **D**isplay	B	R		C
EXD2	**EX**ecution **D**isplay	B	R		
MAR	**M**ap **A**bsolute PCL **R**eference	B	R		
MCD1	**M**odule **C**onfiguration: **D**evice Information	B	R		C
MCM1	**M**odule **Configuration**: **M**odule Information	B	R		C
MCS1	**M**odule **Configuration**: **S**FC- Information	B	R		C
MFD1	**M**essage **F**iles **D**ownload	B		W	
MKS	**M**achine **K**ey **S**tatus	B	R		C
MKT1	**M**achine **K**ey **T**able	B		W	
MSG	**M**e**S**sa**G**e	C			C
MTC	**MT**-**C**NC Slot Software Version	B	R		
PDD1	**P**rovi **D**iagnosis **D**ata	B	R		
PDD2	**P**rovi **D**iagnosis **D**ata	B	R		
PDD3	**P**rovi **D**iagnosis **D**ata	B	R		
PDD4	**P**rovi **D**iagnosis **D**ata	B	R		
PDD5	**P**rovi **D**iagnosis **D**ata	B	R		
PHD1	**Ph**ysical **D**irectory	B	R		
PSM	**P**LC **S**ys **M**essage	B		W	
PVA1	**P**ROVI Messages **A**ccess	B	R	W	
PVA2	**P**ROVI Messages **A**ccess	B		W	
PVF	**P**LC **V**ariable **Formatted**	C		W	C
PVM1	**P**ro**V**i **M**essages	B	R		C
PVM2	**P**ro**V**i **M**essages	B	R		C
PVM3	**P**ro**V**i **M**essages	B	R		C
PVM4	**P**ro**V**i **M**essages	B	R		C
PVR1	**P**LC **V**ariable **R**etain Backup	B	R	W	

Com.	Description	Process	Read	Write	Cyclic
PVT	**P**LC **V**ariable **T**ype	B	R		
SDD1	**S**fc **D**iagnosis **D**ata	B	R		
SDD2	**S**fc **D**iagnosis **D**ata	B	R		
SDD3	**S**fc **D**iagnosis **D**ata	B	R		
SDD4	**S**fc **D**iagnosis **D**ata	B	R		
SDD5	**S**fc **D**iagnosis **D**ata	B	R		
SDD6	**S**fc **D**iagnosis **D**ata	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SFD1	**SF**c **D**ata	B	R		
SFD2	**SF**c **D**ata	B	R		
SFD3	**SF**c **D**ata	B	R		
SFE1	**SF**c **E**rror	B	R		C
SFE2	**SF**c **E**rror	B	R		C
SFM1	**SF**c **M**ode	B	R		C
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SLI	**S**PS (PLC) **L**ong **I**dentification	B	R		C
WLA1	**W**atch **L**ist **A**llocation	B	R		
WLF1	**W**atch **L**ist **F**ree	B	R		
WLF2	**W**atch **L**ist **F**ree	B	R		

Fig. 6-9: Overview of the MWMX device group

Overview of the MWSX Device Group

Com.	Description	Process	Read	Write	Cyclic
ASM1	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM2	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM3	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM4	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM5	**A**ctive **S**ystem Fault **M**essage	B	R		C
CRT	**C**ontrol **R**ese**T**	C		W	
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DIS2	**D**ata **I**dentification **S**tring PLC Program	C	R		
DSI1	**D**evice **S**tatus **I**nformation	B	R		C
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTY1	**D**evice **Ty**pe	B	R		
DWD1	**D**iagnosis **W**indow **D**ata	B	R		C
DWD2	**D**iagnosis **W**indow **D**ata	B	R		C
EDE1	**E**xisting **D**iagnosis **E**rror	B	R		C
EDE2	**E**xisting **D**iagnosis **E**rror	B	R		C

Com.	Description	Process	Read	Write	Cyclic
EDW1	**E**xisting **D**iagnosis **W**indow	B	R		
EDW2	**E**xisting **D**iagnosis **W**indow	B	R		
EDW3	**E**xisting **D**iagnosis **W**indow	B	R		
EPD1	**E**xisting **P**LC **D**iagnosis	B	R		
EPD2	**E**xisting **P**LC **D**iagnosis	B	R		
EPD3	**E**xisting **P**LC **D**iagnosis	B	R		
EPT1	**E**xisting **P**roVi **T**ypes	B	R		
EST1	**E**rror **ST**ate	B	R		C
EXD1	**EX**ecution **D**isplay	B	R		C
EXD2	**EX**ecution **D**isplay	B	R		
MAR	**M**ap **A**bsolute PCL **R**eference	B	R		
MCD1	**M**odule **C**onfiguration: **D**evice Information	B	R		C
MCM1	**M**odule **Configuration**: **M**odule Information	B	R		C
MCS1	**M**odule **Configuration**: **S**FC- Information	B	R		C
MFD1	**M**essage **F**iles **D**ownload	B		W	
MKS	**M**achine **K**ey **S**tatus	B	R		C
MKT1	**M**achine **K**ey **T**able	B		W	
MSG	**M**e**S**sa**G**e	C			C
MTC	**MT**-**C**NC Slot Software Version	B	R		
PDD1	**P**rovi **D**iagnosis **D**ata	B	R		
PDD2	**P**rovi **D**iagnosis **D**ata	B	R		
PDD3	**P**rovi **D**iagnosis **D**ata	B	R		
PDD4	**P**rovi **D**iagnosis **D**ata	B	R		
PDD5	**P**rovi **D**iagnosis **D**ata	B	R		
PHD1	**Ph**ysical **D**irectory	B	R		
PSM	**P**LC **S**ys **M**essage	B		W	
PVA1	**P**ROV**I** Messages **A**ccess	B	R	W	
PVA2	**P**ROV**I** Messages **A**ccess	B		W	
PVF	**P**LC **V**ariable **Formatted**	C		W	C
PVM1	**P**ro**V**i **M**essages	B	R		C
PVM2	**P**ro**V**i **M**essages	B	R		C
PVM3	**P**ro**V**i **M**essages	B	R		C
PVM4	**P**ro**V**i **M**essages	B	R		C
PVR1	**P**LC **V**ariable **R**etain Backup	B	R	W	
PVT	**P**LC **V**ariable **T**ype	B	R		
SDD1	**S**fc **D**iagnosis **D**ata	B	R		
SDD2	**S**fc **D**iagnosis **D**ata	B	R		
SDD3	**S**fc **D**iagnosis **D**ata	B	R		
SDD4	**S**fc **D**iagnosis **D**ata	B	R		
SDD5	**S**fc **D**iagnosis **D**ata	B	R		

Com.	Description	Process	Read	Write	Cyclic
SDD6	**S**fc **D**iagnosis **D**ata	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SFD1	**SF**c **D**ata	B	R		
SFD2	**SF**c **D**ata	B	R		
SFD3	**SF**c **D**ata	B	R		
SFE1	**SF**c **E**rror	B	R		C
SFE2	**SF**c **E**rror	B	R		C
SFM1	**SF**c **M**ode	B	R		C
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SLI	**S**PS (PLC) **L**ong **I**dentification	B	R		C
WLA1	**W**atch **L**ist **A**llocation	B	R		
WLF1	**W**atch **L**ist **F**ree	B	R		
WLF2	**W**atch **L**ist **F**ree	B	R		

Fig. 6-10: Overview of MWSX device group

Overview of the MWAX Device Group

Com.	Description	Process	Read	Write	Cyclic
ADM1	**MTA200 M**essages	B	R		C
ADM2	**MTA200 M**essages	B	R		C
ADM3	**MTA200 M**essages	B	R		
AMM7	**A**ctive **M**echanism **M**essage	B	R		C
APO2	**A**ctual Machine **PO**sition	C	R		C
ASM1	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM2	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM3	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM4	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM5	**A**ctive **S**ystem Fault **M**essage	B	R		C
CMA	**CM**OS RAM **A**SCII Parameter	C	R	W	
CMF	**CM**OS RAM **F**loating point Parameter	C	R	W	
CMI	**CM**OS RAM **I**nteger Parameter	C	R	W	
CRT	**C**ontrol **R**ese**T**	C		W	
DCP1	**D**evice **C**onfiguration **P**arameter	B	R		C
DCP2	**D**evice **C**onfiguration **P**arameter	B	R		C
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DIS1	**D**ata **I**dentification **S**tring Parameter	C	R		
DIS2	**D**ata **I**dentification **S**tring PLC Program	C	R		
DSI1	**D**evice **S**tatus **I**nformation	B	R		C
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTG2	**D**istance **T**o **G**o by log. AxisNo.	C	R		C

Com.	Description	Process	Read	Write	Cyclic
DTY1	**D**evice **Ty**pe	B	R		
DWD1	**D**iagnosis **W**indow **D**ata	B	R		C
DWD2	**D**iagnosis **W**indow **D**ata	B	R		C
EAD1	**E**xisting **MTA200** **D**iagnosis	B	R		
EAD2	**E**xisting **MTA200** **D**iagnosis	B	R		
EDE1	**E**xisting **D**iagnosis **E**rror	B	R		C
EDE2	**E**xisting **D**iagnosis **E**rror	B	R		C
EDW1	**E**xisting **D**iagnosis **W**indow	B	R		
EDW2	**E**xisting **D**iagnosis **W**indow	B	R		
EDW3	**E**xisting **D**iagnosis **W**indow	B	R		
EPT1	**E**xisting **P**roVi **T**ypes	B	R		
EST1	**E**rror **ST**ate	B	R		C
EXD1	**EX**ecution **D**isplay	B	R		C
EXD2	**EX**ecution **D**isplay	B	R		
MAP1	**M**odule **A**ssign of **P**rocess	B	R		C
MAR	**M**ap **A**bsolute PCL **R**eference	B	R		
MCD1	**M**odule **Configuration**: **D**evice Information	B	R		C
MCM1	**M**odule **Configuration**: **M**odule Information	B	R		C
MCP1	**M**odule **Configuration**: **P**rocess Information	B	R		C
MCS1	**M**odule **C**onfiguration: **S**FC- Information	B	R		C
MFD1	**M**essage **F**iles **D**ownload	B		W	
MKS	**M**achine **K**ey **S**tatus	B	R		C
MKT1	**M**achine **K**ey **T**able	B		W	
MSG	**M**e**S**sa**G**e	C			C
MTC	**MT**-**C**NC Slot Software Version	B	R		
PAA2	**PA**rameter **A**ccess	B	R	W	
PDD1	**P**rovi **D**iagnosis **D**ata	B	R		
PDD2	**P**rovi **D**iagnosis **D**ata	B	R		
PDD3	**P**rovi **D**iagnosis **D**ata	B	R		
PDD4	**P**rovi **D**iagnosis **D**ata	B	R		
PDD5	**P**rovi **D**iagnosis **D**ata	B	R		
PHD1	**Ph**ysical **D**irectory	B	R		
POI	**PO**sition **I**nformation	B	R		C
PSM	**P**LC **S**ys **M**essage	B		W	
PVA1	**P**ROV**I** Messages **A**ccess	B	R	W	
PVA2	**P**ROV**I** Messages **A**ccess	B		W	
PVF	**P**LC **V**ariable **Formatted**	C		W	C
PVM1	**P**ro**V**i **M**essages	B	R		C
PVM2	**P**ro**V**i **M**essages	B	R		C
PVM3	**P**ro**V**i **M**essages	B	R		C

Rexroth Indramat

Com.	Description	Process	Read	Write	Cyclic
PVM4	**P**ro**V**i **M**essages	B	R		C
PVR1	**P**LC **V**ariable **R**etain Backup	B	R	W	
PVT	**P**LC **V**ariable **T**ype	B	R		
SDD1	**S**fc **D**iagnosis **D**ata	B	R		
SDD2	**S**fc **D**iagnosis **D**ata	B	R		
SDD3	**S**fc **D**iagnosis **D**ata	B	R		
SDD4	**S**fc **D**iagnosis **D**ata	B	R		
SDD5	**S**fc **D**iagnosis **D**ata	B	R		
SDD6	**S**fc **D**iagnosis **D**ata	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SFD1	**SF**c **D**ata	B	R		
SFD2	**SF**c **D**ata	B	R		
SFD3	**SF**c **D**ata	B	R		
SFE1	**SF**c **E**rror	B	R		C
SFE2	**SF**c **E**rror	B	R		C
SFM1	**SF**c **M**ode	B	R		C
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SLI	**S**PS (PLC) **L**ong **I**dentification	B	R		C
WLA1	**W**atch **L**ist **A**llocation	B	R		
WLF1	**W**atch **L**ist **F**ree	B	R		
WLF2	**W**atch **L**ist **F**ree	B	R		

Fig. 6-11: Overview of the MWAX device group

Overview of the MSYX Device Group

Com.	Description	Process	Read	Write	Cyclic
ASE	**A**ctual **S**ystem **E**rror	C	R		C
CSE	**C**lear **S**ystem **E**rror	C		W	
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DTY	**D**evice **TY**pe	B	R		
MSG	**M**e**S**sa**G**e	C			C
PHD1	**Ph**ysical **D**irectory	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SPA1	**S**ercos **PA**rameter	B	R	W	C
SPH	**S**ercos **PH**ase	C	R	W	C

Fig. 6-12: Overview of the MSYX device group

Overview of the MWYX Device Group

Com.	Description	Process	Read	Write	Cyclic
ASE	**A**ctual **S**ystem **E**rror	C	R		C
ASM1	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM2	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM3	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM4	**A**ctive **S**ystem Fault **M**essage	B	R		C
ASM5	**A**ctive **S**ystem Fault **M**essage	B	R		C
CRT	**C**ontrol **R**ese**T**	C		W	
CSE	**C**lear **S**ystem **E**rror	C		W	
DCT1	**D**evice **C**ommunication **T**imeout	B		W	
DCT2	**D**evice **C**ommunication **T**imeout	B		W	
DIS2	**D**ata **I**dentification **S**tring PLC Program	C	R		
DSI1	**D**evice **S**tatus **I**nformation	B	R		C
DSI2	**D**evice **S**tatus **I**nformation	B	R		C
DTY1	**D**evice **Ty**pe	B	R		
DWD1	**D**iagnosis **W**indow **D**ata	B	R		C
DWD2	**D**iagnosis **W**indow **D**ata	B	R		C
EDE1	**E**xisting **D**iagnosis **E**rror	B	R		C
EDE2	**E**xisting **D**iagnosis **E**rror	B	R		C
EDW1	**E**xisting **D**iagnosis **W**indow	B	R		
EDW2	**E**xisting **D**iagnosis **W**indow	B	R		
EDW3	**E**xisting **D**iagnosis **W**indow	B	R		
EPD1	**E**xisting **P**LC **D**iagnosis	B	R		
EPD2	**E**xisting **P**LC **D**iagnosis	B	R		
EPD3	**E**xisting **P**LC **D**iagnosis	B	R		
EPT1	**E**xisting **P**roVi **T**ypes	B	R		
EST1	**E**rror **ST**ate	B	R		C
EXD1	**EX**ecution **D**isplay	B	R		C
EXD2	**EX**ecution **D**isplay	B	R		
MAR	**M**ap **A**bsolute PCL **R**eference	B	R		
MCD1	**M**odule **C**onfiguration: **D**evice Information	B	R		C
MCM1	**M**odule **C**onfiguration: **M**odule Information	B	R		C
MCS1	**M**odule **C**onfiguration: **S**FC- Information	B	R		C
MFD1	**M**essage **F**iles **D**ownload	B		W	
MKS	**M**achine **K**ey **S**tatus	B	R		C
MKT1	**M**achine **K**ey **T**able	B		W	
MSG	**M**e**S**sa**G**e	C			C
MTC	**MT**-**C**NC Slot Software Version	B	R		

Com.	Description	Process	Read	Write	Cyclic
PDD1	**Pro**vi **D**iagnosis **D**ata	B	R		
PDD2	**Pro**vi **D**iagnosis **D**ata	B	R		
PDD3	**Pro**vi **D**iagnosis **D**ata	B	R		
PDD4	**Pro**vi **D**iagnosis **D**ata	B	R		
PDD5	**Pro**vi **D**iagnosis **D**ata	B	R		
PHD1	**Ph**ysical **D**irectory	B	R		
PSM	**P**LC **S**ys **M**essage	B		W	
PVA1	**P**ROVI Messages **A**ccess	B	R	W	
PVA2	**P**ROVI Messages **A**ccess	B		W	
PVF	**P**LC **V**ariable **Formatted**	C		W	C
PVM1	**P**ro**V**i **M**essages	B	R		C
PVM2	**P**ro**V**i **M**essages	B	R		C
PVM3	**P**ro**V**i **M**essages	B	R		C
PVM4	**P**ro**V**i **M**essages	B	R		C
PVR1	**P**LC **V**ariable **R**etain Backup	B	R	W	
PVT	**P**LC **V**ariable **T**ype	B	R		
SDD1	**S**fc **D**iagnosis **D**ata	B	R		
SDD2	**S**fc **D**iagnosis **D**ata	B	R		
SDD3	**S**fc **D**iagnosis **D**ata	B	R		
SDD4	**S**fc **D**iagnosis **D**ata	B	R		
SDD5	**S**fc **D**iagnosis **D**ata	B	R		
SDD6	**S**fc **D**iagnosis **D**ata	B	R		
SDS1	**S**et **D**evice **S**tatus	B		W	
SDS2	**S**et **D**evice **S**tatus	B		W	
SFD1	**SF**c **D**ata	B	R		
SFD2	**SF**c **D**ata	B	R		
SFD3	**SF**c **D**ata	B	R		
SFE1	**SF**c **E**rror	B	R		C
SFE2	**SF**c **E**rror	B	R		C
SFM1	**SF**c **M**ode	B	R		C
SID1	**S**oftware **I**nstallation **D**ata	B	R		C
SLI	**S**PS (PLC) **L**ong **I**dentification	B	R		C
SPA1	**S**ercos **PA**rameter	B	R	W	C
SPH	**S**ercos **PH**ase	C	R	W	C
WLA1	**W**atch **L**ist **A**llocation	B	R		
WLF1	**W**atch **L**ist **F**ree	B	R		
WLF2	**W**atch **L**ist **F**ree	B	R		

Fig. 6-13: Overview of the MWYX device group

Logical Connection between FI Commands

In the following table, all FI commands are grouped in a logical order.

Group	Device group	FI Commands
Axes	MTCX	AAD, AAS1, AAS2, ARF, ART, CPO1, CPO2, DTG1, DTG2, EPO1, EPO2, OPD1, OPD2, PAC1, PAC2, REP1, REP2, SLA1, SLA2, TQE1, TQE2
	MWCX	AAD, AAS1, AAS2, ARF, ART, CPO1, CPO2, DTG1, DTG2, EPO1, EPO2, OPD1, OPD2, PAC1, PAC2, REP1, REP2, SLA1, SLA2, TQE1, TQE2
	MWAX	DTG2
Axis Parameters	MTCX	PAA1, PAA2, PAD1, PAS1
	MWCX	PAA1, PAA2, PAD1, PAS1
	MWAX	PAA2
D-correction	MTCX	ADN1, DCA1, DCD1, DCR1
	MWCX	ADN1, DCA1, DCD1, DCR1
Diagnosis	MTCX	DWD1, DWD2, EDE1, EDE2, EDW1, EDW2, EDW3, END1, END2
	MWCX	DWD1, DWD2, EDE1, EDE2, EDW1, EDW2, EDW3, END1, END2, EPD1, EPD2, EPD3, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6
	MWMX	DWD1, DWD2, EDE1, EDE2, EDW1, EDW2, EDW3, EPD1, EPD2, EPD3, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6
	MWSX	DWD1, DWD2, EDE1, EDE2, EDW1, EDW2, EDW3, EPD1, EPD2, EPD3, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6
	MWAX	DWD1, DWD2, EAD1, EAD2, EDE1, EDE2, EDW1, EDW2, EDW3, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6
	MWYX	DWD1, DWD2, EDE1, EDE2, EDW1, EDW2, EDW3, EPD1, EPD2, EPD3, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6
Download/ Upload	MTCX	CCA1, DCA1, MDA1, MDA2, MDA4, MFD1, NCA1, NEA1, NUA1, NVA1, PAA1, PAA2
	MWCX	CCA1, DCA1, MDA1, MDA2, MFD1, NCA1, NEA1, NUA1, NVA1, PAA1, PAA2, PVR1
	MWMX	MFD1, PVR1
	MWSX	MFD1, PVR1
	MWAX	MFD1, PAA2, PVR1
	MWYX	MFD1, PVR1
Event	MTCX	AEM, NEA1, NEV
	MWCX	AEM, NEA1, NEV
Device	MPCX	CCP1, CCP2, CCP3, CCP4, CCP5, FCP1, FCP2, FCP3, FDC1
	MTCX	DCP1, DCP2, DSI1, DSI2, DTC1, DTY1, MCD1, SDS1, SDS2
	MWCX	DCP1, DCP2, DSI1, DSI2, DTC1, DTY1, MCD1, SDS1, SDS2
	MSCX	DSI1, DSI2, DTY1, SDS1, SDS2
	MWMX	DSI1, DSI2, DTY1, MCD1, SDS1, SDS2
	MWSX	DSI1, DSI2, DTY1, MCD1, SDS1, SDS2
	MWAX	DCP1, DCP2, DSI1, DSI2, DTY1, MCD1, SDS1, SDS2
	MWYX	DSI1, DSI2, DTY1, MCD1, SDS1, SDS2

Rexroth Indramat

Group	Device group	FI Commands
Configuration	MPCX	CCP1, CCP2, CCP3, CCP4, CCP5, FCP1, FCP2, FCP3, FDC1
	MTCX	DAC1, DAC2, DCP1, DCP2, DTY1, GPC1, GPC2, PAC1, PAC2, PTC1, PTC2
	MWCX	DAC1, DAC2, DCP1, DCP2, DTY1, GPC1, GPC2, PAC1, PAC2, PTC1, PTC2
	MSCX	DTY1
	MWMX	DTY1
	MWSX	DTY1
	MWAX	DCP1, DCP2, DTY1
	MWYX	DTY1
Machine Data	MTCX	DIS5, MDA1, MDA2, MDA4, MDS1, MKS, MTD
	MWCX	DIS5, MDA1, MDA2, MDA4, MDS2, MKS, MTD
Messages	MPCX	ERI1, FIT1, MSG, SSM1, SSM2
	MTCX	AMM1, AMM2, AMM3, AMM4, AMM5, ASM1, ASM2, ASM3, ASM4, ASM5, MSG, NCM1, NCM2, PVM1, PVM2, PVM3, PVM4, SLI
	MWCX	AMM1, AMM2, AMM3, AMM4, AMM5, ASM1, ASM2, ASM3, ASM4, ASM5, MSG, NCM1, NCM2, PDD1, PDD2, PDD3, PDD4, PDD5, PSM1, PVM1, PVM2, PVM3, PVM4, SLI
	MSCX	ASE, CSE, MSG
	MWMX	ASM1, ASM2, ASM3, ASM4, ASM5, MSG, PDD1, PDD2, PDD3, PDD4, PDD5, PSM1, PVM1, PVM2, PVM3, PVM4, SLI
	MWSX	ASM1, ASM2, ASM3, ASM4, ASM5, MSG, PDD1, PDD2, PDD3, PDD4, PDD5, PSM1, PVM1, PVM2, PVM3, PVM4, SLI
	MWAX	ADM1, ADM2, ADM3, AMM7, ASM1, ASM2, ASM3, ASM4, ASM5, MSG, PDD1, PDD2, PDD3, PDD4, PDD5, PSM1, PVM1, PVM2, PVM3, PVM4, SLI
	MWYX	ASM1, ASM2, ASM3, ASM4, ASM5, MSG, PDD1, PDD2, PDD3, PDD4, PDD5, PSM1, PVM1, PVM2, PVM3, PVM4, SLI
Modules	MTCX	MAP1, MCD1, MCM1, MCP1, MCS1
	MWCX	MAP1, MCD1, MCM1, MCP1, MCS1
	MWMX	MCD1, MCM1, MCS1
	MWSX	MCD1, MCM1, MCS1
	MWAX	MAP1, MCD1, MCM1, MCP1, MCS1
	MWYX	MCD1, MCM1, MCS1
NC processing	MTCX	ABI, AGF, AMF, ANM, APM, APN, APP, ASN, CCA1, DCA1, DIS1, DIS2, DIS3, DIS4, DIS5, DIS6, DPN, DPP, IPP, MDI, NCA1, NCA3, NCM1, NCM2, NEA1, NEV, NMM, NPA1, NPA2, NPA3, NPA4, NPC1, NPD1, NPI, NPS, NUA1, NVA1, NVS, PPA, PPD, PPN, PPP, PPS, SPP
	MWCX	ABI, AGF, AMF, ANM, APM, APN, APP, ASN, CCA1, DCA1, DIS1, DIS2, DIS3, DIS4, DIS5, DIS6, DPN, DPP, IPP, MDI, NCA1, NCA3, NCM1, NCM2, NEA1, NEV, NMM, NPA1, NPA2, NPA3, NPA4, NPC1, NPD1, NPI, NPS, NUA1, NVA1, NVS, PPA, PPD, PPN, PPP, PPS, SPP
Override	MTCX	AFO1, ARO1, ASO1, MFO1, MRO1, MSO1
	MWCX	AFO1, ARO1, ASO1, MFO1, MRO1, MSO1

Group	Device group	FI Commands
Position	MTCX	APO1, APO2, CPO1, CPO2, DTG1, DTG2, EPO1, EPO2, OPD1, OPD2, REP1, REP2, SLA1, SLA2
	MWCX	APO1, APO2, CPO1, CPO2, DTG1, DTG2, EPO1, EPO2, OPD1, OPD2, REP1, REP2, SLA1, SLA2
	MWAX	APO2, DTG2, POI1
Process	MPCX	CPR1, CPR2, DPR1, NST1, NST2, RPR1
	MTCX	GPC1, GPC2, GPP1, GPP2, MAP1, MCP1, PAC1, PAC2, PTC1, PTC2
	MWCX	GPC1, GPC2, GPP1, GPP2, MAP1, MCP1, PAC1, PAC2, PTC1, PTC2
	MWAX	MAP1, MCP1
Cutters	MTCX	AEN, TLE1, TLE2
	MWCX	AEN, TLE1, TLE2
Sercos	MWCX	SPA1, SPH
	MSCX	SPA1, SPH
	MSYX	SPA1, SPH
	MWYX	SPA1, SPH
spindle	MTCX	AAD, AAS1, AAS2, ACS, ASD, ASF, ASG, ASO1, ASS, MSO1, MSS, PSS
	MWCX	AAD, AAS1, AAS2, ACS, ASD, ASF, ASG, ASO1, ASS, MSO1, MSS, PSS
PLC	MWCX	DIS2, EDE1, EDE2, EPD1, EPD2, EPD3, EPT1, EST1, EXD1, EXD2, MAR, MKT1, PVF, PVT, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6, SFD1, SFD2, SFE1, SFE2, SFM1, SLI
	MWMX	DIS2, EDE1, EDE2, EPD1, EPD2, EPD3, EPT1, EST1, EXD1, EXD2, MAR, MKT1, PVA1, PVA2, PVF, PVT, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6, SFD1, SFD2, SFE1, SFE2, SFM1, SLI
	MWSX	DIS2, EDE1, EDE2, EPD1, EPD2, EPD3, EPT1, EST1, EXD1, EXD2, MAR, MKT1, PVA1, PVA2, PVF, PVT, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6, SFD1, SFD2, SFE1, SFE2, SFM1, SLI
	MWAX	DIS2, EDE1, EDE2, EST1, EPT1, EXD1, EXD2, MAR, MKT1, PVA1, PVA2, PVF, PVT, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6, SFD1, SFD2, SFE1, SFE2, SFM1, SLI
	MWYX	DIS2, EDE1, EDE2, EPD1, EPD2, EPD3, EPT1, EST1, EXD1, EXD2, MAR, MKT1, PVA1, PVA2, PVF, PVT, SDD1, SDD2, SDD3, SDD4, SDD5, SDD6, SFD1, SFD2, SFE1, SFE2, SFM1, SLI
Feed	MTCX	AAC1, AAD, AAS1, AAS2, ACS, ADN1, AFO1, AFR, ARO1, AZB1, CPO1, CPO2, DCD1, DCR1, DTG1, DTG2, MFO1, MFR, MRO1, OPD1, OPD2, PFR, PSS, REP1, REP2, SLA1, SLA2, TQE1, TQE2, ZOD, ZOD1, ZOD2
	MWCX	AAC1, AAD, AAS1, AAS2, ACS, ADN1, AFO1, AFR, ARO1, AZB1, CPO1, CPO2, DCD1, DCR1, DTG1, DTG2, MFO1, MFR, MRO1, OPD1, OPD2, PFR, PSS, REP1, REP2, SLA1, SLA2, TQE1, TQE2, ZOD, ZOD1, ZOD2
	MWAX	DTG2
Tool	MTCX	AEN, ATN, ATP1, ATP2, ATP3, ATR, ATU, DIS4, DTC1, DTC2, NTN, PTC1, PTC2, TDA1, TDA2, TDR1, TDR2, TIF, TII, TLB1, TLB2, TLD1, TLD2, TLD3, TLD4, TLE1, TLE2, TMV, TPI1, TPI2, TRM, TRS
	MWCX	AEN, ATN, ATP1, ATP2, ATP3, ATR, ATU, DIS4, DTC1, DTC2, NTN, PTC1, PTC2, TDA1, TDA2, TDR1, TDR2, TIF, TII, TLB1, TLB2, TLD1, TLD2, TLD3, TLD4, TLE1, TLE2, TMV, TPI1, TPI2, TRM, TRS

Fig. 6-14: Logical conjunctions of FI commands

Rexroth Indramat

6.4 Command Execution Times

Legends for the Command Execution Times

The command execution times determined are typical measured values. Their capacity for reproduction depends on many factors. Among these factors are the type of computer used (processor, memory, etc.), the existing device configuration, (device; communication port DPR, V24 etc.) as well as the more or less heavy load caused by processes running in parallel.

The measured values determined are subject to a rasterization of 10ms. In principle, therefore, a tolerance of +/- 10ms should be assumed. In addition, sporadic measured values will be determined that lie outside this tolerance range. It cannot therefore be based on deterministic behavior.

The execution times determined do, however, help you to get a feeling for the processing times of the commands. You can therefore try numerous ways of accessing the device “at your desk” and find the best means of access.

For better comparison, the specifications of the PC and device configuration with which the command execution times have been determined are listed below.

Computer Type

The type of computer with which the following measured values have been determined has the following specifications:

Processor	RAM	Operating System
Pentium 166 MHz	32 MByte	Windows NT 4.0

Fig. 6-15: Computer identification data

Device Configuration

To determine the command execution times, a representative device was selected from each device class and the complete range of commands for the device was tested. The communication port used between the PC and the device is of critical importance. As should be expected, access via the DPR involves shorter execution times when compared to access via the serial interface. This should be taken into account when comparing the command execution times.

Refer to the following table for the respective representative devices of the device classes; the execution times have not been determined for each device family.

	Device	PLC Components	NC Components	Communication configuration
MPCX	PC	None	None	None
MWCS	MTC200-P-G2 MTC200-R-G2	MTS-P	MTC-P	DPR, TCON
MSCX	SERCANS-A, SERCANS-P	None	None	V24 19200 Baud TCON
MWMX	VMISP200-P-G2, VMISP200-R-G2	MTS-P02.2	None	DPR, TCON
MWSX	ISP200-P-G2, ISP200-R-G2	MTS-P02.2	None	DPR, TCON
MWAX	MTA200-P	MTS-P	None	SHM
MSYX	SYNAX200-P, SYNAX200-R	None	None	DPR, TCON
MWYX	SYNAXISP200-P-G2, SYNAXISP200-R-G2	MTS-P02.2	None	DPR, TCON

Fig. 6-16: Representative devices

Parallel Processes

The following processes are running while the command execution times are running:

- The application used in determining the times.
- The processes belonging to the function interface, i.e. COMINTFC.EXE, LOGINTFC.EXE, BOFINTFC.EXE.
- The MTA200.EXE process as communication driver to the MTA200-P.
- The MTVNC40V.EXE process as communication driver to the virtual MTC-200.
- The NETINTFC.EXE process as communication driver to the PC network.

Note: *1) The command marked is a job command.
The time given refers to the start of the job. The time it takes for the job to work in the background must be added to the complete time of command execution.
*2) For weighting the command execution time, the note is of decisive importance.

Command execution times for the MPCX device group

Com.	Description	Example	[ms]
CCP1	Cell Configuration Parameter	XX_BR_CCP1	30
CCP2	Cell Configuration Parameter	XX_BR_CCP2_MTC200-P	20
CCP3	Cell Configuration Parameter	XX_BR_CCP3_1	10
CCP4	Cell Configuration Parameter	XX_BR_CCP4_MWCX	20
CCP5	Cell Configuration Parameter	XX_BR_CCP5_02	20
DFJ1	Delete Function Interface Job	XX_BR_DFJ1	10
DFJ2	Delete Function Interface Job	XX_BR_DFJ2_1	20
FCP1	Far Device Configuration Parameter	XX_BR_FCP1	10
FCP2	Far Device Configuration Parameter	XX_BR_FCP2_MWCX	10
FCP3	Far Device Configuration Parameter	XX_BR_FCP3_MTC200	10
FDC1	Far Device Configuration	XX_BR_FDC1	20
FIT1	Further Info Text	XX_BR_FIT1_1_5	20
FPC1	Far PC Configuration	XX_BR_FPC1	10
IFJ1	Information about Function Interface Jobs	XX_BR_IFJ1	10
IFJ2	Information about Function Interface Jobs	XX_BR_IFJ2_1	10
LNG	Active LaNGuage	XX_BR_LNG	10

Fig. 6-17: Command execution times of the MPCX device group

Command execution times for the MWCX device group

Com.	Description	Example	[ms]
AAC1	Actual Acceleration	00_CR_AAC1_0	20
AAD	Active Angle Dimension	00_CR_AAD_0	20
AAS1	Actual Axis Speed	00_CR_AAS1_0_1	20
AAS2	Actual Axis Speed	00_CR_AAS2_2	20
ABI	Actual NC Block Information	00_BR_ABI_0	40
ABN	Active Conditional Banner No.	00_BR_ABN_0	30
ACS	Actual Cutting Speed	00_CR_ACS_0	20
ADN1	Active D-Correction Number	00_CR_ADN1_0	20
AEM	Active Event Monitoring	00_CR_AEM_0	20
AEN	Active Edge Number	00_CR_AEN_0	20
AFO1	Active Feedrate Override	00_CR_AFO1_0	20
AFR	Active FeedRate	00_CR_AFR_0	20
AGF	Active G-Function	00_CR_AGF_0	20
AMF	Active M-Function	00_CR_AMF_0	20
AMM1	Active Mechanism Message	00_BR_AMM1	100
AMM2	Active Mechanism Message	00_BR_AMM2	30
AMM3	Active Mechanism Message	00_BR_AMM3_0	70
AMM4	Active Mechanism Message	00_BR_AMM4_02.0	70
AMM5	Active Mechanism Message	00_BR_AMM5_0_69_0	40
API1	Actual Parameter Index	00_BR_API1	100
API2	Actual Parameter Index	00_BR_API2	60
APM	Active Part Program Message	00_CR_APM_0	20
APN	Active Part Program Message Number	00_CR_APN_0	20
APO	Actual Machine POsition	00_CR_APO_0_2_1	20
APO1	Actual Machine POsition	00_CR_APO1_0_2_1	20
APO2	Actual Machine POsition	00_CR_APO2_3_1	20
APP	Active Part Program Number	00_CR_APP_0	20
ARO1	Actual Rapid Override	00_CR_AFO1_0	20
ASF	Actual Spindle For Process	00_CR_ASF_0	20
ASG	Actual Spindle Gear	00_CR_ASG_0_1	20
ASM1	Active System Fault Message	00_BR_ASM1	60
ASM2	Active System Fault Message	00_BR_ASM2	30
ASM3	Active System Fault Message	00_BR_ASM3_02	30
ASM4	Active System Fault Message	00_BR_ASM4_MWCX	50
ASM5	Active System Fault Message	00_BR_ASM5_74_0	30
ASN	Actual Sequence Number	00_CR_ASN_0	20
ASO1	Actual Spindle Override	00_CR_ASO1_0_1	20
ASS	Actual Spindle Speed	00_CR_ASS_0_1	20
ATN	Active Tool Number	00_CR_ATN_0	20

Com.	Description	Example	[ms]
ATP1	Actual Tool Place Information	00_CR_ATP1_0	20
ATP2	Actual Tool Place Information	00_CR_ATP2_0	20
ATP3	Actual Tool Place Information	00_CR_ATP3_0	20
AZB1	Active Zero Offset Bank	00_CR_AZB1_0	20
CPO1	Command POsition (COMMAND)	00_CR_CPO1_0_2_1	20
CPO2	Command POsition by log.Axis No	00_CR_CPO2_3_1	20
CRT	Control ReseT		20
DAC1	Device Axis Configuration Parameter	00_BR_DAC1	20
DAC2	Device Axis Configuration Parameter	00_BR_DAC2_1	20
DCD1	D-Correction Data	00_CR_DCD1_0_1_1	20
DCP1	Device Configuration Parameter	00_BR_DCP1	30
DCP2	Device Configuration Parameter	00_BR_DCP2	20
DCR1	D-Correction Record	00_CR_DCR_0_1	20
DIS1	Data Identification String Parameter	00_CR_DIS1	20
DIS2	Data Identification String PLC Program	00_CR_DIS2	20
DIS3	Data Identification String NC Program	00_CR_DIS3_1	20
DIS4	Data Identification String Tool List	00_CR_DIS4_0	20
DIS5	Data Identification String Machine	00_CR_DIS5	20
DIS6	Data Identification String Machine	00_CR_DIS6_1_0_1	20
DPN	Delete Part Program NC		140
DPP	Delete Part Program Package	00_BW_DPP_2	40
DTC1	Device Tool Management Configuration	00_BR_DTC1	20
DTG1	Distance To Go	00_CR_DTG1_0_2_1	20
DTG2	Distance To Go by log. Axis No	00_CR_DTG2_3_1	20
DTY1	Device TYpe	00_CR_DTY1	20
EPO1	ProgrammEd POsition (END)	00_CR_EPO1_0_2_1	20
EPO2	ProgrammEd POsition (END)	00_CR_EPO2_3_1	20
GPC1	Global Process Configuration	00_BR_GPC1	100
GPC2	Global Process Configuration	00_BR_GPC2_0	120
GPP1	Global Process Parameter	00_BR_GPP1	20
GPP2	Global Process Parameter	00_BR_GPP2_0	20
MAP1	Module Assign of Process	00_BR_MAP1_4	20
MCD1	Module Configuration: Device Information	00_BR_MCD1	20
MCM1	Module Configuration: Module Information	00_BR_MCM1	20
MCP1	Module Configuration: Process Information	00_BR_MCP1_1	20
MCS1	Module Configuration: SFC- Information	00_BR_MCS1_1	30
MFO1	Maximum Feedrate Override	00_CR_MFO1_0	20
MFR	Maximum FeedRate	00_CR_MFR_0	20
MRO1	Maximum Rapid Override	00_CR_MRO1_0	20
MSO1	Maximum Spindle Override	00_CR_MSO1_0_1	20
MSS	Maximum Spindle Speed	00_CR_MSS_0_1	20
MTD	Machine Table Data	00_CR_MTD_90_0_0_1_7	20

Com.	Description	Example	[ms]
NEV	NC EVent	00_CR_NEV_0_1	20
NMM	NC MeMory selection	00_CW_NMM Value: 2	10
NPA1	NC PArameter	00_BR_NPA1_01_A00.000	90
NPA2	NC PArameter	00_BR_NPA2_01_A00.000_A00.004	90
NPA3	NC PArameter	00_BR_NPA3_01_A00.000_3	100
NPA4	NC PArameter	00_BR_NPA4_01_A00.000	120
NPS	NC Program Selection	00_CW_NPS_0 Value: 2	10
NTN	Next Tool Number	00_CR_NTN_0	20
NVS	NC Variable Single	00_CR_NVS_0_0	20
OPD1	Optimum Position Distance	00_CR_OPD1_0_2	20
OPD2	Optimum Position Distance by log. Axis No	00_CR_OPD2_3	20
PAC1	Process Axis Configuration Parameter	00_BR_PAC1	10
PAC2	Process Axis Configuration Parameter	00_BR_PAC2_0	20
PFR	Programmed FeedRate	00_CR_PFR_0	20
PPD	Part Program Directory		10
PPN	Part Program NC	00_BR_PPN_1_0_1_1	60
PPP	Part Program Package	00_BA_PPP_1/1 Value: PROGNAM	20
PPS	Part Program Sequence	00_CR_PPS_1_0_1_1	20
PSS	Programmed Spindle Speed	00_CR_PSS_0_1	20
PTC1	Process Tool Management Configuration	00_BR_PTC1	20
PTC2	Process Tool Management Configuration	00_BR_PTC2_0	20
PVF	PLC Variable Formatted	00_CR_PVF_VAR1	20
PVS	PLC Variable Single	00_CR_PVS_ErrorFlg	20
PVT	PLC Variable Type	00_BR_PVT_VAR1	10
SID1	Software Installation Data	00_BR_SID1	30
SLA1	Actual Servo LAg	00_CR_SLA1_0_2	20
SLA2	Actual Servo LAg	00_CR_SLA2_3	20
SLI	SPS (PLC) Long Identification	00_BR_SLI	30
SPA1	Sercos PArameter	00_BR_SPA1_1_S-0-0001_40	120
SPH	Sercos PHase	00_CW_SPH_1 Value: 2	20
SPP	Selected Part Program Number	00_CR_SPP_0	20
TDA1	Tool DAta	00_BR_TDA1_0_M_21	60
TDA2	Tool DAta	00_BR_TDA2_0_1_1	70
TDR1	Tool Data Record of Place	00_CR_TDR1_0_M_21_0	30
TDR2	Tool Data Record	00_CR_TDR2_0_1_1_0	20
TIF	Tool Insert Finish	00_CR_TIF_0_M_25	20
TII	Tool Insert Initiated	00_CR_TII_0_M_25	20
TLB1	TooL Basic Data List	00_BR_TLB1_0_M_1_10_2_5_6_7	380 *2)
TLB2	TooL Basic Data List	00_BR_TLB2_0_2_5_6_7	700 *2)
TLD1	TooL Data of Place	00_CR_TLD1_0_M_1_1_1	20
TLD2	TooL Data of Tool	00_CR_TLD2_0_1_1_0_5	20
TLD3	TooL Data of Place	00_CR_TLD3_0_M_2_1	30

Com.	Description	Example	[ms]
TLD4	TooL Data of Tool	00_CR_TLD4_0_1_1_1	30
TLE1	TooL Edge Data List	00_BR_TLE1_0_1_M_1_3_2_3	260 *2)
TLE2	TooL Edge Data List	00_BR_TLE2_0_1_3_4_5_9	770 *2)
TMV	Tool MoVe	00_CR_TMV_0_M_24_M_25	20
TQE1	Actual TorQuE	00_CR_TQE_0_2	20
TQE2	Actual TorQuE	00_CR_TQE1_0_2	20
TRM	Tool ReMove	00_CR_TRM_0_M_25	20
TRS	Tool ReSet	00_CR_TRS_0_M_25	20
ZOD	Zero Offset Data	00_CR_ZOD_1_0_0_4_1	20
ZOD1	Zero Offset Data	00_CR_ZOD1_1_0_0_4	20
ZOD2	Zero Offset Data	00_CR_ZOD2_1_0_0_4_1	20

Fig. 6-18: Command execution times of the MWCX device group

Command execution times for the MSCX device group

Com.	Description	Example	[ms]
ASE	Actual System Error	00_CR_ASE	20
CSE	Clear System Error	00_CW_CSE No Value	20
DTY1	**D**evice **TY**pe	00_CR_DTY1	60
SPA1	Sercos Parameter	00_BR_SPA1_1_S-0-0001_40	150
SPH	Sercos Phase	00_CW_SPH_1 Value: 2	30

Fig. 6-19: Command execution times of MSCX device groups

Command execution times for the MWSX device group

Com.	Description	Example	[ms]
ASM1	Active System Fault Message	00_BR_ASM1	60
ASM2	Active System Fault Message	00_BR_ASM2	20
ASM3	Active System Fault Message	00_BR_ASM3_02	10
ASM4	Active System Fault Message	00_BR_ASM4_MWCX	10
ASM5	Active System Fault Message	00_BR_ASM5_74_0	10
CRT	Control ReseT		20
DIS2	Data Identification String PLC Program	00_CR_DIS2	20
DTY1	Device TYpe	00_CR_DTY1	20
MCD1	Module Configuration: Device Information	00_BR_MCD1	10
MCM1	Module Configuration: Module Information	00_BR_MCM1	10
MCP1	Module Configuration: Process Information	00_BR_MCP1_1	
MCS1	Module Configuration: SFC- Information	00_BR_MCS1_1	10
PVF	PLC Variable Formatted	00_CR_PVF_VAR1	20
PVS	PLC Variable Single	00_CR_PVS_ErrorFlg	20
PVT	PLC Variable Type	00_BR_PVT_VAR1	10
SID1	Software Installation Data	00_BR_SID1	20
SLI	SPS (PLC) Long Identification	00_BR_SLI	10

Fig. 6-20: Command execution times of the MWSX device group

Command execution times for the MWAX device group

Com.	Description	Example	[ms]
AMM7	Active Mechanism Message	01_BR_AMM7	10
APO2	Actual Machine Position	00_CR_APO2_3_1	20
ASM1	Active System Fault Message	00_BR_ASM1	70
ASM2	Active System Fault Message	00_BR_ASM2	60
ASM3	Active System Fault Message	00_BR_ASM3_02	80
ASM4	Active System Fault Message	00_BR_ASM4_MWCX	60
ASM5	Active System Fault Message	00_BR_ASM5_74_0	20
CMA	CMOS RAM ASCII Parameter	00_CR_CMA_10	20
CMF	CMOS RAM Floating Point Parameter	00_CR_CMF_10	20
CMI	CMOS RAM Integer Parameter	00+C13_CR_CMI_10	20
CRT	Control ReseT		50
DCP1	Device Configuration Parameter	00_BR_DCP1	20
DCP2	Device Configuration Parameter	00_BR_DCP2	10
DIS2	Data Identification String PLC Program	00_CR_DIS2	70
DTG2	Distance To Go by log. Axis No	00_CR_DTG2_3_1	20
DTY1	**D**evice **TY**pe	00_CR_DTY1	30
MAP1	Module Assign of Process	00_BR_MAP1_4	20
MCD1	Module Configuration: Device Information	00_BR_MCD1	10
MCM1	Module Configuration: Module Information	00_BR_MCM1	10
MCP1	Module Configuration: Process Information	00_BR_MCP1_1	20
MCS1	Module Configuration: SFC- Information	00_BR_MCS1_1	20
PVF	PLC Variable Formatted	00_CR_PVF_VAR1	40
PVS	PLC Variable Single	00_CR_PVS_ErrorFlg	40
PVT	PLC Variable Type	00_BR_PVT_VAR1	10
SID1	Software Installation Data	00_BR_SID1	20

Fig. 6-21: Command execution times of the MWAX device group

Command execution times for the MSYX device group

Com.	Description	Example	[ms]
ASE	Actual SERCANS Error	00_CR_ASE	
CSE	Clear SERCANS Error	00_CW_CSE No Value	
DTY1	**D**evice **TY**pe	00_CR_DTY1	
SID1	Software Installation Data	00_BR_SID1	
SPA1	Sercos Parameter	00_BR_SPA1_1_S-0-0001_40	
SPH	Sercos Phase	00_CW_SPH_1 Value: 2	

Fig. 6-22: Command execution times of the MSYX device group

7 Function Interface Commands

7.1 FI Commands for the MPCX Device Group

The following FI commands are valid for the MPCX device group. Always make sure to place device address "XX" before the FI command, e.g. XX_BR_CCP1 (also refer to the chapter "Elements of the FI command").

Interrupting Function Interface Jobs: BFJ

MPCX Device Group

Designation **BFJ** **B**reak-**F**unction-Interface **J**obs

Explanation This is a means for interrupting tasks or FI jobs. The FI command "BFJ1" interrupts all interface jobs, "BFJ2" interrupts the selected job.

Note: Not all FI jobs can be interrupted with the BFJ command!

FI command Interrupt all FI jobs that are running.

XX_BR_BFJ1 **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command "BFJ1". If FI jobs are running, the response consists of one to n lines (n = the number of FI jobs running), each with two columns.

Line 1...n:	Column 1	Column 2

Value Range/Meaning of Columns

1 = ID of job to be interrupted [01...20]

2 = FI command string.

Example BFJ1 Interrupts all FI jobs that are running.

Note:
The processing of ALL FI jobs that are currently running and that it is possible to interrupt is stopped by this FI command.

Assumption:
The two FI jobs with the job IDs 1 and 2 are running.

FI command		**XX_BR_BFJ1**
Line	**Column**	**Answer**
1	1	01
	2	02_BW_PAA2_C:\DOWNLOAD1.PDL /3
2	1	02
	2	01_BW_PAA2_C:\DOWNLOAD2.PDL/3\/3

FI command Interrupt the selected FI job.

XX_BR_BFJ2_(1) **(Single Read)**

(1) = ID of job to be interrupted [01...20]

Response Structure

The following table shows the general structure of the response to the FI command "BFJ2". The response consists of a line with two columns.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 =	ID of job to be interrupted	[01...20]
2 =	FI command	[String of the FI Command]

Example BFJ2

Interrupts the FI job 01.

Note:
A parameter download job is currently running with the job ID 01 for the device 00.

FI command		**XX_BR_BFJ2_01**
Line	**Column**	**Answer**
1	1	01
	2	00_BW_PAA2_C:\DOWNLOAD1.PDL/3

Outputting the Device Configuration: CCP

MPCX Device Group

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

Explanation

The configuration settings are read in from the "IND_DEV.INI" file. The configuration of the individual communication addresses and the settings of the various Rexroth Indramat devices are determined in this file (see Chapter 5 “Installation”).

FI command

Output the configuration settings of all devices defined in the "IND_DEV.INI" file.

XX_BR_CCP1 **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command “CCP1”. The response consists of a maximum of n=16 lines (n=16 configurable devices), each with 15 columns.

Line 1...n:	Column 1	...	Column 15

Value Range/Meaning of 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 =	Assignment 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 6.1 “Identifier”)
12 =	PLC component type	IND_DEV.INI entry: Component type1=
13 =	PLC component type	IND_DEV.INI entry: Component type2=
14 =	Device log	IND_DEV.INI entry: DeviceProtocol=

15 = Device simulation IND_DEV.INI entry: DeviceSimulation=

Example CCP1 Read the configuration settings of all devices defined in the "IND_DEV.INI" file.

Assumption:
The following device types have been defined:

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

FI command		XX_BR_CCP1
Line	**Column**	**Answer**
1	1	00
	2	Pressure barrel 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
	14	CNC
	15	OFF
2	1	15
	2	Transport unit
	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
	14	CNC
	15	OFF

FI command Output the configuration settings of the selected device type.

BR_CCP2 **(Single Read)**

(1)= Device type [MTC200-P-G2, MTC200-R-G2, MTVNC, SERCANS-A, SERCANS-P, ISP200-P-G2, ISP200-R-G2, TRA200-P, TRA200-R, MTA200-P]

Response Structure

The following table shows the general structure of the response to the FI command “CCP2”. The response consists of a maximum of n=16 lines (n=16 configurable devices), each with 15 columns.

Line 1...n:	Column 1	...	Column 15

Value Range/Meaning of 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 =	Assignment 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 6.1 “Identifier”)
12 =	PLC component type	IND_DEV.INI entry: Component type1=
13 =	CNC component type	IND_DEV.INI entry: Component type2=
14 =	Device log	IND_DEV.INI entry: DeviceProtocol=
15 =	Device simulation	IND_DEV.INI entry: DeviceSimulation=

Example CCP2

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

Assumption:
The following device types have been defined:

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

FI command		XX_BR_CCP2_SERCANS-A
Line	**Column**	**Answer**
1	1	00
	2	Pressure barrel 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
	14	CNC

FI command		XX_BR_CCP2_SERCANS-A
Line	**Column**	**Answer**
	15	OFF

FI command

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

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

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

Response Structure

The following table shows the general structure of the response to the FI command “CCP3”. The response consists of a maximum of n=16 lines (n=16 configurable devices), each with 15 columns.

Line 1...n:	Column 1	...	Column 15

Value Range/Meaning of 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 =	Assignment 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 6.1 “Identifier”)
12 =	PLC component type	IND_DEV.INI entry: Component type1=
13 =	CNC component type	IND_DEV.INI entry: Component type2=
14 =	Device log	IND_DEV.INI entry: DeviceProtocol=
15 =	Device simulation	IND_DEV.INI entry: DeviceSimulation=

Example CCP3

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

Assumption:
The following device types have been defined:

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

FI command		XX_BR_CCP3_1
Line	**Column**	**Answer**
1	1	15
	2	Transport unit
	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
	14	CNC
	15	OFF

FI command

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

BR_CCP4_(1)	**(Single Read)**
(1) = Device group	[MTCX, MSCX, MISX, MTRX, MTAX] (see Chapter 6.1 "Identifier")

Response Structure

The following table shows the general structure of the response to the FI command “CCP4”. The response consists of a maximum of n=16 lines (n=16 configurable devices), each with 15 columns.

Line 1...n:	Column 1	...	Column 15

Value Range/Meaning of 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 =	Assignment 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 “Identifier”)
12 =	PLC component type	IND_DEV.INI entry: Component type1=
13 =	CNC component type	IND_DEV.INI entry: Component type2=
14 =	Device log	IND_DEV.INI entry: DeviceProtocol=
15 =	Device simulation	IND_DEV.INI entry: DeviceSimulation=

Example CCP4

Read the configuration settings of the defined MSCX devices.

Assumption:
The following device groups have been defined:

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

FI command		XX_BR_CCP4_MSCX
Line	**Column**	**Answer**
1	1	00
	2	Pressure barrel 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
	14	CNC
	15	OFF

FI command

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

BR_CCP5_(1)	**(Single Read)**
(1) = Device address	[00...63]

Response Structure

The following table shows the general structure of the response to the FI command “CCP5”. The response consists of a line with 15 columns.

Line 1...n:	Column 1	...	Column 15

Value Range/Meaning of 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 =	Assignment 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 6.1 “Identifier”)
12 =	PLC component type	IND_DEV.INI entry: Component type1=

13 =	CNC component type	IND_DEV.INI entry: Component type2=
14 =	Device log	IND_DEV.INI entry: DeviceProtocol=
15 =	Device simulation	IND_DEV.INI entry: DeviceSimulation=

Example CCP5

Read the configuration settings of device address 00.

Assumption:
The following device addresses have been defined:

Device address 00: MSCX

Device address 03: MTCX

FI command		**XX_BR_CCP5_00**
Line	**Column**	**Answer**
1	1	00
	2	Pressure barrel 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
	14	CNC
	15	OFF

Reading the FI communication error counts: CEI

MPCX Device Group

Designation

CEI **C**ommunication **E**rror **I**nfo

FI command

Reading the counts for the communication errors recorded in the protocol.

BR_CEI1 **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command "CEI1". A line of 5 columns is output.

Line 1	Column 1	...	Column 5

Value Range/Meaning of Columns

1 = Error counter: PC side	Contains the communication error occurred until that time registered by the PC.
2 = Error counter: SIO side	Contains the communication error occurred until that time registered by the SIO.
3 = Error counter: internal timeout	Contains the internal timeouts occurred until that time – which are compensated through FI, if applicable.
4 = Error counter: number of dispatches of repeat telegrams	Contains the repeat telegrams occurred until that time.

5 = Error counter: timeout	Contains the timeouts occurred until that time – are signaled to the application.

Example: CEI1 Supply the current counts for communication errors.

FI command		**XX_BR_CEI1**
Line	**Column**	**Answer**
1	1	1
	2	0
	3	0
	4	1
	5	0

Commands for Executing WIN32 Applications: CPR

MPCX Device Group

Designation **CPR** **C**reate **PR**ocess

Explanation WIN32 applications can be executed with this FI command. These applications may or may not be logged in the FI.

FI command Execute a WIN32 application that is logged on in the FI.

XX_BW_CPR1_(1)_(2)_(3)	**(Single Write)**
(1) = Complete EXE name	Complete physical path name for the WIN32 application that is to be executed
(2) = Min Info	Control information as to whether or not the current screen window (output window) is to be minimized. The following applies: 0 = do not minimize 1 = minimize
(3) = Wait Info	Control information as to how the output window is focused; here, the following applies: 0 = Re-focussing with the SFW2 command 1 = Automatic re-focussing on termination of the WIN32 application

Response Structure As this concerns a command there is no response data.

Example CPR1 The WIN32 application "VBDEMO.EXE" is executed via the FI. The output window is minimized and automatically focused again after "VBDEMO.EXE" has ended.

Assumption:
The VBDEMO.EXE program is in the subdirectory D:\Programs\Indramat\Mtgui\bin.

FI command	**XX_BW_CPR1_ D:\Programs\Indramat\Mtgui\bin\VBDEMO.EXE_1_1**

FI command Execute a WIN32 application, that is NOT logged on in the FI.

XX_BW_CPR2_(1)_(2)	**(Single Write)**
(1) = Complete EXE name	Complete physical path name for the WIN32 application that is to be executed.
(2) = Min Info	Control information as to whether or not the current screen window (output window) is to be minimized. The following applies: 0 = do not minimize

1 = minimize

Response Structure As this concerns a command there is no response data.

Example CPR2 To start Windows Task Manager via the function interface. The output window is minimized and automatically focused again after "TASKMGR.EXE" has ended.

Assumption:
The program "TASKMGR.EXE" is in the subdirectory C:\Winnt\System32.

FI command **XX_BW_CPR2_C:\Winnt\System32\Taskmgr.exe_1**

Removing Function Interface Jobs: DFJ

MPCX Device Group

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

Explanation Jobs, also referred to as FI jobs, are removed from the management structure of the function interface. These are jobs that have either the status "READY" or "ERROR" . The FI command “DFJ1” removes all interface jobs; “DFJ2” removes the selected job.

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

XX_BR_DFJ1 **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command “DFJ1”. The response consists of a maximum of n=19 lines (n=19 maximum number of FI jobs), each with two columns.

Line 1...n:	Column 1	Column 2

Value Range/Meaning of Columns

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

2 = FI command

Example DFJ1 Delete all FI jobs.

Assumption:
An NC program has been transferred successfully into the device (control unit) using the FI command "NCA1" (see FI commands for the MWCX device group).

- Job ID of the NC download program: 01

FI command		XX_BR_DFJ1
Line	**Column**	**Answer**
1	1	01
	2	02_BR_NCA1_"D:\Download.ini" /3

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

XX_BR_DFJ2_(1) **(Single Read)**

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

Response Structure The following table shows the general structure of the response to the FI command “DFJ2”. The response consists of one line with 13 columns.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

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

2 = FI command [string, in accordance to chapter entitled "Elements of the FI

Command"]

Example DFJ2 Delete the FI job 01.

Assumption:
An NC program has been transferred successfully into the device (control unit) using the FI command "NCA1" (see FI commands for the MWCX device group).

- Job ID of the NC download program: 01

FI command		**XX_BR_DFJ2_01**
Line	**Column**	**Answer**
1	1	01_BR_NCA1_"D:\Download.ini" /3

Deleting of the FI Command Stack Administration: DFS

MPCX Device Group

Designation **DFS** **D**elete **IF** Command **S**tack

Explanation This FI command deletes the FI command stack administration. As a write value, a reference information string must be transmitted in the DataTransfer() function which is supplied as reference information with the SYS message "MSG_MESSAGECH".

FI command **BW_DFS1** **(Single Write)**

Response Structure The response to the "DFS1" FI command consists of one line with one column.

Line 1	**Column 1**

Value Range/Meaning of Columns 1 = Status message (P_ACK) (P_ACK)

Example DFS1 Deleting of the FI Command Stack Administration. As a write value, a reference information string must be transmitted in the DataTransfer() function which is supplied as reference information with the SYS message "MSG_MESSAGECH".

FI command		**XX_BW_DFS1** **Value to be written:** **Reference information string**
Line	**Column**	**Answer**
1	1	(P_ACK)

Command for Terminating WIN32 Applications: DPR

MPCX Device Group

Designation **DPR** **D**elete **PR**ocess

Explanation WIN32 applications that have been logged in the FI **AND** that are processing the termination event can be terminated with this FI command. See LogInIf() description.

FI command Terminate a WIN32 application that is logged on in the FI **AND** that is processing a termination event.

XX_BW_DPR1_(1)	**(Single Write)**
(1) = LogInIf Login name	This refers to the login name entered at the FI (LogInIf()) during the login procedure

Response Structure

As this concerns a command there is no response data.

Example DPR1

The WIN32 application "VBDEMO.EXE" that is running is terminated via the function interface. The FI command is carried out by any WIN32 application logged in the FI.

FI command	**XX_BW_DPR1_VBDEMO.EXE**

Error Information: ERI

MPCX Device Group

Designation

ERI **ER**ror **I**nformation

Explanation

Returns the error text and the additional text of an FI error code or a Windows NT error code.

FI command

Read error text and additional text.

BR_ERI1_(1)_(2) **(Single Read)**

(1) = Error class [1 = NACK error number,
2 = FI error code
3 = reserved
4 = Windows NT error code]

(2) = Error number [LONG]

Response Structure

The following table shows the general structure of the response to the FI command "ERI". Two lines, each with one column, are outputted. Line 1 contains the error text and line 2 contains the additional text.

Lines 1..2	**Column 1**

Meaning of the Column

1 = Error text [language-dependent]

2 = Additional text [language-dependent]

Example ERI

Read the error text including the additional error text with error number 26.

FI command		**XX_BR_ERI1_1_26**
Line	**Column**	**Answer**
1	1	Error in mathematical expression.
2	1	Check mathematical expression. Correct NC program and re-transmit

Far Configuration Parameter: FCP

MPCX Device Group

Designation

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

Explanation

The FI command "FCP" returns the list of the addressable devices on the PC. Differentiation is made between two cases (A and B):

- PC is in the PC network and
- PC is stand-alone.

Case A
PC is in the PC Network

The list of FarDevices defined in the network configuration data on the PC is read (see "FAR_DEV.INI" file). In addition the local devices that are not defined as FarDevices are output.

Case B
PC stand-alone

The list of local devices is outputted if one or more of the following points apply:

- There is no network configuration data on the PC (see "FAR_DEV.INI" file).
- The PC has been disabled in the network configuration data, or
- the "PC Network Active" option is not switched on in the system configurator.

FI command

Read out the addressable devices on the PC.

XX_BR_FCP1{_(1))	**(Single Read)**
(1) = Device selection	[L= only local, F= only FAR] ! Optional !

Read-out of the addressable devices on the PC; but only applies to devices from the stipulated device groups:

XX_BR_FCP2_(1){_(2)}	**(Single Read)**
(1) = Device group	[MPCX, MTCX, MISC, MTAX, MTRX]
(2) = Device selection	[L= only local, F= only FAR] ! Optional !

Read-out of the addressable devices on the PC; however, only applies to devices of the stipulated device type:

XX_BR_FCP3_(1){_(2)}	**(Single Read)**
(1)= Device type	[MTC200-P-G2, MTC200-R-G2, MTVNC, SERCANS-A, SERCANS-P, ISP200-P-G2, ISP200-R-G2, TRA200-P, TRA200-R, MTA200-P]
(2) = Device selection	[L= only local, F= only FAR] ! Optional !

Response Structure

The following table shown the general structure of the FI commands "FCP1", "FCP2" and "FCP3". The number of lines depends on the actual configuration.

Result when network configuration data is available:

Line 1...n:	Column 1	...	Column 10

Value Range/Meaning of the Columns

1 = FarDevice address	[00...15]
2 = Device name	[max. 28 ASCII characters]
3 = Device type	[MTC200-P-G2, MTC200-R-G2, MTVNC, SERCANS-A, SERCANS-P, ISP200-P-G2, ISP200-R-G2, TRA200-P, TRA200-R, MTA200-P]
4 = Local device address	[00...15]
5 = PC No.	[00...15, XX]
6 = Local device	[YES, NO, --]
7 = Device status	ON, OFF
8 =Assignment of a simulation pair.	[00...15, NO]
9 = Device group	[MPCX, MTCX, MISC, ...]
10 = Online	[YES, NO, --]

Explanation of Column 1 FarDevice Address

The contents of Column 1 can always be used to address the local as well as the far (remote) devices. A generic application must have the value of a device address within the FI command.

Explanation of Column 7 Device Status

In case A, the "Disable" entry from the "FAR_DEV.INI" file is evaluated. The following assignment applies:

- ON if "Disable = NO" or if the "Disable" entry is missing
- OFF if "Disable = YES" or
- OFF if the PC is disabled.

	FarDevice, Disable = YES	**FarDevice, Disable = NO**
PC, Disable = YES	OFF	OFF
PC, Disable = NO	OFF	ON

Note: If a PC is disabled then its corresponding devices are also in the “Disable” status.

Explanation of Column 10 Online?

This column indicates whether there is currently a connection to the PC via which the device can be addressed. Differentiation is made between 3 possible cases:

- YES = The network connection to the PC is active
- NO = The network connection is down (interrupted).
- -- = The network connection has not yet been completely checked.

Note: YES is always output for B.

Example FCP1 Case A

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

Assumption:
The following device types have been defined:

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

FI command		**XX_BR_FCP1**
Line	**Column**	**Answer**
1	1	15
	2	Drill left
	3	MTCNC
	4	05
	5	02
	6	YES
	7	ON
	8	11
	9	MTCX
	10	YES
2	1	11
	2	Drill left
	3	MTVNC
	4	01
	5	02
	6	YES
	7	ON

FI command		XX_BR_FCP1
Line	Column	Answer
	8	15
	9	MTCX
	10	YES
3	1	12
	2	Drill right
	3	MTVNC
	4	02
	5	03
	6	NO
	7	OFF
	8	NO
	9	MTCX
	10	NO

Note: If there is an entry [DeviceOrder] in the "IND_DEV.INI" file or in the "FAR_DEV.INI" file, then these entries (lines) are output in the order in which they are listed there. If no entry [DeviceOrder] is given, then the devices are outputted according to the order of the sections in the file.

Example FCP1 Case B

Read the network configuration of all devices defined in the "IND_DEV.INI" file. (Case B)

Assumption:
The following device types have been defined but there is no network configuration data:

- Device address 05: MTC200-P-G2
- Device address 01: MTVNC

Note: No configuration data is available or the local PC is not active in the network or the PC has been disabled in the network configuration data (see the explanation for Case B).

FI command		XX_BR_FCP1
Line	Column	Answer
1	1	05
	2	Drill left
	3	MTC200-P-G2
	4	05
	5	XX
	6	YES
	7	ON
	8	NO
	9	MTCX
	10	YES

FI command		XX_BR_FCP1
Line	**Column**	**Answer**
2	1	01
	2	Drill left
	3	MTVNC
	4	01
	5	XX
	6	YES
	7	ON
	8	NO
	9	MTCX
	10	YES

Note: If there is an entry [DeviceOrder] in the "IND_DEV.INI" file or in the "FAR_DEV.INI" file, then these entries (lines) are output in the order in which they are listed there. If no entry [DeviceOrder] is given, then the devices are outputted according to the order of the sections in the file.

Far Device Configuration Parameter: FDC

MPCX Device Group

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

Explanation FI command "FDC" returns the general data of the PC network. Differentiation is made between two cases (A and B):

- PC is in the PC network and
- PC is stand-alone.

Case A PC is in the PC Network The FI command returns the general data of the PC network. Furthermore, additional data such as the hostname and IP address of the PC is also outputted.

Case B PC stand-alone The data of the local PC is outputted if one or more of the following points apply:

- There is no network configuration data on the PC (see "FAR_DEV.INI" file).
- The PC has been disabled in the network configuration data or
- the “PC Network Active” option is not switched on in the system configurator.

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

Response Structure

The following table shows the general structure of the response to the FI command "FDC1".

Line 1	Column 1	...	Column 4
Line 2	Column 1		
Line 3	Column 1		
Line 4	Column 1		
Line 5	Column 1		
Line 6	Column 1	...	Column 4

Value Range/Meaning of Columns

Line 1:

1 = PC network exists?	[YES, NO]
2 = Name of the PC network	[max. 28 ASCII characters]
3 = Max. number of PCs	(Integer)
4 = Max. number of devices	(Integer)

Line 2:

1 = PC No.	[00...15, XX]

Line 3:

1 = Hostname/ Ethernet hostname possibly expanded by name of domain	(string)

Line 4:

1 = Computer name/ NETBIOS name of computer	(string)

Line 5:

1 = IP address of network card 1	(string)
...	...
4 = IP address of network card 4	(string)

Line 6:

1 = Master PC?	[YES = PC is Master PC (Head PC), NO]

Example FDC1 Case A

Read the general data of the PC network.

Assumption:
A PC with two network cards has been 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**
Line	**Column**	**Answer**
1	1	YES
	2	Operation10
	3	20
	4	64
2	1	XX
3	1	Machine1.Cell1
4	1	MACHINE1
5	1	172.16.0.1
	2	172.16.1.1
6	1	YES

Example FDC1 Case B

Read the general data of the PC network.

Assumption:
No PC is active or defined within the network.

FI command		**XX_BR_FDC1**
Line	**Column**	**Answer**
1	1	NO
	2	--
	3	1
	4	16
2	1	XX
3	1	Machine1.Cell1
4	1	MACHINE1
5	1	172.16.0.1
6	1	--

Further Info Text: FIT

MPCX Device Group

Designation **FIT** **F**urther **I**nfo **T**ext

Explanation Returns the additional text of an FI error code or a NACK error number.

FI command Read additional (further) text.

BR_FIT1_(1)_(2)	**(Single Read)**
(1) = Error class	[1 = NACK error number, 2 = FI error code]
(2) = Error number	[LONG]

Response Structure One line with one column is outputted for the additional text.

Line	Column

Meaning of the Column

Additional text	[language-dependent]

Example FIT Read the additional general error text with the number 26.

FI command		**XX_BR_FIT1_1_26**
Line	**Column**	**Answer**
1	1	Check mathematical expression. Correct NC program and re-transmit

Note: The general error result line contains an "X" in Column 5 when there is an additional text, otherwise simply "--". You can obtain the additional error text by calling up the “XX_BR_FIT1” command with the 1st and 2nd partial result.

Far PC Configuration Parameters: FPC

MPCX Device Group

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

Explanation The FI-Command "FPC" outputs the list of PCs that are defined in the network. Differentiation is made between two cases (A and B):

- PC is in the PC network and
- PC Stand-Alone.

Case A
PC is in the PC Network

The list of PCs defined in the network configuration files on the PC (see "FAR_DEV.INI" file) is outputted.

Fall B
PC Stand-Alone

The data of the local PC is outputted if one or more of the following points apply:

- There is no network configuration data on the PC (see "FAR_DEV.INI" file).
- The PC has been disabled in the network configuration data or
- The “PC Network Active” option is not switched on in the system configurator.

FI command **XX_BR_FPC1** **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command “FPC1”. The number of lines depends on the actual configuration. Result when network configuration data is available:

Line 1...n:	Column 1	...	Column 7

Value Range/Meaning of the Columns

1 = PC No.	[00...15, XX]
2 = Port	[IP address, host name]
3 = Name of PC	[max. 28 ASCII characters]
4 = Local device	[YES = PC is the local PC, NO = PC is a remote PC]
5 = Device status	[OFF = PC is disabled, ON = PC is enabled] corresponds to the "Disable" entry of section "PC<pcnr>"
6 = Master?	[YES = PC is Master PC (Head PC), NO] corresponds to the "Master PC" entry of section "PC<pcnr>"
7 = Online?	[YES, NO, --]

Explanation of Column 7
Online?

This column indicates whether there is currently a connection to the PC via which the device can be addressed. Differentiation is made between 3 possible cases:

- YES = The network connection to the PC is active
- NO = The network connection is down (interrupted).
- -- = The network connection has not yet been completely checked.

Note: YES is always output for B.

Example FPC1 Case A

Read the list of 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 name: st100103

FI command		XX_BR_FPC1
Line	**Column**	**Answer**
1	1	10
	2	192.4.4.91
	3	Drill station 02
	4	YES
	5	OFF
	6	NO
	7	YES
2	1	20
	2	st100103
	3	Drill station 03
	4	NO
	5	ON
	6	YES
	7	NO

Note: If there is an entry [DeviceOrder] in the "IND_DEV.INI" file or in the "FAR_DEV.INI" file, then these entries (lines) are output in the order in which they are listed there. If no entry [DeviceOrder] is given, then the devices are outputted according to the order of the sections in the file.

Example FPC1 Case B

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

FI command		XX_BR_FPC1
Line	**Column**	**Answer**
1	1	XX
	2	--
	3	--
	4	YES
	5	ON
	6	--
	7	YES

Writing/Reading the General FI Data Buffer: GDB

MPCX Device Group

Designation **GDB** **G**lobal **D**ata **B**uffer

Explanation Writes/reads data for the general FI data buffer. A maximum of 100 byte can be transported in this FI data buffer.

Note: As much information as wished (max. 100 byte) can be exchanged between WIN32 applications by using the general FI data buffer. Data is identified by means of the relevant buffer ID.

Note!
The buffer IDs 686 to 695 are available for external applications!

FI command Write data into an FI data buffer.

BW_GDB1_(1) **(Single Write)**

(1) = Buffer ID [686-695]

Value to be written
Data to be transported (max. 100 byte)

Response Structure (P_ACK) is returned following successful transmission.

Line 1	**Column 1**

Value Range/Meaning of Columns 1 = Successfully completed (P_ACK)

Example GDB1 Binary data (max. 100 byte) to be transferred to the general FI data buffer as a write value are written with the Buffer ID 686.

FI command		**XX_BW_GDB1_686**
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command Read data from an FI data buffer.

BR_GDB1_(1) **(Single Write)**

(1) = Buffer ID [686-695]

Response Structure The contents of the addressed FI data buffer are returned following successful transmission.

Line 1	**Column 1**

Value Range/Meaning of Columns 1 = Contents of the addressed FI data buffer [Data of the addressed FI data buffer (max. 100 byte]

Example GDB1 Read the general FI data buffer using the buffer ID 686.

FI command		**XX_BR_GDB1_686**
Line	**Column**	**Answer**
1	1	[Data of the addressed FI data buffer (max. 100 byte]

Initialization of a V24 Communication Address: ICA

MPCX Device Group

Designation **ICA** **I**nitialization **C**ommunication **A**ddress

Explanation By means of this command, a defined communication address (that has been created by the system configurator – CommAddr entry in the configuration file IND_DEV.INI) is initialized with new parameters.

FI command **BW_ICA1_(1)_(2)** **(Single Write)**

(1) = selected communication address (CommAddr entry in the configuration file IND_DEV.INI)

(2) = initialization string according to specification (CommAddr entry in the configuration file IND_DEV.INI)

Response Structure The response to the "ICA1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK) (P_ACK)

Example ICA1 The defined communication address 1 is initialized with the following parameters:

COM-PORT:	1
BAUD RATE:	38400
PARITY:	NONE
MODE:	RS232
PC-COUNTER:	TCON

FI command		**XX_BW_ICA1_1_V24,COM1,38400,NONE,RS232,TCON**
Line	**Column**	**Answer**
1	1	(P_ACK)

Information regarding Function Interface Jobs: IFJ

MPCX Device Group

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

Explanation Status information regarding active FI jobs can be read out. This status prompt allows, for instance, the basis for implementing a progress report (in the form of a display) during NC download as this can be run in the background for some time depending on the size of the NC program.

FI command Return status information on all active FI jobs.

XX_BR_IFJ1 **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command “IFJ1”. The answer consists of a maximum of n=19 lines (n=19 maximum number of FI jobs), each with 16 columns.

Line 1...n:	Column 1	...	Column 16

Value Range/Meaning of Columns

1 =	Job ID	[01...20]
2 =	FI command	[string, in accordance to chapter entitled "Elements of the FI Command"]
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] until TIMEOUT	
7 =	Start time of the job	[hh:mm:ss:ms]
8 =	Processing time up to now in ms	
9 =	Function interface connection (login) name of the application	
10 =	Progress type	[1 = details of progress in %, 2 = details of absolute progress]
11 =	Details of progress as percentage value	[Value, --], depends on Column 10 "Progress type"
12 =	Information on absolute progress	[Value, --], depends on Column 10 "Progress type"
13 =	Absolute end value	[Value, --], depends on Column 10 "Progress type"
14 =	Progress info buffer; contains display information, e.g., NC program line currently being transmitted.	
15 =	FI Job Error Code	(see chapter entitled “Error Codes”)
16 =	Error info buffer	

Note: The results of the columns depend on the FI job that has been started.

Example IFJ1

Read the status information for all active FI jobs.

Assumption:

- The job with ID 01 has been started by the "NCA1" FI command and has been successfully completed with a READY message.

FI command		**XX_BR_IFJ1**
Line	**Column**	**Answer**
1	1	01
	2	02_BR_NCA1_"D:\Download.ini" /3
	3	2
	4	READY
	5	0
	6	600000
	7	16:15:22:123
	8	120
	9	VBDemo
	10	1
	11	100 %
	12	--
	13	--

FI command		XX_BR_IFJ1
Line	**Column**	**Answer**
	14	--
	15	0
	16	--

FI command Return information regarding the selected and active FI job.

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

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

Note: Information regarding the structure of the response is available in the FI command "XX_BR_IFJ1" described above.

Reading of the FI Command Stack Administration: IFS

MPCX Device Group

Designation **IFS** **IF** Command **S**tack Info

Explanation By means of this command, the current allocation status of the FI command stack management can be read.

FI command **BR_IFS1** **(Single Write)**

Response Structure The response to the "IFS1"I command consists of n lines, each with 4 columns.

Line 1...n	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = IF command stack index [1..40]

2 = IF command request string [max. 500 ASCII characters]

3 = Task name (corresponds to LOGINIF name of the application issued by the IF command [max. 20 ASCII characters]

4 = Access counter reading [LONG value]

Example IFS1 Reads the current data of the IF command stack management, with 3 FI request strings being currently in the management.

FI command		XX_BR_IFS1
Line	**Column**	**Answer**
1	1	1
	2	00_BR_ASM2
	3	VBDEMO.EXE
	4	5467
2	1	2
	2	00_BR_AMM2_0
	3	IND400T.EXE
	4	456234
3	1	3
	2	02_BR_ASM2

FI command		XX_BR_IFS1
Line	**Column**	**Answer**
	3	VBDEMO.EXE
	4	534892

Reading of PC Date and PC Time: LDT

MPCX Device Group

Designation **LDT** PC **L**ocal **D**ate **T**ime

FI command With this command, PC date and time are read. At the same time, the local date time information of the PC is supplied.

BR_LDT1 **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command "LDT1". A line of 1 column is output.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Information on date [Day.month.year]

2 = Information on time [hour:minute:second]

3 = LONG value [internal LONG coding]

Example LDT11 Read the current date and time of the PC.

FI command		XX_BR_LDT1
Line	**Column**	**Answer**
1	1	25.04.2002
	2	07:26:06
	3	7192241

Explanation This FI command sets PC date and time.

FI command **BW_LDT1** **(Single Write)**

Value to be written Date and time information to be written [day.month.year hour:minute:second]

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure The following table shows the general structure of the response to the FI command "BW_LDT1". A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK)

Example LDT1 Set the PC clock to the data: 25.04.2002 07:31:33.

Write value: 25.04.2002 07:31:33

FI command		XX_BW_LDT1
Line	**Column**	**Answer**
1	1	(P_ACK)

Activated Language of the Rexroth Indramat GUI: LNG

MPCX Device Group

Designation **LNG** Activated **L**a**NG**uage

Explanation The country code of the activated language for the Rexroth Indramat GUI is output.

FI command **XX_BR_LNG** **(Single Read)**

Response Structure The response to the FI command "LNG" consists of one line with one column for the country code of the activated language.

Value Range of the Column 1 = Country code of the activated language [2 ASCII characters]

Country code	Language
EN	English (US/GB)
DE	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 Rexroth Indramat GUI.

FI command		XX_BR_LNG
Line	**Column**	**Answer**
1	1	SE

Read System Messages: MSG

MPCX Device Group

Designation **MSG** **M**e**S**sa**G**e

Explanation Reading of system messages

FI command Message

CC_MSG_(1) **(Cyclic Read)**

(1) = SYS-Message number

Note: Exists only as a cyclic command

Response Structure The response of the FI command 'MSG' consists of the system message data.

Example MSG 00_CC_MSG_64 (64 = MSG_SYSERRGEN)

FI command		**00_CC_MSG_64/3**
Line	**Column**	**Answer**
1	1	00

Limitation The following system messages:

SYS Message	SYS Message number
MSG_PCLUPDBEG	52
MSG_PARUPDBEG	24
MSG_FWAUPDBEG	82

These commands cannot be used with the following programs:

- Indramat OPC server
- Indramat DDE server

NT Shutdown Functions: NST

MPCX Device Group

Designation **NST** **N**T-**S**hu**T**-Down

Explanation This allows the NT operating system to be shut down.

FI command Triggers NT-Shut-Down (**WITHOUT** shutdown boxes).

BW_NST1 **(Single Write)**

Response Structure As this concerns a command there is no response data.

Example NST1 Triggers NT-Shut-Down (**WITHOUT** shutdown boxes).

FI command	**XX_BW_NST1**

FI command Trigger NT-Shut-Down (**WITH** shutdown boxes).

BW_NST2_(1) **(Single Write)**

(1) = Time in seconds The shutdown box appears for the input time

Response Structure As this concerns a command there is no response data.

Example NST2 NT-Shut-Down **WITH** shutdown boxes; the shutdown boxes appear for 30 seconds.

FI command	**XX_BW_NST2_30**

Formatting a Parameter Download Data File: PAF

MPCX Device Group

Designation **PAF** **PA**rameter **F**ile Converted

FI command By means of this FI command, an existing parameter download file can be re-formatted in such a way that the key names correspond to the parameter numbers. The structure of the download file corresponds to that of a Windows Ini file. Rexroth Indramat's own description in the document V20_Param_08_Definitions_Parameter_Download_01.doc is recommended for a more detailed account of the structure of the parameter download file.

BW_PAF1_(1)_(2)	**(Single Write)**
(1) = Complete parameter download file name **unsorted**	[input file] must be available
(2) = Complete parameter download file name **sorted**	[output file] is generated.

Note: A valid device address can also be passed as a write value. If no write value is passed, this command requires the MTCNC parameter download format.

Response Structure The following table shows the general structure of the response to the FI command "PAF1". The response consists of a line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 =	Status message (P_ACK)	(P_ACK)

Example PAF1 Generate on the basis of the unsorted parameter download file C:\TEMP\PARDATA.DAT the sorted parameter download file C:\TEMP\PARDATA.SOR.

FI command		**XX_BW_PAF1_"C:\TEMP\PARDATA.DAT"_"C:\TEMP\PARDATA.SOR"**
Line	**Column**	**Answer**
1	1	(P_ACK)

Generating Physical Directory Names: PHD

MPCX Device Group

Designation **PHD** **PH**ysical **D**irectory

Explanation Generates physical directory names according to the BDI data written.

Note: This is based on BDI philosophy.

FI command Generate physical directory names.

BR_PHD1_(1)_(2)_(3)_(4)_(5)_(6)	**(Single Write)**
(1) = Project ID	[-1= PROJECT_NEUTRAL -2= PROJECT_DEFAULT]
(2) = Section ID	[0= SECT_NEUTRAL 1= SECT_BIN 2= SECT_BASIC_DATA 3=SECT_OEM_DATA 4=SECT_CUSTOM_DATA 5=SECT_PROG_DATA]
(3) = Device address	[-1= DEVADDR_NEUTRAL otherwise the required device address]
(4) = Process ID	[-1= PROCESS_NEUTRAL otherwise the required process number]
(5) =Data type ID	[possible write values see BDI documentation (BDI_DEFINITIONS.H)]
(6) = Language ID	[possible write values see BDI documentation (WINNT.H)]

Response Structure The following table shows the general structure of the response to the FI command "PHD1".

Line 1	Column 1

Value Range/Meaning of Columns

1 =	Physical directory name	[complete physical directory name in accordance with the BDI data written]

Example PHD1 Requesting the physical directory name for:
PROJECT_NEUTRAL
SECT_BIN
DEVADDR_NEUTRAL
PROCESS_NEUTRAL
DATATYPE_NEUTRAL
LANG_NEUTRAL

FI command		**XX_BR_PHD1_-1_0_-1_-1_0_0**
Line	**Column**	**Answer**
1	1	D:\Programme\Indramat\Mtgui\Bin

Rexroth Indramat

Writing and Reading of a PC Port Address (Byte Access): POB

MPCX Device Group

Designation **POB** **PO**rt **B**yte Access

FI command This command is used for writing a PC port address (byte access).

BW_POB1_(1)_(2)	**(Single Read)**
(1) = requested PC port address	Declaration format: 0x port address
(2) = PC port value to be written	Declaration format: 0x port value

Response Structure The following table shows the general structure of the response to the FI command “POB1”. A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK)

Example POW1 Write the value 0x0000 into the PC port address 0x31C.

FI command		00_BW_POW1_0x31C_0x0000
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command This command is used for reading a PC port address (byte access).

BR_POB1_(1)	**(Single Read)**
(1) = requested PC port address	Declaration format: 0x port address

Response Structure The following table shows the general structure of the response to the FI command “POB1”. A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns 1= PC port value read

Example POB1 Read the PC port address 0x31C.

FI command		00_BR_POB1_0x31C
Line	**Column**	**Answer**
1	1	0x00

Writing and Reading of a PC Port Address (Word Access): POB

MPCX Device Group

Designation **POW** **PO**rt **W**ord Access

FI command This command is used for writing a PC port address (word access).

BW_POW1_(1)_(2) **(Single Read)**

(1) = requested PC port address Declaration format: 0x port address

(2) = PC port value to be written Declaration format: 0x port value

Response Structure The following table shows the general structure of the response to the FI command “POW1”. A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK) (P_ACK)

Example POW1 Write the value 0x0000 into the PC port address 0x31C.

FI command		00_BW_POW1_0x31C_0x0000
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command This command is used for reading a PC port address (word access).

BR_POW1_(1) **(Single Read)**

(1) = requested PC port address Declaration format: 0x port address

Response Structure The following table shows the general structure of the response to the FI command “POW1”. A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns 1= PC port value read

Example POW1 Read the PC port address 0x31C.

FI command		00_BW_POW1_0x31C
Line	**Column**	**Answer**
1	1	0x0000

Ready Message for a WIN32 Application: RP

MPCX Device Group R

Designation **RPR** **R**eady **PR**ocess

Explanation Using this FI command, WIN32 applications logged in the FI can inform the initiating program that they are ready for operation.

Note!
The process for the WIN32 application was generated by means of the FI command: "XX_BW_CPR1_(1)_(2)_(3)".

FI command Inform the initiating program that the WIN32 program invoked is ready for operation.

XX_BW_RPR1 **(Single Write)**

Response Structure As this concerns a command there is no response data.

Triggering an FI Device Polling Cycle: SDP

MPCX Device Group

Designation **SDP** **S**tart **D**evice **P**olling

Explanation This FI command triggers an FI device polling cycle.

FI command **BW_SDP1** **(Single Write)**

Response Structure The response to the "SDP1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status report (P_ACK)

Example SDP1 Triggering the FI device polling cycle.

FI command		XX_BW_SDP1
Line	**Column**	**Answer**
1	1	(P_ACK)

Focusing Commands: SFW

MPCX Device Group

Designation **SFW** **S**et **F**ocus to **W**indow

Explanation The screen can be focused with these FI commands.

FI command Focus the screen on the DOS-BOF user interface.

XX_BW_SFW1 **(Single Write)**

Response Structure As this concerns a command there is no response data.

Example SFW1 Focus on the DOD-BOF user interface screen.

Assumption:
This FI command is set from a WIN32 application and used to focus the display window of the DOS-BOF user interface currently running.

FI command	XX_BW_SFW1

FI command Focus the screen (target window) of a WIN32 application connected to the FI via LogInIf().

XX_BW_SFW2_(1)_(2)_(3)	(Single Write)
(1) = LogInIf Login names	This refers to the login name entered at the FI (LogInIf()) during the login procedure.
(2) = Min Info	Control information as to whether or not the current screen window (output window) is to be minimized. The following applies: 0 = do not minimize 1 = minimize
(3) = Wait Info	Control information as to how the output window is focused; here, the following applies: 0 = Re-focusing with the SFW2 command 1 = Automatic re-focusing on termination of the WIN32 application

Response Structure As this concerns a command there is no response data.

Example SFW2 Focus on the WIN32 application that has logged on in the FI with the login name "VBDEMO.EXE". The current screen window (output window) is minimized and refocusing takes place automatically at the end of the WIN32 application (VBDEMO.EXE).

FI command	XX_BW_SFW2_VBDEMO.EXE_1_1

Issuing a SYS Message: SSM

MPCX Device Group

Designation **SSM** **S**et **S**ys **M**essage

Explanation This allows SYS messages to be issued.

Note: The SYS message handling of the FI **MUST** BE known!

FI command This allows SYS messages to be issued. Acknowledgement must first be received from the WIN32 applications that want to receive the issued SYS message.

Note!
Any additional information with a maximum length of 200 characters can be transmitted simultaneously as a write value.

BW_SSM1_(1)_(2)	**(Single Write)**
(1) = SYS message number (ALWAYS an even number)	Value range: 2..4000 Note: The SYS message number is ALWAYS even, while the acknowledgement number associated with it is always an odd number!
(2) = input acknowledgement time in msec	Input acknowledgement time – the WIN32 applications that want to receive the SYS message MUST acknowledge the fact within this time.

Value to be written
Reference information

Response Structure The following table shows the general structure of the response to the FI command "SSM1".

Line 1	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Status report	[READY=SYS message has been correctly acknowledged by the WIN32 applications] [ERROR=SYS message has NOT been acknowledged by a WIN32 application within the pre-set time]
2 =	Task name (LogInIf name)	[Task name that has triggered the SYS message]
3 =	SYS message number	[contains the issued SYS message number]
4 =	Acknowledgement time	[contains the pre-set acknowledgement time]
5 =	Reference information	[contains, where applicable, the additional information transferred as a write value]
6 =	Length of additional information	[0 where NO additional information has been transferred]
7 =	Where applicable, LOG channel of the FI that has NOT acknowledged	[-- = acknowledgements have been completed in time or the LOG channel number of the WIN32 application that has NOT acknowledged in time]
8 =	Where applicable, task name that has	[-- = acknowledgements have been completed in time or the task name that

NOT acknowledged in time.	has NOT acknowledged in time]

Example SSM1

Issues SYS message 3302 with a pre-set acknowledgement time of 20000 msec. The additional information, device address 00, is also transferred as a write value.

FI command		**XX_BW_SSM1_3302_20000**
Line	**Column**	**Answer**
1	1	READY
	2	VBDEMO.EXE
	3	3302
	4	20000
	5	00
	6	2
	7	--
	8	--

FI command

This allows SYS messages to be issued **WITHOUT** the necessary acknowledgements.

Note!
Any reference information with a maximum length of 200 characters can be transmitted simultaneously as a write value.

BW_SSM2_(1)	**(Single Write)**
(1) = SYS message number (ALWAYS an even number)	Value range: 2..4000 Note: The SYS message number is ALWAYS even, while the acknowledgement number associated with it is always an odd number!

Value to be written
Reference information

Response Structure

The following table shows the general structure of the response to the FI command "SSM2".

Line 1	**Column 1**	**...**	**Column 8**

Value Range/Meaning of Columns

1 =	Status report	[READY=SYS message has been issued correctly]
2 =	Task name (LogInIf name)	[Task name that has triggered the SYS message]
3 =	SYS message number	[contains the issued SYS message number]
4 =	Acknowledgement time	[0]
5 =	Reference information	[contains, where applicable, the additional information transferred as a write value]
6 =	Length of additional information	[0 where NO additional information has been transferred]
7 =	Where applicable, LOG channel of the FI that has NOT acknowledged	[--]
8 =	Where applicable,	[--]

task name that has
NOT acknowledged in
time.

Example SSM2

Issues SYS message 3302 **WITHOUT** acknowledgement. The additional information, device address 00, is also transferred as a write value.

FI command		XX_BW_SSM2_3302
Line	**Column**	**Answer**
1	1	READY
	2	VBDEMO.EXE
	3	3302
	4	0
	5	00
	6	2
	7	--
	8	--

7.2 FI Commands for the MTCX Device Group

Com.	Description

The FI Commands described in this chapter are valid for the MTCX device group. In this device group, the following types are listed as well as possible device addresses:

Group	Accompanying Types	Address
MTCX	MTVNC	[00...63]

Note: Please note that the device address must be set before the respective FI command, e.g. 00_BR_ASM1 (refer also here to Chapter 6.1 “Elements of the FI Command”).

With a few exceptions, commands for the MWCX device group are also valid for the MTCX device group. The chapter “Overview of the FI Commands”, "Overview of the MTCX Device Group" contains a summary of the possible FI commands for the MTCX device group.

7.3 FI Commands for the MWCX Device Group

The FI Commands described in this chapter are valid for the MWCX device group. In this device group, the following types are listed as well as possible device addresses:

Group	Accompanying Types	Address
MWCX	MTC200-P-G2, MTC200-R-G2	[00...63]

Note: Please note that the device address must be set before the respective FI command, e.g. 00_CR_AAC_0 (refer also here to the chapter "Elements of the FI Command").

Active Acceleration Value: AAC

MWCX device group

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

Explanation The current acceleration value of an NC process is read out. Within an NC program, an acceleration limit can be programmed by means of the ”programmable acceleration ACC” function. This is the case when, for instance, the axes of the workpiece carrier is to be moved depending on the weight of the workpiece.

FI Command Output the active acceleration value of an CNC process of the selected device from the MWCX device group.

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

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

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

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

Response Structure The following table shows the general structure of the response to the FI command "AAC". One line with three columns is output for the NC command, the acceleration value and the unit.

Line 1	Column 1	....	Column 3

Value Range of the Columns

1 = NC command [ACC]

2 = Acceleration value [0...100]

3 = Unit [%]

Example AAC1 Reads the active acceleration value in NC process 0 of device address 00.

FI command	00_CR_AAC1_0		
Answer			
Line	Column 1	Column 2	Column 3
1	ACC	50	[%]

Reference to Literature See chapter entitled “Literature” [1].

Active Angle Dimension (RAD/DEG): AAD

MWCX device group

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

Explanation The active angle dimension of an NC process is read out. 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 both in "radiants" (RAD) as well as in "degrees" (DEG).

FI command Output the active angle dimension of an NC process of the selected device from the MWCX device group.

CR_AAD_(1)	**(Single Read)**
CC_AAD_(1)	**(Cyclic Read)**
CB_AAD_(1)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure The response to the FI command “AAD” consists of one line with one column for the unit [RAD/DEG].

Line 1	Column 1

Example AAD Reads the active angle dimension in NC process 0 of device address 00.

FI command		00_CR_AAD_0
Line	**Column**	**Answer**
1	1	RAD

Reference to Literature See chapter entitled “Literature” [2].

Actual (Current) Axis Speed (Spindle Speed): AAS

MWCX device group

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

Explanation The current axis speed and spindle speed of an NC process for the selected device are read out. The FI command "AAS1" refers to the NC process number and to the code of the axis meaning, whereas the FI command ”AAS2” allows the current speed to be queried in relation to the physical axis number.

FI command Output the current axis speed related to the NC process number and to the code of the meaning of the axis.

Using the optional third parameter it is possible to pre-select conversion of the result into mm/min or inch/min. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_AAS1_(1)_(2){_(3)}	**(Single Read)**
CC_AAS1_(1)_(2){_(3)}	**(Cyclic Read)**
CB_AAS1_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Axis meaning	[0...11] (see Chapter "Data Tables")
(3) = Required measurement system (opt.)	[mm, inch]

FI command

Output the current axis speed of the selected device related to the physical axis number.

Using the optional second parameter it is possible to pre-select conversion of the result into mm/min or inch/min. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_AAS2_(1){_(2)}	**(Single Read)**
CC_AAS2_(1){_(2)}	**(Cyclic Read)**
CB_AAS2_(1){_(2)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1...32, according to settings of the system parameters]
(2) = Required measurement system (opt.)	[mm, inch]

Response Structure

The following table shows the general structure of the response to the FI command "AASx" . One line is output with 4 columns for the axis designation, axis speed, unit and the axis speed limited to "indicated decimal places".

Line 1	Column 1	Column 2	Column 3	Column 4

Meaning of the Columns

1 = Axis name	[according to settings of axis parameters]
2 = Speed	[according to settings of axis parameters]
3 = Unit	[according to settings of process parameters and required measurement system]
4 = Speed	[as Column 2, but rounded up or down according to the parameter "indicated decimal places"]

Note: If the selected axis is not defined then the response in all columns is [--].

Example AAS1

Reads the current axis speed of the Z axis in NC process of device address 00.

FI command	**00_CR_AAS1_0_2**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	-1.2345	[mm/min]	-1.235

Example AAS1

Reads the current axis speed of the Z axis in NC process of device address 00. Output of values in inch/min.

FI command	**00_CR_AAS1_0_2_inch**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	-0.0486	[inch/min]	-0.049

Example AAS2

Reads the current speed of spindle S (e.g., physical axis number 4) of device address 00.

FI command	**00_CR_AAS2_4**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	S1	4000.0	1/min	4000.0

Reference to Literature

See chapter entitled "Literature" [3].

Active NC Block: ABI

MWCX device group

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

Explanation The active NC record or a user-defined NC block is read out. This allows an NC record display to be constructed with an active NC record as well as the number of the previous and following NC records.

FI command Output the active NC block as well as the previous and following NC blocks of an NC process for the selected device from the MWCX 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) = NC process number [0...6]

(2) = Number of previous NC blocks [1..4] ! Optional !

(3) = Number of following NC blocks [1..4] ! Optional !

Note: If the optional parameters are not specified then only the current NC record is output.

Response Structure The number of lines (1...n = 9) in the response depends on the number of NC records requested. Each line consists of a column containing the respective NC record.

Note: If there is no valid NC program in the device then the value of all columns is [--] .

Example ABI Reads the active NC record and the two previous and two following NC records of NC process 0 of device address 00.

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

Reference to Literature See chapter entitled “Literature” [4].

Active Cutting Speed of the Reference Spindle: ACS

MWCX device group

Designation **ACS** Active **C**utting **S**peed

Explanation Output of the active cutting speed of the reference spindle of an NC process for the selected device from the MWCX device group.

FI command

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

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

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

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

Response Structure The following table shows the general structure of the response to the FI command "ACS". One line with three columns is output for the S number of the reference spindle, the cutting speed and the unit according to the settings of the system parameters.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

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

Note: If no reference spindle is defined in the selected NC process then the value of Column 1 is [*S]; Columns 2 and 3 are given the value [--].

Example ACS Reads the active cutting speed in NC process 0 of device address 00.

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

Reference to Literature See chapter entitled “Literature” [5].

Active D-Correction Number: ADN

MWCX device group

Designation **ADN** Active **D**-Correction **N**umber

Explanation The active D-correction number of an NC process of the MWCX device group is output. The D-corrections are cumulative to the tool-geometry data of the register effecting the tool management.

FI command Output the active D-correction numbers of an NC process of the selected device from the MWCX device group.

FI command

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

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

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

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

Response Structure

One line with two columns is output for the active D-correction number of the indicated NC process. The meaning of the elements is as follows:

1 = Identifier [D]

2 = D-correction number [0] =De-selection of D-correction
[1..99] = Selection of D-correction

Example ADN

Read the active D-correction number of NC process 0 of device address 00.

FI command		00_CR_ADN1_0
Line	**Column**	**Answer**
1	1	D
	2	0

Reference to Literature

See chapter entitled “Literature” [6].

Active Event Monitoring: AEM

MWCX device group

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

Explanation

The status of the event monitoring of the specified NC process of the MWCX device group is output. Events are binary variables that can be used by the NC program; these variables represent any status defined by the programmer just like flags in the PLC program. Waiting for a defined status of an event therefore allows the possibility of process synchronization.

FI command

Output the status of the event monitoring of an NC process of the selected device from the MWCX device group.

CR_AEM_(1) **(Single Read)**

CC_AEM_(1) **(Cyclic Read)**

CB_AEM_(1) **(Break Cyclic Read)**

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

Response Structure

One line and one column are output for the status of the event monitoring. The meaning of the elements is as follows:

EEV = Activation of event monitoring

DEV = Suppressing of event monitoring

Example AEM

Read the status of the event monitoring of NC process 0 of device address 00.

FI command		00_CR_AEM_0
Line	**Column**	**Answer**
1	1	EEV

Reference to Literature

See chapter entitled “Literature” [7].

Active Edge Number: AEN

MWCX device group

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

Explanation The active edge number of an NC process is output. Changing the active cutter in the NC program results in the provision of the corresponding correction and tool life data which the tool management then accesses during subsequent processing.

FI command Output the active edge number of an NC process of the selected device from the MWCX device group.

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

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

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

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

Response Structure One line with two columns is output for the identifier "E = Edge" and for the active edge number. The active cutter corresponds to the single-digit decimal number [1...9] that is assigned the address letter "E".

Example AEN Read the active edge number of NC process 0 of device address 00.

FI command		**00_CR_AEN_0**
Line	**Column**	**Answer**
1	1	E
	2	1

Reference to Literature See chapter entitled “Literature” [8].

Active Feedrate Override: AFO

MWCX device group

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

Explanation The current value of the feedrate override of an NC process is outputted. Override is interpreted in the NC, irrespective of the mode; it has an effect on any axis movement (except on homing digital axes).

FI command Output the current value of the feedrate override of an NC process of the selected device from the MWCX device group.

CR_AFO1_(1) **(Single Read)**

CC_AFO1_(1) **(Cyclic Read)**

CB_AFO1_(1) **(Break Cyclic Read)**

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

Response Structure

The following table shows the general structure of the response to the FI command "AFO". One line with three columns is output for the identifier, the current value of the feedrate override and the unit [%].

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier	[OVR=Override]
2 = Current value of the feedrate override	[0...255]
3 = Unit	[%]

Note: The valid range of override weighting by the PLC program is between 0 and 255%. The NC limits the axis and/or processor speed to the maximum values set in the parameters if an override value is set that is too large.

Example AFO1

Reads the current value of the feedrate override in NC process 0 of device address 00.

FI command	00_CR_AFO1_0		
Answer			
Line	Column 1	Column 2	Column 3
1	OVR	60	[%]

Reference to Literature

See chapter entitled “Literature” [9].

Actual (Current) Feedrate: AFR

MWCX device group

Designation **AFR** Actual **F**eed **R**ate

Explanation

The current value of the feedrate of an NC process is output. The details of the feedrate in an NC program are expressed by means of a feedrate value with the address letter “F” and a feedrate that is input directly as a constant or by means of an expression.

FI command

Output the current value of the feedrate of an NC process of the selected device from the MWCX device group.

Using the optional second parameter it is possible to pre-select conversion of the result into mm or inches.

CR_AFR_(1){_(2)}	**(Single Read)**
CC_AFR_(1){_(2)}	**(Cyclic Read)**
CB_AFR_(1){_(2)}	**(Break Cyclic Read)**
(1) = NC Process number	[0...6]
(2) = Required measurement system (opt.)	[mm, inch]

Response Structure

The following table shows the general structure of the response to the FI command "AFR". One line with three columns is output for the identifier, the current value of the feedrate and the unit.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Identifier	[F = feedrate]
2 = Value	[format according to settings of the parameters]
3 = Unit	[according to settings of the process parameters]

Example AFR Reads the current feedrate in NC process 0 of device address 00.

FI command	**00_CR_AFR_0**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	F	30000.0	[mm/min]

Example AFR Reads the current feedrate in NC process 0 of device address 00. The displayed value is to be converted into inch/min:

FI command	**00_CR_AFR_0_inch**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	F	1181.1	[inch/min]

Reference to Literature see chapter entitled “Literature” [!)].

Active G Functions: AGF

MWCX device group

Designation **AGF** Active **G**-**F**unction

Explanation The active G functions of an NC process of the selected device from the MTCX device group are read out.

FI command

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

CC_AGF_(1){_(2)} **(Cyclic Read)**

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

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

(2) = G code group [1...21] ! Optional !

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

Response Structure One line is output, whereby the number of columns depends on the number of G code groups that are requested. If the optional parameter has <u>not</u> been specified, the response consists of one line with 21 columns. If the optional parameter has been specified then the response consists of one line with one column which contains the active G function of the selected G code group.

Note: In cases where no G function of the selected G code group is active, the response consists of the characters [--].

Example AGF Reads the active G function of G code group 17 in the NC process 0 of device address 00.

FI command		**00_CR_AGF_0_17**
Line	**Column**	**Answer**
1	1	G30

Reference to Literature See chapter entitled “Literature” [11].

Active M Functions: AMF

MWCX device group

Designation **AMF** Active **M**-**F**unction

Explanation The active M functions of an NC process of the selected device from the MWCX device group are read out.

FI command

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

CC_AMF_(1){_(2)} **(Cyclic Read)**

CB_AMF_(1){_(2)} **(Break Cyclic Read)**

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

(2) = M function Group [1...16] ! Optional !

Note: If the optional parameter is not specified then all active M functions of all M function groups are output.

Response Structure One line is output, whereby the number of columns depends on the number of M function groups that are requested. When the optional parameter has not been specified, the response consists of one line with 16 columns. If the optional parameter has been specified then the response consists of one line with one column which contains the active M function of the selected M function group.

Note: In cases where no M function of the selected M function group is active, the answer consists of the characters [--].

Example AMF Read the active M function of M function group 2 in NC process 0 of device address 00.

FI command		**00_CR_AMF_0_2**
Line	**Column**	**Answer**
1	1	M005

Reference to Literature See chapter entitled “Literature” [12].

Active Mechanism Messages: AMM

MWCX device group

Designation **AMM** Active **M**echanism **M**essages

Explanation Messages regarding active mechanism errors and mechanism diagnostics are output. These messages are assigned to a particular mechanism or process. Depending on the FI command, the device address, device name, mechanism number, mechanism name, type of message, message source, type of message (2), message number, short text and reference text are all output.

FI command Output mechanism messages currently pending for all active devices.

BR_AMM1 **(Single Read)**

BC_AMM1 **(Cyclic Read)**

BB_AMM1 **(Break Cyclic Read)**

Note:	The “AMM1” FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see Example AMM1).

Response Structure

The following table shows the general structure of the response to the FI command “AMM1”. The response consists of a maximum of n=512 lines (n=16 devices x 32 mechanisms = 512), each with 12 columns.

Line 1...n	Column 1	...	Column 12

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Type of message	[F = fault/error, D = diagnosis]
6 =	Message source	[CNC, PLC]
7 =	Type of message (2)	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Message text	[max. 54 ASCII characters]
10 =	Reference text	[x= exists, -- = does not exist]
11 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see AMM5)
12 =	Filename for additional information for message text	e.g. in HTML format

Example AMM1

Read the current mechanism messages of all active devices.
<u>Assumption:</u>
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		03_BR_AMM1
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S:
	8	79
	9	Station waiting until tool-change command has ended.
	10	x
	11	0
	12	

FI command		03_BR_AMM1
Line	**Column**	**Answer**
2	1	01
	2	Drill center
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V supply.
	10	x
	11	0
	12	
3	1	03
	2	Milling center
	3	0
	4	Camshaft 30.40.25.0S
	5	D
	6	CNC
	7	S:
	8	71
	9	Circular interpolation
	10	x
	11	0
	12	

FI command

Output the currently pending mechanism messages of the selected device.

BR_AMM2 **(Single Read)**

BC_AMM2 **(Cyclic Read)**

BB_AMM2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the FI command "AMM2". The response consists of up to a maximum of n=31 lines, each with 12 columns.

Line 1...n	Column 1	...	Column 12

Value Range/Meaning of Columns

1 = Device address [00...63]
2 = Device name [32 ASCII characters]
3 = Mechanism number [0...31]
4 = Mechanism name [max. 28 ASCII characters]
5 = Type of message [F = fault/error, D = diagnosis]
6 = Message source [CNC, PLC]
7 = Type of message (2) [S = Status, O = Operator, E = External, I = Internal]

8 =	Message number	[0...600]
9 =	Message text	[max. 54 ASCII characters]
10 =	Reference text	[x= exists, -- = does not exist]
11 =	2 byte additional information for the message number	is required to resolve the information “@” (see AMM5)
12 =	Filename for additional information for message text	e.g. in HTML format

Example AMM2

Reads the current mechanism messages of device address 01.
Assumption:
Device address 01 with 2 defined mechanisms 0 and 1.

FI command		**01_BR_AMM2**
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S:
	8	79
	9	Station waiting until tool-change command has ended.
	10	x
	11	0
	12	
2	1	01
	2	Drill center
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V supply.
	10	x
	11	0
	12	

Reference to Literature

See chapter entitled “Literature” [13].

FI command

Output the currently pending messages of the mechanisms listed for the selected device of the MWCX device group.

BR_AMM3_(1)	**(Single Read)**
BC_AMM3_(1)	**(Cyclic Read)**
BB_AMM3_(1)	**(Break Cyclic Read)**
(1) = Selection list for a max. of 10 mechanisms	[0_1_2_ ... _31]

Response Structure

The following table shows the general structure of the response to the FI command "AMM3". The number of lines (1 .. n=32) depends on the number of requested mechanism messages. Each line in turn consists of 12 columns.

Line 1...n	Column 1	...	Column 12

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Type of message	[F = fault/error, D = diagnosis]
6 =	Message source	[NC, SPS]
7 =	Type of message (2)	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Message text	[max. 54 ASCII characters]
10 =	Reference text	[x= exists, -- = does not exist]
11 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see AMM5)
12 =	Filename for additional information for message text	e.g. in HTML format

Reference to Literature

See chapter entitled “Literature” [13].

Example AMM3

Reads the current messages of mechanisms 0 and 1 of device address 01.

<u>Assumption:</u>
Device address 01 with 2 defined mechanisms 0 and 1.

FI command		**01_BR_AMM3_0_1**
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S:
	8	79
	9	Station waits until tool-change command has ended.
	10	x

FI command		01_BR_AMM3_0_1
Line	**Column**	**Answer**
	11	0
	12	
2	1	01
	2	Drill center
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V supply.
	10	x
	11	0
	12	

FI command

Output the currently pending messages of the mechanisms listed for the devices of the MWCX device group.

BR_AMM4_(1) **(Single Read)**

BC_AMM4_(1) **(Cyclic Read)**

BB_AMM4_(1) **(Break Cyclic Read)**

(1) = Selection list for a max. of 10 mechanisms [Format: x.y]

Format x.y	Value Range
X	Device address [00..0.63]
Y	Mechanism number [0...31]

Response Structure

The following table shows the general structure of the response to the FI command "AMM4". The number of lines (n=10 mechanisms, maximum) depends on the number of requested mechanism messages. Each line in turn consists of 12 columns.

Line 1...n	Column 1	...	Column 12

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Type of message	[F = fault/error, D = diagnosis]
6 =	Message source	[CNC, PLC]
7 =	Type of message (2)	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Message text	[max. 54 ASCII characters]
10 =	Reference text	[x= exists, -- = does not exist]
11 =	2 byte additional information for the message number	is required to resolve the information “@” (see AMM5)

Rexroth Indramat

12 =	Filename for additional information for message text	e.g. in HTML format

Reference to Literature See chapter entitled “Literature” [13].

Example AMM4 Reads the current messages of mechanisms 0 and 1 of device address 01 as well as 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**
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	0
	4	Station 1
	5	D
	6	CNC
	7	S:
	8	79
	9	Station waits until tool-change command has ended.
	10	x
	11	0
	12	
2	1	01
	2	Drill center
	3	1
	4	Station 2
	5	F
	6	CNC
	7	O
	8	1
	9	No external 24V supply.
	10	x
	11	0
	12	
3	1	03
	2	Milling center
	3	0
	4	Camshaft 30.40.25.0S
	5	D
	6	CNC
	7	S:
	8	71

FI command		01_BR_AMM4_01.0_01.1_03.0
Line	**Column**	**Answer**
	9	Circular interpolation
	10	x
	11	0
	12	

FI command

Device and mechanism related output of the reference text for the selected message number for the devices of the MWCX device group.

BR_AMM5_(1)_(2)_(3) **(Single Read)**

1 = Mechanism number [0...31]

(2) = Message number [0...600]

(3) = 2 bytes of additional information for the message number

Note: The third parameter of AMM5 is given as the 11th partial result of commands AMM1 ... AMM4

Response Structure

The following table shows the general structure of the response to the FI command "AMM5". The number of lines n=512 lines (n=16 devices x 32 mechanisms = 512) depends on the number of requested mechanism messages. Each line in turn consists of 10 columns.

Line 1...n	Column 1	...	Column 10

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0...31]
4 =	Mechanism name	[max. 28 ASCII characters]
5 =	Type of message	[F = fault/error, D = diagnosis]
6 =	Message source	[CNC, PLC]
7 =	Type of message (2)	[S = Status, O = Operator, E = External, I = Internal]
8 =	Message number	[0...600]
9 =	Reference text	[max. [max. 14 lines with a max. 78 characters/line]
10 =	Filename for additional information for reference text	e.g. in HTML format

Example AMM5

Reads the reference text for the required message number 79 of selected mechanism 0 for selected device 01.

FI command		01_BR_AMM5_0_79_0
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	0
	4	Station 1
	5	D
	6	CNC
	7	--
	8	79
	9	Station waits for completed execution of the active tool change command.
	10	

Reference to Literature

See chapter entitled “Literature” [13].

Active NC Memory Size: ANM

MWCX device group

Designation

ANM Active **N**C **M**emory Size

Explanation

The size of the active NC memory is read out.

FI command

CR_ANM **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command "ANM". A line with 3 columns is output for identification, size of the total NC memory, and largest free block:

Line 1	Column 1	Column 2	Column 3

Value Range/Meaning of Columns

1 = NC Memory Size [string]
2 = Total size of the NC memory [long]
3 = Largest free block of the NC memory [long]

Example: ANM

Read the size of the active NC memory.

FI command		00_CR_ANM	
		Answer	
Line	**Column 1**	**Column 2**	**Column 3**
1	NC Memory Size	654321	234567

Active Machine Parameter Index: API

MWCX device group

Designation **API** **A**ctive Machine-**P**arameter **I**ndex

Explanation Information regarding the active machine-parameter records of all defined devices of the MWCX device group are output. The following are output: the device addresses, index, GUI display, name, size, date and time of creation or of the last change and details of the defined processes of the active machine parameter record.

FI command

BR_API1 **(Single Read)**

BC_API1 **(Cyclic Read)**

BB_API1 **(Break Cyclic Read)**

Note: The “API1” FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see Example “API1”).

Response Structure The following table shows the general structure of the response to the FI command “API1”. The response consists of up to a maximum of n=16 lines, each with 8 columns.

Line 1...n:	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	Index of active parameter record	[0]
3 =	Identification string of the parameter record	[max. 84 ASCII characters]
4 =	Name of parameter record	[max. 32 ASCII characters]
5 =	Size of parameter record [byte]	[max. 8 ASCII characters]
6 =	Date of creation or of the last change in the parameter record.	[8 ASCII characters in format: DD.MM.YY]
7 =	Time of creation or of the last change in the parameter record.	[8 ASCII characters in format: HH:MM:SS]
8 =	Additional information (e.g. details of defined processes).	[max. 24 ASCII characters]

Note: In cases where there is no active machine parameter record in the device or where the active machine parameter record has been changed, Column 1 is given the device address and Columns 2 to 8 the value [--].

Example API1 Reads the information on the active machine parameter records of all defined devices.

Assumption:

The following device addresses of the MWCX device group have been defined:

Device address 00: MTC200-P,

Device address 01: MTC200-P, and

Device address 02: MTVNC.

FI command		01_BR_API1
Line	**Column**	**Answer**
1	1	00
	2	0
	3	00MSD 0209-15 15625 28.01.9913:29:10M123456
	4	MSD 0209-15
	5	15625
	6	28.01.99
	7	13:29:10
	8	M123456
2	1	01
	2	0
	3	88PCI 12.45.12.34 10584 11.11.9811:11:11M12
	4	PCI 12.45.12.34
	5	10584
	6	11.11.98
	7	11:11:11
	8	M12
3	1	02
	2	0
	3	11Lab 5 DRV 24464 01.03.9914:25:10M13456
	4	Lab 5 DRV
	5	24464
	6	01.03.99
	7	14:25:10
	8	M13456

Reference to Literature

See chapter entitled “Literature” [14].

FI command

BR_API2 **(Single Read)**
BC_API2 **(Cyclic Read)**
BB_API2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the FI command “API2”. The response consists of a line with eight columns.

Line 1	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Index of active parameter record	[0]
3 =	Identification string of the parameter record.	[max. 84 ASCII characters]
4 =	Name of parameter record	[max. 32 ASCII characters]
5 =	Size of parameter record [byte]	[max. 8 ASCII characters]
6 =	Date of creation or of the last change in the parameter record.	[8 ASCII characters in format: DD.MM.YY]

7 =	Time of creation or of the last change in the parameter record.	[8 ASCII characters in format: HH:MM:SS]
8 =	Additional information (e.g. details of defined processes).	[max. 24 ASCII characters]

Note: In cases where there is no active machine parameter record in the device or where the active machine parameter record has been changed, Column 1 is given the device address and Columns 2 to 8 the value [--].

Example API2

Reads the information on the active machine parameter record of device address 02.

Assumption:
The following device addresses of the MWCX device group have been defined:

Device address 00: MTC200-P,

Device address 01: MTC200-R, and

Device address 02: MTVNC.

FI command		02_BR_API2
Line	**Column**	**Answer**
1	1	02
	2	0
	3	00MSD 0209-15 15625 28.01.9913:29:10M123456
	4	MSD 0209-15
	5	15625
	6	28.01.99
	7	13:29:10
	8	M123456

Reference to Literature

See chapter entitled “Literature” [14].

Active Note in NC Program (Note and NC Record Number): APM

MWCX device group

Designation **APM** **A**ctive **P**art-Program **M**essage

Explanation

The active note of the NC record as well as the NC record number of an NC process of the MWCX device group is output. Every NC record can contain a note that is displayed in the diagnostics menu of the Rexroth Indramat GUI after the NC record has been processed. The note in the diagnostics line remains active until it is overwritten by a new note (also refer to "Active Note in NC Program (only NC Record Number): APN").

FI command

CR_APM_(1) **(Single Read)**

CC_APM_(1) **(Cyclic Read)**

CB_APM_(1) **(Break Cyclic Read)**

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

Response Structure

The following table shows the general structure of the response to the FI command "APM". One line with two columns is output for the NC record number and the NC note is output.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 = NC record number of the note [0000...9999]
2 = Note [max. 48 ASCII characters]

Note: If the current NC program does not contain a note, then the result of Column 1 is [0000] and that of Column 2 is [--].

Example APM Read the active note in the NC process of device address 00.

FI command		00_CR_APM_0
Line	**Column**	**Answer**
1	1	0002-{}-
	2	Technological instructions

Reference to Literature See chapter entitled “Literature” [15].

Active Note in NC Program (only NC Record Number): APN

MWCX device group

Designation **APN** **A**ctive **P**art-Program Message-**N**umber

Explanation The NC record number of the active note of an NC process of the MWCX device group is output. Every NC record can contain a note that is displayed in the diagnostics menu of the Rexroth Indramat GUI after the NC record has been processed. The note in the diagnostics line remains active until it is overwritten by a new note (also refer to chapter entitled “Literature”

FI command

CR_APN_(1) **(Single Read)**
CC_APN_(1) **(Cyclic Read)**
CB_APN_(1) **(Break Cyclic Read)**
(1) = NC process number [0...6]

Response Structure One line with one column is output for the NC record number of the active note.

Line 1	Column 1

Value Range/Meaning of Columns

1 = NC record number of the note [0000...9999]

Note: If the current NC program does not contain a note, then the result of Column 1 is [0000].

Example APN Read the NC record number of the active note in NC process 0 of device address 00.

FI command		00_CR_APN_0
Line	**Column**	**Answer**
1	1	0002

Reference to Literature See chapter entitled “Literature” [15].

Actual (Current) Position Value of an Axis: APO

MWCX device group

Designation **APO** **A**ctual Axis **PO**sition

Explanation The actual position of a selected axis is read out. The FI command "APO1" returns the position of an axis, related to the code of the axis meaning. On the other hand, the FI command "APO2" returns the position of an axis, related to the physical axis number.

FI command Output the position of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_APO1_(1)_(2)_(3){_(4)}	**(Single Read)**
CC_APO1_(1)_(2)_(3){_(4)}	**(Cyclic Read)**
CB_APO1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Axis meaning	[0...11] (see Chapter "Data Tables")
(3) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(4) = Required measurement system (opt.)	[mm, inch]

FI command Output the position of the selected axis of the device specified, related to the physical axis number.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_APO2_(1)_(2){_(3)}	**(Single Read)**
CC_APO2_(1)_(2){_(3)}	**(Cyclic Read)**
CB_APO2_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1...32, according to settings of the system parameters]
(2) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(3) = Required measurement system (opt.)	[mm, inch]

Response Structure The following table shows the general structure of the response to the FI commands “APO1” and “APO2”. One line is output with 4 columns for the axis designation, position, unit and the position limited to “indicated decimal places”.

Line 1	Column 1	Column 2	Column 3	Column 4

Value Range/Meaning of Columns

1 = Axis name	[according to settings of axis parameters]
2 = Position	[according to settings of process parameters]
3 = Unit	[according to settings of process parameters: mm, inch]
4 = Position	[as Column 2, but rounded up or down according to the parameter “indicated decimal places”]

Note: If the selected axis is not defined then the response in all columns is "--".

Example APO1

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in the basic measurement system.

FI command	00_CR_APO1_0_2_1			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z1	-1.2345	[mm]	-1.235

Example APO1

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in inches.

FI command	00_CR_APO1_0_2_1_inch			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z1	-0.0486	[inch]	-0.049

Example APO2

Reads the current position of the Z axis (physical axis number = 3) in machine coordinates for the device address 00. The values are displayed in the basic measuring system.

FI command	00_CR_APO2_3_1			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z1	-1.2345	[mm]	-1.235

Reference to Literature

See chapter entitled “Literature” [16].

Active NC Program Number: APP

MWCX device group

Designation

APP **A**ctive **P**art-**P**rogram Number

Explanation

The active NC program number of an NC process is read out.

FI command

CR_APP_(1) **(Single Read)**
CC_APP_(1) **(Cyclic Read)**
CB_APP_(1) **(Break Cyclic Read)**
(1) = NC process number [0...6]

Response Structure

The following table shows the general structure of the response to the FI command "APP". One line with 2 columns is output for the NC memory and the NC program number.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 = NC memory [A = memory A; B = memory B]
2 = NC program number [01...99]

Example APP Read the active NC program number in NC process 0.

FI command		00_CR_APP_0
Line	**Column**	**Answer**
1	1	A
	2	01

Reference to Literature See chapter entitled “Literature” [17].

Axis Reference Flags: ARF

MWCX device group

Designation **ARF** **A**xis **R**eference **F**lags

Explanation The reference flags for a process are to be displayed. These flags exist for the interpolation axes {X,Y,Z,U,V,W,A,B,C}

FI command

CR_ARF_(1) **(Single Read)**

CC_ARF_(1) **(Cyclic Read)**

CB_ARF_(1) **(Cyclic Break)**

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

Response Structure A line with 9 columns is output, each for the axis meaning: X, Y, Z, U, V, W, A, B, C axis.

An axis reference flag can have the following three values:

0 Axis not in reference

1 Axis in reference

-- Axis not present

Example ARF Displays the axis reference flags for process 0

<u>Assumption</u>:

- X, Y, Z axes are in reference,
- U, V, W axes are not in reference,
- A, B, C axes are not present

FI command		00_CR_ARF_0							
				Answer					
Line	**Col.1**	**Col.2**	**Col.3**	**Col.4**	**.5**	**.6**	**.7**	**.8**	**.9**
1	1	1	1	0	0	0	--	--	--

Actual (Current) Rapid Override: APO

MWCX device group

Designation **ARO** Actual **R**apid **O**verride

Explanation The current value of the rapid override of an NC process of the MWCX device group is output. This value is evaluated by the NC for all axis movements that are executed with “G00”. The valid range of override weighting by the PLC program is between 0 and 255%.

FI command Output the current value of the feedrate override of an NC process of the selected device from the MWCX device group.

CR_ARO1_(1)	**(Single Read)**
CC_ARO1_(1)	**(Cyclic Read)**
CB_ARO1_(1)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure The following table shows the general structure of the response to the FI command "ARO". One line with three columns is output for the identifier, the current value of the rapid override and the unit [%].

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Identifier	[ROV= rapid override]
2 = Current value of the rapid override	[0...255]
3 = Unit	[%]

Note: The valid range of override weighting by the PLC program is between 0 and 255%. The NC limits the axis and/or processor speed to the maximum values set in the parameters if an override value is set that is too large.

Example ARO1 Read the current value of the rapid override in NC process 0 of device address 00.

FI command	**00_CR_AFO1_0**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	ROV	100	[%]

Reference to Literature See chapter entitled “Literature” [18].

Axis Reference Table: ART

MWCX device group

Designation **ART** **A**xis **R**eference **T**able

Explanation The complete axis reference tables for a system are requested. They can be used to determine to which process the physical axes are assigned and with which axis meaning. They can also determine in which process dynamically assigned axes are possible.

FI command Read the axis reference tables of all processes 0-6. If the optional parameter is input, the output is limited to one process.

CR_ART1{_(1)} **(Single Read)**

(1) = NC process number (optional) [0...6]

Response Structure Seven lines are output (1 optional line) each with 12 columns and each having the axis number of the assigned axis. The first to twelfth column receives the physical axis number corresponding to the axis meaning X,Y,Z,U,V,W, A,B,C,S1,S2,S3. If a process does not have an axis for an axis meaning, then the result in this column is [- -].

If an axis can be assigned dynamically during operation, then the result in the first column is "*X" and the result in the other columns is "*Y" to "*S". If there is no process present at all, then the result in all columns for this line is [- -].

Line1...7	Column 1	Column 2	...	Column 12

With the following meaning:

Line 1, 2...7: Axis reference table for process 0, 1...6

Column 1, 2...12: Axis number for axis meaning X, Y...S3

Example ART1 Reads the complete axis reference table for device 00

Assumption:

Processes 0, 1 and 4 are present
and the axes:

1 (X axis in process 0),

2 (Y axis in process 0 or process 1),

3 (X axis in process 4),

4 (S axis in process 0 or process 4),

FI command	**00_CR_ART1**			
		Answer		
Line	**Column 1**	**Column 2**	**...**	**Column 12**
1	1	2	...	--
2	--	*Y	...	4
3	--	--	...	--
4	--	--	...	--
5	3	--	...	*S
6	--	--	...	--
7	--	--	...	--

Example ART1 Reads the axis reference table for process 1 of device 00:

FI command	00_CR_ART1_1			
Answer				
Line	Column 1	Column 2	...	Column 12
1	--	*Y	...	4

FI command

For a device which is offline (DeviceStatus=OFF), the axis reference data is simulated according to the current parameter record.

BR_ART **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command "ART". A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Binary axis reference table data

Example: ART

Read the binary axis reference table data of the device 00.

FI command	00_BR_ART	
Line	Column	Answer
1	1	Read axis reference table data

Actual (Current) Spindle Data: ASD

MWCX device group

Designation

ASD Actual **S**pindle **D**ata

Explanation

The current spindle data of an NC process of the selected device from the MWCX device group is read out.

This command is a compilation of PSS, ASS, MSS, ASO, MSO and ASG.

FI command

Output the current axis data of an NC process related to the spindle number.

CR_ASD_(1)_(2) **(Single Read)**

CC_ASD_(1)_(2) **(Cyclic Read)**

CB_ASD_(1)_(2) **(Break Cyclic Read)**

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

(2) = Spindle number [1...3]

Response Structure

The following table shows the general structure of the response to the FI command "ASD". A line with 9 columns is output for axis denomination, current spindle speed, programmed spindle speed, maximum spindle speed, and the unit according to settings of the process parameters, current spindle override, maximum spindle override, and the current gear level.

Line 1	Column 1	...	Column 9

Value Range/Meaning of Columns

1 = Axis name	[S, S1, S2, S3]
2 = Current spindle speed	[according to settings of axis parameters]
3=programmed Spindle speed	[according to settings of axis parameters]
4= max. spindle speed	[according to settings of axis parameters]
5 = Unit	1/min

6 = Current spindle override	[0 ... MAXSOVR]
7= max. spindle override	[according to settings of axis parameters]
8 = Gear identifier	[g]
9 = Current gear level	[1 ... 3, --]

Note: If the selected spindle is not defined in the selected NC process, the result contains the value "--" in all the columns; if it is presently not assigned, "*S" will appear in the first, and "--" in the remaining columns.

Example: ASD

Read the current data of the 1st spindle in NC process 0 of device address 00.

FI command	00_CR_ASD_0_1		
Answer			
Line	**Column**	**Value**	Meaning
1	1	S1	Axis Designation
	2	3000.0	Progr. Spindle speed
	3	2999.9	Current spindle speed
	4	5000.0	Maximum spindle speed
	5	1/min	Unit
	6	100%	Current spindle override
	7	120%	Maximum spindle override
	8	g	Gear level identifier
	9	1	Current gear level

Reference to Literature

See chapter entitled “Literature” [21].

See chapter entitled “Literature” [22].

Active Spindle for Process: ASF

MWCX device group

Designation **ASF** **A**ctive **S**pindle **F**or Process

Explanation

The active (selected) spindle of the selected NC process is output. As there can be several spindles in an NC process, it is necessary for certain NC functions such as G96 (constant cutting speed), that these are active on another spindle as well as on the first spindle. The following NC functions are dependent on the selected main spindle:

- G33 thread cutting
- G63/G64 tapping
- G65 tapping; spindle serves as leading axis
- G95 feed per turn and
- G96 constant cutting speed.

FI command

CR_ASF_(1)	**(Single Read)**
CC_ASF_(1)	**(Cyclic Read)**
CB_ASF_(1)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure

The response to the FI-Command “ASF” consists of one line with one column for the selected active spindle.

Active Spindle for Process: [S1, S2, S3, *S]

Note: If no active spindle is selected in the NC process, then the response for Column 1 is [*S].

Example ASF

Reads the selected active spindle in an NC process 0 of device address 00.

Assumption:

- A main circular-axis spindle (S1) has been defined in NC process 0,
- The spindle has been selected as active spindle by the NC command "SPF 1" and
- The G function "G96" is active in the NC program.

FI command		00_CR_ASF_0
Line	**Column**	**Answer**
1	1	S1

Reference to Literature

See chapter entitled “Literature” [19].

Actual (Current) Spindle Gear Level: ASG

MWCX device group

Designation **ASG** Actual **S**pindle **G**ear

Explanation

The current spindle gear level of an NC process of the selected device from the MWCX device group is read out. The control signals of the gear selection are only evaluated by the CNC when one gear with at least two gear levels has been entered within 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) = NC process number [0...6]

(2) = Spindle number [1...3]

Response Structure

The response to the “ASG” FI command consists of one line with two columns for the identifier and for the current spindle gear level of the selected NC process.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 = Identifier [g = gear]

2 = Current spindle gear level [1...3, -]

Note: If no current spindle gear level is selected in the NC process or in the NC program then Column 1 receives the value [g] and Column 2 the value [-].

Example ASG

Read the current spindle gear level of the 1st spindle in NC process 0 of device address 00.

FI command		00_CR_ASG_0_1
Line	**Column**	**Answer**
1	1	g
	2	1

Reference to Literature See chapter entitled “Literature” [20].

Active System Error Messages: ASM

MWCX device group

Designation **ASM** **A**ctive **S**ystem **M**essages

Explanation The active system error messages that effect the functioning of the entire electrical device are output Depending on the FI command, the device address, device name, message number, type of message, short text and reference text are all output.

FI command Output the current system error messages pending for all active devices from the MWCX device group.

BR_ASM1 **(Single Read)**

BC_ASM1 **(Cyclic Read)**

BB_ASM1 **(Break Cyclic Read)**

Note: The “ASM1” FI command refers to all devices within this device group. This means that any valid device address can be indicated in the command line (see Example ASM1).

Response Structure The following table shows the general structure of the response to the FI command “ASM1”. The number of lines (1 .. n=15) depends on the number of defined devices. Each line consists of 8 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Message text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see ASM5)
8 =	Filename for additional information for message text	e.g. in HTML format

Example ASM1 Reads the current system error messages of all defined devices of the MWCX device group.

<u>Assumption:</u>
The following three devices are defined:

- Device address 01
- Device address 07 and
- Device address 10.

FI command		07_BR_ASM1
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	
2	1	07
	2	Milling center 1
	3	74
	4	F
	5	SLM time monitoring
	6	X
	7	0
	8	
3	1	10
	2	Milling center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0
	8	

FI command

Output the current system error message that is pending for the selected device from the MWCX device group.

BR_ASM2 **(Single Read)**

BC_ASM2 **(Cyclic Read)**

BB_ASM2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the FI command "ASM2". The answer consists of a line of 8 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Message text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]

7 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see ASM5)
8 =	Filename for additional information for message text	e.g. in HTML format

Example ASM2

Read the current system error 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
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	

FI command

Output the current system error messages of the device listed from the MWCX device group.

BR_ASM3_(1) **(Single Read)**

BC_ASM3_(1) **(Cyclic Read)**

BB_ASM3_(1) **(Break Cyclic Read)**

(1) = Selection list for a max. of 10 MWCX devices [00_01_ ... _15]

Response Structure

The following table shows the general structure of the response to the FI command “ASM3”. The number of lines (1 .. n=15) depends on the number of MWCX devices listed. Each line consists of 8 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Message text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 byte additional information for the message number	is required to resolve the information “@” (see ASM5)
8 =	Filename for additional information for message text	e.g. in HTML format

Example ASM3

Reads the current system error messages of the selected MWCX devices.

Assumption:
The following devices addresses are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		01_BR_ASM3_01_10
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	
2	1	10
	2	Milling center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0
	8	

FI command

Output the current system error messages of all defined devices (in accordance with the system configuration) from the MWCX device group.

BR_ASM4_(1)	**(Single Read)**
BC_ASM4_(1)	**(Cyclic Read)**
BB_ASM4_(1)	**(Break Cyclic Read)**
(1) = Device group	[MWCX, MISX]

Response Structure

The following table shows the general structure of the response to the FI command “ASM4”. The number of lines (1 .. n=15) depends on the number of MWCX devices defined. Each line consists of 8 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Message text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information	is required to resolve the information “@” (see ASM5)

	for the message number	
8 =	Filename for additional information for message text	e.g. in HTML format

Example ASM4

Reads the current system error messages of all defined devices of the MWCX device group.
Assumption:
The following devices addresses are defined:

- Device address 01 and
- Device address 10.

FI command		01_BR_ASM4_MWCX
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	
2	1	10
	2	Milling center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0
	8	

FI command

Output the reference text for the currently pending error message, related to the device and the message number.

BR_ASM5_(1)_(2) **(Single Read)**

(1) = Message number [0...150]

(2) = 2 bytes of additional information for the message number

Response Structure

The following table shows the general structure of the response to the FI command “ASM5”. The response consists of a line with 6 columns for device address, device name, message number and reference text.

Line 1...n	Column 1	...	Column 6

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Reference text	[max. [max. 14 lines with a max. 78 characters/line]
6 =	Filename for additional	e.g. in HTML format

information for reference text

Example ASM5 Read the reference text relating to the system error with message number 74 of device address 01.

FI command		**01_BR_ASM5_74_0**
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	74
	4	F
	5	Replace the SLM module on the PLC card (slot 3).
	6	

Reference to Literature See chapter entitled “Literature” [13].

Actual (Current) NC Sequence Number: ASN

MWCX device group

Designation **ASN** **A**ctual **S**equence **N**umber

Explanation The active NC sequence number of an NC process of the selected device from the MWCX device group is output.

FI command

CR_ASN_(1) **(Single Read)**
CC_ASN_(1) **(Cyclic Read)**
CB_ASN_(1) **(Break Cyclic Read)**
(1) = NC process number [0...6]

Response Structure The response to the “ASN” FI command consists of one line with one column for the active NC sequence number [N0000...N9999].

Line 1	**Column 1**

Note: If no valid NC program exists then Column 1 receives the value [N0000].

Example ASN Read the active NC sequence number of NC process 0 of device address 00.

FI command		**00_CR_ASN_0**
Line	**Column**	**Answer**
1	1	N0002

Reference to Literature See chapter entitled “Literature” [4].

Actual (Current) Spindle Override: ASO

MWCX device group

Designation **ASO** **A**ctual **S**pindle **O**verride

Explanation The current value of the spindle override of an NC process of the MWCX device group is output. Override is valid for all non-interpolating axes (i.e. for spindle axes and magazine axes). Override is interpreted in the NC, irrespective of the mode; it has an effect on any axis movement (except on homing digital axes).

FI command Output the current value of the override of the selected device of the MWCX device group related to the NC process and the spindle number.

CR_ASO1_(1)_(2) **(Single Read)**

CC_ASO1_(1)_(2) **(Cyclic Read)**

CB_ASO1_(1)_(2) **(Break Cyclic Read)**

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

(2) = Spindle number [1...3]

Response Structure The following table shows the general structure of the response to the FI command “ASO1”. One line with three columns is output for the identifier, the current value of the override and the unit [%].

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Identifier [S= Spindle]

2 = Current value of the override with unit [0...255]

3 = Unit [%]

Note: The valid range of override weighting by the PLC program is between 0 and 255 %. The NC limits the axis and/or processor speed to the maximum values set in the parameters if an override value is set that is too large.

If the spindle number is not defined within the selected process then the result in Column 1 is [--].

Example ASO1 Read the current value of the override of Spindle 1 in NC process 0 of device address 00.

FI command	00_CR_ASO1_0_1		
Answer			
Line	Column 1	Column 2	Column 3
1	S:	60	[%]

Reference to Literature See chapter entitled “Literature” [21].

Actual (Current) Spindle Speed: ASS

MWCX device group

Designation **ASS** **A**ctual **S**pindle **S**peed

Explanation The current spindle speed (axis velocity) of an NC process of the selected device from the MWCX device group is read out.

FI command Output the current axis speed of an NC process related to the spindle number.

CR_ASS_(1)_(2)	**(Single Read)**
CC_ASS_(1)_(2)	**(Cyclic Read)**
CB_ASS_(1)_(2)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Spindle number	[1...3]

Response Structure The following table shows the general structure of the response to the FI command "ASS". One line with three columns for the name of the axis, the axis speed and the unit is output in accordance with the settings of the process parameters.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Axis name	[S, S1, S2, S3]
2 = Spindle speed	[according to settings of axis parameters]
3 = Unit	[1/min; according to parameter setting]

Note: If the spindle number is not defined in the selected NC process, then the result in Column 1 is [--], the result in Column 2 is [0.0] and that in Column 3 is [1/min].

Example ASS Read the current axis speed of the 1st spindle in NC process 0 of device address 00.

FI command	**00_CR_ASS_0_1**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	S1	4000.0	1/min

Reference to Literature See chapter entitled “Literature” [22].

Active Tool Number: ATN

MWCX device group

Designation **ATN** **A**ctive **T**ool-**N**umber

Explanation The active tool number of an NC process of the selected device from the MWCX device group is read out.

FI command

CR_ATN_(1)	**(Single Read)**
CC_ATN_(1)	**(Cyclic Read)**
CB_ATN_(1)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure The response for the "ATN" FI command consists of one line with two columns for the identifier and the number of the active tool.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 = Identifier	[T = tool]
2 = Number of active tool	[1...9999999]

Note: If no tool is active in the selected NC process then Column 1 receives the value [T] and Column 2 the value [0].

Example ATN Read the number of the active tool in NC process 0 of device address 00.

FI command		00_CR_ATN_0
Line	**Column**	**Answer**
1	1	T
	2	4

Reference to Literature See chapter entitled "Literature" [23].

Reading Actual (Current) Tool Place Information: ATP

MWCX device group

Designation **ATP** **A**ctual **T**ool **P**lace Information

Explanation Information regarding the tool place and the current edge of the pre-selected tool is returned by the "ATP" command. The control unit response telegram also returns information on the current position of the tool magazine. For this reason, the "ATP" access has 3 filter options. The following information is returned by the control unit upon the FI command "ATP":

- ATP1 Set and actual position of the tool magazine and edge place information for the active tool
- ATP2 Edge and place information for the active tool.
- ATP3 Set and actual position of the tool magazine.

The FI command refers to the indicated NC process. If the control unit is not able to return any data, then the corresponding partial result [--] is transmitted.

FI command

Set and actual position of the tool magazine and edge place information of the active tool

CR_ATP1_(1)	**(Single Read)**
CC_ATP1_(1)	**(Cyclic Read)**
CB_ATP1_(1)	**(Break_Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure

The following table shows the general structure of the response to the "ATP1" FI command . One line with 4 columns is output for the returned values.

Line 1	Column 1	...	Column 4

Value Range/Meaning of the Columns

1 =	Set position of magazine	[1...999]	
2 =	Actual position of magazine	[1...999]	
3 =	Active tool edge number	[1...9]	
4 =	Tool place (type + place number)	[Mx= magazine/turret Sx = spindle Gx = gripper	[x=1...999] [x=1...4] [x=1...4]]

Note: Details of the current command and actual position of the tool magazine refer to the reference point of the magazine controller.

Example ATP1

Read the position of the tool magazine plus edge and tool-place information for the active tool from NC process 0 of device 00.

FI command		**00_CR_ATP1_0**
Line	**Column**	**Answer**
1	1	3
	2	3
	3	1
	4	S1

FI command

Edge and place information for the active tool.

CR_ATP2_(1)	**(Single Read)**
CC_ATP2_(1)	**(Cyclic Read)**
CB_ATP2_(1)	**(Break_Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure

The following table shows the general structure of the response to the "ATP2" FI command. One line with 2 columns is output for the returned values.

Line 1	Column 1	Column 2

Value Range/Meaning of the Columns

1 =	Active tool edge number	[1...9]	
2 =	Tool place (type + place number)	[Mx= magazine/turret Sx = spindle Gx = gripper	[x=1...999] [x=1...4] [x=1...4]]

Example ATP2

Reads the edge and tool place information of the active tool from NC process 0 of device 00.

FI command		00_CR_ATP2_0
Line	**Column**	**Answer**
1	1	1
	2	S1

FI command

Output of the position information of the tool magazine.

CR_ATP3_(1) **(Single Read)**

CC_ATP3_(1) **(Cyclic Read)**

CB_ATP3_(1) **(Break_Cyclic Read)**

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

Response Structure

The following table shows the general structure of the response to the "ATP3" FI command. One line with 2 columns is output for the returned values.

Line 1	Column 1	Column 2

Value Range/Meaning of the Columns

1 = Command position of magazine [1...999]

2 = Actual position of magazine [1...999]

Note: Details of the current command and actual position of the tool magazine refer to the reference point of the magazine controller.

Example ATP3

Read the command and actual position of the tool magazine from NC process 0 of device 00.

FI command		00_CR_ATP3_0
Line	**Column**	**Answer**
1	1	3
	2	3

Reference to Literature

See chapter entitled "Literature" [8].

Access to Actual (Current) Tool Data Record: ATR

MWCX device group

Designation

ATR Actual **T**ooldata **R**ecord

Explanation

Returns a complete basic data record and/or cutter data record of the current processing tool.

FI command

Read basic data record or cutter data record of the current tool.

CR_ATR_(1)_(2) **(Single Read)**

CC_ATR_(1)_(2) **(Cyclic Read)**

CB_ATR_(1)_(2) **(Break Cyclic Read)**

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

(2) = Data record [0 = base tool data, 1...9 = cutter data]

Response Structure

The following table shows the general structure of the response to the "CR_ATR" FI command. One line is output with 28 (basic data) or 40 (cutter data) columns for the returned values.

Line 1	Column 1	...	Column 28/40

Value Range/Meaning of the Columns

1...28 =	Requested base tool data	[max. 28 data elements] (see basic value range data)
1...40 =	Requested tool cutter data	[max. 40 data elements] (see value range of cutter data)

Example TDR1

Read the base tool data record in NC process 0 of the tool currently processing.

FI command		00_CR_ATR_0_0
Line	**Column**	**Answer**
1	1	928
	2	Miller D20
	3	S:
	4	1
	5	1234567
	6	1234
	7	2
	8	1
	9	..............+..p............
	10	0
	11	M1
	12	M
	13	--
	14	M
	15	--
	16	[cycl]
	17	[mm]
	18	4
	19	102
	20	0.000000
	21	0.000000
	22	0.000000
	23	0.000000
	24	0.000000
	25	0.000000
	26	0.000000
	27	0.000000
	28	0.000000

Reference to Literature

See chapter entitled “Literature” [8].

Accepting the Data Record for the Current Tool: ATU

MWCX device group

Designation **ATU** **A**ctual **T**ooldata **U**pdate

Explanation The current tool data record that has been changed following editing is accepted by the CNC.

FI command **CR_ATU_(1)** **(Single Read)**

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

Response Structure One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge New data record of current tool has been accepted

Example ATU Accept the changed data record of the current tool in NC process 0 of device address 00.

FI command		00_CR_ATU_0
Line	**Column**	**Answer**
1	1	(P_ACK)

Active Zero-Offset Bank: AZB

MWCX device group

Designation **AZB** **A**ctive **Z**ero-**O**ffset Bank

Explanation The number of the active zero-offset bank of an NC process of the selected device from the MWCX device group is read out.

The zero offsets allow the origin of a coordinate axis to be shifted (offset) by a set value, related to the original position of the machine. A record of these shifts is held in the zero-offset banks.

FI command **CR_AZB1_(1)** **(Single Read)**

CC_AZB1_(1) **(Cyclic Read)**

CB_AZB1_(1) **(Break Cyclic Read)**

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

Response Structure The response to the “AZB1” FI command consists of one line with two columns for the identifier (O = offset) and the number of the active zero-offset bank [0...2].

Line 1	Column 1	Column 2

Example AZB Read the number of the active zero-offset bank in NC process 0 of device address 00.

FI command		00_CR_AZB1_0
Line	**Column**	**Answer**
1	1	O
	2	2

Reference to Literature See chapter entitled “Literature” [24].

NC Cycle Download: CCA

MWCX device group

Designation **CCA** N**C**-**C**ycle **A**ccess

Explanation NC cycles are downloaded by means of the download file and NC cycle files via all active processes.

FI command NC Cycle Download.

BW_CCA1_(1) **(Single Write)**

(1) = Download file with path details.

Note: File and path details must be enclosed in inverted commas.

Response Structure The response to the "CCA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID [01...20]
(see Chapter "FI Commands for the MPCX Device Group: IFJ").

Line 2 = FI command
[String, in accordance with Chapter "Elements of the FI Command"]

Line 3 = FI Job Error Code
(see Chapter "Error Codes")

Example CCA1 00_BW_CCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		**00_BW_CCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_CCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
3	1	0

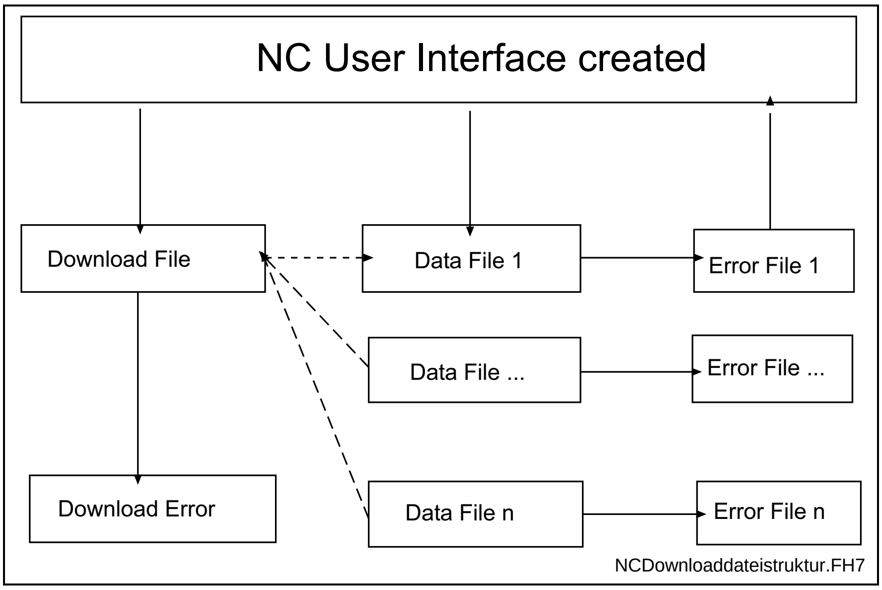


Fig. 7-1: File structure of the download file

Structure of Download File The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
General information is stored in the “Common” section.

Key **Max_Compiler**
Number of compilers to be called. The compiler contains the control file as a pass parameter and translates the data into the respective data files. A pass value of zero signifies no compiler call.

This key is an optional value. If this value is not present, no compiler is active.

Key **DownloadError**
Indicates whether or not an error has occurred during downloading. This value is only set in the event of an error.

<u>Example:</u>

[Common]
DownloadError = YES ; Error
Max_Compiler = 2

Section [CompilerXX]
This section contains information regarding the compiler. There is a separate section for each compiler. The name of the section consists of the “Compiler” text and a two digit number.

XX: is a two digit index which begins at 1 and has a maximum size of Max_Compiler.

Section [CycPackage_Info]

Key **Cycle package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters
Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters
Package date: "**PackageDate**" max. 8 characters
Package default:"**PackageDefault**" max. 26 characters (optional)

Total: max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

<u>Example:</u>

[CycPackage_Info]
PackageNo = 1
PackageName = NC program package
PackageSize = 1234
PackageTime = 13:10:10
PackageDate = 24.12.00

Section list of NC cycle programs [ListOfCycPrograms]
The list of the NC cycle programs to be transferred is stored in the section "ListOfCycPrograms".

Key **Max_Index_Data**
Corresponds to the number of NC cycle programs to be transferred.

Key **consecutive index of the NC cycle programs**
Four-digit number starting with 1, identifies with a value the full file name of the NC cycle programs:

zzzzzzz	Data type (CYC-PRG)
xx	Process number
yyy	Program number of the cycle programs (1 – 255)

The file extension can be freely selected. ".dat" has been used in the following example.

Examples:

CYC-PRG-00-086	Cycle program for process 0 program 86
CYC-PRG-01-001	Cycle program for process 1 program 1

Example:

[ListOfCycPrograms]

Max_Index_Data=50

0001=K:\Program Files\Indramat\Mtgui\Project_000\CYC-PRG-00-01.dat
0002=K:\Program Files\Indramat\Mtgui\Project_000\CYC-PRG-01-01.dat
...
0050=K:\Program Files\Indramat\Mtgui\Project_000\CYC-PRG-06-99.dat

Data File Structure

These files contain the data for downloading and for the compiler. Their structure corresponds to the Windows "Ini" structure. The compiler uses this file for the input and output data.

Note: Care must be taken in the use of upper and lower case letters.

Data for the NC program is stored in the respective files as a section. It is composed of general data and the actual program.

Section [Common]

Program version:	**Version**	
Process:	**Process**	[0..6]
Program number:	**No**	[0..255]
Program name:	**Name**	max. 32 characters
Program size:	**Size**	
Program time:	**Time**	max. 8 characters
Program date:	**Date**	max. 8 characters
Program short identification:	**ShortID**	max. 8 characters
Program status:	**Status, (optional)**	

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Status flag	Description
C	Compiled
E	Error
The marked section is then printed out.	Not compiled
No details	No compiler call

Fig. 7-2: Status flags

Section Data

Key **Max_Index_Data**
Corresponds to the number of NC blocks to be transmitted

Key **consecutive index of NC records**
Five-digit number starting with 00001.

Note: An NC block should not contain any unnecessary blank spaces or NC comments. Equally, “PROGRAM END” may not occur as it is language-dependent.

Example:

[Data]
Max_Index_Data=25
00001=N0000 G0 X0 Y0 Z0
...
00025=N0024 .Start

NC Cycle Upload: CCA

MWCX device group

Designation **CCA** N**C**-**C**ycle **A**ccess

Explanation NC cycles are uploaded via all active processes. During upload, a basic file (upload file) and an NC cycle file are created.

FI command NC cycle upload.

BR_CCA1_(1) **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.

Response Structure The response to the CCA1 FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
 (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
 [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
 (see Chapter "Error Codes")

Example: CCA 00_BR_CCA1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_CCA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_CCA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini" /3
3	1	0

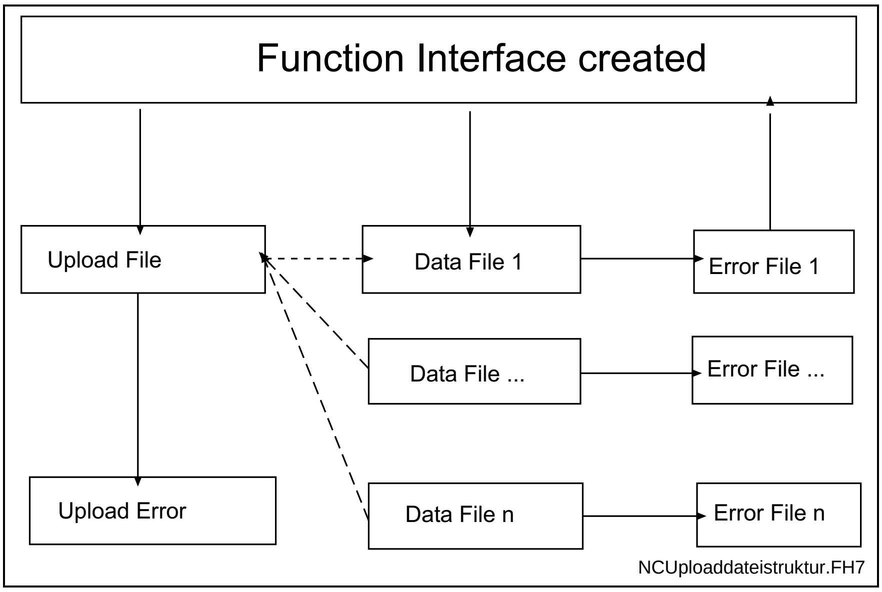


Fig. 7-3: File structure of the upload file

Structure of Upload File

The upload file is structured in the Windows – "Ini" format structure.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
General information is stored in the "Common" section.

Key **UploadError**
Indicates whether or not an error has occurred during uploading. This value is only set in the event of an error.

Example:

[Common]
UploadError = YES ; error

Section NC cycles package information [CycPackage_Info]

Key **Cycle package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters
Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters
Package date: "**PackageDate**" max. 8 characters
Package default:"**PackageDefault**" max. 26 characters (optional)

Total: max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

[CycPackage_Info]
PackageNo = 1
PackageName = NC program package

PackageSize =	1234
PackageTime =	13:10:10
PackageDate =	24.12.00

Section list of NC cycle programs [ListOfCycPrograms]
The list of the NC cycle programs to be transferred is stored in the section "ListOfCycPrograms".

Key **Max_Index_Data**
Corresponds to the number of NC cycle programs to be transferred.

Key **consecutive index of the NC cycle programs**
Four-digit number starting with 1, identifies with a value the full file name of the NC cycle programs:

zzzzzzz	Data type (CYC-PRG)
xx	Process number
yyy	Program number of the cycle programs (1 – 255)

The file extension can be freely selected. “.dat” has been used in the following example.

Examples:

CYC-PRG-00-086	Cycle program for process 0 program 86
CYC-PRG-01-001	Cycle program for process 1 program 1

Example:

```
[ListOfCycPrograms]
Max_Index_Data=50
0001=K:\Program Files\Indramat\Mtgui\Project_000\CYC-PRG-00-001.dat
0002=K:\Program Files\Indramat\Mtgui\Project_000\CYC-PRG-01-001.dat
...
0050=K:\Program Files\Indramat\Mtgui\Project_000\CYC-PRG-06-099.dat
```

Data File Structure

Contains the actual data for the upload. Their structure corresponds to the Windows “Ini” structure.

Note: Care must be taken in the use of upper and lower case letters.

Data for the cycle program is stored in the respective files as a section. It is composed of general data and the actual program.

Section [Common]

Program version:	**Version**	
Process:	**Process**	[0..6]
Program number:	**No**	[0..255]
Program name:	**Name**	max. 32 characters
Program size:	**Size**	
Program time:	**Time**	max. 8 characters
Program date:	**Date**	max. 8 characters
Program short identification:	**ShortID**	max. 8 characters
Program status:	**Status, (optional)**	

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Status flag	Description
C	Compiled
E	Error
The marked section is then printed out.	Not compiled
No details	No compiler call

Fig. 7-4: Status flags

Section [Data]

Key **Max_Index_Data**
Corresponds to the number of NC blocks to be transmitted

Key **consecutive index of NC records**
Five-digit number starting with 1.

Example:

[Data]
Max_Index_Data=25
00001=N0000 G0 X0 Y0 Z0
...
00025=N0024 .Start

Position Set point of an Axis: CPO

MWCX device group

Designation **CPO** **C**ommand **PO**sition

Explanation The actual position set point of a selected axis is read out. The "CPO1" FI command returns the command value of an axis related to the code of the axis meaning. The "CPO2" FI command, on the other hand, returns the command value of an axis related to the physical axis number.

FI command Output the command value of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_CPO1_(1)_(2)_(3){_(4)}	**(Single Read)**
CC_CPO1_(1)_(2)_(3){_(4)}	**(Cyclic Read)**
CB_CPO1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Axis meaning	[0...11] (see Chapter "Data Tables")
(3) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(4) = Required measurement system (opt.)	[mm, inch]

FI command Output the command position of an axis of the device specified, related to the physical axis number.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_CPO2_(1)_(2){_(3)}	**(Single Read)**
CC_CPO2_(1)_(2){_(3)}	**(Cyclic Read)**

CB_CPO2_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1...32, according to settings of the system parameters]
(2) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(3) = Required measurement system (opt.)	[mm, inch]

Response Structure

The following table shows the general structure of the response to the FI commands "CPO1" and "CPO2". One line is output with 4 columns for the axis designation, position, unit and the position limited to “indicated decimal places”.

Line 1	Column 1	Column 2	Column 3	Column 4

Value Range/Meaning of Columns

1 = Axis name	[according to settings of axis parameters]
2 = Position	[according to settings of process parameters]
3 = Unit	[according to the settings of process parameters: mm, inch]
4 = Position	[as Column 2, but rounded up or down according to the parameter “indicated decimal places”]

Note: If the specified axis is not defined in the selected NC process then the response in all columns is [--].

Example CPO1

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00.

FI command	**00_CR_CPO1_0_2_1**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	-5.98975	[mm]	-5.990

Example CPO1

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00. The result is to be output in inches.

FI command	**00_CR_CPO1_0_2_1_inch**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	-0.23582	[inch]	-0.236

Example CPO2

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00 (e.g. physical axis number = 3) in machine coordinates.

FI command	**00_CR_CPO2_3_1**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	-5.98975	[mm]	-5.990

Reference to Literature

See chapter entitled “Literature” [25].

Trigger Control Reset: CRT

MWCX device group

Designation **CRT** **Control-Reset**

Explanation The control reset allows the selected device to be reset during an active system error. If there is no system error at the selected device then the job is ignored.

CAUTION

Carrying out a reset completely re-initializes the device.
During initialization, communication is temporarily interrupted (inherent to design).

FI command **CW_CRT** **(Single Write)**

Value to be written Trigger reset 0

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure The return value of the "DataTransfer" routine is [0] if the write procedure has been successfully completed. In the event of an error, more information in the form of a general error result line can be requested by the routine "ReadGroupItem" (refer here to chapter 8, “Error Codes” and “General Error Result Line”).

Example CRT Triggers a control reset on the selected device.

FI command	**00_CW_CRT**
Value to be written	0

Reference to Literature See chapter entitled “Literature” [26].

Device Axis Configuration Parameter: DAC

MWCX device group

Designation **DAC** **D**evice **A**xis **C**onfiguration Parameter

Explanation The configuration of the device axes that are configured in the active machine parameter record is read out. The following belong to the configuration data of the device axes: axis number, corresponding process, assigned processes, type of axis, APR number, APR axis number, main axis meaning, secondary axis meaning, main axis name, secondary axis name and corresponding axis number.

FI command Output the current parameters of all configured device axes.

BR_DAC1 **(Single Read)**

Response Structure The following table shows the general structure of the response to the “DAC1” FI command. The number of answer lines [1...32 per NC process] depends on the number of configured device axes. Each line consists of 11 columns.

Line 1...n:	**Column 1**	**...**	**Column 11**

Note: If there is no active machine parameter record in the device then the columns [1...11] are not applicable.

Value Range/Meaning of Columns

1 =	Physical axis number	[1...32]
2 =	NC process number	[0...6]
3 =	Assigned processes	[0...6,--]
4 =	Type of axis	[see Chapter 6.2 “Data Tables”]
5 =	APR number	[1...5]
6 =	APR axis number	[1...8]
7 =	Main axis meaning	[see the chapter entitled “Data Tables”]
8 =	Secondary axis meaning	[see the chapter entitled “Data Tables”]
9 =	Main axis name	[Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
10 =	Secondary axis name	[Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
11 =	Assigned axis number	[1...32, --]

Reference to Literature

See chapter entitled “Literature” [27].

Example DAC1

Reads the current parameters of all configured device axes of the active machine parameter record of device address 00.

Assumption:
The following three device axes have been defined:

- Digital linear axis (axis number 1)
- Digital linear axis (axis number 2)
- Main circular axis spindle (axis number 3).

FI command		**00_BR_DAC1**
Line	**Column**	**Answer**
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

FI command		00_BR_DAC1
Line	**Column**	**Answer**
	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 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]

Response Structure

The following table shows the general structure of the response to the "DAC2" FI command. The number of answer lines [1...32] depends on the number of configured device axes. Each line consists of 11 columns.

Line 1...n	Column 1	...	Column 11

Note: If there is no active machine parameter record in the device then the columns [1...11] are not applicable.

Value Range/Meaning of Columns

1 =	Axis number	[1...32]
2 =	NC process number	[0...6]
3 =	Assigned processes	[0...6,--]
4 =	Type of axis	[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 name	[Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
10 =	Secondary axis name	[Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
11 =	Assigned axis number	[1...32, --]

Example DAC2

Reads the current parameters of all configured digital device axes of the active machine parameter record of device address 00.

Assumption:
A digital, linear axis with axis number 1 has been defined.

FI command		00_BR_DAC2
Line	**Column**	**Answer**
1	1	1
	2	0
	3	--
	4	81
	5	1
	6	1
	7	0
	8	--
	9	X
	10	--
	11	--

NC D-Correction Download: DCA

MWCX device group

Designation **DCA** NC-**D**-**C**orrection **A**ccess

Explanation D-corrections are downloaded by means of the download file via all active processes.

FI command NC D-correction download.

BW_DCA1_(1) **(Single Write)**

(1) = Download file with path details.

Note: File and path details must be enclosed in inverted commas.

Response Structure The response to the "DCA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID [01...20]
(see Chapter "FI Commands for the MPCX Device Group", IFJ).

Line 2 = FI command
[String, in accordance with Chapter "Elements of the FI Command"]

Line 3 = FI Job Error Code
(see Chapter "Error Codes")

Example DCA1 00_BW_DCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		00_BW_DCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_DCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
3	1	0

Structure of the download file

The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
This is currently only used for error processing, i.e., if an error is detected during a process, then the *DownloadError* key is written with "YES" within this section.

<u>Example:</u>

[Common]
DownloadError = YES ; error

Section [DCorrectionPackage_Info]
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters
Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters
Package date: "**PackageDate**" max. 8 characters
Package default:"**PackageDefault**" max. 26 characters (optional)

Total: max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

<u>Example:</u>

[DCorrectionPackage_Info]
PackageNo = 1
PackageName = D correction
PackageSize = 1234
PackageTime = 13:10:10
PackageDate = 24.12.00

Section [DCorrection_A]
A: corresponds to a process number [0..6]

A section entry ([DCorrection_A]) is an optional entry, i.e., if a section for a process is absent it is not regarded as an error.

The key values correspond to the D-correction numbers [1..99] and the values are the write values of D-corrections (L1, L2, L3, R, unit optional). Missing key values are not regarded as errors.

[DCorrection_0]
001=L1 1.0 L2 2.0 L3 3.0 R 4.0
...
099=L1 10.0 L2 20.0 L3 30.0 R 40.0

[DCorrection_1]
001=L1 1.0 L2 2.0 L3 3.0 R 4.0 mm
...
050=L1 10.0 L2 20.0 L3 30.0 R 40.0 mm

[DCorrection_6]
001=L1 1.0 L2 2.0 L3 3.0 R 4.0
...
099=L1 10.0 L2 20.0 L3 30.0 R 40.0

NC D-Correction Upload: DCA

MWCX device group

Designation **DCA** NC-**D**-**C**orrection **A**ccess

Explanation D corrections are uploaded via all active processes.

FI command D corrections upload.

BR_DCA1_(1) **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.

In this command, the progress information is implemented in %. It can be interrogated via the command IFJ of the MPCX device group.

Response Structure The response to the DCA1 FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example DCA1 00_BR_DCA1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_DCA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_DCA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Section [Common]
General information is stored in the COMMON section.

Key **UploadError**
Indicates whether or not an error has occurred during uploading. This value is only set in the event of an error.

Example:

[Common]
UploadError = YES ; error

Section NC variables information [DCorrectionPackage_Info]
Key **program package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifcations is described in the following:
Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters

Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters
Package date: "**PackageDate**" max. 8 characters
Package default:"**PackageDefault**" max. 26 characters (optional)

Total: max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

[DCorrectionPackage_Info]
PackageNo = 1
PackageName = D correction
PackageSize = 1234
PackageTime = 13:10:10
PackageDate = 24.12.00

Section NC variables download [DCorrection_A]
A: corresponds to a process number [0..6]

The key values correspond to the D-correction numbers [1..99] and the values are L1, L2, L3, R, and the unit.

[DCorrection_0]
001= L1 1.0 L2 2.0 L3 3.0 R 4.0 mm
...
099= L1 1.0 L2 2.0 L3 3.0 R 4.0 mm

[DCorrection_1]
001= L1 1.0 L2 2.0 L3 3.0 R 4.0 mm
...
099= L1 1.0 L2 2.0 L3 3.0 R 4.0 mm

[DCorrection_6]
001= L1 1.0 L2 2.0 L3 3.0 R 4.0 mm
...
099= L1 1.0 L2 2.0 L3 3.0 R 4.0 mm

Reading D-Correction Data: DCD

MWCX device group

Designation **DCD** **D-C**orrection **D**ata

Explanation The values of a D-correction register of the selected NC process are read out.

The D-corrections are additive to the tool geometry data of the register that effects the tool management, i.e. they are additive to the existing geometry registers L1, L2, L3 and R.

There are 99 D-correction numbers available for each of the seven NC processes. Each D-correction number therefore contains the registers L1, L2, L3 and R. Value assignment of the D-correction register is via the Rexroth Indramat GUI or via the function interface.

FI command Reading of a D-correction register value of an NC process of the selected device.

CR_DCD1_(1)_(2)_(3){_(4)} **(Single Read)**

CC_DCD1_(1)_(2)_(3){_(4)} **(Cyclic Read)**

CB_DCD1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC 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]
(4) = Required measurement system (opt.)	[mm, inch]

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches.

Response Structure

The response consists of one line with three columns for the identifier (length correction L1 to L3 and radius correction R), the value of the requested D-correction register, and the unit in accordance with the settings of the process parameters.

Line	Column 1	Column 2	Column 3

Value Range/Meaning of Columns

1 =	Identifier	[L1, L2, L3, R]
2 =	Value of D-correction	[formatting according to setting of the process parameters]
3 =	Unit	[mm, inch; according to settings of the process parameters]

Note: If the requested D-correction number or the D-correction register is not assigned a value then the value 0 is output as response, formatted according to the settings in the process parameters.

Example DCD1

Read the value of the D-correction register 4 at device address 00 in NC process 0 of the D-correction number 1 (radius correction R).

FI command	**00_CR_DCD1_0_1_4**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	R	0.0860	[mm]

Reference to Literature

See chapter entitled “Literature” [6].

Device Configuration Parameter: DCP

MWCX device group

Designation

DCP **D**evice **C**onfiguration **P**arameter

Explanation

The device configuration parameters that are entered in the active machine parameter record as well as in the "IND_DEV.INI" file are output. The configuration parameters of the device include the device address, the device name, device type, mechanism number, mechanism name, and the process types.

FI command

Output the configuration parameters of all defined devices.

BR_DCP1 **(Single Read)**

Note: The “DCP1” FI command refers to all defined devices. Therefore, any valid device address can be indicated in the command line (see example DCP1).

Response Structure

The following table shows the general structure of the response to the "DCP1" FI command. The response consists of a maximum of n=512 lines (n=16 devices x 32 mechanisms = 512), each with 7 columns.

Line 1...n:	Column 1	...	Column 7

Note: If no active machine parameter record exists in the device, then the columns [1...7] for the respective device are not applicable.

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Device Type	[MTC200-P-G2, MTC200-R-G2, MTVNC]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process type	[1= internal, 2 = external process]
7 =	Process type	[1 = NC Process, 2 = PLC Process]

Example DCP1

Read the device configuration parameters of all defined devices.

<u>Assumption:</u>
Three devices have been defined

- Device address 00 (MTC200-P-G2)
- Device address 01 (MTC200-R-G2) and
- Device address 02 (MTC200-P-G2)

FI command		00_BR_DCP1
Line	**Column**	**Answer**
1	1	00
	2	Rotary transfer machine
	3	MTC200-P-G2
	4	1
	5	Master
	6	1
	7	2
2	1	01
	2	0
	3	MTC200-R-G2
	4	0
	5	Milling machine 01
	6	1
	7	1
3	1	02
	2	0
	3	MTC200-P-G2
	4	1
	5	Milling machine 02
	6	1
	7	1

FI command

Output the configuration parameters of the selected device.

BR_DCP2 **(Single Read)**

Response Structure

The following table shows the general structure of the response to the “DCP2” FI command. The response consists of a line with 7 columns.

Line 1	Column 1	...	Column 7

Note: If no active machine parameter record exists in the device, then the columns [1...7] for the respective device are not applicable.

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Device Type	[MTC200-P-G2, MTC200-R-G2, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process type	[1= internal, 2 = external process]
7 =	Process type	[1 = NC Process, 2 = PLC Process]

Example DCP2

Read the device configuration parameter of the selected device (device address 01).

Assumption:
Three devices have been defined

- Device address 00 (MTC200-R)
- Device address 01 (MTC200-P)
- Device address 02 (MTC200-P)

FI command		**01_BR_DCP2**
Line	**Column**	**Answer**
1	1	01
	2	0
	3	MTC200-P
	4	0
	5	Milling machine 01
	6	1
	7	1

Reference to Literature

See chapter entitled “Literature” [28].

C Correction Register: DCR

MWCX device group

Designation **DCR** **D**-**C**orrection **R**ecord

Explanation

The values of a D-correction record of the selected NC process are read out or written.

The D-corrections are additive to the tool geometry data of the register that effects the tool management, i.e. they are additive to the exisitng geometry registers L1, L2, L3 and R.

There are 99 D-correction numbers available for each of the seven NC processes. Each D-correction number therefore contains the registers L1, L2, L3 and R. Value assignment of the D-correction register is via the Rexroth Indramat GUI or via the function interface.

FI command

Reading of a D-correction record of an NC 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) = NC process number	[0...6]
(2) = D-correction number	[1...99]
(3) = Required measurement system (opt.)	[mm, inch]

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches.

Note: If the value of a single D-correction register is to be read then the command "CR_DCD1" should be used.

Response Structure

The response consists of four lines, each with three columns for the identifier (length correction L1 to L3 and radius correction R), the value of the requested D-correction register, and the unit in accordance with the settings of the process parameters.

Line 1..0.4	Column 1	...	Column 3

Value Range/Meaning of Columns

1 =	Identifier	[L1, L2, L3, R]
2 =	Value of D-correction	[Formatting of values according to the settings of process parameters]
3 =	Unit	[mm, inch; according to parameter settings]

Note: If the requested D-correction number or the D-correction register is not assigned a value then the value 0 is output as response, formatted according to the settings in the process parameters.

Example DCR1 without optional Parameters

Read the value of all D-correction registers at device address 00 of NC process 0 of D-correction number 1.

FI command	**00_CR_DCR1_0_1**		
Answer			
Line	**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]

FI command

Write all D-correction register values of an NC process of the selected device.

CW_DCR_(1)_(2)	**(Single Write)**
(1) = NC 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>] opt. unit]

If there is no optional information for the unit {mm, inch}, then the values refer to the base programming unit of the process. If the unit entered differs from the basic programming unit then the values entered are converted into the values of the base programming unit.

Note: In the conversion from mm → inch, rounding errors are unavoidable, as precision is lost!

The single values are separated by a space, whereby the formatting should be carried out according to the settings of the process parameters. (see example DCR1: write D correction register).

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Value has been written

Example DCR
Write D-Correction Register

Write all D-correction registers at device address 00 of NC process 0 of D-correction number 1 with the following 5 values:

1. Value L1: 1.2586
2. Value L2: 3.5892
3. Value L3: 0.0000 and
4. Value R: 0.0860
5. Unit of the values: mm (optional)

Note: The values to be written are passed in the “Data Transfer” routine to the "acValue" parameter and must be separated from each other by a space " ".

FI command		**00_CW_DCR_0_1** **Value to write:** **L1 1.2586 L2 3.5892 L3 0 R 0.086 mm**
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature

See chapter entitled “Literature” [6].

Setting the Communication Timeout Time DCTMWCX device group

Designation

DCT **D**evice **C**ommunication **T**imeout

Explanation

By means of this command, the timeout time for the selected device is set dynamically (timeout time in ms).

FI command

BW_DCT1_(1) **(Single Write)**

(1) = requested timeout time in ms

Response Structure

The response to the "DCT1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCT1

For the device 00, the timeout time is set 1500 ms.

FI command		**00_BW_DCT1_1500**
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

With this command, the timeout time for the selected device can be reset to default value.

BW_DCT2 **(Single Write)**

Response Structure

The response to the "DCT2" FI command consists of one line with one column.

Line 1	**Column 1**

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCP2

For the device 00, the timeout time is reset to the default value.

FI command		**00_BW_DCT2**
Line	**Column**	**Answer**
1	1	(P_ACK)

Long Identification of NC/PLC Data Records: DIS

MWCX device group

Designation

DIS **D**ata **I**dentification **S**tring

Explanation

Reads the long identification (directory entries) of NC/PLC data records. Included in the directory entries are the number of the entry in the directory, the name, length and date and time of creation and/or details of the last time the respective data record was changed. The long identifications of the following NC/PLC data records are output:

NC parameter record (FI command: DIS1)

PLC program (FI command: DIS2)

NC package (FI command: DIS3)

Tool list (FI command): DIS4)

Machine data (FI command): DIS5) and

NC program (FI command: DIS6)

FI command

Output the directory entries of the valid NC parameter record in the selected device.

BR_DIS1 **(Single Read)**

BC_DIS1 **(Cyclic Read)**

BB_DIS1 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the “DIS1” FI command. The response consists of a line with five columns.

Line 1	**Column 1**	**...**	**Column 5**

Value Range/Meaning of Columns

1 = Number in NC parameter directory [01...99]

2 = Name of the NC parameter record [max. 32 ASCII characters]

3 =	Length of the NC parameter record	[byte]
4 =	Date of creation/last change to NC parameter record	[DD.MM.YY]
5 =	Time of creation/last change to NC parameter record	[HH:MM:SS]

Note: If there is no valid NC parameter record in the selected device then all columns contain [--] . This command can also be used when the selected device is in OFFLINE mode (DeviceStatus=OFF).

Example DIS1

Read the directory entries of the NC parameter record at device address 00.

Assumption:

There is a valid NC parameter record in the selected device.

FI command		00_BR_DIS1
Line	**Column**	**Answer**
1	1	01
	2	KEY1
	3	3579
	4	16.05.99
	5	10:41:08

Reference to Literature

See chapter entitled “Literature” [29].

FI command

Output the directory entries of the valid PLC program in the selected device.

BR_DIS2 **(Single Read)**

BC_DIS2 **(Cyclic Read)**

BB_DIS2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the “DIS2” FI command. The response consists of a line with six columns.

Line 1	Column 1	...	Column 6

Value Range/Meaning of Columns

1 =	Number in PLC directory	[01...99]
2 =	Name of the PLC program	[max. 8 ASCII characters]
3 =	Length of the PLC program	[byte]
4 =	Date of creation/last change to PLC program	[DD.MM.YY]
5 =	Time of creation/last change to the PLC program	[HH:MM:SS]
6 =	Date of creation/last change to PLC program	[DD.MM.YYYY]

Note: If there is no valid PLC program in the selected device then all columns contain [--] .

Example DIS2

Read the directory entries of the PLC program at address 00.
Assumption:
There is a valid PLC program in the selected device.

FI command		00_BR_DIS2
Line	Column	Answer
1	1	01
	2	KEY1
	3	20018
	4	10.05.99
	5	12:42:00
	6	10.05.1999

Reference to Literature

See chapter entitled "Literature" [30].

FI command

Output the directory entries of the valid NC package of the selected NC memory.

BR_DIS3_(1)	**(Single Read)**
BC_DIS3_(1)	**(Cyclic Read)**
BB_DIS3_(1)	**(Break Cyclic Read)**
(1) = NC memory	[1 = NC memory A; 2 = NC memory B]

Response Structure

The following table shows the general structure of the response to the "DIS3" FI command. The response consists of a line with five columns.

Line 1	Column 1	...	Column 5

Value Range/Meaning of Columns

1 =	Number in NC package directory	[01...99]
2 =	Name of the NC package	[max. 32 ASCII characters]
3 =	Length of the NC package	[byte]
4 =	Date of creation/last change to NC package	[DD.MM.YY]
5 =	Time of creation/last change to NC package	[HH:MM:SS]

Note: If there is no valid NC package in the selected NC memory then all columns contain [--] .

Example DIS3

Read the directory entries of the NC package in NC memory A at device address 00.
Assumption:
There is a valid NC package in memory A of the selected device.

FI command		00_BR_DIS3_1
Line	Column	Answer
1	1	01
	2	KEY1
	3	3579
	4	16.05.99
	5	10:41:08

Reference to Literature

See chapter entitled "Literature" [31].

FI command

Output the directory entries of the valid tool list of the selected NC process.

BR_DIS4_(1) **(Single Read)**

BC_DIS4_(1) **(Cyclic Read)**

BR_DIS4_(1) **(Break Cyclic Read)**

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

Response Structure

The following table shows the general structure of the response to the "DIS4" FI command. The response consists of a line with five columns.

Line 1	Column 1	...	Column 5

Value Range/Meaning of Columns

1 =	Number in the tool list index	[01...99]
2 =	Name of the tool list	[max. 32 ASCII characters]
3 =	Length of the tool list	[byte]
4 =	Date of creation/last change to the tool list	[DD.MM.YY]
5 =	Time of creation/last change to the tool list	[HH:MM:SS]

Note: If there is no valid tool list in the selected NC process then all columns contain [--] .

Example DIS4

Read the directory entries of the tool list of NC process 0 at device address 00.
Assumption:
There is a valid tool list in NC process 0 of the selected device.

FI command		00_BR_DIS4_0
Line	Column	Answer
1	1	01
	2	KEY1
	3	2048
	4	17.09.99
	5	10:45:08

Reference to Literature

See chapter entitled "Literature" [32].

FI command

Output the directory entries of the valid machine data record in the selected device.

BR_DIS5 **(Single Read)**

BC_DIS5 **(Cyclic Read)**

BB_DIS5 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the "DIS5" FI command. The response consists of a line with five columns.

Line 1	Column 1	...	Column 5

Value Range/Meaning of Columns

1 =	Number in machine data directory	[01...99]
2 =	Name of the data record	[max. 32 ASCII characters]
3 =	Length of data record	[byte]

4 =	Date of creation/last change to the data record	[DD.MM.YY]
5 =	Time of creation/last change to the data record	[HH:MM:SS]

Note: If there is no valid machine data in the selected device then all columns contain [--] .

Example DIS5

Read the directory entries of the machine data record in device address 00.

Assumption:

There is valid machine data in the selected device

FI command		**00_BR_DIS5**
Line	**Column**	**Answer**
1	1	01
	2	KEY1
	3	3180
	4	18.12.98
	5	21:20:02

Reference to Literature

See chapter entitled “Literature” [32].

FI command

Output the directory entries of the valid NC program.

BR_DIS6_(1)_(2)_(3)	**(Single Read)**
BC_DIS6_(1)_(2)_(3)	**(Cyclic Read)**
BB_DIS6_(1)_(2)_(3)	**(Break Cyclic Read)**
(1) = NC memory	[1 = NC memory A, 2 = NC memory B]
(2) = NC process number	[0...6]
(3) = NC program number	[1...99]

Response Structure

The following table shows the general structure of the response to the “DIS6” FI command. The response consists of a line with six columns.

Line 1	**Column 1**	**...**	**Column 6**

Value Range/Meaning of Columns

1 =	Package number	[01...99]
2 =	Number of the NC program	[01...99]
3 =	Name of the NC program	[max. 32 ASCII characters]
4 =	Length of the NC program	[byte]
5 =	Date of creation/last change to NC program	[DD.MM.YY]
6 =	Time of creation/last change to NC program	[HH:MM:SS]

Note: If there is no valid NC package in the selected NC process, then all columns contain [--] .

Example DIS6

Read the directory entries of the third NC program (NC package number 2, NC memory A, NC process 0) at device address 00.
Assumption:
There is valid data in the selected device.

FI command		**00_BR_DIS6_1_0_3**
Line	**Column**	**Answer**
1	1	03
	2	Audi A4
	3	3579
	4	16.05.99
	5	10:41:08

Reference to Literature

See chapter entitled “Literature” [17].

Delete NC Program: DPN

MWCX device group

Designation

DPN **D**elete **P**rogram **NC**

Explanation

An NC program located in an NC package directory is deleted.

FI command

BW_DPN_(1)_(2)_(3)_(4)	**(Single Write)**
(1) = NC package directory number	[1...99]
(2) = NC process number	[0 6]
(3) = NC program number	[1...99]
(4) = with check / without check	[1 / 0]

Response Structure

One line with one column is output to acknowledge the FI command issued. The following meanings then apply, depending on parameter 4 (check):

With check (1)	
(BOF_C_NCPROG_CREATED)	NC program not deleted.

Without check (0)	
(BOF_FCT_OK) = **BOF_F**un**CT**ion**_OK**	NC program has been deleted.

Example DPN

The NC program numbered 1 in NC package directory 3 of process 2 is to be deleted.

FI command		**00_BW_DPN_1_2_3_0**
Line	**Column**	**Answer**
1	1	(BOF_FCT_OK)

Reference to Literature

See chapter entitled “Literature” [31].

Rexroth Indramat

Delete NC Program Package: DPP

MWCX device group

Designation **DPP** **D**elete **P**rogram **P**ackage

Explanation An NC program package is deleted in the NC package directory of the selected MWCX device group.

FI command **BW_DPP_(1)** **(Single Write)**

(1) = NC program package [1...99]

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(BOF_FCT_OK) = **BOF_FunCTion_OK**	Program package has been deleted.

Example DPP The NC program package numbered 1 in the NC package directory is to be deleted.

FI command		**00_BW_DPN_1_2_3_0**
Line	**Column**	**Answer**
1	1	(BOF_FCT_OK)

Reference to Literature See chapter entitled “Literature” [17].

Reading the Device Status Information: DSI

MWCX device group

Designation **DSI** **D**evice **S**tatus **I**nformation

Explanation This allows the most important device status information to be read out. The following information is returned:

Type of information	**Status**	**Statement**
System error information		
Information on mechanism error		
Machine key information	valid	Yes/No
Machine key information		
Machine status information		
Sercans information		
Parameter download	running	Yes/No
PLC download	running	Yes/No
Firmware download	running	Yes/No
Offline/Online information		
Device simulation	switched on	Yes/No
Device status information		ON, OFF

FI command Read out device status information for ALL defined devices.

BR_DSI1 **(Single Read)**

BC_DSI1 **(Cyclic Read)**

BB_DSI1 **(Break Cyclic Read)**

Note: The "DSI1" FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see example DSI1). The FI device polling mechanism **MUST** be switched on (see system configurator)!

Response Structure

The following table shows the general structure of the response to the "DSI1" FI command.

Line 1...n	Column 1	...	Column 13

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Information on mechanism error	[0 = there is no mechanism error 1 = there is a mechanism error]
4 =	Machine key information	[4 byte in HEX coding]
5 =	Is machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = DeviceStatus=OFF 1 = DeviceStatus=ON]

Example DSI1

Read the current device status information.
<u>Assumption:</u>
The following devices addresses are defined:

- Device address 01 (MWCX device)
- Device address 03 (MWSX device)

FI command		01_BR_DSI1
Line	Column	Answer
1	1	01
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1
2	1	03
	2	1
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

FI command

Read out device status information for a selected device.

BR_DSI2 **(Single Read)**

BC_DSI2 **(Cyclic Read)**

BB_DSI2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the "DSI2" FI command.

Line 1...n	Column 1	...	Column 13

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Information on mechanism error	[0 = there is no mechanism error 1 = there is a mechanism error]
4 =	Machine key information	[4 byte in HEX coding]
5 =	Is machine key information valid?	[0 = not valid, 1=valid]

6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = DeviceStatus=OFF 1 = DeviceStatus=ON]

Example DSI2 Read the current device status information for the selected device.

FI command		**01_BR_DSI2**
Line	**Column**	**Answer**
1	1	00
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

Tool Management Configuration Data: DTC

MWCX device group

Designation **DTC** **D**evice **T**ool Management **C**onfiguration

Explanation Supplies the most important system parameter data for tool management.

FI command Read tool management data.

BR_DTC1 **(Single Read)**

BC_DTC1 **(Cyclic Read)**

Response Structure One line with 10 columns is output for the returned values.

Line 1	**Column 1**	**...**	**Column 10**

Value Range/Meaning of the Columns

1 =	Tool Management	[YES, NO]
2 =	Setup list	[[STATION], [PROGRAM]]
3 =	Max. number of tool edges	[1...9]
4 =	Wear register	[YES, NO]

5 =	Offset register	[YES, NO]
6 =	Comment	[YES, NO]
7 =	Wear factors	[YES, NO]
8 =	Tool life	[YES, NO]
9 =	Geometry limit values	[YES, NO]
10 =	Tool technology	[[TURN./MILL.], [GRINDING]]

Note: If there is no tool management (Column 1: NO), then all partial results from Column 2 are marked as [--].

Example DTC1

Returns the system parameter data from the tool management

FI command		**00_BR_DTC1**
Line	**Column**	**Answer**
1	1	YES
	2	[STATION]
	3	4
	4	YES
	5	YES
	6	NO
	7	NO
	8	YES
	9	YES
	10	[TURN./MILL.]

FI command

Data is read from tool management, as e.g. basic user data and tool edge user data.

BR_DTC2 **(Single Read)**

BC_DTC2 **(Cyclic Read)**

Response Structure

One line with 48 columns is output for the returned values.

Line 1	Column 1	...	Column 48

Value Range/Meaning of the Columns

1 =	Tool Management	[YES, NO]
2 =	Setup list	[[STATION], [PROGRAM]]
3 =	Max. number of tool edges	[1...9]
4 =	Wear register	[YES, NO]
5 =	Offset register	[YES, NO]
6 =	Comment	[YES, NO]
7 =	Wear factors	[YES, NO]
8 =	Tool life	[YES, NO]
9 =	Geometry limit values	[YES, NO]
10 =	Tool technology	[[TURN./MILL.], [GRINDING]]
11 =	Tool user date 1	[YES, NO]
12 =	Tool user date 1	[Tool user date,--]
13 =	Tool user date 2	[YES, NO]
14 =	Tool user date 2	[Tool user date,--]
15 =	Tool user date 3	[YES, NO]

16 =	Tool user date 3	[Tool user date,--]
17 =	Tool user date 4	[YES, NO]
18 =	Tool user date 4	[Tool user date,--]
19 =	Tool user date 5	[YES, NO]
20 =	Tool user date 5	[Tool user date,--]
21 =	Tool user date 6	[YES, NO]
22 =	Tool user date 6	[Tool user date,--]
23 =	Tool user date 7	[YES, NO]
24 =	Tool user date 7	[Tool user date,--]
25 =	Tool user date 8	[YES, NO]
26 =	Tool user date 8	[Tool user date,--]
27 =	Tool user date 9	[YES, NO]
28 =	Tool user date 9	[Tool user date,--]
29 =	Cutter user date 1	[YES, NO]
30 =	Cutter user date 1	[Cutter user date,--]
31 =	Cutter user date 2	[YES, NO]
32 =	Cutter user date 2	[Cutter user date,--]
33 =	Cutter user date 3	[YES, NO]
34 =	Cutter user date 3	[Cutter user date,--]
35 =	Cutter user date 4	[YES, NO]
36 =	Cutter user date 4	[Cutter user date,--]
37 =	Cutter user date 5	[YES, NO]
38 =	Cutter user date 5	[Cutter user date,--]
39 =	Cutter user date 6	[YES, NO]
40 =	Cutter user date 6	[Cutter user date,--]
41 =	Cutter user date 7	[YES, NO]
42 =	Cutter user date 7	[Cutter user date,--]
43 =	Cutter user date 8	[YES, NO]
44 =	Cutter user date 8	[Cutter user date,--]
45 =	Cutter user date 9	[YES, NO]
46 =	Cutter user date 9	[Cutter user date,--]
47 =	Cutter user date 10	[YES, NO]
48 =	Cutter user date 10	[Cutter user date,--]

Note: If there is no tool management (Column 1: NO), then all partial results from Column 2 are marked as [--].

Example DTC2 Supply the system parameter data from the tool management.

FI command		**00_BR_DTC2**
Line	**Column**	**Answer**
1	1	YES
	2	[STATION]
	3	4
	4	YES
	5	YES
	6	NO
	7	NO
	8	YES
	9	YES
	10	[TURN./MILL.]
	11	NO
	12	--
	13	NO
	14	--
	15	NO
	16	--
	17	NO
	18	--
	19	NO
	20	--
	21	NO
	22	--
	23	NO
	24	--
	25	NO
	26	--
	27	NO
	28	--
	29	NO
	30	--
	31	NO
	32	--
	33	NO
	34	--
	35	NO
	36	--
	37	NO
	38	--
	39	NO

FI command		00_BR_DTC2
Line	**Column**	**Answer**
	40	--
	41	NO
	42	--
	43	NO
	44	--
	45	NO
	46	--
	47	NO
	48	--

Reference to Literature See chapter entitled “Literature” [8].

Distance to Go of Axis Movement: DTG

MWCX device group

Designation **DTG** **D**istance **T**o **G**o

Explanation The distance to go of the movement of a selected axis is output. The FI command "DTG1" returns the distance to go of an axis, related to the code of the axis meaning. The FI command "DTG2", on the other hand, returns the distance to go of an axis, related to the physical axis number.

FI command Output the distance to go of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_DTG1_(1)_(2)_(3){_(4)}	**(Single Read)**
CC_DTG1_(1)_(2)_(3){_(4)}	**(Cyclic Read)**
CB_DTG1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter “Data Tables”)
(3) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(4) = Required measurement system (opt.)	[mm, inch]

FI command Output the distance to go of the movement of the selected axis of the device specified related to the physical axis number.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_DTG2_(1)_(2){_(3)}	**(Single Read)**
CC_DTG2_(1)_(2){_(3)}	**(Cyclic Read)**
CB_DTG2_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1...32, according to settings of the system parameters]
(2) = System of coordinates	[1 = machine coordinates

2 = program coordinates]

(3) = Required measurement system (opt.) [mm, inch]

Response Structure

The following table shows the general structure of the response to the FI commands "DTG1" and "DTG2". One line is output with 4 columns for the axis designation, distance to go, unit and the distance to go limited to “indicated decimal places”.

Line 1	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = Axis name [according to settings of axis parameters]

2 = Distance to go [according to settings of process parameters]

3 = Unit [mm, inch]

4 = Distance to go [as Column 2, but rounded up or down according to the parameter “indicated decimal places”]

Note: If the specified axis or a spindle is not defined in the selected NC process then the answer in all columns is [--].

Example DTG1

Read the distance to go of the movement of the Z axis in machine coordinates in NC process 0 of device address 00.

FI command	00_CR_DTG1_0_2_1			
Answer				
Line 1	Column 1	Column 2	Column 3	Column 4
1	Z1	-1.2345	[mm]	-1.235

Example DTG1

Read the distance to go of the movement of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in inches.

FI command	00_CR_DTG1_0_2_1_inch			
Answer				
Line 1	Column 1	Column 2	Column 3	Column 4
1	Z1	-0.0486	[inch]	-0.049

Example DTG2

Read the distance to go of the movement of the Z axis (physical axis number = 3) in machine coordinates at the device address 00.

FI command	00_CR_DTG2_3_1			
Answer				
Line 1	Column 1	Column 2	Column 3	Column 4
1	Z1	-1.2345	[mm]	-1.235

Reference to Literature

See chapter entitled “Literature” [16].

Device Type and Accompanying Components: DTY

MWCX device group

Designation **DTY** **D**evice **TY**pe

Explanation The device type and the accompanying components of the selected device address are output.

FI command **BR_DTY1** **(Single Read)**

Response Structure The following table shows the general structure of the response to the “DTY1” FI command. A line with three columns is output for the device type, as well as the names of the first device component and the name of the second device component.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Device Type (see chapter entitled “Elements of the FI Command”, and “Identifier”)

2 = Component type1 IND_DEV.INI entry: Componenttype1=

3 = Component type 2 IND_DEV.INI entry: Componenttype2=

Example DTY1 Output the device type and the accompanying components of device address 00.

FI command	00_BR_DTY1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	MTC200-P-G2	MTS-P	MTC-P

Diagnosis Window Data: DWD

MWCX device group

Designation **DWD** **D**iagnosis **W**indow **D**ata

Explanation Diagnostic messages are output. The data is edited in such a way that it can be output directly in the diagnosis overview, i.e., where applicable, different types of diagnosis, such as a ProVi message and a process message, are returned simultaneously.

FI command Output all diagnostic messages.

BR_DWD1_(1){_(2)} **(Single Read)**

BC_DWD1_(1){_(2)} **(Cyclic Read)**

(1) = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start requirements, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Output first diagnostic messages.

BR_DWD2_(1){_(2)} **(Single Read)**

BC_DWD2_(1){_(2)} **(Cyclic Read)**

(1) = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start requirements,

	11 = warnings, 12 = setup diagnosis]
(2) = Module number	[1...99] ! only for window type 1 -4 !

Response Structure

The following table shows the general structure of the "DWD1" and "DWD2" FI commands. The number of lines depends on the number of messages pending. Different columns are valid according to the type of diagnosis. If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 12

Meaning of the Columns

1 =	Message text	[ASCII characters]
2 =	Time stamp day	[mm.dd.yyyy]
3 =	Time stamp hour	[hh:mm:ss]
4 =	Reference text available	[YES, NO]
5 =	Type of diagnosis	[1 = ProVi, 2 = SFC, 3 = MTC-NC, 4 = MTA-NC]
6 =	Message number	[ASCII characters]
7 =	Message ID	[ASCII characters] (DWORD, decimal) (ProVi)
8 =	Mechanism number	[0..31] (MTC-NC) [0] (MTA-NC)
9 =	2 byte additional information	[ASCII characters] (MTC NC)
10 =	Message group	[1...9999] (MTA-NC)
11 =	SFC entity name	[ASCII characters]
12 =	NC note	[ASCII characters] (MTC NC)
13 =	Analysis of criteria available	[YES, NO] (ProVi, SFC)
14 =	Message HTML file	[ASCII characters] (ProVi, MTC-NC)

Example DWD1

All diagnostic messages from Module 3 in Control unit 0.
There are two messages present:

FI command		00_BR_DWD1_4_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	01.27.2000
	3	14:56:32
	4	YES
	5	1
	6	34
	7	43923028
	8	
	9	
	10	
	11	
	12	
	13	YES
	14	

FI command		00_BR_DWD1_4_3
Line	**Column**	**Answer**
2	1	Station waiting until tool-change command has ended.
	2	01.27.2000
	3	15:03:10
	4	YES
	5	3
	6	79
	7	
	8	1
	9	0
	10	
	11	
	12	
	13	NO
	14	

Example DWD2

The first diagnostic message from Module 3 in Control unit 0.
There are two messages present:

FI command		00_BR_DWD2_4_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	01.27.2000
	3	14:56:32
	4	YES
	5	1
	6	34
	7	43923028
	8	
	9	
	10	
	11	
	12	
	13	YES
	14	

Reference to Literature

See chapter entitled “Literature” [13].

Existing errors: EDE

MWCX device group

Designation **EDE** **E**xisting **D**iagnosis **E**rror

Explanation Whether or not errors exist in a control unit or in a module is queried. These can be step chain errors, NC errors or ProVi errors.

FI command Query whether there are errors in this control unit.

BR_EDE1 **(Single Read)**

BC_EDE1 **(Cyclic Read)**

Response Structure The following table shows the general structure of the "EDE1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Error exists [YES, NO]

Example EDE1 Do errors exist in control unit 0?

FI command		00_BR_EDE1
Line	**Column**	**Answer**
1	1	YES

FI command Query whether or not errors exist in a specific module.

BR_EDE2_(1) **(Single Read)**

BC_EDE2_(1) **(Cyclic Read)**

(1) = Module number [1...99]

Response Structure The following table shows the general structure of the "EDE2" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Error exists [YES, NO]

Example EDE2 Do errors exist in Module 1 on Control unit 0?

FI command		00_BR_EDE2_2
Line	**Column**	**Answer**
1	1	NO

Reference to Literature See chapter entitled “Literature” [13].

Existing Diagnosis Window: EDW

MWCX device group

Designation **EDW** **E**xisting **D**iagnosis **W**indow

Explanation Which types of diagnosis window exist is queried.

FI command Output all types of diagnosis window.

BR_EDW1 **(Single Read)**

Response Structure The following table shows the general structure of the "EDW1" FI command. The number of lines depends on the number of types of window existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type of diagnosis window [1 = NC error, 2 = sequence errors,
3 = general errors, 4 = messages,
10 = start requirements,
11 = warnings, 12 = setup diagnosis]

2 = Module number [ASCII characters]
0 = Diagnosis window type does not belong to any module

Example EDW1 All types of diagnosis window in control unit 0.

There are three diagnosis windows.

FI command		00_BR_EDW1
Line	Column	Answer
1	1	10
	2	0
2	1	1
	2	3
3	1	2
	2	3

FI command Output all types of diagnosis window for one module.

BR_EDW2_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure The following table shows the general structure of the "EDW2" FI command. The number of lines depends on the number of types of window existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type of diagnosis window [1 = NC error, 2 = sequence errors,
3 = general errors, 4 = messages]

2 = Module number [ASCII characters]
0 = Diagnosis window type does not belong to any module

Example EDW2

All types of diagnosis window in Module 3, Control unit 0.

There are two diagnosis windows.

FI command		00_BR_EDW2_3
Line	**Column**	**Answer**
1	1	1
	2	3
2	1	2
	2	3

FI command

Query a specific type of diagnosis window.

BR_EDW3_(1){_(2)} **(Single Read)**

(1) = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start requirements, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Response Structure

The following table shows the general structure of the "EDW3" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Type of diagnosis window exists [YES, NO]

Example EDW3

Query whether or not a NC error window exists in module 3, control unit 0.

FI command		00_BR_EDW3_1_3
Line	**Column**	**Answer**
1	1	YES

Existing NC Diagnoses: END

MWCX device group

Designation

END **E**xisting **N**C **D**iagnosis

Explanation

Which NC diagnostic types exist is queried. Depending on the FI command, specific types are queried or else the diagnostic types for one module are output together.

FI command

Query which NC diagnostic types are available in a module.

BR_END1_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure

The following table shows the general structure of the "END1" FI command.

Line 1	Column 1-2

Meaning of the Columns

1 = Messages exist [YES, NO]

2 = Errors exist [YES, NO]

Example END1 Query the NC diagnostic types in Module 2 on Control unit 0.

FI command		00_BR_END1_2
Line	**Column**	**Answer**
1	1	NO
	2	YES

FI command Query a specific NC diagnostic type.

BR_END2_(1)_(2) **(Single Read)**

(1) = Message type [1 = error, 2 = messages]

(2) = Module number [1...99]

Response Structure The following table shows the general structure of the "END2" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Diagnosis type exists [YES, NO]

Example END2 Are there any messages in module 4 in control unit 0?

FI command		00_BR_END2_2_4
Line	**Column**	**Answer**
1	1	YES

Existing PLC Diagnoses: EPD

MWCX device group

Designation **EPD** **E**xisting **P**LC **D**iagnosis

Explanation Which PLC diagnostic types exist is queried. Depending on the FI command, specific types are queried or else the diagnostic types for a device or a module are output together.

FI command Query which PLC diagnostic types are available on a control unit.

BR_EPD1 **(Single Read)**

Response Structure The following table shows the general structure of the "EPD1" FI command.

Line 1	Column 1-3

Meaning of the Columns

1 = Start requirement exists [YES, NO]

2 = Warning exists [YES, NO]

3 = Setup diagnosis exists [YES, NO]

Example EPD1 Query PLC diagnostic types in control unit 0.

FI command		00_BR_EPD1
Line	**Column**	**Answer**
1	1	YES
	2	NO
	3	YES

FI command Query which PLC diagnostic types are available in a module.

BR_EPD2_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure

The following table shows the general structure of the "EPD2" FI command.

Line 1	Column 1-3

Meaning of the Columns

1 = Messages exist [YES, NO]

2 = Errors exist [YES, NO]

3 = Step chains exist [YES, NO]

Example EPD2

Query the PLC diagnostic types in Module 2 on Control unit 0.

FI command		**00_BR_EPD2_2**
Line	**Column**	**Answer**
1	1	NO
	2	YES
	3	YES

FI command

Query a specific PLC diagnostic type.

BR_EPD3_(1){_(2)} **(Single Read)**

(1) = Message type [1 = error, 2 = messages, 3 = SFC, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -3!

Response Structure

The following table shows the general structure of the "EPD3" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Diagnosis type exists [YES, NO]

Example EPD3

Are there any messages in module 4 in control unit 0?

FI command		**00_BR_EPD3_2_4**
Line	**Column**	**Answer**
1	1	YES

End Point of an Axis Movement: EPO

MWCX device group

Designation

EPO **E**nd **PO**sition

Explanation

The end point of the movement of a selected axis is output. The FI command "EPO1" returns the end point of the movement, related to the code of the axis meaning. The FI command "EPO2", on the other hand, returns the end point of the movement of an axis related to the physical axis number.

FI command

Output the end point of the selected device related to the code of the axis meaning.

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_EPO1_(1)_(2)_(3){_(4)}	**(Single Read)**
CC_EPO1_(1)_(2)_(3){_(4)}	**(Cyclic Read)**
CB_EPO1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see chapter “Data Tables”)
(3) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(4) = Required measurement system (opt.)	[mm, inch]

FI command

Output the end point of the selected axis of the device specified, related to the physical axis number.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_EPO2_(1)_(2){_(3)}	**(Single Read)**
CC_EPO2_(1)_(2){_(3)}	**(Cyclic Read)**
CB_EPO2_(1)_(2){_(3)}	**(Break Cyclic Read)**

(1) = Physical axis number	[1...32, according to settings of the system parameters]
(2) = System of coordinates	[1 = machine coordinates 2 = program coordinates]
(3) = Required measurement system (opt.)	[mm, inch]

Response Structure

The following table shows the general structure of the response to the FI commands "EPO1" and "EPO2". One line is output with 4 columns for the axis designation, end point of the movement, unit and the position limited to “indicated decimal places”.

Line 1	Column 1	Column 2	Column 3	Column 4

Value Range/Meaning of Columns

1 = Axis name	[according to settings of axis parameters]
2 = end point	[according to settings of process parameters]
3 = Unit	[mm, inch]
4 = end point	[as Column 2, but rounded up or down according to the parameter “indicated decimal places”]

Note: If the specified axis is not defined in the selected NC process then the response in all columns is [--].

Example EPO1

Read the end point of the movement of the Z axis in machine coordinates in NC process 0 of device address 00.

FI command	**00_CR_EPO1_0_2_1**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	-1.2345	[mm]	-1.235

Example EPO1

Read the end point of the movement of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in inches:

FI command	00_CR_EPO1_0_2_1_inch			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z1	-0.0486	[inch]	-0.049

Example EPO2

Read the end point of the movement of the Z axis (physical axis number = 3) in machine coordinates at device address 00.

FI command	00_CR_EPO2_3_1			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z1	-1.2345	[mm]	-1.235

Reference to Literature

See chapter entitled “Literature” [16].

Existing ProVi Types: EPT

MWCX device group

Designation

EPT **E**xisting **P**roVi **T**ypes

Explanation

Which ProVi types are programmed in the current PLC program is queried. The data is returned in a suitable form for the message texts of the small control panels. There is no need to define modules in Moduldef.ini.

FI command

Output all ProVi types.

BR_EPT1 **(Single Read)**

Response Structure

The following table shows the general structure of the "EPT1" FI command. The number of lines depends on the number of ProVi types existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type [11 = Error, 12 = Messages, 20 = Start requirements, 21 = Warnings, 22 = Setup diagnosis]

2 = Index [ASCII characters]

Example EPT1

All ProVi types in control unit 0. There are three diagnosis windows:

FI command		00_BR_EPT1
Line	Column	Answer
1	1	20
	2	0
2	1	11
	2	3
3	1	12
	1	3

Error Status: EST

MWCX device group

Designation **EST** **E**rror **ST**ate

Explanation Queries the error state of a variable.

FI command Query the frozen error status of a variable.

BR_EST1!(1)!(2)	**(Single Read)**
BC_EST1!(1)!(2)	**(Cyclic Read)**
(1) = Error ID	[ASCII characters] (DWORD, decimal)
(2) = Variable name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "EXD1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Error state

WinPcl - Example EST Read the value of WinPcl variable "IB_EXT24" in WinPcl program "Prog", at device address 00.

Assumption:
The WinPcl variable "IB_EXT24" is declared as BOOL in WinPcl program "Prog" .

FI command		**00_BR_EST1!5892855!:Prog.IB_EXT24**
Line	**Column**	**Answer**
1	1	1

Execution Display: EXD

MWCX device group

Designation **EXD** **EX**ecution **D**isplay

Explanation Information for displaying the execution of a movement is output.

FI command Query the execution of a step or of an action.

BR_EXD1!(1)!(2)!(3)	**(Single Read)**
BC_EXD1!(1)!(2)!(3)	**(Cyclic Read)**
(1) = SFC entity name	[ASCII characters]
(2) = Step or action name	[ASCII - characters]
(3) = Behaviour of mode	[1 – all modes, 2 – manual mode]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "EXD1" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Execution [1 – can be executed, 0 – cannot be executed]

Example EXD1

Query the execution of the step “open” for the chain “clamp” in control unit 0 for all modes.

FI command		**00_BR_EXD1!Station03A.Clamp!Open!1**
Line	**Column**	**Answer**
1	1	1

FI command

Query whether the condition analysis (control image) of a step chain is enabled.

BR_EXD2!(1) **(Single Read)**

(1) = SFC entity name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "EXD2" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Enabled [1 - enabled, 0 – not enabled]

Example EXD2

Query whether the condition analysis of the “clamp” chain has been enabled.

FI command		**00_BR_EXD2!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	1

Global Process Parameter Configuration: GPC

MWCX device group

Designation

GPC **G**lobal **P**rocess **C**onfiguration

Explanation

The configuration of the global process parameter of the active machine parameter record of the selected device from the MWCX device group is read out.

The following are all a part of the global process parameters: the programmable and actually displayed digits after the decimal point for the displacement, the name of the NC process, the base programming unit, the max. zero-point-data bank number, D-corrections, whether a basic setting is required, whether a reference is required, whether a transformation between Cartesian and polar coordinates is possible, jogging axis results in a reset and the re-positioning of the tool memory axis.

Note: The FI commands "GPPx" (see Global Process Parameters “GPP”) should be preferred to the FI commands "GPCx" as the access speed there has been optimized.

FI command

Output of the configuration of the global process parameters of all defined NC processes of the active machine parameter record.

BR_GPC1 **(Single Read)**

Response Structure

The following table shows the general structure of the response to the "GPC1" FI command. The response consists of between one and a maximum of n=7 lines (n= max. number of defined NC processes [0...6] = 7), each with 12 columns.

Line 1	Column 1	...	Column 12

Note: If there is no active machine parameter record in the device then the columns [1..0.12] are not applicable.

Value Range/Meaning of Columns

1 =	NC process number	[0...6]
2 =	Name of the NC process	[max. 20 ASCII characters]
3 =	Basic coordinate system	[mm, inch]
4 =	Programmed number of positions after decimal point	[4, 5]
5 =	Displayed positions after the decimal point	[0...4]
6 =	Max. zero-point-data bank number	[0...9]
7 =	D corrections	[YES, NO]
8 =	Home position required	[YES, NO]
9 =	Reference required	[YES, NO]
10 =	Cartesian-polar coordinate transformation	[YES, NO]
11 =	Manual axis jogging causes reset	[YES, NO]
12 =	Tool storage axis repositioning	[YES, NO]

Example GPC1

Read the configuration of the global process parameters of all defined NC processes of the active machine parameter record of device address 00.
Assumption:
The following three NC processes are defined:
Sled 1 (NC process number 0),
Turret 1 (NC process number 1) and
Turret 2 (NC process number 3).

FI command		00_BR_GPC1
Line	**Column**	**Answer**
1	1	0
	2	Sled 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]

FI command		00_BR_GPC1
Line	Column	Answer
	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

Reference to Literature

See chapter entitled “Literature” [34].

FI command

Output the configuration of the global process parameters of the active machine parameter record of the selected device related to the NC process.

BR_GPC2_(1) **(Single Read)**

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

Response Structure

The following table shows the general structure of the response to the “GPC2” FI command. The response consists of a line with 12 columns.

Line 1	Column 1	...	Column 12

Note: If there is no active machine parameter record in the device or the selected NC process is not defined then the columns [1...12] are not applicable.

Value Range/Meaning of Columns

1 =	NC process number	[0...6]
2 =	Name of the NC process	[max. 20 ASCII characters]
3 =	Basic coordinate system	[mm, inch]
4 =	Programmed number of positions after decimal point	[4, 5]
5 =	Displayed positions after the decimal point	[0...4]

6 =	Max. zero-point-data bank number	[0...9]
7 =	D corrections	[YES, NO]
8 =	Home position required	[YES, NO]
9 =	Reference required	[YES, NO]
10 =	Cartesian-polar coordinate transformation	[YES, NO]
11 =	Manual axis jogging causes reset	[YES, NO]
12 =	Tool storage axis repositioning	[YES, NO]

Example GPC2

Read the global process parameter in NC process 0 of the active machine parameter record of device address 00.
Assumption:
The following three NC processes are defined:
Sled 1 (NC process number 0),
Turret 1 (NC process number 1) and
Turret 2 (NC process number 3).

FI command		**00_BR_GPC2_0**
Line	**Column**	**Answer**
1	1	0
	2	Sled 1
	3	[mm]
	4	4
	5	3
	6	0
	7	YES
	8	NO
	9	NO
	10	NO
	11	YES
	12	NO

Reference to Literature

See chapter entitled “Literature” [34].

Global Process Parameters: GPP

MWCX device group

Designation **GPP** **G**lobal **P**rocess **P**arameter

Explanation

The global process parameters of the active machine parameter record of the selected device from the MWCX device group is read out. This includes the programmable and actually displayed digits after the decimal point for the displacement, the name of the CC process, the basic coordinate system and the max. zero-point-data bank number.

Note: The FI commands "GPPx" should be preferred to the FI commands "GPCx" as the access speed has been optimized.

FI command

Output of the configuration of the global process parameters of all defined NC processes of the active machine parameter record.

BR_GPP1 **(Single Read)**

Response Structure

The following table shows the general structure of the response to the "GPC1" FI command. The response consists of one up to a maximum of n=7 lines (n= max. number of defined NC processes [0...6] = 7), each with six columns.

Line 1	Column 1	...	Column 6

Note: If there is no active machine parameter record in the device then the columns [1...6] are not applicable.

Value Range/Meaning of Columns

1 =	NC process number	[0...6]
2 =	Name of the NC process	[max. 20 ASCII characters]
3 =	Basic coordinate system	[mm, inch]
4 =	Programmed number of positions after decimal point	[4, 5]
5 =	Displayed positions after the decimal point	[0...4]
6 =	Max. zero-point-data bank number	[0...9]

Example GPP1

Read the global process parameters of all defined NC processes of the active machine parameter record of device address 00.
Assumption:
The following three NC processes are defined:
Sled 1 (NC process number 0),
Turret 1 (NC process number 1) and
Turret 2 (NC process number 3).

FI command		**00_BR_GPP1**
Line	**Column**	**Answer**
1	1	0
	2	Sled 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

Reference to Literature

See chapter entitled "Literature" [35].

FI command Output the configuration of the global process parameters of the active machine parameter record of the selected device related to the NC process.

BR_GPP2_(1) **(Single Read)**

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

Response Structure The following table shows the general structure of the response to the “GPP2” FI command. The response consists of a line with six columns.

Line 1	Column 1	...	Column 6

Note: If there is no active machine parameter record in the device or the selected NC process is not defined then the columns [1...6] are not applicable.

Value Range/Meaning of Columns

1 =	NC process number	[0...6]
2 =	Name of the NC process	[max. 20 ASCII characters]
3 =	Basic coordinate system	[mm, inch]
4 =	Programmed number of positions after decimal point	[4, 5]
5 =	Displayed positions after the decimal point	[0...4]
6 =	Max. zero-point-data bank number	[0...9]

Example GPP2 Read the global process parameter in NC process 0 of the active machine parameter record of device address 00.

Assumption:
The following three NC processes are defined:
Sled 1 (NC process number 0),
Turret 1 (NC process number 1) and
Turret 2 (NC process number 3).

FI command		**00_BR_GPP2_0**
Line	**Column**	**Answer**
1	1	0
	2	Sled 1
	3	[mm]
	4	4
	5	3
	6	0

Reference to Literature See chapter entitled “Literature” [35].

Insert NC Program Package: IPP

MWCX device group

Designation **IPP** **I**nsert NC-**P**rogram **P**ackage

Explanation Inserts an NC program package into the NC package directory.

FI command **BW_IPP_(1){_(2)}** **(Single Write)**

(1) =	Number in NC package directory	[1...99]
(2) =	Is the NC package	[0 = without check (preset);

directory entry empty? 1 = with check] ! Optional !

Note: If an NC program package already exists at the selected number of the NC package directory, an error is signaled if execution of the check has been selected.

Value to be written

Name of the NC package [max. 32 ASCII characters]

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge NC package has been entered.

Example IPP

Enter the NC program package with the designation "KEY1" into number 1 of the NC package directory.

Assumption:

It is to be checked whether the selected entry in the NC package directory is empty.

FI command		**00_BA_PPP_1** **Value to be written: KEY1**
Line	**Column**	**Answer**
1	1	(P_ACK)

Module Assignment of a Process: MAP

MWCX device group

Designation

MAP **M**odul **A**ssign of **P**rocess

Explanation

The module to which a particular process is assigned is read out from the "Moduldef.ini" file. This file is located in the directory "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" and contains the data for all module configurations.

The process data is located in three sections:

[DeviceAddrX\ModulY\Process]

whereby "X" stands for the device addressed and "Y" for the configured module numbers.

FI command

Determine the module to which the process belongs. Information is read out from the module configuration of the MWCX device group.

BR_MAP1_(1) **(Single Read)**

BC_MAP1_(1) **(Cyclic Read)**

BB_MAP1_(1) **(Break Cyclic Read)**

1 = Mechanism number [0...31]

Response Structure

The following table shows the general structure of the response to the “MAP1” FI command. One line with one column is output for the module number that has been determined.

Line 1	**Column 1**

Value Range of the Column

1 = Module number [0...99]

Example MAP1

Read the module number to which NC process number 4 is assigned from the module configuration.

Assumption:
The module to which NC process 4 is assigned has module number 5.

FI command		00_BR_MAP1_4
Line	**Column**	**Answer**
1	1	5

Reference to Literature

See chapter entitled “Literature” [36].

Read Reference Name of a PLC Variable: MAR

MWCX device group

Designation

MAR **M**ap **A**bsolute PCL-**R**eference

PLC Explanation

The absolute reference name of a symbolic PLC variable is read out.

FI command

Read the absolute reference name of a PLC variable.

BR_MAR_(1) **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command "MAR". One line with one column is output for the reference name that has been determined.

Line 1	Column 1

Meaning of the Column

1 = Identifier of the PLC variable

PLC – Example MAR

Read the absolute reference name of the PLC variable with the identifier “abref” at device address 00.

Assumption:
The PLC variable with the identifier "abref" is of the type "INTEGER".

FI command		00_BR_MAR_abref
Line	**Column**	**Answer**
1	1	%M100.0

WinPcl-Explanation

The absolute reference name of a symbolic WinPcl PLC variable with program entity is read out.

FI command

Read the absolute reference name of a WinPcl PLC variable.

BR_MAR1_(1) **(Single Read)**

(1) = Identifier of the PLC variable

WinPcl - Example MAR1

Read the absolute reference name of the WinPcl variable with the identifier "Prog.abref" at device address 00.

Assumption:
The WinPcl variable with the identifier "Prog.abref" is of the type "INTEGER" and is present in WinPcl program "Prog".

FI command		00_BR_MAR1_:Prog.abref
Line	**Column**	**Answer**
1	1	%M100.0

Reference to Literature

See chapter entitled “Literature” [30].

Device Data of the Module Configuration: MCD

MWCX device group

Designation **MCD** **M**odul **C**onfiguration: **D**evice Information

Explanation All device data for module configuration is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. The device data is in the sections [DeviceAddrX], whereby "X" stands for the configured device addresses.

FI command Read out device data within the module configuration of the MWCX device groups.

BR_MCD1 **(Single Read)**

BC_MCD1 **(Cyclic Read)**

BB_MCD1 **(Break Cyclic Read)**

Note: The “MCD1” FI command refers to all devices within the MWCX device group. Therefore, any valid device address can be indicated in the command line (see example MCD1).

Response Structure The following table shows the general structure of the response to the “MCD1” FI command. The number of lines depends on the number of configured devices. Each line consists of four columns for the device address as well as PLC-FB names for providing setup diagnostics, warning messages and start requirements.

Line 1	Column 1	...	Column 4

Value Range of the Columns

1 = Device address	[0...63]
2 = PLC-FB name for the setup diagnostics	[max. 9 ASCII characters]
3 = PLC-FB name for the warning messages	[max. 9 ASCII characters]
4 = PLC-FB name for the start requirements	[max. 9 ASCII characters]

Example MCD1 Read all device data of the module configuration.

Assumption:
The following devices have been configured in the MWCX device group:

- Device address 01 (MTC200-P)
- Device address 03 (MTC200-P)

FI command	**03_BR_MCD1**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	01	PVSetup_1	PVWarn_1	PVStart_1
2	03	PVSetup_3	PVWarn_3	PVStart_3

Reference to Literature See chapter entitled “Literature” [36].

Module Data of the Module Configuration: MCM

MWCX device group

Designation **MCM** **M**odul **C**onfiguration: **M**odul Information

Explanation All module data for module configuration is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. This file contains all module configuration data. The module data is located in sections [DeviceAddrX\ModulY], whereby "X" stands for the device addressed and "Y" for the configured module numbers.

FI command Read out module data from the module configuration with respect to a device from the MWCX device group.

BR_MCM1 **(Single Read)**

BC_MCM1 **(Cyclic Read)**

BB_MCM1 **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the “MCM1” FI command. The number of lines depends on the number of configured modules of a device. Each line consists of four columns for the module number, module name and PLC-FB names for general module errors and module messages.

Line 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 general module errors	[max. 9 ASCII characters]
4 = PLC-FB name for module messages	[max. 9 ASCII characters]

Example MCM1 Read the module data of device 03 from the module configuration:

<u>Assumption:</u>
The following modules have been defined:

- Module number 5
- Module number 7

FI command	03_BR_MCM1			
Answer				
Line	**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

Reference to Literature See chapter entitled “Literature” [36].

Process Data of the Module Configuration: MCP

MWCX device group

Designation **MCP** **M**odul **C**onfiguration: **P**rocess Information

Explanation All process data of a certain module is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. This file contains all module configuration data. The process data is located in sections [DeviceAddrX\ModulY\Process], whereby "X" stands for the device addressed and "Y" for the selected module number.

FI command

BR_MCP1_(1) **(Single Read)**

BC_MCP1_(1) **(Cyclic Read)**

BB_MCP1_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Response Structure The response to the FI command “MCP1” consists of one up to a maximum number of n=32 lines with 1 column for the number of the NC process or of the external mechanisms.

Line 1...32	Column 1

Value Range of the Column 1 = Mechanism number [0...31]

Example MCP1 Read the NC process number of module 5 of device 00 of the module configuration.

Assumption:
The following NC processes are defined:

NC process number 1

NC process number 4

FI command		00_BR_MCP1_5
Line	**Column**	**Answer**
1	1	1
2	1	4

Reference to Literature See chapter entitled “Literature” [36].

SFC Data of the Module Configuration: MCS

MWCX device group

Designation **MCS** **M**odul **C**onfiguration: **S**FC Information

Explanation All SFC data of a certain module is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. This file contains all module configuration data. The SFC data is located in sections [DeviceAddrX\ModulY\Sfc], whereby "X" stands for the device addressed and "Y" for the selected module number.

FI command Read out the SFC data with respect to the module of a device from the module configuration of the MWCX device group.

BR_MCS1_(1) **(Single Read)**

BC_MCS1_(1)	**(Cyclic Read)**
BB_MCS1_(1)	**(Break Cyclic Read)**
(1) = Module number	[0...99]

Response Structure

The number of lines depends on the number of configured Indrastep step chains for a device. Each line contains a column for the name of the Indrastep step chains.

Value Range of the Column

1 = Name of the Indrastep step chain [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 name of the Indrastep step chain of module 5 from device 03 of the module configuration.

Assumption:
The following Indrastep step chains have been defined:

ISFB_1

- FB_US.ISFB_3
- FB_US.ISFB_3.SW1
- FB_US.ISFB_3.SW1.ABBA

FI command		03_BR_MCS1_5
Line	**Column**	**Answer**
1	1	ISFB_1
2	1	FB_US.ISFB_3
3	1	FB_US.ISFB_3.SW1
4	1	FB_US.ISFB_3.SW1.ABBA

Reference to Literature

See chapter entitled “Literature” [36].

Machine Data Download: MDA

MWCX device group

Designation

MDA **M**achine **D**ata **A**ccess

Explanation

Complete machine data records are downloaded by means of a download file.

FI command

Machine data record download command whereby two predefined functions are to be programmed by the user. These two functions concern:

1. Function for creating the download file itself:

```
Long MachineDataDownloadBegin( Long lProjectNumber,
                               Long lDeviceNumber,
                               Long lIndexNumber,
                               Char* pcMDLFileName,
                               Long lMaxLengthFileNameBuffer,
                               Char* pcErrorText,
                               Long lMaxLengthErrorTextBuffer)
```

Pass parameters:

- lProjectNumber: Currently selected project number
- lDeviceNumber: Currently selected device address
- lIndexNumber: Currently selected machine data record directory number [1..99]
- pcMDLFileName: Contains the complete file names for the created machine data record download file.
- lMaxLengthFileNameBuffer: Max. length of the buffer for the name of the machine data record download file.
- pcErrorText: Text of user error, if applicable
- lMaxLengthErrorTextBuffer: Max. length of the buffer for the user error text.

2. Function called up at the end of the parameter download:

Long MachineDataDownloadEnd(Char* pcMDLFileName,
Long lResult)

Pass parameters:

pcMDLFileName: Contains the complete file names for the created machine data record download file.

lResult: Contains the status message of the downloading process of the machine data record. Here,

0 = Machine data record download procedure O.K.

> 0 = An error occurred

The two functions must be programmed in a DLL by the user and also exported from it.

BW_MDA1_(1)_{(2)} **(Single Write)**

(1) = (1) = Machine data record directory number; the two functions to be implemented are located in INDIF410.DLL.

(2) = (2) = Complete DLL name, if required, in which the two functions to be implemented are located.

Response Structure

The response to the "MDA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example MDA1

00_BW_MDA1_3_"D:\UserDir\USER.DLL"

FI command		00_BW_MDA1_3_D:\UserDir\USER.DLL
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_MDA1_3_D:\UserDir\USER.DLL
3	1	0

FI command

Machine data record download command whereby the machine data record download file is directly indicated.

BW_MDA2_(1) **(Single Write)**

(1) = Download file name of complete machine data record

Response Structure

The response to the "MDA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example MDA2

00_BW_MDA2_"D:\DOWNLOAD.DAT"

FI command		00_BW_MDA2_"D:\DOWNLOAD.DAT"
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_MDA2_"D:\DOWNLOAD.DAT"
3	1	0

Structure of Download File

The structure of the download file corresponds to that of a Windows Ini file. Rexroth Indramat‘s own description in V21_Madat_Download_Upload_01.doc, is recommended for a more detailed account of the structure of the download file.

Summary:

Section [ID_MACHINE_DATA]
Information concerning the identification of the machine data record.

Section [ID_TYPE_DEFINITION]
Information concerning the identification of the type definition.

Section [TYPE_DEFINITION_INFO]
Max. data type identification number

Section [TYPE_DEFINITION_XXX]
Data for the various type definitions.

Section [PAGE_INFO]
Max. defined page number.

Section ID_PAGE_DEFINITION_XXX]
Information concerning the identification of the page definition.

Section [PAGE_DEFINITION_XXX]
Data for the page definitions.

Section [PAGE_DESCRIPTION_XXX_YYY]
Data for writing the individual data elements of a page.

Section [PAGE_DATA_INFO]
Max. defined page number for writing of machine data.

Section [PAGE_DATA_ELEMENTS_XXX]
Information for the machine data that is to be written.

Section [PAGE_DATA_XXX] Data for the machine data that is to be written.

FI command With this command, ALL machine data page definitions in the selected device are deleted.

BW_MDA4 **(Single Write)**

Response Structure The response to the "MDA4" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 =	List of the deleted machine data page numbers, or -- if NO machine data page numbers have been deleted.	List of page numbers separated by comma or by --

Example MDA4 The following machine data page definitions have been deleted for the device 00.

FI command		00_BW_MDA4
Line	**Column**	**Answer**
1	1	1,2,10,11,12,21,30,40,50,60,61,62,90,91,92,101,102,103,104

Machine Data Upload: MDA

MWCX device group

Designation **MDA** **M**achine **D**ata **A**ccess

Explanation Uploads complete machine data records from a selected device. The data read is written into an upload file with an identical structure to that of a download file.

FI command Machine data record upload command whereby two predefined functions are to be programmed by the user. These two functions concern:

1. The function supplies the complete name of the upload file:

```
Long MachineDataUploadBegin(   Long lProjectNumber,
                               Long lDeviceNumber,
                               Char* pcUploadFileName,
                               Long lMaxLengthFileNameBuffer,
                               Char* pcErrorText,
                               Long lMaxLengthErrorTextBuffer)
```

<u>Pass parameters:</u>

- lProjectNumber: Currently selected project number
- lDeviceNumber: Currently selected device address
- pcUploadFileName: Contains the complete file name for the machine data record upload file to be created.

- lMaxLengthFileNameBuffer: max. length of the buffer for the name of the machine data record upload file.
- pcErrorText: If necessary, user error text
- lMaxLengthErrorTextBuffer: Max. length of the buffer for the user error text.

2. Function called up at the end of the machine data record upload:

Long MachineDataUploadEnd(Char* pcUploadFileName,
Long lResult)

Pass parameters:

pcUploadFileName: Contains the complete file names for the created machine data record upload file.

lResult: Contains the status message of the uploading process of the machine data record. Here,
0 = Parameter upload procedure O.K.
> 0 = Error has occurred

The two functions must be programmed in a DLL by the user and also exported from it.

BR_MDA1_(1)_{(2)} **(Single Read)**

(1) = Machine data record directory number; the two functions to be implemented are located in INDIF410.DLL.

(2) = Complete DLL name, if required, in which the two functions to be implemented are located.

Response Structure

The response to the "MDA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID
  [01...20] (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example MDA1

00_BR_MDA1_2_"D:\UserDir\USER.DLL"

FI command		**00_BR_MDA1_2_"D:\UserDir\USER.DLL"**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_MDA1_2_"D:\UserDir\USER.DLL"
3	1	0

FI command

Machine data record upload command whereby the machine data record upload file is directly indicated.

BR_MDA2_(1) **(Single Read)**

(1) = complete machine data record upload file name

Response Structure

The response to the "MDA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID
  [01...20] (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example MDA2

00_BR_MDA2_"D:\UPLOAD.DAT"

FI command		00_BR_MDA2_"D:\UPLOAD.DAT"
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_MDA2_"D:\UPLOAD.DAT"
3	1	0

Structure of Upload File

The structure of the upload file corresponds to that of a Windows Ini file. Rexroth Indramat‘s own description in "V21_Madat_Download_Upload_01.doc", is recommended for a more detailed account of the structure of the download file.

For a summary refer to the description under Machine Data Record Download Command.

Inputting an NC Record: MDI

MWCX device group

Designation

MDI **M**anual **D**ata **I**nput

FI command

Input an NC record for direct execution in manual mode.

CW_MDI_(1) **(Single Write)**

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

Value to be written

NC record (see DOK-MTC200-NC**PRO*V..)

Note: The value to be written is passed to the "acValue" parameter as an ASCII string in the "DataTransfer" routine.

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Data element has been set

Example MDI

Write an NC record for direct execution in NC process 0.

Conditions

The control unit must be in “Setup” (“Manual”) mode. Axes X1 and Y1 exist.

Value to be written

G01 X1 50.45 Y1 35.456 F 1000

FI command		00_CW_MDI_0
Line	**Column**	**Answer**
1	1	(P_ACK)

or, if the process is not ready for the next NC record: Error 1014 = BOF_NEGATIVE_ACKNOWLEDGE (N_ACK):

Line	Column	Answer
1	1	**1** (=N_ACK)
	2	**37** (=text number of N_ACK)
	3	**0x00000000** (=additional information for some texts)
	4	**Process still active** (=text of the N_ACK error)

or, if a syntax error is detected in the passed NC record: Error 1014 = BOF_NEGATIVE_ACKNOWLEDGE (N_ACK):

Line	Column	Answer
1	1	**1** (=N_ACK)
	2	**18** (=text number of N_ACK)
	3	**0xFFFFFFFC** (=additional information for some texts)
	4	**Unrecognized expression in the NC program**

Monitoring the MDI Status

During MDI operation the status of the process should be monitored by reading the diagnostic message:

BR_AMM3_(1) **(Single Read)**

.

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

Example 1

Before inputting an NC record:

FI command		00_BR_AMM3_0
Line	**Column**	**Answer**
1	1...8	(For process information see Documentation BR_AMM3)
	9	Ready to start: Operating mode 'setup"

Example 2

After inputting an NC record:

FI command		00_BR_AMM3_0
Line	**Column**	**Answer**
1	1...8	(For process information see documentation BR_AMM3)
	9	Ready to start for processing of MDI record

Possible error codes:
Example 3

Assumption:
It is not possible to process the NC record because of an erroneous expression.

FI command		00_BR_AMM3_0
Line	**Column**	**Answer**
1	1...8	(For process information see documentation BR_AMM3)
	9	Invalid axis

Example 4

Assumption:
External start conditions are missing for the process to execute the NC record.

FI command		00_BR_AMM3_0
Line	Column	Answer
1	1...8	(For process information see documentation BR_AMM3)
	9	Failure of external 24 Volt supply

Reference to Literature See chapter entitled “Literature” [4].

Reading and Writing Machine Data: MDS

MWCX device group

Designation **MDS** **M**achine **D**ata **S**ingle

Explanation For reading and writing a list of machine data. A maximum of 10 machine data items can be written or read at the same time.

FI command Write machine data.

BW_MDS1_(1)_{(2)..(10)} **(Single Write)**

(1) = Info string for the machine data date to be written

Structure of the info strings:
Data type\page number\data element\
Travel variable1\travel variable2\data value\
data unit
Data type:
Value according to the machine data type definition (normal: 1..29)
Page number:
Value according to page definition
Data element:
Value according to page definition
Travel variable1:
Value according to page definition
Travel variable2:
Value according to page definition
Data value:
Value to be written
Data unit:
String of units

Response Structure The following table shows the general structure of the response to the "MDS1" FI command. There is one response line for each write value.

Line 1...n	Column 1	...	Column 5

Value Range/Meaning of Columns

1 = Status message with regard to write procedure
- 1 = Write procedure has been executed successfully.
- 0 = Write procedure could NOT be executed.

2 = Info string for the machine data date to be written
- See syntax of the MDS command

3 = Error class
- 0 = No error
- 1 = Communication error (NACK)
- 2 = FI error
- 3 = Error text

4 =	Short message text	If the write procedure has been executed successfully, then --, otherwise the short error text is given.
5 =	Reference information	If the write procedure has been executed successfully, then --, otherwise reference information is given.

Example MDS1

Two machine data values are written:

1. Value:

Data type: DREAL (ID number: 13)

Page number: 123

Data element: 1

Travel variable1: 0

Travel variable2: 0

Data value: 123.23

Data unit: NONE (encoded as – !)

2. Value:

Data type: POS (ID number: 14)

Page number: 103

Data element: 3

Travel variable1: 1

Travel variable2: 2

Data value: 100.00

Data unit: mm

FI command		**00_BW_MDS1_13\123\1\0\0\123.23_-\14\103\3\1\2\100.00\mm**
Line	**Column**	**Answer**
1	1	1
	2	13\123\1\0\0\123.23\-\
	3	0
	4	--
	5	--
2	1	1
	2	14\103\3\1\2\100.00\mm
	3	0
	4	--
	5	--

FI command

Read machine data.

BR_MDS1_(1)_{(2)..(10)}		**(Single Read)**
(1) =	Info string for the machine data date to be read	Structure of the info strings: Data type\page number\data element\ Travel variable1\travel variable2 **Data type:** Value according to the machine data type definition (normal: 1..29) **Page number:** Value according to page definition **Data element:** Value according to page definition **Travel variable1:** Value according to page definition **Travel variable2:** Value according to page definition

Response Structure

The following table shows the general structure of the response to the "MDS1" FI command. There is one response line for each value read.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	Data value read as a string	If an error occurs during reading – is given.
2 =	Data unit read as a string	If an error occurs during reading – is given.
3 =	Number of places after the decimal point	If an error occurs during reading – is given.
4 =	Info string for the machine data date to be read	See syntax of the MDS command.
5 =	Error class	0 = No error 1 = Communication error (NACK) 2 = FI error 3 = Error text
6 =	Short message text	If the read procedure has been executed successfully, then --, otherwise the short error text is given.
7 =	Reference information	If the read procedure has been executed successfully, then --, otherwise reference information is given.

Example MDS1

Three machine data values are read:

1. Value:

Data type: DREAL (ID number: 13)

Page number: 101

Data element: 1

Travel variable1: 0

Travel variable2: 0

2. Value:

Data type: POS (ID number: 14)

Page number: 122

Data element: 3

Travel variable1: 1

Travel variable2: 2

3. Value:

Data type: WORD (ID number: 3)

Page number: 122

Data element: 4

Travel variable1: 1

Travel variable2: 2

FI command		**00_BR_MDS1_13\101\1\0\0_14\122\3\1\2_3\122\4\1\2**
Line	**Column**	**Answer**
1	1	111.11
	2	
	3	0
	4	13\101\1\0\0
	5	0
	6	--
	7	--
2	1	66.6666
	2	MM
	3	4
	4	14\122\3\1\2
	5	0
	6	--
	7	--
3	1	10
	2	
	3	0
	4	3\122\4\1\2
	5	0
	6	--
	7	--

Reference to Literature See chapter entitled "Literature" [33].

Downloading Message Texts: MFD

MWCX device group

Designation **MFD** **M**essage **F**iles **D**ownload

FI command This is used to load the message texts into the device indicated. These message texts are required for small devices. The following message texts are transmitted, depending on the type of device:

- system error messages
- transmission error messages
- mechanism messages

Note: This FI command is an FI job!

BW_MFD1 **(Single Write)**

Response Structure

The response to the "MFD1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID	[01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).
Line 2 = FI command	[String] (in accordance with the chapter entitled "Elements of the FI Command")
Line 3 = FI job error code	(see chapter entitled “Error Codes”)

Example MFD1

Load message texts into the device with device address 00.

FI command		**00_BW_MFD1**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_MFD1
3	1	0

Maximum Feedrate Override: MFO

MWCX device group

Designation

MFO **M**aximal **F**eedrate **O**verride

Explanation

The value of the maximum feedrate override for the selected device of the MWCX device group is read out.

FI command

CR_MFO1_(1)	**(Single Read)**
CC_MFO1_(1)	**(Cyclic Read)**
CB_MFO1_(1)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]

Response Structure

The following table shows the general structure of the response to the “MFO1” FI command. One line with three columns is output for the identifier, the current value of the maximum feedrate override and the unit [%].

Line 1	**Column 1**	**....**	**Column 3**

Value Range/Meaning of Columns

1 = Identifier	[MAX]
2 = Value of maximum feedrate override	[0...100]
3 = Unit	[%]

Example MFO1

Read the current value of the maximum feedrate override in NC process 0 of device address 00.

FI command	**00_CR_MFO1_0**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	MAX	100	[%]

Reference to Literature

See chapter entitled “Literature” [9].

Maximum Feedrate: MFR

MWCX device group

Designation **MFR** **M**aximal **F**eed **R**ate

Explanation The value of the maximum feedrate of an NC process is output.

FI command Output the value of the maximum feedrate.

Using the optional second parameter it is possible to pre-select conversion of the result into mm or inches.

CR_MFR_(1){_(2)}	**(Single Read)**
CC_MFR_(1){_(2)}	**(Cyclic Read)**
CB_MFR_(1){_(2)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Required measurement system (opt.)	[mm, inch]

Response Structure The following table shows the general structure of the response to the FI command "MFR". One line with three columns is output for the identifier, the current value of the maximum feedrate and the unit.

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier	[F = feedrate]
2 = Feedrate overrides	[format, according to settings of the parameters,--]
3 = Unit	[according to settings of the parameters]

Example MFR Read the value of the maximum feedrate in NC process 0 of device address 00.

FI command	00_CR_MFR_0		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	F	120000.0	[mm/min]

Example MFR Read the value of the maximum feedrate in NC process 0 of device address 00. The displayed value is to be converted into inch/min:

FI command	00_CR_MFR_0_inch		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	F	4724.4	[inch/min]

Reference to Literature See chapter entitled “Literature” [18].

Reading Machine Key Information : MKS

MWCX device group

Designation **MKS** **M**achine **K**ey **S**tatus

Explanation Current machine key information can be read for the selected device.

FI command Read machine key information for selected device.

BR_MKS **(Single Read)**

BC_MKS **(Cyclic Read)**

BB_MKS **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the FI command "MKS".

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 = Machine key information [4 byte in HEX coding]

2 = Information valid? [0 = not valid, 1=valid]

Example MKS Read the current machine key information for device 0.

FI command		00_BR_MKS
Line	**Column**	**Answer**
1	1	00000000
	2	0

Writing the GUI-SK Block: MKT

MWCX device group

Designation **MKT** **M**achine **K**ey **T**able

Explanation Writes the GUI-SK16 block in the PLC.

FI command Write GUI-SK16 block.

BW_MKT1_(1) **(Single Write)**

(1) = List of the 48 PLC variables for writing the GUI-SK16 block.

A distinction is made between the following cases:

1. Clear GUI-SK16 block.
2. Write the GUI-SK16 block with the 48 PLC variables, filling gaps with $SPACE.

Response Structure (P_ACK) is returned following successful transmission.

Line 1	Column 1

Value Range/Meaning of the Columns

1 = Successfully completed (P_ACK)

1. Example MKT1 1.Clear GUI-SK16 block:

FI command		00_BW_MKT1 Value to write: $EMPTY
Line	**Column**	**Answer**
1	1	(P_ACK)

2. Example MKT1

Write GUI-SK16 block:

FI command		00_BW_MKT1 Value to write: SPSVAR1,SPSVAR2,$SPACE,...
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

Write the GUI-SK16 block, writing only those PLC variables which are defined in the current PLC program. All undefined PLC variables are automatically replaced by $SPACE and returned as a partial result (column 2).

BW_MKT2_(1) **(Single Write)**

(1) = List of the 48 PLC variables for writing the GUI-SK16 block.

A distinction is made between the following cases:

1. Clear GUI-SK16 block:
   BW_MKT2 $EMPTY
2. Write the GUI-SK16 block with the 48 PLC variables, filling gaps with $SPACE:
   BW_MKT1
   SPSVAR1,SPSVAR2,
   $SPACE,$SPACE,....

Response Structure

After successful transmission, one line with two columns is returned.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 =	Status report	[0 = at least 1 PLC variable in the current PLC program is NOT defined 1 = ALL PLC variables could be written]
2 =	List of the NON-defined PLC variables in the current PLC program	[-- = ALL PLC variables could be written, or else list of the PLC variables that could not be written.] The individual PLC variables are separated by a comma.

Example MKT1

Write GUI-SK16 component with 48 PLC variables, while the PLC variables SPSVAR11 and SPSVAR12 are NOT defined in the current PLC program.

FI command		00_BW_MKT2 Value to be written: SPSVAR1,SPSVAR2,...SPSVAR48
Line	**Column**	**Answer**
1	1	(P_ACK)

Extended information

The variables are divided into 3 groups of 16 variables each and have the following meaning:

1. Variables 1 – 16: Machine function keys
2. Variables 17 – 32: Status pressed
3. Variables 33 -48: Status shining

Notes:	When, for example, only the first 8 M keys are used, the telegram will contain only these 8 PLC variables. The other 40 variables need not be defined in the transmission parameter. When certain areas, e.g. of M keys, are left unused, they must be filled up with '$SPACE' up to the next variable.

Maxim Rapid Override: MRO

MWCX device group

Designation **MRO** **M**aximal **R**apid **O**verride

Explanation The value of the maximum rapid override of the selected device of the MWCX device group is read.

FI command

CR_MRO1_(1) **(Single Read)**

CC_MRO1_(1) **(Cyclic Read)**

CB_MRO1_(1) **(Break Cyclic Read)**

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

Response Structure The following table shows the general structure of the response to the “MRO1” FI command. One line with three columns is output for the identifier, the current value of the maximum rapid override and the unit [%].

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier [RMAX]

2 = Value of maximum rapid override [0...100]

3 = Unit [%]

Example MRO1 Read the maximum value of the rapid override in NC process 0 of device address 00.

FI command	00_CR_MRO1_0		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	RMAX	100	[%]

Reference to Literature See chapter entitled “Literature” [18].

Read System Messages: MSG

MWCX device group

Designation **MSG** **M**e**S**sa**G**e

Explanation Reading of system messages

FI command Message

CC_MSG_(1) **(Cyclic Read)**

(1) = SYS-Message number

Note:	Exists only as a cyclic command

Response Structure The response of the FI command 'MSG' consists of the system message data.

Example MSG 00_CC_MSG_64 (64 = MSG_SYSERRGEN)

FI command		00_CC_MSG_64/3
Line	**Column**	**Answer**
1	1	00

Restriction The following system messages:

SYS Message	SYS message numbers
MSG_PCLUPDBEG	52
MSG_PARUPDBEG	24
MSG_FWAUPDBEG	82

cannot be used with the following programs:

Indramat OPC-Server

Indramat DDE-Server

Maximum Spindle Override: MSP

MWCX device group

Designation **MSO** **M**aximal **S**pindle **O**verride

Explanation The value of the maximum spindle override of the selected device of the MWCX device group is read.

FI command

CR_MSO1_(1)_(2)	**(Single Read)**
CC_MSO1_(1)_(2)	**(Cyclic Read)**
CB_MSO1_(1)_(2)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Number of spindle	[1...3]

Response Structure The following table shows the general structure of the response to the “MSO1” FI command. One line with three columns is output for the identifier, the value of the maximum spindle override and the unit [%].

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier	[SMAX]
2 = Value of maximum rapid override	[0...100]
3 = Unit	[%]

Example MSO1 Read the maximum value of the spindle override in NC process 0 of device address 00.

FI command	00_CR_MSO1_0_1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	SMAX	100	[%]

Reference to Literature See chapter entitled “Literature” [21].

Maximum Spindle Speed: MSS

MWCX device group

Designation **MSS** **M**aximal **S**pindle **S**peed

Explanation The value of the maximum spindle speed of the selected device of the MWCX device group is read out.

FI command

CR_MSS_(1)_(2)	**(Single Read)**
CC_MSS_(1)_(2)	**(Cyclic Read)**
CB_MSS_(1)_(2)	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Number of spindle	[1...3]

Response Structure The following table shows the general structure of the response to the "MSS" FI command. One line with three columns is output for the identifier, the speed and the unit [1/min].

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier	[S = spindle]
2 = Speed	[format according to settings of the parameters]
3 = Unit	1/min

Example MSS Read the maximum value of the speed of the 1st spindle in NC process 0 of device address 00.

FI command	00_CR_MSS_0_1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	S:	7500.0	1/min

Reading the Firmware Identification: MTC

MWCX device group

Designation **MTC** **MT-C**NC Slot Software Version

FI command This command is used to read the firmware identification from the various control components (slot numbers).

Note: For the time this FI command is executed, the internal FI communication interlocks (fast timeout monitoring, offline operation, etc.) are switched off.

FI command

BR_MTC_(1)	**(Single Read)**
(1) = Slot number	[1=CNC, 2=SIO, 3=PLC, 4=APR1 5=APR2, 6=APR3, 7=APR4]

Response Structure The following table shows the general structure of the response to the FI command "MTC". A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Firmware identification string [max. 16 ASCII characters]

Example MTC Read the firmware identification of slot number 1 (CPU) of device 00.

FI command		00_BR_MTC_1
Line	**Column**	**Answer**
1	1	CPU01/0004-20V00

FI command

This command is used to read the firmware identification from the various control components (slot numbers).

CR_MTC_(1)	**(Single Read)**
(1) = Slot number	[1=CNC, 2=SIO, 3=PLC, 4=APR1 5=APR2, 6=APR3, 7=APR4]

Response Structure

The following table shows the general structure of the response to the FI command "MTC". A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Firmware identification string [max. [16 ASCII characters]

Example MTC

Read the firmware identification of slot number 1 (CPU) of device 00.

FI command		00_CR_MTC_1
Line	**Column**	**Answer**
1	1	CPU01/0004-20V00

User Machine Data: MTD

MWCX device group

Designation **MTD** **M**achine **T**able **D**ata

FI command

Output of user machine data.

CR_MTD1_(1)_(2)_(3)_(4)_(5)	**(Single Read)**
CC_MTD1_(1)_(2)_(3)_(4)_(5)	**(Cyclic Read)**
(1) = Page number	[1...299]
(2) = Run variable 1	[-1000 ... +1000]
(3) = Run variable 2	[-1000 ... +1000]
(4) = Element number	[1...1000]
(5) = Name	[1...13]

Answer

Data element
10110100
Read from the MD page 152 via LV1: 0 and LV2: 1 the 13th element of the type UDINT CR_MTD_152_0_1_13_8 150

FI command

Write machine table data.

CW_MTD1_(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) = identifier code	[1...13]

Code	Identifier	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 [acc. to the display format of the BOF]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure

One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge Value has been successfully transmitted

NC Program Compile and Download: NCA

MWCX device group

Designation

NCA **NC** Program **A**ccess

Explanation

NC programs are downloaded via a download file and NC program files and via all active processes.

FI command

NC program download.

BW_NCA1_(1) **(Single Write)**

(1) = Download file with path details.

Note: Enclose file and path details in inverted commas.

Response Structure

The response to the "NCA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

....

Example NCA1 00_BW_NCA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		**00_BW_NCA1_"D:\Program Files\Indramat\Mtgui\ Temp\download.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_NCA1_"D:\Program Files\Indramat\Mtgui\ Temp\download.ini"/3
3	1	0

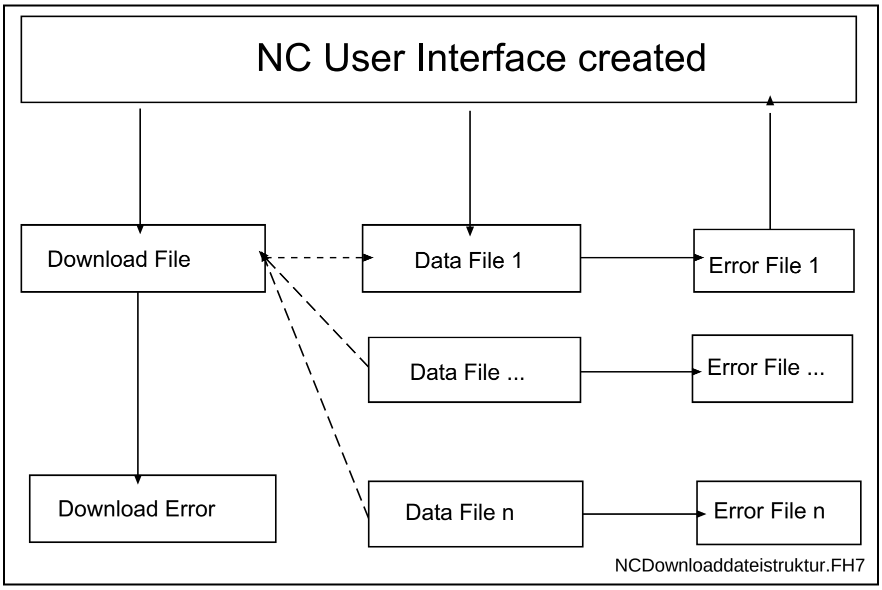


Fig. 7-5: File structure of the download file

Structure of Download File The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
General information is stored in the COMMON section.

Key **Max_Compiler**
Number of compilers to be called. The compiler contains the control file as a pass parameter and translates the data into the respective data files. A pass value of zero signifies no compiler call.

This key is an optional value. If this value is not present, no compiler is active.

Key **DownloadError**
Indicates whether or not an error has occurred during downloading. This value is only set in the event of an error.

Example:

[Common]
DownloadError = YES ; Error
Max_Compiler = 2

Section [CompilerXX]
This section contains information regarding the compiler. There is a separate section for each compiler. The name of the section consists of the “Compiler” text and a two digit number.

XX: is a two digit index which begins at 1 and has a maximum size of Max_Compiler.

Section [NCPackage_Info]

Key **Memory**
Indicates the memory into which the NC program package is loaded.
Memory=1 ;Memory A
Memory=2 ;Memory B

Key **Program package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number	"**PackageNo**"	max. 2 characters
Package name	"**PackageName**"	max. 32 characters
Package size:	"**PackageSize**"	max. 8 characters left-justified
Package time:	"**PackageTime**"	max. 8 characters
Package date:	"**PackageDate**"	max. 8 characters
Package default:	"**PackageDefault**"	max. 26 characters (optional)
Total:		max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

[NCPackage_Info]
Memory=
PackageNo = 1
PackageName = NC program package
PackageSize = 1234
PackageTime = 13:10:10
PackageDate = 24.12.00

Section [ListOfNCPrograms]
The list of NC programs to be transferred is stored in the ListOfNCPrograms section.

Key **Max_Index_Data**
Corresponds to the number of NC programs to be transmitted.

Key **consecutive index of the NC programs**
Four-digit number starting with 1, identifies with a value the full file name of the NC programs including the setup lists. The names of the NC programs and setup lists are structured as follows:

zzzzzzz	Data type (NC-PRG or SetupList)
xx	Process number
yyy	Program number of the NC program (with free NC programs, the index number)

The file extension can be freely selected. ".dat" has been used in the following example.

Examples:

NC-PRG-00-86	N program for process 0 program 86
SETUPLIST-03-25	Setup list for process 3 program 25

Example:

[ListOfNCPrograms]
Max_Index_Data=50
0001=K:\Program Files\Indramat\Mtgui\Project_000\\NC-PRG-00-01.Dat
0002=K:\Program Files\Indramat\Mtgui\Project_000\\NC-PRG-01-01.Dat

0050=K:\Program Files\Indramat\Mtgui\Project_000\\NC-PRG-06-99.Dat

Data File Structure

These contain the actual data for downloading or for the compiler. The structure corresponds to that of the Windows "Ini" structure. The compiler uses this file for the input and output data.

Note: Care must be taken in the use of upper and lower case letters.

Data for the NC program is stored in the respective files as a section. It is composed of general data and the actual program.

Section [Common]

Program version:	**Version**	
Process:	**Process**	[0..6]
Program number:	**No**	[0.0.99]
Program name:	**Name**	max. 32 characters
Program size:	**Size**	
Program time:	**Time**	max. 8 characters
Program date:	**Date**	max. 8 characters
Program short identification:	**ShortID**	max. 8 characters
Program status:	**Status, (optional)**	

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Status flag	Description
C	Compiled
E	Error
The marked section is then printed out.	Not compiled
No details	No compiler call

Fig. 7-6: Description of the status flags

Section Data
Key **Max_Index_Data**
Corresponds to the number of NC blocks to be transferred.

Key **consecutive index of NC records**
Five-digit number starting with 1.

Note: An NC block should not contain any unnecessary blank spaces or NC comments. Equally, “PROGRAM END” may not occur as it is language-dependent.

Example:

[Data]
Max_Index_Data=25
00001=N0000 G0 X0 Y0 Z0
...
00025=N0024 .Start

Explanation

This FI command merely compiles NC programs without triggering the subsequent download. Compiling of NC programs is done through an administration file and NC program files.

FI command

NC program compile.

BW_NCA3_(1) **(Single Write)**

(1) = Administration file with path details.

Note: Enclose file and path details in inverted commas.

Response Structure

The response to the NCA3 FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

....

Example NVA

00_BW_NCA3_"D:\Program Files\Indramat\Mtgui\Temp\compile.ini"/3

FI command		**00_BW_NCA3_"D:\Program Files\Indramat\Mtgui\ Temp\compile.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_NCA3_"D:\Program Files\Indramat\Mtgui\ Temp\compile.ini"/3
3	1	0

NC Program Upload: NCA

MWCX device group

Designation

NCA **NC**-Program **A**ccess

Explanation

NC programs are uploaded via all active processes; during upload, a basic file (upload file) and NC program files are created.

FI command

NC-Program upload.

BR_NCA1_(1)_(2) **(Single Read)**

(1) = NC memory [1 = NC memory A, 2 = NC memory B]

(2) = Upload file with path details

Note: Enclose file and path details in inverted commas.

In this command, the progress information is implemented in %. It can be interrogated via the command IFJ of the MPCX device group.

Response Structure

The response to the "NCA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]

- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NCA 00_BR_NCA1_1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_NCA1_"D:\Program Files\Indramat\Mtgui\ Temp\upload.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_NCA1_"D:\Program Files\Indramat\Mtgui\ Temp\upload.ini"/3
3	1	0

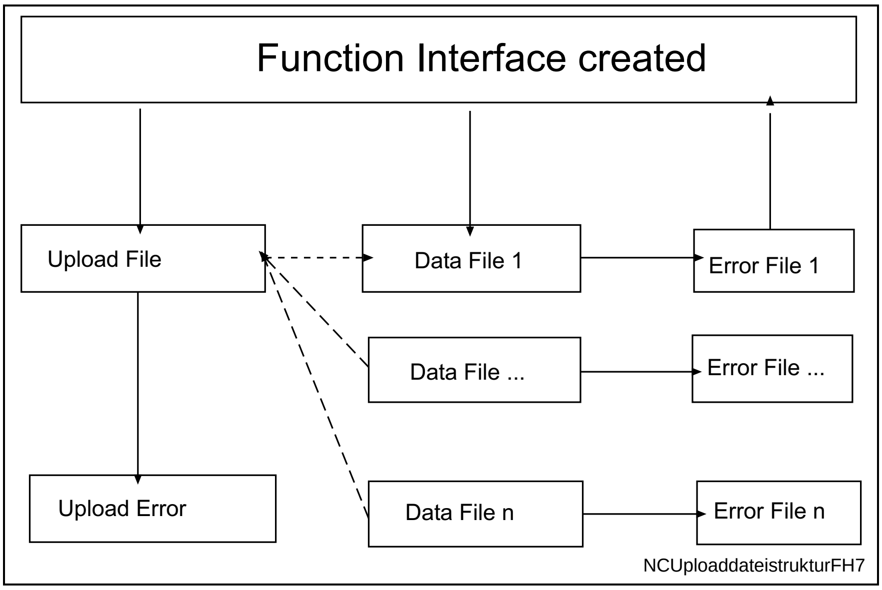


Fig. 7-7: File structure of the upload file

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
General information is stored in the COMMON section.

Key **UploadError**
Indicates whether or not an error has occurred during uploading. This value is only set in the event of an error.

Example:

[Common]
UploadError = YES ; error

Section NC Program package information [NCPackage_Info]
Key **Memory**
Identifies the memory into which the NC program package is loaded.
Memory=1 ;Memory A
Memory=2 ;Memory B

Key **Program package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters
Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters
Package date: "**PackageDate**" max. 8 characters
Package default:"**PackageDefault**" max. 26 characters (optional)

Total: max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

[NCPackage_Info]
Memory=
PackageNo = 1
PackageName = NC program package
PackageSize = 1234
PackageTime = 13:10:10
PackageDate = 24.12.00

Section list of NC programs [ListOfNCPrograms]
The list of the NC programs to be transferred is stored in the section "ListOfCycPrograms".

Key **Max_Index_Data**
Corresponds to the number of NC programs to be transmitted.

Key **consecutive index of the NC programs**
Four-digit number starting with 1, identifies with a value the full file name of the NC programs including the setup lists. The names of the NC programs and setup lists are structured as follows:

zzzzzzz	Data type (NC-PRG or SetupList)
xx	Process number
yyy	Program number of the NC program (with free NC programs, the index number)

The file extension can be freely selected. ".dat" has been used in the following example.

Examples:

NC-PRG-00-086	NC program for process 0 program 86
SETUPLIST-03-025	Setup list for process 3 program 25

Example:

[ListOfNCPrograms]
Max_Index_Data=50
0001=K:\Program Files\Indramat\Mtgui\Project_000\NC-PRG-00-001.dat
0002=K:\Program Files\Indramat\Mtgui\Project_000\NC-PRG-01-001.dat
...
0050=K:\Program Files\Indramat\Mtgui\Project_000\NC-PRG-06-099.dat

Data File Structure

Contains the actual data for the upload. Their structure corresponds to the Windows “Ini” structure.

Note: Care must be taken in the use of upper and lower case letters.

Data for the NC program is stored in the respective files as a section. It is composed of general data and the actual program.

Section [Common]

Program version:	**Version**	
Process:	**Process**	[0..6]
Program number:	**No**	[0.0.99]
Program name:	**Name**	max. 32 characters
Program size:	**Size**	
Program time:	**Time**	max. 8 characters
Program date:	**Date**	max. 8 characters
Program short identification:	**ShortID**	max. 8 characters
Program status:	**Status, (always 'N')**	

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Status flag	Description
C	Compiled
E	Error
The marked section is then printed out.	Not compiled
No details	No compiler call

Fig. 7-8: Status flags

Section [Data]

Key **Max_Index_Data**
Corresponds to the number of NC blocks to be transmitted

Key **consecutive index of NC records**
Five-digit number starting with 1.

Example:

```
[Data]
Max_Index_Data=25
00001=N0000 G0 X0 Y0 Z0
...
00025=N0024 .Start
```

NC Messages: NCM

MWCX device group

Designation **NCM** **NC M**essages

Explanation Indramat NC messages are output. These messages are assigned to a specific module and message type.

FI command Output all NC messages.

BR_NCM1_(1)_(2)	**(Single Read)**
BC_NCM1_(1)_(2)	**(Cyclic Read)**
(1) = Message type	[1 = error, 2 = messages]
(2) = Module number	[1...99]

Output of first NC message.

BR_NCM2_(1)_(2)	**(Single Read)**
BC_NCM2_(1)_(2)	**(Cyclic Read)**
(1) = Message type	[1 = error, 2 = messages]
(2) = Module number	[1...99]

Response Structure The following table shows the general structure of the FI commands "NCM1" and "NCM2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 8

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[ASCII characters]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Mechanism number	[0..31]
6 = 2 byte additional information	[ASCII characters]
7 = NC note	[ASCII characters]
8 = Reference text exists	[YES, NO]
9 = Filename for additional information for message text	[e.g.HTML format]

Example NCM1 All NC errors from module 3 in control unit 0.

There are two messages.

FI command		00_BR_NCM1_1_3
Line	**Column**	**Answer**
1	1	24 volt supply absent
	2	12
	3	01.27.2000
	4	14:56:32
	5	0
	6	
	7	[Note 1]
	8	YES
	9	
2	1	Program stop
	2	152
	3	01.27.2000
	4	15:03:10
	5	1
	6	
	7	
	8	NO
	9	

Example NCM2

The first NC errors from module 3 in control unit 0.

There are two messages:

FI command		00_BR_NCM2_1_3
Line	**Column**	**Answer**
1	1	24 volt supply absent
	2	12
	3	01.27.2000
	4	14:56:32
	5	0
	6	
	7	[Note 1]
	8	YES
	9	

NC Events Download: NEA

MWCX device group

Designation **NEA** **N**C-**E**vent **A**ccess

Explanation NC events are downloaded by means of the download file via all processes.

FI command Download NC events.

BW_NEA1_(1) **(Single Write)**

(1) = Download file with path details.

Note: Enclose file and path details in inverted commas.

Response Structure The response to the "NEA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID
  [01...20] (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NEA1 00_BW_NEA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		**00_BW_NEA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_NEA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
3	1	0

Structure of the download file The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
This is currently only used for error processing, i.e., if an error is detected during a process, then the *DownloadError* key is written with "YES" within this section.

Example:

[Common]
DownloadError = YES ; error

Section NC events information [NCEventsPackage_Info]
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters
Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters

Package date:	"**PackageDate**"	max. 8 characters
Package default:	"**PackageDefault**"	max. 26 characters (optional)
----------------	----------------	----------------
Total:		max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

```
[NCEventsPackage_Info]
PackageNo =        1
PackageName =      NC events
PackageSize =      1234
PackageTime =      13:10:10
PackageDate =      24.12.00
```

Section NC events download [NCEvents_A]

A: corresponds to a process number [0..6]

For external events the section name is extended with X in place of the process number.

A section entry ([NCEvents_A]) is an optional entry, i.e., if a section for a process is absent it is not regarded as an error.

Key values correspond to the event numbers [0..31] and values are the write values of the NC events. Missing key values are not regarded as errors.

```
[NCEvents_0]
000=0
001=1
...
031=1

[NCEvents_1]
000=1
...
016=1

[NCEvents_6]
000=1
010=0
031=1

[NCEvents_X]
000=1
010=0
031=1
```

NC Events Upload: NEA

MWCX device group

Designation **NEA** **N**C-**E**vent **A**ccess

Explanation NC events are uploaded through all processes and external events.

FI command Upload NC events.

BR_NEA1_(1) **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.

In this command, the progress information is implemented in %. It can be interrogated via the command IFJ of the MPCX device group.

Response Structure The response to the NEA1 FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NVA 00_BR_NEA1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_NEA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_NEA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Section [Common]
General information is stored in the COMMON section.

Key **UploadError**
Indicates whether or not an error has occurred during uploading. This value is only set in the event of an error.

Example:

[Common]
UploadError = YES ; error

Section NC Variables Information [NCEventsPackage_Info]

Key **Program package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**"	max. 2 characters
Package name "**PackageName**"	max. 32 characters
Package size: "**PackageSize**"	max. 8 characters left-justified
Package time: "**PackageTime**"	max. 8 characters
Package date: "**PackageDate**"	max. 8 characters
Package default:"**PackageDefault**"	max. 26 characters (optional)
--	--
Total:	max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

```
[NCEventsPackage_Info]
PackageNo =         1
PackageName =       NC events
PackageSize =       1234
PackageTime =       13:10:10
PackageDate =       24.12.00
```

Section NC variables download [NCEvents_A]
A: corresponds to a process number [0..6]

For external events the section name is extended with "X" in place of the process number.

Key values correspond to the variable numbers [0..31] and values are the NC events values.

```
[NCEvents_0]
000=0
001=1
...
031=0

[NCEvents_1]
000=1
...
031=0

[NCEvents_6]
000=1
...
031=1

[NCEvents_X]
000=1
...
031=1
```

Status of NC Events: NEV

MWCX device group

Designation **NEV** **N**C-**EV**ent

FI command Read the status of an NC event of the selected device of the MWCX device group.

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

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

(2) = Number of the NC event [0...31] ! Optional !

Note: If the optional parameter is not specified then the status of all NC events is output.

Response Structure One line is output, whereby the number of columns depends on the number of event statuses requested. When the optional parameter has not been specified, the response consists of one line with 32 columns. If the optional parameter has been specified then the answer consists of one line with one column which contains the status [0] or [1] of the requested NC event.

Example NEV Read the status of the 17th NC event in NC process 0 of device address 00.

FI command		**00_CR_NEV_0_17**
Line	**Column**	**Answer**
1	1	0

FI command Write the status of an NC event of the selected device of the MWCX device group.

CW_NEV_(1)_(2) **(Single Write)**

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

(2) = Event number [0...31]

Value to be written Status of NC Event 0 = delete NC event; 1 = set NC event

Response Structure One line is output with a column for acknowledgement of 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 NC process 0 at device address 00.

FI command		**00_CW_NEV_0_17** **Value to write: 1**
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature See chapter entitled “Literature” [7].

Selection of NC Memory: NMM

MWCX device group

Designation **NMM** **N**C-**M**e**M**ory

Explanation Used in selecting the NC memory for processing the NC program. The NC programs are managed on the NC in two NC memories. During the processing of an NC program, for instance in NC memory A, another NC program package can be transmitted into NC memory B. Both NC memories (A and B) are identically structured and completely equal; however, only one NC memory can ever be active at any given time.

FI command **CW_NMM** **(Single Write)**

Value to be written NC memory [1 = memory A; 2 = memory B]

Note: It is only possible to select an NC memory when the NC is ready for operation or is in the starting position. Otherwise, the request is acknowledged by an error message. The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge The selected NC memory has been selected.

Example NMM Select NC memory B at device 00 for processing the NC program.

FI command		**00_CW_NMM** **Value to write: 2**
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature See chapter entitled “Literature” [37].

Reading NC Parameters: NPA

MWCX device group

Designation **NPA** **N**C-**PA**rameter

FI command Read a parameter line.

BR_NPA1_(1)_(2) **(Single Read)**

(1) = Parameter record number [1..99]

(2) = Parameter number [A00.000..Cxx.120]

Response Structure The following table shows the general structure of the response to the FI command “NPA1”. One line is output with 3 columns for the identifier, the value and the name respectively.

Line 1	Column 1	Column 2	Column 3

Value Range/Meaning of Columns

1 = Identifier Parameter ID [max. 32 ASCII characters].

2 = Value [ASCII text]

3 = Name [unit, related to the value or empty]

Example NPA1

Return the parameter line from parameter record 10 with parameter number B00.007.

Assumption:
Parameter record 10 has been created and process 00 has been defined. In this place, the following information is located:
Max. path acceleration 75 mm/sec^2.

FI command		**00_BR_NPA1_10_B00.007**
Line	**Column**	**Answer**
1	1	B00.007
	2	75
	3	mm/sec^2

FI command

Read out several parameter lines from a parameter record.

BR_NPA2_(1)_(2)_(3) **(Single Read)**

(1) = Parameter record number [1..99]

(2) = Parameter number [from] [A00.000..Cxx.120]

(3) = Parameter number [to] [A00.000..Cxx.120]

Response Structure

The following table shows the general structure of the response to the FI command “NPA2”. As many lines as are requested are output, each with three columns for the identifier, the value and the name respectively.

Line 1...n:	**Column 1**	**...**	**Column 3**

Value Range/Meaning of Columns

1 = Identifier [max. 32 ASCII characters]

2 = Value [ASCII text]

3 = Name [unit, related to the value or empty]

Example NPA2

Return the parameter lines from parameter record 10 of parameter number A00.000 to parameter number A00.001.

Assumption:
Parameter record 10 has been created and contains the following information in this location:

FI command		**00_BR_NPA2_10_A00.000_A00.001**
Line	**Column**	**Answer**
1	1	A00.000
	2	Master
	3	
2	1	A00.001
	2	Process 1
	3	

FI command

Read a particular element of a parameter line.

Note: Command NPA3 is not supported in FI Version 06!

BR_NPA3_(1)_(2)_(3) **(Single Read)**

(1) = Parameter record number [1.99]

(2) = Parameter number [A00.000..Cxx.120]

(3) = Element number [1..1000]

Response Structure

The following table shows the general structure of the response to the FI command “NPA3”. One line is output with one column for either the name or value or designated name.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Name/value/designated name [ASCII-Text]

Example NPA3

Return element 1 of the parameter line from parameter record 10 with parameter number C01.079.

Assumption:
The parameter record has been created and contains the following information in this location:

FI command		00_BR_NPA3_10_C01.079_19
Line	**Column**	**Answer**
1	1	Required value (here 19) from existing compensation table of axis 1.

FI command

Read all elements from a parameter line (such as “NPA1”).

Note: Command "NPA4" is not supported in FI Version 06!

BR_NPA4_(1)_(2) **(Single Read)**

(1) = Parameter record number [1..99]

(2) = Parameter number [A00.000..Cxx.120]

Response Structure

The following table shows the general structure of the response to the FI command “NPA4”. One line is output with 3 columns for the identifier, the value and the name respectively.

Line 1	Column 1	Column 2	Column 3

Value Range/Meaning of Columns

1 = Identifier [max. 32 ASCII characters]

2 = Value [ASCII text]

3 = Name [unit, related to the value]

Example NPA4

Return the parameter line from parameter record 10 with parameter number A00.000.

Assumption:
The parameter record has been created and contains the following information in this location: Master.

FI command		00_BR_NPA4_10_A00.000
Line	**Column**	**Answer**
1	1	A00.000
	2	Master
	3	

Note: The commands supported in this version are listed using the command "00_NPA1_?".

Explanation

It is possible to read a list with a maximum of 10 parameters of the same type (system parameters, process parameters or axis parameters).

FI command

Read NC parameters for a selected device.

BR_NPA5_(1)_(2)_{(3)..(12)} **(Single Read)**

(1) = Parameter type	1 = System parameter 2 = Process parameter 3 = Axis parameter
(2) = Process number or axis number	If "system parameter" has been selected as the type of parameter, then this parameter is NOT evaluated – set to 0.
(3).....(12) = List of requested parameters	A maximum of 10 parameters of the same type may be listed here. Please take the parameter number from the general description of parameters for the control unit.

Response Structure

The following table shows the general structure of the response to the FI command "NPA5".

Line 1...n	Column 1	...	Column 3

Value Range/Meaning of Columns

1 =	Parameter number	Parameter number that has been requested.
2 =	Parameter value	Data setup – see general description of parameters.
3 =	Parameter unit	Data setup – see general description of parameters.

Example NPA5

NC parameter request for system parameters 0,52,53.

FI command		**00_BR_NPA5_1_0_0_52_53**
Line	**Column**	**Answer**
1	1	0
	2	Master
	3	--
2	1	52
	2	0
	3	--
3	1	53
	2	1
	3	--

Reference to Literature

See chapter entitled "Literature" [38].

Activate NC Compiler: NPC

MWCX device group

Designation **NPC** **N**C-**P**ackage **C**ompiling

FI command Compiles the selected NC package.

BR_NPC1_(1) **(Single Read)**

(1) = Number in NC package directory [1...99]

Response Structure The following table shows the general construction of the answer of the FI command NPC1. A line with three columns for job ID, FI command and the FI job ErrorCode is output.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Job ID [01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).

2 = FI command [string, in accordance to chapter entitled "Elements of the FI Command"]

3 = FI job error code (see chapter entitled “Error Codes”)

Example NPC Compile the 2nd NC package.

FI command	00_BR_NPC1_2		
	Answer		
Line	Column 1	Column 2	Column 3
1	01	00_BR_NPC1_2	0

Activate NC Download: NPD

MWCX device group

Designation **NPD** **N**C-**P**ackage **D**ownload

FI command Downloads the selected NC package into the identified device **without the setup lists**.

BW_NPD1_(1)_(2) **(Single Write)**

(1) = NC memory [1 = NC memory A, 2 = NC memory B]

(2) = Number in NC package directory [1...99]

Value to be written Initialization 1 = Trigger NC download

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure The answer of the FI command NPD1 consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID [01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).

Line 2 = FI command [string, in accordance to chapter entitled "Elements of the FI Command"]

Line 3 = FI job error code (see chapter entitled “Error Codes”)

Example NPA1

Load the 2nd NC package (**without setup lists)** into the NC memory A of the device with device address 00.

FI command		**00_BW_NPD1_1_2** **Value to be written: 1**
Line	**Column**	**Answer**
1	1	02
2	1	00_BW_NPD1_1_2
3	1	0

Note: If an attempt is made to transfer once again an NC package which is already in the device, the "DataTransfer" routine will terminate the process with error code 1030 (see chapter entitled "Error Codes").

FI command

Downloads the selected NC package into the identified device **with the setup lists**.

BW_NPD2_(1)_(2)	**(Single Write)**
(1) = NC memory	[1= NC memory A, 2= NC memory B]
(2) = Number in NC package directory	[1...99]

Value to be written

Initialization 1 = Trigger NC download

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The answer of the FI command NPD2 consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID	[01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).
Line 2 = FI command	[string, in accordance to chapter entitled "Elements of the FI Command"]
Line 3 = FI job error code	(see chapter entitled “Error Codes”)

Example NPD2

Load the 3rd NC package (**with setup lists)** into the NC memory B of the device with device address 00.

FI command		**00_BW_NPD2_2_3** **Value to be written: 1**
Line	**Column**	**Answer**
1	1	03
2	1	00_BW_NPD2_2_3
3	1	0

Note: If an attempt is made to transfer once again an NC package which is already in the device, the "DataTransfer" routine will terminate the process with error code 1030 (see chapter entitled "Error Codes").

Notes on NP3 and NP4

These FI command have been speed-optimized. They are suited especially well for the transmission of small NC programs (guide value: up to a maximum of 100 NC program lines). As the transmission of small

NC programs takes less than two seconds, a status query does not make much sense. Therefore, the function interface job administration was left out with these FI commands (see chapter entitled "FI Command for the MPCX Device Group", IFJ).

Note: The "DataTransfer" routine remains for all the transmission period (remain period = transmission period). This is only valid for these FI commands.

FI command

Downloads the selected NC package into the identified device **without the setup lists**.

BW_NPD3_(1)_(2)	**(Single Write)**
(1) = NC memory	[1 = NC memory A, 2 = NC memory B]
(2) = Number in NC package directory	[1...99]

Value to be written

Initialization 1 = Trigger NC download

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The answer of the FI command NPD3 consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID	[01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).
Line 2 = FI command	[string, in accordance to chapter entitled "Elements of the FI Command"]
Line 3 = FI job error code	(see chapter entitled “Error Codes”)

Example NPA3

Load the 2nd NC package (**without setup lists)** into the NC memory A of the device with device address 00.

FI command		**00_BW_NPD3_1_2** **Value to be written: 1**
Line	**Column**	**Answer**
1	1	03
2	1	00_BW_NPD3_1_2
3	1	0

Note: If an attempt is made to transfer once again an NC package which is already in the device, the "DataTransfer" routine will terminate the process with error code 1030 (see chapter entitled "Error Codes").

FI command

Downloads the selected NC package into the identified device **with the setup lists**.

BW_NPD4_(1)_(2)	**(Single Write)**
(1) = NC memory	[1 = NC memory A, 2 = NC memory B]
(2) = Number in NC package directory	[1...99]

Value to be written

Initialization 1 = Trigger NC download

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The answer of the FI command NPD4 consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID	[01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).
Line 2 = FI command	[string, in accordance to chapter entitled "Elements of the FI Command"]
Line 3 = FI job error code	(see chapter entitled “Error Codes”)

Example NPA4

Load the 3rd NC package (**with setup lists)** into the NC memory B of the device with device address 00.

FI command		**00_BW_NPD4_2_3** **Value to be written: 1**
Line	**Column**	**Answer**
1	1	03
2	1	00_BW_NPD4_2_3
3	1	0

Note: If an attempt is made to transfer once again an NC package which is already in the device, the "DataTransfer" routine will terminate the process with error code 1030 (see chapter entitled "Error Codes").

Read NC Package Directory: NPI

MWCX device group

Designation **NPI** **N**C-**P**ackage D**I**rectory

Explanation Reads the entries of the NC package directories.

FI command **BR_NPI** **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command NPI. The response consists of up to a maximum of n=99 lines, each with 5 columns.

Line 1...n:	**Column 1**	**...**	**Column 5**

Value Range/Meaning of Columns

1 =	Number in NC package directory	[01...99]
2 =	Name of the NC package	[max. 32 ASCII characters]
3 =	Length of the NC package	[byte]
4 =	Date of creation/last change to NC package	[DD.MM.YY]
5 =	Time of creation/last change to NC package	[HH:MM:SS]

Example NPI

Read the entries in the NC package directory at device address 00.

FI command		00_BR_NPI
Line	**Column**	**Answer**
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

Reference to Literature See chapter entitled “Literature” [31].

Selection of the NC Program in the Active NC Memory: NPS

MWCX device group

Designation **NPS** **N**C-**P**rogram **S**election

Explanation Used in selecting the NC program located for processing in the active NC memory. The NC programs are managed on the NC in two NC memories. During the processing of an NC program, for instance in NC memory A, another NC program package can be transmitted into NC memory B. Both NC memories (A and B) are identically structured and completely equal; however, only one NC memory can ever be active at any given time.

FI command **CW_NPS_(1)** **(Single Write)**

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

Value to be written Number in NC package directory [1...99]

Note: It is only possible to select an NC program when there is a valid NC program package in the active NC memory. Otherwise, the request is acknowledged by an error message. The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge The selected NC program has been selected.

Example NPS Select NC process number 0 for processing NC program 01 in the active NC memory.

Assumption:
There is a valid NC program package in the active NC memory.

FI command		00_CW_NPS_0 Value to write: 1
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature See chapter entitled “Literature” [37].

Next Tool Number: NTN

MWCX device group

Designation **NTN** **N**ext **T**ool-**N**umber

Explanation Returns the next pre-selected tool number of the selected device of the MWCX device group.

FI command
CR_NTN_(1) **(Single Read)**
CC_NTN_(1) **(Cyclic Read)**
CB_NTN_(1) **(Break Cyclic Read)**
(1) = NC process number [0...6]

Response Structure One line with two columns is output for the identifier [T= Tool] and for the next tool number.

Example NTN Read the next tool number in NC process 0 of device address 00.

FI command		00_CR_NTN_0
Line	**Column**	**Answer**
1	1	T
	2	1

Reference to Literature See chapter entitled “Literature” [43].

NC Zero Point Download: NUA

MWCX device group

Designation **NUA** **N**C-Offset Data **A**ccess

Explanation Zero points are downloaded by means of the download file via all active processes.

FI command Download NC zero points.
BW_NUA1_(1) **(Single Write)**
(1) = Download file with path details.

Note: Enclose file and path details in inverted commas.

Response Structure The response to the "NUA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NUA1 00_BW_NUA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		00_BW_NUA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_NUA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
3	1	0

Structure of the download file The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
This is currently only used for error processing, i.e., if an error is detected during a process, then the *DownloadError* key is written with "YES" within this section.

Example:

[Common]
DownloadError = YES ; error

Section [OffsetDataPackage_Info]
The package identification consists of several keys; the total length of all package identifications may not exceed a **maximum of 84** characters. The length of the individual identifications is described below:

Key **Memory**
Indicates the memory into which the NC package is loaded.
Memory=1 ;Memory A
Memory=2 ;Memory B

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters
Package size: "**PackageSize**" max. 8 characters left-justified
Package time: "**PackageTime**" max. 8 characters
Package date: "**PackageDate**" max. 8 characters
Package default:"**PackageDefault**" max. 26 characters (optional)

Total: max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

[OffsetDataPackage_Info]
Memory=
PackageNo = 1
PackageName = Offset Data
PackageSize = 1234
PackageTime = 13:10:10
PackageDate = 24.12.00

Section Zero-point data download

Consists of several pieces of information and is structured as follows:

[OffsetData_A\Number of zero-point bank\code of axis meaning]

A:	Process number [0..6]
Number of zero-point bank:	[0..9]
Code of axis meaning: [0..8]	
	[9] angle of rotation "PHI"

A section entry is an optional entry, i.e., if a section for a process is absent it is not regarded as an error.

Key values correspond to the types of offset [3..9] and values are the write values of the types of offset in the base unit. Missing key values are not regarded as errors.

Offset Type

Code	**Meaning**	**Explanation**
3	General	acts additive to all offset types
4	G54	adjustable zero offset
...		
9	G59	adjustable zero offset

Note: The axis meanings are contained in chapter 6.2, “Data Tables”.

```
[OffsetData_0\0\0]      ;Process 0, zero-point data bank 0, axis X
03=1.0000               ;Gen. offset
04=2.0000               ;G54
05=3.0000               ;G55
06=4.0000               ;G56
07=5.0000               ;G57
08=6.0000               ;G58
09=7.0000               ;G59

[OffsetData_0\2\3]      ;Process 0, zero-point data bank 3, axis X
03=1.0000               ;Gen. offset
04=2.0000               ;G54
05=3.0000               ;G55
06=4.0000               ;G56
07=5.0000               ;G57
08=6.0000               ;G58
09=7.0000               ;G59
```

NC Zero Point Upload: NUA

MWCX device group

Designation **NUA** **N**C-Offset Data **A**ccess

Explanation Zero-points are uploaded via all active processes.

FI command Zero-point upload.

BR_NUA1_(1)_(2) **(Single Read)**

(1) = memory [1 = memory A;
2 = memory B]

(2) = Upload file with path details

Note: Enclose file and path details in inverted commas.

In this command, the progress information is implemented in %. It can be interrogated via the command IFJ of the MPCX device group.

Response Structure The response to the NUA1 FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NVA 00_BR_NUA1_1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_NUA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_NUA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Section [Common]
General information is stored in the COMMON section.

Key **UploadError**
Indicates whether or not an error has occurred during uploading. This value is only set in the event of an error.

Example:

[Common]
UploadError = YES ; error

Section NC variables information [OffsetDataPackage_Info]
Key **Memory**
Identifies the memory from which the NC package is loaded.
Memory=1 ;Memory A
Memory=2 ;Memory B

Key **Program package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**"	max. 2 characters
Package name "**PackageName**"	max. 32 characters
Package size: "**PackageSize**"	max. 8 characters left-justified
Package time: "**PackageTime**"	max. 8 characters
Package date: "**PackageDate**"	max. 8 characters
Package default:"**PackageDefault**"	max. 26 characters (optional)
Total:	max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

```
[OffsetDataPackage_Info]
Memory=
PackageNo =         1
PackageName =       Offset Data
PackageSize =       1234
PackageTime =       13:10:10
PackageDate =       24.12.00
```

Section Zero-point data download
Consists of several pieces of information and is structured as follows:

[OffsetData_A\Number of zero-point bank\code of axis meaning]
A: Process number [0..6]
Number of zero-point bank: [0..9]
Code of axis meaning: [0..8]
[9] angle of rotation "PHI"

Key values correspond to the types of offset [3..9] and values are the read values of the types of offset in the base unit.

Offset Type

Code	Meaning	Explanation
3	General	acts additive to all offset types
4	G54	adjustable zero offset
...		
9	G59	adjustable zero offset

Note: The axis meanings are contained in chapter entitled “Data Tables”.

```
[OffsetData_0\0\0]   ;Process 0, zero-point data bank 0, axis X
03=1.0000            ;Gen. offset
04=2.0000            ;G54
05=3.0000            ;G55
06=4.0000            ;G56
07=5.0000            ;G57
08=6.0000            ;G58
09=7.0000            ;G59
```

```
[OffsetData_0\2\3]      ;Process 0, zero-point data bank 3, axis X
03=1.0000               ;Gen. offset
04=2.0000               ;G54
05=3.0000               ;G55
06=4.0000               ;G56
07=5.0000               ;G57
08=6.0000               ;G58
09=7.0000               ;G59
```

NC Variables Download: NVA

MWCX device group

Designation **NVA** **N**C-**V**ariable **A**ccess

Explanation NC variables are downloaded by means of the download file via all processes.

FI command Download NC variables.

BW_NVA1_(1) **(Single Write)**

(1) = Download file with path details.

Note: Enclose file and path details in inverted commas.

Response Structure The response to the "NVA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NVA1 00_BW_NVA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		**00_BW_NVA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_NVA1_"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
3	1	0

Structure of Download File The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Section [Common]
This is currently only used for error processing, i.e., if an error is detected during a process, then the *DownloadError* key is written with "YES" within this section.

Example:

```
[Common]
DownloadError = YES   ; error
```

Section [NCVariablesPackage_Info]
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number	"**PackageNo**"	max. 2 characters
Package name	"**PackageName**"	max. 32 characters
Package size:	"**PackageSize**"	max. 8 characters left-justified
Package time:	"**PackageTime**"	max. 8 characters
Package date:	"**PackageDate**"	max. 8 characters
Package default:	"**PackageDefault**"	max. 26 characters (optional)
Total:		max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

Example:

```
[NCVariablesPackage_Info]
PackageNo =         1
PackageName =       NC variables
PackageSize =       1234
PackageTime =       13:10:10
PackageDate =       24.12.00
```

Section NC variables download [NCVariables_A]
A: corresponds to a process number [0..6]

A section entry ([NCVariables_A]) is an optional entry, i.e., if a section for a process is absent it is not regarded as an error.

Key values correspond to the variable numbers [0..255] and values are the write values of the NC events. Missing key values are not regarded as errors.

```
[NCVariables_0]
000=1
001=3.14
...
255=255

[NCVariables_1]
000=1
...
100=255

[NCVariables_6]
000=1
010=3.14
255=255
```

NC Variables Download: NVA

MWCX device group

Designation **NVA** **N**C-**V**ariable **A**ccess

Explanation NC variables are uploaded via all processes.

FI command NC variables upload.

BR_NVA1_(1) **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.

In this command, the progress information is implemented in %. It can be interrogated via the command IFJ of the MPCX device group.

Response Structure The response to the NVA1 FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example NVA 00_BR_NVA1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		00_BR_NVA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_NVA1_"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Section [Common]
General information is stored in the COMMON section.

Key **UploadError**
Indicates whether or not an error has occurred during uploading. This value is only set in the event of an error.

Example:

[Common]
UploadError = YES ; error

NC variables information section [NCVariablesPackage_Info]

Key **Program package information**
The package identification is compiled from several keys. The total length of all package identifications must not exceed **a maximum of 84** characters. The length of the individual identifications is described below:

Package number"**PackageNo**" max. 2 characters
Package name "**PackageName**" max. 32 characters

Package size:	"**PackageSize**"	max. 8 characters left-justified
Package time:	"**PackageTime**"	max. 8 characters
Package date:	"**PackageDate**"	max. 8 characters
Package default:	"**PackageDefault**"	max. 26 characters (optional)
Total:		max. 84 characters

Information on date and time is given in the format
Date : dd.mm.yy
Time: hh:mm:ss

<u>Example:</u>

```
[NCVariablesPackage_Info]
PackageNo =         1
PackageName =       NC variables
PackageSize =       1234
PackageTime =       13:10:10
PackageDate =       24.12.00
```

Section NC variables download [NCVariables_A]

A: corresponds to a process number [0..6]

Key values correspond to the variable numbers [0..255] and values are the NC variables values.

```
[NCVariables_0]
000=1
001=3.14
...
255=255

[NCVariables_1]
000=1
...
100=255

[NCVariables_6]
000=1
010=3.14
255=255
```

Reading and Writing NC Variables: NVS

MWCX device group

Designation **NVS** **N**C-**V**ariable **S**ingle

Explanation Reads the NC variables of the selected device of the MWCX device group.

FI command

CR_NVS_(1)_(2){_(3)}	**(Single Read)**
CC_NVS_(1)_(2){_(3)}	**(Cyclic Read)**
CB_NVS_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = NC 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 then up to 20 NC variables are output.

Response Structure One line with a maximum of 20 columns containing the corresponding value of the requested NC variable is output.

Note: If the requested NC variable does not exist then [--] is entered in the corresponding column.

Example NVS without optional Parameter Read the value of the NC variable numbered 1 at device address 00 in NC process 0.

FI command		**00_CR_NVS_0_1**
Line	**Column**	**Answer**
1	1	1.111000

Example NVS with optional Parameter Read the value of the 1st NC variable to the 3rd NC variable at device address 00 in NC process 0.

Assumption:
The 2nd NC variable is not defined.

FI command	**00_CR_NVS_0_1_3**		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	1.111000	--	23.100000

Explanation Writes an NC variable of the selected device of the MWCX device group.

FI command

CW_NVS_(1)_(2)	**(Single Write)**
(1) = NC process number	[0...6]
(2) = NC variable number	[0...255]

Value to be written NC variable [Format, long, or doublereal]

Note: The of the NC variables is set to long or doublereal in accordance with the entered format. With doublereal, the decimal point '.' must be used by all means.

Note: Only defined NC variables can be written. The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge variable has been written.

Example NVS

Write the value 1.111000 in the 1st NC variable in NC process 0 at device address 00.

FI command		00_CW_NVS_0_1 Value to be written: 1.111000
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature

See chapter entitled “Literature” [39].

Optimum Position Distance from Axes: POD

MWCX device group

Designation

OPD **O**ptimal **P**osition **D**istance

Explanation

The optimum position distance of a selected axis of the MWCX device group is read out. The FI command "OPD1" returns the position distance of an axis, related to the code of the axis meaning. On the other hand, the FI command "OPD2" returns the position distance of an axis, related to the physical axis number.

FI command

Output of the optimum position distance of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_OPD1_(1)_(2){_(3)} **(Single Read)**

CC_OPD1_(1)_(2){_(3)} **(Cyclic Read)**

CB_OPD1_(1)_(2){_(3)} **(Break Cyclic Read)**

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

(2) = Axis meaning [0...11; 20] (see Chapter 6.2, “Data Tables”)

(3) = Required measurement system (opt.) [mm, inch]

FI command

Output the optimum position distance of the selected axis of the device specified, related to the physical axis number.

Using the optional second parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_OPD2_(1){_(2)} **(Single Read)**

CC_OPD2_(1){_(2)} **(Cyclic Read)**

CB_OPD2_(1){_(2)} **(Break Cyclic Read)**

(1) = Physical axis number [1...32]

(2) = Required measurement system (opt.) [mm, inch]

Response Structure

The following table shows the general structure of the response to the FI commands "OPD1" and "OPD2". One line with four columns is output for the name of the axis, value of the optimum position distance, the unit and the opt. position distance limited to “indicated decimal places”.

Line 1	Column 1	Column 2	Column 3	Column 4

Value Range/Meaning of the Columns

1 = Axis name	[Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi, Ci, Si] with i = [,1,2,3]
2 = Optimum position distance	[acc. to settings in the process parameters]
3 = Unit	[mm, inch]
4 = Optimum position distance	[as Column 2, but rounded up or down according to the parameter “Indicated decimal places”]

Note: If the specified axis is not defined in the selected NC process then the response in all columns is [--].

Example OPD1

Read the optimum position distance of the Z axis in NC process 0 of device address 00.

FI command	00_CR_OPD1_0_2			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z	-5.9897	[mm]	-5.990

Example OPD1

Read the optimum position distance of the Z axis in NC process 0 of device address 00. Values are displayed in inches:

FI command	00_CR_OPD1_0_2			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z	-0.2358	[inch]	-0.236

Example OPD2

Read the optimum position distance of the Z axis (physical axis number = 3) at device address 00.

FI command	00_CR_OPD2_3			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z	-5.9897	[mm]	-5.990

Parameter Download: PAA

MWCX device group

Designation **PAA** **PA**rameter **A**ccess

Explanation Complete parameter records are downloaded by means of a download file.

FI command Parameter download command whereby two predefined functions are to be programmed by the user. These two functions concern:

1. Function for creating the download file itself:

```
LONG ParameterDownloadBegin(  Long lProjectNumber,
                              Long lDeviceNumber,
                              Long lIndexNumber,
                              Char* pcPDLFileName,
                              Long lMaxLengthFileNameBuffer,
                              Char* pcErrorText,
                              Long lMaxLengthErrorTextBuffer)
```

Pass parameters:

- lProjectNumber: Currently selected project number
- lDeviceNumber: Currently selected device address
- lIndexNumber: Currently selected parameter directory number [1..99]
- pcPDLFileName: Contains the complete file name for the created parameter download file.
- lMaxLengthFileNameBuffer: max. length of the buffer for the name of the parameter download file.
- pcErrorText: If necessary, user error text
- lMaxLengthErrorTextBuffer: Max. length of the buffer for the user error text.

2. Function called up at the end of the parameter download:

```
Long ParameterDownloadEnd(  Char* pcPDLFileName,
                            Long lResult)
```

Pass parameters:

- pcPDLFileName:
  Contains the complete file names for the created parameter download file.
- lResult:
  Contains the status message of the parameter download procedure
  Here, 0 = Parameter download procedure O.K.
  > 0 = Error has occurred

The two functions must be programmed in a DLL by the user and also exported from it.

BW_PAA1_(1)_{(2)} **(Single Write)**

(1) = Parameter directory number; the two functions to be implemented are located in INDIF410.DLL.

(2) = Complete DLL name, if required, in which the two functions to be implemented are located.

Response Structure

The response to the "PAA1" FI command consists of three lines, each with one column.
The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
 (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
 [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
 (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example PAA1

00_BW_PAA1_2_"D:\UserDir\USER.DLL"

FI command		00_BW_PAA1_2_D:\UserDir\USER.DLL
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PAA1_2_D:\UserDir\USER.DLL
3	1	0

FI command

Parameter download command whereby the parameter download file is directly indicated.

BW_PAA2_(1) **(Single Write)**

(1) = Complete parameter download file name

Response Structure

The response to the "PAA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
 (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
 [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
 (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example PAA2

00_BW_PAA2_"D:\DOWNLOAD.DAT"

FI command		00_BW_PAA2_"D:\DOWNLOAD.DAT"
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PAA2_"D:\DOWNLOAD.DAT"
3	1	0

Structure of Download File

The structure of the download file corresponds to that of a Windows Ini file. Indramat's own description in V20_Param_08_Definitions_Parameter_Download_01.doc is recommended for a more detailed account of the structure of the download file.

Summary:

Section [ID_PARAMETER]
Information concerning parameter identification.

Section [ID_SYSTEM]
Information concerning system parameter identification.

Section [DATA_SYSTEM]
Listing of system parameter data.

Section [ID_PROCESSX]
Information concerning process parameter identification.

Section [DATA_PROCESSX]
Listing of process parameter data.

Section [ID_AXISX]
Information concerning axis parameter identification.

Section [DATA_AXISX]
Listing of axis parameter data.

Parameter Upload: PAA

MWCX device group

Designation **PAA** **PA**rameter **A**ccess

Explanation Uploads complete parameter records from a selected device. The data read is written into an upload file with an identical structure to that of a download file.

FI command Parameter upload command whereby two predefined functions are to be programmed by the user. These two functions concern:

1. The function supplies the complete name of the upload file:

```
LONG ParameterUploadBegin(   Long lProjectNumber,
                             Long lDeviceNumber,
                             Char* pcUploadFileName,
                             Long lMaxLengthFileNameBuffer,
                             Char* pcErrorText,
                             Long lMaxLengthErrorTextBuffer)
```

Pass parameters:

- lProjectNumber:
  Currently selected project number
- lDeviceNumber:
  Currently selected device address
- pcUploadFileName:
  Contains the complete file name for the parameter upload file to be created.
- lMaxLengthFileNameBuffer:
  Max. length of the buffer for the name of the parameter upload file.
- pcErrorText:
  If necessary, user error text
- lMaxLengthErrorTextBuffer:
  max. length of the buffer for the user error text

2. Function called up at the end of the parameter upload:

```
LONG ParameterUploadEnd(   Char* pcUploadFileName,
                           Long lResult)
```

Pass parameters:

- pcUploadFileName:
  Contains the complete file names for the created parameter upload file.
- lResult:
  Contains the status message of the parameter upload procedure
  Here: 0 = Parameter upload procedure O.K.
  > 0 = Error has occurred

The two functions must be programmed in a DLL by the user and also exported from it.

BR_PAA1_(1)_{(2)} **(Single Read)**

(1) = Parameter directory number; the two functions to be implemented are located in INDIF410.DLL.

(2) = Complete DLL name, if required, in which the two functions to be implemented are located.

Response Structure

The response to the "PAA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example PAA1

00_BR_PAA1_2_"D:\UserDir\USER.DLL"

FI command		**00_BR_PAA1_2_D:\UserDir\USER.DLL**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_PAA1_2_D:\UserDir\USER.DLL
3	1	0

FI command

Parameter upload command whereby the parameter upload file is directly indicated.

BR_PAA2_(1) **(Single Read)**

(1) = complete name of the parameter upload file

Response Structure

The response to the "PAA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example PAA2 00_BR_PAA2_"D:\UPLOAD.DAT"

FI command		**00_BR_PAA2_"D:\UPLOAD.DAT"**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_PAA2_"D:\UPLOAD.DAT"
3	1	0

Structure of Upload File The structure of the upload file corresponds to that of a Windows Ini file. Rexroth Indramat's own description in V20_Param_08_Definitions_Parameter_Download_01.doc is recommended for a more detailed account of the structure of the upload file.

For a summary refer to the description under Parameter Download Command.

Process Axis Configuration Data: PAC

MWCX device group

Designation **PAC** **P**rocess **A**xis **C**onfiguration Parameter

Explanation The axis configuration data of a process is returned.

FI command Output the axis configuration parameters of all NC processes.

BR_PAC1 **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command "PAC1". The number of lines depends on the number of defined CN processes. Each line consists of five columns for the NC process number, the physical axis number, the main axis meaning, the main axis name and the axis type.

Line 1...n:	**Column 1**	**....**	**Column 5**

Value Range/Meaning of Columns

(1) = NC process number	[0...6]
2 = Physical axis number	[1...32]
3 = Main axis meaning	[see Chapter 6.2 "Data Tables"]
4 = Main axis name	[Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])
5 = Axis type	[see Chapter 6.2 "Data Tables"]

Example PAC1 Read all process axis configuration data of device address 00.

FI command		**00_BR_PAC1**
Line	**Column**	**Answer**
1	1	0
	2	1
	3	0
	4	X1
	5	0x81
2	1	1
	2	2
	3	1
	4	Y1

FI command		00_BR_PAC1
Line	**Column**	**Answer**
	5	0x82
3	1	2
	2	3
	3	5
	4	--
	5	--

FI command

Output the axis configuration data of an NC process.

BR_PAC2_(1) **(Single Read)**

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

Response Structure

The following table shows the general structure of the response to the FI command “PAC2”. One line is output with five columns for the NC process number, the physical axis number, the main axis meaning, the main axis name and the axis type.

Line 1	Column 1	....	Column 5

Value Range/Meaning of Columns

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

2 = Physical axis number [1...32]

3 = Main axis meaning [see the chapter entitled “Data Tables”]

4 = Main axis name [Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] (i=[ ], [1...3])

5 = Axis type [see Chapter 6.2 “Data Tables”]

Example PAC2

Read the axis configuration data of process 0 at device address 00.

FI command		00_BR_PAC2_0
Line	**Column**	**Answer**
1	1	0
	2	1
	3	0
	4	X1
	5	0x81

Deactivate Parameters for an Offline Device PAD

MWCX device group

Designation

PAD **PA**rameter **D**eactivate

Explanation

If a device is in offline mode (DeviceStatus=OFF), this FI command deactivates the parameter record in the offline device; then, NO valid parameter record is present.

FI command

BW_PAD1 **(Single Write)**

Response Structure

The response to the "PAD1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example PAS1

The parameter records are deactivated for the offline device 00, i.e., there is NO valid parameter record in the device 00.

FI command		**00_BW_PAD1**
Line	**Column**	**Answer**
1	1	(P_ACK)

Setting Parameters Active for an Offline Device: PAS

MWCX device group

Designation

PAS **PA**rameter **S**et Active

Explanation

If a device is in offline mode (DeviceStatus=OFF), this FI command sets a parameter record active.

FI command

BW_PAS1_(1) **(Single Write)**

(1) = Complete parameter download file name

Response Structure

The response to the "PAS1" FI command consists of three lines, each with one column.
The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

Note: File names must be enclosed in inverted commas. This command is an FI job command.

Example PAS1

For the offline device 00, the parameter data of the parameter download file D:\DOWNLOAD.DAT are set active.

FI command		**00_BW_PAS1_"D:\DOWNLOAD.DAT"**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PAS1_"D:\DOWNLOAD.DAT"
3	1	0

ProVi Diagnosis Data: PDD

MWCX device group

Designation

PDD **P**rovi **D**iagnosis **D**ata

Explanation

Data for ProVi criteria analysis is output.

FI command

Output of files to indicate the detail in the editor.

BR_PDD1_(1)_(2){_(3)} **(Single Read)**

(1) = Message ID [ASCII characters]

(2) = Message type [1 = errors, 2 = messages, 10 = warnings, 11 = start requirements, 12=setup diagnosis]

(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD1 FI command.

Line 1	Column 1	...	Column 5

Meaning of the Columns

1 = POU ID [ASCII characters]
2 = Detail morpheme [ASCII characters] (DWORD, decimal)
3 = Error ID [ASCII characters] (DWORD, decimal)
4 = POU entity name [ASCII characters]
5 = Nw ID (network ID) [ASCII characters]

Example PDD1

Indication of data of a ProVi error with ID 43923028 from module 3 in control unit 0.

FI command		00_BR_PDD1_43923028_1_1
Line	**Column**	**Answer**
1	1	STATION_1_2
	2	98243823
	3	34985304
	4	Station2.Module3
	5	43493454

FI command

Output the I/O addresses to display a detail.

BR_PDD2_(1)_(2){_(3)} **(Single Read)**

(1) = Message ID [ASCII characters]
(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]
(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD2 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Variable morpheme [ASCII characters] (DWORD, decimal)
2 = I/O address [ASCII characters]

Example PDD2

Query of the I/O addresses of a ProVi error with ID 43923028 from module 3 in control unit 0.

Three variables have an I/O address.

FI command		00_BR_PDD2_43923028_1_1
Line	**Column**	**Answer**
1	1	98243823
	2	%I3.2.0
2	1	40923423
	2	%Q23.21.7
3	1	34985304
	2	%I100.3.5

FI command

Determine the multilingual comments for displaying a detail.

BR_PDD3_(1)_(2){_(3)}	**(Single Read)**
(1) = Message ID	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(3) = Module number	[1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD3 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Comment morpheme [ASCII characters] (DWORD, decimal)

2 = New comment [ASCII characters]

Example PDD3

Query of the comments for indication of a ProVi error with ID 43923028 from module 3 in control unit 0.

Two comments are replaced by another text.

FI command		00_BR_PDD3_43923028_1_1
Line	**Column**	**Answer**
1	1	98243823
	2	Clamp open
2	1	40923423
	2	Clamp closed

FI command

Query of the status of a certain message

BR_PDD4_(1)_(2){_(3)}	**(Single Read)**
(1) = Message number	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(3) = Module number	[1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD4 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Message is present [YES, NO]

2 = Criteria analysis exists [YES, NO]

Example PDD4

Query of the status of a ProVi error, number 1001 from module 3 in control 0.

This message is not present at the moment, and there is a criteria analysis.

FI command		00_BR_PDD4_1001_1_1
Line	**Column**	**Answer**
1	1	NO
	2	YES

FI command

Determination of the MessageID of a certain message

BR_PDD5!(1)!(2)!(3)!(4){!(5)} **(Single Read)**

(1) = POU entity name [ASCII characters]
(2) = Nw ID [ASCII characters]
(3) = Message number [ASCII characters]
(4) = Message type [1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(5) = Module number [1...99] ! only for message type 1 -2!

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the PDD5 FI command.

Line 1-n	Column 1	...	Column 3

Meaning of the Columns

1 = Message ID [ASCII characters] (DWORD, decimal)

2 = Message is present [YES, NO]

3 = Criteria analysis exists [YES, NO]

Example PDD5

Determination of the MessageID of a ProVi error, number 1001 from module 25.40 mm control 0.

Assumption:
This message is not present at the moment, and there is a criteria analysis.

FI command		**00_BR_PDD5!Station2.Modul3!43493454!1001!1!1**
Line	**Column**	**Answer**
1	1	240872342
	2	NO
	3	YES

Reading the Parameter Definition Table: PDT

MWCX device group

Designation

PDT **P**arameter **D**efinition **T**able

Explanation

The parameter definition table for the selected device can be read. Note: This command ONLY returns binary data, which means that knowledge of the structure of the parameter definition table is necessary in order to interpret this binary data!

FI command

Read parameter definition table for the selected device.

BR_PDT **(Single Read)**

Response Structure

The following table shows the general structure of the response to the FI command "PDT".

Line 1	Column 1

Value Range/Meaning of Columns

1 = Parameter definition table in binary form — Binary encoding of the parameter definition table in accordance with conventional control

Example PDT

Read the parameter definition table for device 0.

FI command		00_BR_PDT1
Line	Column	Answer
1	1	Binary data for the parameter definition table

Programmed Feed Velocity: PFR

MWCX device group

Designation PFR **P**rogrammed **F**eed **R**ate

Explanation The value of the programmed feedrate of the selected device of the MWCX device group is read out.

FI command Output the current value of the programmed feedrate of an NC process.

Using the optional second parameter it is possible to pre-select conversion of the result into mm or inches.

FI command

CR_PFR_(1){_(2)}	**(Single Read)**
CC_PFR_(1){_(2)}	**(Cyclic Read)**
CB_PFR_(1){_(2)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Required measurement system (opt.)	[mm, inch]

Response Structure The following table shows the general structure of the response to the FI command "PFR". One line is output with three columns for the identifier, the current value of the programmed feedrate and the unit.

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier	[F = feedrate]
2 = Feedrate	[format according to settings of the parameters]
3 = Unit	[according to settings of the parameters]

Example PFR Read the programmed feedrate in NC process 0 of device address 00.

FI command	00_CR_PFR_0		
Answer			
Line	Column 1	Column 2	Column 3
1	F	30000.0	[mm/min]

Example PFR Read the programmed feedrate in NC process 0 of device address 00. The displayed value is to be converted into inch/min:

FI command	00_CR_PFR_0		
Answer			
Line	Column 1	Column 2	Column 3
1	F	1181.1	[inch/min]

Generating Physical Directory Names: PHD

MWCX device group

Designation **PHD** **PH**ysical **D**irectory

Explanation Generates physical directory names according to the BDI data written.

Note: This is based on BDI philosophy.

FI command Generate physical directory names.

BR_PHD1_(1)_(2)_(3)_(4)_(5)_(6) **(Single Write)**

(1) = Project ID [-1= PROJECT_NEUTRAL
-2= PROJECT_DEFAULT]

(2) = Section ID [0= SECT_NEUTRAL
1= SECT_BIN
2= SECT_BASIC_DATA
3=SECT_OEM_DATA
4=SECT_CUSTOM_DATA
5=SECT_PROG_DATA]

(3) = Device address [-1= DEVADDR_NEUTRAL
otherwise the required
device address]

(4) = Process ID [-1= PROCESS_NEUTRAL
otherwise the required
process number]

(5) = Data type ID [possible write values see
BDI documentation
(BDI_DEFINITIONS.H)]

(6) = Language ID [possible write values see
BDI documentation (WINNT.H)]

Response Structure The following table shows the general structure of the response to the FI command "PHD1".

Line 1	Column 1

Value Range/Meaning of Columns 1 = Physical directory name [complete physical directory name in accordance with the BDI data written]

Example PHD1 Requesting the physical directory name for:

PROJECT_NEUTRAL
SECT_BIN
DEVADDR_NEUTRAL
PROCESS_NEUTRAL
DATATYPE_NEUTRAL
LANG_NEUTRAL

FI command		**XX_BR_PHD1_-1_0_-1_-1_0_0**
Line	**Column**	**Answer**
1	1	D:\Programme\Indramat\Mtgui\Bin

Active NC Program Information: PPA

MWCX device group

Designation **PPA** **P**art **P**rogram **A**ctive

Explanation Reads the active NC program with information about the NC memory and NC program number.

FI command

BR_PPA_(1)	**(Single Read)**
BC_PPA_(1)	**(Cyclic Read)**
(1) = Process number	[0...6]

Response Structure The following table shows the general structure of the response to the FI command "PPA". One line is output with 3 columns for the NC memory, NC program number and NC program name.

Line 1	Column 1	Column 2	Column 3

Value Range/Meaning of Columns

1 = NC memory	[A = memory A, B = memory B]
2 = NC program number	[01...99]
3 = NC program name	[max. 32 ASCII characters]

Example PPA Read in NC process 0 at device address 00.

Assumption:
The NC program numbered 01 and the with the name “Block4” is located in NC memory A; the memory is currently active.

FI command		00_BR_PPA_0
Line	**Column**	**Answer**
1	1	A
	2	01
	3	Block4

Reference to Literature See chapter entitled “Literature” [37].

Read NC Program Directory: PPD

MWCX device group

Designation **PPD** **P**art-**P**rogram **D**irectory

FI command Reads the entries of the NC program directory.

BR_PPD_(1)_(2)	**(Single Read)**
(1) = Number in NC package directory	[1...99]
(2) = NC process number	[0...6]

Response Structure The following table shows the general structure of the response to the FI command "PPD". The response consists of up to a maximum of n=99 lines, each with 5 columns.

Line 1...n:	Column 1	...	Column 5

Value Range/Meaning of Columns

1 = NC program number	[00...99]
2 = Program designation	[max. 32 ASCII characters]
3 = Program length	[byte]
4 = Date of creation/last change of	[DD.MM.YY]

program

5 = Time of creation/last change of program [HH:MM:SS]

Example PPD Read the entries in the NC program directory of the NC package number 1 of the NC process 0 at device address 00.

FI command		**00_BR_PPD_1_0**
Line	**Column**	**Answer**
1	1	01
	2	TransAM
	3	3579
	4	16.05.99
	5	10:41:08
2	1	10
	2	BMW 3 series
	3	4589
	4	18.05.99
	5	10:12:10

Expert or Import NC Program: PPN

MWCX device group

Designation **PPN** **P**art-**P**rogram **N**C

FI command Transfers an NC program from the NC program directory into an ASCII file (export).

BR_PPN_(1)_(2)_(3)_(4) **(Single Read)**

(1) = Number in NC package directory [1...99]

(2) = NC process number [0...6]

(3) = Number of the NC program [1...99]

(4) = NC block numbering [0 = without number; 1 = with numbers] !

Response Structure The response of the FI command PPN consists of one line and one column for information on the drive, the directory, and the file which contains the NC program.

Example PPN Without any NC block numbering, import the NC program with the NC program number 1 of the 2nd NC package of the NC process 0 at the device address 00 into a file.

FI command		**00_BR_PPN_2_0_1**
Line	**Column**	**Answer**
1	1	C:\MT-CNC\ANLAGE01\MT_TEMP\T1010001.TMP

Excerpt 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 [infeed motion]
G01 X60 Y0 F2000
X5 Y0
Z100
M05 [spindle stop]

T0 BSR .M6
BST .START
PROGRAM END
```

FI command

Transfers an NC program from an ASCII file into the NC program directory (import).

BW_PPN_(1)_(2)_(3)_(4)_(5)_(6)		**(Single Write)**
(1) =	Number in NC package directory	[1...99]
(2) =	NC process number	[0...6]
(3) =	Number of the NC program	[1...99]
(4) =	NC block numbering	[0 = without number; 1 = with numbers] !
(5) =	Is the NC package directory entry empty?	[0 = without check (preset); 1 = with check] ! Optional !
(6) =	Complete information on the directory	[DRIVE:\..\X.Y]

Note: This FI command does not have any "Value to be written".

Response Structure

One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge NC programs was exported.

Example PPN

From the file "C:\Data\T1010001.TMP", export the NC program in NC program number 1 of the 2nd NC package of the NC process 0 at the device address 0.

FI command		**00_BW_PPN_2_0_1_0_1_C:\Data\T1010001.TMP**
Line	**Column**	**Answer**
1	1	(P_ACK)

Renaming of an NC Part Program: PPN

MWCX device group

Designation **PPN** **P**art **P**rogram **N**C

Explanation This FI command renames an NC part program.

FI command **BA_PPN_(1)_(2)_(3)** **(Single Write)**

(1) = Number in NC package directory [1...99]

(2) = NC process number [0...6]

(3) = Number of the NC program [1...99]

Value to be written Name of the NC part program [max. 32 ASCII characters]

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(BOF_FCT_OK) = **BOF_F**un**CT**ion**_OK** program package has been renamed.

Example PPP The name of the NC part program numbered 1 in the NC package directory is to be renamed "PART1".

FI command		**00_BA_PPN_2_0_1** **Value to be written: PART1**
Line	**Column**	**Answer**
1	1	(BOF_FCT_OK)

Reference to Literature See chapter entitled “Literature” [37].

Renaming of an NC Program Package: PPP

MWCX device group

Designation **PPP** **P**art **P**rogram **P**ackage

Explanation The name of an NC program package of the selected device of the MWCX device group is changed.

FI command **BA_PPP_(1)** **(Single Alternate)**

(1) = NC program package [1...99]

Value to be written Name of the NC program package [max. 32 ASCII characters]

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(BOF_FCT_OK) = **BOF_F**un**CT**ion**_OK** program package has been renamed.

Example PPP The name of the NC program package numbered 1 in the NC package directory is to be renamed "FORM1".

FI command		00_BA_PPP_1 Value to be written: FORM1
Line	**Column**	**Answer**
1	1	(BOF_FCT_OK)

Reference to Literature See chapter entitled “Literature” [37].

Reading an NC Record: PPS

MWCX device group

Designation **PPS** **P**art **P**rogram **S**equence

Explanation An NC record of an NC program from the selected device of the MWCX device group is read out.

FI command **CR_PPS_(1)_(2)_(3)_(4)** **(Single Read)**

(1) = NC memory [1=memory A, 2=memory B]

(2) = NC process number [0...6]

(3) = NC program number [0...99]

(4) = NC record number [0...9999]

Response Structure One line with one column containing the requested NC record is output.

Example PPS Read NC record number 2 from NC program memory A, NC process number 0 or NC program number 1.

FI command		00_CR_PPS_1_0_1_2
Line	**Column**	**Answer**
1	1	N0002 G01 X50.0000 Y50.0000 Z20.0000 F2500.0

Reference to Literature See chapter entitled “Literature” [4].

Issuing SYS Messages Specific to the PCL: PSM

MWCX device group

Designation **PSM** **P**CL **S**ys **M**essage

Explanation Issues the most important SYS messages regarding the PCL programming interface – required for remote programming.

Note:
The appropriate device address is passed as the write value.

It allows the following SYS messages to be initiated:

- Start of PCL download,
- end of PCL download,
- start of PLC online edit,
- end PLC online edit,
- start of PCL declaration change, and
- end of PCL declaration change.

FI command

Issue the most important PCL SYS messages.

BW_PSM1_(1) **(Single Write)**

(1) = Requested SYS message [1= start of PCL download
2= end of PCL download
3= start of PCL online edit
4= end of PCL online edit
5= start of PCL declaration change
6= end of PCL declaration change]

?Value to be written: Device address

Response Structure

The following table shows the general structure of the response to the FI command "PSM1".

Line 1	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Status report	[READY=SYS message has been correctly acknowledged by the WIN32 applications] [ERROR=SYS message has NOT been acknowledged by a WIN32 application within the pre-set time]
2 =	Task name (LogInIf name)	[Task name that has triggered the SYS message]
3 =	SYS message number	[contains the issued SYS message number]
4 =	Acknowledgement time	[contains the pre-set acknowledgement time]
5 =	Reference information	[contains, where applicable, the reference information transferred as a write value]
6 =	Length of reference information	[0 where NO reference information has been transferred]
7 =	Where applicable, LOG channel of the FI that has NOT acknowledged	[-- = acknowledgements have been completed in time or the LOG channel number of the WIN32 application that has NOT acknowledged in time]
8 =	Where applicable, task name that has NOT acknowledged in time.	[-- = acknowledgements have been completed in time or the task name that has NOT acknowledged in time]

Example PSM1

Issue the SYS message Beginning PCL Download. The reference information, device address 00, is also transferred as a write value.

FI command		**XX_BW_PSM1_1 – value to be written: 00**
Line	**Column**	**Answer**
1	1	READY
	2	WINPCL.EXE
	3	14
	4	30000
	5	00
	6	2
	7	--
	8	--

Programmed Spindle Speed: PSS

MWCX device group

Designation **PSS** **P**rogrammed **S**pindle **S**peed

Explanation The value of the programmed spindle speed of the selected device of the MWCX device group is read out.

FI command

CR_PSS_(1)_(2) **(Single Read)**

CC_PSS_(1)_(2) **(Cyclic Read)**

CB_PSS_(1)_(2) **(Break Cyclic Read)**

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

(2) = Number of spindle [1...3]

Response Structure The following table shows the general structure of the response to the FI command "PSS". One line with three columns is output for the axis name, the speed and the unit [1/min].

Line 1	Column 1	....	Column 3

Value Range/Meaning of Columns

1 = Identifier [S = spindle]

2 = Speed [format according to settings of the parameters]

3 = Unit 1/min

Example PSS Read the speed of the 1st spindle in NC process 0 of device address 00.

FI command	00_CR_PSS_0_1		
Answer			
Line	Column 1	Column 2	Column 3
1	S:	7500.0	1/min

Reference to Literature See chapter entitled “Literature” [4].

Process Tool Management Configuration: PTC

MWCX device group

Designation **PTC** **P**rocess **T**ool Management **C**onfiguration

Explanation Returns the most significant process parameter data of the tool magazine of the selected device of the MWCX device group.

FI command Read tool management data of all defined NC processes.

BR_PTC1 **(Single Read)**

BC_PTC1 **(Cyclic Read)**

Response Structure The following table shows the general structure of the response to the FI command “PTC1”. The number of lines depends on the number of defined CN processes. Each line consists of 9 columns for the returned values.

Line 1...n:	Column 1	...	Column 9

Value Range/Meaning of the Columns

1 = NC process number [0...6]

2 = Process name

3 = Tool management [YES, NO]

4 = Tool memory [[MAGAZINE], [TURRET]]

5 = Endlessly turning tool memory [YES, NO]

6 = Number of tool memory locations　[0...999]
7 = Number of tool spindles　[0...4]
8 = Number of tool grabbers　[0...4]
9 = Axis number of tool axis　[0...20]

Note: If there is no tool management (Column 3: NO), then all partial results from Column 4 are marked as [--].

Example PTC1

Returns the process parameter data of the defined processes. This example assumes that there are two processes, On process with and another one without tool management.

FI command		**00_BR_PTC1**
Line	**Column**	**Answer**
1	1	0
	2	MILLING
	3	YES
	4	[MAGAZINE]
	5	YES
	6	8
	7	1
	8	2
	9	4
2	1	1
	2	TRANSFER
	3	NO
	4	--
	5	--
	6	--
	7	--
	8	--
	9	--

FI command

Read tool management data of an NC process.

BR_PTC2_(1)　**(Single Read)**

BC_PTC2_(1)　**(Cyclic Read)**

(1) = NC process number　[0...6]

Response Structure

The following table shows the general structure of the response to the FI command “PTC2”. One line with 9 columns is output for the returned values.

Line 1	**Column 1**	**...**	**Column 9**

Meaning of the Columns

1 = NC process number　[0...6]
2 = Process name
3 = Tool management　[YES, NO]
4 = Tool memory　[[MAGAZINE], [TURRET]]
5 = Endlessly turning tool memory　[YES, NO]
6 = Number of tool memory locations　[0...999]

7 = Number of tool spindles	[0...4]
8 = Number of tool grabbers	[0...4]
9 = Axis number of tool axis	[0...20]

Note: If there is no tool management (Column 3: NO), then all partial results from Column 4 are marked as [--].

If the requested process does not exist then there is no results line.

Example PTC2 Returns the process parameter data of the process 0.

FI command		**00_BR_PTC2_0**
Line	**Column**	**Answer**
1	1	0
	2	MILLING
	3	YES
	4	MAGAZINE
	5	YES
	6	8
	7	1
	8	2
	9	4

Edit PROVI Message Files: PVA

MWCX device group

Designation **PVA** **P**ROV**I**-Messages **A**ccess

Explanation This write command creates PROVI message files. With this write value, it is possible to decide whether the PROVI messages are to be generated according to the current PLC project, or selectively.

FI command **BW_PVA1** **(Single Write)**

Note: This command is an FI job command.

Value to be written

No write value exists	PROVI message files according to the current PLC project.
Write value exists	List of the requested PROVI message files (separated by a comma) according to the format: [PROVI-Diag-type: module number] Example: 01:01,01:02,02:02

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The response to the "BW_PVA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

....

Example PVA1

No write value is passed, i.e. the PROVI message files are generated according to the current PLC project.

FI command		00_BW_PVA1
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVA1
3	1	0

Explanation

The read command returns the most significant information on the created PROVI message files.

FI command **BR_PVA1** **(Single Read)**

....

Response Structure

The following table shows the general construction of the answer of the FI command BR_PVA1. For each available PROVI message file, 1 line with 10 columns each is created.

Line 1...n	Column 1	...	Column 10

Value Range/Meaning of Columns

1 =	PROVI diagnosis type	[1..20]
2 =	PROVI diagnosis type designation	[The following designations can be returned: StartCondition, Error, Message, Warning, Setup]
3 =	Module number	[1..99]
4 =	PROVI diagnosis type and module number	[PROVI diagnosis type: module number, see write value for BW_PVA2]
5 =	Complete name of the PROVI message text file	[max. 200 ASCII characters]
6 =	Memory required for the PROVI messages in the control	[figure in ASCII format]
7 =	Complete name of the PROVI index file	[max. 200 ASCII characters]
8 =	Memory required for the PROVI index data in the control	[figure in ASCII format]
9 =	Total memory (for text + index) required in the control	[figure in ASCII format]
10 =	Total memory for ALL PROVI data (text + index) required in the control	[figure in ASCII format]

Example PVA1

The most significant information of 2 available PROVI message files are returned.

FI command		00_BR_PVA1_1
Line	**Column**	**Answer**
1	1	1
	2	Error
	3	1
	4	01:01
	5	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	6	1345
	7	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	8	234
	9	1579
	10	4491
2	1	2
	2	Message
	3	1
	4	02:01
	5	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	6	2456
	7	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	8	456
	9	2912
	10	4491

Explanation

This write command transmits PROVI message files into the selected device. Through the write value, it is possible to chose whether ALL or only the PROVI messages selected via the write value are to be transmitted.

FI command **BW_PVA2** **(Single Write)**

Note: This command is an FI job command.

Value to be written

No write value exists	All PROVI message files are transmitted into the selected device
Write value exists	List of the requested PROVI message files (separated by a comma) according to the format: [PROVI-Diag-type: module number] Example: 01:01,01:02,02:02

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The response to the "BW_PVA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

....

Example PVA2

No write value is passed, i.e. all PROVI message files should be transmitted.

FI command		00_BW_PVA2
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVA2
3	1	0

Formatted Input / Output of PLC Variables: PVF

MWCX device group

Designation

PVF **P**LC **V**ariable **F**ormatted

Explanation

Formatted reading and writing of PLC variables, arrays and structures.

FI command

Read PLC variables.

CR_PVF_(1)	**(Single Read)**
CC_PVF_(1)	**(Cyclic Read)**
CB_PVF_(1)	**(Break Cyclic Read)**
(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]

Response Structure

One line with one column is output for single variables. For array and structure variables, one line per element is output, depending on the number of elements.

Line 1...n:	Column 1

n = number of elements.

Note: Only defined PLC variables can be read and written. Addressing a non-declared variable results in an error message. A PLC variable can only be read if its data length does not exceed 240 byte. (Refer also to chapter on “Programming” and “Guidelines”).

Value Ranges ANSI / ASCII

The value range of the response depends on the data type of the variable read. The following table indicates the range in which the results string is to be expected when reading out a single variable and into which C-data type this string can be converted without loss of information:

Data Type	Value Range	Can be converted to C-data type
BOOL	[0;1]	unsigned char
SINT	[-128...127]	char
INT	[-32768...32767]	short
DINT	[2147483648...2147483647]	long
USINT	[0...255]	unsigned char
UINT	[0...65535]	unsigned short
UDINT	[0...4294967295]	unsigned long
BYTE	[0x00...0xFF]	unsigned char
WORD	[0x0000....0xFFFF]	unsigned short
DWORD;	[0x00000000...0xFFFFFFFF]	unsigned long
TIME	[0...4294967295]	unsigned long (msec)
CHAR	[$00...$20,!...~,$7F...$FF]	char
STRING	<String> whereby <String> string is a character string with a maximum of as many characters as are declared for the string in the PLC	Char[xx+1]] +1 i.e. room for the zero byte
REAL	[-3.402823567E+38...3.402823567E+38]	Float

Note: An empty string is identified by two single inverted commas: ' ' (do not confuse with the double inverted commas '')!

All single variables can be part of array and structure variables. The value ranges maintain their validity, even when within structured data types.

Binary Value Range

The value range of the response depends on the data type of the variable read. The following table indicates the value range in which to expect the binary value of a single variable and how many bytes are included in the binary byte sequence:

Data Type	Value Range	Length (bytes)
BOOL	[00_H...01_H]	1
SINT	[80_H...$7F_H$] i.e. –128...127	1
INT	[8000_H (-32768)...$7FFF_H$ (32767)]	2
DINT	[80000000_H (-2147483648)... $7FFFFFFF_H$ (2147483647)]	4
USINT	[00_H (0)...FF_H (255)]	1
UINT	[00_H (0)...$FFFF_H$ (65535)]	2
UDINT	[0...4294967295]	4
BYTE	[0x00...0xFF]	1
WORD	[0x0000....0xFFFF]	2
DWORD;	[0x00000000...0xFFFFFFFF]	4
TIME	[0...4294967295]	4
CHAR	[$00...$20,!...~,$7F...$FF]	1

Data Type	Value Range	Length (bytes)
STRING	<String> whereby <String> string is a character string with a maximum of as many characters as are declared for the string in the PLC	XX+1
REAL	[-3.402823567E+38...3.402823567E+38]	4

Note: Binary array and structure elements will be connected to without space between (1 Byte Alignment).

PLC - Example 1 PVF

Read the value of the PLC variable "STK_TXT" in ASCII format from device address 00.

Assumption:
The "STK_TXT" variable is declared as STRING in the PLC program.

FI command		00_CR_PVF_STK_TXT/1
Line	**Column**	**Answer**
1	1	Repeat counter

WinPcl - Example 1 PVF

Read the value of WinPcl variable "STK_TXT" in ASCII format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "STK_TXT" is declared in WinPcl program "Prog" as STRING.

FI command		00_CR_PVF_:Prog.STK_TXT/1
Line	**Column**	**Answer**
1	1	Repeat counter

PLC - Example 2 PVF

Read the value of the PLC array "BEG_END" in ANSI format from device address 00.

Assumption:
The "BEG_END" variable is declared as BYTE with 2 elements in the PLC program.

FI command		00_CR_PVF_BEG_END/3
Line	**Column**	**Answer**
1	1	0x00
2	1	0x1F

WinPcl - Example 2 PVF

Read the value of WinPcl array "BEG_END" in ANSI format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "BEG_END" is declared in WinPcl program "Prog" as BYTE with two elements.

FI command		00_CR_PVF_:Prog.BEG_END/3
Line	**Column**	**Answer**
1	1	0x00
2	1	0x1F

PLC - Example 3 PVF

Read the value of the PLC structure "MSTRCT" in ASCII format from device address 00.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
            T1        BOOL
            T2        CHAR
            T3        STRING[16]
            T4        TIME
END
```

FI command		00_CR_PVF_MSTRCT/1
Line	**Column**	**Answer**
1	1	0
2	1	A
3	1	ROBOT AXIS X
4	1	2000

WinPcl - Example 3 PVF

Read the value of WinPcl structure "MSTRCT" in ASCII format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
            T1        BOOL
            T2        CHAR
            T3        STRING[16]
            T4        TIME
END
```

FI command		00_CR_PVF_:Prog.MSTRCT/1
Line	**Column**	**Answer**
1	1	0
2	1	A
3	1	ROBOT AXIS X
4	1	2000

FI command

Write PLC variable.

CW_PVF_(1)	**(Single Write)**
(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]

Value to be written

Value of data element	[see value ranges]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine. The data code of the value is passed to the parameter “ValType”.

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge   Data element has been set

Value Range of the value to be written in ANSI / ASCII Format

The value ranges agree for the most part with the ANSI / ASCII result-value ranges during read access. ANSI umlauts are thereby converted into ASCII umlauts. Only ASCII umlauts are stored in the control unit. For deviations to this, please refer to the following note:

Note: Strings are enclosed by two single inverted commas ' ' , e.g. 'drill'.

Special characters can be indicated in accordance with DIN-1131 by a $ sign.

The following are used:

- $' '
- $$ $
- $R \r (Carriage Return)
- $L \n (Linefeed)
- $P \f (Formfeed)
- $T \t (Tab)
- $xx refers to a character written as a hexadecimal value. e.g. $20 (space)

Array and structure elements are separated by a space.

Value Range of the Value to be written in Binary Format

The value ranges agree with the binary result-value range during read access. For deviations to this, please refer to the following note:

PLC - Example 4 PVF

Write into the PLC variable "STK_TXT" at device address 00. The value is passed in ANSI format.

Assumption:
The "STK_TXT" variable is declared as STRING in the PLC program.

FI command		00_CW_PVF_STK_TXT/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	'item counter'
Data code	/3

WinPcl - Example 4 PVF

Write into the WinPcl variable "STK_TXT" in WinPcl program "Prog" at device address 00. The value is passed in ANSI format.

Assumption:
The WinPcl variable "STK_TXT" is declared in WinPcl program "Prog" as STRING.

FI command		00_CW_PVF_:Prog.STK_TXT/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	'item counter'
Data code	/3

PLC - Example 5 PVF

Write into the PLC byte array "BEG_END" at device address 00. The value is passed in ANSI format.

Assumption:
The "BEG_END" variable is declared as a BYTE array with 2 elements in the PLC program.

FI command		00_CR_PVF_BEG_END/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	0x20 0x3f
Data code	/3

WinPcl - Example 5 PVF

Write into the WinPcl byte array "BEG_END" in WinPcl program "Prog" at device address 00. The value is passed in ANSI format.

Assumption:
The WinPcl variable "BEG_END" is declared in WinPcl program "Prog" as BYTE with two elements.

FI command		00_CW_PVF_:Prog.BEG_END/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	0x20 0x3f
Data code	/3

PLC - Example 6 PVF

Write the value of element T3 of the PLC structure "MSTRCT" at device address 00. The string “COUNTER” is transferred in binary format.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CW_PVF_MSTRCT.T3/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	Binary sequence: 43 4F 55 4E 54 45 52 00
Data code	/2

WinPcl - Example 6 PVF

Write the value of element T3 of the WinPcl structure "MSTRCT" at device address 00. The string “COUNTER” is transferred in binary format.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CW_PVF_:Prog.MSTRCT.T3/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	Binary sequence: 43 4F 55 4E 54 45 52 00
Data code	/2

PLC - Example 7 PVF

Write the value of the PLC structure "MSTRCT" from the structure “mstrct” previously stored in the C program at device address 00.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

To exchange binary data in a C program, the following “C” data type can be used:

```
#pragma pack(1)   //Write all elements
                  //without spaces next to each other.
typeder struct
{
      unsigned char T1;
      char          T2;
      char          T3[17]; //Space for zero byte
      unsigned long T4;
} Tymstrct;         // Declare structure
Tymstrct mstrct;  // Apply structure
```

FI command		00_CW_PVF_MSTRCT/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written: address of the C structure.

Value of data element	&mstrct
Data code	/2

WinPcl - Example 7 PVF

Write the value of the WinPcl structure "MSTRCT" from the structure “mstrct” previously stored in the C program at device address 00.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl programm "Prog" as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

To exchange binary data in a C program, the following “C” data type can be used:

```
#pragma pack(1)    //Write all elements
                   //without spaces next to each other.
typeder struct
{
      unsigned char T1;
      char         T2;
      char         T3[17]; //Space for zero byte
      unsigned long T4;
} Tymstrct;        // Declare structure
Tymstrct mstrct;  // Apply structure
```

FI command		00_CW_PVF_:Prog.MSTRCT/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written: address of the C structure.

Value of data element &mstrct

Data code /2

ProVi Messages: PVM

MWCX device group

Designation **PVM** **P**ro**V**i **M**essages

Explanation ProVi messages are output. These messages are assigned to a particular message type or module.

FI command Output all ProVi messages.

BR_PVM1_(1){_(2)} **(Single Read)**

BC_PVM1_(1){_(2)} **(Cyclic Read)**

(1) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -2!

Output first ProVi messages.

BR_PVM2_(1){_(2)} **(Single Read)**

BC_PVM2_(1){_(2)} **(Cyclic Read)**

(1) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the FI commands "PVM1" and "PVM2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 6

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[ASCII characters]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Message ID	[ASCII characters] (DWORD, decimal)
6 = Reference text available	[YES, NO]
7 = Criteria analysis exists	[YES, NO]
8 = Filename for additional information for message text	[e.g.HTML format]

Example PVM1

All ProVi errors from module 3 in control unit 0.

There are two messages.

FI command		**00_BR_PVM1_1_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	01.27.2000
	4	14:56:32
	5	43923028
	6	YES
	7	NO
	8	
2	1	Oil pressure too low
	2	124
	3	01.27.2000
	4	15:03:10
	5	98234039
	6	NO
	7	YES
	8	

Example PVM2

The first ProVi error from module 3 in control unit 0.

There are two messages:

FI command		**00_BR_PVM2_1_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	01.27.2000
	4	14:56:32
	5	43923028
	6	YES
	7	NO
	8	

FI command

Output the reference information of a ProVi message.

BR_PVM3_(1)_(2){_(3)}	**(Single Read)**
(1) = Message ID	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(3) = Module number	[1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the "PVM3" FI command.

Line 1	Column 1	...	Column 16

Meaning of the Columns

1 =	Message text	[ASCII characters]
2 =	Message number	[ASCII characters]
3 =	Error category	[ASCII characters] (empty no category)
4 =	Time stamp day	[mm.dd.yyyy]
5 =	Time stamp hour	[hh:mm:ss]
6 =	Reference text available	[YES, NO]
7 =	Reference text	[ASCII characters]
8 =	Message ID	[ASCII characters] (DWORD, decimal)
9 =	Diagnosis source	[ASCII characters] (PLC, CNC)
10 =	POE name	[ASCII characters]
11 =	Detail name	[ASCII characters] (leer Implementation)
12 =	Detail type	[1 = Action block, 3 = Transition, 4 = Implementation]
13 =	Network number	[ASCII characters]
14 =	Variable name	[ASCII characters]
15 =	POU entity name	[ASCII characters]
16 =	POU type	[2 = program, 3 = function block]
17 =	Analysis of criteria available	[YES, NO]
18 =	File name for additional information for message text	[e.g.HTML format]
19 =	File name for additional information for reference text	[e.g.HTML format]

Example PVM3

Reference text of a ProVi error with ID 43923028 from module 3 in control unit 0.

FI command		00_BR_PVM3_43923028_1_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	1
	4	01.27.2000
	5	14:56:32
	6	YES
	7	Oil pressure too low Oil pipe leaking or insufficient oil.
	8	43923028
	9	PLC
	10	MODULE3
	11	
	12	4
	13	34
	14	EschutzT
	15	Station2.Module3
	16	3
	17	NO
	18	
	19	D:\Programme\Indramat\MtGui\Project_000\ ProgramData\HMTL\DE\Error34.html

FI command

One after the other, all active ProVi messages are output. In the result, one line each is returned. After expiry of the set time, the next message is returned. The clock frequency can be set via the last parameter. This value can only be set once for the PC. The value transmitted last is always the valid value. Default setting is 1 second.

BR_PVM4_(1){_(2)_(3)}	**(Single Read)**
BC_PVM4_(1){_(2)_(3)}	**(Cyclic Read)**
(1) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(2) = Module number	[1...99] ! only for message type 1 -2!
(3) = Clock frequency	[ASCII characters] Time in ms

Response Structure

The following table shows the general structure of the "PVM4" FI command.

If there are no messages, the number of lines is 0.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[ASCII characters]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]

5 = Message ID	[ASCII characters] (DWORD, decimal)
6 = Reference text available	[YES, NO]
7 = Criteria analysis exists	[YES, NO]
8 = Message index (1 = 1. message)	[ASCII characters]
9 = Filename for additional information for message text	[e.g.HTML format]

Example PVM1 ProVi errors from module 3 in control unit 0.

The 2nd message is being output. The clock frequency is to be 2 seconds.

FI command		00_BR_PVM4_1_3_2000
Line	**Column**	**Answer**
1	1	Oil pressure too low
	2	124
	3	01.27.2000
	4	15:03:10
	5	98234039
	6	NO
	7	YES
	8	2
	9	

Download of PLC Retain Variables: PVR

MWCX device group

Designation **PVR** **P**LC **V**ariable **R**etain Backup

Explanation Download of PLC retain variables.

FI command **BW_PVR1!(1)** **(Single Write)**

(1) = Download file with path details.

Note: File and path details must be enclosed in inverted commas.

The separator "!" is used in this command.

Response Structure The response to the "PVR1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example PVR1 00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		**00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\ Temp\download.ini" /3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\ Temp\download.ini" /3
3	1	0

Structure of Download File The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Upload of PLC Retain Variables: PVR

MWCX device group

Designation **PVR** **P**LC **V**ariable **R**etain Backup

Explanation PLC retain variables are uploaded via all active processes.

FI command **BR_PVR1!(1)** **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.

The separator "!" is used in this command.

Response Structure The response to the "PVR1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

Example PVR 00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\ Temp\upload.ini" /3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\ Temp\upload.ini" /3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Note: Care must be taken in the use of upper and lower case letters.

Reading the PLC Variable Declaration: PVT

MWCX device group

Designation

PVT **P**LC **V**ariable **T**ype

Explanation

A PLC variable has a particular type. To evaluate complex variables such as structures and arrays, their components and types must be read out.

Refer also to PVF, Reading Structured PLC Variables.

FI command

Read the PLC variable type.

BR_PVT_(1)	**(Single Read)**
(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]

Response Structure

One line with 2 columns is output for each element of the variables.

Line 1...n:	Column 1	Column 2

n = number of elements.

Value Range/Meaning of Columns

(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]
2 = Type	[see value range PVF]

Examples:
PLC: Reading of a variable

Assumption:
The "TEST" variable is declared as WORD in the PLC program.

FI command	**00_BR_PVT_TEST**	
Answer		
Line	**Column 1 (Name)**	**Name**
1	TEST	WORD

WinPcl: Reading a Variable

Assumption:
The WinPcl variable "TEST" is declared as WORD in WinPcl program "Prog".

FI command	**00_BR_PVT_:Prog.TEST**	
Answer		
Line	**Column 1 (Name)**	**Name**
1	TEST	WORD

PLC: Reading a Structure

Assumption:
The "TEST1" variable is declared as STRUCT in the PLC program.

```
STRUCT
        E1      BOOL
        E2      INT
        E3      SINT
END
```

FI command	**00_BR_PVT_TEST1**	
Answer		
Line	**Column 1**	**Column 2**
1	TEST1.E1	BOOL
2	TEST1.E2	INT
3	TEST1.E3	SINT

WinPcl: Reading a Structure

Assumption:
The WinPcl variable "TEST1" is declared as STRUCT in WinPcl program "Prog".

STRUCT
E1 BOOL
E2 INT
E3 SINT
END

FI command	00_BR_PVT_:Prog.TEST1	
Answer		
Line	**Column 1**	**Column 2**
1	TEST1.E1	BOOL
2	TEST1.E2	INT
3	TEST1.E3	SINT

PLC: Reading an Array

Assumption:
The "TEST2" variable is declared as ARRAY in the PLC program.

ARRAY [
0 .. 3
] OF BOOL

FI command	00_BR_PVT_TEST2	
Answer		
Line	**Column 1**	**Column 2**
1	TEST2[0]	BOOL
2	TEST2[1]	BOOL
3	TEST2[2]	BOOL
4	TEST2[3]	BOOL

WinPcl: Reading an Array

Assumption:
The WinPcl variable "TEST2" is declared as ARRAY in WinPcl program "Prog".

ARRAY [
0 .. 3
] OF BOOL

FI command	00_BR_PVT_:Prog.TEST2	
Answer		
Line	**Column 1**	**Column 2**
1	TEST2[0]	BOOL
2	TEST2[1]	BOOL
3	TEST2[2]	BOOL
4	TEST2[3]	BOOL

PLC: Reading an Array of a Structure

Assumption:
The "TEST3" variable is declared as ARRAY in the PLC program.

ARRAY [
0 .. 1
] OF STRUCT1,

where STRUCT1 is declared as follows:

STRUCT
E1 BOOL
E2 INT
E3 SINT
END

FI command	00_BR_PVT_TEST3	
Answer		
Line	**Column 1**	**Column 2**
1	TEST3[0].E1	BOOL
2	TEST3[0].E2	INT
3	TEST3[0].E3	SINT
1	TEST3[1].E1	BOOL
2	TEST3[1].E2	INT
3	TEST3[1].E3	SINT

WinPcl: Reading an Array of a Structure

Assumption:
The WinPcl variable "TEST3" is declared as ARRAY in WinPcl program "Prog".

ARRAY [
0 .. 1
] OF STRUCT1,

where STRUCT1 is declared as follows:

STRUCT
E1 BOOL
E2 INT
E3 SINT
END

FI command	00_BR_PVT_:Prog.TEST3	
Answer		
Line	**Column 1**	**Column 2**
1	TEST3[0].E1	BOOL
2	TEST3[0].E2	INT
3	TEST3[0].E3	SINT
1	TEST3[1].E1	BOOL
2	TEST3[1].E2	INT
3	TEST3[1].E3	SINT

Assumption:
The data types are output according to IEC1131.

See also command PVF.

Repositioning Data: REP

MWCX device group

Designation **REP** **REP**ositioning Data

Explanation The data for re-approaching to contour of the selected device of the MWCX device group is read.

At the start of repositioning, the fixed data (end/setpoint values) can be called up with a 1st read command (CR_REP1). After that, the variable data must be called up repeatedly through a 2nd read command (CR_REP2, or CC_REP2) to show the actual status. With both commands, the reference system (machine or work piece coordinates) can be selected via the 2nd parameter.

FI command **CR_REP1_(1)_(2)** **(Single Read)**

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

(2) = Reference system [1...2] Machine/ workpiece coordinates

Response Structure 16 lines with varying numbers of columns are output:

- Line 1 contains the adjusting bit mask (13 bit values for the axes X …. S3, tool) which informs on whether an axis has already been adjusted.
- Line 1 contains the repos status (5 bit values).
- Line 3 to 14 for each possible axis (X ... S3) axis name, repos end position with full resolution, unit and repos end position with limited resolution.
- Line 15 contains the setpoint M functions of the spindles S1... S3.
- Line 16 contains the setpoint magazine position.

Example REP1 Read the fixed repositioning data in the NC process 0 of device address 00. The values are to be indicated in machine coordinates.

FI command		**00_CR_REP1_0_1**	
Line	**Column**	**Answer**	
1	1	0,1	Adjustment bit: X axis
	2	0,1	Adjustment bit: Y axis
	3	0,1	Adjustment bit: Z axis
	4	0,1	Adjustment bit: U axis
	5	0,1	Adjustment bit: V axis
	6	0,1	Adjustment bit: W axis
	7	0,1	Adjustment bit: A axis
	8	0,1	Adjustment bit: B axis
	9	0,1	Adjustment bit: C axis
	10	0,1	Adjustment bit: S1 axis
	11	0,1	Adjustment bit: S2 axis
	12	0,1	Adjustment bit: S3 axis
	13	0,1	Adjustment bit: tool axis

FI command		**00_CR_REP1_0_1**	
Line	**Column**	**Answer**	
2	1	0,1	Status bit: repos active
	2	0,1	Status bit: restart active
	3	0,1	Status bit: NPV data changed
	4	0,1	Status bit: tool corrections changed
	5	0,1	Status bit: repos/restart data prepared
3...14	1	X ... S3	Axis designation
	2	0.0000 ... +-999.9999	End position (full resolution)
	3	[mm], [inch], [deg]	Unit
	4	0.00 ... +- 999.99	End position (limited resol.)
15	1	M103 ... M119	Setpoint M function S1
	2	M203 ... M219	Setpoint M function S2
	3	M303 ... M319	Setpoint M function S3
16	1	1 ... 999	Setpoint magazine position

FI command

Reading variable repositioning data.

CR_REP2_(1)_(2) **(Cyclic Read)**

CC_REP2_(1)_(2) **(Cyclic Read)**

CB_REP2_(1)_(2) **(Cyclic Break)**

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

(2) = Reference system [1...2] Machine/ workpiece coordinates

Response Structure

16 lines with varying numbers of columns are output:

- Line 1 contains the adjusting bit mask (13 bit values for the axes X …. S3, tool) which informs on whether an axis has already been adjusted.
- Line 2 contains the repos status (5 bit values).
- Line 3 to 14 for each possible axis (X ... S3) current setpoint value (full resolution), unit, current setpoint value (limited resolution), repos distance to go (full resolution), unit and repos distance to go (limited resolution).
- Line 15 contains the setpoint M functions of the spindles S1... S3.
- Line 16 contains the setpoint magazine position.

Example REP2

Read the variable repositioning data in the NC process 0 of device address 00. The values are to be indicated in machine coordinates.

FI command		**00_CR_REP2_0_1**	
Line	**Column**	**Answer**	
1	1	0,1	Adjustment bit: X axis
	2	0,1	Adjustment bit: Y axis
	3	0,1	Adjustment bit: Z axis
	4	0,1	Adjustment bit: U axis
	5	0,1	Adjustment bit: V axis
	6	0,1	Adjustment bit: W axis
	7	0,1	Adjustment bit: A axis
	8	0,1	Adjustment bit: B axis
	9	0,1	Adjustment bit: C axis
	10	0,1	Adjustment bit: S1 axis
	11	0,1	Adjustment bit: S2 axis
	12	0,1	Adjustment bit: S3 axis
	13	0,1	Adjustment bit: tool axis
2	1	0,1	Status bit: repos active
	2	0,1	Status bit: restart active
	3	0,1	Status bit: NPV data changed
	4	0,1	Status bit: tool corrections changed
	5	0,1	Status bit: repos/restart data prepared
3...14	1	0.0000 ... +-999.9999	Set path (full resolution)
	2	[mm], [inch], [deg]	Unit
	3	0.00 ... +- 999.99	Set path (limited resol.)
	4	0.00 ... +- 999.99	Distance to go (full resol.)
	5	[mm], [inch], [deg]	Unit
	6	0.00 ... +- 999.99	Distance to go (limited resol.)
15	1	M103 ... M119	Setpoint M function S1
	2	M203 ... M219	Setpoint M function S2
	3	M303 ... M319	Setpoint M function S3
16	1	1 ... 999	Setpoint magazine position

Reference to Literature

See chapter entitled “Literature” [11].

SFC Diagnosis Data: SDD

MWCX device group

Designation **SDD** **S**FC **D**iagnosis **D**ata

Explanation Data for step chain diagnosis is output. Depending on the FI command this data can concern disrupted steps, actions, transitions or a definite ID to display the action or transition.

FI command Output the disrupted step of a step chain.

BR_SDD1!(1)!(2)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the FI command "SDD1".

Line 1	Column 1	...	Column 7

Meaning of the Columns

1 = Step name	[ASCII characters]
2 = Detail type	[1 = action block, 2 = action network, 3 = transition]
3 = Detail name	[ASCII characters]
4 = POU ID	[ASCII characters]
5 = Detail morpheme	[ASCII characters] (DWORD, decimal)
6 = Error ID	[ASCII characters] (DWORD, decimal)
7 = POU entity name	[ASCII characters]

Example SDD1 Query disrupted step of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD1!3!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	Open
	2	1
	3	Aopen
	4	SFC_1_2
	5	98243823
	6	34985304
	7	Station2.Module3

FI command Output the faulty action, monitor error or transition of a disrupted step.

BR_SDD2!(1)!(2)!(3)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Step name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD2".

Line 1	Column 1	...	Column 6

Meaning of the Columns

1 = Detail type	[1 = action block, 2 = action network, 3 = transition]
2 = Detail name	[ASCII characters]
3 = POU ID	[ASCII characters]
4 = Detail morpheme	[ASCII characters] (DWORD, decimal)
5 = Error ID	[ASCII characters] (DWORD, decimal)
6 = POU entity name	[ASCII characters]

Example SDD2

Query faulty action of the disrupted step “open” of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD1!3!Station03A.Clamp_Open**
Line	**Column**	**Answer**
1	1	1
	2	AOpen
	3	SFC_1_2
	4	98243823
	5	34985304
	6	Station2.Module3

FI command

Output the definite ID to display the action, monitor error or transition.

BR_SDD3!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD3".

Line 1	Column 1	...	Column 4

Meaning of the Columns

1 = POU ID	[ASCII characters]
2 = Detail morpheme	[ASCII characters] (DWORD, decimal)
3 = Error ID	[ASCII characters] (DWORD, decimal)
4 = POU entity name	[ASCII characters]

Example SDD3

Query ID to display the action “aOpen” of the “clamp” chain in module 3 in control unit 0.

FI command		00_BR_SDD3!3!Station03A.Clamp!1!aOpen
Line	**Column**	**Answer**
1	1	SFC_1_2
	2	98243823
	3	34985304
	4	Station2.Module3

FI command

Output the I/O addresses to display a detail.

BR_SDD4!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD4".

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Variable morpheme	[ASCII characters] (DWORD, decimal)
2 = I/O address	[ASCII characters]

Example SDD4

Query I/O addresses to display the action “aOpen” of the “clamp” chain in module 3 in control unit 0.

Three variables have an I/O address.

FI command		00_BR_SDD4!3!Station03A.Clamp!1!aOpen
Line	**Column**	**Answer**
1	1	98243823
	2	%I3.2.0
2	1	40923423
	2	%Q23.21.7
3	1	34985304
	2	%I100.3.5

FI command

Determine the multilingual comments for displaying a detail.

BR_SDD5!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD5".

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Comment morpheme [ASCII characters] (DWORD, decimal)

2 = New comment [ASCII characters]

Example SDD5

Query comments to display the action “aOpen” of the “clamp” chain in module 3 in control unit 0.

Two comments are replaced by another text.

FI command		**00_BR_SDD5!3!Station03A.Clamp!1!aOpen**
Line	**Column**	**Answer**
1	1	98243823
	2	Clamp open
2	1	40923423
	2	Clamp closed

FI command

Output the action that has not been performed, or the transition of a step calculated based on the online status.

BR_SDD6!(1)!(2)!(3) **(Single Read)**

(1) = Module number [1...99]

(2) = SFC entity name [ASCII characters]

(3) = Step name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD6".

Line 1	Column 1	...	Column 6

Meaning of the Columns

1 = Detail type [1 = action block, 3 = transition]

2 = Detail name [ASCII characters]

3 = POU ID [ASCII characters]

4 = Detail morpheme [ASCII characters] (DWORD, decimal)

5 = Error ID [ASCII characters] (DWORD, decimal)

6 = POE entity name [ASCII characters]

Example SDD6

Query the action that has not been performed for the step “open” of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD6!3!Station03A.Clamp_Open**
Line	**Column**	**Answer**
1	1	1
	2	AOpen
	3	SFC_1_2
	4	98243823
	5	34985304
	6	Station2.Module3

Set the Device Status Information: SDS

MWCX device group

Designation **SDS** **S**et **D**evice **S**tatus

Explanation By this command, the device status information can be set; here, the configuration file IND_DEV.INI is adjusted as well.

Note: When this command is transmitted, the following system messages are generated:
MSG_DEVICEOFF or MSG_DEVICE_ON !

FI command With this command, the device status information of **ALL** defined devices can be set.

BW_SDS1_(1) **(Single Write)**

Device status information to be set
0 = Device status information OFF
1 = Device status information ON

Response Structure The following table shows the general structure of the response to the "SDS1" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example SDS1 Set device status information to OFF for **ALL** defined devices.

FI command		00_BW_SDS1_0
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command With this command, the device status information for a selected device can be set.

BW_SDS2_(1) **(Single Write)**

(1) = Device status information to be set
0 = Device status information OFF
1 = Device status information ON

Response Structure The following table shows the general structure of the response to the "SDS2" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example: SDS2 Set device status information to OFF for the selected device 00.

FI command		00_BW_SDS2_0
Line	**Column**	**Answer**
1	1	(P_ACK)

Sequencer Data: SFD

MWCX device group

Designation **SFD** **SF**C **D**ata

Explanation Data for a step chain is output. Depending on the FI command this can concern a step chain comment, POU name, step comment, maximum time, action / transition / monitor error name (comment), qualifier and time value.

FI command Query the data for a step chain

BR_SFD1!(1)!(2)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "SFD1" FI command.

Line 1	Column 1	Column 2

Meaning of the Columns

1 = Step chain comment	[ASCII characters]
2 = POU name	[ASCII characters]

Example SFD1 Query data of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SFD1!3!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	Clamping device
	2	CLAMP

FI command Query the data of a step.

BR_SFD2!(1)!(2)!(3)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Step name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "SFD2" FI command. The number of lines depends on the number of actions and transitions.

If there are no details the line number is 1.

Line 1	Column 1	...	Column 3
Line 2...n:	**Column 1**	**...**	**Column 6**

Meaning of the Columns

Line 1

1 = Step comment	[ASCII characters]
2 = Maximum time	[ASCII characters]
3 = Minimum time	[ASCII characters]

Line 2...n:

1 = Detail type	[1 = action block, 3 = transition]
2 = Name	[ASCII characters]
3 = Comment	[ASCII characters]
4 = Boolean variable	[YES, NO]
5 = Qualifier	[ASCII characters]
6 = Time value	[ASCII characters]

Example SFD2

Data for the step “Open” in the “clamp” chain in module 3 on control unit 0.

FI command		**00_BR_SFD2!3!Station03A.Clamp!Open**
Line	**Column**	**Answer**
1	1	Open clamping device
	2	T#5s
	3	
2	1	1
	2	aOpen
	3	Clamp open
	4	NO
	5	D
	6	T#3s
3	1	3
	2	tOpen
	3	Clamping device is open
	4	NO
	5	
	6	

FI command

Output the data for a detail.

BR_SFD3!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "SFD3" FI command.

Line 1	Column 1	Column 2

Meaning of the Columns

1 = Comment	[ASCII characters]
2 = Boolean variable	[YES, NO]

Example SFD3 Data for the action “aOpen” in the “clamp” chain in module 3 on control unit 0.

FI command		00_BR_SFD3!3!Station03A.Clamp!aOpen
Line	**Column**	**Answer**
1	1	Clamp open
	2	NO

Sequencer Messages: SFE

MWCX device group

Designation **SFE** **SF**C **E**rror

Explanation The step chain messages of a module are output.

FI command Output all SFC messages.

BR_SFE1_(1) **(Single Read)**

BC_SFE1_(1) **(Cyclic Read)**

(1) = Module number [1...99]

Output first SFC messages.

BR_SFE2_(1) **(Single Read)**

BC_SFE2_(1) **(Cyclic Read)**

(1) = Module number [1...99]

Response Structure The following table shows the general structure of the FI commands "SFE1" and "SFE2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n:	Column 1	...	Column 7

Meaning of the Columns

1 = Message text [ASCII characters]

2 = SFC entity name [ASCII characters]

3 = Step name [ASCII characters]

4 = Time stamp day [mm.dd.yyyy]

5 = Time stamp time [hh:mm:ss]

6 = Type of error [1 = time error, 2 = monitor error, 3 = monitor event]

7 = Is there condition analysis? [YES, NO]

Example SFD1 All SFC messages from module 2 in control unit 0.

There are two messages.

FI command		00_BR_SFE1_2
Line	**Column**	**Answer**
1	1	TIME ERROR: Chain: chucking Step: up malfunction
	2	Station03A.Clamp
	3	Open
	4	01.27.2000
	5	11:56:32 AM
	6	1
	7	YES
2	1	ASSY ERROR: Chain: drilling Step: down malfunction
	2	Station02A.Drill
	3	Down
	4	01.27.200
	5	13:03:12
	6	2
	7	NO

Example SFE2 First SFC message from module 2 in control unit 0.

There are two messages.

FI command		00_BR_SFE2_2
Line	**Column**	**Answer**
1	1	TIME ERROR: Chain: chucking Step: up malfunction
	2	Station03A.Clamp
	3	Open
	4	01.27.2000
	5	14:56:32
	6	1
	7	YES

Sequencer Mode: SFM

MWCX device group

Designation **SFM** **SF**C **M**ode

Explanation Queries step chain mode.

FI command Query the mode of a step chain.

BR_SFM1!(1)!(2)	**(Single Read)**
BC_SFM1!(1)!(2)	**(Cyclic Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "SFM1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Mode [1 = time error, 2 = monitor error, 3 = monitor event, 10 = stop, 11 = auto, 12 = manual, 13 = jog]

Example SFM1 Query mode of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SFM1!3!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	1

Software Installation Data: SID

MWCX device group

Designation **SID** **S**oftware **I**nstallation **D**ata

Explanation Information is returned regarding installation. This information includes installation paths, the software version used, DLL mode, plus service pack and release information.

FI command Read-in the installation data.

BR_SID1	**(Single Read)**
BC_SID1	**(Cyclic Read)**

Response Structure One line with 8 columns is output for the returned values.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Basic directory	[EXE files of the BOF]
2 = FI installation directory	[FI directory]
3 = Data directory	[in accordance with BOF]
4 = GBO version	[from INDRAMAT.ini]
5 = IF-DLL mode	[from INDRAMAT.ini]
6 = IF version	[from INDRAMAT.ini from DLL mode 400]
7 = Service pack info	[from INDRAMAT.ini from DLL mode 420]
8 = Release info	[from INDRAMAT.ini from DLL mode 420]

Example SID1 Return information on the current installation.

FI command		00_BR_SID1
Line	**Column**	**Answer**
1	1	--
	2	D:\Programme\Indramat\MTGUI\Bin
	3	--
	4	005-22Vxx
	5	07.20
	6	07V00
	7	--
	8	--

Note: Refer to FI command “PHD” for working with absolute paths.

Servo Lag of an Axis: SLA

MWCX device group

Designation **SLA** **S**ervo **LA**g

Explanation The current servo lag of a selected axis of the MWCX device group is read out. The FI command "SLA1" returns the servo lag of an axis, related to the code of the axis meaning. The FI command "SLA2", on the other hand, returns the servo lag of an axis, related to the physical axis number.

FI command Output the servo lag of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_SLA1_(1)_(2){_(3)}	**(Single Read)**
CC_SLA1_(1)_(2){_(3)}	**(Cyclic Read)**
CB_SLA1_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = NC process number	[0...6]
(2) = Axis meaning	[0...11; 20]; (see Chapter 6.2 "Data Tables")
(3) = Required measurement system (opt.)	[mm, inch]

FI command Output the servo lag of the selected axis of the device specified, related to the physical axis number.

Using the optional second parameter it is possible to pre-select conversion of the result into mm or inches. If, however, a spindle is selected as an axis, indicating a measurement system serves no purpose.

CR_SLA2_(1){_(2)}	**(Single Read)**
CC_SLA2_(1){_(2)}	**(Cyclic Read)**
CB_SLA2_(1){_(2)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1...32]
(2) = Required measurement system (opt.)	[mm, inch]

Response Structure

The following table shows the general structure of the response to the FI commands "SLA1" and "SLA2". One line is output with 4 columns for the axis designation, servo lag, unit and the servo lag limited to “indicated decimal places”.

Line 1	Column 1	Column 2	Column 3	Column 4

Value Range/Meaning of Columns

1 = Axis name	[according to settings of axis parameters]
2 = Servo lag	[according to settings of process parameters]
3 = Unit	[according to settings of process parameters: [mm, inch]
4 = Servo lag	[as Column 2, but rounded up or down according to the parameter “indicated decimal places”]

Note: If the specified axis is not defined in the selected NC process then the response in all columns is [--].

Example SLA1

Read the servo lag of the Z axis in NC process 0 of device address 00.

FI command	00_CR_SLA1_0_2			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	2.9124	[mm]	2.912

Example SLA1

Read the servo lag of the Z axis in NC process 0 of device address 00. Values are displayed in inches:

FI command	00_CR_SLA1_0_2_inch			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	0.1147	[inch]	0.115

Example SLA2

Read the servo lag of the Z axis (e.g., physical axis number = 3) at device address 00.

FI command	00_CR_SLA2_3			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	Z1	2.9124	[mm]	2.912

Reference to Literature

See chapter entitled “Literature” [40].

PLC Long Identification: SLI

MWCX device group

Designation **SLI** **S**PS **L**ong **I**dentification

Explanation Returns the unit data from the PLC long identification.

FI command Read PLC long identification.

BR_SLI **(Single Read)**

Response Structure One line with 15 columns is output for the returned values.

Line 1	Column 1	Column...	Column 15

Value Range/Meaning of the Columns

1 =	Device address	[00...63]
2 =	Program number	[01...99]
3 =	Project name	[max. 8 ASCII characters]
4 =	Program name	[max. 8 ASCII characters]
5 =	User name	[acc. to password entry]
6 =	Program length	[bytes]
7 =	Compilation time	[LONG] (coded in long value)
8 =	Compilation date	[8 ASCII characters]
9 =	Compilation time	[8 ASCII characters]
10 =	Download time	[LONG] (coded in long value)
11 =	Download date	[8 ASCII characters]1
12 =	Download time	[8 ASCII characters]
13 =	Version of PLC long identification	[LONG]
14 =	RUN flags	[HEX value]
15 =	Compiler info	[LONG]

Example SLI Read the unit data from the PLC long identification.

FI command		**00_BR_SLI**
Line	**Column**	**Answer**
1	1	02
	2	01
	3	--
	4	MOT12
	5	TEST
	6	17672
	7	630163960
	8	15.12.99
	9	17:15:48
	10	630163961
	11	15.12.99
	12	17:15:50
	13	2
	14	0x0000

FI command		00_BR_SLI
Line	Column	Answer
	15	13

Reference to Literature see chapter entitled “Literature” [30].

SERCOS Parameters: SPA

MWCX device group

Designation **SPA** **S**ERCOS **PA**rameter

Explanation A SERCOS drive parameter is output or written. Each parameter consists of 7 elements, whereby 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	[1...7]
(2) = Parameter No.	in the format: X-Y-ZZZZ
(3) = Element coding	[standard or advanced format]

Parameter No.

Format X-Y-ZZZZ	Value Range
X	S = standard data P = product data
Y	[0..00.15] = parameter record
Z	[0...4095] = data block no.

Element Coding Element coding in standard format allows individual elements, such as the operating date, to be requested. If several elements are to be read out in one request, then the element coding can be OR’d in advanced format, e.g. operating date (0x40) and unit (0x08) produces OR’d (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	The marked section is then printed out.	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
Operating date	D	40H	(see Displaying the Operating Date	
Operating date, when no list		80H		

Displaying the Operating Date The display of the operating date depends on the parameter number requested.

Decimal

Decimal values are given as floating points, e.g. 1.5. Leading spaces, zeros, plus and minus signs as well as trailing spaces are allowed.

Hexadecimal

Hexadecimal values are displayed by "0x...", e.g. 0x80. Up to a maximum of eight positions are allowed. Leading or trailing spaces are allowed. Leading additional zeros or plus and minus signs are not allowed.

Binary (max. 32 characters)

Leading or trailing spaces are allowed. The decimal point serves as separator:

e.g., 1111.0000.1010.1100.1111.0000.1010.1100

Note: Leading additional zeros or plus and minus signs are not allowed.

ID number

The following table shows the general way in which the ID number is displayed:

Format X-Y-ZZZZ	Value Range
X	S = standard data P = product data
Y	[0..0.7] = parameter record
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 maximum length of the list. The length specification refers to the length of the list in the drive and therefore designates the number of bytes for storage (storage bytes). The number of elements in the list can be calculated using the attribute. The list elements are displayed according to the attribute. All parts of the list are separated from each other 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 variable length lists. The individual string characters are not separated by a line feed. When displaying the lists, a distinction is made between standard format and advanced format. In standard format, only the character string is returned; in advanced format, the actual length and the maximum length of the list (string) are also transmitted.

Example:

Parameter S-0-0030, operation date
Standard format: "DKC2.1-SSE-01V09"
Advanced format: "16\n16\nDKC2.1-SSE-01V09"

Response Structure

The following table shows the general structure of the response to the FI command “SPA1”. Line 1 is output both when reading and when writing. Additional lines are only output when reading depending on the element coding.

Note: If the element coding has been requested in standard format then the first line is not applicable.

Line 1 is a status line that either contains the Sercos error or displays the successful processing of the FI command. If the command has been processed successfully, then columns 1 and 3 contain the value [0x0000].

The number of the drive that reports the SERCOS error is output in the second column of the first line.

Line	Column 1	Column 2	Column 3	Column 4
1	<Sercos error>	<Drive no. SERCOS error>	0x0000	0x0000
2	Read: 1. Element corresponding to the element coding.			
...	...			
n	Reading: (n-1). Element corresponding to the element coding.			

Example SPA1/ read

Read parameter S-0-0003 of the 3rd drive (element coding 0x48)

FI command	**00_BR_SPA1_3_S-0-0003_48**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0000	0x0000	0x0000
2	µs			
3	2000			

Example SPA1/ write

Write the ID number P-0-0037 in parameter S-0-0305 of the 3rd drive (element coding 0x40).

Technical background:

- Realtime status bit 1 is to 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**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0003	0x0000	0x0000

Reference to Literature

See chapter entitled “Literature” [41].

See chapter entitled “Literature” [46].

Active SERCOS Phase Switch-Over: SPH

MWCX device group

Designation **SPH** **S**ERCOS **PH**ase

Explanation

All drives within a SERCOS ring are in the same communication phase. The phase status can be read-out or changed by this command.

FI command

CR_SPH_(1) **(Single Read)**

CW_SPH_(1) **(Single Write)**

(1) = Physical axis number [1...32]

Value to be written Phase [2, 4]

Response Structure

One line with one column is output for the returned value.

Line 1	Column 1

Note: The value to be written is passed to 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
Line	**Column**	**Answer**
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 write: 4	
Answer		
Line	**Column 1**	**Column 2**
1	52	1

Note: Switching over from phase 2 to phase 4 returns the value [52] as the result in column 1. On switching over from phase 4 to phase 2, column 1 contains the value [50]. The result of column 2 is the physical axis number in both cases.

Reference to Literature

See chapter entitled “Literature” [42].

Selected NC Program: SPP

MWCX device group

Designation **SPP** **S**elected **P**art-**P**rogram Number

Explanation The selected NC program is read.

FI command

CR_SPP_(1) **(Single Read)**

CC_SPP_(1) **(Cyclic Read)**

CB_SPP_(1) **(Break Cyclic Read)**

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

Response Structure

The response to the FI command “SPP” consists of one line with two columns for the identifier of the NC memory and the number of the selected NC program.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 = NC memory [A = NC memory A; B = NC memory B]

2 = Number of selected NC program [according to settings of process parameters]

Example SPP

Read the selected NC program in NC process 0 of device address 00.

FI command	00_CR_SPP_0	
Answer		
Line	**Column 1**	**Column 2**
1	B	55

Reading or Writing Tool Data Record: TDA

MWCX device group

Designation **TDA** **T**ool **DA**ta

Explanation A complete tool data record consisting of basic data and defined cutter data is read from or written into the control unit.

FI command Read the complete tool data record. For this FI command, the tool data record is addressed via the NC process number, the tool memory and the location number.

BR_TDA1_(1)_(2)_(3)	**(Single Read)**
(1) = NC process number	[0...6]
(2) = Tool memory	[M = magazine/turret, S = spindle, G = grabber]
(3) = Location number	[1...999]

Response Structure The following table shows the general structure of the response to the FI command "BR_TDA1". The number of lines depends on the number of cutters. The first line contains the basic data. The cutter data is listed from line 2 onwards. The basic data consist of 28 basic data elements, and the tool edge data of 40 tool edge data elements.

Line 1	Column 1	...	Column 28	
Line 2	Column 1	Column 2	...	Column 40
...	...	...	...	...
Line n+1	Column 1	Column 2		Column 40

n = number of cutters

Example TDA1 Read the complete tool data record

FI command		**03_BR_TDA1_0_M_21**
Line	**Column**	**Answer**
1	01	10156
	02	Cutter head D80
	03	M
	04	21
	05	1
	06	1
	07	2
	08	1
	09	-p
	10	0
	11	M 21
	12	M
	13	--
	14	M
	15	--
	16	[cycl]

Rexroth Indramat

FI command		03_BR_TDA1_0_M_21
Line	**Column**	**Answer**
	17	[mm]
	18	0
	19	0
	20	0.000000
	21	0.000000
	22	0.000000
	23	0.000000
	24	0.000000
	25	0.000000
	26	0.000000
	27	0.000000
	28	0.000000
2	01	1
	02	_
	03	100.000000
	04	5.000000
	05	0.000000
	06	0.000000
	07	0.0000
	08	0.0000
	09	104.8000
	10	40.0000
	11	0.0000
	12	0.0000
	13	0.0000
	14	0.0000
	15	0.0000
	16	0.0000
	17	0.0000
	18	0.0000
	19	-999.0000
	20	999.0000
	21	-999.0000
	22	999.0000
	23	-999.0000
	24	999.0000
	25	-999.0000
	26	999.0000
	27	0.0000

FI command		03_BR_TDA1_0_M_21
Line	Column	Answer
	28	0.0000
	29	0.0000
	30	0.0000
	31	0.000000
	32	0.000000
	33	0.000000
	34	0.000000
	35	0.000000
	36	0.0000
	37	0.0000
	38	0.0000
	39	0.0000
	40	0.0000

FI command

Write the complete tool data record. For this FI command, the tool data record is addressed via the NC process number, the tool memory and the location number.

Note: To create a tool data record, the name (ID) must be transferred to the device (see example “TDA1”, Write Tool Data).

BW_TDA1_(1)_(2)_(3)	**(Single Write)**
(1) = NC process number	[0...6]
(2) = Tool memory	[M = magazine/turret, S = spindle, G = grabber]
(3) = Location number	[1...999]

Values to be written

The values to be written are passed in a table. First, the 3-digit code of the data element must be passed and then the value to be written must be passed. The first position addresses the data record (0 = basic data record, 1 to 9 the corresponding cutter data record) and the second and third positions address the actual data element (also refer to "Basic Data" and "Tool edge data").

Data Element Code

1. Position	2. Position	3. Position
0 = basic data record or 1...9 = cutter data record	two-digit data element number	

Note: The character “|” (= 0x7D) is used as separator between the number of the data element and the value to be written. The individual lines of the table are also separated by a "|". <Element number n> <|> <Value n> <|> <Element number m> <|> <Value m> <|>

Example TDA1
Write Tool Data Record

Write the following data elements of the tool data record:

- Element number 002: Name (ID) "drill Z72"
- Element number 008: Number of tool edges "1" and
- Element number 107: Length L1 "100"

Assumption:

NC process number: 0
Tool magazine: M = magazine and
location number: 2

FI command	03_BW_TDA1_0_M_2
Values to be written	
002|Drill Z72|008|1|107|100	

FI command

Read the complete tool data record. For this FI command, the tool data record is addressed via the NC process number, the tool number and the index number.

BR_TDA2_(1)_(2)_(3)	**(Single Read)**
(1) = NC process number	[0...6]
(2) = Tool number	[1...9999999]
(3) = Index number	[1...9999]

Response Structure

The following table shows the general structure of the response to the FI command "BR_TDA2". The number of lines depends on the number of cutters. The first line contains the basic data. The cutter data is listed from line 2 onwards. The basic data consist of 28 basic data elements, and the tool edge data of 40 tool edge data elements.

Line 1	Column 1	...	Column 28	
Line 2	Column 1	Column 2	...	Column 40
...	...	...	...	...
Line n+1	Column 1	Column 2	...	Column 40

n = number of cutters

Example TDA2

Read the complete tool data record

FI command		03_BR_TDA2_0_1_1
Line	**Column**	**Answer**
1	01	10156
	02	Cutter head D80
	03	M
	04	21
	05	1
	06	1
	07	2
	08	1
	09	-p
	10	0
	11	M 21
	12	M
	13	--
	14	M
	15	--
	16	[cycl]
	17	[mm]

FI command		03_BR_TDA2_0_1_1
Line	**Column**	**Answer**
	18	0
	19	0
	20	0.000000
	21	0.000000
	22	0.000000
	23	0.000000
	24	0.000000
	25	0.000000
	26	0.000000
	27	0.000000
	28	0.000000
2	01	1
	02	_
	03	100.000000
	04	5.000000
	05	0.000000
	06	0.000000
	07	0.0000
	08	0.0000
	09	104.8000
	10	40.0000
	11	0.0000
	12	0.0000
	13	0.0000
	14	0.0000
	15	0.0000
	16	0.0000
	17	0.0000
	18	0.0000
	19	-999.0000
	20	999.0000
	21	-999.0000
	22	999.0000
	23	-999.0000
	24	999.0000
	25	-999.0000
	26	999.0000
	27	0.0000
	28	0.0000

FI command		03_BR_TDA2_0_1_1
Line	**Column**	**Answer**
	29	0.0000
	30	0.0000
	31	0.000000
	32	0.000000
	33	0.000000
	34	0.000000
	35	0.000000
	36	0.0000
	37	0.0000
	38	0.0000
	39	0.0000
	40	0.0000

Reference to Literature See chapter entitled “Literature” [43].

Loading Tool Data into the Control Unit: TDD

MWCX device group

Designation **TDD** **T**ool **D**ata **D**ownload

Explanation Downloading of a tool data record. After the tool list download has been initiated with "CR_TDI", the entire data for a tool is transferred into the control unit for each position of the tool memory. The data consists of a data record for the basic data and a data record for the cutter data for each cutter of the tool.

FI command Write the basic data or cutter data of a tool data record.

CW_TDD_(1)_(2)_(3)_(4) **(Single Read)**

(1) = NC process number	[0...6]
(2) = Tool memory	[M = magazine/turret, S =spindle, G = grabber, P = change position]
(3) = Tool memory location	In the magazine/turret: [1...999] In the spindle: [1...4] In the gripper: [1...4] In the change position: [1...4]
(4) = Cutter number	[0 = basic data, 1...9 = cutter data]

Value to be written Tool data record [basic and cutter data]

Note: The value to be written is passed to the "acValue" parameter as an ASCII string in the "DataTransfer" routine.

A tool data record consists of the individual writable tool data of the basic and cutter data, each separated from one another by a space (see Basic Data, Cutter Data).

The tool name (element No. 2 = 1st writable data of the basic data) can itself contain any characters (including spaces) and should therefore be character filled with exactly 28 characters with spaces.

Depending on the parameter setting it is possible that some of the basic or cutter data might not be relevant. Such data should nonetheless be included in the data record, e.g., with 0!

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Data element has been set

Example TDD

In NC process 0 of device 00, write into the control unit the data record for the basic data of the tool in the magazine at location number 2.

FI command		00_CW_TDD_0_M_2_0 <Data record>
Line	**Column**	**Answer**
1	1	(P_ACK)

<Data record> =
"Tool 1 1234567 1234 4 3 0xFF301900 0 0 1 6 5 1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0"

Example TDD

In NC process 0 of device 00, write into the control unit the data record for the 3rd cutter of the tool in the magazine at location number 2.

FI command		00_CW_TDD_0_M_2_3 <Data record>
Line	**Column**	**Answer**
1	1	(P_ACK)

<Data record> =
"1 0xF000 100.0 5 20000 1 2 3 4.0 0.1 0.2 0.3 0.4 0.01 0.02 0.03 0.04 0.001 0.002 0.003 0.004 1 2 3.1 4.1 5 11 0.222 -0.0333 9 10"

Status bits

The values for the status bits shown in the examples must be entered as a hexadecimal number (0x...), whereby the sequence should begin with the most significant bit 32 (tool status) or bit 16 (cutter status).

As only part of the status bit can be changed by the user, the form of these changeable bits is given here as an example:

W.Status: 0xFF301900 = 1111 1111 0011 0000 0001 1001 0000 0000

S.Status: 0xF000 = 1111 0000 0000 0000

Refer to Basic Data and Cutter Data for the meaning of the status bits.

Access to Tool Data Record: TDR

MWCX device group

Designation **TDR** **T**ool **D**ata **R**ecord

Explanation Returns a complete basic data record and/or cutter data record of a tool.

FI command

Read the basic data record or cutter data record of a tool in the tool memory.

CR_TDR1_(1)_(2)_(3)_(4) **(Single Read)**

CC_TDR1_(1)_(2)_(3)_(4) **(Cyclic Read)**

CB_TDR1_(1)_(2)_(3)_(4) **(Break Cyclic Read)**

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

(2) = Tool memory [M = magazine/turret, S = spindle, G = grabber, P = change position, X = index address]

(3) = Tool location	In the magazine/turret:	[1...999]
	In the spindle:	[1...4]
	In the gripper:	[1...4]
	In the change position:	[1...4]
	As an index address:	[0...9999999]
(4) = Data record	[0 = tool basic data, 1...9 = cutter data]	

Note: The index address of a tool is set by the device. For this reason, during the first access, access can only be made via tool memories M, S, G and P. Thereafter, the tool can also be addressed via the received index address.

Response Structure

The following table shows the general structure of the response to the "CR_TDR1" FI command. One line is output with 28 (basic data) or 40 (cutter data) columns for the returned values.

Line 1	Column 1	...	Column 28/40

Value Range/Meaning of the Columns

1.0.28 = requested basic tool data	[max. 28 data elements] (see basic value range data)
1..0.40 = requested tool cutter data	[max. 40 data elements] (see value range of cutter data)

Data elements 20 to 28 of the basic data and data elements 31 to 40 of the cutter data are only available as options (depending on the system parameters).

Example TDR1

Read the basic tool data record of the 2nd tool in the magazine in NC process 0.

FI command		00_CR_TDR1_0_M_2_0
Line	**Column**	**Answer**
1	1	928
	2	Miller D20
	3	M
	4	2
	5	1234567
	6	1234
	7	2
	8	1
	9	..............+..p............
	10	0
	11	M1
	12	M
	13	--
	14	M
	15	--
	16	[cycl]
	17	[mm]
	18	4
	19	102

FI command		00_CR_TDR1_0_M_2_0
Line	Column	Answer
	20	0.000000
	21	0.000000
	22	0.000000
	23	0.000000
	24	0.000000
	25	0.000000
	26	0.000000
	27	0.000000
	28	0.000000

Designation **TDR2** **T**ool **D**ata **R**ecord

FI command Read basic data record or cutter data record of a tool. Addressing is by means of the tool number and index number.

Attention: Before this command is executed, a tool identification run is required!

CR_TDR2_(1)_(2)_(3)_(4) **(Single Read)**

CC_TDR2_(1)_(2)_(3)_(4) **(Cyclic Read)**

CB_TDR2_(1)_(2)_(3)_(4) **(Break Cyclic Read)**

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

(2) = Tool number [1...9999999]

(3) = Index number [1...9999]

(4) = Data record [0 = tool basic data, 1...9 = cutter data]

Response Structure The following table shows the general structure of the response to the "CR_TDR2" FI command. One line is output with 28 (basic data) or 40 (cutter data) columns for the returned values.

Line 1	Column 1	...	Column 28/40

Value Range/Meaning of the Columns

1.0.28 = requested basic tool data [max. 28 data elements] (see basic value range data)

1..0.40 = requested tool cutter data [max. 40 data elements] (see value range of cutter data)

Data elements 20 to 28 of the basic data and data elements 31 to 40 of the cutter data are only available as options (depending on the system parameters).

Example TDR2

Read the basic tool-data record of tool 2 / duplo number 1 in NC process 0.

FI command		**00_CR_TDR2_0_2_1_0**
Line	**Column**	**Answer**
1	1	928
	2	Miller D20
	3	M
	4	2
	5	2
	6	1
	7	2
	8	1
	9	-p
	10	0
	11	M0
	12	M
	13	0
	14	M
	15	0
	16	~~0~~ [cycl]
	17	~~0~~ [mm]
	18	4
	19	102
	20	0.000000
	21	0.000000
	22	0.000000
	23	0.000000
	24	0.000000
	25	0.000000
	26	0.000000
	27	0.000000
	28	0.000000

Reference to Literature

See chapter entitled “Literature” [43].

Tool Insert Finish: TIF

MWCX device group

Designation **TIF** **T**ool **I**nsert **F**inish

Explanation Complete the insertion of a tool. The reservation of the tool memory location is lifted.

Refer also to: **CR_TII** and **CW_TLD1**

FI command Complete insertion.

CR_TIF_(1)_(2)_(3)		**(Single Read)**
(1) =	NC process number	[0...6]
(2) =	Tool memory	[M = magazine/turret, S =spindle, G = grabber, P = change position]
(3) =	Location number in the tool storage	in the magazine/turret: [1...999] in the spindle: [1...4] in the gripper: [1...4] in the change position: [1...4]

Response Structure One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Data element has been set

Example TIF Finish the insertion of a tool at location 5 in magazine in NC process 0 of device 00.

FI command		**00_CR_TIF_0_M_5**
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature See chapter entitled “Literature” [43].

Tool Insert Initiate: TII

MWCX device group

Designation **TII** **T**ool **I**nsert **I**nitiate

Explanation Initiate the insertion of an individual tool. Reserves a location in the tool memory.

After this, the basic data and the cutter data are to be entered by repeated inputting of **"**CW_TLD1". After the tool has actually been inserted in the tool memory, the procedure is completed by "CR_TIF".

FI command Initiate insertion.

CR_TII_(1)_(2)_(3)		**(Single Read)**
(1) =	NC process number	[0...6]
(2) =	Tool memory	[M = magazine/turret, S =spindle, G = grabber, P = change position]
(3) =	Location number in the tool storage	in the magazine/turret: [1...999] in the spindle: [1...4] in the gripper: [1...4] in the change position: [1...4]

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Data element has been set.

Example TII

Initiate the procedure for inserting tools in tool location at location number 5 in NC process 0 of device 00.

FI command		00_CR_TII_0_M_5
Line	**Column**	**Answer**
1	1	(P_ACK)

In the event of an error:

Error is returned by N_ACK error:

FI command		00_CR_TII_0_M_5
Line	**Column**	**Answer**
1	1	1 (= N_ACK error class)
	2	131 (= error number)
	3	0x00000000 (= additional information 0)
	4	Tool storage occupied (= error text)

Reference to Literature

See chapter entitled “Literature” [43].

Tool Basic Data List: TLB

MWCX device group

Designation **TLB** **T**oo**L** **B**asic Data List

Explanation

Returns the basic data of the tool list of the selected device of the MWCX device group.

FI command

Read selected basic data of the tool list.

BR_TLB1_(1)_(2)_(3)_(4)_(5)	**(Single Read)**
(1) = NC process number	[0...6]
(2) = Tool memory	[M = magazine/turret, S = spindle, G = grabber]
(3) = Location from	[1...999]
(4) = Location to	[1...999]
(5) = Data element	[1...28]

If more than one element is required as the 5th entry parameter then these are attached to the command with "_" and corresponding numbers.

Response Structure

The following table shows the general structure of the response to the FI command "BR_TLB1". The number of lines depends on the number of tools. One line with 2 columns is output per tool for the returned values. If more than one data element is requested then the number of columns increases accordingly.

Line 1...n:	Column 1	Column 2	...	Column 29

Value Range/Meaning of the Columns

1 =	Tool memory	[xxx = magazin/turret, SPx = spindle, GRx = gripper]
2...29 =	Requested base tool data	[max. 28 data elements] (see value range)

Example TLB1

Read data elements 2, 5, 6, 7.

Explanation of elements:

- Element number 002: Name (ID) [max. 28 ASCII characters]
- Element number 005: Tool number [1..9999999]
- Element number 006: Index number [1...9999] and
- Element number 007: Compensation type [1...5]

For additional elements, refer to basic data value range p. 7-287

Assumption:

- NC process number: 0
- Tool magazine: M = magazine and
- location number from: 2
- Location number to: 4

FI command		00_BR_TLB1_0_M_2_4_2_5_6_7
Line	**Column**	**Answer**
1	1	002
	2	TAPPER M6
	3	0
	4	1
	5	2
2	1	003
	2	DRILL MILLER D12
	3	0
	4	1
	5	1
3	1	004
	2	TWIST DRILL D4.8
	3	0
	4	1
	5	2

FI command

Read all basic data of the tool list.

BR_TLB2_(1)_(2) **(Single Read)**

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

(2) = Data element [1...28]

If more than one element is required as the 2nd entry parameter then these are attached to the command with "_" and corresponding numbers.

Response Structure

The following table shows the general structure of the response to the FI command "BR_TLB2". The number of lines depends on the number of tools. One line with 2 columns is output per tool for the returned values. If more than one data element is requested then the number of columns increases accordingly.

Line 1...n:	Column 1	Column 2	...	Column 29

Value Range/Meaning of the Columns

1 =	Tool memory	[xxx = magazine/turret, SPx = spindle, GRx = gripper]
2...29 =	Requested base tool data	[max. 28 data elements] (refer to basic data value range, p. 7-287)

Example TLB2

Read data elements 2, 5, 6, 7 in NC process 0.

Explanation of elements:

- Element number 002: Name (ID) [max. 28 ASCII characters]
- Element number 005: Tool number [1..9999999]
- Element number 006: Index number [1...9999] and
- Element number 007: Compensation type [1...5]

For more elements, refer to value range "Basic Data".

FI command		**00_BR_TLB2_0_2_5_6_7**
Line	**Column**	**Answer**
1	1	SP1
	2	--
	3	0
	4	0
	5	0
2	1	001
	2	END MILL D16
	3	0
	4	1
	5	2
3	1	002
	2	TAPPER M6
	3	0
	4	1
	5	2
4	1	003
	2	DRILL MILLER D12
	3	0
	4	1
	5	1
5	1	004
	2	TWIST DRILL D4.8
	3	0
	4	1
	5	2
6	1	005
	2	DRILL MILLER D8
	3	0
	4	1
	5	2

FI command		00_BR_TLB2_0_2_5_6_7
Line	**Column**	**Answer**
7	1	006
	2	SLAB MILLING CUTTER D60
	3	0
	4	1
	5	1
8	1	007
	2	--
	3	0
	4	0
	5	0

Reference to Literature See chapter entitled “Literature” [43].

Tool Data Record Elements: TLD

MWCX device group

Designation **TLD** **T**oo**L** **D**ata

Explanation Returns elements of the basic data or cutter data of a tool in the tool memory. Only basic data or data from one cutter can be returned in any one telegram. If data elements are to be combined from basic data and cutter data then the command "CR_TLD3" or "CR_TLD4" must be used.

For a complete data record of basic data or cutting data, please refer to "CR_TDR".

FI command Read element(s) of the basic data or cutter data of a tool.

CR_TLD1_(1)_(2)_(3)_(4)_(5) **(Single Read)**

CC_TLD1_(1)_(2)_(3)_(4)_(5) **(Cyclic Read)**

CB_TLD1_(1)_(2)_(3)_(4)_(5) **(Break Cyclic Read)**

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

(2) = Tool memory [M = magazine/turret, S = spindle, G = gripper, P = change position, X = index address]

(3) = Tool location
in the magazine/turret: [1...999]
in the spindle: [1...4]
in the gripper: [1...4]
in the change position: [1...4]
as Index address: [0...9999999]

(4) = Data record [0 = tool basic data, 1...9 = cutter data]

(5) = Data element
The basic data: [1...28]
of the tool edge data: [1...40]

Data elements 20 to 28 of the basic data and data elements 31 to 40 of the cutter data are only available as options (depending on the system parameters). The response to access to data elements that are not available is "N_ACK" (Negative Acknowledge).

If more than one element is required as the 5th entry parameter then these are attached to the command with "_" and corresponding numbers.

Note: The index address of a tool is set by the device. For this reason, during the first access, access can only be made via tool memories M, S, G and P. Thereafter, the tool can also be addressed via the received index address.

Response Structure

The following table shows the general structure of the response to the FI command "CR_TLD1". One line with one column is output for the returned value. If more than one data element is requested then the number of columns increases correspondingly.

Line 1	Column 1	...	Up to column 28/40

Value Range/Meaning of the Columns

1.0.28 = requested basic tool data — [max. 28 data elements] (see basic value range data)

1..0.40 = requested tool cutter data — [max. 40 data elements] (see value range of cutter data)

Example TLD1

Read the name (basic data 2) of the 4th tool in the magazine in NC process 0.

FI command		00_CR_TLD1_0_M_4_0_2
Line	**Column**	**Answer**
1	1	MILLER D24

FI command

Write single element of basic data or cutter data of a tool in the tool memory.

CW_TLD1_(1)_(2)_(3)_(4)_(5) (Single Write)

(1) = NC process number — [0...6]

(2) = Tool memory — [M = magazine/turret, S = spindle, G = gripper, P = change position, X = index address]

(3) = Tool memory location — [in the magazine/turret: [1...999]
in the spindle: [1...4]
in the gripper: [1...4]
in the change position: [1...4]
as Index address: [0...9999999]

(4) = Data record — [0 = tool basic data, 1...9 = cutter data]

(5) = Data element — The basic data: [1...28]
of the tool edge data: [1...40]

Data elements 20 to 28 of the basic data and data elements 31 to 40 of the cutter data are only available as options (depending on the system parameters). The response to access to data elements that are not available is N_ACK (Negative Acknowledge).

Note: The index address of a tool is set by the device. For this reason, during the first access, access can only be made via tool memories M, S, G and P. Thereafter, the tool can also be addressed via the received index address.

Value to be written

Value of data element — see value ranges for basic and cutter data

---

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge — Data element has been set.

Example TLD1

Write data element 4 (warning limit) in NC process 0 for the tool at the 3rd magazine position in cutter 1.

FI command		**00_CW_TLD1_0_M_3_1_4** **Value to be written: 6.5**
Line	**Column**	**Answer**
1	1	(P_ACK)

Designation

TLD2 — **T**oo**L** **D**ata

Explanation

Read basic data or cutter data element(s) of a tool. Addressing is by means of the tool number and index number.

Attention: Before this command is executed, a tool identification run is required!

FI command

CR_TLD2_(1)_(2)_(3)_(4)_(5)	**(Single Read)**
CC_TLD2_(1)_(2)_(3)_(4)_(5)	**(Cyclic Read)**
CB_TLD2_(1)_(2)_(3)_(4)_(5)	**(Break Cyclic Read)**
(1) = NC-Process number	[0...6]
(2) = Tool number	[1...9999999]
(3) = Index number	[1...9999]
(4) = Data record	[0 = tool basic data, 1...9 = cutter data]
(5) = Data element	of the base data: [1...28] of the tool edge data: [1...40]

Data elements 20 to 28 of the basic data and data elements 31 to 40 of the cutter data are only available as options (depending on the system parameters). The response to access to data elements that are not available is N_ACK (Negative Acknowledge).

If more than one element is required as the 5th entry parameter then these are attached to the command with "_" and corresponding numbers.

Response Structure

The following table shows the general structure of the response to the FI command "CR_TLD2". One line with one column is output for the returned value. If more than one data element is requested then the number of columns increases correspondingly.

Line 1	Column 1	...	Column 28/40

Value Range/Meaning of the Columns

1.0.28 = requested basic tool data	[max. 28 data elements] (see basic value range data)
1..0.40 = requested tool cutter data	[max. 40 data elements] (see value range of cutter data)

Rexroth Indramat

Example TLD2

Read the name (basic data 2) of the 3rd tool/index no. 1 in NC process 0.

FI command		**00_CR_TLD2_0_3_1_0_2**
Line	**Column**	**Answer**
1	1	TAPPER M5

FI command

Write single element of basic data or cutter data of a tool. Addressing via tool number + index number.

CW_TLD2_(1)_(2)_(3)_(4)_(5)	**(Single Write)**
(1) = NC-Process number	[0...6]
(2) = Tool number	[1...9999999]
(3) = Index number	[1...9999]
(4) = Data record	[0 = tool basic data, 1...9 = cutter data]
(5) = Data element	of the base data: [1...28] of the tool edge data: [1...40]

Data elements 20 to 28 of the basic data and data elements 31 to 40 of the cutter data are only available as options (depending on the system parameters). The response to access to data elements that are not available is N_ACK (Negative Acknowledge).

Value to be written

Value of data element — see value ranges for basic and cutter data

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge — Data element has been set

Example TLD2

Write data element 4 (warning limit) in NC process 0 for tool number 3/index number 1 in cutter 1.

FI command		**00_CW_TLD2_0_3_1_1_4** **Value to be written: 6.5**
Line	**Column**	**Answer**
1	1	(P_ACK)

Designation

TLD3 **T**oo**L** **D**ata

Explanation

Returns any element of the basic data or cutter data of a tool in any order.

In contrast with the command "TLD1", for this FI command addressing an element is extended to three positions. The first position addresses the data record (0 = basic data record, 1-9= cutter data) and the second and third positions address the actual data element.

Addressing Examples

002 Basic data – tool name

103 Cutter 1 – remaining tool life

203 Cutter 2 – remaining tool life

Note:	You should always make sure when requesting tool data that the maximum net data length of 240 bytes is not exceeded. If more than 240 bytes are requested then the control unit returns the error message (NACK) /FI (1014).

FI command

Reading of basic data and cutter data of a tool in the tool memory.

CR_TLD3_(1)_(2)_(3)_(4) **(Single Read)**
CC_TLD3_(1)_(2)_(3)_(4) **(Cyclic Read)**
CB_TLD3_(1)_(2)_(3)_(4) **(Break Cyclic Read)**

(1) = NC-Process number	[0...6]	
(2) = Tool memory	[M = magazine/turret, S = spindle, G = gripper, P = change position, X = index address]	
(3) = Tool memory location	In the magazine/turret:	[1...999]
	In the spindle:	[1...4]
	In the gripper:	[1...4]
	In the change position:	[1...4]
	As an index address:	[0...9999999]
(4) = Data element	[001...940]	

Data elements 020 to 028 of the basic data and data elements x31 to x40 of the cutter data are only available as options (depending on the system parameters). The response to access to data elements that are not available is "N_ACK" (Negative Acknowledge).

If more than one element is required as the 4th entry parameter then these are attached to the command with "_" and corresponding numbers.

Note:	The index address of a tool is set by the device. For this reason, during the first access, access can only be made via tool memories M, S, G and P. Thereafter, the tool can also be addressed via the received index address.

Response Structure

The following table shows the general structure of the response to the FI command “CR_TLD3”. One line with one column is output for the returned value. If more than one data element is requested then the number of columns increases accordingly.

Line 1...n:	Column 1	...	Column xxx

Value Range/Meaning of the Columns

1...xxx = requested basic tool data and cutter data	see value ranges for basic and cutter data

Example TLD3

Read the name of the basic tool data of the 4th tool in the magazine and the remaining tool life of cutter 1 in NC process 0.

FI command		**00_CR_TLD3_0_M_4_002_103**
Line	**Column**	**Answer**
1	1	MILLER D24
	2	100.00

Designation

TLD4 **T**oo**L** **D**ata

Explanation

Returns any element of the basic data or cutter data of a tool in any order.

In contrast with the command "TLD2", for this FI command addressing an element is extended to three positions. The first position addresses the data record (0 = basic data record, 1-9= cutter data) and the second and third positions address the actual data element.

Addressing Examples

002 Basic data – tool name

103 Cutter 1 – remaining tool life

203 Cutter 2 – remaining tool life

Note: You should always make sure when requesting tool data that the maximum net data length of 240 bytes is not exceeded. If more than 240 bytes are requested then the control unit returns the error message (NACK) /FI (1014).

FI command

Read the basic and cutter data of a tool according to the tool number and index number.

CR_TLD4_(1)_(2)_(3)_(4)	**(Single Read)**
CC_TLD4_(1)_(2)_(3)_(4)	**(Cyclic Read)**
CB_TLD4_(1)_(2)_(3)_(4)	**(Break Cyclic Read)**
(1) = NC-Process number	[0...6]
(2) = Tool number	[1...9999999]
(3) = Index number	[1...9999]
(4) = Data element	[001...940]

Data elements 020 to 028 of the basic data and data elements x31 to x40 of the cutter data are only available as options (depending on the system parameters). The response to access to data elements that are not available is N_ACK (Negative Acknowledge).

If more than one element is required as the 4th entry parameter then these are attached to the command with "_" and corresponding numbers.

Response Structure

The following table shows the general structure of the response to the FI command “CR_TLD4”. One line with one column is output for the returned value. If more than one data element is requested then the number of columns increases correspondingly.

Line 1...n:	Column 1	...	Column xxx

Value Range/Meaning of the Columns

1...xxx = requested basic tool data and cutter data	see value ranges for basic and cutter data

Example TLD4

Read the name of tool number 3/index number 1 and the remaining tool life of cutter 4 in NC process 0 of device 00.

FI command		00_CR_TLD4_0_3_1_002_403
Line	**Column**	**Answer**
1	1	TAPPER M5
	2	100.00

Reference to Literature

See chapter entitled “Literature” [43].

Tool Edge Data List: TLE

MWCX device group

Designation **TLE** **T**oo**L** **E**dge Data List

Explanation Returns the cutter data of the tool list.

FI command Read selected cutter data of the tool list.

BR_TLE1_(1)_(2)_(3)_(4)_(5)_(6)	**(Single Read)**
(1) = NC process number	[0...6]
(2) = Tool edge	[1...9]
(3) = Tool memory	[M = magazine/turret, S = spindle, G = gripper]
(4) = Location from	[0...999]
(5) = Location to	[0...999]
(6) = Data element	[1...40]

If more than one element is required as the 6th entry parameter then these are attached to the command with "_" and corresponding numbers.

Response Structure The following table shows the general structure of the response to the FI command "BR_TLE1". The number of lines depends on the number of tools. One line with 2 columns is output per tool for the returned values. If more than one data element is requested then the number of columns increases accordingly.

Line 1...n:	**Column 1**	**Column 2**	**...**	**Column 41**

Value Range/Meaning of the Columns

1 =	Tool memory	[xxx = magazine/turret, SPx = spindle, GRx = gripper]
2...41 =	Requested tool cutter data	[max. 40 data elements] (see value range Tool edge data)

Example TLE1 Element number 002: Tool edge status is requested.

Assumption:

- NC process number: 0
- Tool edge: 1
- Tool magazine: M = magazine and
- location number from: 1
- location number to: 3

Read data elements 2 and 3.

FI command		**00_BR_TLE1_0_1_M_1_3_2_3**
Line	**Column**	**Answer**
1	1	001
	2	d (tool worn out)
2	1	002
	2	_ (tool ok)
3	1	003
	2	w (fallen below warning limit)

FI command Read all cutter data of the tool list.

BR_TLE2_(1)_(2)_(3) **(Single Read)**

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

(2) = Cutter position [0...8]

(3) = Data element [1...40]

If more than one element is required as the 3rd entry parameter then these are attached to the command with "_" and corresponding numbers.

Response Structure

The following table shows the general structure of the response to the FI command "BR_TLE2". The number of lines depends on the number of cutters. One line with 2 columns is output per cutter for the returned values. If more than one data element is requested then the number of columns increases accordingly.

Line 1...n:	Column 1	Column 2	...	Column 41

Value Range/Meaning of the Columns

1 =	Tool memory	[00 = magazine/turret, SP = spindle, GR = gripper]
2...41 =	Requested base tool data	[max. 40 data elements] (see value range Tool edge data)

Example TLE2

- Element number 003: Residual tool life [0.0000...100.0000]
- Element number 004: Warning limit [0.1...100.00]
- Element number 005: Maximum period of use [0...9999999]
- Element number 009: Length L3 [-9999.9999...9999.9999]

Read in NC process 0 the data elements 3, 4, 5, 9 for all tools at cutter position 1.

FI command		**00_BR_TLE2_0_1_3_4_5_9**
Line	**Column**	**Answer**
1	1	SP1
	2	0.0000
	3	0.0000
	4	0.0000
	5	0.0000
2	1	001
	2	100.0000
	3	5.0000
	4	0.0000
	5	106.8500
3	1	002
	2	100.0000
	3	5.0000
	4	0.0000
	5	132.9600
4	1	003
	2	48.0000
	3	5.0000
	4	100.0000

FI command		00_BR_TLE2_0_1_3_4_5_9
Line	Column	Answer
	5	106.8000
5	1	004
	2	99,8617
	3	5.0000
	4	0.0000
	5	180.0900
6	1	005
	2	100.0000
	3	5.0000
	4	0.0000
	5	78.7000
7	1	006
	2	100.0000
	3	0.0000
	4	0.0000
	5	116.0000
8	1	007
	2	0.0000
	3	0.0000
	4	0.0000
	5	0.0000

Reference to Literature See chapter entitled “Literature” [43].

Tool Move : TMV

MWCX device group

Designation **TMV** **T**ool **M**o**V**e

Explanation A complete tool data record consisting of basic data and defined cutter data is moved. This corresponds to the Rexroth Indramat BOF function "Tool Move".

FI command Move the selected tool data record.

CR_TMV_(1)_(2)_(3)_(4)_(5) **(Single Read)**

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

(2) = Current tool memory [M = magazine/turret, S = spindle, G = grabber]

(3) = Current location number [1...999]

(4) = Target tool memory [M = magazine/turret, S = spindle, G = grabber]

(5) = Target location number [1...999]

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge Data record has been moved

Example TMV

Move the 24th tool data record in the magazine to the 25th tool data record in the magazine.

Assumption:
There is a valid tool in magazine location 24 in NC process 0 at device address 00.

FI command		00_CR_TMV_0_M_24_M_25
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature

See chapter entitled “Literature” [43].

Information on Grippers/Spindles/Tool Magazine Locations: TPI

MWCX device group

Designation

TPI **T**ool **P**osition **I**nformation

FI command

This command is used to read the index addresses of the currently occupied tool storage locations. Through the parameters <VON location> ('f'om location') and <BIS location> ('to location'), the requested range can be determined. ' When these two parameters are NOT set, ALL occupied index addresses of the tool storage locations defined in the process parameters are returned.

BR_TPI1_(1)_{(2)_(3)}		**(Single Read)**
(1) =	Process number	[0..6]
(2) =	<VON location> - Start location index in the tool storage location administration (optional parameter)	[1..1007] 1..4 = Gripper 1..4 5..8 = Spindle 1..4 9..1007 = Magazine location 1..999
(3) =	<VON location> - End location index in the tool storage location administration (optional parameter)	[1..1007] 1..4 = Gripper 1..4 5..8 = Spindle 1..4 9..1007 = Magazine location 1..999

Response Structure

The following table shows the general structure of the response to the FI command “TPI1”. N lines, each with 3 columns, are output. Each line corresponds to one occupied tool storage location.

Line 1...n	Column 1	Column 2	Column 3

Value Range/Meaning of Columns

1 =	Location name	[G1..G4 = Gripper 1..4 S1..S2 = Spindle 1..4 M = Tool storage location]
2 =	Location name	[1..999]
3 =	Index address of the tool as a LONG value	[LONG value]

Example TPI1 Reads the index addresses of ALL occupied tool storage locations of the process 0 of device 00.

FI command		00_BR_TPI1_0
Line	**Column**	**Answer**
1	1	G1
	2	1
	3	1834
2	1	S1
	2	1
	3	2345
3	1	M
	2	1
	3	1456
4	1	M
	2	3
	3	3456
5	1	M
	2	9
	3	1678

FI command This command is used to read the location status bytes of the occupied or free tool storage locations. Through the parameters <VON location> ('f'om location') and <BIS location> ('to location'), the requested range can be determined. ' When these two parameters are NOT set, ALL location status bytes of the tool storage locations defined in the process parameters are returned.

BR_TPI2_(1)_{(2)_(3)}		(Single Read)
(1) =	Process number	[0..6]
(2) =	<VON location> - Start location index in the tool storage location administration (optional parameter)	[1..1007] 1..4 = Gripper 1..4 5..8 = Spindle 1..4 9..1007 = Magazine location 1..999
(3) =	<VON location> - End location index in the tool storage location administration (optional parameter)	[1..1007] 1..4 = Gripper 1..4 5..8 = Spindle 1..4 9..1007 = Magazine location 1..999

Response Structure The following table shows the general structure of the response to the FI command “TPI2”. A line of n columns is output. Here, the column index corresponds to the location index.

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 = Location status byte for location index	[0x00-0xFF]
2 = Location status byte for location index+1	[0x00-0xFF]
3 = Location status byte for location index+2	[0x00-0xFF]
....	[0x00-0xFF]
n = Location status byte for location index+n	[0x00-0xFF]

Example TPI2 Read the location status bytes of ALL tool storage locations of the process 0 of device 00. Here, 1 gripper and 1 spindle and 2 magazine locations are defined in the process parameters of the process 0. However, the location status bytes of the grippers 1…4 and the spindles 1…4 are ALWAYS returned.

FI command		**00_BR_TPI2_0**
Line	**Column**	**Answer**
1	1	0x80 (gripper 1)
	2	0x00 (gripper 2)
	3	0x00 (gripper 3)
	4	0x00 (gripper 4)
	5	0x80 (spindle 1)
	6	0x00 (spindle 2)
	7	0x00 (spindle 3)
	8	0x00 (spindle 4)
	9	0x80 (magazine location 1)
	10	0x80 (magazine location 2)

Torque: TQE

MWCX device group

Designation **TQE** **T**or**Q**u**E**

Explanation The torque at a selected axis of the MWCX device group is read. The FI command "TQE1" returns the torque of an axis, related to the code of the axis meaning. On the other hand, the FI command "TQE2" returns the torque of an axis, related to the physical axis number.

FI command Output the torque of the selected device of the MWCX device group, related 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) = NC process number	[0...6]
(2) = Axis meaning	[0...11; 20];

Response Structure The following table shows the general structure of the response to the FI command “TQE1”. One line with three columns is output for the name of the axis, the torque and the unit [%].

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Axis name	[according to settings of axis parameters]
2 = Torque	[format acc. to settings of the process parameter]
3 = Unit	[%]

Note: If the specified axis is not defined in the selected NC process then the response in all columns is [--].

Example TQE1 Read the torque at the Z axis in NC process 0 of device address 00.

FI command	00_CR_TQE1_0_2		
Answer			
Line	Column 1	Column 2	Column 3
1	Z	-25.6	[%]

FI command Output the torque at the selected axis of the device specified, related 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]

Response Structure The following table shows the general structure of the response to the FI command “TQE2”. One line with three columns is output for the name of the axis, the torque and the unit [%].

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Axis name [according to settings of axis parameters]

2 = Torque [format acc. to settings of the process parameter]

3 = Unit [%]

Note: If the specified axis is not defined in the selected NC process then the response in all columns is [--].

Example TQE2 Read the torque at the Z axis (physical axis number = 3) at device address 00.

FI command	00_CR_TQE2_3		
Answer			
Line	Column 1	Column 2	Column 3
1	Z	-25.6	[%]

Removing Tool Data Record: TRM

MWCX device group

Designation **TRM** **T**ool **R**e**M**ove

Explanation A complete tool data record consisting of basic data and defined cutter data is removed from the device. This corresponds to the Rexroth Indramat BOF function "Remove Tool from the Magazine List".

FI command Remove the selected tool data record.

CR_TRM_(1)_(2)_(3) **(Single Read)**

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

(2) = Tool memory [M = magazine/turret, S = spindle, G = gripper]

(3) = Location number [1...999]

Response Structure One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge Data record has been removed

Example TRM Remove the 24th tool data record.

Assumption:
There is a valid tool in magazine location 24 in NC process 0 at device address 00.

FI command		00_CR_TRM_0_M_24
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature See chapter entitled “Literature” [43].

Resetting Remaining Tool Life of a Tool: TRS

MWCX device group

Designation **TRS** **T**ool **R**e**S**et

Explanation Resets the percentage remaining tool life of a tool to 100%. The positive or negative execution of the function is reported back via the return value of the routine.

FI command Reset remaining tool life of a tool:

CR_TRS_(1)_(2)_(3) **(Single Read)**

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

(2) = Tool memory [M = magazine/turret, S = spindle, G = gripper, P = change position, X = index address]

(3) = Tool location
in the magazine/turret: [1...999]
in the spindle: [1...4]
in the gripper: [1...4]
in the change position: [1...4]
as Index address: [0...9999999]

Note: The index address of a tool is set by the device. For this reason, during the first access, access can only be made via tool memories M, S, G and P. Thereafter, the tool can also be addressed via the received index address.

Response Structure One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Tool has been reset

Example TRS Reset the remaining tool life for the tool located in change position 1 in NC process 0 of device 00.

FI command		00_CR_TRS_0_P_1
Line	**Column**	**Answer**
1	1	(P_ACK)

Reference to Literature See chapter entitled “Literature” [43].

Requesting Watch List Allocations: WLA

MWCX device group

Designation **WLA** **W**atch **L**ist **A**llocation

Explanation Requests free watch list allocations. A maximum of ten free watch list allocations can be requested with one FI command.

BR_WLA1_(1)		**(Single Read)**
(1) =	Number of the requested free watch list numbers	The required number of free watch list allocations is identified here. The allowed value range: 1..10

Response Structure The following table shows the general structure of the response to the FI command "WLA1".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. free watch list allocation	Value range: 1..16
2 =	2. free watch list allocation	Value range: 1..16
3 =	3. free watch list allocation	Value range: 1..16
n =	nth free watch list allocation	Value range: 1..16

Example WLA1 Request four free watch list allocations.

<u>Assumption:</u>
Watch list allocations 3 and 5 are already assigned!

FI command		**00_BR_WLA1_4**
Line	**Column**	**Answer**
1	1	1
	2	2
	3	4
	4	6

Freeing Watch List Allocations: WFL

MWCX device group

Designation **WLF** **W**atch **L**ist **F**ree

Explanation Previously requested watch list allocations are freed again.

FI command Free ALL assigned watch list allocations for the selected device.

BR_WLF1 **(Single Read)**

Note: The FI command "WLF1" frees ALL assigned watch list allocations, including those of other WIN32 applications.

Response Structure The following table shows the general structure of the response to the FI command "WLF1".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. freed watch list allocation	Value range: 1..16
2 =	2. freed watch list allocation	Value range: 1..16
3 =	3. freed watch list allocation	Value range: 1..16

n = nth freed watch list allocation Value range: 1..16

Example WLF1

Free ALL assigned watch list allocations.

Assumption:
The following watch list numbers have been allocated: 1, 2, 3, 4.

FI command		00_BR_WLF1
Line	**Column**	**Answer**
1	1	1
	2	2
	3	3
	4	4

FI command

Free the required watch list allocations for a selected device.

BR_WLF2_(1)_{(2)..(10)} **(Single Read)**

(1)..(10) = List of watch list allocations to be released

A maximum of 10 watch list allocations can be transferred here to be freed again.

Response Structure

The following table shows the general structure of the response to the FI command "WLF2".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 = 1. freed watch list allocation Value range: 1..16

2 = 2. freed watch list allocation Value range: 1..16

3 = 3. freed watch list allocation Value range: 1..16

n = nth freed watch list allocation Value range: 1..16

Example WLF2

Free required watch list allocations:
Assumption: Watch list allocations 1,3,4, and 8 have first been requested using the FI command "WLA1".

FI command		00_BR_WLF2_1_3_4_8
Line	**Column**	**Answer**
1	1	1
	2	3
	3	4

Data of the Zero Offset Table: ZOD

MWCX device group

Designation **ZOD** **Z**ero **O**ffset **D**ata

Explanation

The zero-offset table data can be read and written. The zero offsets allow the origin of a coordinate axis to be shifted (offset) by a set value, related to the original position of the machine. The location of the machine zero point remains securely stored in the NC controls and is not changed by the zero offset.

Offset Type

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 as well as
- adjustable general offset in the zero (origin) table.

Using the zero offsets G50, G51 and G54 to G59 and the workpiece zero point (origin) G52, the coordinate zero point of every NC axis can be laid onto any coordinate position within or outside of the respective travelling range. It is thereby possible to process and identical NC program at different machine positions. The position of the machine zero point of every axis is entered in the drive parameters as a difference to the reference point, whereby the value entered in the drive parameters corresponds to the coordinate value of the reference point in the machine coordinate system.

Code of displacement types

Code	Meaning	Explanation
0	Total	Sum of all active offset values
1	G50/G51	Programmable absolute / incremental zero offset
3	General offset	acts additive to all offset types
4...9	G54 - G59	Selectable zero offsets

Zero point database

As memory for a record of zero offsets, 10 zero offset tables (O0 … O9) are provided.

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) = NC process number	[0...6]
(3) = Offset table number	[0...9]
(4) = Offset type	[offset type code]
(5) = Code of the axis meanings	[0...8] axis meanings [9] offset angle "PHI"

Value to be written

Offset	[with axes: format acc. to the parameter settings] [offset angle PHI always in format Y.XXXX]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Offset Type

Code	Meaning	Explanation
3	General offset	acts additive to all offset types
4	G54	Adjustable zero offset
....	....	....
9	G59	Adjustable zero offset

Note: The axis meanings are contained in chapter entitled “Data Tables”.

Response Structure

One line with one column is output to acknowledge the FI command issued. The meaning of the elements is as follows:

(P_ACK) = **P**ositive **ACK**nowledge Value has been written

Example ZOD

Write into zero offset table O2 the value of the general offset of axis X in NC memory A of NC process number 0 at device address 00.

Assumption:

- There is a valid parameter record 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
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

The values of the zero offset of all defined axes are output for the selected offset (shift) type.

CR_ZOD1_(1)_(2)_(3)_(4){_(6)} **(Single Read)**

CC_ZOD1_(1)_(2)_(3)_(4){_(6)} **(Cyclic Read)**

CB_ZOD1_(1)_(2)_(3)_(4){_(6)} **(Break Cyclic Read)**

(1) = NC memory [1 = memory A; 2 = memory B]

(2) = NC-Process number [0...6]

(3) = Offset table number [0...9]

(4) = Offset type [0...9 Code of offset type]

(5) = Measuring unit (optional) [mm, inch]

If there is no optional information for the unit {mm, inch}, then the length values are given in the base programming unit of the process. If the entered unit is different from the basic coordinate system, the length values are converted into the requested unit.

Note: The axis meanings are contained in chapter entitled “Data Tables”.

Offset Type

Code	Meaning	Explanation
0	Total	Sum of all active offset values
1	G50/G51	Programmable absolute / incremental zero offset
2	G52	Programmable work piece zero point
3	General offset	Acts additive to all offset types
4	G54	Adjustable zero offset
....	....	....
9	G59	Adjustable zero offset

Response Structure

The following table shows the general structure of the response to the FI command “ZOD1”. The answer consists of one to a maximum of n=10 lines (1 per axis), each with three columns for the name of the axis, value of zero offset and the unit.

Line 1...n:	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Axis designation [acc. to settings of the axis parameters; PHI] [Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] i=[1...3])

2 = Value format acc. to parameter settings] [offset angle PHI always in format Y.XXXX]

3 = Unit [mm, inch], [offset angle PHI: deg]

Example ZOD1

Read in the zero offset table O2 the values of the general offset of all defined axes in NC memory A of CNC process number 0 at device address 00. The values are to be output in the basic coordinate system.

Assumption:

- There is a valid parameter record in the device and
- the axes X, Y, Z (assigned at certain times) are defined.

FI command	00_CR_ZOD1_1_0_2_3		
Answer			
Line	**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]

FI command

Output all zero offset values for the axes selected in a list.

CR_ZOD2_(1)_(2)_(3)_(4)_(5){_(6)} **(Single Read)**

CC_ZOD2_(1)_(2)_(3)_(4)_(5){_(6)} **(Cyclic Read)**

CB_ZOD2_(1)_(2)_(3)_(4)_(5){_(6)} **(Break Cyclic Read)**

(1) = NC memory [1 = memory A; 2 = memory B]

(2) = NC process number [0...6]

(3) = Offset table number [0...9]

(4) = Offset type [offset type code]

(5) = Selection list for a max. of 10 elements [0...8] axis meanings [9] offset angle "PHI"

(6) = Measuring unit (optional) [mm, inch]

If there is no optional information for the unit {mm, inch}, then the length values are given in the base programming unit of the process. If the entered unit is different from the basic coordinate system, the length values are converted into the requested unit.

Offset Type

Code	Meaning	Explanation
0	Total	Sum of all active offset values
1	G50/G51	Programmable absolute / incremental zero offset
2	G52	Programmable work piece zero point
3	General offset	Acts additive to all offset types
4	G54	Adjustable zero offset
....	....	....
9	G59	Adjustable zero offset

Note: The axis meanings are contained in chapter 6.2, “Data Tables”.

Response Structure

The following table shows the general structure of the response to the FI command “ZOD2”. The answer consists of one to a maximum of n=10 lines (1 per requested axis), each with three columns for the code of the axis meaning, value of zero offset and the unit. The number of lines depends on the number of list elements.

Line 1...n:	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Axis designation [acc. to settings of the axis parameters; PHI] [Xi, Yi, Zi, Ui, Vi, Wi, Ai, Bi ,Ci, Si, --] i=[1...3])

2 = Value format acc. to parameter settings] [offset angle PHI always in format Y.XXXX]

3 = Unit [mm, inch], [offset angle PHI: deg]

Note: If a requested axis is not defined then the value of columns 1 to 3 is [--]. If the axis name is preceeded by "*", e.g. "*Z", then this access is only assigned to the process at certain times (GAX/FAX).

Example ZOD2

Read in zero offset table O2 the values of the general offset of axes X, Y, Z and U as well as the offset angle "PHI" in NC memory A of CNC process number 0 at device address 00.

Assumption:

- There is a valid parameter record in the device and
- the axes X, Y, Z (assigned at certain times) are defined.

FI command	**00_CR_ZOD2_1_0_2_3_0_1_2_3_9**		
Answer			
Line	**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]

Reference to Literature

See chapter entitled “Literature” [44].

Value Ranges

Basic Data

MWCX device group

Element No.	Name of the File Element		Writable?
1	Index address	0...9999999	No
2	Name (ID)	Max. 28 ASCII characters	Yes
3	Memory	M = magazine/turret, S = spindle, G = grabber	No
4	Location	0...999	No
5	Tool number	1...9999999	Yes
6	Duplo number	1...9999	Yes
7	Correction type	1...5	Yes
8	Number of tool edges	1...9	Yes
9	Tool status	32 status bits with 0/1 (see following table)	Yes
10	Unassigned half-location	0...4	Yes
11	Former tool location	Memory [M/S/G] location [0..999]	No
12	Memory of the next replacement tool	M = magazine/turret, S = spindle, G = grabber	No
13	Location of the next replacement tool	0...999	No
14	Memory of the previous replacement tool	M = magazine/turret, S = spindle, G = grabber	No
15	Location of the previous replacement tool	0..999	No
16	Time unit	0 = min, 1 = cycle	Yes
17	Unit of length	0 = mm, 1 = inch	Yes
18	Tool code	0...9	Yes
19	Display type	0...65535	Yes
20	User data 1	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
21	User data 2	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
22	User data 3	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
23	User data 4	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
24	User data 5	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
25	User data 6	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
26	User data 7	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
27	User data 8	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes
28	User data 9	+/- $1.2*10^{38}$ .. +/- $3.4*10^{-38}$	Yes

Note: Box 19 applies from FI version 06 of the tool classification. It can no longer be edited by the user. Refer also to the documentation “Tool Management”.

Tool status bits

Bit	Symbol	Value	Group name	Group information	Change able	Comment
1	!	1 0	Presence	Tool not available Tool available	No	Tool is missing
2	?	1 0		Tool not required Tool required	No	Tool not required for machining
3	t	1 0	Error correction type	Correction type faulty	No	Correction type does not comply with requirements
4	e	1 0	Error number of cutters	Wrong number of cutters Correct number	No	Number of tool edges does not comply with requirements
5	f	1 0	Error tool edge	Cutter faulty Cutter not faulty	No	Tool edge data does not comply with requirements
6	$	1 0	Error tool code	Tool code faulty Tool code not faulty	No	
7	*				No	Reserved
8	*				No	Reserved
9	B	1 0	Location locking	Location blocked Location not blocked	Yes	Location is damaged, for example
10		1 0		Upper half-location blocked. Not blocked	No	Blocked for fpc tool located in grabber or spindle
11		1 0		Lower half-location blocked. Not blocked	No	Blocked for fpc tool located in grabber or spindle
12		1 0	Location reservation	Upper half-location reserved. Not reserved	Yes	For a tool that is to be inserted, for example
13		1 0		Lower half-location reserved. Not reserved	Yes	For a tool that is to be inserted, for example
14		1 0	Location reservation	Upper half-location covered Not covered	No	The upper half-location is covered by a tool
15		1 0		Lower half-location covered Not covered	No	The lower half-location is covered by a tool
16		1 0		Location assigned Not assigned	No	There is a tool at this location
17	d	1 0	Wear state	Tool is worn Tool is not worn	No	The tool can no longer be used (replace)
18	w	1 0		Warning limit reached Warning limit not reached	No	The remaining tool life is near its end (replace)
19	p	1 0	Alternate tool identification	Processing tool No processing tool	No	There is a processing tool for every sister tool group

Bit	Symbol	Value	Group name	Group information	Change able	Comment
20	s	1 0		Replacement tool No replacement tool	No	A replacement tool is a tool still to be used, not a processing tool
21	C	1 0	Fixed position coding	Fixed position coding, tool No fixed position coding, tool	Yes	The tool always remains at the same location in the magazine
22	L	1 0	Tool status	Tool blocked Tool not blocked	Yes	E.g., cutter is broken by user or application
23	*				No	Reserved
24	*				No	Reserved
25	1	1 0	ANW 1	User tool status bit 1	Yes	Any meaning
26	2	1 0	ANW 2	User tool status bit 2	Yes	Any meaning
27	3	1 0	ANW 3	User tool status bit 3	Yes	Any meaning
28	4	1 0	ANW 4	User tool status bit 4	Yes	Any meaning
29	5	1 0	ANW 5	User tool status bit 5	Yes	Any meaning
30	6	1 0	ANW 6	User tool status bit 6	Yes	Any meaning
31	7	1 0	ANW 7	User tool status bit 7	Yes	Any meaning
32	8	1 0	ANW 8	User tool status bit 8	Yes	Any meaning

Tool edge data

Element Number	Name of the Data Element	Value Range	Writable?
1	Tool edge position	0...8	Yes
2	Tool edge status	16 status bits with 0/1 (see following table)	Yes
3	Remaining tool life	-99.99...100.00	Yes
4	Warning limit	0.1...100.00	Yes
5	Max. life time	0...9999999	Yes
6	Time used	0...9999.999	No
7	Length L1	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
8	Length L2	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
9	Length L2	-99999.9999...+99999.9999 or -9999.99999..+9999.99999	Yes
10	Radius R	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
11	Wear L1	-99999.9999...+99999.9999 or -9999.99999..+9999.99999	Yes
12	Wear L2	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
13	Wear L3	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
14	Wear R	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
15	Offset L1	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
16	Offset L2	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
17	Offset L3	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes

Element Number	Name of the Data Element	Value Range	Writable?
18	Offset R	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
19	L1_min	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
20	L1_max	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
21	L2_min	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
22	L2_max	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
23	L3_min	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
24	L3_max	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
25	R_min	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
26	R_max	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	No
27	Wear factor L1	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
28	Wear factor L2	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
29	Wear factor L3	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
30	Wear factor R	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
31	User data 1	+/- $1.2*10^{38}$ ... +/- $3.4*10^{-38}$	Yes
32	User data 2	+/- $1.2*10^{38}$ ... +/- $3.4*10^{-38}$	Yes
33	User data 3	+/- $1.2*10^{38}$ ... +/- $3.4*10^{-38}$	Yes
34	User data 4	+/- $1.2*10^{38}$ ... +/- $3.4*10^{-38}$	Yes
35	User data 5	+/- $1.2*10^{38}$ ... +/- $3.4*10^{-38}$	Yes
36	User data 6	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
37	User data 7	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
38	User data 8	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
39	User data 9	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes
40	User data 10	-99999.9999...+99999.9999 or -9999.99999...+9999.99999	Yes

Cutter Status Bits

Bit	Symbol	Value	Group name	Group information	Change able	Comment
1	e	1 0	Wrongcutting edge position	Wrong cutter position Correct position	No	
2	1	1 0	L1 incorrect	L1 faulty Not faulty	No	
3	2	1 0	L2 incorrect	L2 faulty Not faulty	No	
4	3	1 0	L3 incorrect	L3 faulty Not faulty	No	
5	r	1 0	R incorrect	R faulty Not faulty	No	
6	*				No	Reserved
7	*				No	Reserved
8	*				No	Reserved
9	d	1 0	Wear condition	Cutter worn Cutter not worn	No	The cutter can no longer be used (replace)
10	w	1 0		Warning limit reached Warning limit not reached	No	The remaining life time is going to expire (replace).
11	*				No	Reserved

Bit	Symbol	Value	Group name	Group information	Change able	Comment
12	*				No	Reserved
13	A	1 0	ANW 1	User cutter status bit 1	Yes	Any meaning
14	B	1 0	ANW 2	User cutter status bit 2	Yes	Any meaning
15	C	1 0	ANW 3	User cutter status bit 3	Yes	Any meaning
16	D	1 0	ANW 4	User cutter status bit 4	Yes	Any meaning

Flow Diagram for Command Groups

Handling Tool Data Records: TDA, TRM

MWCX device group

The following diagram shows by way of an example the sequence (flow) required for editing complete tool data records.

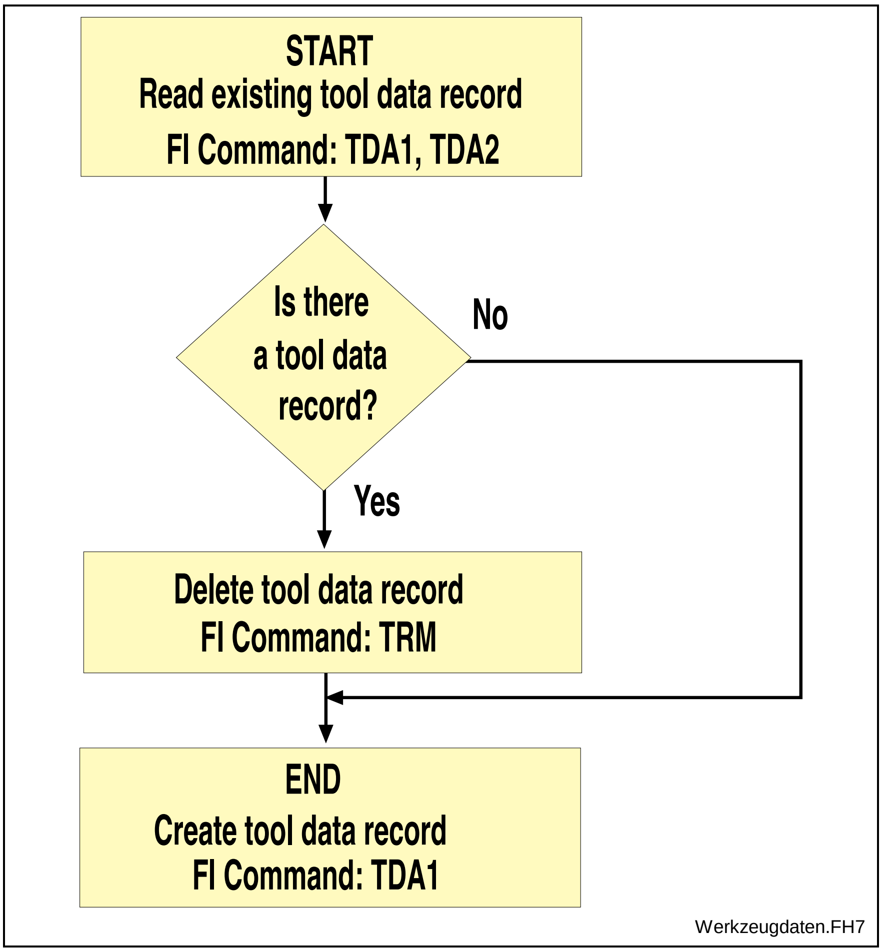


Fig. 7-9: Structure for handling tool data records

Using the Tool Command in Practice

In addition to exact knowledge of the individual commands, the multitude of tool commands also requires further information for practical usage.

This chapter therefore deals with this subject from the point of view of the user.

Fundamentals when Replacing a Tool

The control unit supports two different strategies:

i. The tool is transported to its previous location after use. The location remains reserved for the tool.

ii. The tool is transported to another, unassigned location after use. Only the control unit knows which tool is actually located where.

Point two is significant when a machine is equipped with a replacement grabber that fetches the tool from its tool location before it is actually used and then queues it. In some circumstances after use the old tool location may already be occupied by a tool that has previously been put down and therefore the next free location must be allocated.

Reading Tool Data

Note: Only the values from the tool database are read. No recognition is made of the tool that is physically inserted.

CR_TLD Returns data elements of a tool of the basic data or cutter data from the tool memory.

Note: No additional command required.

BR_TLB Returns one or more elements of the basic tool data of several tools from the tool memory.

Note: No additional command required.

BR_TLE Returns one or more elements of the tool cutter data of several tools from the tool memory.

Note: No additional command required.

CR_TDR Returns a complete basic data record or cutter data record of a tool in the tool memory.

Note: No additional command required.

BR_TDA Returns a complete tool data record consisting of the basic data and cutter data of a tool in the tool memory.

Note: No additional command required.

Block Tool Location

CR_TII The specified tool location is temporarily blocked from automatic assignment by the control unit.

Precondition: The tool location must be free (unassigned).

Note: No additional command required.

Release Tool Location

CR_TIF The indicated tool location is released after a temporary block.

Note: No additional command required.

Remove Tool

Note: Removing a tool means deleting the tool from the tool database. The actual tool itself must be removed previously by the user.

CR_TRM The tool data at this tool location is deleted from the database.

Note: No additional command required.

Modifying a Tool

Note: Only the tool data record in the tool memory is modified. The actual tool itself is not affected.

CW_TLD Writes a single element of the basic tool data or cutter data in the tool memory.

Note: No additional command required.

Replacing a Tool of the same Type

Note: Inserting a tool should be understood as an updating of the tool database. The tool itself must have been previously inserted by the user at its location.

CW_TLD Writes a single element of the basic tool data or cutter data in the tool memory.

Note: This command possibly requires repeated calling up when a tool of the same type is to be replaced.

Replacing a Tool of a different Type

Note: Inserting a tool should be understood as an updating of the tool database. The tool itself must have been previously inserted by the user at its location.

BW_TDA Writes a complete tool data record in the tool memory in a single access.

Note: This command must be carried out in the following order:

- CR_TRM Remove old tool.
- BW_TDA Write complete new tool data record.

Note: CR_TII and CR_TIF are already implemented in this command.

Moving a Tool

Note: Moving a tool should be understood as an updating of the tool database. The tool itself must have been previously inserted by the user at its new location.

CR_TMV A complete tool data record consisting of basic data and cutter data is moved.

Precondition: The target location must be free (unassigned).

Note: No additional command required.

Read Active Tool Number

CR_ATN The number of the active tool is read out.

Note: No additional command required.

Read Active Cutter Number

CR_AEN The number of the active cutter is read out.

Note: No additional command required.

Read Long Identification

CR_DIS4 The directory entry of the valid tool list is read out. It is updated after every download by CW_TDF.

Note: No additional command required.

Set Remaining Tool Life to 100%

CR_TRS The remaining tool life of a tool as a percentage is set to 100%.

Note: No additional command required.

Initiate Download

CW_TDI The control unit is prepared for the download of tool data.

Note: No additional command required.

Downloading Tool Data

CW_TDD The tool data for one or more tools is downloaded.

Note: This command must be carried out in the following order:

- CW_TDI Initiate download
- CW_TDD Write complete basic or cutting edge record data By means of repeated CW_TDD, all basic and cutting edge data of all tool of a tool magazine can be written (download).
- CW_TDF End download. the tool magazine is once more released

End Download.

CW_TDF Download of tool data is completed.

Note: No additional command required.

7.4 FI Commands for the MSCX Device Group

The FI Commands described in this chapter are valid for the MSCX device group. The device types of this device group are listed in the following table:

Group	Device Type	Address
MSCX	SERCANS-A, SERCANS-P	[00]

Note: Please note that the device address must be set before the respective FI command, e.g., 00_BR_SPA1_3_S-0-0003_48 (refer also here to Chapter 6.1 “Elements of the FI Command”).

Determining the Actual (Current) System Error: ASE

MSCX Device Group

Designation **ASE** **A**ctual **S**ystem **E**rror

Explanation The current system error is read out, whereby the response 0x0000 indicates that the SERCANS card is functioning correctly.

FI command

CR_ASE **(Single Read)**

CC_ASE **(Cyclic Read)**

CB_ASE **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the FI command ASE. In line 1, column 4, the number of the drive is output that reports the current system error. Not all current system errors can be directly allocated to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line 1	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = 0x0000

2 = 0x0000

3 = Actual (current) system error

4 = Drive No.

Example ASE Reading the current system error returns LWL ring interrupted.

FI command		00_CR_ASE
Line	**Column**	**Answer**
1	1	0x0000
	2	0x0000
	3	0x8009
	4	0x0000

Reference to Literature See chapter entitled “Literature” [42].

Deleting the Actual (Current) System Error: CSE

MSCX Device Group

Designation **CSE** **C**lear **S**ystem **E**rror

Explanation An error reported by the SERCANS card is deleted.

FI command **CW_CSE** **(Single Write)**

Value to be written: The contents of the value parameter is not evaluated.

Response Structure The following table shows the general structure of the response to the FI command "CSE". In line 1, column 4, the number of the drive is output that reports the current system error. Not all current system errors can be directly allocated to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line 1	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = 0x0000

2 = 0x0000

3 = Actual (current) system error

4 = Drive No.

Example CSE Deleting the actual (current) system error:

FI command		00_CW_CSE
Line	**Column**	**Answer**
1	1	0x0000
	2	0x0000
	3	0x0000
	4	0x0000

Reference to Literature see chapter entitled “Literature” [45].

Setting the Communication Timeout Time DCT

MSCX Device Group

Designation **DCT** **D**evice **C**ommunication **T**imeout

Explanation By means of this command, the timeout time for the selected device is set dynamically (timeout time in ms).

FI command **BW_DCT1_(1)** **(Single Write)**

(1) = requested timeout time in ms

Response Structure The response to the "DCT1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCT1 For the device 00, the timeout time is set 1500 ms.

FI command		00_BW_DCT1_1500
Line	Column	Answer
1	1	(P_ACK)

FI command With this command, the timeout time for the selected device can be reset to default value.

BW_DCT2 **(Single Write)**

Response Structure The response to the "DCT2" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK) (P_ACK)

Example DCP2 For the device 00, the timeout time is reset to the default value.

FI command		00_BW_DCT2
Line	Column	Answer
1	1	(P_ACK)

Reading the Device Status Information: DSI

MSCX Device Group

Designation **DSI** **D**evice **S**tatus **I**nformation

Explanation This allows the most important device status information to be read out. The following information is returned:

Type of information	Status	Statement
System error information		
Mechanism error information		
Machine key information	valid	Yes/No
Machine key information		
Machine status information		
Sercans information		
Parameter download	running	Yes/No
PLC download	running	Yes/No
Firmware download	running	Yes/No
Offline/Online information	Communication?	Yes/No
Device simulation	switched on	Yes/No
Device status information		ON/ OFF

FI command Read out device status information for ALL defined devices.

BR_DSI1 **(Single Read)**

BC_DSI1 **(Cyclic Read)**

BB_DSI1 **(Break Cyclic Read)**

Note: The "DSI1" FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see example DSI1). The FI device polling mechanism **MUST** be switched on (see system configurator)!

Response Structure

The following table shows the general structure of the response to the "DSI1" FI command.

Line 1...n	Column 1	...	Column 11

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Mechanism error information	[0 = there is no mechanism error 0 = there is a mechanism error
4 =	Machine key information	[4 byte in HEX coding]
5 =	Is machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = Device status=OFF 1 = Device status=ON]

Example DSI1

Read the current device status information.
Assumption:
The following devices addresses are defined:

Device address 01 (MWCX device)

Device address 03 (MWSX device)

FI command		**01_BR_DSI1**
Line	**Column**	**Answer**
1	1	01
	2	0
	3	0
	4	00000000
	6	00000000
	5	0
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

FI command		01_BR_DSI1
Line	**Column**	**Answer**
2	1	03
	2	1
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

FI command

Read out device status information for a selected device.

BR_DSI2 **(Single Read)**

BC_DSI2 **(Cyclic Read)**

BB_DSI2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the "DSI2" FI command.

Line 1...n	Column 1	...	Column 11

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Mechanism error information	[0 = there is no mechanism error 1 = there is a mechanism error]
4 =	Machine key information	[4 byte in HEX coding]
5 =	Machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = Device status=OFF 1 = Device status=ON]

Example DSI2 Read the current device status information for the selected device.

FI command		00_BR_DSI2
Line	**Column**	**Answer**
1	1	00
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

Device Type and Accompanying Components: DTY

MSCX Device Group

Designation **DTY** **D**evice **TY**pe

Explanation The device type and the accompanying components of the selected device address are output.

FI command **BR_DTY1** **(Single Read)**

Response Structure The following table shows the general structure of the response to the “DTY1” FI command. A line with three columns for the device type is output as well as the names of the first device component and the name of the second device component.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Device Type (see chapter entitled “Elements of the FI Command”, and “Identifier”)

2 = Component type1 IND_DEV.INI entry: Component type1=

3 = Component type 2 IND_DEV.INI entry: Component type2=

Example DTY1 Output the device type and the accompanying components of device address 00.

FI command	00_BR_DTY1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	SERCANS-A	NONE	NONE

Read System Messages: MSG

MSCX Device Group

Designation **MSG** **M**e**S**sa**G**e

Explanation Reading of system messages

FI command Message

CC_MSG_(1) **(Cyclic Read)**

(1) = SYS-Message number

Note: Exists only as a cyclic command

Response Structure The response of the FI command 'MSG' consists of the system message data.

Example MSG 00_CC_MSG_64 (64 = MSG_SYSERRGEN)

FI command		**00_CC_MSG_64/3**
Line	**Column**	**Answer**
1	1	00

Restriction The following system messages:

SYS Message	SYS message numbers
MSG_PCLUPDBEG	52
MSG_PARUPDBEG	24
MSG_FWAUPDBEG	82

These commands cannot be used with the following programs:

Indramat OPC server

Indramat DDE server

Generating Physical Directory Names: PHD

MSCX Device Group

Designation **PHD** **PH**ysical **D**irectory

Explanation Generates physical directory names according to the BDI data written.

Note:
This is based on BDI philosophy.

FI command Generate physical directory names.

BR_PHD1_(1)_(2)_(3)_(4)_(5)_(6)	**(Single Write)**
(1) = Project ID	[-1= PROJECT_NEUTRAL -2= PROJECT_DEFAULT]
(2) = Section ID	[0= SECT_NEUTRAL 1= SECT_BIN 2= SECT_BASIC_DATA 3=SECT_OEM_DATA 4=SECT_CUSTOM_DATA 5=SECT_PROG_DATA]
(3) = Device address	[-1= DEVADDR_NEUTRAL, otherwise the required device address]

(4) = Process ID [-1= PROCESS_NEUTRAL, otherwise the required process number]

(5) =Data type ID [possible write values see BDI documentation (BDI_DEFINITIONS.H)]

(6) = Language ID [possible write values see BDI documentation (WINNT.H)]

Response Structure

The following table shows the general structure of the response to the FI command "PHD1".

Line 1	Column 1

Value Range/Meaning of Columns

1 = Physical directory name [complete physical directory name in accordance with the BDI data written]

Example PHD1

Requesting the physical directory name for:

PROJECT_NEUTRAL
SECT_BIN
DEVADDR_NEUTRAL
PROCESS_NEUTRAL
DATATYPE_NEUTRAL
LANG_NEUTRAL

FI command		**XX_BR_PHD1_-1_0_-1_-1_0_0**
Line	**Column**	**Answer**
1	1	D:\Programme\Indramat\Mtgui\Bin

Set the Device Status Information: SDS

MSCX Device Group

Designation **SDS** **S**et **D**evice **S**tatus

Explanation

By this command, the device status information can be set; here, the configuration file IND_DEV.INI is adjusted as well.

Note: When this command is transmitted, the following system messages are generated:
MSG_DEVICEOFF or MSG_DEVICE_ON !

FI command

With this command, the device status information of **ALL** defined devices can be set.

BW_SDS1_(1) **(Single Write)**

(1) = Device status information to be set
0 = Device status information OFF
1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS1" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example: SDS1

Set device status information to OFF for **ALL** defined devices.

FI command		00_BW_SDS1_0
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

With this command, the device status information for a selected device can be set.

BW_SDS2_(1)	**(Single Write)**
Device status information to be set	0 = Device status information OFF 1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS2" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example: SDS2

Set device status information to OFF for the selected device 00.

FI command		00_BW_SDS2_0
Line	**Column**	**Answer**
1	1	(P_ACK)

Software Installation Data: SID

MSCX Device Group

Designation

SID **S**oftware **I**nstallation **D**ata

Explanation

Information is returned regarding installation. This information includes installation paths, the software version used, DLL mode, plus service pack and release information.

FI command

Read-in the installation data.

BR_SID1 **(Single Read)**

BC_SID1 **(Cyclic Read)**

Response Structure

One line with 8 columns is output for the returned values.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Basic directory	[EXE files of the DOS-BOF]
2 = FI installation directory	[FI directory]
3 = Data directory	[in accordance with DOS-BOF]
4 = GBO version	[from INDRAMAT.ini]
5 = IF-DLL mode	[from INDRAMAT.ini]
6 = IF version	[from INDRAMAT.ini from DLL mode 400]
7 = Service pack info	[from INDRAMAT.ini from DLL mode 420]
8 = Release info	[from INDRAMAT.ini from DLL mode 420]

Example SID1 Return information on the current installation.

FI command		00_BR_SID1
Line	Column	Answer
1	1	--
	2	D:\Programme\Indramat\MTGUI\Bin
	3	--
	4	005-22Vxx
	5	07.00
	6	07V00
	7	--
	8	--

Note: Refer to FI command “PHD” for working with absolute paths.

SERCOS Parameters: SPA

MSCX Device Group

Designation **SPA** **S**ERCOS **PA**rameter

Explanation One SERCOS parameter of a drive or a SERCANS parameter is read out or is written. Each parameter consists of 7 elements, whereby 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 the format: X-Y-ZZZZ

(3) = Element coding [standard or advanced format]

Parameter No.

Format X-Y-ZZZZ	Value Range
X	S = standard data P = product data Y = SERCANS parameter
Y	[0..00.15] = parameter record
Z	[0...4095] = data block no.

Element Coding Element coding in standard format allows individual elements, such as the operating date, to be requested. If several elements are to be read out in one request, then the element coding can be OR’d in advanced format, e.g. operating date (0x40) and unit (0x08) produces OR’d (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	The marked section is then printed out.	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
Operating date	D	40H	(see Displaying the Operating Date	
Operating date, when no list		80H		

Displaying the Operating Date

The display of the operating date depends on the parameter number requested.

Decimal

Decimal values are given as floating points, e.g. 1.5. Leading spaces, zeros, plus and minus signs as well as trailing spaces are allowed.

Hexadecimal

Hexadecimal values are displayed by "0x...", e.g. 0x80. Up to a maximum of eight positions are allowed. Leading or trailing spaces are allowed. Leading additional zeros or plus and minus signs are not allowed.

Binary (max. 32 characters)

Leading or trailing spaces are allowed. The decimal point serves as separator:

e.g., 1111.0000.1010.1100.1111.0000.1010.1100

Note: Leading additional zeros or plus and minus signs are not allowed.

ID number

The following table shows the general way in which the ID number is displayed:

Format X-Y-ZZZZ	Value Range
X	S = standard data P = product data
Y	[0..0.7] = parameter record
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 maximum length of the list. The length specification refers to the length of the list in the drive and therefore designates the number of bytes for storage (storage bytes). The number of elements in the list can be calculated using the attribute. The list elements are displayed according to the attribute. All parts of the list are separated from each other 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 variable length lists. The individual string characters are not separated by a line feed. When displaying the lists, a distinction is made between standard format and advanced format. In standard format, only the character string is returned, whereas in advanced format the actual length and the maximum length of the list (string) is also transmitted.

Example:

Parameter S-0-0030, operation date
Standard format: "DKC2.1-SSE-01V09"
Advanced format: "16\n16\nDKC2.1-SSE-01V09"

Note: When requesting SERCANS parameters the drive address can be anywhere within the range [0..254].

Response Structure

The following table shows the general structure of the response to the FI command “SPA1”. Line 1 is output both when reading and when writing. Additional lines are only output when reading depending on the element coding.

Note: If the element coding has been requested in standard format then the first line is not applicable.

Line 1 is a status line that either contains SERCOS / SERCANS errors or displays the successful processing of the FI command. If the command has been processed successfully, then columns 1 and 3 contain the value [0x0000].

In the first line, column 2 or column 4, the number of the drive is output that reports the SERCOS error or the global SERCANS error. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line	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: Element corresponding to the element coding.			
...	...			
n	Read: (n-1). Element corresponding to the element coding.			

Example SPA1 / read

Read parameter S-0-0003 of the 3rd drive (element coding 0x48)

FI command	00_BR_SPA1_3_S-0-0003_48			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0000	0x0000	0x0000
2	µs			
3	2000			

Example SPA1 / write

Write the ID number P-0-0037 in parameter S-0-0305 of the 3rd drive (element coding 0x40).

Technical background:

- Realtime status bit 1 is to 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			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	0x0000	0x0003	0x0000	0x0000

Reference to Literature See chapter entitled “Literature” [41].

See chapter entitled “Literature” [46].

Active SERCOS Phase Switch-Over: SPH

MSCX Device Group

Designation **SPH** **S**ERCOS **PH**ase

Explanation All drives within a SERCOS ring are in the same communication phase. The phase status can be read-out or changed 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 phase conditions allowed are shown by the numbers [0...4].

Response Structure The following table shows the general structure of the response to the FI command "SPH". In the first line, column 2 or column 4, the number of the drive is output that reports the SERCOS error or the global SERCANS error. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line	Column 1	Column 2	Column 3	Column 4
1	<SERCOS error>	<Drive No. SERCOS error>	<Global SERCANS error>	<Drive no. that has caused the global SERCANS error>
2	Read: current phase Write: previously phase			

Example SPH Switch-over (write) of the SERCANS control after phase 4; phase 2 is active.

FI command	00_CW_SPH Value to be written: 4			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	0x0000	0x0000	0x0000	0x0000
2	2			

Reference to Literature See chapter entitled “Literature” [42].

7.5 FI Commands for the MWMX and MWSX Device Group

The FI Commands described in this chapter are valid for the MWMX and MWSX device group. In this device group, the following types are listed as well as possible device addresses:

Group	Accompanying Types	Address
MWMX	VMISP200-P-G2, VMISP200-R-G2	[00...63]
MWSX	ISP200-P-G2, ISP200-R-G2	[00...63]

Note: The Visual Motion component has been realized under SCP (Scalable Communication Platform).

Please note that the device address must be set before the respective FI command, e.g. 00_BR_ASM1 (refer also here to the chapter entitled “Elements of the FI Command”).

Active System Error Messages: ASM

MWMX and MWSX device groups

Designation **ASM** **A**ctive **S**ystem **M**essages

Explanation The active system error messages that affect the functioning of the entire electrical device are output. Depending on the FI command, the device address, device name, message number, type of message, short text and reference text are all output.

FI command Output the current system error messages pending of all active devices from the MWSX device group.

BR_ASM1 **(Single Read)**

BC_ASM1 **(Cyclic Read)**

BB_ASM1 **(Break Cyclic Read)**

Note: The “ASM1” FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see example “ASM1”).

Response Structure The following table shows the general structure of the response to the FI command “ASM1”. The number of lines (1 .. n=15) depends on the number of defined devices. Each line consists of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see ASM5)

Example ASM1

Read the current system error messages of all defined devices of the MWSX device group.

Assumption: The following three devices are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		07_BR_ASM1
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	07
	2	Drill station 1
	3	74
	4	F
	5	SLM time monitoring
	6	X
	7	0
3	1	10
	2	Drill station 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0

FI command

Output the currently pending system error message of the selected device from the MWSX device group.

BR_ASM2 **(Single Read)**

BC_ASM2 **(Cyclic Read)**

BB_ASM2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the FI command “ASM2”. The answer consists of a line of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]

5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information for the message number	is required to resolve the information "@" (see ASM5)

Example ASM2

Read the current system error 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
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0

FI command

Output the current system error messages of the device listed from the MWSX device group.

BR_ASM3_(1) **(Single Read)**

BC_ASM3_(1) **(Cyclic Read)**

BB_ASM3_(1) **(Break Cyclic Read)**

(1) = Selection list for a max. of 10 MWSX devices [00_01_02_ ... _15]

Response Structure

The following table shows the general structure of the response to the FI command "ASM3". The number of lines (1 .. n=15) depends on the number of listed MWSX devices. Each line consists of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information for the message number	is required to resolve the information "@" (see ASM5)

Example ASM3

Read the current system error messages of the selected MWSX devices.

Assumption:
The following devices addresses are defined:

Device address 01,

- Device address 07 and
- Device address 10.

FI command		01_BR_ASM3_01_10
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC - battery voltage too low
	6	X
	7	0
2	1	10
	2	Drill center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0

FI command

Output the current system error messages of all defined devices (in accordance with the system configuration) from the MWSX device group.

BR_ASM4_(1) **(Single Read)**

BC_ASM4_(1) **(Cyclic Read)**

BB_ASM4_(1) **(Break Cyclic Read)**

(1) = Device group [MWSX]

Response Structure

The following table shows the general structure of the response to the FI command “ASM4”. The number of lines (1 .. n=15) depends on the number of defined MWSX devices. Each line consists of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an reference text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 byte additional information for the message number	is required to resolve the information “@” (see ASM5)

Example ASM4

Read the current system error messages of all defined devices of the MWSX device group.

Assumption:
The following devices are defined:

- Device address 01 and
- Device address 10.

FI command		01_BR_ASM4_MWSX
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
2	1	10
	2	Drill center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0

FI command

Output the reference text for the currently pending error message, related to the device and the message number.

BR_ASM5_(1)_(2) **(Single Read)**

(1) = Message number [0...150]

(2) = 2 bytes of additional information for the message number

Response Structure

The following table shows the general structure of the response to the FI command "ASM5". The answer consists of a line with 5 columns for the device address, device name, message number and reference text.

Line 1...n	Column 1	...	Column 5

Value Range/Meaning of Columns

1 =	device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
6 =	Reference text	[max. 14 lines with a max. 78 characters/line]

Example ASM5 Read the reference text relating to the system error with message number 74 of device address 01.

FI command		**01_BR_ASM5_74_0**
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	74
	4	F
	5	Replace the SLM module on the PLC card (slot 3).

Reference to Literature See chapter entitled “Literature” [13].

Trigger Control Reset: CRT

MWMX and MWSX device groups

Designation **CRT** **Control-Reset**

Explanation The control reset allows the selected device to be reset after a system error. If there is no system error at the selected device then the job is ignored.

Note: Carrying out a reset completely re-initializes the device.
During initialization, communication is temporarily interrupted (inherent to design).

FI command **CW_CRT** **(Single Write)**

Value to be written Trigger reset 0

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure The return value of the "DataTransfer" routine is [0] if the write procedure has been successfully completed. In the event of an error, more information in the form of a general error result line can be requested by the routine "ReadGroupItem" (refer here to chapter 8, “Error Codes” and “General Error Result Line”).

Example CRT Trigger a control reset on the selected device.

FI command	**00_CW_CRT**
Value to be written	0

Reference to Literature See chapter entitled “Literature” [26].

Setting the Communication Timeout Time DCT

MWMX and MWSX device groups

Designation **DCT** **D**evice **C**ommunication **T**imeout

Explanation By means of this command, the timeout time for the selected device is set dynamically (timeout time in ms).

FI command **BW_DCT1_(1)** **(Single Write)**

(1) = requested timeout time in ms

Response Structure The response to the "DCT1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK) (P_ACK)

Example DCT1 For the device 00, the timeout time is set 1500 ms.

FI command		00_BW_DCT1_1500
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command With this command, the timeout time for the selected device can be reset to default value.

BW_DCT2 **(Single Write)**

Response Structure The response to the "DCT2" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns 1 = Status message (P_ACK) (P_ACK)

Example DCP2 For the device 00, the timeout time is reset to the default value.

FI command		00_BW_DCT2
Line	**Column**	**Answer**
1	1	(P_ACK)

Long ID of PLC Data Block: DIS

MWMX and MWSX device groups

Designation **DIS** **D**ata **I**dentification **S**tring

Explanation Reads the long ID (directory entries) of the PLC program. Included in the directory entries are the number of the entry in the directory, the name, length and date and time of creation and/or details of the last time the respective data record was changed.

FI command
BR_DIS2 **(Single Read)**
BC_DIS2 **(Cyclic Read)**
BB_DIS2 **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the "DIS2" FI command. The response consists of a line with six columns.

Line 1	Column 1	...	Column 6

Value Range/Meaning of Columns

1 =	Number in PLC directory	[01...99]
2 =	Name of the PLC program	[max. 8 ASCII characters]
3 =	Length of the PLC program	[byte]
4 =	Date of creation/last change to PLC program	[DD.MM.YY]
5 =	Time of creation/last change to the PLC program	[HH:MM:SS]
6 =	Date of creation/last change to PLC program	[DD.MM.YYYY]

Note: If there is no valid NC package in the selected NC memory then all columns contain [--] .

Example DIS2

Read the directory entries of the PLC program at address 00.
<u>Assumption:</u>
There is a valid PLC program in the selected device.

FI command		00_BR_DIS2
Line	**Column**	**Answer**
1	1	01
	2	KEY1
	3	20018
	4	10.05.99
	5	12:42:00
	6	10.05.1999

Reference to Literature

See chapter entitled “Literature” [30].

Reading the Device Status Information: DSI

MWMX and MWSX device groups

Designation **DSI** **D**evice **S**tatus **I**nformation

Explanation

This allows the most important device status information to be read out. The following information is returned:

Type of information	Status	Statement
System error information		
Mechanism error information		
Machine key information	valid	Yes/No
Machine key information		
Machine status information		
Sercans information		
Parameter download	running	Yes/No
PLC download	running	Yes/No
Firmware download	running	Yes/No
Offline/Online information		
Device simulation	switched on	Yes/No
Device status information		ON/OFF

FI command

Read out device status information for ALL defined devices.

BR_DSI1	**(Single Read)**
BC_DSI1	**(Cyclic Read)**
BB_DSI1	**(Break Cyclic Read)**

Note: The "DSI1" FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see example DSI1). The FI device polling mechanism **MUST** be switched on (see system configurator)!

Response Structure

The following table shows the general structure of the response to the "DSI1" FI command.

Line 1...n	Column 1	...	Column 11

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Mechanism error information	[0 = there is no mechanism error 0 = there is a mechanism error
4 =	Machine key information	[4 byte in HEX coding]
5 =	Machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = Device-Status=OFF 1 = Device-Status=ON]

Rexroth Indramat

Example DSI1

Read the current device status information.

Assumption:
The following devices addresses are defined:

- Device address 01 (MWCX device)
- Device address 03 (MWSX device)

FI command		01_BR_DSI1
Line	**Column**	**Answer**
1	1	01
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
2	1	03
	2	1
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1

FI command

Read out device status information for a selected device.

BR_DSI2 **(Single Read)**

BC_DSI2 **(Cyclic Read)**

BB_DSI2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the "DSI2" FI command.

Line 1...n	Column 1	...	Column 11

Value Range/Meaning of Columns

1 =	device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Mechanism error information	[0 = there is no mechanism error 1 = there is a mechanism error]
4 =	Machine key information	[4 byte in HEX coding]

5 =	Is machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]

Example DSI2 Read the current device status information for the selected device.

FI command		00_BR_DSI2
Line	**Column**	**Answer**
1	1	00
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1

Device Type and Accompanying Components: DTY

MWMX and MWSX device groups

Designation **DTY** **D**evice **TY**pe

Explanation The device type and the accompanying components of the selected device address are output.

FI command **BR_DTY1** **(Single Read)**

Response Structure The following table shows the general structure of the response to the "DTY1" FI command. A line with three columns is output for the device type, as well as the names of the first device component and the name of the second device component.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 =	Device Type	(see Chapter 6.1 "Elements of the FI Command" and "Identifier")
2 =	Component type1	IND_DEV.INI-Entry: Componenttype1=
3 =	Component type 2	IND_DEV.INI-Entry: Componenttype2=

Example DTY1

Output the device type and the accompanying components of device address 00.

FI command	00_BR_DTY1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	ISP200-P	MTS-P	NONE

Diagnosis Window Data: DWD

MWMX and MWSX device groups

Designation

DWD **D**iagnosis **W**indow **D**ata

Explanation

Diagnostic messages are output. The data is edited in such a way that they can be output directly in the diagnosis overview, i.e., where applicable, different types of diagnosis, such as ProVi and a process report, are returned simultaneously.

FI command

Output all diagnostic messages.

BR_DWD1_(1){_(2)} **(Single Read)**

BC_DWD1_(1){_(2)} **(Cyclic Read)**

(1) = Type of diagnosis window [1 = CNC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start preconditions, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Output first diagnostic messages.

BR_DWD2_(1){_(2)} **(Single Read)**

BC_DWD2_(1){_(2)} **(Cyclic Read)**

(1) = Type of diagnosis window [1 = CNC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start preconditions, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Response Structure

The following table shows the general structure of the "DWD1" and "DWD2" FI commands. The number of lines depends on the number of messages pending. Different columns are valid according to the type of diagnosis.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 12

Meaning of the Columns

1 =	Message text	[ASCII characters]
2 =	Time stamp day	[mm.dd.yyyy]
3 =	Time stamp hour	[hh:mm:ss]
4 =	Reference text available	[YES, NO]
5 =	Type of diagnosis	[1 = ProVi, 2 = SFC, 3 = MTC-NC, 4 = MTA-NC]
6 =	Message number	[ASCII characters]
7 =	Message ID	[ASCII characters] (DWORD, decimal) (ProVi)
8 =	Mechanism number	[0..31] (MTC-NC) [0] (MTA-NC)
9 =	2 byte additional information	[ASCII characters] (MTC NC)
10 =	Message group	[1...9999] (MTA-NC)

11 =	SFC entity name	[ASCII characters]
12 =	NC note	[ASCII characters] (MTC NC)
13 =	Analysis of criteria available	[YES, NO] (ProVi, SFC)
14 =	Message HTML file	[ASCII characters] (ProVi, MTC-NC)

Example DWD1 All diagnostic messages from module 3 in control unit 0.

There are two messages.

FI command		**00_BR_DWD1_4_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	01.27.2000
	3	14:56:32
	4	YES
	5	1
	6	34
	7	43923028
	8	
	9	
	10	
	11	
	12	
	13	YES
	14	
2	1	Station waiting until tool-change command has ended.
	2	01.27.2000
	3	15:03:10
	4	YES
	5	3
	6	79
	7	
	8	1
	9	0
	10	
	11	
	12	
	13	NO
	14	

Example DWD2 First diagnostic message from module 3 in control unit 0.

There are two messages.

FI command		00_BR_DWD2_4_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	01.27.2000
	3	14:56:32
	4	YES
	5	1
	6	34
	7	43923028
	8	
	9	
	10	
	11	
	12	
	13	YES
	14	

Reference to Literature See chapter entitled “Literature” [13].

Existing errors: EDE

MWMX and MWSX device groups

Designation **EDE** **E**xisting **D**iagnosis **E**rror

Explanation Whether or not errors exist in a control unit or in a module is queried. These can be step chain errors, NC errors, MTA200 errors or ProVi errors.

FI command Query whether there are errors in this control unit.

BR_EDE1 **(Single Read)**

BC_EDE1 **(Cyclic Read)**

Response Structure The following table shows the general structure of the "EDE1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Error exists [YES, NO]

Example EDE1 Do errors exist in control unit 0?

FI command		00_BR_EDE1
Line	**Column**	**Answer**
1	1	YES

FI command Query whether or not errors exist in a specific module.

BR_EDE2_(1) **(Single Read)**

BC_EDE2_(1) **(Cyclic Read)**

(1) = Module number [1...99]

Response Structure

The following table shows the general structure of the "EDE2" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Error exists [YES, NO]

Example EDE2

Do errors exist in module 1 on control unit 0?

FI command		00_BR_EDE2_2
Line	**Column**	**Answer**
1	1	NO

Existing Diagnosis Window: EDW

MWMX and MWSX device groups

Designation

EDW **E**xisting **D**iagnosis **W**indow

Explanation

Which types of diagnosis window exist is queried.

FI command

Output all types of diagnosis window.

BR_EDW1 **(Single Read)**

Response Structure

The following table shows the general structure of the "EDW1" FI command. The number of lines depends on the number of types of window existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type of diagnosis window [1 = CNC error, 2 = sequence errors,
3 = general errors,
4 = messages,
10 = start requirements,
11 = warnings, 12 = setup diagnosis]

2 = Module number [ASCII characters]
0 = Diagnosis window type does not belong to any module

Example EDW1

All types of diagnosis window in control unit 0.

There are three diagnosis windows.

FI command		00_BR_EDW1
Line	**Column**	**Answer**
1	1	10
	2	0
2	1	1
	2	3
3	1	2
	2	3

FI command

Output all diagnosis window types for a module.

BR_EDW2_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure

The following table shows the general structure of the "EDW2" FI command. The number of lines depends on the number of types of window existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type of diagnosis window	[1 = CNC error, 2 = sequence errors, 3 = general errors, 4 = messages]
2 = Module number	[ASCII characters] 0 = Diagnosis window type does not belong to any module

Example EDW2

All types of diagnosis window in Module 3, Control unit 0.

There are two diagnosis windows.

FI command		00_BR_EDW2_3
Line	**Column**	**Answer**
1	1	1
	2	3
2	1	2
	2	3

FI command

Query a specific type of diagnosis window.

BR_EDW3_(1){_(2)} **(Single Read)**

(1) = Type of diagnosis window	[1 = CNC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start requirements, 11 = warnings, 12 = setup diagnosis]
(2) = Module number	[1...99] ! only for window type 1 -4 !

Response Structure

The following table shows the general structure of the "EDW3" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Type of diagnosis window exists [YES, NO]

Example EDW3

Query whether or not an NC error window exists in module 3, control unit 0.

FI command		00_BR_EDW3_1_3
Line	**Column**	**Answer**
1	1	YES

Reference to Literature

See chapter literature [13].

Existing SPS Diagnoses: EPD

MWMX and MWSX device groups

Designation

EPD **E**xisting **P**LC **D**iagnosis

Explanation

Which PLC diagnostic types exist is queried. Depending on the FI command, specific types are queried or else the diagnostic types for a device or a module are output together.

FI command

Query which PLC diagnostic types are available on a control unit.

BR_EPD1 **(Single Read)**

Response Structure

The following table shows the general structure of the "EPD1" FI command.

Line 1	Column 1-3

Meaning of the Columns

1 = Start requirement exists [YES, NO]

2 = Warning exists [YES, NO]

3 = Setup diagnosis exists [YES, NO]

Example EPD1

Query PLC diagnostic types in control unit 0.

FI command		00_BR_EPD1
Line	**Column**	**Answer**
1	1	YES
	2	NO
	3	YES

FI command

Query which PLC diagnostic types are available in a module.

BR_EPD2_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure

The following table shows the general structure of the "EPD2" FI command.

Line 1	Column 1-3

Meaning of the Columns

1 = Messages exist [YES, NO]

2 = Errors exist [YES, NO]

3 = Step chains exist [YES, NO]

Example EPD2

Query the PLC diagnostic types in Module 2 on Control unit 0.

FI command		00_BR_EPD2_2
Line	**Column**	**Answer**
1	1	NO
	2	YES
	3	YES

FI command

Query a specific PLC diagnostic type.

BR_EPD3_(1){_(2)} **(Single Read)**

(1) = Message type [1 = error, 2 = messages, 3 = SFC, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -3!

Response Structure

The following table shows the general structure of the "EPD3" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Diagnosis type exists [YES, NO]

Example EPD3

Are there any messages in module 4 in control unit 0?

FI command		00_BR_EPD3_2_4
Line	**Column**	**Answer**
1	1	YES

Existing ProVi Types: EPT

MWMX and MWSX device groups

Designation **EPT** **E**xisting **P**roVi **T**ypes

Explanation Which ProVi types are programmed in the current PLC program is queried. The data is returned in a suitable form for the message texts of the small control panels. There is no need to define modules in Moduldef.ini.

FI command Output all ProVi types.

BR_EPT1 **(Single Read)**

Response Structure The following table shows the general structure of the "EPT1" FI command. The number of lines depends on the number of ProVi types existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type [11 = error, 12 = messages,
20 = start requirements,
21 = warnings, 22 = setup diagnosis]

2 = Index [ASCII characters]

Example EPT1 All ProVi types in control unit 0.

There are three diagnosis windows.

FI command		00_BR_EPT1
Line	**Column**	**Answer**
1	1	20
	2	0
2	1	11
	2	3
3	1	12
	1	3

Error Status: EST

MWMX and MWSX device groups

Designation **EST** **E**rror **ST**ate

Explanation Queries the error state of a variable.

FI command Query the frozen error state of a variable.

BR_EST1!(1)!(2) **(Single Read)**

BC_EST1!(1)!(2) **(Cyclic Read)**

(1) = Error ID [ASCII characters] (DWORD, decimal)

(2) = Variable name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "EXD1" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Error state

WinPcl - Example EST

Read the value of WinPcl variable "IB_EXT24" in WinPcl program "Prog", at device address 00.

Exception:
The WinPcl variable "IB_EXT24" is declared in the WinPcl Program "Prog" as BOOL.

FI command		00_BR_EST1!5892855!:Prog.IB_EXT24
Line	**Column**	**Answer**
1	1	1

Execution Display: EXD

MWMX and MWSX device groups

Designation

EXD **EX**ecution **D**isplay

Explanation

Information for displaying the execution of a movement is output.

FI command

Query the execution of a step or of an action.

BR_EXD1!(1)!(2)!(3) **(Single Read)**

BC_EXD1!(1)!(2)!(3) **(Cyclic Read)**

(1) = SFC entity name [ASCII characters]

(2) = Step or action name [ASCII - characters]

(3) = Behaviour of mode [1 = all modes,
2 = manual mode]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "EXD1" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Execution [1 = can be executed, 0 = cannot be executed]

Example EXD1

Query the execution of the step “open” for the chain “clamp” in control unit 0 for all modes.

FI command		00_BR_EXD1!Station03A.Clamp!Open!1
Line	**Column**	**Answer**
1	1	1

FI command

Query whether the condition analysis (control image) of a step chain is enabled.

BR_EXD2!(1) **(Single Read)**

(1) = SFC entity name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "EXD2" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Enabled [1 = enabled, 0 = not enabled]

Example EXD2 Query whether the condition analysis of the “clamp” chain has been enabled.

FI command		00_BR_EXD2!Station03A.Clamp
Line	**Column**	**Answer**
1	1	1

Read Reference Name of a PLC Variable: MAR

MWMX and MWSX device groups

Designation **MAR** **M**ap **A**bsolute PCL-**R**eference

PLC Explanation The absolute reference name of a symbolic PLC variable is read out.

FI command Read the absolute reference name of a PLC variable.

BR_MAR_(1) **(Single Read)**

(1) = Identifier of the PLC variable

PLC – Example MAR Read the absolute reference name of the PLC variable with the identifier “abref” at device address 00.

Assumption:
The PLC variable with the identifier "abref" is of the type "INTEGER".

FI command		00_BR_MAR_abref
Line	**Column**	**Answer**
1	1	%M100.0

WinPlc Explanation The absolute reference name of a symbolic WinPlc PLC variable with program entity is read out.

FI command Read the absolute reference name of a WinPlc PLC variable.

BR_MAR1_(1) **(Single Read)**

(1) = Identifier of the PLC variable

Win PLC - Example MAR1 Read the absolute reference name of the Win PLC variable with the identifier "Prog.abref" at device address 00.

Assumption:
The Win PLC variable with the identifier "Prog.abref" is of the type "INTEGER" and is present in Win PLC program "Prog".

FI command		00_BR_MAR1_:Prog.abref
Line	**Column**	**Answer**
1	1	%M100.0

Reference to Literature See chapter entitled “Literature” [30].

Device Data of the Module Configuration: MCD

MWMX and MWSX device groups

Designation **MCD** **M**odule **C**onfiguration: **D**evice Information

Explanation All device data for module configuration is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory on standard installation. The device data is in the sections [DeviceAddrX], whereby "X" stands for the configured device addresses.

FI command Read out device data within the module configuration of the MWSX device group.

BR_MCD1 **(Single Read)**

BC_MCD1 **(Cyclic Read)**

BB_MCD1 **(Break Cyclic Read)**

Note: The “MCD1” FI command refers to all devices within the MWSX device group. Therefore, any valid device address can be indicated in the command line (see example MCD1).

Response Structure The following table shows the general structure of the response to the “MCD1” FI command. The number of lines depends on the number of configured devices. Each line consists of four columns for the device address as well as PLC-FB names for providing setup diagnostics, warning messages and start requirements.

Value Range of the Columns

1 = Device address	[0...15]
2 = PLC-FB name for the setup diagnostics	[max. 9 ASCII characters]
3 = PLC-FB name for the warning messages	[max. 9 ASCII characters]
4 = PLC-FB name for the start requirements	[max. 9 ASCII characters]

Example MCD1 Read all device data of the module configuration

Assumption:
The following devices have been configured in the MWSX device group:

- Device address 01 (ISP200-P)
- Device address 03 (ISP200-R)

FI command	**03_BR_MCD1**			
		Answer		
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	01	PVSetup_1	PVWarn_1	PVStart_1
2	03	PVSetup_3	PVWarn_3	PVStart_3

Reference to Literature See chapter entitled “Literature” [36].

Module Data of the Module Configuration: MCM

MWMX and MWSX device groups

Designation **MCM** **M**odule **C**onfiguration: **M**odul Information

Explanation All module data of a particular device is read out from the "Moduldef.ini"' file. This file is located in the directory "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" and contains the data for all module configurations. The module data is located in sections [DeviceAddrX\ModulY], whereby "X" stands for the device addressed and "Y" for the configured module numbers.

FI command Read-out module data from the module configuration with respect to a device from the MWSX device group.

BR_MCM1 **(Single Read)**

BC_MCM1 **(Cyclic Read)**

BB_MCM1 **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the “MCM1” FI command. The number of lines depends on the number of configured modules of a device. Each line consists of four columns for the module number, module name and PLC-FB names for general module errors and module messages.

Line 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 general module errors	[max. 9 ASCII characters]
4 = PLC-FB name for module messages	[max. 9 ASCII characters]

Example MCM1 Read the module data of device 03 from the module configuration:

Assumption:
The following modules have been defined:

- Module number 5
- Module number 7

FI command	03_BR_MCM1			
Answer				
Line	**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

Reference to Literature See chapter entitled “Literature” [36].

SFC Data of the Module Configuration: MCS

MWMX and MWSX device groups

Designation **MCS** **M**odule **C**onfiguration: **S**FC Information

Explanation All SFC data of a particular module is read out from the "Moduldef.ini" file. This file is located in the directory "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" and contains the data for all module configurations. The SFC data is located in sections [DeviceAddrX\ModulY\Sfc], whereby "X" stands for the device addressed and "Y" for the selected module number.

FI command Read out the SFC data with respect to the module of a device from the module configuration of the MWSX device group.

BR_MCS1_(1) **(Single Read)**

BC_MCS1_(1) **(Cyclic Read)**

BB_MCS1_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Response Structure The number of lines depends on the number of configured Indrastep step chains for a device. Each line contains a column for the name of the Indrastep step chains.

Value Range of the Column 1 = Name of the Indrastep step chain [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 name of the Indrastep step chain of module 5 from device 03 of the module configuration.

Assumption:
The following Indrastep step chains have been defined:

- ISFB_1
- FB_US.ISFB_3
- FB_US.ISFB_3.SW1
- FB_US.ISFB_3.SW1.ABBA

FI command		**03_BR_MCS1_5**
Line	**Column**	**Answer**
1	1	ISFB_1
2	1	FB_US.ISFB_3
3	1	FB_US.ISFB_3.SW1
4	1	FB_US.ISFB_3.SW1.ABBA

Downloading Message Texts: MFD

MWMX and MWSX device groups

Designation **MFD** **M**essage **F**iles **D**ownload

FI command This is used to load the message texts into the device indicated. These message texts are required for small devices. The following message texts are transmitted, depending on the type of device:

- system error messages
- transmission error messages, and/or
- mechanism messages.

Note: This FI command is an FI job!

BW_MFD1 **(Single Write)**

Response Structure The response to the "MFD1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID	[01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).
Line 2 = FI command	[string, in accordance to chapter entitled "Elements of the FI Command"]
Line 3 = FI job error code	(see chapter entitled “Error Codes”)

Example MFD1 Load message texts into the device with device address 00.

FI command		00_BW_MFD1
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_MFD1
3	1	0

Reading Machine Key Information : MKS

MWMX and MWSX device groups

Designation **MKS** **M**achine **K**ey **S**tatus

Explanation Current machine key information can be read for the selected device.

FI command Read machine key information for selected device.

BR_MKS **(Single Read)**

BC_MKS **(Cyclic Read)**

BB_MKS **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the FI command "MKS".

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 =	Information of machine key	[4 byte in HEX coding]
2 =	Information valid?	[0 = not valid, 1=valid]

Example MKS Read the current machine key information for device 0.

FI command		00_BR_MKS
Line	**Column**	**Answer**
1	1	00000000
	2	0

Writing the GUI-SK Block: MKT

MWMX and MWSX device groups

Designation MKT **M**achine **K**ey **T**able

Explanation Writes the GUI-SK16 block in the PLC.

FI command Write GUI-SK16 block.

BW_MKT1_(1) **(Single Write)**

(1) = List of the 48 PLC variables for writing the GUI-SK16 block.

A distinction is made between the following cases:

1. Clear GUI-SK16 block.
2. Write the GUI-SK16 block with the 48 PLC variables, filling gaps with $SPACE.

Response Structure (P_ACK) is returned following successful transmission.

Line 1	Column 1

Value Range/Meaning of the Columns 1 = Successfully completed (P_ACK)

1. Example MKT1 1.Clear GUI-SK16 block:

FI command		00_BW_MKT1 Value to be written: $EMPTY
Line	**Column**	**Answer**
1	1	(P_ACK)

2. Example MKT1 Write GUI-SK16 block:

FI command		00_BW_MKT1 Value to be written: $EMPTY SPSVAR1,SPSVAR2,$SPACE,...
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command Write the GUI-SK16 block, writing only those PLC variables which are defined in the current PLC program. All undefined PLC variables are automatically replaced by $SPACE and returned as a partial result (column 2).

BW_MKT2_(1) **(Single Write)**

(1) = List of the 48 PLC variables for writing the GUI-SK16 block.

A distinction is made between the following cases:

1. Clear GUI-SK16 block:
   BW_MKT2 $EMPTY
2. Write the GUI-SK16 block with the 48 PLC variables, filling gaps with $SPACE:
   BW_MKT1 SPSVAR1,SPSVAR2,
   $SPACE,$SPACE,....

Response Structure

After successful transmission, one line with two columns is returned.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 =	Status report	[0 = at least 1 PLC variable in the current PLC program is NOT defined 1 = ALL PLC variables could be written]
2 =	List of the NON-defined PLC variables in the current PLC program	[-- = ALL PLC variables could be written, or else list of the PLC variables that could not be written.] The individual PLC variables are separated by a comma.

Example MKT1

Write GUI-SK16 component with 48 PLC variables, while the PLC variables SPSVAR11 and SPSVAR12 are NOT defined in the current PLC program.

FI command		**00_BW_MKT1** **Value to be written:** **SPSVAR1,SPSVAR2,...SPSVAR48**
Line	**Column**	**Answer**
1	1	(P_ACK)

Extended information

The variables are divided into 3 groups of 16 variables each and have the following meaning:

Variables 1 – 16:	Machine function keys
Variables 17 – 32:	Status pressed
Variables 33 -48:	Status shining

Note: When, for example, only the first 8 M keys are used, the telegram will contain only these 8 PLC variables. The other 40 variables need not be defined in the transmission parameter.

When certain areas, e.g. of M keys, are left unused, they must be filled up with '$SPACE' up to the next variable.

Read System Messages: MSG

MWMX and MWSX device groups

Designation

MSG **M**e**S**sa**G**e

Explanation

Reading of system messages

FI command

Message

CC_MSG_(1) **(Cyclic Read)**

(1) = SYS-Message number

Note: Exists only as a cyclic command

Response Structure

The response of the FI command 'MSG' consists of the system message data.

Example MSG

00_CC_MSG_64 (64 = MSG_SYSERRGEN)

FI command		00_CC_MSG_64/3
Line	Column	Answer
1	1	00

Restriction

The following system messages:

SYS Message	SYS Message number
MSG_PCLUPDBEG	52
MSG_PARUPDBEG	24
MSG_FWAUPDBEG	82

These commands cannot be used with the following programs:

- Indramat OPC server
- Indramat DDE server

Reading the Firmware Identification: MTC

MWMX and MWSX device groups

Designation

MTC **MT**-**C**NC Slot Software Version

FI command

This command is used to read the firmware identification from the various control components (slot numbers).

Note: For the time this FI command is executed, the internal FI communication interlocks (fast timeout monitoring, offline operation, etc.) are switched off.

FI command

BR_MTC_(1) **(Single Read)**

(1) = Slot number [1=CNC, 2=SIO, 3=PLC, 4=APR1 5=APR2, 6=APR3, 7=APR4]

Response Structure

The following table shows the general structure of the response to the FI command "MTC". A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Firmware identification string [max. 16 ASCII characters]

Example MTC

Read the firmware identification of slot number 1 (CPU) of device 00.

FI command		00_BR_MTC_1
Line	Column	Answer
1	1	CPU01/0004-20V00

FI command

This command is used to read the firmware identification from the various control components (slot numbers).

CR_MTC_(1) **(Single Read)**

(1) = Slot number [1=CNC, 2=SIO, 3=PLC, 4=APR1 5=APR2, 6=APR3, 7=APR4]

Response Structure

The following table shows the general structure of the response to the FI command "MTC". A line of 1 column is output.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Firmware Identification string [max. 16 ASCII characters]

Example MTC Read the firmware identification of slot number 1 (CPU) of device 00.

FI command		00_CR_MTC_1
Line	**Column**	**Answer**
1	1	CPU01/0004-20V00

ProVi Diagnosis Data: PDD

MWMX and MWSX device groups

Designation **PDD** **P**rovi **D**iagnosis **D**ata

Explanation Data for ProVi criteria analysis is output.

FI command Output of files to indicate the detail in the editor.

BR_PDD1_(1)_(2){_(3)} **(Single Read)**

(1) = Message ID [ASCII characters]

(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the PDD1 FI command.

Line 1	Column 1	...	Column 5

Meaning of the Columns

1 = POU ID [ASCII characters]

2 = Detail morpheme [ASCII characters] (DWORD, decimal)

3 = Error ID [ASCII characters] (DWORD, decimal)

4 = POE entity name [ASCII characters]

5 = Nw ID (network ID) [ASCII characters]

Example PDD1 Indication of data of a ProVi error with ID 43923028 from module 3 in control unit 0.

FI command		00_BR_PDD1_43923028_1_1
Line	**Column**	**Answer**
1	1	STATION_1_2
	2	98243823
	3	34985304
	4	Station2.Module3
	5	43493454

FI command Output the I/O addresses to display a detail.

BR_PDD2_(1)_(2){_(3)} **(Single Read)**

(1) = Message ID [ASCII characters]

(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the PDD2 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Variable morpheme [ASCII characters] (DWORD, decimal)
2 = I/O address [ASCII characters]

Example PDD2

Query of the I/O addresses of a ProVi error with ID 43923028 from module 3 in control unit 0.

Three variables have an I/O address.

FI command		00_BR_PDD2_43923028_1_1
Line	**Column**	**Answer**
1	1	98243823
	2	%I3.2.0
2	1	40923423
	2	%Q23.21.7
3	1	34985304
	2	%I100.3.5

FI command

Determine the multilingual comments for displaying a detail.

BR_PDD3_(1)_(2){_(3)} **(Single Read)**
(1) = Message ID [ASCII characters]
(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]
(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD3 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Comment morpheme [ASCII characters] (DWORD, decimal)
2 = New comment [ASCII characters]

Example PDD3

Query of the comments for indication of a ProVi error with ID 43923028 from module 3 in control unit 0.

Two comments are replaced by another text.

FI command		00_BR_PDD3_43923028_1_1
Line	**Column**	**Answer**
1	1	98243823
	2	Clamp open
2	1	40923423
	2	Clamp closed

FI command

Query of the status of a certain message

BR_PDD4_(1)_(2){_(3)} **(Single Read)**
(1) = Message number [ASCII characters]
(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]
(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD4 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Message is present [YES, NO]

2 = Criteria analysis exists [YES, NO]

Example PDD4

Query of the status of a ProVi error, number 1001 from module 3 in control 0.

This message is not present at the moment, and there is a criteria analysis.

FI command		00_BR_PDD4_1001_1_1
Line	**Column**	**Answer**
1	1	NO
	2	YES

FI command

Determination of the MessageID of a certain message

BR_PDD5!(1)!(2)!(3)!(4){!(5)} **(Single Read)**

(1) = POU entity name [ASCII characters]

(2) = Nw ID [ASCII characters]

(3) = Message number [ASCII characters]

(4) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(5) = Module number [1...99] ! only for message type 1 -2!

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the PDD5 FI command.

Line 1-n	Column 1	...	Column 3

Meaning of the Columns

1 = Message ID [ASCII characters] (DWORD, decimal)

2 = Message is present [YES, NO]

3 = Criteria analysis exists [YES, NO]

Example PDD5

Determination of the MessageID of a ProVi error, number 1001 from module 25.40 mm control 0.

Assumption:
This message is not present at the moment, and there is a criteria analysis.

FI command		00_BR_PDD5!Station2.Modul3!43493454!1001!1!1
Line	**Column**	**Answer**
1	1	240872342
	2	NO
	3	YES

Generating Physical Directory Names: PHD

MWMX and MWSX device groups

Designation **PHD** **PH**ysical **D**irectory

Explanation Generates physical directory names according to the BDI data written.

Note: This is based on BDI philosophy.

FI command Generate physical directory names.

BR_PHD1_(1)_(2)_(3)_(4)_(5)_(6) **(Single Write)**

(1) = Project ID	[-1= PROJECT_NEUTRAL -2= PROJECT_DEFAULT]
(2) = Section ID	[0= SECT_NEUTRAL 1= SECT_BIN 2= SECT_BASIC_DATA 3=SECT_OEM_DATA 4=SECT_CUSTOM_DATA 5=SECT_PROG_DATA]
(3) = Device address	[-1= DEVADDR_NEUTRAL otherwise the required device address]
(4) = Process ID	[-1= PROCESS_NEUTRAL otherwise the required process number]
(5) =Data type ID	[possible write values see BDI documentation (BDI_DEFINITIONS.H)]
(6) = Language ID	[possible write values see BDI documentation (WINNT.H)]

Response Structure The following table shows the general structure of the response to the FI command "PHD1".

Line 1	Column 1

Value Range/Meaning of Columns 1 = Physical directory name [complete physical directory name in accordance with the BDI data written]

Example PHD1 Requesting the physical directory name for:

PROJECT_NEUTRAL
SECT_BIN
DEVADDR_NEUTRAL
PROCESS_NEUTRAL
DATATYPE_NEUTRAL
LANG_NEUTRAL

FI command		XX_BR_PHD1_-1_0_-1_-1_0_0
Line	**Column**	**Answer**
1	1	D:\Programme\Indramat\Mtgui\Bin

Issuing SYS Messages Specific to the PCL: PSM

MWMX and MWSX device groups

Designation **PSM** **P**CL **S**ys **M**essage

Explanation Issues the most important SYS messages regarding the PCL programming interface – required for remote programming.

Note:
The appropriate device address is passed as the write value.

It allows the following SYS messages to be initiated:

- Start of PCL download,
- end of PCL download,
- start of PLC online edit,
- end PLC online edit,
- start of PCL declaration change, and
- end of PCL declaration change.

FI command Issue the most important PCL SYS messages.

BW_PSM1_(1) **(Single Write)**

(1) = Requested SYS message [1= start of PCL download
2= end of PCL download
3= start of PCL online edit
4= end of PCL online edit
5= start of PCL declaration change
6= end of PCL declaration change]

Value to be written
device address

Response Structure The following table shows the general structure of the response to the FI command "PSM1".

Line 1	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Status report	[READY=SYS message has been correctly acknowledged by the WIN32 applications] [ERROR=SYS message has NOT been acknowledged by a WIN32 application within the pre-set time]
2 =	Task name (LogInIf name)	[Task name that has triggered the SYS message]
3 =	SYS message number	[contains the issued SYS message number]
4 =	Acknowledgement time	[contains the pre-set acknowledgement time]
5 =	Reference information	[contains, where applicable, the additional information transferred as a write value]
6 =	Length of reference information	[0 where NO reference information has been transferred]
7 =	Where applicable, LOG channel of the FI that has NOT acknowledged	[-- = acknowledgements have been completed in time or the LOG channel number of the WIN32 application that has NOT acknowledged in time]

8 =	Where applicable, task name that has NOT acknowledged in time	[-- = acknowledgements have been completed in time or the task name that has NOT acknowledged in time]

Example PSM1

Issue the SYS message Beginning PCL Download. The reference information, device address 00, is also transferred as a write value.

FI command		**XX_BW_PSM1_1**
Line	**Column**	**Answer**
1	1	READY
	2	WINPCL.EXE
	3	14
	4	30000
	5	00
	6	2
	7	--
	8	--

Edit PROVI Message Files: PVA

MWMX and MWSX device groups

Designation **PVA** **P**ROV**I**-Messages **A**ccess

Explanation

This write command creates PROVI message files. With this write value, it is possible to decide whether the PROVI messages are to be generated according to the current PLC project, or selectively.

FI command **BW_PVA1** **(Single Write)**

Note: This command is an FI job command.

Value to be written

No write value exists	PROVI message files according to the current PLC project.
Write value exists	List of the requested PROVI message files (separated by a comma) according to the format: [PROVI-Diag-type: module number] Example: 01:01,01:02,02:02

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The response to the "BW_PVA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

...

Example PVA1

No write value is passed, i.e. the PROVI message files are generated according to the current PLC project.

FI command		**00_BW_PVA1**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVA1
3	1	0

Explanation

The read command returns the most significant information on the created PROVI message files.

FI command **BR_PVA1** **(Single Read)**

....

Response Structure

The following table shows the general construction of the answer of the FI command BR_PVA1. For each available PROVI message file, 1 line with 10 columns each is created.

Line 1...n	**Column 1**	**...**	**Column 10**

Value Range/Meaning of Columns

1 =	PROVI diagnosis type	[1..20]
2 =	PROVI diagnosis type designation	[The following designations can be returned: StartCondition, Error, Message, Warning, Setup]
3 =	Module number	[1..99]
4 =	PROVI diagnosis type and module number	[PROVI diagnosis type: module number, see write value for BW_PVA2]
5 =	Complete name of the PROVI message text file	[max. 200 ASCII characters]
6 =	Memory required for PROVI messages in the control	[figure in ASCII format]
7 =	Complete name of the PROVI index file	[max. 200 ASCII characters]
8 =	Memory required for PROVI index files in the control	[figure in ASCII format]
9 =	Total memory (text+index) required in the control	[figure in ASCII format]
10 =	Total memory for ALL PROVI files (text+index) required in the control	[figure in ASCII format]

Example PVA1 The most significant information of 2 available PROVI message files are returned.

FI command		00_BR_PVA1_1
Line	**Column**	**Answer**
1	1	1
	2	Error
	3	1
	4	01:01
	5	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	6	1345
	7	D:\Programme\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.IDX
	8	234
	9	1579
	10	4491
2	1	2
	2	Message
	3	1
	4	02:01
	5	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	6	2456
	7	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	8	456
	9	2912
	10	4491

Explanation This write command transmits PROVI message files into the selected device. Through the write value, it is possible to chose whether ALL or only the PROVI messages selected via the write value are to be transmitted.

FI command **BW_PVA2** **(Single Write)**

Note: This command is an FI job command.

Value to be written

No write value exists	All PROVI message files are transmitted into the selected device
Write value exists	List of the requested PROVI message files (separated by a comma) according to the format: [PROVI-Diag-type: module number] Example: 01:01,01:02,02:02

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The response to the "BW_PVA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

....

Example PVA1

No write value is passed, i.e. all PROVI message files should be transmitted.

FI command		**00_BW_PVA2**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVA2
3	1	0

Formatted Input / Output of PLC Variables: PVF

MWMX and MWSX device groups

Designation

PVF **P**LC **V**ariable **F**ormatted

Explanation

Formatted reading and writing of PLC variables, arrays and structures.

FI command

Read PLC variables.

CR_PVF_(1)	**(Single Read)**
CC_PVF_(1)	**(Cyclic Read)**
CB_PVF_(1)	**(Break Cyclic Read)**
(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]

Response Structure

One line with one column is output for single variables. For array and structure variables, one line per element is output, depending on the number of elements.

Line 1...n:	**Column 1**

n = number of elements.

Note: Only defined PLC variables can be read and written. Addressing a non-declared variable results in an error message. A PLC variable can only be read if its data length does not exceed 240 byte. (Refer also to chapter on “Programming” and “Guidelines”).

Value Ranges ANSI / ASCII

The value range of the response depends on the data type of the variable read. The following table indicates the range in which the results string is to be expected when reading out a single variable and into which C-data type this string can be converted without loss of information:

Data Type	Value Range	Can be converted to C-data type
BOOL	[0;1]	unsigned char
SINT	[-128...127]	char
INT	[-32768...32767]	short
DINT	[2147483648...2147483647]	long
USINT	[0...255]	unsigned char
UINT	[0...65535]	unsigned short
UDINT	[0...4294967295]	unsigned long
BYTE	[0x00...0xFF]	unsigned char
WORD	[0x0000....0xFFFF]	unsigned short
DWORD;	[0x00000000...0xFFFFFFFF]	unsigned long
TIME	[0...4294967295]	unsigned long (msec)
CHAR	[$00...$20,!...~,$7F...$FF]	char
STRING	<String> whereby <String> string is a character string with a maximum of as many characters as are declared for the string in the PLC	Char[xx+1]] +1 i.e. room for the zero byte
REAL	[-3.402823567E+38...3.402823567E+38]	Float

Note: An empty string is identified by two single inverted commas: ' ' (do not confuse with the double inverted commas ")!

All single variables can be part of array and structure variables. The value ranges maintain their validity, even when within structured data types.

Binary Value Range

The value range of the response depends on the data type of the variable read. The following table indicates the value range in which to expect the binary value of a single variable and how many bytes are included in the binary byte sequence:

Data Type	Value Range	Length (bytes)
BOOL	[00_H...01_H]	1
SINT	[80_H...$7F_H$] i.e. –128...127	1
INT	[8000_H (-32768)...$7FFF_H$ (32767)]	2
DINT	[80000000_H (-2147483648)... $7FFFFFFF_H$ (2147483647)]	4
USINT	[00_H (0)...FF_H (255)]	1
UINT	[00_H (0)...$FFFF_H$ (65535)]	2
UDINT	[0...4294967295]	4
BYTE	[0x00...0xFF]	1
WORD	[0x0000....0xFFFF]	2
DWORD;	[0x00000000...0xFFFFFFFF]	4
TIME	[0...4294967295]	4
CHAR	[$00...$20,!...~,$7F...$FF]	1

Data Type	Value Range	Length (bytes)
STRING	<String> whereby <String> string is a character string with a maximum of as many characters as are declared for the string in the PLC	XX+1
REAL	[-3.402823567E+38...3.402823567E+38]	4

Note: Binary array and structure elements are joined together without any spaces between (1-byte alignment).

PLC - Example 1 PVF

Read the value of the PLC variable "STK_TXT" in ASCII format from device address 00.

Assumption:
The "STK_TXT" variable is declared as STRING in the PLC program.

FI command		00_CR_PVF_STK_TXT/1
Line	**Column**	**Answer**
1	1	Repeat counter

WinPcl - Example 1 PVF

Read the value of WinPcl variable "STK_TXT" in ASCII format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "STK_TXT" is declared in WinPcl program "Prog" as STRING.

FI command		00_CR_PVF_:Prog.STK_TXT/1
Line	**Column**	**Answer**
1	1	Repeat counter

PLC - Example 2 PVF

Read the value of the PLC array "BEG_END" in ANSI format from device address 00.

Assumption:
The "BEG_END" variable is declared as BYTE with 2 elements in the PLC program.

FI command		00_CR_PVF_BEG_END/3
Line	**Column**	**Answer**
1	1	0x00
2	1	0x1F

WinPcl - Example 2 PVF

Read the value of WinPcl array "BEG_END" in ANSI format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "BEG_END" is declared in WinPcl program "Prog" as BYTE with two elements.

FI command		00_CR_PVF_:Prog.BEG_END/3
Line	**Column**	**Answer**
1	1	0x00
2	1	0x1F

PLC - Example 3 PVF

Read the value of the PLC structure "MSTRCT" in ASCII format from device address 00.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

TYP STRUCT
    T1 BOOL
    T2 CHAR
    T3 STRING[16]
    T4 TIME
END

FI command		00_CR_PVF_MSTRCT/1
Line	**Column**	**Answer**
1	1	0
2	1	A
3	1	ROBOT AXIS X
4	1	2000

WinPcl - Example 3 PVF

Read the value of WinPcl structure "MSTRCT" in ASCII format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

TYP STRUCT
    T1 BOOL
    T2 CHAR
    T3 STRING[16]
    T4 TIME
END

FI command		00_CR_PVF_:Prog.MSTRCT/1
Line	**Column**	**Answer**
1	1	0
2	1	A
3	1	ROBOT AXIS X
4	1	2000

FI command

Write PLC variable.

CW_PVF_(1) **(Single Write)**

(1) = Identifier of the PLC variable [acc. to declaration part of the PLC]

Value to be written

Value of data element [see value ranges]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine. The data code of the value is passed to the parameter “ValType”.

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Data element has been set

Value Range of the value to be written in ANSI / ASCII Format

The value ranges agree for the most part with the ANSI / ASCII result-value ranges during read access. ANSI umlauts are thereby converted into ASCII umlauts. Only ASCII umlauts are stored in the control unit. For deviations to this, please refer to the following note:

Note: Strings are enclosed by two single inverted commas ' ' , e.g. 'drill'.

Special characters can be indicated in accordance with DIN-1131 by a $ sign.

The following are used:

- $' '
- $$ $
- $R \r (Carriage Return)
- $L \n (Linefeed)
- $P \f (Form feed)
- $T \t (Tab)
- $xx refers to a character written as a hexadecimal value, e.g. $20 (space)

Array and structure elements are separated by a space.

Value Range of the Value to be written in Binary Format

The value ranges agree with the binary result-value range during read access. For deviations to this, please refer to the following note:

PLC - Example 4 PVF

Write into the PLC variable "STK_TXT" at device address 00. The value is passed in ANSI format.

Assumption:
The "STK_TXT" variable is declared as STRING in the PLC program.

FI command		**00_CW_PVF_STK_TXT/3**
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	'item counter'
Data code	/3

WinPcl - Example 4 PVF

Write into the WinPcl variable "STK_TXT" in WinPcl program "Prog" at device address 00. The value is passed in ANSI format.

Assumption:
The WinPcl variable "STK_TXT" is declared in WinPcl program "Prog" as STRING.

FI command		**00_CW_PVF_:Prog.STK_TXT/3**
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	'item counter'
Data code	/3

PLC - Example 5 PVF

Write into the PLC byte array "BEG_END" at device address 00. The value is passed in ANSI format.

Assumption:
The "BEG_END" variable is declared as a BYTE array with 2 elements in the PLC program.

FI command		00_CR_PVF_BEG_END/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	0x20 0x3f
Data code	/3

WinPcl - Example 5 PVF

Write into the WinPcl byte array "BEG_END" in WinPcl program "Prog" at device address 00. The value is passed in ANSI format.

Assumption:
The WinPcl variable "BEG_END" is declared in WinPcl program "Prog" as BYTE with two elements.

FI command		00_CW_PVF_:Prog.BEG_END/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	0x20 0x3f
Data code	/3

PLC - Example 6 PVF

Write the value of element T3 of the PLC structure "MSTRCT" at device address 00. The string “COUNTER” is transferred in binary format.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CW_PVF_MSTRCT.T3/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	Binary sequence: 43 4F 55 4E 54 45 52 00
Data code	/2

WinPcl - Example 6 PVF

Write the value of element T3 of the WinPcl structure "MSTRCT" at device address 00. The string “COUNTER” is transferred in binary format.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CW_PVF_:Prog.MSTRCT.T3/2
Line	Column	Answer
1	1	(P_ACK)

Value to be written:

Value of data element	Binary sequence: 43 4F 55 4E 54 45 52 00
Data code	/2

PLC - Example 7 PVF

Write the value of the PLC structure "MSTRCT" from the structure “mstrct” previously stored in the C program at device address 00.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
      T1     BOOL
      T2     CHAR
      T3     STRING[16]
      T4     TIME
END
```

To exchange binary data in a C program, the following “C” data type can be used:

```
#pragma pack(1)    //Write all elements
                   //without spaces next to each other.
typeder struct
{
      unsigned char T1;
      char         T2;
      char         T3[17]; //Space for zero byte
      unsigned long T4;
} Tymstrct;        // Declare structure
Tymstrct mstrct;   // Apply structure
```

FI command		00_CW_PVF_MSTRCT/2
Line	Column	Answer
1	1	(P_ACK)

Value to be written: address of the C structure.

Value of data element	&mstrct
Data code	/2

WinPcl - Example 7 PVF

Write the value of the WinPcl structure "MSTRCT" from the structure “mstrct” previously stored in the C program at device address 00.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
      T1     BOOL
      T2     CHAR
      T3     STRING[16]
      T4     TIME
END
```

To exchange binary data in a C program, the following “C” data type can be used:

```
#pragma pack(1)    //Write all elements
                   //without spaces next to each other.
typeder struct
{
      unsigned char T1;
      char          T2;
      char          T3[17]; //Space for zero byte
      unsigned long T4;
} Tymstrct;        // Declare structure
Tymstrct mstrct;  // Apply structure
```

FI command		00_CW_PVF_:Prog.MSTRCT/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written: address of the C structure.

Value of data element &mstrct

Data code /2

ProVi Messages: PVM

MWMX and MWSX device groups

Designation **PVM** **Pro**Vi **M**essages

Explanation ProVi messages are output. These messages are assigned to a particular message type or module.

FI command Output all ProVi messages.

BR_PVM1_(1){_(2)} **(Single Read)**

BC_PVM1_(1){_(2)} **(Cyclic Read)**

(1) = Message type [1 = error, 2 = messages, 10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -2!

Output first ProVi messages.

BR_PVM2_(1){_(2)} **(Single Read)**

BC_PVM2_(1){_(2)} **(Cyclic Read)**

(1) = Message type [1 = error, 2 = messages, 10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the FI commands "PVM1" and "PVM2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 6

Meaning of the Columns 1 = Message text [ASCII characters]

2 = Message number [ASCII characters]

3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Message ID	[ASCII characters] (DWORD, decimal)
6 = Reference text exists	[YES, NO]
7 = Criteria analysis exists	[YES, NO]
Message HTML file	[ASCII characters]

Example PVM1

All ProVi errors from module 3 in control unit 0. There are two messages:

FI command		**00_BR_PVM1_1_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	01.27.2000
	4	14:56:32
	5	43923028
	6	YES
	7	NO
	8	
2	1	Oil pressure too low
	2	124
	3	01.27.2000
	4	15:03:10
	5	98234039
	6	NO
	7	YES
	8	

Example PVM2

The first ProVi error from module 3 in control unit 0.

There are two messages:

FI command		**00_BR_PVM2_1_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	01.27.2000
	4	14:56:32
	5	43923028
	6	YES
	7	NO
	8	

FI command

Output the reference information of a ProVi message.

BR_PVM3_(1)_(2){_(3)}	**(Single Read)**
(1) = Message ID	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages,

10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the "PVM3" FI command.

Line 1	Column 1	...	Column 16

Meaning of the Columns

1 =	Message text	[ASCII characters]
2 =	Message number	[ASCII characters]
3 =	Error category	[ASCII characters] (empty no category)
4 =	Time stamp day	[mm.dd.yyyy]
5 =	Time stamp hour	[hh:mm:ss]
6 =	Reference text available	[YES, NO]
7 =	Reference text	[ASCII characters]
8 =	Message ID	[ASCII characters] (DWORD, decimal)
9 =	Diagnosis source	[ASCII characters] (PLC, CNC)
10 =	POE name	[ASCII characters]
11 =	Detail name	[ASCII characters] (empty implementation)
12 =	Detail type	[1 = action block, 3 = transition, 4 = implementation]
13 =	Network number	[ASCII characters]
14 =	Variable name	[ASCII characters]
15 =	POU entity name	[ASCII characters]
16 =	POU type	[2 = program, 3 = function block]
17 =	Analysis of criteria available	[YES, NO]
18 =	Message HTML file	[ASCII characters]
19 =	Reference info HTML file	[ASCII characters]

Example PVM3

Reference text of a ProVi error with ID 43923028 from module 3 in control unit 0.

FI command		00_BR_PVM3_43923028_1_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	1
	4	01.27.2000
	5	14:56:32
	6	YES
	7	Oil pressure too low Oil pipe leaking or insufficient oil.
	8	43923028
	9	PLC

FI command		00_BR_PVM3_43923028_1_3
Line	**Column**	**Answer**
	10	MODULE3
	11	
	12	4
	13	34
	14	EschutzT
	15	Station2.Module3
	16	3
	17	NO
	18	
	19	D:\Programme\Indramat\MtGui\Project_000\ ProgramData\HMTL\DE\Error34.html

FI command

One after the other, all active ProVi messages are output. In the result, one line each is returned. After expiry of the set time, the next message is returned. The clock frequency can be set via the last parameter. This value can only be set once for the PC. The value transmitted last is always the valid value. Default setting is one second.

BR_PVM4_(1){_(2)_(3)}	**(Single Read)**
BC_PVM4_(1){_(2)_(3)}	**(Cyclic Read)**
(1) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(2) = Module number	[1...99] ! only for message type 1 -2!
(3) = Clock frequency	[ASCII characters] Time in ms

Response Structure

The following table shows the general structure of the "PVM4" FI command.

If there are no messages, the number of lines is 0.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[ASCII characters]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Message ID	[ASCII characters] (DWORD, decimal)
6 = Reference text available	[YES, NO]
7 = Criteria analysis exists	[YES, NO]
8 = Message index (1 = 1. message)	[ASCII characters]
9 = Message HTML file	[ASCII characters]

Example PVM1

ProVi errors from module 3 in control unit 0.

The 2nd message is being output. The clock frequency is to be 2 seconds.

FI command		00_BR_PVM4_1_3_2000
Line	**Column**	**Answer**
1	1	Oil pressure too low
	2	124
	3	01.27.2000
	4	15:03:10
	5	98234039
	6	NO
	7	YES
	8	2
	9	

Download of PLC Retain Variables: PVR

MWMX and MWSX device groups

Designation **PVR** **P**LC **V**ariable **R**etain Backup

Explanation Download of PLC retain variables.

FI command **BW_PVR1!(1)** **(Single Write)**

(1) = Download file with path details.

Note: File and path details must be enclosed in inverted commas.
The separator "!" is used in this command.

Response Structure The response to the "PVR1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example PVR1 00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini" /3
3	1	0

Structure of Download File The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Upload of PLC Retain Variables: PVR

MWMX and MWSX device groups

Designation **PVR** **P**LC **V**ariable **R**etain Backup

Explanation PLC retain variables are uploaded via all active processes.

FI command **BR_PVR1!(1)** **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.
The separator "!" is used in this command.

Response Structure The response to the "PVR1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

Example PVR 00_BR_PVR1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\upload.ini"/3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\upload.ini" /3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Note: Care must be taken in the use of upper and lower case letters.

Reading the PLC Variable Declaration: PVT

MWMX and MWSX device groups

Designation **PVT** **P**LC **V**ariable **T**ype

Explanation A PLC variable has a particular type. To evaluate complex variables such as structures and arrays, their components and types must be read out.

Refer also to PVF, Reading Structured PLC Variables.

FI command Read the PLC variable type.

BR_PVT_(1) **(Single Read)**

(1) = Identifier of the PLC variable [acc. to declaration part of the PLC]

Response Structure One line with 2 columns is output for each element of the variables.

Line 1...n:	**Column 1**	**Column 2**

n = number of elements.

Value Range/Meaning of Columns

1 = Identifier of the PLC variable	[acc. to declaration part of the PLC]
2 = Type	[see value range PVF]

Examples:
PLC: Reading of a variable

Assumption:
The "TEST" variable is declared as WORD in the PLC program.

FI command	**00_BR_PVT_TEST**	
Answer		
Line	**Column 1 (Name)**	**Name**
1	TEST	WORD

WinPcl: Reading a Variable

Assumption:
The WinPcl variable "TEST" is declared as WORD in WinPcl program "Prog".

FI command	**00_BR_PVT_:Prog.TEST**	
Answer		
Line	**Column 1 (Name)**	**Name**
1	TEST	WORD

PLC: Reading a Structure

Assumption:
The "TEST1" variable is declared as STRUCT in the PLC program.

```
STRUCT
    E1    BOOL
    E2    INT
    E3    SINT
END
```

FI command	**00_BR_PVT_TEST1**	
Answer		
Line	**Column 1**	**Column 2**
1	TEST1.E1	BOOL
2	TEST1.E2	INT
3	TEST1.E3	SINT

WinPcl: Reading a Structure

Assumption:

The WinPcl variable "TEST1" is declared as STRUCT in WinPcl program "Prog".

```
STRUCT
    E1    BOOL
    E2    INT
    E3    SINT
END
```

FI command	**00_BR_PVT_:Prog.TEST1**	
Answer		
Line	**Column 1**	**Column 2**
1	TEST1.E1	BOOL
2	TEST1.E2	INT
3	TEST1.E3	SINT

PLC: Reading an Array

Assumption:
The "TEST2" variable is declared as ARRAY in the PLC program.

ARRAY [
0 .. 3
] OF BOOL

FI command	00_BR_PVT_TEST2	
Answer		
Line	**Column 1**	**Column 2**
1	TEST2[0]	BOOL
2	TEST2[1]	BOOL
3	TEST2[2]	BOOL
4	TEST2[3]	BOOL

WinPcl: Reading an Array

Assumption:
The WinPcl variable "TEST2" is declared as ARRAY in WinPcl program "Prog".

ARRAY [
0 .. 3
] OF BOOL

FI command	00_BR_PVT_:Prog.TEST2	
Answer		
Line	**Column 1**	**Column 2**
1	TEST2[0]	BOOL
2	TEST2[1]	BOOL
3	TEST2[2]	BOOL
4	TEST2[3]	BOOL

PLC: Reading an Array of a Structure

Assumption:
The "TEST3" variable is declared as ARRAY in the PLC program.

ARRAY [
0 .. 1
] OF STRUCT1,

where STRUCT1 is declared as follows:

STRUCT
E1 BOOL
E2 INT
E3 SINT
END

END

FI command	00_BR_PVT_TEST3	
Answer		
Line	**Column 1**	**Column 2**
1	TEST3[0].E1	BOOL
2	TEST3[0].E2	INT
3	TEST3[0].E3	SINT
1	TEST3[1].E1	BOOL
2	TEST3[1].E2	INT
3	TEST3[1].E3	SINT

WinPcl: Reading an Array of a Structure

Assumption:
The WinPcl variable "TEST3" is declared as ARRAY in WinPcl program "Prog".

```
ARRAY [
        0 .. 1
] OF    STRUCT1,
```

where STRUCT1 is declared as follows:

```
STRUCT
        E1      BOOL
        E2      INT
        E3      SINT
END
```

FI command	**00_BR_PVT_:Prog.TEST3**	
Answer		
Line	**Column 1**	**Column 2**
1	TEST3[0].E1	BOOL
2	TEST3[0].E2	INT
3	TEST3[0].E3	SINT
1	TEST3[1].E1	BOOL
2	TEST3[1].E2	INT
3	TEST3[1].E3	SINT

Assumption:
The data types are output according to IEC1131.

See also command PVF.

Set the Device Status Information: SDS

MWMX and MWSX device groups

Designation **SDS** **S**et **D**evice **S**tatus

Explanation By this command, the device status information can be set; here, the configuration file IND_DEV.INI is adjusted as well.

Note: When this command is transmitted, the following system messages are generated:
MSG_DEVICEOFF or MSG_DEVICE_ON !

FI command With this command, the device status information of **ALL** defined devices can be set.

BW_SDS1_(1)	**(Single Write)**
(1) = Device status information to be set	0 = Device status information OFF 1 = Device status information ON

Response Structure The following table shows the general structure of the response to the "SDS1" FI command.

Line 1	**Column 1**

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example SDS1

Set device status information to OFF for **ALL** defined devices.

FI command		00_BW_SDS1_0
Line	Column	Answer
1	1	(P_ACK)

FI command

With this command, the device status information for a selected device can be set.

BW_SDS2_(1) **(Single Write)**

(1) = Device status information to be set

0 = Device status information OFF
1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS2" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example: SDS2

Set device status information to OFF for the selected device 00.

FI command		00_BW_SDS2_0
Line	Column	Answer
1	1	(P_ACK)

Sequencer Data: SFD

MWMX and MWSX device groups

Designation

SFD **SF**C **D**ata

Explanation

Data for a step chain is outputted. Depending on the FI command this can concern a step chain comment, POE name, step comment, maximum time, action / transition / monitor error name (comment), qualifier and time value.

FI command

Query the data for a step chain.

BR_SFD1!(1)!(2) **(Single Read)**

(1) = Module number [1...99]

(2) = SFC entity name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "SFD1" FI command.

Line 1	Column 1	Column 2

Meaning of the Columns

1 = Step chain comment [ASCII characters]

2 = POE name [ASCII characters]

Example SFD1 Query data of the “clamp” chain in module 3 in control unit 0.

FI command		00_BR_SFD1!3!Station03A.Clamp
Line	**Column**	**Answer**
1	1	Clamping device
	2	CLAMP

FI command Query the data of a step.

BR_SFD2!(1)!(2)!(3)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Step name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "SFD2" FI command. The number of lines depends on the number of actions and transitions.

If there are no details the line number is 1.

Line 1	Column 1	...	Column 3
Line 2...n:	**Column 1**	**...**	**Column 6**

Meaning of the Columns

Line 1

1 = Step comment	[ASCII characters]
2 = Maximum time	[ASCII characters]
3 = Minimum time	[ASCII characters]

Line 2...n:

1 = Detail type	[1 = action block, 3 = transition]
2 = Name	[ASCII characters]
3 = Comment	[ASCII characters]
4 = Boolean variable	[YES, NO]
5 = Qualifier	[ASCII characters]
6 = Time value	[ASCII characters]

Example SFD2 Data for the step “Open” in the “clamp” chain in module 3 on control unit 0.

FI command		00_BR_SFD2!3!Station03A.Clamp!Open
Line	**Column**	**Answer**
1	1	Open clamping device
	2	T#5s
	3	
2	1	1
	2	aOpen
	3	Clamp open
	4	NO
	5	D
	6	T#3s
3	1	3

FI command		00_BR_SFD2!3!Station03A.Clamp!Open
Line	Column	Answer
	2	tOpen
	3	Clamping device is open
	4	NO
	5	
	6	

FI command Output the data for a detail.

BR_SFD3!(1)!(2)!(3)!(4) **(Single Read)**

(1) = Module number [1...99]

(2) = SFC entity name [ASCII characters]

(3) = Detail type [1 = action block, 2 = action network, 3 = transition]

(4) = Detail name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "SFD3" FI command.

Line 1	Column 1	Column 2

Meaning of the Columns

1 = Comment [ASCII characters]

2 = Boolean variable [YES, NO]

Example SFD3 Data for the action “aOpen” in the “clamp” chain in module 3 in control unit 0.

FI command		00_BR_SFD3!3!Station03A.Clamp!aOpen
Line	Column	Answer
1	1	Clamp open
	2	NO

Sequencer Messages: SFE

MWMX and MWSX device groups

Designation **SFE** **SF**C **E**rror

Explanation The sequencer messages of a module are output.

FI command Output all SFC messages.

BR_SFE1_(1) **(Single Read)**

BC_SFE1_(1) **(Cyclic Read)**

(1) = Module number [1...99]

Output first SFC messages.

BR_SFE2_(1) **(Single Read)**

BC_SFE2_(1) **(Cyclic Read)**

(1) = Module number [1...99]

Response Structure

The following table shows the general structure of the FI commands "SFE1" and "SFE2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n:	Column 1	...	Column 7

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = SFC entity name	[ASCII characters]
3 = Step name	[ASCII characters]
4 = Time stamp day	[mm.dd.yyyy]
5 = Time stamp time	[hh:mm:ss]
6 = Type of error	[1 = time error, 2 = monitor error, 3 = monitor event]
7 = Is there condition analysis?	[YES, NO]

Example SFD1

All SFC messages from module 2 in control unit 0.

There are two messages:

FI command		**00_BR_SFE1_2**
Line	**Column**	**Answer**
1	1	TIME ERROR: Chain: chucking Step: up malfunction
	2	Station03A.Clamp
	3	Open
	4	01.27.2000
	5	11:56:32 AM
	6	1
	7	YES
2	1	ASSY ERROR: Chain: drilling Step: down malfunction
	2	Station02A.Drill
	3	Down
	4	01.27.200
	5	13:03:12
	6	2
	7	NO

Example SFE2

First SFC message from module 2 in control unit 0.

There are two messages.

FI command		**00_BR_SFE2_2**
Line	**Column**	**Answer**
1	1	TIME ERROR: Chain: chucking Step: up malfunction
	2	Station03A.Clamp
	3	Open
	4	01.27.2000
	5	14:56:32
	6	1
	7	YES

Sequencer Mode: SFM

MWMX and MWSX device groups

Designation **SFM** **SF**C **M**ode

Explanation Queries step chain mode.

FI command Query the mode of a step chain.

BR_SFM1!(1)!(2) **(Single Read)**

BC_SFM1!(1)!(2) **(Cyclic Read)**

(1) = Module number [1...99]

(2) = SFC entity name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "SFM1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Mode [1 = time error, 2 = monitor error, 3 = monitor event, 10 = stop, 11 = auto, 12 = manual, 13 = jog]

Example SFM1 Query mode of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SFM1!3!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	1

Software Installation Data: SID

MWMX and MWSX device groups

Designation **SID** **S**oftware **I**nstallation **D**ata

Explanation Information is returned regarding installation. This information includes installation paths, the software version used, DLL mode, plus service pack and release information.

FI command Read-in the installation data.

BR_SID1 **(Single Read)**

BC_SID1 **(Cyclic Read)**

Response Structure One line with 8 columns is output for the returned values.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Basic directory [EXE files of the BOF]

2 = FI installation directory [FI directory]

3 = Data directory [in accordance with BOF]

4 = GBO version [from INDRAMAT.ini]

5 = IF-DLL mode [from INDRAMAT.ini]

6 = IF version [from INDRAMAT.ini - from DLL mode 400]

7 = Service package info [from INDRAMAT.ini - from DLL mode 420]

8 = Release info	[from INDRAMAT.ini - from DLL mode 420]

Example SID1 Return information on the current installation.

FI command		**00_BR_SID1**
Line	**Column**	**Answer**
1	1	--
	2	D:\Programme\Indramat\MTGUI\Bin
	3	--
	4	005-22Vxx
	5	07.00
	6	07V00
	7	--
	8	--

Note: Refer to FI command “PHD” for working with absolute paths.

PLC Long Identification: SLI

MWMX and MWSX device groups

Designation **SLI** **S**PS **L**ong **I**dentification

Explanation Returns the unit data from the PLC long identification.

FI command Read PLC long identification.

BR_SLI **(Single Read)**

Response Structure One line with 15 columns is output for the returned values.

Line 1	**Column 1**	**Column...**	**Column 15**

Value Range/Meaning of the Columns

1 =	device address	[00...15]
2 =	program number	[01...99]
3 =	Project name	[max. 8 ASCII characters]
4 =	Program name	[max. 8 ASCII characters]
5 =	User name	[acc. to password entry]
6 =	Program length	[bytes]
7 =	Compilation time	[LONG] (coded in long value)
8 =	Compilation date	[8 ASCII characters]
9 =	Compilation time	[8 ASCII characters]
10 =	Download time	[LONG] (coded in long value)
11 =	Download date	[8 ASCII characters]1
12 =	Download time	[8 ASCII characters]
13 =	Version of PLC long identification	[LONG]
14 =	RUN flags	[HEX value]
15 =	Compiler info	[LONG]

Example SLI Read the unit data from the PLC long identification.

FI command		00_BR_SLI
Line	Column	Answer
1	1	02
	2	01
	3	--
	4	MOT12
	5	TEST
	6	17672
	7	630163960
	8	15.12.99
	9	17:15:48
	10	630163961
	11	15.12.99
	12	17:15:50
	13	2
	14	0x0000
	15	13

Reference to Literature see chapter entitled “Literature” [30].

Requesting Watch List Allocations: WLA

MWMX and MWSX device groups

Designation **WLA** **W**atch **L**ist **A**llocation

Explanation Requests free watch list allocations. A maximum of ten free watch list allocations can be requested with one FI command.

BR_WLA1_(1)	**(Single Read)**
(1) =Number of requested free watch list allocations	The required number of free watch list allocations is identified here. The allowed value range: 1..10.

Response Structure The following table shows the general structure of the response to the FI command "WLA1".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. free watch list allocation	Value range: 1..16
2 =	2. free watch list allocation	Value range: 1..16
3 =	3. free watch list allocation	Value range: 1..16
n =	nth free watch list allocation	Value range: 1..16

Example WLA1

Request four free watch list allocations.

Assumption:
Watch list allocations 3 and 5 are already assigned!

FI command		00_BR_WLA1_4
Line	**Column**	**Answer**
1	1	1
	2	2
	3	4
	4	6

Freeing Watch List Allocations: WFL

MWMX and MWSX device groups

Designation **WLF** **W**atch **L**ist **F**ree

Explanation

Previously requested watch list allocations are freed again.

FI command

Free ALL assigned watch list allocations for the selected device.

BR_WLF1 **(Single Read)**

Note: The FI command "WLF1" frees ALL assigned watch list allocations, including those of other WIN32 applications.

Response Structure

The following table shows the general structure of the response to the FI command "WLF1".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. freed watch list allocation	Value range: 1..16
2 =	2. freed watch list allocation	Value range: 1..16
3 =	3. freed watch list allocation	Value range: 1..16
n =	nth freed watch list allocation	Value range: 1..16

Example WLF1

Free ALL assigned watch list allocations.

Assumption:
The following watch list numbers have been allocated: 1, 2, 3, 4.

FI command		00_BR_WLF1
Line	**Column**	**Answer**
1	1	1
	2	2
	3	3
	4	4

FI command

Free the required watch list allocations for a selected device.

BR_WLF2_(1)_{(2)..(10)}	**(Single Read)**
(1)..(10) = List of watch list allocations to be released	A maximum of 10 watch list allocations can be transferred here to be freed again.

Response Structure

The following table shows the general structure of the response to the FI command "WLF2".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. freed watch list allocation	Value range: 1..16
2 =	2. freed watch list allocation	Value range: 1..16
3 =	3. freed watch list allocation	Value range: 1..16
n =	nth freed watch list allocation	Value range: 1..16

Example WLF2

Free required watch list allocations:
Assumption: Watch list allocations 1,3,4, and 8 have first been requested using the FI command "WLA1".

FI command		**00_BR_WLF2_1_3_4_8**
Line	**Column**	**Answer**
1	1	1
	2	3
	3	4

7.6 FI Commands for the MWAX Device Group

The FI Commands described in this chapter are valid for the MWAX device group. In this device group, the following types are listed as well as possible device addresses:

Group	Accompanying Types	Address
MWAX	MTA200-P (MTA200-controller)	[00...63]

Note: Please note that the device address must be set before the respective FI command, e.g. “00_BR_ASM1” (refer also here to the Chapter entitled “Elements of the FI Command”).

MTA200 Messages: ADM

MWAX device group

Designation **ADM** MT**A**200 **M**essages

Explanation MTA200 NC messages are output. These messages are assigned to a specific module and message type.

FI command Output all MTA200 messages.

BR_ADM1_(1)_(2)	(Single Read)
BC_ADM1_(1)_(2)	(Cyclic Read)
(1) = Message type	[1 = error, 2 = messages]
(2) = Module number	[1...99]

Output of first MTA200 message.

BR_ADM2_(1)_(2)	(Single Read)
BC_ADM2_(1)_(2)	(Cyclic Read)
(1) = Message type	[1 = error, 2 = messages]
(2) = Module number	[1...99]

Response Structure The following table shows the general structure of the FI commands "ADM1" and "ADM2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 6

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[0..32768]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Message group	[1..9999]
6 = Reference text exists	[YES, NO]

Example ADM1

All MTA200 errors from module 3 in control unit 0.

There are two messages:

FI command		**00_BR_ADM1_1_3**
Line	**Column**	**Answer**
1	1	24 volt supply absent
	2	1002
	3	01.27.2000
	4	14:56:32
	5	12
	6	YES
2	1	Program stop
	2	152
	3	01.27.2000
	4	15:03:10
	5	13
	6	NO

Example ADM2

The first MTA200 error from module 3 in control unit 0.

There are two messages:

FI command		**00_BR_ADM2_1_3**
Line	**Column**	**Answer**
1	1	24 volt supply absent
	2	1002
	3	01.27.2000
	4	14:56:32
	5	12
	6	YES

FI command

Output the additional information of a MTA200 message.

BR_ADM3_(1)_(2)_(3)	**(Single Read)**
(1) = Module number	[1...99]
(2) = Message number	[0..32768]
(3) = Message group	[1..9999]

Response Structure

The following table shows the general structure of the "ADM3" FI command.

Line 1	**Column 1**	**...**	**Column 7**

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[0..32768]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Message group	[1..9999]
6 = Additional text exists	[YES, NO]
7 = Additional text	[ASCII characters]

Example ADM3

Additional text of an MTA200 error in module 3 in control unit 0.

FI command		00_BR_ADM3_3_1002_12
Line	**Column**	**Answer**
1	1	24 volt supply absent
	2	1002
	3	01.27.2000
	4	14:56:32
	5	12
	6	YES
	7	Switch on voltage

Active Mechanism Messages: AMM

MWAX device group

Designation

AMM **A**ctive **M**echanism **M**essages

Explanation

Messages regarding active mechanism errors and mechanism diagnostics are output. These messages are assigned to a particular mechanism or process. Depending on the FI command, the device address, device name, mechanism number, mechanism name, type of message, message source, messages group, message number and messages text are all output.

FI command

Output mechanism messages currently pending.

BR_AMM7 **(Single Read)**

BC_AMM7 **(Cyclic Read)**

BB_AMM7 **(Break Cyclic Read)**

Note: The “AMM7” FI command refers only to devices within the MWAX device group. You should therefore make sure that only MTA devices are addressed via the system address.

Response Structure

The following table shows the general structure of the response to the FI command "AMM7". The answer consists of one up to a maximum of n=512 lines, each with 11 columns. The order of the individual error messages is oriented towards the time stamp, i.e. the oldest (triggering) error message is inserted into the first line. The maximum content for a result may not exceed 56 kbyte.

Line 1...n:	Column 1	Column...	Column 11

Value Range/Meaning of the Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Mechanism number	[0, default value always 0]
4 =	Mechanism name	[max. 28 ASCII characters, default value always the MTA process]
5 =	Type of message	[F = Fault/Error, D = Diagnosis]
6 =	Message source	[CNC, SPS, default value always "CNC"]
7 =	Message group	[1...9999]
8 =	Message number	[0...32768]
9 =	Message text	[max. 1024 ASCII characters]

10 =	Additional text	[X = exists, - - = does not exist, Default value - - does not exist (compatibility with Rexroth Indramat control units)]
11 =	2 bytes of additional information for the message number	[is required to resolve the information "@", default value "0" (compatibility with Rexroth Indramat control units)]
12 =	File name for additional information	e.g. in HTML format

Example AMM7

Read the current mechanism messages of device address 3 (MTA200).

FI command		**00_BR_AMM7**
Line	**Column**	**Answer**
1	1	03
	2	Crankshaft grinding machine
	3	0
	4	MTA process
	5	F
	6	CNC
	7	1
	8	5
	9	Programming error
	10	- -
	11	0
	12	
2	1	03
	2	Crankshaft grinding machine
	3	0
	4	MTA process
	5	F
	6	CNC
	7	1
	8	6
	9	Cycle point error
	10	- -
	11	0
	12	
3	1	03
	2	Crankshaft grinding machine
	3	0
	4	MTA process
	5	F
	6	CNC

FI command		00_BR_AMM7
Line	**Column**	**Answer**
	7	1
	8	19
	9	Emergency-OFF with immediate stop
	10	- -
	11	0
	12	

Actual (Current) Position of an Axis: APO

MWAX device group

Designation **APO** Actual **A**xis **PO**sition

Explanation The actual position of a selected axis is read out. The FI command "APO1" returns the position of an axis, related to the code of the axis meaning. On the other hand, the FI command "APO2" returns the position of an axis, related to the physical axis number.

FI command Output the position of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches.

CR_APO1_(1)_(2)_(3){_(4)}	**(Single Read)**
CC_APO1_(1)_(2)_(3){_(4)}	**(Cyclic Read)**
CB_APO1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC process number	[0]
(2) = Axis meaning	[1...16, the axis meaning corresponds to the physical axis number]
(3) = System of coordinates	[1 = machine coordinates 2 = program coordinates 3 = relative coordinates]
(4) = Required measurement system (opt.)	[mm, inch]

FI command Output the position of the selected axis of the device specified, related to the physical axis number.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches.

CR_APO2_(1)_(2){_(3)}	**(Single Read)**
CC_APO2_(1)_(2){_(3)}	**(Cyclic Read)**
CB_APO2_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1..0.16, according to settings of the system parameters]
(2) = System of coordinates	[1 = machine coordinates 2 = program coordinates 3 = relative coordinates]
(3) = Required measurement system (opt.)	[mm, inch]

Response Structure

The following table shows the general structure of the response to the FI commands “APO1” and “APO2”. One line is output with 4 columns for the axis designation, position, unit and the position limited to “indicated decimal places”.

Line 1	Column 1	Column 2	Column 3	Column 4

Value Range/Meaning of Columns

1 = Axis name [according to settings of axis parameters]

2 = Position [according to settings of process parameters]

3 = Unit [according to settings of process parameters: mm, inch]

4 = Position [as in column 2]

Note: If the selected axis is not defined then the response in all columns is [--].

Example APO1

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in the basic measurement system.

FI command	00_CR_APO1_0_3_1			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z	-1.2345	[mm]	-1.2345

Example APO1

Read the current position of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in inches.

FI command	00_CR_APO1_0_3_1_inch			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z	-0.0486	[inch]	-0.0486

Example APO2

Reads the current position of the Z axis (physical axis number = 3) in machine coordinates for the device address 00. The values are indicated in the basic measuring system.

FI command	00_CR_APO2_3_1			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	Z1	-1.2345	[mm]	-1.2345

Reference to Literature

See chapter entitled “Literature” [16].

Active System Error Messages: ASM

MWAX device group

Designation **ASM** **A**ctive **S**ystem **M**essages

Explanation The active system error messages that affect the functioning of the entire electrical device are output. Depending on the FI command, the device address, device name, message number, type of message, short text and additional text are all output. Access to system error messages only refers to the PLC part (ISP200).

FI command Output the system error messages currently pending for all active devices from the MWAX device group.

BR_ASM1 **(Single Read)**

BC_ASM1 **(Cyclic Read)**

BB_ASM1 **(Break Cyclic Read)**

Note: The “ASM1” FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see example “ASM1”).

Response Structure The following table shows the general structure of the response to the FI command “ASM1”. The number of lines (1 .. n=15) depends on the number of defined devices. Each line consists of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an additional text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see ASM5)
8 =	File name for additional information	e.g. in HTML format

Example ASM1 Read the current system error messages of all defined devices within the MWAX device group.

<u>Assumption:</u>
The following three devices are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		07_BR_ASM1
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	
2	1	07
	2	Milling center 1
	3	74
	4	F
	5	SLM time monitoring
	6	X
	7	0
	8	
3	1	10
	2	Milling center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0
	8	

FI command

Output the currently pending system error message of the selected device from the MWAX device group.

BR_ASM2 **(Single Read)**

BC_ASM2 **(Cyclic Read)**

BB_ASM2 **(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the FI command "ASM2". The response consists of a line of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an additional text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Additional text	[x= exists, -- = does not exist]

7 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see ASM5)
8 =	File name for additional information	e.g. in HTML format

Example ASM2

Read the current system error 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**
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	

FI command

Output current system error messages of the device listed from the MWAX device group.

BR_ASM3_(1) **(Single Read)**

BC_ASM3_(1) **(Cyclic Read)**

BB_ASM3_(1) **(Break Cyclic Read)**

(1) = Selection list for a max. of 10 MWAX [00_01_02_ ... _15] devices

Response Structure

The following table shows the general structure of the response to the FI command “ASM3”. The number of lines (1 .. n=15) depends on the number of listed MWAX devices. Each line consists of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an additional text for this error message.

Line 1...n	**Column 1**	**...**	**Column 7**

Value Range/Meaning of Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information for the message number	is required to resolve the information “@” (see ASM5)
8 =	File name for additional information	e.g. in HTML format

Example ASM3

Read the current system error messages for the selected MWAX devices.

Assumption:
The following devices addresses are defined:

- Device address 01,
- Device address 07 and
- Device address 10.

FI command		01_BR_ASM3_01_10
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	
2	1	10
	2	Milling center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0
	8	

FI command

Output current system error messages of all defined devices (in accordance with the system configuration) from the MWAX device group.

BR_ASM4_(1) **(Single Read)**

BC_ASM4_(1) **(Cyclic Read)**

BB_ASM4_(1) **(Break Cyclic Read)**

(1) = Device group [MTRX, MWCX, MWSX, MWAX]

Response Structure

The following table shows the general structure of the response to the FI command “ASM4”. The number of lines (1 .. n=15) depends on the number of defined MWAX devices. Each line consists of 7 columns for the device address, device name, message number, message status, short text and indication of whether there is an additional text for this error message.

Line 1...n	Column 1	...	Column 7

Value Range/Meaning of Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Short text	[max. 54 ASCII characters]
6 =	Reference text	[x= exists, -- = does not exist]
7 =	2 bytes of additional information	is required to resolve the information “@” (see ASM5)

	for the message number	
8 =	File name for additional information	e.g. in HTML format

Example ASM4

Read the current system error messages of all defined devices within the MWAX device group.
Assumption:
The following devices are defined:

- Device address 01 and
- Device address 10.

FI command		01_BR_ASM4_MWAX
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	71
	4	F
	5	PLC battery voltage too low.
	6	X
	7	0
	8	
2	1	10
	2	Milling center 2
	3	1
	4	D
	5	Error has been corrected.
	6	X
	7	0
	8	

FI command

Output the additional text for the currently pending error message, related to the device and the message number.

BR_ASM5_(1)_(2) **(Single Read)**

(1) = Message number [0...150]

(2) = 2 bytes of additional information for the message number

Response Structure

The following table shows the general structure of the response to the FI command "ASM5". The response consists of a line with 5 columns for device address, device name, message number and additional text.

Line 1...n	Column 1	...	Column 5

Value Range/Meaning of Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Message number	[0...150]
4 =	Type of message	[F = fault/error, D = diagnosis]
5 =	Reference text	[max. 14 lines with a max. 78 characters/line]
6 =	File name for additional information	e.g. in HTML format

Example ASM5 Read the additional text relating to the system error with message number 74 of device address 01.

FI command		01_BR_ASM5_74_0
Line	**Column**	**Answer**
1	1	01
	2	Drill center
	3	74
	4	F
	5	Replace the SLM module on the PLC card (slot 3).
	6	

Reference to Literature See chapter entitled “Literature” [13].

Reading and Writing CMOS RAM ASCII Parameters: CMA

MWAX device group

Designation **CMA** **CM**OS RAM **A**SCII Parameter

Explanation CMOS RAM ASCII parameters can be read and written.

FI command Read CMOS RAM ASCII parameters.

CR_CMA_(1) **(Single Read)**

(1) = CMOS RAM ASCII parameter numbers [0..79]

Response Structure One line with one column is output for the value of the selected CMOS RAM ASCII parameter.

Example Read CMA Parameter Read the value of the CMOS RAM ASCII parameter with the number 0 at device address 00.

FI command		00_CR_CMA_0
Line	**Column**	**Answer**
1	1	Waiting for tool change

FI command Write CMOS RAM ASCII parameters.

CW_CMA_(1) **(Single Write)**

(1) = CMOS RAM ASCII parameter numbers [0..79]

Value to be written Value of the parameter [ASCII characters]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure The return value of the "DataTransfer" routine is [0] if the write procedure has been successfully completed. In the event of an error, more information in the form of a general error result line can be requested by the routine "ReadGroupItem" (also refer to the chapter “Error Codes” and “General Error Result Line”).

Example Write CMA Parameter Write "Waiting for tool change" in the CMOS RAM ASCII parameter numbered 0 at device address 00.

FI command	00_CW_CMA_0
Value to be written	Waiting for tool change

Reading and Writing CMOS RAM Floating Point Parameters: CMF

MWAX device group

Designation **CMF** **CM**OS RAM **F**loatingpoint Parameter

Explanation CMOS RAM Floating Point parameters can be read and written.

FI command Read CMOS RAM Floating Point parameters.

CR_CMF_(1) **(Single Read)**

(1) = CMOS RAM Floating Point parameter numbers [0..79]

Response Structure One line with one column is output for the value of the selected CMOS RAM Floating Point parameter.

Example Read CMF Parameters Read the value of the CMOS RAM Floating Point parameter numbered 1 at device address 00.

FI command		**00_CR_CMF_1**
Line	**Column**	**Answer**
1	1	4711.0123

FI command Write CMOS RAM Floating Point parameters.

CW_CMF_(1) **(Single Write)**

(1) = CMOS RAM Floating Point parameter numbers [0..79]

Value to be written Value of the parameter [Type: floating point]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure The return value of the "DataTransfer" routine is [0] if the write procedure has been successfully completed. In the event of an error, more information in the form of a general error result line can be requested by the routine "ReadGroupItem" (also refer to the chapter “Error Codes” and “General Error Result Line”).

Example Write CMF Parameter Write the value [4711.0123] in the CMOS RAM Floating Point parameter numbered 1 at device address 00.

FI command	**00_CW_CMF_1**
Value to be written	4711.0123

Read and Write CMOS RAM Integer Parameters: CMI

MWAX device group

Designation **CMI** **CM**OS RAM **I**nteger Parameter

Explanation CMOS RAM Integer parameters can be read and written.

FI command Read CMOS RAM Integer parameters.

CR_CMI_(1) **(Single Read)**

(1) = CMOS RAM integer parameter numbers [0..79]

Response Structure One line with one column is output for the value of the selected CMOS RAM integer parameter.

Example Read CMI Parameters

Read the value of the CMOS RAM Integer parameter numbered 2 at device address 00.

FI command		**00_CR_CMI_2**
Line	**Column**	**Answer**
1	1	120270

FI command

Write CMOS RAM Integer parameters.

CW_CMI_(1) **(Single Write)**

(1) = CMOS RAM integer parameter numbers [0..79]

Value to be written

Value of the parameter [Type: integer]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure

The return value of the "DataTransfer" routine is [0] if the write procedure has been successfully completed. In the event of an error, more information in the form of a general error result line can be requested by the routine "ReadGroupItem" (also refer to the chapter “Error Codes” and “General Error Result Line”).

Example Write CMI Parameter

Write the value [120270] in the CMOS RAM Integer parameter numbered 2 at device address 00.

FI command	**00_CW_CMI_2**
Value to be written	120270

Trigger Control Reset: CRT

MWAX device group

Designation **CRT** **Control-Reset**

Explanation

The control reset allows the selected device to be reset after a system error. If there is no system error at the selected device then the job is ignored.

CAUTION

Carrying out a reset completely re-initializes the device.
During initialization, communication is temporarily interrupted (inherent to design).

FI command **CW_CRT** **(Single Write)**

Value to be written Trigger reset 0

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Response Structure

The return value of the "DataTransfer" routine is [0] if the write procedure has been successfully completed. In the event of an error, more information in the form of a general error result line can be requested by the routine "ReadGroupItem" (refer here to chapter 8, “Error Codes” and “General Error Result Line”).

Example CRT Trigger a control reset on the selected device.

FI command	**00_CW_CRT**
Value to be written	0

Reference to Literature See chapter entitled “Literature” [26].

Device Configuration Parameter: DCP

MWAX device group

Designation **DCP** **D**evice **C**onfiguration **P**arameter

Explanation The device configuration parameters that are entered in the active machine parameter record as well as in the "IND_DEV.INI" file are output. The configuration parameters of the device include the device address, the device name, device type, mechanism number, mechanism name, and the process types.

FI command Output the configuration parameters of all defined devices.

BR_DCP1 **(Single Read)**

Note: The “DCP1” FI command refers to all defined devices. Therefore, any valid device address can be indicated in the command line (see example DCP1).

Response Structure The following table shows the general structure of the response to the “DCP1” FI command. The response consists of a maximum of n=512 lines (n=16 devices x 32 mechanisms = 512), each with 7 columns.

Line 1...n:	**Column 1**	**...**	**Column 7**

Note: If no active machine parameter record exists in the device, then the columns [1...7] for the respective device are not applicable.

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	Device name	[max. 32 ASCII characters]
3 =	Device Type	[MTC200-P-G2, MTC200-R-G2, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process type	[1= internal, 2 = external process]
7 =	Process type	[1 = NC process, 2 = PLC process]

Example DCP1 Read the device configuration parameters of all defined devices.

Assumption:
Three devices have been defined

- Device address 00 (MTC200-R-G2)
- Device address 01 (MTC200-P-G2) and
- Device address 02 (MTC200-P-G2)

FI command		00_BR_DCP1
Line	**Column**	**Answer**
1	1	00
	2	Rotary transfer machine
	3	MC200-R-G2
	4	1
	5	Master
	6	1
	7	2
2	1	01
	2	0
	3	MTC200-P-G2
	4	0
	5	Milling machine 01
	6	1
	7	1
3	1	02
	2	0
	3	MTC200-P-G2
	4	1
	5	Milling machine 02
	6	1
	7	1

FI command Output the configuration parameters of the selected device.

BR_DCP2 **(Single Read)**

Response Structure The following table shows the general structure of the response to the “DCP2” FI command. The response consists of a line with 7 columns.

Line 1	Column 1	...	Column 7

Note: If no active machine parameter record exists in the device, then the columns [1...7] for the respective device are not applicable.

Value Range/Meaning of Columns

1 =	Device address	[00...15]
2 =	Device name	[max. 32 ASCII characters]
3 =	Device Type	[MTC200-P-G2, MTC200-R-G2, MTVNC, MTRA-P, MTRA-R]
4 =	Mechanism number	[0...31]
5 =	Mechanism name	[max. 28 ASCII characters]
6 =	Process type	[1= internal, 2 = external process]
7 =	Process type	[1 = NC process, 2 = PLC process]

Example DCP2

Read the device configuration parameters of the selected device (device address 01).

Assumption:
Three devices have been defined

- Device address 00 (MTCNC)
- Device address 01 (MTC200-P)
- Device address 02 (MTC200-P)

FI command		**01_BR_DCP2**
Line	**Column**	**Answer**
1	1	01
	2	0
	3	MTC200-P
	4	0
	5	Milling machine 01
	6	1
	7	1

Reference to Literature

See chapter entitled “Literature” [28].

Setting the Communication Timeout Time DCT

MWAX device group

Designation

DCT **D**evice **C**ommunication **T**imeout

Explanation

By means of this command, the timeout time for the selected device is set dynamically (timeout time in ms).

FI command

BW_DCT1_(1) **(Single Write)**

(1) = requested timeout time in ms

Response Structure

The response to the "DCT1" FI command consists of one line with one column.

Line 1	**Column 1**

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCT1

For the device 00, the timeout time is set 1500 ms.

FI command		**00_BW_DCT1_1500**
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

With this command, the timeout time for the selected device can be reset to default value.

BW_DCT2 **(Single Write)**

Response Structure

The response to the "DCT2" FI command consists of one line with one column.

Line 1	**Column 1**

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCP2 For the device 00, the timeout time is reset to the default value.

FI command		00_BW_DCT2
Line	Column	Answer
1	1	(P_ACK)

Long ID of PLC Data Block: DIS

MWAX device group

Designation **DIS** **D**ata **I**dentification **S**tring

Explanation Reads the long identification (directory entries) of MTA200/PLC data records. Included in the directory entries are the number of the entry in the directory, the name, length and date and time of creation and/or details of the last time the respective data record was changed. The long identifications of the following MTA200/PLC data records are output:

- MTA200 parameter record (FI command: DIS1)
- PLC program (FI command: DIS2)

FI command Output the directory entries of the valid NC parameter record in the selected device.

BR_DIS1 **(Single Read)**

BC_DIS1 **(Cyclic Read)**

BB_DIS1 **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the “DIS1” FI command. The response consists of a line with five columns.

Line 1	Column 1	...	Column 5

Value Range/Meaning of Columns

1 =	Number in MTA200 parameter directory	[01...99]
2 =	Name of the MTA200parameter record	[max. 32 ASCII characters]
3 =	Length of the MTA200 parameter record	[byte]
4 =	Date of creation/last change to MTA200 parameter record	[DD.MM.YY]
5 =	Time of creation/last change to MTA200 parameter record	[HH:MM:SS]

Note: If there is no valid MTA200 parameter record in the selected device then all columns contain [--] . This command can also be used when the selected device is in OFFLINE mode (DeviceStatus=OFF).

Example DIS1

Read the directory entries of the MTA200 parameter record at device address 00.
Assumption:
There is a valid MTA200 parameter record in the selected device.

FI command		00_BR_DIS1
Line	Column	Answer
1	1	01
	2	KEY1
	3	3579
	4	16.05.99
	5	10:41:08

Reference to Literature

See chapter entitled “Literature” [29].

FI command

BR_DIS2 (Single Read)
BC_DIS2 (Cyclic Read)
BB_DIS2 (Break Cyclic Read)

Response Structure

The following table shows the general structure of the response to the “DIS2” FI command. The response consists of a line with six columns.

Line 1	Column 1	...	Column 6

Value Range/Meaning of Columns

1 =	Number in PLC directory	[01...99]
2 =	Name of the PLC program	[max. 8 ASCII characters]
3 =	Length of the PLC program	[byte]
4 =	Date of creation/last change to PLC program	[DD.MM.YY]
5 =	Time of creation/last change to the PLC program	[HH:MM:SS]
6 =	Date of creation/last change to PLC program	[DD.MM.YYYY]

Note: If there is no valid NC package in the selected NC memory then all columns contain [--] .

Example DIS2

Read the directory entries of the PLC program at address 00.
Assumption:
There is a valid PLC program in the selected device.

FI command		00_BR_DIS2
Line	Column	Answer
1	1	01
	2	KEY1
	3	20018
	4	10.05.99
	5	12:42:00
	6	10.05.1999

Reference to Literature

see chapter entitled “Literature” [30].

Reading the Device Status Information: DSI

MWAX device group

Designation **DSI** **D**evice **S**tatus **I**nformation

Explanation This enables the most important device status information to be read. The following information is returned:

Type of information	Status	Statement
System error information		
Information on mechanism error		
Machine key information	valid	Yes/No
Machine key information		
Machine status information		
Sercans information		
Parameter download	running	Yes/No
PLC download	running	Yes/No
Firmware download	running	Yes/No
Offline/Online information		
Device simulation	switched on	Yes/No
Device status information		ON/ OFF

FI command Read out device status information for ALL defined devices.

BR_DSI1 **(Single Read)**

BC_DSI1 **(Cyclic Read)**

BB_DSI1 **(Break Cyclic Read)**

Note: The "DSI1" FI command refers to all devices within this device group. Therefore, any valid device address can be indicated in the command line (see example DSI1). The FI device polling mechanism **MUST** be switched on (see system configurator)!

Response Structure The following table shows the general structure of the response to the "DSI1" FI command.

Line 1...n	Column 1	...	Column 11

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Information on mechanism error	[0 = there is no mechanism error 1 = there is a mechanism error]
4 =	Machine key information	[4 byte in HEX coding]
5 =	Is machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]

10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = Device status=OFF 1 = Device status=ON]

Example DSI1

Read the current device status information.
Assumption:
The following devices addresses are defined:

- Device address 01 (MWCX device)
- Device address 03 (MWSX device)

FI command		**01_BR_DSI1**
Line	**Column**	**Answer**
1	1	01
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1
2	1	03
	2	1
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

FI command Read out device status information for a selected device.

BR_DSI2	**(Single Read)**
BC_DSI2	**(Cyclic Read)**
BB_DSI2	**(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the "DSI2" FI command.

Line 1...n	Column 1	...	Column 11

Value Range/Meaning of Columns

1 =	Device address	[00...63]
2 =	System error information	[0 = there is no system error 1 = there is a system error]
3 =	Information on mechanism error	[0 = there is no mechanism error 1 = there is a mechanism error]
4 =	Machine key information	[4 byte in HEX coding]
5 =	Machine key information valid?	[0 = not valid, 1=valid]
6 =	Machine status information	[4 byte in HEX coding]
7 =	Sercans information	[4 byte in HEX coding]
8 =	Is parameter download active?	[0 = parameter download not running 1 = parameter download running]
9 =	Is PLC download active?	[0 = PLC download not running 1 = PLC download running]
10 =	Is firmware download active?	[0 = PLC download not running 1 = PLC download running]
11 =	Offline/Online information	[0 = device connection interrupted 1 = device connection O.K.]
12 =	Device simulation switched on?	[0 = NO Simulation mode 1 = simulation mode]
13 =	Current device status information	[0 = Device status=OFF 1 = Device status=ON]

Example DSI2 Read the current device status information for the selected device.

FI command		**00_BR_DSI2**
Line	**Column**	**Answer**
1	1	00
	2	0
	3	0
	4	00000000
	5	0
	6	00000000
	7	00000000
	8	0
	9	0
	10	0
	11	1
	12	0
	13	1

Distance to Go of Axis Movement: DTG

MWAX device group

Designation **DTG** **D**istance **T**o **G**o

Explanation The distance to go of the movement of a selected axis is output. The FI command "DTG1" returns the distance to go of an axis, related to the code of the axis meaning. The FI command "DTG2", on the other hand, returns the distance to go of an axis, related to the physical axis number.

FI command Output the distance to go of the selected axis of the device specified, related to the code of the axis meaning.

Using the optional fourth parameter it is possible to pre-select conversion of the result into mm or inches.

CR_DTG1_(1)_(2)_(3){_(4)}	**(Single Read)**
CC_DTG1_(1)_(2)_(3){_(4)}	**(Cyclic Read)**
CB_DTG1_(1)_(2)_(3){_(4)}	**(Break Cyclic Read)**
(1) = NC process number	[0]
(2) = Axis meaning	[1...16, the axis meaning corresponds to the physical axis number]
(3) = System of coordinates	[1 = machine coordinates 2 = program coordinates 3 = relative coordinates]
(4) = Required measurement system (opt.)	[mm, inch]

FI command Output the distance to go of the movement of the selected axis of the device specified related to the physical axis number.

Using the optional third parameter it is possible to pre-select conversion of the result into mm or inches.

CR_DTG2_(1)_(2){_(3)}	**(Single Read)**
CC_DTG2_(1)_(2){_(3)}	**(Cyclic Read)**
CB_DTG2_(1)_(2){_(3)}	**(Break Cyclic Read)**
(1) = Physical axis number	[1..0.16, according to settings of the system parameters]
(2) = System of coordinates	[1 = machine coordinates 2 = program coordinates 3 = relative coordinates]
(3) = Required measurement system (opt.)	[mm, inch]

Response Structure The following table shows the general structure of the response to the FI commands "DTG1" and "DTG2". One line is output with 4 columns for the axis designation, distance to go, unit and the distance to go limited to “indicated decimal places”.

Line 1	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = Axis name	[according to settings of axis parameters]
2 = Distance to go	[according to settings of process parameters]
3 = Unit	[mm, inch]
4 = Distance to go	[as column 2]

Note: If the specified axis or a spindle is not defined in the selected NC process then the answer in all columns is [--].

Example DTG1

Read the distance to go of the movement of the Z axis in machine coordinates in NC process 0 of device address 00.

FI command	00_CR_DTG1_0_3_1			
Answer				
Line 1	Column 1	Column 2	Column 3	Column 4
1	Z	-1.2345	[mm]	-1.2345

Example DTG1

Read the distance to go of the movement of the Z axis in machine coordinates in NC process 0 of device address 00. Values are displayed in inches.

FI command	00_CR_DTG1_0_3_1_inch			
Answer				
Line 1	Column 1	Column 2	Column 3	Column 4
1	Z	-0.0486	[inch]	-0.0486

Example DTG2

Read the distance to go of the movement of the Z axis (physical axis number = 3) in machine coordinates at the device address 00.

FI command	00_CR_DTG2_3_1			
Answer				
Line 1	Column 1	Column 2	Column 3	Column 4
1	Z	-1.2345	[mm]	-1.2345

Reference to Literature

See chapter entitled “Literature” [16].

Device Type and Accompanying Components: DTY

MWAX device group

Designation **DTY** **D**evice **TY**pe

Explanation

The device type and the accompanying components of the selected device address are output.

FI command **BR_DTY1** **(Single Read)**

Response Structure

The following table shows the general structure of the response to the “DTY1” FI command. A line with three columns for the device type is output as well as the name of the first device component and the name of the second device component.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 =	Device Type	(see Chapter “Identifier”)
2 =	Component type1	IND_DEV.INI-Entry: Componenttype1=
3 =	Component type 2	IND_DEV.INI-Entry: Componenttype2=

Example DTY1

Output the device type and the accompanying components of device address 00.

FI command	00_BR_DTY1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	MTA200-P	MTS-P	MTC-P

Diagnosis Window Data: DWD

MWAX device group

Designation **DWD** **D**iagnosis **W**indow **D**ata

Explanation Diagnostic messages are output. The data are edited in such a way that they can be output directly in the diagnosis overview, i.e., where applicable, different types of diagnosis, such as a ProVi message and a process message, are returned simultaneously.

FI command Output all diagnostic messages.

BR_DWD1_(1){_(2)} **(Single Read)**

BC_DWD1_(1){_(2)} **(Cyclic Read)**

(1) = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start preconditions, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Output first diagnostic messages.

BR_DWD2_(1){_(2)} **(Single Read)**

BC_DWD2_(1){_(2)} **(Cyclic Read)**

(1) = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start preconditions, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Response Structure The following table shows the general structure of the "DWD1" and "DWD2" FI commands. The number of lines depends on the number of messages pending. Different columns are valid according to the type of diagnosis.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 12

Meaning of the Columns

1 =	Message text	[ASCII characters]
2 =	Time stamp day	[mm.dd.yyyy]
3 =	Time stamp hour	[hh:mm:ss]
4 =	Reference text available	[YES, NO]
5 =	Type of diagnosis	[1 = ProVi, 2 = SFC, 3 = MTC-NC, 4 = MTA-NC]
6 =	Message number	[ASCII characters]
7 =	Message ID	[ASCII characters] (DWORD, decimal) (ProVi)
8 =	Mechanism number	[0..31] (MTC-NC) [0] (MTA-NC)
9 =	2 byte additional information	[ASCII characters] (MTC NC)
10 =	Message group	[1...9999] (MTA-NC)
11 =	SFC entity name	[ASCII characters]

12 =	NC note	[ASCII characters] (MTC NC)
13 =	Analysis of criteria available	[YES, NO] (ProVi, SFC)
14 =	Message HTML file	[ASCII characters] (ProVi, MTC-NC)

Example DWD1

All diagnostic messages from module 3 in control unit 0.

There are two messages:

FI command		**00_BR_DWD1_4_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	01.27.2000
	3	14:56:32
	4	YES
	5	1
	6	34
	7	43923028
	8	
	9	
	10	
	11	
	12	
	13	YES
	14	
2	1	Station waiting until tool-change command has ended.
	2	01.27.2000
	3	15:03:10
	4	YES
	5	3
	6	79
	7	
	8	1
	9	0
	10	
	11	
	12	
	13	YES
	14	NO

Example DWD2 First diagnostic message from module 3 in control unit 0.

There are two messages:

FI command		00_BR_DWD2_4_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	01.27.2000
	3	14:56:32
	4	YES
	5	1
	6	34
	7	43923028
	8	
	9	
	10	
	11	
	12	
	13	YES
	14	

Reference to Literature See chapter entitled “Literature” [13].

Existing MTA200 Diagnoses: EAD

MWAX device group

Designation **EAD** **E**xisting MT**A**200 **D**iagnosis

Explanation Which MTA200 diagnostic types exist is queried. Depending on the FI command, specific types are queried or else the diagnostic types for one module are output together.

FI command Query which MTA200 diagnostic types are available in a module.

BR_EAD1_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure The following table shows the general structure of the "EAD1" FI command.

Line 1	Column 1-2

Meaning of the Columns

1 = Messages exist [YES, NO]

2 = Errors exist [YES, NO]

Example EAD1 Query the MTA200 diagnostic types in Module 2 on Control unit 0.

FI command		00_BR_EAD1_2
Line	**Column**	**Answer**
1	1	NO
	2	YES

FI command Query a specific MTA200 diagnostic type.

BR_EAD2_(1)_(2)	**(Single Read)**
(1) = Message type	[1 = error, 2 = messages]
(2) = Module number	[1...99]

Response Structure The following table shows the general structure of the "EAD2" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Diagnosis type exists [YES, NO]

Example EAD2 Are there any messages in module 4 in control unit 0?

FI command		00_BR_EAD2_2_4
Line	**Column**	**Answer**
1	1	YES

Existing errors: EDE

MWAX device group

Designation **EDE** **E**xisting **D**iagnosis **E**rror

Explanation Whether or not errors exist in a control unit or in a module is queried. These can be sequencer errors, NC errors, MTA200 errors or ProVi errors.

FI command Query whether there are errors in this control unit.

BR_EDE1	**(Single Read)**
BC_EDE1	**(Cyclic Read)**

Response Structure The following table shows the general structure of the "EDE1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Error exists [YES, NO]

Example EDE1 Do errors exist in control unit 0?

FI command		00_BR_EDE1
Line	**Column**	**Answer**
1	1	YES

FI command Query whether or not errors exist in a specific module.

BR_EDE2_(1)	**(Single Read)**
BC_EDE2_(1)	**(Cyclic Read)**
(1) = Module number	[1...99]

Response Structure The following table shows the general structure of the "EDE2" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Error exists [YES, NO]

Example EDE2 Do errors exist in Module 1 on Control unit 0?

FI command		00_BR_EDE2_2
Line	Column	Answer
1	1	NO

Existing Diagnosis Window: EDW

MWAX device group

Designation **EDW** **E**xisting **D**iagnosis **W**indow

Explanation Which types of diagnosis window exist is queried.

FI command Output all types of diagnosis window.

BR_EDW1 **(Single Read)**

Response Structure The following table shows the general structure of the "EDW1" FI command. The number of lines depends on the number of types of window existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start preconditions, 11 = warnings, 12 = setup diagnosis]

2 = Module number [ASCII characters]
0 = Diagnosis window type does not belong to any module

Example EDW1 All types of diagnosis window in control unit 0.

There are three diagnosis windows:

FI command		00_BR_EDW1
Line	Column	Answer
1	1	10
	2	0
2	1	1
	2	3
3	1	2
	2	3

FI command Output all diagnosis window types for a module.

BR_EDW2_(1) **(Single Read)**

(1) = Module number [1...99]

Response Structure The following table shows the general structure of the "EDW2" FI command. The number of lines depends on the number of types of window existing.

Line 0...n	Column 1	Column 2

Meaning of the Columns

1 = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages]

2 = Module number [ASCII characters]
0 = Diagnosis window type does not belong to any module

Example EDW2 All types of diagnosis window in Module 3, Control unit 0.

There are two diagnosis windows.

FI command		**00_BR_EDW2_3**
Line	**Column**	**Answer**
1	1	1
	2	3
2	1	2
	2	3

FI command Query a specific type of diagnosis window.

BR_EDW3_(1){_(2)} **(Single Read)**

(1) = Type of diagnosis window [1 = NC error, 2 = sequence errors, 3 = general errors, 4 = messages, 10 = start preconditions, 11 = warnings, 12 = setup diagnosis]

(2) = Module number [1...99] ! only for window type 1 -4 !

Response Structure The following table shows the general structure of the "EDW3" FI command.

Line 1	**Column 1**

Meaning of the Columns 1 = Type of diagnosis window exists [YES, NO]

Example EDW3 Query whether or not a NC error window exists in module 3, control unit 0.

FI command		**00_BR_EDW3_1_3**
Line	**Column**	**Answer**
1	1	YES

Existing ProVi Types: EPT

MWAX device group

Designation **EPT** **E**xisting **P**roVi **T**ypes

Explanation Which ProVi types are programmed in the current PLC program is queried. The data is returned in a suitable form for the message texts of the small control panels. There is no need to define modules in Moduldef.ini.

FI command Output all ProVi types.

BR_EPT1 **(Single Read)**

Response Structure The following table shows the general structure of the "EPT1" FI command. The number of lines depends on the number of ProVi types existing.

Line 0...n	**Column 1**	**Column 2**

Meaning of the Columns 1 = Type [11 = error, 12 = messages, 20 = start requirements, 21 = warnings, 22 = setup diagnosis]

2 = Index [ASCII characters]

Example EPT1

All ProVi types in control unit 0.

There are three diagnosis windows.

FI command		00_BR_EPT1
Line	**Column**	**Answer**
1	1	20
	2	0
2	1	11
	2	3
3	1	12
	1	3

Error Status: EST

MWAX device group

Designation

EST **E**rror **ST**ate

Explanation

Queries the error state of a variable.

FI command

Query the frozen error state of a variable.

BR_EST1!(1)!(2) **(Single Read)**

BC_EST1!(1)!(2) **(Cyclic Read)**

(1) = Error ID [ASCII characters] (DWORD, decimal)

(2) = Variable name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "EXD1" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Error state

WinPcl - Example EST

Read the value of WinPcl variable "IB_EXT24" in WinPcl program "Prog", at device address 00.

Suggestion:
The WinPcl variable "IB_EXT24" is declared as BOOL in the WinPcl program "Prog".

FI command		00_BR_EST1!5892855!:Prog.IB_EXT24
Line	**Column**	**Answer**
1	1	1

Execution Display: EXD

MWAX device group

Designation **EXD** **EX**ecution **D**isplay

Explanation Information for displaying the execution of a movement is output.

FI command Query the execution of a step or of an action.

BR_EXD1!(1)!(2)!(3)	**(Single Read)**
BC_EXD1!(1)!(2)!(3)	**(Cyclic Read)**
(1) = SFC entity name	[ASCII characters]
(2) = Step or action name	[ASCII - characters]
(3) = Behaviour of mode	[1 – all modes, 2 – manual mode]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "EXD1" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Execution [1 – can be executed, 0 – cannot be executed]

Example EXD1 Query the execution of the step “open” for the chain “clamp” in control unit 0 for all modes.

FI command		00_BR_EXD1!Station03A.Clamp!Open!1
Line	**Column**	**Answer**
1	1	1

FI command Query whether the condition analysis (control image) of a step chain is enabled.

BR_EXD2!(1)	**(Single Read)**
(1) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the "EXD2" FI command.

Line 1	Column 1

Meaning of the Columns 1 = Enabled [1 - enabled,
0 – not enabled]

Example EXD2 Query whether the condition analysis of the “clamp” chain has been enabled.

FI command		00_BR_EXD2!Station03A.Clamp
Line	**Column**	**Answer**
1	1	1

Module Assignment of a Process: MAP

MWAX device group

Designation **MAP** **M**odul **A**ssign of **P**rocess

Explanation The module to which a particular process is assigned is read from the "Moduldef.ini'" file. This file is located in the directory "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" and contains the data for all module configurations. The process data is located in three sections:

[DeviceAddrX\ModulY\Process]

whereby "X" stands for the device addressed and "Y" for the configured module numbers.

FI command Determine the module to which the process belongs. Information is read out from the module configuration of the MWAX device group.

BR_MAP1_(1) **(Single Read)**

BC_MAP1_(1) **(Cyclic Read)**

BB_MAP1_(1) **(Break Cyclic Read)**

1 = Mechanism number [0]

Response Structure The following table shows the general structure of the response to the “MAP1” FI command. One line with one column is output for the module number that has been determined.

Line 1	Column 1

Value Range of the Column 1 = Module number [0...99]

Example MAP1 Read the module number which is assigned to NC process number 0 from the module configuration.

Assumption:
The module to which NC process 0 is assigned has module number 5.

FI command		00_BR_MAP1_0
Line	**Column**	**Answer**
1	1	5

Reference to Literature See chapter entitled “Literature” [36].

Read Reference Name of a PLC Variable: MAR

MWAX device group

Designation **MAR** **M**ap **A**bsolute PCL-**R**eference

PLC Explanation The absolute reference name of a symbolic PLC variable is read out.

FI command Read the absolute reference name of a PLC variable.

BR_MAR_(1) **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command "MAR". One line with one column is output for the reference name that has been determined.

Line 1	Column 1

Meaning of the Column 1 = Identifier of the PLC variable

PLC – Example MAR

Read the absolute reference name of the PLC variable with the identifier “abref” at device address 00.

Assumption:
The PLC variable with the identifier "abref" is of the type "INTEGER".

FI command		00_BR_MAR_abref
Line	**Column**	**Answer**
1	1	%M100.0

WinPlc Explanation

The absolute reference name of a symbolic WinPlc PLC variable with program entity is read out.

FI command

Read the absolute reference name of a WinPlc PLC variable.

BR_MAR1_(1) **(Single Read)**

(1) = Identifier of the PLC variable

WinPLC - Example MAR1

Read the absolute reference name of the WinPLC variable with the identifier "Prog.abref" at device address 00.

Assumption:
The WinPLC variable with the identifier "Prog.abref" is of the type "INTEGER" and is present in WinPLC program "Prog".

FI command		00_BR_MAR1_:Prog.abref
Line	**Column**	**Answer**
1	1	%M100.0

Reference to Literature

See chapter entitled “Literature” [30].

Device Data of the Module Configuration: MCD

MWAX device group

Designation

MCD **M**odul **C**onfiguration: **D**evice Information

Explanation

All device data for module configuration is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory on standard installation. The device data is in the sections [DeviceAddrX], whereby "X" stands for the configured device addresses.

FI command

Read-out device data within the module configuration of the MWAX device group.

BR_MCD1 **(Single Read)**

BC_MCD1 **(Cyclic Read)**

BB_MCD1 **(Break Cyclic Read)**

Note: The “MCD1” FI command refers to all devices within the MWAX device group. Therefore, any valid device address can be indicated in the command line (see example “MCD1”).

Response Structure

The following table shows the general structure of the response to the "MCD1" FI command. The number of lines depends on the number of configured devices. Each line consists of four columns for the device address as well as PLC-FB names for providing setup diagnostics, warning messages and start requirements.

Line 1	Column 1	...	Column 4

Value Range of the Columns

1 = Device address	[0...15]
2 = PLC-FB name for the setup diagnostics	[max. 9 ASCII characters]
3 = PLC-FB name for the warning messages	[max. 9 ASCII characters]
4 = PLC-FB name for the start requirements	[max. 9 ASCII characters]

Example MCD1

Read all device data of the module configuration

Assumption:
The following devices have been configured in the MWAX device group:

- Device address 01 (MTA200-P)
- Device address 03 (MTA200-R)

FI command	**03_BR_MCD1**			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	01	PVSetup_1	PVWarn_1	PVStart_1
2	03	PVSetup_3	PVWarn_3	PVStart_3

Reference to Literature

See chapter entitled "Literature" [36].

Module Data of the Module Configuration: MCM

MWAX device group

Designation **MCM** **M**odul **C**onfiguration: **M**odul Information

Explanation

All module data for module configuration is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. This file contains all module configuration data. The module data is located in sections [DeviceAddrX\ModulY], whereby "X" stands for the device addressed and "Y" for the configured module numbers.

FI command

Read module data from the module configuration with respect to a device from the MWAX device group.

BR_MCM1	**(Single Read)**
BC_MCM1	**(Cyclic Read)**
BB_MCM1	**(Break Cyclic Read)**

Response Structure

The following table shows the general structure of the response to the "MCM1" FI command. The number of lines depends on the number of configured modules of a device. Each line consists of four columns for the module number, module name and PLC-FB names for general module errors and module messages.

Line 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 general module errors	[max. 9 ASCII characters]

4 = PLC-FB name for module messages [max. 9 ASCII characters]

Example MCM1

Read the module data of device 03 from the module configuration:

Assumption:
The following modules have been defined:

- Module number 5
- Module number 7

FI command	03_BR_MCM1			
Answer				
Line	**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

Reference to Literature

See chapter entitled “Literature” [36].

Process Data of the Module Configuration: MCP

MWAX device group

Designation

MCP **M**odul **C**onfiguration: **P**rocess Information

Explanation

All process data of a certain module is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. This file contains all module configuration data. The process data is located in sections [DeviceAddrX\ModulY\Process], whereby "X" stands for the device addressed 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]

Response Structure

The response to the FI command “MCP1” consists of one up to a maximum number of n=32 lines with 1 column for the number of the NC process or of the external mechanisms.

Line 1...32	Column 1

Value Range of the Column

1 = Mechanism number [0]

Example MCP1

Read the NC process number of module 5 of device 03 of the module configuration.

Assumption:
The following NC processes are defined:

- NC process number 0

FI command		00_BR_MCP1_5
Line	**Column**	**Answer**
1	1	1
2	1	0

Reference to Literature

See chapter entitled “Literature” [36].

SFC Data of the Module Configuration: MCS

MWAX device group

Designation **MCS** **M**odul **C**onfiguration: **S**FC Information

Explanation All SFC data of a certain module is read out from the "Moduldef.ini" file which is stored in the "[LW]:\Programme\Indramat\MTGUI\CustomData\Resource" directory following standard installation. This file contains all module configuration data. The SFC data is located in sections [DeviceAddrX\ModulY\Sfc], whereby "X" stands for the device addressed and "Y" for the selected module number.

FI command Read the SFC data with respect to the module of a device from the module configuration of the MWAX device group.

BR_MCS1_(1) **(Single Read)**

BC_MCS1_(1) **(Cyclic Read)**

BB_MCS1_(1) **(Break Cyclic Read)**

(1) = Module number [0...99]

Response Structure The number of lines depends on the number of configured Indrastep step chains for a device. Each line contains a column for the name of the Indrastep step chains.

Value Range of the Column 1 = Name of the Indrastep step chain [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 name of the Indrastep step chain of module 5 from device 03 of the module configuration.

Assumption:
The following Indrastep step chains have been defined:

- ISFB_1
- FB_US.ISFB_3
- FB_US.ISFB_3.SW1
- FB_US.ISFB_3.SW1.ABBA

FI command		03_BR_MCS1_5
Line	**Column**	**Answer**
1	1	ISFB_1
2	1	FB_US.ISFB_3
3	1	FB_US.ISFB_3.SW1
4	1	FB_US.ISFB_3.SW1.ABBA

Reference to Literature See chapter entitled “Literature” [36].

Downloading Message Texts: MFD

MWAX device group

Designation **MFD** **M**essage **F**iles **D**ownload

FI command This is used to load the message texts into the device indicated. These message texts are required for small devices. The following message texts are transmitted, depending on the type of device:

- system error messages
- transmission error messages
- mechanism messages

Note: This FI command is an FI job!

BW_MFD1 **(Single Write)**

Response Structure The response to the "MFD1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

Line 1 = Job ID	[01...20] (refer to chapter entitled "FI Commands for the MPCX Device Group", IFJ).
Line 2 = FI command	[string, in accordance to chapter entitled "Elements of the FI Command"]
Line 3 = FI job error code	(see chapter entitled “Error Codes”)

Example MFD1 Load message texts into the device with device address 00.

FI command		**00_BW_MFD1**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_MFD1
3	1	0

Reading Machine Key Information : MKS

MWAX device group

Designation **MKS** **M**achine **K**ey **S**tatus

Explanation Current machine key information can be read for the selected device.

FI command Read machine key information for selected device.

BR_MKS **(Single Read)**

BC_MKS **(Cyclic Read)**

BB_MKS **(Break Cyclic Read)**

Response Structure The following table shows the general structure of the response to the FI command "MKS".

Line 1	**Column 1**	**Column 2**

Value Range/Meaning of Columns

1 =	Machine key information	[4 byte in HEX coding]
2 =	Information valid?	[0 = not valid, 1=valid]

Example MKS Read the current machine key information for device 0.

FI command		00_BR_MKS
Line	**Column**	**Answer**
1	1	00000000
	2	0

Writing the GUI-SK Block: MKT

MWAX device group

Designation **MKT** **M**achine **K**ey **T**able

Explanation Writes the GUI-SK16 block in the PLC.

FI command Write GUI-SK16 block.

BW_MKT1_(1) **(Single Write)**

(1) = List of the 48 PLC variables for writing the GUI-SK16 block.

The following cases are to be differentiated:
1.Delete the GUI-SK16 block:
2.Write the GUI-SK16 block with the 48 PLC variables, filling gaps with $SPACE.

Response Structure (P_ACK) is returned following successful transmission.

Line 1	Column 1

Value Range/Meaning of the Columns 1 = Successfully completed (P_ACK)

1. Example MKT1 1.Clear GUI-SK16 block:

FI command		00_BW_MKT1 Value to be written: $EMPTY
Line	**Column**	**Answer**
1	1	(P_ACK)

2. Example MKT1 Write GUI-SK16 block:

FI command		00_BW_MKT1 Value to be written: $EMPTY SPSVAR1,SPSVAR2,$SPACE,...
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command Write the GUI-SK16 block, writing only those PLC variables which are defined in the current PLC program. All undefined PLC variables are automatically replaced by $SPACE and returned as a partial result (column 2).

BW_MKT2_(1) **(Single Write)**

(1) = List of the 48 PLC variables for writing the GUI-SK16 block.

The following cases are to be differentiated:
1. Delete the GUI-SK-16 block:
BW_MKT2 $EMPTY
2.Write the GUI-SK16 block with the 48 PLC variables, filling gaps with $SPACE:
BW_MKT1 SPSVAR1,SPSVAR2, $SPACE,$SPACE,....

Response Structure

After successful transmission, one line with two columns is returned.

Line 1	Column 1	Column 2

Value Range/Meaning of Columns

1 =	Status report	[0 = at least 1 PLC variable in the current PLC program is NOT defined 1 = ALL PLC variables could be written]
2 =	List of the NON-defined PLC variables in the current PLC program	[-- = ALL PLC variables could be written, or else list of the PLC variables that could not be written.] The individual PLC variables are separated by a comma.

Example MKT1

Write GUI-SK16 component with 48 PLC variables, while the PLC variables SPSVAR11 and SPSVAR12 are NOT defined in the current PLC program.

FI command		**00_BW_MKT1** **Value to be written:** **SPSVAR1,SPSVAR2,...SPSVAR48**
Line	**Column**	**Answer**
1	1	(P_ACK)

Extended information

The variables are divided into 3 groups of 16 variables each and have the following meaning:

1. Variables 1 - 16: Machine function keys
2. Variables 17 - 32: Status pressed
3. Variables 33 - 48: Status shining

Note: When, for example, only the first 8 M keys are used, the telegram will contain only these 8 PLC variables. The other 40 variables need not be defined in the transmission parameter.

When certain areas, e.g. of M keys, are left unused, they must be filled up with '$SPACE' up to the next variable.

Read System Messages: MSG

MWAX device group

Designation **MSG** **M**e**S**sa**G**e

Explanation Reading of system messages

FI command Message

CC_MSG_(1) **(Cyclic Read)**

(1) = SYS-Message number

Note: Exists only as a cyclic command

Response Structure

The response of the FI command 'MSG' consists of the system message data.

Example MSG 00_CC_MSG_64 (64 = MSG_SYSERRGEN)

FI command		**00_CC_MSG_64/3**
Line	**Column**	**Answer**
1	1	00

Restriction

- The following system messages:

SYS Message	SYS Message number
MSG_PCLUPDBEG	52
MSG_PARUPDBEG	24
MSG_FWAUPDBEG	82

These commands cannot be used with the following programs:

- Indramat OPC server
- Indramat DDE server

Reading the Firmware Identification: MTC

MWCX device group

Designation **MTC** **MT**-**C**NC Slot Software Version

FI command This command is used to read the firmware identification from the various control components (slot numbers).

Note: For the time this FI command is executed, the internal FI communication interlocks (fast timeout monitoring, offline operation, etc.) are switched off.

FI command

BR_MTC_(1) **(Single Read)**

(1) = Slot number [1=CNC, 2=SIO, 3=PLC, 4=APR1 5=APR2, 6=APR3, 7=APR4]

Response Structure The following table shows the general structure of the response to the FI command "MTC". A line of 1 column is output.

Line 1	**Column 1**

Value Range/Meaning of Columns 1 = Firmware identification string [max. 16 ASCII characters]

Example MTC Read the firmware identification of slot number 1 (CPU) of device 00.

FI command		**00_BR_MTC_1**
Line	**Column**	**Answer**
1	1	CPU01/0004-20V00

Parameter Download: PAA

MWAX device group

Designation **PAA** **PA**rameter **A**ccess

Explanation Complete parameter records are downloaded by means of a download file.

FI command Parameter download command whereby the parameter download file is directly indicated.

BW_PAA2_(1) **(Single Write)**

(1) = Complete parameter download file name

Response Structure The response to the "PAA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Note: File and path details must be enclosed in inverted commas.

Example PAA2 00_BW_PAA2_"D:\DOWNLOAD.DAT"

FI command		**00_BW_PAA2_"D:\DOWNLOAD.DAT"**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PAA2_"D:\DOWNLOAD.DAT"
3	1	0

Structure of Download File The structure of the download file corresponds to that of a Windows Ini file. Indramat's own description in V20_Param_08_Definitions_Parameter_Download_01.doc is recommended for a more detailed account of the structure of the download file.

Summary:

Section [ID_PARAMETER]
Information concerning parameter identification.

Section [ID_SYSTEM]
Information concerning system parameter identification.

Section [DATA_SYSTEM]
Listing of system parameter data.

Section [ID_PROCESSX]
Information concerning process parameter identification.

Section [DATA_PROCESSX]
Listing of process parameter data.

Section [ID_AXISX]
Information concerning axis parameter identification.

Section [DATA_AXISX]
Listing of axis parameter data.

ProVi Diagnosis Data: PDD

MWAX device group

Designation **PDD** **P**rovi **D**iagnosis **D**ata

Explanation Data for ProVi criteria analysis is output.

FI command Output of files to indicate the detail in the editor.

BR_PDD1_(1)_(2){_(3)}	**(Single Read)**
(1) = Message ID	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(3) = Module number	[1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the PDD1 FI command.

Line 1	Column 1	...	Column 5

Meaning of the Columns

1 = POU ID	[ASCII characters]
2 = Detail morpheme	[ASCII characters] (DWORD, decimal)
3 = Error ID	[ASCII characters] (DWORD, decimal)
4 = POU entity name	[ASCII characters]
5 = Nw ID (network ID)	[ASCII characters]

Example PDD1 Indication of data of a ProVi error with ID 43923028 from module 3 in control unit 0.

FI command		00_BR_PDD1_43923028_1_1
Line	**Column**	**Answer**
1	1	STATION_1_2
	2	98243823
	3	34985304
	4	Station2.Module3
	5	43493454

FI command Output the I/O addresses to display a detail.

BR_PDD2_(1)_(2){_(3)}	**(Single Read)**
(1) = Message ID	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(3) = Module number	[1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the PDD2 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Variable morpheme	[ASCII characters] (DWORD, decimal)
2 = I/O address	[ASCII characters]

Example PDD2

Query of the I/O addresses of a ProVi error with ID 43923028 from module 3 in control unit 0.

Three variables have an I/O address.

FI command		00_BR_PDD2_43923028_1_1
Line	**Column**	**Answer**
1	1	98243823
	2	%I3.2.0
2	1	40923423
	2	%Q23.21.7
3	1	34985304
	2	%I100.3.5

FI command

Determine the multilingual comments for displaying a detail.

BR_PDD3_(1)_(2){_(3)} **(Single Read)**

(1) = Message ID [ASCII characters]

(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD3 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Comment morpheme [ASCII characters] (DWORD, decimal)

2 = New comment [ASCII characters]

Example PDD3

Query of the comments for indication of a ProVi error with ID 43923028 from module 3 in control unit 0.

Two comments are replaced by another text.

FI command		00_BR_PDD3_43923028_1_1
Line	**Column**	**Answer**
1	1	98243823
	2	Clamp open
2	1	40923423
	2	Clamp closed

FI command

Query of the status of a certain message

BR_PDD4_(1)_(2){_(3)} **(Single Read)**

(1) = Message number [ASCII characters]

(2) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start preconditions,
12 = setup diagnosis]

(3) = Module number [1...99] ! only for message type 1 -2!

Response Structure

The following table shows the general structure of the PDD4 FI command.

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Message is present [YES, NO]

2 = Criteria analysis exists [YES, NO]

Example PDD4

Query of the status of a ProVi error, number 1001 from module 3 in control 0.

This message is not present at the moment, and there is a criteria analysis.

FI command		00_BR_PDD4_1001_1_1
Line	**Column**	**Answer**
1	1	NO
	2	YES

FI command

Determination of the MessageID of a certain message

BR_PDD5!(1)!(2)!(3)!(4){!(5)} **(Single Read)**

(1) = POU entity name [ASCII characters]

(2) = Nw ID [ASCII characters]

(3) = Message number [ASCII characters]

(4) = Message type [1 = error, 2 = messages, 10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(5) = Module number [1...99] ! only for message type 1 -2!

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the PDD5 FI command.

Line 1-n	Column 1	...	Column 3

Meaning of the Columns

1 = Message ID [ASCII characters] (DWORD, decimal)

2 = Message is present [YES, NO]

3 = Criteria analysis exists [YES, NO]

Example PDD5

Determination of the MessageID of a ProVi error, number 1001 from module 25.40 mm control 0.

Assumption:
This message is not present at the moment, and there is a criteria analysis.

FI command		00_BR_PDD5!Station2.Modul3!43493454!1001!1!1
Line	**Column**	**Answer**
1	1	240872342
	2	NO
	3	YES

Rexroth Indramat

Generating Physical Directory Names: PHD

MWAX device group

Designation **PHD** **PH**ysical **D**irectory

Explanation Generates physical directory names according to the BDI data written.

Note: This is based on BDI philosophy.

FI command Generate physical directory names.

BR_PHD1_(1)_(2)_(3)_(4)_(5)_(6)	**(Single Write)**
(1) = Project ID	[-1= PROJECT_NEUTRAL -2= PROJECT_DEFAULT]
(2) = Section ID	[0= SECT_NEUTRAL 1= SECT_BIN 2= SECT_BASIC_DATA 3=SECT_OEM_DATA 4=SECT_CUSTOM_DATA 5=SECT_PROG_DATA]
(3) = Device address	[-1= DEVADDR_NEUTRAL otherwise the required device address]
(4) = Process ID	[-1= PROCESS_NEUTRAL otherwise the required process number]
(5) =Data type ID	[possible write values see BDI documentation (BDI_DEFINITIONS.H)]
(6) = Language ID	[possible write values see BDI documentation (WINNT.H)]

Response Structure The following table shows the general structure of the response to the FI command "PHD1".

Line 1	Column 1

Value Range/Meaning of Columns

1 =	Physical directory name	[complete physical directory name in accordance with the BDI data written]

Example PHD1 Requesting the physical directory name for:

PROJECT_NEUTRAL
SECT_BIN
DEVADDR_NEUTRAL
PROCESS_NEUTRAL
DATATYPE_NEUTRAL
LANG_NEUTRAL

FI command		**XX_BR_PHD1_-1_0_-1_-1_0_0**
Line	**Column**	**Answer**
1	1	D:\Programme\Indramat\Mtgui\Bin

Actual (Current) Information on Position of All Axes: POI

MWAX device group

Designation **POI** **PO**sition **I**nformation

Explanation The current position information for all axes are read. The FI command "POI1" returns all necessary data for indicating the position.

FI command

BR_POI1_(1){_(2)} **(Single Read)**

BC_POI1_(1){_(2)} **(Cyclic Read)**

BB_POI1_(1){_(2)} **(Break Cyclic Read)**

(1) = updated position information [0...31,
1 = axis has been homed
2 = machine coordinates
4 = program coordinates
8 =relative coordinates
16 = distance to go]
all combinations are possible!

(2) = Required measurement system (opt.) [mm, inch]

Response Structure The following table shows the general structure of the response to the FI command “POI1”. 16 lines with 8 columns are returned for axis type, axis name, axis has been homed, position values in the various systems of coordinates, distance to go, and unit.

Line 1	Column 1	Column 2	Column 3	Column 4	Column 5	Column 6	Column 7	Column 8
Line 2								
...								
Line 16								

Value range/Meaning of lines

Line = axis meaning [1 = A axis, 2 = X axis
3 = Z axis, 4 = Y axis
5 = B axis, 6 = C axis
7 = D axis, 8 = E axis
9 = X' axis, 10 = Y‘ axis
11 = P axis, 12 = Q axis
13 = R axis, 14 = U axis
15 = V axis, 16 = W axis]

Value Range/Meaning of Columns

1 = Axis type [0 = axis not defined
1 = Linear axis 2 = rotary axis
3 = Modulo axis
4 = main spindle]

2 = Axis name [according to settings of axis parameters]

3 = Axis has been homed [0 = axis has not been homed
1 = axis has been homed]

4 = Machine coordinates

5 = Program coordinates

6 = Relative coordinates

7 = Distance to go

8 = Required measurement system (opt.) [mm, inch]

Note: If an axis is not defined then the response in all columns is [--].

Example POI1

Read for all axes: axis type, axis name, machine coordinates, program coordinates, distance to go, and unit. Values are displayed in the basic measurement system.

Assumption:
The axes X, Y, Z, C, B and X' are defined.

FI command	00_BR_POI1_22							
Answer								
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**	**Column 5**	**Column 6**	**Column 7**	**Column 8**
1	0	--	--	--	--	--	--	--
2	1	X	--	148.0000	123.0000	--	0.0000	[mm]
3	1	Z	--	23.4548	56.0000	--	0.0000	[mm]
4	1	Y	--	0.0000	0.0000	--	0.0000	[mm]
5	2	B	--	180.0000	180.0000	--	16.0000	[deg]
6	2	C	--	270.0000	90.0000	--	0.0000	[deg]
7	0	--	--	--	--	--	--	--
8	0	--	--	--	--	--	--	--
9	4	X'	--	0.0000	0.0000	--	0.0000	[%]
10	0	--	--	--	--	--	--	--
11	0	--	--	--	--	--	--	--
12	0	--	--	--	--	--	--	--
13	0	--	--	--	--	--	--	--
14	0	--	--	--	--	--	--	--
15	0	--	--	--	--	--	--	--
16	0	--	--	--	--	--	--	--

Issuing SYS Messages Specific to the PCL: PSM

MWAX device group

Designation **PSM** PCL Sys Message

Explanation

Issues the most important SYS messages regarding the PCL programming interface – required for remote programming.

Note:
The appropriate device address is passed as the write value.

It allows the following SYS messages to be initiated:

- Start of PCL download,
- end of PCL download,
- start of PLC online edit,
- end PLC online edit,
- start of PCL declaration change.
- end of PCL declaration change.

FI command

Issue the most important PCL SYS messages.

BW_PSM1_(1)	**(Single Write)**
(1) =Required SYS message	[1= start of PCL download 2= end of PCL download 3= start of PCL online edit 4= end of PCL online edit

5= start of PCL declaration change
6= end of PCL declaration change]

Value to be written: Device address

Response Structure

The following table shows the general structure of the response to the FI command "PSM1".

Line 1	Column 1	...	Column 8

Value Range/Meaning of Columns

1 =	Status report	[READY=SYS message has been correctly acknowledged by the WIN32 applications] [ERROR=SYS message has NOT been acknowledged by a WIN32 application within the pre-set time]
2 =	Task name (LogInlf name)	[Task name that has triggered the SYS message]
3 =	SYS message number	[contains the issued SYS message number]
4 =	Acknowledgement time	[contains the pre-set acknowledgement time]
5 =	Reference information	[contains, where applicable, the additional information transferred as a write value]
6 =	Length of the reference information	[0 where NO additional information has been transferred]
7 =	Where applicable, LOG channel of the FI that has NOT acknowledged	[-- = acknowledgements have been completed in time or the LOG channel number of the WIN32 application that has NOT acknowledged in time]
8 =	Where applicable, task name that has NOT acknowledged in time.	[-- = acknowledgements have been completed in time or the task name that has NOT acknowledged in time]

Example PSM1

Issue the SYS message Beginning PCL Download. The additional information, device address 00, is also transferred as a write value.

FI command		**XX_BW_PSM1_1 – value to be written: 00**
Line	**Column**	**Answer**
1	1	READY
	2	WINPCL.EXE
	3	14
	4	30000
	5	00
	6	2
	7	--
	8	--

Edit PROVI Message Files: PVA

MWAX device group

Designation **PVA** **P**ROV**I**-Messages **A**ccess

Explanation This write command creates PROVI message files. With this write value, it is possible to decide whether the PROVI messages are to be generated according to the current PLC project, or selectively.

FI command **BW_PVA1** **(Single Write)**

Note: This command is an FI job command.

Value to be written

No write value exists	PROVI message files according to the current PLC project.
Write value exists	List of the requested PROVI message files (separated by a comma) according to the format: [PROVI-Diag-type: module number] Example: 01:01,01:02,02:02

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure The response to the "BW_PVA1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

....

Example PVA1 No write value is passed, i.e. the PROVI message files are generated according to the current PLC project.

FI command		**00_BW_PVA1**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVA1
3	1	0

Explanation The read command returns the most significant information on the created PROVI message files.

FI command **BR_PVA1** **(Single Read)**

....

Response Structure The following table shows the general construction of the answer of the FI command BR_PVA1. For each available PROVI message file, 1 line with 10 columns each is created.

Line 1...n	**Column 1**	**...**	**Column 10**

Value Range/Meaning of Columns

1 =	PROVI diagnostic type	[1..20]
2 =	PROVI diagnosis type designation	[The following designations can be returned: StartCondition, Error, Message, Warning, Setup]
3 =	Module number	[1..99]
4 =	PROVI diagnosis type and module number	[PROVI diagnosis type: module number, see write value for BW_PVA2]
5 =	Complete name of the PROVI message text file	[max. 200 ASCII characters]
6 =	Memory required for PROVI messages in the control	[figure in ASCII format]
7 =	Complete name of the PROVI index file	[max. 200 ASCII characters]
8 =	Memory required for PROVI index files in the control	[figure in ASCII format]
9 =	Total memory (text+index) required in the control	[figure in ASCII format]
10 =	Total memory for ALL PROVI files (text+index) required in the control	[figure in ASCII format]

Example PVA1

The most significant information of 2 available PROVI message files are returned.

FI command		**00_BR_PVA1_1**
Line	**Column**	**Answer**
1	1	1
	2	Error
	3	1
	4	01:01
	5	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	6	1345
	7	D:\Programme\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.IDX
	8	234
	9	1579
	10	4491
2	1	2
	2	Message
	3	1
	4	02:01
	5	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	6	2456

FI command		00_BR_PVA1_1
Line	Column	Answer
	7	D:\Programs\indramat\Mtgui\Project_000\ Programmdata\Device_000\Diag\De\ERROR 1.TXT
	8	456
	9	2912
	10	4491

Explanation

This write command transmits PROVI message files into the selected device. Through the write value, it is possible to chose whether ALL or only the PROVI messages selected via the write value are to be transmitted.

FI command

BW_PVA2 **(Single Write)**

Note: This command is an FI job command.

Value to be written

No write value exists	All PROVI message files are transmitted into the selected device
Write value exists	List of the requested PROVI message files (separated by a comma) according to the format: [PROVI-Diag-type: module number] Example: 01:01,01:02,02:02

Note: The value to be written is passed to the "acValue" parameter as an ASCII value in the "DataTransfer" routine.

Response Structure

The response to the "BW_PVA2" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

....

Example PVA2

No write value is passed, i.e. all PROVI message files should be transmitted.

FI command		00_BW_PVA2
Line	Column	Answer
1	1	01
2	1	00_BW_PVA2
3	1	0

Formatted Input / Output of PLC Variables: PVF

MWAX device group

Designation **PVF** **P**LC **V**ariable **F**ormatted

Explanation Formatted reading and writing of PLC variables, arrays and structures.

FI command Read PLC variables.

CR_PVF_(1)	**(Single Read)**
CC_PVF_(1)	**(Cyclic Read)**
CB_PVF_(1)	**(Break Cyclic Read)**
(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]

Response Structure One line with one column is output for single variables. For array and structure variables, one line per element is output, depending on the number of elements.

Line 1...n:	Column 1

n = number of elements.

Note: Only defined PLC variables can be read and written. Addressing a non-declared variable results in an error message. A PLC variable can only be read if its data length does not exceed 240 byte. (Refer also to chapter on “Programming” and “Guidelines”).

Value Ranges ANSI / ASCII The value range of the response depends on the data type of the variable read. The following table indicates the range in which the results string is to be expected when reading out a single variable and into which C-data type this string can be converted without loss of information:

Data Type	Value Range	Can be converted to C-data type
BOOL	[0;1]	unsigned char
SINT	[-128...127]	char
INT	[-32768...32767]	short
DINT	[2147483648...2147483647]	long
USINT	[0...255]	unsigned char
UINT	[0...65535]	unsigned short
UDINT	[0...4294967295]	unsigned long
BYTE	[0x00...0xFF]	unsigned char
WORD	[0x0000....0xFFFF]	unsigned short
DWORD;	[0x00000000...0xFFFFFFFF]	unsigned long
TIME	[0...4294967295]	unsigned long (msec)
CHAR	[$00...$20,!...~,$7F...$FF]	char
STRING	<String> whereby <String> string is a character string with a maximum of as many characters as are declared for the string in the PLC	Char[xx+1]] +1 i.e. room for the zero byte
REAL	[-3.402823567E+38...3.402823567E+38]	Float

Note: An empty string is identified by two single inverted commas: ' ' (do not confuse with the double inverted commas ")!

All single variables can be part of array and structure variables. The value ranges maintain their validity, even when within structured data types.

Binary Value Range

The value range of the response depends on the data type of the variable read. The following table indicates the value range in which to expect the binary value of a single variable and how many bytes are included in the binary byte sequence:

Data Type	Value Range	Length (bytes)
BOOL	[00_H...01_H]	1
SINT	[80_H...$7F_H$] i.e. –128...127	1
INT	[8000_H (-32768)...$7FFF_H$ (32767)]	2
DINT	[80000000_H (-2147483648)... $7FFFFFFF_H$ (2147483647)]	4
USINT	[00_H (0)...FF_H (255)]	1
UINT	[00_H (0)...$FFFF_H$ (65535)]	2
UDINT	[0...4294967295]	4
BYTE	[0x00...0xFF]	1
WORD	[0x0000....0xFFFF]	2
DWORD;	[0x00000000...0xFFFFFFFF]	4
TIME	[0...4294967295]	4
CHAR	[$00...$20,!...~,$7F...$FF]	1
STRING	<String> whereby <String> string is a character string with a maximum of as many characters as are declared for the string in the PLC	XX+1
REAL	[-3.402823567E+38...3.402823567E+38]	4

Note: Binary array and structure elements are joined together without any spaces between (1-byte alignment).

PLC - Example 1 PVF

Read the value of the PLC variable "STK_TXT" in ASCII format from device address 00.

Assumption:
The "STK_TXT" variable is declared as STRING in the PLC program.

FI command		**00_CR_PVF_STK_TXT/1**
Line	**Column**	**Answer**
1	1	Repeat counter

WinPcl - Example 1 PVF

Read the value of WinPcl variable "STK_TXT" in ASCII format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "STK_TXT" is declared in WinPcl program "Prog" as STRING.

FI command		**00_CR_PVF_:Prog.STK_TXT/1**
Line	**Column**	**Answer**
1	1	Repeat counter

PLC - Example 2 PVF

Read the value of the PLC array "BEG_END" in ANSI format from device address 00.

Assumption:
The "BEG_END" variable is declared as BYTE with 2 elements in the PLC program.

FI command		**00_CR_PVF_BEG_END/3**
Line	**Column**	**Answer**
1	1	0x00
2	1	0x1F

WinPcl - Example 2 PVF

Read the value of WinPcl array "BEG_END" in ANSI format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "BEG_END" is declared in WinPcl program "Prog" as BYTE with two elements.

FI command		**00_CR_PVF_:Prog.BEG_END/3**
Line	**Column**	**Answer**
1	1	0x00
2	1	0x1F

PLC - Example 3 PVF

Read the value of the PLC structure "MSTRCT" in ASCII format from device address 00.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		**00_CR_PVF_MSTRCT/1**
Line	**Column**	**Answer**
1	1	0
2	1	A
3	1	ROBOT AXIS X
4	1	2000

WinPcl - Example 3 PVF

Read the value of WinPcl structure "MSTRCT" in ASCII format in WinPcl program "Prog" at device address 00.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CR_PVF_:Prog.MSTRCT/1
Line	**Column**	**Answer**
1	1	0
2	1	A
3	1	ROBOT AXIS X
4	1	2000

FI command

Write PLC variable.

CW_PVF_(1) **(Single Write)**

(1) = Identifier of the PLC variable [acc. to declaration part of the PLC]

Value to be written

Value of data element [see value ranges]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine. The data code of the value is passed to the parameter “ValType”.

Response Structure

One line is output with a column for acknowledgement of whether or not the FI command has been executed successfully.

(P_ACK) = **P**ositive **ACK**nowledge Data element has been set

Value Range of the value to be written in ANSI / ASCII Format

The value ranges agree for the most part with the ANSI / ASCII result-value ranges during read access. ANSI umlauts are thereby converted into ASCII umlauts. Only ASCII umlauts are stored in the control unit. For deviations to this, please refer to the following note:

Note: Strings are enclosed by two single inverted commas ' ' , e.g. 'drill'.

Special characters can be indicated in accordance with DIN-1131 by a $ sign.

The following are used:

- $' '
- $$ $
- $R \r (Carriage Return)
- $L \n (Linefeed)
- $P \f (Form feed)
- $T \t (Tab)
- $xx xx refers to a character written as a hexadecimal value. e.g. $20 (space)

Array and structure elements are separated by a space.

Value Range of the Value to be written in Binary Format

The value ranges agree with the binary result-value range during read access. For deviations to this, please refer to the following note:

PLC - Example 4 PVF

Write into the PLC variable "STK_TXT" at device address 00. The value is passed in ANSI format.

Assumption:
The "STK_TXT" variable is declared as STRING in the PLC program.

FI command		00_CW_PVF_STK_TXT/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	'item counter'
Data code	/3

WinPcl - Example 4 PVF

Write into the WinPcl variable "STK_TXT" in WinPcl program "Prog" at device address 00. The value is passed in ANSI format.

Assumption:
The WinPcl variable "STK_TXT" is declared in WinPcl program "Prog" as STRING.

FI command		00_CW_PVF_:Prog.STK_TXT/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	'item counter'
Data code	/3

PLC - Example 5 PVF

Write into the PLC byte array "BEG_END" at device address 00. The value is passed in ANSI format.

Assumption:
The "BEG_END" variable is declared as a BYTE array with 2 elements in the PLC program.

FI command		00_CR_PVF_BEG_END/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	0x20 0x3f
Data code	/3

WinPcl - Example 5 PVF

Write into the WinPcl byte array "BEG_END" in WinPcl program "Prog" at device address 00. The value is passed in ANSI format.

Assumption:
The WinPcl variable "BEG_END" is declared in WinPcl program "Prog" as BYTE with two elements.

FI command		00_CW_PVF_:Prog.BEG_END/3
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	0x20 0x3f
Data code	/3

PLC - Example 6 PVF

Write the value of element T3 of the PLC structure "MSTRCT" at device address 00. The string “COUNTER” is output in binary format.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CW_PVF_MSTRCT.T3/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	Binary sequence: 43 4F 55 4E 54 45 52 00
Data code	/2

WinPcl - Example 6 PVF

Write the value of element T3 of the WinPcl structure "MSTRCT" at device address 00. The string “COUNTER” is output in binary format.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

FI command		00_CW_PVF_:Prog.MSTRCT.T3/2
Line	**Column**	**Answer**
1	1	(P_ACK)

Value to be written:

Value of data element	Binary sequence: 43 4F 55 4E 54 45 52 00
Data code	/2

PLC - Example 7 PVF

Write the value of the PLC structure "MSTRCT" from the structure “mstrct” previously stored in the C program at device address 00.

Assumption:
The "MSTRCT" variable is declared as a structure in the PLC program as follows:

```
TYP STRUCT
        T1      BOOL
        T2      CHAR
        T3      STRING[16]
        T4      TIME
END
```

To exchange binary data in a C program, the following “C” data type can be used:

```
#pragma pack(1)    //Write all elements
                   //without spaces next to each other.
typeder struct
{
      unsigned char T1;
      char          T2;
      char          T3[17]; //Space for zero byte
      unsigned long T4;
} Tymstrct;        // Declare structure
Tymstrct mstrct;   // Apply structure
```

FI command		00_CW_PVF_MSTRCT/2
Line	Column	Answer
1	1	(P_ACK)

Value to be written: address of the C structure.

Value of data element	&mstrct
Data code	/2

WinPcl - Example 7 PVF

Write the value of the WinPcl structure "MSTRCT" from the structure “mstrct” previously stored in the C program at device address 00.

Assumption:
The WinPcl variable "MSTRCT" is declared as a structure in WinPcl program "Prog" as follows:

```
TYP STRUCT
      T1      BOOL
      T2      CHAR
      T3      STRING[16]
      T4      TIME
END
```

To exchange binary data in a C program, the following “C” data type can be used:

```
#pragma pack(1)    //Write all elements
                   //without spaces next to each other.
typeder struct
{
      unsigned char T1;
      char          T2;
      char          T3[17]; //Space for zero byte
      unsigned long T4;
} Tymstrct;        // Declare structure
Tymstrct mstrct;   // Apply structure
```

FI command		00_CW_PVF_:Prog.MSTRCT/2
Line	Column	Answer
1	1	(P_ACK)

Value to be written: address of the C structure.

Value of data element	&mstrct
Data code	/2

ProVi Messages: PVM

MWAX device group

Designation **PVM** **Pro**Vi **M**essages

Explanation ProVi messages are output. These messages are assigned to a particular message type or module.

FI command Output all ProVi messages.

BR_PVM1_(1){_(2)} **(Single Read)**

BC_PVM1_(1){_(2)} **(Cyclic Read)**

(1) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -2!

Output first ProVi messages.

BR_PVM2_(1){_(2)} **(Single Read)**

BC_PVM2_(1){_(2)} **(Cyclic Read)**

(1) = Message type [1 = error, 2 = messages,
10 = warnings,
11 = start requirements,
12 = setup diagnosis]

(2) = Module number [1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the FI commands "PVM1" and "PVM2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n	Column 1	...	Column 6

Meaning of the Columns

1 = Message text [ASCII characters]

2 = Message number [ASCII characters]

3 = Time stamp day [mm.dd.yyyy]

4 = Time stamp time [hh:mm:ss]

5 = Message ID [ASCII characters] (DWORD, decimal)

6 = Reference text exists [YES, NO]

7 = Criteria analysis exists [YES, NO]

8 = Message HTML file [ASCII characters]

Example PVM1 All ProVi errors from module 3 in control unit 0.

There are two messages.

FI command		00_BR_PVM1_1_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	01.27.2000
	4	14:56:32
	5	43923028
	6	YES
	7	NO
	8	
2	1	Oil pressure too low
	2	124
	3	01.27.2000
	4	15:03:10
	5	98234039
	6	NO
	7	YES
	8	

Example PVM2 The first ProVi error from module 3 in control unit 0.

There are two messages:

FI command		00_BR_PVM2_1_3
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	01.27.2000
	4	14:56:32
	5	43923028
	6	YES
	7	NO
	8	

FI command Output the additional information of a ProVi message.

BR_PVM3_(1)_(2){_(3)}	**(Single Read)**
(1) = Message ID	[ASCII characters]
(2) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(3) = Module number	[1...99] ! only for message type 1 -2!

Response Structure The following table shows the general structure of the "PVM3" FI command.

Line 1	Column 1	...	Column 16

Meaning of the Columns

1 =	Message text	[ASCII characters]
2 =	Message number	[ASCII characters]
3 =	Error category	[ASCII characters] (empty no category)
4 =	Time stamp day	[mm.dd.yyyy]
5 =	Time stamp hour	[hh:mm:ss]
6 =	Additional text available	[YES, NO]
7 =	Additional text	[ASCII characters]
8 =	Message ID	[ASCII characters] (DWORD, decimal)
9 =	Diagnosis source	[ASCII characters] (PLC, CNC)
10 =	POU name	[ASCII characters]
11 =	Detail name	[ASCII characters] (empty implementation)
12 =	Detail type	[1 = action block, 3 = transition, 4 = implementation]
13 =	Network number	[ASCII characters]
14 =	Variable name	[ASCII characters]
15 =	POU entity name	[ASCII characters]
16 =	POU type	[2 = program, 3 = function block]
17 =	Analysis of criteria available	[YES, NO]
18 =	Message HTML file	[ASCII characters]
19 =	Reference info HTML file	[ASCII characters]

Example PVM3

Additional text of a ProVi error with ID 43923028 from module 3 in control unit 0.

FI command		**00_BR_PVM3_43923028_1_3**
Line	**Column**	**Answer**
1	1	Guard not closed
	2	34
	3	1
	4	01.27.2000
	5	14:56:32
	6	YES
	7	Oil pressure too low Oil pipe leaking or insufficient oil.
	8	43923028
	9	PLC
	10	MODULE3
	11	
	12	4
	13	34
	14	EschutzT
	15	Station2.Module3

FI command		00_BR_PVM3_43923028_1_3
Line	Column	Answer
	16	3
	17	NO
	18	
	19	D:\Programme\Indramat\MtGui\Project_000\ ProgramData\HMTL\DE\Error34.html

FI command

One after the other, all active ProVi messages are output. In the result, one line each is returned. After expiry of the set time, the next message is returned. The clock frequency can be set via the last parameter. This value can only be set once for the PC. The value transmitted last is always the valid value. Default setting is 1 second.

BR_PVM4_(1){_(2)_(3)}	**(Single Read)**
BC_PVM4_(1){_(2)_(3)}	**(Cyclic Read)**
(1) = Message type	[1 = error, 2 = messages, 10 = warnings, 11 = start requirements, 12 = setup diagnosis]
(2) = Module number	[1...99] ! only for message type 1 -2!
(3) = Clock frequency	[ASCII characters] Time in ms

Response Structure

The following table shows the general structure of the "PVM4" FI command.

If there are no messages, the number of lines is 0.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = Message number	[ASCII characters]
3 = Time stamp day	[mm.dd.yyyy]
4 = Time stamp time	[hh:mm:ss]
5 = Message ID	[ASCII characters] (DWORD, decimal)
6 = Additional text available	[YES, NO]
7 = Criteria analysis exists	[YES, NO]
8 = Message index (1 = 1. message)	[ASCII characters]
9 = Message HTML file	[ASCII characters]

Rexroth Indramat

Example PVM1

ProVi errors from module 3 in control unit 0.
The 2nd message is being output. The clock frequency is to be 2 seconds.

FI command		**00_BR_PVM4_1_3_2000**
Line	**Column**	**Answer**
1	1	Oil pressure too low
	2	124
	3	01.27.2000
	4	15:03:10
	5	98234039
	6	NO
	7	YES
	8	2
	9	

Download of PLC Retain Variables: PVR

MWAX device group

Designation **PVR** **P**LC **V**ariable **R**etain Backup

Explanation Download of PLC retain variables.

FI command **BW_PVR1!(1)** **(Single Write)**

(1) = Download file with path details.

Note: File and path details must be enclosed in inverted commas.
The separator "!" is used in this command.

Response Structure

The response to the "PVR1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group: IFJ").
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code
  (see Chapter "Error Codes")

Example PVR1

00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini"/3

FI command		**00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini" /3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BW_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\download.ini" /3
3	1	0

Structure of Download File

The structure of the "download.ini" file used in this example corresponds to an Ini file in Windows.

Note: Care must be taken in the use of upper and lower case letters.

Upload of PLC Retain Variables: PVR

MWAX device group

Designation **PVR** **P**LC **V**ariable **R**etain Backup

Explanation PLC retain variables are uploaded via all active processes.

FI command **BR_PVR1!(1)** **(Single Read)**

(1) = Upload file with path details

Note: Enclose file and path details in inverted commas.
The separator "!" is used in this command.

Response Structure The response to the "PVR1" FI command consists of three lines, each with one column. The meaning of the elements is as follows:

- Line 1 = Job ID [01...20]
  (see Chapter "FI Commands for the MPCX Device Group", IFJ).
- Line 2 = FI command
  [String, in accordance with Chapter "Elements of the FI Command"]
- Line 3 = FI Job Error Code (see Chapter "Error Codes")

Example PVR 00_BR_PVR1_"D:\Program Files\Indramat\Mtgui\Temp\Upload.ini"/3

FI command		**00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\upload.ini" /3**
Line	**Column**	**Answer**
1	1	01
2	1	00_BR_PVR1!"D:\Program Files\Indramat\Mtgui\Temp\upload.ini" /3
3	1	0

Structure of Upload File The upload file is structured in the Windows – "Ini" format structure.

Note: Care must be taken in the use of upper and lower case letters.

Reading the PLC Variable Declaration: PVT

MWAX device group

Designation **PVT** **P**LC **V**ariable **T**ype

Explanation A PLC variable has a particular type. To evaluate complex variables such as structures and arrays, their components and types must be read out.

Refer also to PVF, Reading Structured PLC Variables.

FI command Read the PLC variable type.

BR_PVT_(1) **(Single Read)**

(1) = Identifier of the PLC variable [acc. to declaration part of the PLC]

Response Structure

One line with 2 columns is output for each element of the variables.

Line 1...n:	Column 1	Column 2

n = number of elements.

Value Range/Meaning of Columns

(1) = Identifier of the PLC variable	[acc. to declaration part of the PLC]
2 = Type	[see value range PVF]

Examples:
PLC: Reading of a variable

Assumption:
The "TEST" variable is declared as WORD in the PLC program.

FI command	00_BR_PVT_TEST	
Answer		
Line	Column 1 (Name)	Name
1	TEST	WORD

WinPcl: Reading a Variable

Assumption:
The WinPcl variable "TEST" is declared as WORD in WinPcl program "Prog".

FI command	00_BR_PVT_:Prog.TEST	
Answer		
Line	Column 1 (Name)	Name
1	TEST	WORD

PLC: Reading a Structure

Assumption:
The "TEST1" variable is declared as STRUCT in the PLC program.

```
STRUCT
        E1      BOOL
        E2      INT
        E3      SINT
END
```

FI command	00_BR_PVT_TEST1	
Answer		
Line	Column 1	Column 2
1	TEST1.E1	BOOL
2	TEST1.E2	INT
3	TEST1.E3	SINT

WinPcl: Reading a Structure

Assumption:
The WinPcl variable "TEST1" is declared as STRUCT in WinPcl program "Prog".

```
STRUCT
        E1      BOOL
        E2      INT
        E3      SINT
END
```

FI command	00_BR_PVT_:Prog.TEST1	
Answer		
Line	Column 1	Column 2
1	TEST1.E1	BOOL
2	TEST1.E2	INT
3	TEST1.E3	SINT

PLC: Reading an Array

Assumption:
The "TEST2" variable is declared as ARRAY in the PLC program.

ARRAY [
    0 .. 3
] OF   BOOL

FI command	00_BR_PVT_TEST2	
	Answer	
Line	**Column 1**	**Column 2**
1	TEST2[0]	BOOL
2	TEST2[1]	BOOL
3	TEST2[2]	BOOL
4	TEST2[3]	BOOL

WinPcl: Reading an Array

Assumption:
The WinPcl variable "TEST2" is declared as ARRAY in WinPcl program "Prog".

ARRAY [
    0 .. 3
] OF   BOOL

FI command	00_BR_PVT_:Prog.TEST2	
	Answer	
Line	**Column 1**	**Column 2**
1	TEST2[0]	BOOL
2	TEST2[1]	BOOL
3	TEST2[2]	BOOL
4	TEST2[3]	BOOL

PLC: Reading an Array of a Structure

Assumption:
The "TEST3" variable is declared as ARRAY in the PLC program.

ARRAY [
    0 .. 1
] OF   STRUCT1,

where STRUCT1 is declared as follows:

STRUCT
    E1   BOOL
    E2   INT
    E3   SINT
END

FI command	00_BR_PVT_TEST3	
	Answer	
Line	**Column 1**	**Column 2**
1	TEST3[0].E1	BOOL
2	TEST3[0].E2	INT
3	TEST3[0].E3	SINT
1	TEST3[1].E1	BOOL
2	TEST3[1].E2	INT
3	TEST3[1].E3	SINT

WinPcl: Reading an Array of a Structure

Assumption:
The WinPcl variable "TEST3" is declared as ARRAY in WinPcl program "Prog".

```
ARRAY [
        0 .. 1
] OF    STRUCT1,
```

where STRUCT1 is declared as follows:

```
STRUCT
        E1      BOOL
        E2      INT
        E3      SINT
END
```

FI command	00_BR_PVT_:Prog.TEST3	
Answer		
Line	**Column 1**	**Column 2**
1	TEST3[0].E1	BOOL
2	TEST3[0].E2	INT
3	TEST3[0].E3	SINT
1	TEST3[1].E1	BOOL
2	TEST3[1].E2	INT
3	TEST3[1].E3	SINT

Assumption:
The data types are output according to IEC1131.

See also command PVF.

SFC Diagnosis Data: SDD

MWAX device group

Designation

SDD **S**FC **D**iagnosis **D**ata

Explanation

Data for step chain diagnosis is output. Depending on the FI command this data can concern disrupted steps, actions, transitions or a definite ID to display the action or transition.

FI command

Output the disrupted step of a step chain.

BR_SDD1!(1)!(2)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD1".

Line 1	Column 1	...	Column 7

Meaning of the Columns

1 = Step name	[ASCII characters]
2 = Detail type	[1 = action block, 2 =action network, 3 = transition]
3 = Detail name	[ASCII characters]
4 = POU ID	[ASCII characters]

5 = Detail morpheme	[ASCII characters] (DWORD, decimal)
6 = Error ID	[ASCII characters] (DWORD, decimal)
7 = POE entity name	[ASCII characters]

Example SDD1

Query disrupted step of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD1!3!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	Open
	2	1
	3	Open
	4	SFC_1_2
	5	98243823
	6	34985304
	7	Station2.Module3

FI command

Output the faulty action, monitor error or transition of a disrupted step.

BR_SDD2!(1)!(2)!(3)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Step name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD2".

Line 1	Column 1	...	Column 6

Meaning of the Columns

1 = Detail type	[1 = action block, 2 =action network, 3 = transition]
2 = Detail name	[ASCII characters]
3 = POU ID	[ASCII characters]
4 = Detail morpheme	[ASCII characters] (DWORD, decimal)
5 = Error ID	[ASCII characters] (DWORD, decimal)
6 = POE entity name	[ASCII characters]

Example SDD2

Query faulty action of the disrupted step “open” of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD1!3!Station03A.Clamp_Open**
Line	**Column**	**Answer**
1	1	1
	2	Open
	3	SFC_1_2
	4	98243823
	5	34985304
	6	Station2.Module3

FI command Output the definite ID to display the action, monitor error or transition.

BR_SDD3!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the FI command "SDD3".

Line 1	Column 1	...	Column 4

Meaning of the Columns

1 = POU ID	[ASCII characters]
2 = Detail morpheme	[ASCII characters] (DWORD, decimal)
3 = Error ID	[ASCII characters] (DWORD, decimal)
4 = POE entity name	[ASCII characters]

Example SDD3 Query ID to display the action “aOpen” of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD3!3!Station03A.Clamp!1!aOpen**
Line	**Column**	**Answer**
1	1	SFC_1_2
	2	98243823
	3	34985304
	4	Station2.Module3

FI command Output the I/O addresses to display a detail.

BR_SDD4!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the FI command "SDD4".

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Variable morpheme	[ASCII characters] (DWORD, decimal)
2 = I/O address	[ASCII characters]

Example SDD4 Query I/O addresses to display the action “aOpen” of the “clamp” chain in module 3 in control unit 0.

Three variables have an I/O address.

FI command		00_BR_SDD4!3!Station03A.Clamp!1!aOpen
Line	**Column**	**Answer**
1	1	98243823
	2	%I3.2.0
2	1	40923423
	2	%Q23.21.7
3	1	34985304
	2	%I100.3.5

FI command Determine the multilingual comments for displaying a detail.

BR_SDD5!(1)!(2)!(3)!(4)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure The following table shows the general structure of the FI command "SDD5".

Line 1-n	Column 1	Column 2

Meaning of the Columns

1 = Comment morpheme	[ASCII characters] (DWORD, decimal)
2 = New comment	[ASCII characters]

Example SDD5 Query comments to display the action “aOpen” of the “clamp” chain in module 3 in control unit 0.

Two comments are replaced by another text.

FI command		00_BR_SDD5!3!Station03A.Clamp!1!aOpen
Line	**Column**	**Answer**
1	1	98243823
	2	Clamp open
2	1	40923423
	2	Clamp closed

FI command Output the action that has not been performed, or the transition of a step calculated based on the online status.

BR_SDD6!(1)!(2)!(3)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]
(3) = Step name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the FI command "SDD6".

Line 1	Column 1	...	Column 6

Meaning of the Columns

1 = Detail type	[1 = action block, 3 = transition]
2 = Detail name	[ASCII characters]
3 = POU ID	[ASCII characters]
4 = Detail morpheme	[ASCII characters] (DWORD, decimal)
5 = Error ID	[ASCII characters] (DWORD, decimal)
6 = POU entity name	[ASCII characters]

Example SDD6

Query the action that has not been performed for the step “open” of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SDD6!3!Station03A.Clamp_Open**
Line	**Column**	**Answer**
1	1	1
	2	Open
	3	SFC_1_2
	4	98243823
	5	34985304
	6	Station2.Module3

Setting of Device Status Information: SDS

MWAX device group

Designation **SDS** **S**et **D**evice **S**tatus

Explanation

By this command, the device status information can be set; here, the configuration file IND_DEV.INI is adjusted as well.

Note: When this command is transmitted, the following system messages are generated:
MSG_DEVICEOFF or MSG_DEVICE_ON !

FI command

With this command, the device status information of **ALL** defined devices can be set.

BW_SDS1_(1) **(Single Write)**

(1) = Device status information to be set

0 = Device status information OFF
1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS1" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example SDS1

Set device status information to OFF for **ALL** defined devices.

FI command		**00_BW_SDS1_0**
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

With this command, the device status information for a selected device can be set.

BW_SDS2_(1)	**(Single Write)**
(1) = Device status information to be set	0 = Device status information OFF 1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS2" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example: SDS2

Set device status information to OFF for the selected device 00.

FI command		**00_BW_SDS2_0**
Line	**Column**	**Answer**
1	1	(P_ACK)

Sequencer Data: SFC

MWAX device group

Designation

SFD **SF**C **D**ata

Explanation

Data for a step chain is outputted. Depending on the FI command this can concern a sequencer comment, POU name, step comment, maximum time, action / transition / monitor error name (comment), qualifier and time value.

FI command

Query the data for a step chain.

BR_SFD1!(1)!(2)	**(Single Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "SFD1" FI command.

Line 1	Column 1	Column 2

Meaning of the Columns

1 = Step chain comment	[ASCII characters]
2 = POU name	[ASCII characters]

Example SFD1

Query data of the “clamp” chain in module 3 in control unit 0.

FI command		**00_BR_SFD1!3!Station03A.Clamp**
Line	**Column**	**Answer**
1	1	Clamping device
	2	CLAMP

FI command

Query the data of a step.

BR_SFD2!(1)!(2)!(3)	**(Single Read)**
(1) = Module number	[1...99]

(2) = SFC entity name [ASCII characters]

(3) = Step name [ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "SFD2" FI command. The number of lines depends on the number of actions and transitions.

If there are no details the line number is 1.

Line 1	Column 1	...	Column 3
Line 2...n:	Column 1	...	Column 6

Meaning of the Columns

Line 1

1 = Step comment [ASCII characters]

2 = Maximum time [ASCII characters]

3 = Minimum time [ASCII characters]

Line 2...n:

1 = Detail type [1 = action block, 3 = transition]

2 = Name [ASCII characters]

3 = Comment [ASCII characters]

4 = Boolean variable [YES, NO]

5 = Qualifier [ASCII characters]

6 = Time value [ASCII characters]

Example SFD2

Data for the step “Open” in the “clamp” chain in module 3 on control unit 0.

FI command		00_BR_SFD2!3!Station03A.Clamp!Open
Line	**Column**	**Answer**
1	1	Open clamping device
	2	T#5s
	3	
2	1	1
	2	aOpen
	3	Clamp open
	4	NO
	5	D
	6	T#3s
3	1	3
	2	tOpen
	3	Clamping device is open
	4	NO
	5	
	6	

FI command

Output the data for a detail.

BR_SFD3!(1)!(2)!(3)!(4) **(Single Read)**

(1) = Module number [1...99]

(2) = SFC entity name	[ASCII characters]
(3) = Detail type	[1 = action block, 2 = action network, 3 = transition]
(4) = Detail name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "SFD3" FI command.

Line 1	Column 1	Column 2

Meaning of the Columns

1 = Comment	[ASCII characters]
2 = Boolean variable	[YES, NO]

Example SFD3

Data for the action “aOpen” in the “clamp” chain in module 3 on control unit 0.

FI command		00_BR_SFD3!3!Station03A.Clamp!aOpen
Line	**Column**	**Answer**
1	1	Clamp open
	2	NO

Sequencer Messages: SFE

MWAX device group

Designation

SFE **S**F**C** **E**rror

Explanation

The sequencer messages of a module are output.

FI command

Output all SFC messages.

BR_SFE1_(1)	**(Single Read)**
BC_SFE1_(1)	**(Cyclic Read)**
(1) = Module number	[1...99]

Output first SFC messages.

BR_SFE2_(1)	**(Single Read)**
BC_SFE2_(1)	**(Cyclic Read)**
(1) = Module number	[1...99]

Response Structure

The following table shows the general structure of the FI commands "SFE1" and "SFE2". The number of lines depends on the number of messages pending.

If there are no messages, the number of lines is 0.

Line 1...n:	Column 1	...	Column 7

Meaning of the Columns

1 = Message text	[ASCII characters]
2 = SFC entity name	[ASCII characters]
3 = Step name	[ASCII characters]
4 = Time stamp day	[mm.dd.yyyy]
5 = Time stamp time	[hh:mm:ss]
6 = Type of error	[1 = time error, 2 = monitor error, 3 = monitor event]

7 = Is there condition analysis?	[YES, NO]

Example SFD1 All SFC messages from module 2 in control unit 0.

There are two messages.

FI command		00_BR_SFE1_2
Line	**Column**	**Answer**
1	1	TIME ERROR: Chain: chucking Step: up malfunction
	2	Station03A.Clamp
	3	Open
	4	01.27.2000
	5	11:56:32 AM
	6	1
	7	YES
2	1	ASSY ERROR: Chain: drilling Step: down malfunction
	2	Station02A.Drill
	3	Down
	4	01.27.200
	5	13:03:12
	6	2
	7	NO

Example SFE2 First SFC message from module 2 in control unit 0.

There are two messages.

FI command		00_BR_SFE2_2
Line	**Column**	**Answer**
1	1	TIME ERROR: Chain: chucking Step: up malfunction
	2	Station03A.Clamp
	3	Open
	4	01.27.2000
	5	14:56:32
	6	1
	7	YES

Sequencer Mode: SFM

MWAX device group

Designation **SFM** **SF**C **M**ode

Explanation Queries step chain mode.

FI command Query the mode of a step chain.

BR_SFM1!(1)!(2)	**(Single Read)**
BC_SFM1!(1)!(2)	**(Cyclic Read)**
(1) = Module number	[1...99]
(2) = SFC entity name	[ASCII characters]

Note: The separator "!" is used in this command.

Response Structure

The following table shows the general structure of the "SFM1" FI command.

Line 1	Column 1

Meaning of the Columns

1 = Mode [1 = time error, 2 = monitor error, 3 = monitor event, 10 = stop, 11 = auto, 12 = manual, 13 = jog]

Example SFM1

Query mode of the “clamp” chain in module 3 in control unit 0.

FI command		00_BR_SFM1!3!Station03A.Clamp
Line	**Column**	**Answer**
1	1	1

Software Installation Data: SID

MWCX device group

Designation

SID **S**oftware **I**nstallation **D**ata

Explanation

Information is returned regarding installation. This information includes installation paths, the software version used, DLL mode, plus service pack and release information.

FI command

Read-in the installation data.

BR_SID1 **(Single Read)**

BC_SID1 **(Cyclic Read)**

Response Structure

One line with 8 columns is output for the returned values.

Line 1	Column 1	...	Column 8

Meaning of the Columns

1 = Basic directory [EXE files of the BOF]
2 = FI installation directory [FI directory]
3 = Data directory [in accordance with BOF]
4 = GBO version [from INDRAMAT.ini]
5 = IF-DLL mode [from INDRAMAT.ini]
6 = IF version [from INDRAMAT.ini from DLL mode 400]
7 = Service package info [from INDRAMAT.ini from DLL mode 420]
8 = Release info [from INDRAMAT.ini from DLL mode 420]

Example SID1

Return information on the current installation.

FI command		00_BR_SID1
Line	**Column**	**Answer**
1	1	--
	2	D:\Programme\Indramat\MTGUI\Bin
	3	--
	4	005-22Vxx
	5	07.00
	6	07V00
	7	--
	8	--

Note: Refer to FI command “PHD” for working with absolute paths.

PLC Long Identification: SLI

MWAX device group

Designation **SLI** **S**PS **L**ong **I**dentification

Explanation Returns the unit data from the PLC long identification.

FI command Read PLC long identification.

BR_SLI **(Single Read)**

Response Structure One line with 15 columns is output for the returned values.

Line 1	Column 1	Column...	Column 15

Value Range/Meaning of the Columns

1 =	Device address	[00...15]
2 =	Program number	[01...99]
3 =	Project name	[max. 8 ASCII characters]
4 =	Program name	[max. 8 ASCII characters]
5 =	User name	[acc. to password entry]
6 =	Program length	[bytes]
7 =	Compilation time	[LONG] (coded in long value)
8 =	Compilation date	[8 ASCII characters]
9 =	Compilation time	[8 ASCII characters]
10 =	Download time	[LONG] (coded in long value)
11 =	Download date	[8 ASCII characters]1
12 =	Download time	[8 ASCII characters]
13 =	Version of PLC long identification	[LONG]
14 =	RUN flags	[HEX value]
15 =	Compiler info	[LONG]

Example SLI Read the unit data from the PLC long identification.

FI command		**00_BR_SLI**
Line	**Column**	**Answer**
1	1	02
	2	01
	3	--
	4	MOT12
	5	TEST
	6	17672
	7	630163960
	8	15.12.99
	9	17:15:48
	10	630163961
	11	15.12.99
	12	17:15:50

FI command		00_BR_SLI
Line	Column	Answer
	13	2
	14	0x0000
	15	13

Reference to Literature see chapter entitled “Literature” [30].

Requesting Watch List Allocations: WLA

MWAX device group

Designation **WLA** **W**atch **L**ist **A**llocation

Explanation Requests free watch list allocations. A maximum of ten free watch list allocations can be requested with one FI command.

BR_WLA1_(1) **(Single Read)**

(1) = Number of the requested free watch list numbers — The required number of free watch list allocations is identified here. The allowed value range: 1…10

Response Structure The following table shows the general structure of the response to the FI command "WLA1".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. free watch list allocation	Value range: 1..16
2 =	2. free watch list allocation	Value range: 1..16
3 =	3. free watch list allocation	Value range: 1..16
n =	nth free watch list allocation	Value range: 1..16

Example WLA1 Request four free watch list allocations.

Assumption:
Watch list allocations 3 and 5 are already assigned!

FI command		00_BR_WLA1_4
Line	Column	Answer
1	1	1
	2	2
	3	4
	4	6

Freeing Watch List Allocations: WFL

MWAX device group

Designation **WLF** **W**atch **L**ist **F**ree

Explanation Previously requested watch list allocations are freed again.

FI command Free ALL assigned watch list allocations for the selected device.

BR_WLF1 **(Single Read)**

Note: The FI command "WLF1" frees ALL assigned watch list allocations, including those of other WIN32 applications.

Response Structure

The following table shows the general structure of the response to the FI command "WLF1".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. freed watch list allocation	Value range: 1..16
2 =	2. freed watch list allocation	Value range: 1..16
3 =	3. freed watch list allocation	Value range: 1..16
n =	nth freed watch list allocation	Value range: 1..16

Example WLF1

Free ALL assigned watch list allocations.

Assumption:
The following watch list numbers have been allocated: 1, 2, 3, 4.

FI command		00_BR_WLF1
Line	**Column**	**Answer**
1	1	1
	2	2
	3	3
	4	4

FI command

Free the required watch list allocations for a selected device.

BR_WLF2_(1)_{(2)..(10)}	**(Single Read)**
(1)..(10) = List of watch list allocations to be released	A maximum of 10 watch list allocations can be transferred here to be freed again.

Response Structure

The following table shows the general structure of the response to the FI command "WLF2".

Line 1	Column 1	...	Column n

Value Range/Meaning of Columns

1 =	1. enabled watch list allocation	Value range: 1..16
2 =	2. enabled watch list allocation	Value range: 1..16
3 =	3. enabled watch list allocation	Value range: 1..16
n =	nth freed watch list allocation	Value range: 1..16

Example WLF2

Free required watch list allocations:
Assumption:
Watch list allocations 1,3,4, and 8 have first been requested using the FI command "WLA1".

FI command		00_BR_WLF2_1_3_4_8
Line	**Column**	**Answer**
1	1	1
	2	3
	3	4

7.7 FI Commands for the MSYX Device Group

The FI Commands described in this chapter are valid for the MSYX device group. In this device group, the following types are listed as well as possible addresses:

Group	Accompanying Types	Address
MSYX	SYNAX200-P, SYNAX200-R	[00...63]

Note: Please note that the device address must be set before the respective FI command, e.g. 00_CR_AAC_0 (refer also here to the chapter "Elements of the FI Command").

Parameters for the MSYX device group are grouped together in the chapter entitled "Construction and Availability of the FI Commands", "Data Tables".

Determining the Actual (Current) System Error: ASE

MSYX device group

Designation **ASE** **A**ctual **S**ystem **E**rror

Explanation The current system error is read out, whereby the answer 0x0000 indicates that the Synax device is functioning correctly.

FI command **CR_ASE** **(Single Read)**

Response Structure The following table shows the general structure of the response to the FI command "ASE". In line 1, column 4, the number of the drive is output that reports the current system error. Not all current system errors can be directly allocated to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line 1	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = 0x0000

2 = 0x0000

3 = Current system error

4 = Drive No.

Example ASE Reading the current system error returns LWL ring interrupted.

FI command		00_CR_ASE
Line	**Column**	**Answer**
1	1	0x0000
	2	0x0000
	3	0x8009
	4	0x0000

Reference to Literature See chapter entitled “Literature” [42].

Deleting the Actual (Current) System Error: CSE

MSYX device group

Designation **CSE** **C**lear **S**ystem **E**rror

Explanation An error reported by the Synax device is deleted again.

FI command **CW_CSE** **(Single Write)**

Value to be written: The contents of the value parameter is not evaluated.

Response Structure The following table shows the general structure of the response to the FI command "CSE". In line 1, column 4, the number of the drive is output that reports the current system error. Not all current system errors can be directly allocated to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line 1	Column 1	...	Column 4

Value Range/Meaning of Columns

1 = 0x0000

2 = 0x0000

3 = Actual (current) system error

4 = Drive No.

Example CSE Deleting the actual (current) system error:

FI command		00_CW_CSE
Line	**Column**	**Answer**
1	1	0x0000
	2	0x0000
	3	0x0000
	4	0x0000

Reference to Literature see chapter entitled “Literature” [45].

Setting the Communication Timeout Time DCT

MSYX device group

Designation **DCT** **D**evice **C**ommunication **T**imeout

Explanation By means of this command, the timeout time for the selected device is set dynamically (timeout time in ms).

FI command **BW_DCT1_(1)** **(Single Write)**

(1) = requested timeout time in ms

Response Structure The response to the "DCT1" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCT1 For the device 00, the timeout time is set 1500 ms.

FI command		00_BW_DCT1_1500
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command With this command, the timeout time for the selected device can be reset to default value.

BW_DCT2 **(Single Write)**

Response Structure The response to the "DCT2" FI command consists of one line with one column.

Line 1	Column 1

Value Range/Meaning of Columns

1 = Status message (P_ACK) (P_ACK)

Example DCP2 For the device 00, the timeout time is reset to the default value.

FI command		00_BW_DCT2
Line	**Column**	**Answer**
1	1	(P_ACK)

Device Type and Accompanying Components: DTY

MSYX device group

Designation **DTY** **D**evice **TY**pe

Explanation The device type and the accompanying components of the selected device address are output.

FI command **BR_DTY1** **(Single Read)**

Response Structure The following table shows the general structure of the response to the “DTY1” FI command. A line with three columns for the device type is output as well as the name of the first device component and the name of the second device component.

Line 1	Column 1	...	Column 3

Value Range/Meaning of Columns

1 = Device Type (see chapter entitled “Elements of the FI Command”, and “Identifier”)

2 = Component type1 IND_DEV.INI entry: Component type1=

3 = Component type 2 IND_DEV.INI entry: Component type2=

Example DTY1 Output the device type and the accompanying components of device address 00.

FI command	00_BR_DTY1		
Answer			
Line	**Column 1**	**Column 2**	**Column 3**
1	SYNAX200-R	NONE	PPC-R

Read System Messages: MSG

MSYX device group

Designation **MSG** **M**e**S**sa**G**e

Explanation Reading of system messages

FI command Message

CC_MSG_(1) **(Cyclic Read)**

(1) = SYS message numbers

Note: Exists only as a cyclic command

Response Structure The response of the FI command 'MSG' consists of the system message data.

Example MSG 00_CC_MSG_64 (64 = MSG_SYSERRGEN)

FI command		**00_CC_MSG_64/3**
Line	**Column**	**Answer**
1	1	00

Restriction The following system messages:

SYS Message	SYS Message number
MSG_PCLUPDBEG	52
MSG_PARUPDBEG	24
MSG_FWAUPDBEG	82

These commands cannot be used with the following programs:

Indramat OPC server

Indramat DDE server

Generating Physical Directory Names: PHD

MSYX device group

Designation **PHD** **PH**ysical **D**irectory

Explanation Generates physical directory names according to the BDI data written.

Note: This is based on BDI philosophy.

FI command Generate physical directory names.

BR_PHD1_(1)_(2)_(3)_(4)_(5)_(6) **(Single Write)**

(1) = Project ID [-1= PROJECT_NEUTRAL
-2= PROJECT_DEFAULT]

(2) = Section ID [0= SECT_NEUTRAL
1= SECT_BIN
2= SECT_BASIC_DATA
3=SECT_OEM_DATA
4=SECT_CUSTOM_DATA
5=SECT_PROG_DATA]

(3) = Device address	[-1= DEVADDR_NEUTRAL otherwise the required device address]
(4) = Process ID	[-1= PROCESS_NEUTRAL otherwise the required process number]
(5) =Data type ID	[possible write values see BDI documentation (BDI_DEFINITIONS.H)]
(6) = Language ID	[possible write values see BDI documentation (WINNT.H)]

Response Structure

The following table shows the general structure of the response to the FI command "PHD1".

Line 1	Column 1

Value Range/Meaning of Columns

1 =	Physical directory name	[complete physical directory name in accordance with the BDI data written]

Example PHD1

Requesting the physical directory name for:

PROJECT_NEUTRAL
SECT_BIN
DEVADDR_NEUTRAL
PROCESS_NEUTRAL
DATATYPE_NEUTRAL
LANG_NEUTRAL

FI command		XX_BR_PHD1_-1_0_-1_-1_0_0
Line	Column	Answer
1	1	D:\Programme\Indramat\Mtgui\Bin

Setting of Device Status Information: SDS

MSYX device group

Designation **SDS** **S**et **D**evice **S**tatus

Explanation

By this command, the device status information can be set; here, the configuration file IND_DEV.INI is adjusted as well.

Note: When this command is transmitted, the following system messages are generated:
MSG_DEVICEOFF or MSG_DEVICE_ON !

FI command

With this command, the device status information of **ALL** defined devices can be set.

BW_SDS1_(1)	**(Single Write)**
(1) = Device status information to be set	0 = Device status information OFF 1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS1" FI command.

Line 1	Column 1

Value Range/Meaning of Columns

1 =	Status report	[(P_ACK)]

Example SDS1

Set device status information to OFF for **ALL** defined devices.

FI command		**00_BW_SDS1_0**
Line	**Column**	**Answer**
1	1	(P_ACK)

FI command

With this command, the device status information for a selected device can be set.

BW_SDS2_(1)	**(Single Write)**
(1) = Device status information to be set	0 = Device status information OFF 1 = Device status information ON

Response Structure

The following table shows the general structure of the response to the "SDS2" FI command.

Line 1	**Column 1**

Value Range/Meaning of Columns

1 = Status report [(P_ACK)]

Example: SDS2

Set device status information to OFF for the selected device 00.

FI command		**00_BW_SDS2_0**
Line	**Column**	**Answer**
1	1	(P_ACK)

Software Installation Data: SID

MSYX device group

Designation

SID **S**oftware **I**nstallation **D**ata

Explanation

Information is returned regarding installation. This information includes the installation paths, the software version used, DLL mode, plus service pack and release information.

FI command

Read-in the installation data.

BR_SID1 **(Single Read)**

BC_SID1 **(Cyclic Read)**

Response Structure

One line with 8 columns is output for the returned values.

Line 1	**Column 1**	**...**	**Column 8**

Meaning of the Columns

1 = Basic directory	[EXE files of the DOS-BOF]
2 = FI installation directory	[FI directory]
3 = Data directory	[in accordance with DOS-BOF]
4 = GBO version	[from INDRAMAT.ini]
5 = IF-DLL mode	[from INDRAMAT.ini]
6 = IF version	[from INDRAMAT.ini from DLL mode 400]
7 = Service pack info	[from INDRAMAT.ini from DLL mode 420]
8 = Release info	[from INDRAMAT.ini from DLL mode 420]

Example SID1 Return information on the current installation.

FI command		00_BR_SID1
Line	**Column**	**Answer**
1	1	--
	2	D:\Programme\Indramat\MTGUI\Bin
	3	--
	4	005-22Vxx
	5	07.00
	6	07V00
	7	--
	8	--

Note: Refer to FI command "PHD" for working with absolute paths.

SERCOS Parameters: SPA

MSYX device group

Designation **SPA** **S**ERCOS **PA**rameter

Explanation A SERCOS drive parameter is output or written. Each parameter consists of 7 elements, whereby 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 or advanced format]

Parameter No.

Format X-Y-ZZZZ	Value Range
X	S = standard data P = product data Y = SERCANS parameter
Y	[0..00.15] = parameter record
Z	[0...4095] = data block no.

Element Coding Element coding in standard format allows individual elements, such as the operating date, to be requested. If several elements are to be read out in one request, then the element coding can be OR'd in advanced format, e.g. operating date (0x40) and unit (0x08) produces OR'd (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	The marked section is then printed out.	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
Operating date	D	40H	(see Displaying the Operating Date	
Operating date, when no list		80H		

Displaying the Operating Date

The display of the operating date depends on the parameter number requested.

Decimal

Decimal values are given as floating points, e.g. 1.5. Leading spaces, zeros, plus and minus signs as well as trailing spaces are allowed.

Hexadecimal

Hexadecimal values are displayed by "0x...", e.g. 0x80. Up to a maximum of eight positions are allowed. Leading or trailing spaces are allowed. Leading additional zeros or plus and minus signs are not allowed.

Binary (max. 32 characters)

Leading or trailing spaces are allowed. The decimal point serves as separator:

e.g., 1111.0000.1010.1100.1111.0000.1010.1100

Note: Leading additional zeros or plus and minus signs are not allowed.

ID number

The following table shows the general way in which the ID number is displayed:

Format X-Y-ZZZZ	Value Range
X	S = standard data P = product data
Y	[0..0.7] = parameter record
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 maximum length of the list. The length specification refers to the length of the list in the drive and therefore designates the number of bytes for storage (storage bytes). The number of elements in the list can be calculated using the attribute. The list elements are displayed according to the attribute. All parts of the list are separated from each other 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 variable length lists. The individual string characters are not separated by a line feed. When displaying the lists, a distinction is made between standard format and advanced format. In standard format, only the character string is returned, whereas in advanced format the actual length and the maximum length of the list (string) is also transmitted.

Example:

Parameter S-0-0030, operation date
Standard format: "DKC2.1-SSE-01V09"
Advanced format: "16\n16\nDKC2.1-SSE-01V09"

Note: When requesting SERCANS parameters the drive address can be anywhere within the range [0..254].

Response Structure

The following table shows the general structure of the response to the FI command "SPA1". Line 1 is output both when reading and when writing. Additional lines are only output when reading depending on the element coding.

Note: If the element coding has been requested in standard format then the first line is not applicable.

Line 1 is a status line that either contains SERCOS / SERCANS errors or displays the successful processing of the FI command. If the command has been processed successfully, then columns 1 and 3 contain the value [0x0000].

In the first line, column 2 or column 4, the number of the drive is output that reports the SERCOS error or the global SERCANS error. Not all global SERCANS errors can be directly assigned to a drive. In this case, the single result "Drive No." is set to 0x0000.

Line	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: Element corresponding to the element coding.			
...	...			
n	Reading: (n-1). Element corresponding to the element coding.			

Example SPA1 / read

Read parameter S-0-0003 of the 3rd drive (element coding 0x48)

FI command	00_BR_SPA1_3_S-0-0003_48			
Answer				
Line	**Column 1**	**Column 2**	**Column 3**	**Column 4**
1	0x0000	0x0000	0x0000	0x0000
2	µs			
3	2000			

Example SPA1 / write

Write the ID number P-0-0037 in parameter S-0-0305 of the 3rd drive (element coding 0x40).

Technical background:

- Real time status bit 1 is to 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			
Answer				
Line	Column 1	Column 2	Column 3	Column 4
1	0x0000	0x0003	0x0000	0x0000

Reference to Literature See chapter entitled “Literature” [41].

See chapter entitled “Literature” [46].

Active SERCOS Phase Switch-Over: SPH

MSYX device group

Designation **SPH** **S**ERCOS **PH**ase

Explanation All drives within a SERCOS ring are in the same communication phase. The phase status can be read-out or changed by this command.

FI command **CR_SPH** **(Single Read)**

CW_SPH **(Single Write)**

Value to be written Phase [2, 4]

Note: The value to be written is passed to the "acValue" parameter in the "DataTransfer" routine.

Example SPH Read SERCOS Phase

Read the active phase of the synax control at device address 00.

FI command		00_CR_SPH
Line	Column	Answer
1	1	2

Example SPH Write SERCOS Phase

Switch-over the synax control (write) after phase 4; phase 2 is active.

FI command		00_CW_SPH Value to be written: 4
Line	Column	Answer
1	1	52
	2	1

Note: Switching over from phase 2 to phase 4 returns the value [52] as the result in column 1. On switching over from phase 4 to phase 2, column 1 contains the value [50]. The result of column 2 is the physical axis number in both cases.

Reference to Literature See chapter entitled “Literature” [42].

8 Error codes

8.1 General Error Result Line

If the "DataTransfer" routine returns an error code, then the requested data is not returned by the "ReadGroupItem" routine, but a general error result line is returned instead. This general error result line contains additional information regarding the possible causes of the error.

Note: How the routines work, as well as tips and tricks for working with the Rexroth Indramat function interface, is described in Chapter “Programming”.

The following table shows the general structure of the error result line. One line is output consisting of 5 columns for the class of error, error code, expanded additional information, error text and additional text.

Line 1	Column 1	...	Column 5

Meaning of the Columns

1 = Error class

2 = NACK code or return error code (depends on error classes)

3 = Expanded additional information [hexadecimal LONG value]

4 = Error text [ASCII characters]

5 = Additional text [x= exists, -- = does not exist]

The following error classes are contained in the file "INDIF000.h" or "INDIF000.BAS":

Error class	Meaning
FI_ERROR_CLASS_NACK	NACK control messages
FI_ERROR_CLASS_FCT	Protocol function error codes

8.2 Error Codes 200 to 999

Code	Error Text	Name and Meaning of Error
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 found.
204	BOF_NO_MAP_FILE_NAME	No entry in the "PLCMAP.DAT" file has been found for the device address.
205	BOF_FILE_ERROR	File processing error.
206	BOF_VERSION_ERROR	More than the MAXIMUM NUMBER OF LINES contained 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 file mapping.
209	BOF_MEMORY_ERROR	Memory allocation error.

Code	Error Text	Name and Meaning of Error
210	BOF_DATA_MAP_ERROR	DATA MAP access error.
211	BOF_MUTEX_TIMEOUT	MAP file access error.
212	BOF_DATA_LENGTH_ERROR	Data buffer is too small.
213	BOF_FILE_NOT_FOUND	File not found.
214	BOF_SYS_MAP_ERROR	SYSTEM MAP access error.
215	BOF_MAP_ELEMENT_ERROR	No valid MAP structure element.
216	BOF_INVALID_CHANNEL_ERROR	LOG channel number invalid.
217	BOF_TIMEOUT_ERROR	Pre-set timeout has expired.
218	BOF_SHMEM_ALREADY_EXIST	SHARED MEM already exists.
219	BOF_PROCESS_NOT_EXIST	Process addressed 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 be started already running.
223	BOF_COMM_ADDRESS_ERROR	No valid communication 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 name.
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 false task ID.
230	BOF_TASK_ADM_ERROR	Task administration error.
231	BOF_TASK_TRIGGER_ERROR	Task trigger-event error.
232	BOF_EVENT_NOT_FOUND	Event object does not 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 is not known in SYS-MSG STACK.
237	BOF_SYS_STACK_ACCEPT_ERROR	SYS-MSG message could not be accepted by the SYS-MSG STACK within the pre-set time.
238	BOF_SYS_MSG_SET_ERROR	Access to SYS-MSG channel not possible in the pre-set time. (SYS-Message is issued).
239	BOF_SYS_MSG_GET_ERROR	Access to SYS-MSG channel not possible in the pre-set time (SYS-Message is fetched).
240	BOF_DATA_TIME_ERROR	A data element in the shared memory area was not released in the pre-set time.
241	BOF_DATA_ACCESS_ERROR	Access to a data element in the shared memory area is locked.
242	BOF_FCT_PAR_ERROR	An incorrect parameter value has been passed within the function.
243	BOF_SYS_STACK_QUIT_ERROR	SYS-MSG acknowledgement event has not been released in the pre-set time.
244	BOF_NO_SYS_MSG_RDY	No SYS-MSG message.

Code	Error Text	Name and Meaning of Error
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 details missing in "IfDllMode=" entry in "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.
251	BOF_LOGIN_IF_EVENT_ERROR	LOGINIF event could not be created.
252	BOF_LOGIN_SYS_MSG_ERROR	LOGINIF could not be carried out in the pre-set time due to pending SYS-MSGs.
253	BOF_LOGIN_EVENT_TIME_ERROR	No SYS-MSG could be issued because the login event took too long.
254	BOF_DLL_MUTEX_TIMEOUT_ERROR	Access to the DLL-lock Mutex was not possible in the pre-set time.
255	BOF_DLL_ALREADY_INSTALLED	DLL already installed.
256	BOF_DLL_LOAD_ERROR	DLL could not be loaded by the load library.
257	BOF_DLL_LIST_FULL_ERROR	DLL list already full.
258	BOF_DLL_LIST_DELETE_ERROR	DLL to be deleted is not in the list.
259	BOF_DOS_NT_SYS_MSG_ERROR	Invalid SYS-MSG message number in 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 is outside the valid number range.
262	BOF_DOS_NT_JOB_INFO_ERROR	DOS → NT command information is too long
263	BOF_DOS_NT_SYS_MSG_Q_ERROR	An odd SYS-MSG message number (acknowledgement) was passed by the DOS → NT command SYSC_xxx.
264	BOF_DOS_NT_FKT_NOT_FOUND_ERROR	DOS → NT command issued for which there is no processing function in the "BOFINTFC.DAT" file.
265	BOF_DOS_NT_DLL_NAME_NOT_FOUND_ERROR	No DLL name exists for DOS → NT commands in the "BOFINTFC.DAT" file.
266	BOF_DOS_NT_DLL_NOT_FOUND_ERROR	DLL for the DOS → NT commands not found.
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 in the task list.
270	BOF_TASK_ID_NOT_FOUND_ERROR	No task exists for the task ID.
271	BOF_NT_CODE_ERROR	WIN-32 API error has occurred.
272	BOF_DOS_NT_PROCESS_PRIORITY_ERROR	Invalid process priority class.
273	BOF_DOS_TASK_NAME_ERROR	Error in generating the DOS-BOF task name.
274	BOF_PARENT_WIN_NAME_LEN_ERROR	Name of the parent window is too long.
275	BOF_TERMINATE_EVENT_NAME_LEN_ERROR	Name of the terminate event is too long.
276	BOF_PARENT_WIN_NOT_EXIST_ERROR	Registered task does not have a parent window.
277	BOF_DLL_NOT_EXIST_ERROR	DLL sought does not exist.
278	BOF_DLL_FUNCTION_NOT_FOUND_ERROR	Function sought does not exist in the specified DLL.

Code	Error Text	Name and Meaning of Error
279	BOF_PROCESS_NOT_LOGIN_ERROR	An FI command has been called although the client is not yet logged in. The "LogInIf" routine has not yet been run.
280	BOF_DEVICE_HANDLE_ERROR	Device handle could not be generated.
281	BOF_DEVICE_ASSIGN_ERROR	There is no "DeviceAssign" entry in the "IND_DEV.INI" file, or the entry is invalid.
282	BOF_MEMORY_CLASS_ERROR	No valid memory class for DOS → NT command RDNT/WRNT.
283	BOF_MEMORY_DOS_NT_DATA_LEN_ERROR	Data length of RDNT/WRNT command is too long.
284	BOF_SHMEM_INDEX_ERROR	No valid SHARED MEMORY INDEX.
285	BOF_NO_PORT_ADDR_ERROR	No port address in the communication address.
286	BOF_NO_PORT_VALUE_ERROR	No port value in the communication address.
287	BOF_VRT_MANAGER_MODE_ERROR	Invalid VRT manager mode
288	BOF_VRT_START_MODE_ERROR	There is no MTVNC mode entry in the "IND_DEV.INI" file, or the entry is invalid.
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 pass parameters in the FI command.
291	BOF_PAR_MAX_ERROR	No PARMAX value in the "BOFINTFC.DAT" file for the 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.
294	BOF_PAR_TYPE_ERROR	Invalid FI parameter type.
295	BOF_PAR_TYPE_NOT_FOUND_ERROR	No FI parameter type description found.
296	BOF_PAR_DATA_ERROR	Invalid FI parameter data, i.e. FI data not defined in 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 there is no CYCLIC entry in the "BOFINTFC.DAT" file or the CYCLIC entry is invalid.
300	BOF_PAR_NO_CYCLIC_FI_COMMAND_ERROR	No cyclic FI command released.
301	BOF_PAR_NO_BINAER_ERROR	Either there is no binary entry, or an invalid binary entry in the "BOFINTFC.DAT" file.
302	BOF_PAR_NO_BINAER_FI_COMMAND_ERROR	No binary operation of FI command released.
303	BOF_NT_DOS_CHANNEL_ACCESS_ERROR	Access to NT → DOS job channel not possible in the pre-set time.
304	BOF_NT_DOS_COMMAND_LENGTH_ERROR	NT → DOS command string is too long.
305	BOF_NT_DOS_COMMAND_INFO_LENGTH_ERRO R	NT → DOS command info string is too long.
306	BOF_NT_DOS_TIMEOUT_ERROR	NT → DOS job could not be executed in the pre-set time.
307	BOF_NT_DOS_FKT_NOT_FOUND_ERROR	An NT → DOS command was issued that had not been declared in the "BOFINTFC.DAT" file.
308	BOF_NT_DOS_DLL_NAME_NOT_FOUND_ERROR	No DLL is declared in the "BOFINTFC.DAT" file for the NT→DOS command issued.

Code	Error Text	Name and Meaning of Error
309	BOF_NT_DOS_DLL_NOT_FOUND_ERROR	DLL for the NT → DOS commands not found.
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 information is too long
313	BOF_NT_DOS_DPR_TIMEOUT_ERROR	Access to NT-DOS-DPR memory not possible in the pre-set time.
314	BOF_NT_DOS_NO_COMMAND_ERROR	No NT → DOS command string.
315	BOF_NT_DOS_BOF_INDEX_ERROR	Invalid DOS-BOF INDEX when issuing an NT → DOS command.
316	BOF_PAR_INVALID_VALUE_ERROR	Pass parameter to the function has an invalid value.
317	BOF_DOS_BOF_EXE_PATH_ERROR	DOS-BOF EXE file must not contain details of path.
318	BOF_LOG_IN_LOG_OUT_TIMEOUT_ERROR	Login/Logout not possible in the pre-set time.
319	BOF_DEVICE_TYP_GROUP_ERROR	Selected device address does not exist in this device group.
320	BOF_INVALID_PROCESS_NUMBER_ERROR	Invalid NC process number
321	BOF_PROCESS_NAME_LENGTH_ERROR	Process name is too long or invalid.
322	BOF_PARAM_IDENT_REQUEST_ERROR	Invalid data was returned by the interface on requesting the parameters.
323	BOF_SWITCH_DEVICE_ERROR	An attempt was made to switch to a virtual MTC that is assigned to a real MTC.
324	BOF_DEVICE_TYPE_REQUEST_ERROR	Invalid data was returned on requesting the device type ID.
325	BOF_DEVICE_SPS_IDENT_ERROR	Invalid data was returned by the interface on requesting the long ID of the PLC MAP file.
326	BOF_INVALID_AXIS_NUMBER_ERROR	Invalid axis number received [1...32].
327	BOF_NO_GBOVERSION_ERROR	There is no "GBOVERSION=" entry in the "INDRAMAT.INI" file, or the entry is invalid.
328	BOF_NO_ACHSREF_TABLE_ERROR	Axis reference table error.
329	BOF_DEVICE_GROUP_ERROR	The device group for this job is invalid.
330	BOF_PROCESS_NOT_DEFINED	Process is not defined in the current parameters.
331	BOF_INVALID_DEVICE_GROUP_VALUE_ERRO R	Invalid device group number.
332	BOF_INVALID_DEVICE_ID_STR_ERROR	Invalid device ID 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 more FI JOBs possible.
336	BOF_FI_JOB_ID_ERROR	No valid FI JOB ID.
337	BOF_FI_JOB_NO_ID_FOUND_ERROR	No FI JOB ID found in the management structure.
338	BOF_FI_JOB_PROGRESS_TYPE_ERROR	Invalid request for the progress of an FI JOB.
339	BOF_FI_JOB_EXECUTE_FKT_NOT_FOUND_ ERROR	Execute function for the FI JOB was 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 executed in the pre-set time.

Code	Error Text	Name and Meaning of Error
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	NT → DOS job description is too long.
345	BOF_FILE_CLASS_OBJECT_INSTALL_ERROR	File class object for access to the BOF files could not be created.
346	BOF_FILE_DIAGOFF_NOT_FOUND_ERROR	BOF file "DIAGOFF.XXX" not found.
347	BOF_FILE_DIAGOFF_OPEN_ERROR	Error opening the "DIAGOFF.XXX" file.
348	BOF_SH_MEM_DIAGOFF_NOT_FOUND_ERROR	No SHARED MEMORY for DIAGOFFxxx found.
349	BOF_FILE_DIAGTAB_NOT_FOUND_ERROR	BOF file "DIAGTAB.XXX" not found.
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" not found.
352	BOF_FILE_STERR_FILE_CLOSE_ERROR	Diagnostic 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 diagnostic text file "STERRxx.YYY" could not be carried out.
355	BOF_FILE_STERR_FILE_READ_ERROR	Read function of diagnostic text file "STERRxx.YYY" could not be carried out.
356	BOF_FILE_STERR_FILE_NOT_FOUND_ERROR	Diagnostic text file "STERRxx.YYY" not found.
357	BOF_FILE_DIAGTAB_POSITION_ERROR	File positioning in "DIAGTAB.xxx" could not be carried out.
358	BOF_FILE_STERR_FILE_TIMEOUT_ERROR	TIMEOUT when waiting for the MUTEX release for access to the STERR files.
359	BOF_TASK_THREAD_TRIGGER_INFO_TO_LONG	Additional information passed for the TASK THREAD triggering is too long.
360	BOF_TASK_THREAD_TRIGGER_TIMEOUT_ERROR	TIMEOUT of MUTEX release for access to the TASK-THREAD triggering.
361	BOF_FILE_SPRACHE_FILE_OPEN_ERROR	"SPRACHE.DAT" file could not be opened.
362	BOF_COMMAND_RESULT_DATA_TYPE_ERROR	A result data type that is not permissible (e.g. 00_BR_AMM1/2) was requested for an FI-command (BR_...).
363	BOF_FILE_TEXT_FILE_NOT_FOUND_ ERROR	Relevant TEXTxx.YY file does not exist.
364	BOF_FILE_TIND_FILE_NOT_FOUND_ERROR	Relevant TINDxx.YY file does not exist.
365	BOF_FILE_TIND_FILE_OPEN_ERROR	TINDxx.YY could not be opened.
366	BOF_TEXT_NUMBER_TO_LARGE_ERROR	Text number to be read from BOF text file is too large.
367	BOF_FILE_TEXT_FILE_OPEN_ERROR	TEXTxx.YY could not be opened.
368	BOF_FILE_TEXT_FILE_POSITION_ERROR	File positioning in the text file "TEXTxx.YY" could not be carried out.
369	BOF_FILE_TEXT_FILE_READ_ERROR	Read function in the text file "TEXTxx.YY" could not be carried out.
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" not found.
372	BOF_FILE_SYSERI_OPEN_ERROR	Error opening the "SYSERI.XXX" file.

Code	Error Text	Name and Meaning of Error
373	BOF_FILE_SYSERI_POSITION_ERROR	File positioning in SYSERI.xxx could not be carried out.
374	BOF_SH_MEM_SYSERI_NOT_FOUND_ERROR	No SHARED MEMORY for SYSERI.xxx found.
375	BOF_FILE_SYSANW_NOT_FOUND_ERROR	Diagnosis text file SYSANW.YY is not available.
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 could not be carried out.
379	BOF_FILE_SYSANW_READ_ERROR	Read function in the diagnosis text file "SYSANW.YY" could not be carried out.
380	BOF_FILE_SYSANW_FILE_TIMEOUT_ERROR	TIMEOUT when waiting for the MUTEX release for access to the SYSANW.YY files.
381	BOF_FILE_TXERR_FILE_NOT_FOUND_ERROR	Relevant TXERR.YY file not found.
382	BOF_FILE_TXERI_FILE_NOT_FOUND_ERROR	Relevant TXERI.YY file not found.
383	BOF_FILE_TXERI_FILE_OPEN_ERROR	TXERI.YY could not be opened.
384	BOF_FILE_TXERR_FILE_OPEN_ERROR	TXERR.YY could not be opened.
385	BOF_FILE_TXERR_FILE_POSITION_ERROR	File positioning in the text file "TXERR.YY" could not be carried out.
386	BOF_FILE_TXERR_FILE_READ_ERROR	Read function in the text file "TXERR.YY" could not be carried out.
387	BOF_COMMAND_NOT_AVAILABLE_DLL_MODE	The requested FI command not available for the "IfDllMode=" set in the "INDRAMAT.INI" file.
388	BOF_NO_PARAMETER_SET_IN_CONTROL	No valid parameter record in the control unit.
389	BOF_MTA200_COMMANDLINE_ERROR	No valid command line for the MTA200 DRIVER.
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 protocol entry.
393	BOF_DEVICE_IP_ERROR	No, or invalid, DEVICEIP entry.
394	BOF_DOS_NT_TASK_CHANNEL_TIMEOUT_ERRO R	Access to DOS → NT job channel not possible in the pre-set time.
395	BOF_PROCESS_NAME_ERROR	A syntax error has been detected in the 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_ERRO R	No terminate event found for the registered TASK.
399	BOF_PARENT_WIN_ALREADY_EXIST_ERROR	PARENT WINDOW name already exists in the task management file.
400	BOF_NO_IFVERSION_ERROR	No "IfVersion=" entry exists in the "Indramat.INI" file.
401	BOF_NO_IFVERSION_ERROR	No IFVERSION entry in INDRAMAT.INI
402	BOF_NO_MTA200_INST_PATH	No MTA200 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_ERRO R	The temporary sub-directory TEMPDATA could not be created for the data files of the small devices.

Code	Error Text	Name and Meaning of Error
405	BOF_DELETE_FILE_ERROR	Data file (small devices) 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 for TXERR.XX (SHORT MESSAGES only!) does not exist in the temporary sub-directory TEMPDATA.
409	BOF_TXERR_FILTER_FILE_OPEN_ERROR	Filter file TXEST.XX for TXERR.XX (SHORT MESSAGES only!) could not be opened in the temporary sub-directory TEMPDATA.
410	BOF_TXEST_INDEX_FILE_CREATE_ERROR	INDEX file TXEST.XX (SHORT MESSAGES only!) could not be created.
411	BOF_BUFFER_LENGTH_ERROR	The PROCESSING BUFFER is too small for the data to be processed.
412	BOF_MSG_NUMBER_0_NOT_EXIST_ERROR	NO message number 0 exists in the message file.
413	BOF_MSG_NUMBER_TO_BIG_ERROR	Message number in message file is too big.
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 the temporary sub-directory TEMPDATA.
416	BOF_SYSSTW_INDEX_FILE_CREATE_ERROR	Index file SYSSTW.XX (SHORT MESSAGES!) 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 directory TEMPDATA.
418	BOF_STERR_FILTER_FILE_NOT_FOUND_ERROR	Filter file STESTyy.XX for STERRyy.XX (SHORT MESSAGES only!) does not exist in the temporary directory TEMPDATA.
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 sub-directory TEMPDATA.
421	BOF_WRONG_TELEGRAMM_CODE_ERROR	An INCORRECT TELEGRAM CODE has been returned by the control unit.
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 is not open.
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 cannot be read.
431	BOF_DEVICE_TYPE_VALUE_TO_LARGE	DEVICE TYPE number is too large.

Code	Error Text	Name and Meaning of Error
432	BOF_NOT_ENOUGH_MEMORY_IN_CONTROL	The required memory is not available 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!) could not be read (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).
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 large.
443	BOF_INVALID_COMPONENT_NUMBER_ERROR	Invalid component number.
444	BOF_DEVICE_COMPONENT_TYPE_REQUEST_ ERROR	INVALID DATA was returned by the interface on requesting the DEVICE COMPONENT TYPES.
445	BOF_DEVICE_DAT_FILE_NOT_FOUND_ERROR	Relevant DEVICE-DAT file not found for the BOF configuration.
446	BOF_MAIN_MENU_ITEM_ERROR	Invalid GUI main menu item.
447	BOF_MAIN_DEF_FILE_CONTENT_ERROR	BOF configuration file \MT_TEXTE\MAIN_DEF.INI not entered in sought device type.
448	BOF_DEVICE_INI_FILE_NOT_FOUND_ERROR	Relevant DEVICE-INI file not found for the BOF configuration.
449	BOF_DEVICE_INI_FILE_SYNTAX_ERROR	Format error in DEVICE-INI file for the BOF configuration.
450	BOF_DEVICE_POLLING_STATUS_ERROR	NO, or invalid, PollDeviceStatus in IND_DEV.INI.
451	BOF_DEVICE_POLLING_RATE_ERROR	NO, or invalid, PollDeviceStatusRate in IND_DEV.INI.
452	BOF_DEVICE_POLLING_CHECK_FACTOR_ERR OR	NO, or invalid, PollStatusCheckFactor in IND_DEV.INI.
453	BOF_DOS_BOF_EXE_SYNTAX_ERROR	NO "_" character may be included in DOS-BOF-EXE file names (WITH TSR connection).
454	BOF_DOS_BOF_EXE_CMDLINE_SYNTAX_ERROR	NO "_" character may be included in the call parameters for the DOS-BOF-EXE (WITH TSR connection).
455	BOF_SYS_MSG_LENGTH_ERROR	The additional information for the SYS message is too long.
456	BOF_DEVICE_STATUS_INFO_ERROR	More than one “critical” condition is managed in the DEVICE-STATUS INFO (SYSTEM MAP) e.g.: parameter download.
457	BOF_SYS_MSG_HOOK_LIST_TIMEOUT_ERROR	The SYS-MSG HOOK LIST cannot be accessed within the pre-set time.

Code	Error Text	Name and Meaning of Error
458	BOF_PROCESS_LOGOUT_TIMEOUT_NETINTFC	NETINTFC has not logged out from the TASK MANAGEMENT LIST within the pre-set DELAY TIME.
459	BOF_PROCESS_LOGOUT_TIMEOUT_DESKTOP	DESKTOP has not logged out from the TASK MANAGEMENT LIST within the pre-set DELAY TIME.
460	BOF_PROCESS_LOGOUT_TIMEOUT_ CONTROLDATA	CONTROL DATA has not logged out from the TASK MANAGEMENT LIST within the pre-set DELAY TIME.
461	BOF_PROCESS_LOGOUT_TIMEOUT_LOGDBCO M	LOGDBCOM has not logged out from the TASK MANAGEMENT LIST within the pre-set DELAY TIME.
462	BOF_PROCESS_LOGOUT_TIMEOUT_MPI	MPI has not logged out from the TASK MANAGEMENT LIST within the pre-set DELAY TIME.
463	BOF_PROCESS_LOGOUT_TIMEOUT_BOFINTFC	BOFINTFC has not logged out from the TASK MANAGEMENT LIST within the pre-set DELAY TIME.
464	BOF_IF_DLL_MODE_TO_SMALL	IF-DLL MODE set is too small for the function to be executed.
465	BOF_WATCH_LIST_OVERRUN_ERROR	NO WATCH LIST available (overrun) for the selected device.
466	BOF_INVALID_WATCH_LIST_NUMBER_ERROR	INVALID WATCH LIST NUMBER for the selected DEVICE.
467	BOF_NO_SYSTEM_ERRORTEXT_ADM	There is NO management system for access to the SYSTEM ERROR TEXTS (SYSANW.XX)
468	BOF_NO_TX_ERRORTEXT_ADM	There is NO management system for access to the TRANSMISSION ERROR TEXTS (TXERR.XX)
469	BOF_NO_MECH_ERRORTEXT_ADM	There is NO management system for access to the MECHANISM ERROR TEXTS (STERRyy.XX)
470	BOF_INVALID_PLC_TYPE	An INVALID PLC type has been detected for the selected device.
471	BOF_SHMEM_INTERN_INDEX_ERROR	Maximum INTERNAL SHM INDEX is too big.
472	BOF_SHMEM_INDEX_TO_BIG_ERROR	SHM INDEX is too big.
473	BOF_INTERNAL_PROCESS_NUMBER_ERROR	NO INTERNAL PROCESS NUMBER (MT-NCPROZESSE : 0..6).
474	BOF_AXIS_NUMBER_NOT_DEFINED_ERROR	Requested AXIS NUMBER is not defined in the ACTUAL PARAMETER RECORD.
475	BOF_NO_PARAMETER_DOWNLOAD_FILE_EXI ST	NO PARAMETER DOWNLOAD FILE available.
476	BOF_PARAMETER_DOWNLOAD_FILE_LOAD_ ERROR	The PARAMETER DOWNLOAD FILE CANNOT be loaded.
477	BOF_PARAMETER_DOWNLOAD_FILE_ID_ERROR	A LENGTH ERROR has occurred in the various parameter IDs OR necessary entries are missing.
478	BOF_ALLOCATE_MEMORY_ERROR	NO memory could be allocated.
479	BOF_PARAMETER_DOWNLOAD_FILE_INDEX_E RROR	Max. DATA INDEX ENTRY in the PARAMETER DOWNLOAD FILE is NOT available.
480	BOF_PARAMETER_DOWNLOAD_DATA_TO_LON G	An individual PARAMETER DATA STRING to be passed is too long for the telegram.
481	BOF_PARAMETER_IDENTIFICATION_NOT_EXIS T	NO [ID_PARAMETER] section exists in the PARAMETER DOWNLOAD FILE.

Code	Error Text	Name and Meaning of Error
482	BOF_SYSTEM_PARAMETER_IDENTIFICATION_NOT_EXIST	NO [ID_SYSTEM] section exists in the PARAMETER DOWNLOAD FILE.
483	BOF_SYSTEM_PARAMETER_DATA_NOT_EXIST	NO [DATA_SYSTEM] section exists in the PARAMETER DOWNLOAD FILE.
484	BOF_PROCESS_PARAMETER_DATA_NOT_EXIST	NO [DATA_PROCESSx] section exists in the PARAMETER DOWNLOAD FILE.
485	BOF_AXIS_PARAMETER_DATA_NOT_EXIST	NO [DATA_AXISx] section exists in the PARAMETER DOWNLOAD FILE.
486	BOF_INVALID_PARAMETER_DATA	The control unit has detected INVALID PARAMETER DATA during PARAMETER DOWNLOAD.
487	BOF_PARAMETER_DATA_NOT_COMPLETE	INCOMPLETE PARAMETER DATA has been passed during a PARAMETER DOWNLOAD.
488	BOF_NECESSARY_FUNCTION_NOT_FOUND	The necessary FUNCTIONS are NOT contained in the selected DLL, or the DLL is not found.
489	BOF_PARAMETER_DATA_LINE_TO_MUCH	The NUMBER of DATA LINES in the PARAMETER DOWNLOAD FILE is too great.
490	BOF_PARAMETER_MAX_ALLOCATE_MEMORY_ERROR	MEMORY REQUIREMENT for the INTERNAL DATA STRUCTURE during PARAMETER DOWNLOAD is too great.
491	BOF_PARAMETER_DOWNLOAD_BREAK_ERROR	A PARAMETER DOWNLOAD PROCEDURE has been interrupted by BREAK-INFO.
492	BOF_PARAMETER_UPLOAD_FILE_ALREADY_EXIST	PARAMETER UPLOAD FILE already exists.
493	BOF_PARAMETER_IDENT_SECTION_CREATE_ERROR	NO [ID_PARAMETER] section could be created in the PARAMETER UPLOAD FILE.
494	BOF_PARAMETER_UPLOAD_DATA_WRITE_ERROR	NO UPLOAD DATA could be written in the PARAMETER UPLOAD FILE.
495	BOF_SYSTEM_PARAMETER_IDENT_SECTION_CREATE_ERROR	NO [ID_SYSTEM] section could be created in the PARAMETER UPLOAD FILE.
496	BOF_DATA_SYSTEM_SECTION_CREATE_ERROR	NO [DATA_SYSTEM] section could be created in the PARAMETER UPLOAD FILE.
497	BOF_MAX_INDEX_DATA_SECTION_CREATE_ERROR	NO [MAX_INDEX_DATA] section could be created in the PARAMETER UPLOAD FILE.
498	BOF_ID_PROCESS_SECTION_CREATE_ERROR	NO [ID_PROCESSx] section could be created in the PARAMETER UPLOAD FILE.
499	BOF_DATA_PROCESS_SECTION_CREATE_ERROR	NO [DATA_PROCESSx] section could be created in the PARAMETER UPLOAD FILE.
500	BOF_ID_AXIS_SECTION_CREATE_ERROR	NO [ID_AXISx] section could be created in the PARAMETER UPLOAD FILE.
501	BOF_SAVE_ARRAY_PROCESSING_ERROR	An error occurred when processing SAVE ARRAYS.
502	BOF_COM_INTERFACE_REQUEST_ERROR	Requested COM INTERFACE could NOT be returned.
503	BOF_COM_DIAG_SERVER_INIT_ERROR	An error occurred when INITIALIZING THE COM DIAG TEXT SERVER.
504	BOF_CO_INITIALIZE_ERROR	A co-initialize procedure has NOT been carried out in the user process.
505	BOF_COM_DIAG_TEXT_ACCESS_ERROR	An error occurred fetching a message text.
506	BOF_COM_INTERFACE_DIAG_SERVER_NULL_ERROR	COM INTERFACE POINTER for accessing the DIAG SERVER is ZERO.
507	BOF_OBJECT_CREATE_ERROR	NO OBJECT could be generated – GENERAL CREATE ERROR.

Code	Error Text	Name and Meaning of Error
508	BOF_LANGUAGE_CONVERT_ERROR	A CONVERSION ERROR has occurred during the various conversions regarding language management.
509	BOF_NO_WRITE_DATA_AVAILABLE_ERROR	NO write value is available when calling the BW command.
510	BOF_TO_MANY_SPS_VARIABLES_ERROR	Too many PLC variables were transferred when calling up the BW-MKT command.
511	BOF_COM_INTERFACE_LANGUAGE_SERVER_ NULL_ERROR	COM INTERFACE POINTER for accessing the LANGUAGE SERVER is ZERO.
512	BOF_COM_CURRENT_LANGUAGE_ACCESS_ ERROR	Language extension CURRENTLY set could not be determined via LANGSUPP.
513	BOF_COM_CURRENT_CONTEXT_NAME_ERROR	The CURRENT CONTEXT NAME could NOT be determined.
514	BOF_SYS_MSG_NUMBER_RANGE_ERROR	SYS-MESSAGE number is NOT within the permissible value range – ONLY EVEN NUMBERS are permitted.
515	BOF_IF_STARTUP_TIMEOUT_ERROR	STARTUP of the BOFINTFC.EXE exceeds the max. permissible startup time.
516	BOF_TCP_IP_OBJECT_CREATE_ERROR	No further TCP-IP communication object can be created. The maximum number has already been opened.
517	BOF_TCP_IP_OBJECT_NOT_EXIST_ERROR	The TCP-IP communication object addressed does NOT exist.
518	BOF_SYSTEM_PARAMETER_AXIS_INFO_ERROR	The AXIS INFORMATION (AXIS TYPE, APR NUMBER) could NOT be converted in the parameter download file – FORMATTING ERROR!
519	BOF_NO_MACHINE_DATA_DOWNLOAD_FILE_ EXIST	MACHINE DATA DOWNLOAD FILE does not exist.
520	BOF_MACHINE_DATA_DOWNLOAD_FILE_LOAD_ ERROR	The MACHINE DATA DOWNLOAD FILE CANNOT be loaded.
521	BOF_MACHINE_DATA_IDENTIFICATION_NOT_ EXIST	NO [ID_MACHINE_DATA] section exists in the MACHINE DATA DOWNLOAD FILE
522	BOF_MACHINE_DATA_DOWNLOAD_FILE_ID_ ERROR	A LENGTH ERROR has occurred in the various machine data IDs OR necessary entries are missing.
523	BOF_MACHINE_DATA_TYPEDEF_INFORMATION _NOT_EXIST	[TYPE_DEFINITION_INFO] section does not exist in the MACHINE DATA DOWNLOAD FILE.
524	BOF_MACHINE_DATA_TYPEDEF_INFORMATION _NOT_EXIST	[TYPE_DEFINITION_INFO] section does not exist in the MACHINE DATA DOWNLOAD FILE.
525	BOF_MACHINE_DATA_TYPEDEF_INFORMATION _ERROR	A FORMATTING ERROR has occurred in the [TYPE_DEFINITION_INFO] section in the MACHINE DATA DOWNLOAD FILE.
526	BOF_MACHINE_DATA_TO_MUCH_TYPEDEF_ ERROR	Too many TYPE DEFINITIONS in the MACHINE DATA DOWNLOAF FILE.
527	BOF_MACHINE_DATA_TO_MUCH_TYPEDEF_ ERROR	Too many TYPE DEFINITIONS in the MACHINE DATA DOWNLOAF FILE.
528	BOF_MACHINE_DATA_PAGE_INFO_ERROR	NO [PAGE_INFO] section present in the MACHINE DATA DOWNLOAD FILE, or there is a formatting error.
529	BOF_MACHINE_DATA_TO_MUCH_PAGEDEF_ ERROR	Too many PAGE DEFINITIONS present in the MACHINE DATA DOWNLOAD FILE, OR the PAGE NUMBER is too high.
530	BOF_MACHINE_DATA_PAGE_DEFINITION_NOT_ EXIST	NO [PAGE_DEFINITION_XXX] section exists in the MACHINE DATA DOWNLOAD FILE, although a relevant [ID_PAGE_DEFINITION_XXX] section has been defined.

Code	Error Text	Name and Meaning of Error
531	BOF_MACHINE_DATA_PAGE_DEFINITION_ERROR	A FORMATTING ERROR has occurred in the [PAGE_DEFINITION_XXX] section in the MACHINE DATA DOWNLOAD FILE.
532	BOF_MACHINE_DATA_ELEMENT_DESCRIPTION_ NOT_EXIST	The [PAGE_DESCRIPTION_XXX_YYY] section which should be present based on the PAGE DEFINITION does NOT exist in the MACHINE DATA DOWNLOAD FILE – NO DATA ELEMENT DESCRIPTION exists.
533	BOF_MACHINE_DATA_ELEMENT_DESCRIPTION_ ERROR	A FORMATTING ERROR has occurred in the [PAGE_DESCRIPTION_XXX_YYY] section in the MACHINE DATA DOWNLOAD FILE.
534	BOF_MACHINE_DATA_PAGE_INFORMATION_ ERROR	A FORMATTING ERROR has occurred in the [PAGE_DATA_INFO] section in the MACHINE DATA DOWNLOAD FILE or necessary entries are missing.
535	BOF_MACHINE_DATA_PAGE_NOT_EXIST	NO [PAGE_DATA_XXX] section exists in the MACHINE DATA DOWNLOAD FILE
536	BOF_MACHINE_DATA_ELEMENT_INFORMATION _ERROR	A FORMATTING ERROR or logic error has occurred in the [PAGE_DATA_ELEMENTS_XXX] section in the MACHINE DATA DOWNLOAD FILE.
537	BOF_MACHINE_DATA_PAGE_LINE_ERROR	Either the requested data line does NOT exist in the [PAGE_DATA_XXX] section in the MACHINE DATA DOWNLOAD FILE, OR there is a formatting error in the data line.
538	BOF_MACHINE_DATA_VALUE_STRING_ CONVERT_ERROR	A MACHINE DATA STRING that is to be written CANNOT be converted – FORMATTING ERROR, or a formatting error has occurred in the parameter string of the MDS command.
539	BOF_MACHINE_DATA_VALUE_RANGE_ERROR	A VALUE RANGE ERROR has occurred when converting the MACHINE DATA STRING.
540	BOF_MACHINE_DATA_STRUCT_TO_LARGE_ERR OR	A DATA STRUCTURE that is too large is present in the MACHINE DATA that is to be written – this CANNOT be written as a COMPLETE DATA STRUCTURE.
541	BOF_MACHINE_DATA_PAGE_INFO_NOT_EXIST	NO [PAGE_INFO] section exists in the MACHINE DATA DOWNLOAD FILE
542	BOF_MACHINE_DATA_VALUE_STRING_TO_LONG	A data string that is TOO LARGE exists in the [PAGE_DATA_XXX] section in the MACHINE DATA DOWNLOAD FILE (max. 50 characters possible).
543	BOF_MACHINE_DATA_DOWNLOAD_BREAK_ERRO R	A MACHINE DATA DOWNLOAD PROCEDURE has been interrupted by BREAK-INFO.
544	BOF_MACHINE_DATA_VALUE_TO_MUCH_ERROR	Too many MACHINE DATA VALUES to be written are entered in the download file, or too many MACHINE DATA VALUES have been indicated in the MDS command.
545	BOF_GLOBAL_DATA_BUFFER_INDEX_TO_LARGE	The GLOBAL DATA BUFFER INDEX is outside the permissible range.
546	BOF_GLOBAL_DATA_BUFFER_CONTENT_TO_ LARGE	Too many BYTES are to be copied into the GLOBAL DATA BUFFER.
547	BOF_MACHINE_DATA_UPLOAD_FILE_ALREADY_ EXIST	PARAMETER UPLOAD FILE already exists.
548	BOF_NO_MACHINE_DATA_SET_IN_CONTROL	NO MACHINE DATA RECORD exists in the CONTROL.
549	BOF_MACHINE_DATA_UPLOAD_BREAK_ERROR	A MACHINE DATA UPLOAD PROCEDURE has been interrupted by BREAK-INFO.

Code	Error Text	Name and Meaning of Error
550	BOF_MACHINE_DATA_ELEMENT_NUMBER_TO_ LARGE	The pre-set MACHINE DATA ELEMENT NUMMER is outside the valid range [1..110].
551	BOF_MACHINE_DATA_ELEMENT_COMPUTING_ ERROR	A computation error has occurred in computing the necessary number of telegrams for reading the machine data.
552	BOF_NO_TOOL_MANAGEMENT_EXIST	NO tool management has been switched on in the system parameters or process parameters.
553	BOF_NO_DIAG_SERVER_AVAILABLE	NO DIAG SERVER available.
554	BOF_UNKNOWN_MESSAGE_TYPE	An unknown message type has occurred during message download for the small devices via the MFD command.
555	BOF_MESSAGE_DOWNLOAD_BREAK_ERROR	A MESSAGE DOWNLOAD PROCEDURE has been interrupted by BREAK-INFO.
556	BOF_MACHINE_DATA_IDENT_SECTION_CREATE_ ERROR	NO [ID_MACHINE_DATA] section could be created in the MACHINE DATA UPLOAD FILE.
557	BOF_MACHINE_DATA_UPLOAD_DATA_WRITE_ ERROR	NO UPLOAD DATA could be written in the MACHINE DATA UPLOAD FILE.
558	BOF_MACHINE_DATA_ID_TYPE_SECTION_ CREATE_ERROR	NO [ID_TYPE_DEFINITION] section could be created in the MACHINE DATA UPLOAD FILE.
559	BOF_MACHINE_DATA_TYPE_INFO_SECTION_ CREATE_ERROR	NO [TYPE_DEFINITION_INFO] section could be created in the MACHINE DATA UPLOAD FILE.
560	BOF_MACHINE_DATA_TYPE_DEF_SECTION_ CREATE_ERROR	NO [TYPE_DEFINITION_XXX] section could be created in the MACHINE DATA UPLOAD FILE.
561	BOF_MACHINE_DATA_PAGE_INFO_SECTION_ CREATE_ERROR	NO [PAGE_INFO] section could be created in the MACHINE DATA UPLOAD FILE.
562	BOF_MACHINE_DATA_PAGE_ID_SECTION_ CREATE_ERROR	NO [ID_PAGE_DEFINITION_XXX] section could be created in the MACHINE DATA UPLOAD FILE.
563	BOF_MACHINE_DATA_PAGE_DEF_SECTION_ CREATE_ERROR	NO [PAGE_DEFINITION_XXX] section could be created in the MACHINE DATA UPLOAD FILE.
564	BOF_MACHINE_DATA_PAGE_DESCRIPTION_ SECTION_CREATE_ERROR	NO [PAGE_DESCRIPTION_XXX_YYY] section could be created in the MACHINE DATA UPLOAD FILE.
565	BOF_MACHINE_DATA_PAGE_DATA_INFO_ SECTION_CREATE_ERROR	NO [PAGE_DATA_INFO] section could be created in the MACHINE DATA UPLOAD FILE.
566	BOF_MACHINE_DATA_PAGE_DATA_ELEMENT_ SECTION_CREATE_ERROR	NO [PAGE_DATA_ELEMENTS_XXX] section could be created in the MACHINE DATA UPLOAD FILE.
567	BOF_MACHINE_DATA_PAGE_DATA_SECTION_ CREATE_ERROR	NO [PAGE_DATA_XXX] section could be created in the MACHINE DATA UPLOAD FILE.
568	BOF_LOGIN_COMINTFC_ERROR	An attempt has been made to log in the COMINTFC PROZESS ROOM.
569	BOF_LOGIN_LOGINTFC_ERROR	An attempt has been made to log in the LOGINTFC PROZESS ROOM.
570	BOF_PARAMETER_UPLOAD_BREAK_ERROR	A PARAMETER UPLOAD PROCEDURE has been interrupted by BREAK-INFO.
571	BOF_HOMATIC_DRIVER_DLL_NOT_FOUND_ ERROR	The HOMATIC DRIVER DLL (INDIFY00.DLL) could NOT be found.
572	BOF_HOMATIC_DRIVER_DLL_COULD_NOT_BE_ LOAD_ERROR	The HOMATIC DRIVER DLL (INDIFY00.DLL) could NOT be loaded.
573	BOF_HOMATIC_DRIVER_DLL_FUNCTION_LOAD_ ERROR	The HOMATIC DRIVER DLL (INDIFY00.DLL) does NOT contain ALL THE NECESSARY functions.
574	BOF_TIME_DATE_SET_STATUS_ERROR	NO, or invalid, TimeDateSetStatus entry in IND_DEV.INI.

Code	Error Text	Name and Meaning of Error
575	BOF_TIME_DATE_SET_STATUS_RATE_ERROR	NO, or invalid, TimeDateSetStatusRate entry in IND_DEV.INI.
576	BOF_NO_PCL_PROGRAMM_IN_CONTROL	NO valid PLC program exists in the control.
577	BOF_FI_START_DISPLAY_ERROR	Invalid FIStartDisplay entry in IND_DEV.INI.
578	BOF_FI_START_DISPLAY_MODE_ERROR	NO, or invalid, FIStartDisplayExtendedMode entry in IND_DEV.INI.
579	BOF_INVALID_DEVICE_SIMULATION_ERROR	Invalid DeviceSimulation entry in IND_DEV.INI.
580	BOF_NO_SIMULATION_DEVICE_TYPE_EXIST_ ERROR	NO SIMULATION is possible for the pre-selected device address.
581	BOF_SOFT_MONITOR_START_MODE_ERROR	Invalid SoftMonitorMode entry in IND_DEV.INI.
582	BOF_SIMISP_START_MODE_ERROR	Invalid SimIspMode entry in IND_DEV.INI.
583	BOF_SEKTION_NOT_FOUND_ERROR	The desired SECTION NAME is not AVAILABLE in the file to be modified
584	BOF_SIMTRA_START_MODE_ERROR	Invalid SimTraMode entry in IND_DEV.INI.
585	BOF_DESTINATION_FILE_ALREADY_EXIST_ ERROR	The destination file already exists on file copying
586	BOF_PARAMETER_OFFLINE_FILE_LOAD_ERROR	The File ParamOff.dat cannot be loaded
587	BOF_PARAMETER_OFFLINE_FILE_ID_ERROR	A LENGTH ERROR has occurred in the various parameter IDs OR necessary entries are missing.
588	BOF_PARAMETER_ONLINE_FILE_LOAD_ERROR	The File ParamOn.dat cannot be loaded
589	BOF_PARAMETER_ONLINE_FILE_ID_ERROR	A LENGTH ERROR has occurred in the various parameter IDs OR necessary entries are missing.
590	BOF_PROCESS_TYPE_ERROR	Invalid process type
591	BOF_PARAMETER_DATA_NOT_FOUND_ERROR	In the parameter data file, necessary parameter data are not available or not correct
592	BOF_DEBUG_DEVICE_MODE_ERROR	Invalid DeviceDebugMode entry in IND_DEV.INI.
593	BOF_FI_COMMAND_LENGTH_ERROR	The FI command string is too long
594	BOF_GENERAL_FILE_NOT_FILE_ERROR	General: file NOT found
595	BOF_FI_COMMAND_DEVICE_STATUS_ERROR	There is an invalid DeviceStatus for the FI command
596	BOF_OPERATING_SYSTEM_NOT_SUPPORTED_ ERROR	The operating system is NOT supported by FI.
597	BOF_SOFTMONITOR_SHUTDOWN_ERROR	Invalid SoftMonitorShutDown entry in IND_DEV.INI.
598	BOF_STARTED_PROCESS_TERMINATE_ERROR	The PROCESS to be started from FI BEFORE the READY MESSAGE
599	BOF_COMM_ADDRESS_TYPE_ERROR	The communication type is invalid
600	BOF_INVALID_AXIS_MEANING_INFO	The axis meaning is invalid
601	BOF_PAR_MIN_NUMBER_ERROR	Necessary PARAMETERS in the FI requirement command are not available
602	BOF_IND_DEV_READ_ERROR	IND_DEV.INI can NOT be read in with the PROFILE CLASS
603	BOF_PROFILE_SECTION_NOT_FOUND_ERROR	The searched SECTION is NOT available in the profile
604	BOF_FI_ERROR_TEXT_INFO_ERROR	Invalid FiErrorTextInfo entry in IND_DEV.INI.
605	BOF_DEVICE_ADDRESS_CONSISTENT_ERROR	The selected DEVICE ADDRESS in the FI command does NOT correspond to the DEVICE ADDRESS in the binary telegram.

Code	Error Text	Name and Meaning of Error
606	BOF_WRONG_DEVICE_PROTOCOL_ERROR	A false data protocol was chosen for the selected DEVICE ADDRESS
607	BOF_TOOL_POSITION_TO_LARGE_ERROR	The information on max. tool storage is too large according to the process parameters.
608	BOF_FI_COMMAND_STACK_FULL_ERROR	There is NO more free space in the FI command stack
609	BOF_FI_COMMAND_STACK_SOURCE_DATA_ LENGTH_ERROR	The SOURCE data is too long for the FI command stack
610	BOF_FI_COMMAND_STACK_RESULT_DATA_ LENGTH_ERROR	The RESULT data is too long for the FI command stack
611	BOF_FI_COMMAND_STACK_INDEX_OUT_OF_ RANGE_ERROR	Invalid FI command stack index – out of range
612	BOF_FI_COMMAND_STACK_ERROR	Invalid FiCommandStack entry in IND_DEV.INI.
613	BOF_FI_COMMAND_STACK_RATE_ERROR	NO, or invalid, FiCommandStackRate entry in IND_DEV.INI.
614	BOF_DEVICE_POLLING_OFF_ERROR	NO device polling is switched on in IND-DEV.INI
615	BOF_TIMEOUT_FOR_FURTHER_INFO_FROM_ DEVICE_ERROR	In the selected delay time, the controller did NOT supply the additional information
616	BOF_ERROR_TEXT_NUMBER_CONVERT_ERROR	The error number can NOT be converted through sscanf()
617	BOF_ERROR_TEXT_NOT_FOUND_ERROR	The error number can NOT be resolved in an error text
618	BOF_MACHINE_DATA_PAGE_SIZE_ERROR	A defined PAGE is larger than 64Kbyte
619	BOF_RECEIVED_TELEGRAMM_LENGTH_TO_ LARGE_ERROR	The received telegram data is too long
620	BOF_SPS_VARIABLE_NAME_LENGTH_ERROR	The names of the PLC variables are too long
621	BOF_LOGDBCOM_TIMEOUT_ERROR	LOGDBCOM could NOT be started in the preselected delay time
622	BOF_SSCANF_CONVERT_ERROR	Conversion through sscanf() is NOT possible
623	BOF_PROVI_ADM_FILE_LOAD_ERROR	The PROVI administration file can NOT be read
624	BOF_PROVI_ADM_FILE_ALREADY_EXIST	The PROVI administration file already exists
625	BOF_PROFILE_SECTION_CREATE_ERROR	The SECTION to be written could NOT be generated through the profile class
626	BOF_PROVI_TEXT_FILE_NOT_FOUND_ERROR	The PROVI TEXT FILE does NOT exist
627	BOF_PROVI_INDEX_FILE_NOT_FOUND_ERROR	The PROVI INDEX FILE does NOT exist
628	BOF_PROVI_TEXT_FILE_OPEN_ERROR	The PROVI TEXT FILE can NOT be opened
629	BOF_PROVI_INDEX_FILE_OPEN_ERROR	The PROVI INDEX FILE can NOT be opened
630	BOF_PROVI_MESSAGE_FILE_CREATE_BREAK_ ERROR	The generation of the PROVI MESSAGE FILES was interrupted by BREAK INFO
631	BOF_PROVI_MESSAGE_ACCESS_ERROR	General access error with PROVI MESSAGES via the DIAG SERVER
632	BOF_PROVI_MESSAGE_TYPE_ERROR	There is an Invalid PROVI MESSAGE TYPE
633	BOF_PROVI_ADM_FILE_DATA_ERROR	Necessary data is NOT available in the PROVI administration file
634	BOF_PROVI_ADM_FILE_SIZE_ERROR	There are too many data entries in the PROVI administration file
635	BOF_PROVI_TEXT_FILE_NOT_EXIST_ERROR	The selected PROVI message text file is NOT available

Code	Error Text	Name and Meaning of Error
636	BOF_NO_MEMORY_EXIST_IN_SLOT_NUMBER_ ERROR	There is NO free memory in the requested SLOT NUMBER
637	BOF_PROVI_MESSAGE_FILE_DOWNLOAD_ BREAK_ERROR	The download of the PROVI MESSAGE FILES was interrupted by BREAK INFO
638	BOF_M_KEY_ADM_TIMEOUT_ERROR	The interlocking mutex for the M key administration could NOT be assigned in the preselected time
639	BOF_WRITE_DATA_TO_LONG_ERROR	On calling the BW command. the write value is too long
640	BOF_SERCOS_CHANNEL_ERROR	Access via the SERCOS channel NOT possible
641	BOF_DATE_TIME_READ_ERROR	The information on date and time could NOT be read from the controller
642	BOF_DATE_TIME_STRING_ERROR	There is a FORMAT error in the transferred date/time string
643	BOF_SYNAX_IN_SLAVE_MODE_ERROR	The addresses SYNAX controller is in SLAVE mode
644	BOF_PC_TIME_DATE_SET_STATUS_ERROR	Invalid PCTimeDateSetStatusFromDeviceAddr entry in IND_DEV.INI.
645	BOF_PC_TIME_DATE_SET_RATE_ERROR	NO or invalid PCTimeDateSetStatusRateFromDeviceAddr entry in IND_DEV.INI
646	BOF_PC_TIME_DATE_SET_DEVICE_ADDR_ERROR	NO or invalid PCTimeDateSetFromDeviceAddr entry in IND_DEV.INI
647	BOF_PCI_DRIVER_DLL_NOT_FOUND_ERROR	The PCI DRIVER DLL (PCIDP_IF32.DLL) could NOT be found
648	BOF_PCI_DRIVER_DLL_COULD_NOT_BE_LOAD_ ERROR	The PCI DRIVER DLL (PCIDP_IF32.DLL) could NOT be loaded
649	BOF_PCI_DRIVER_DLL_FUNCTION_LOAD_ERROR	The PCI DRIVER DLL (PCIDP_IF32.DLL) does NOT CONTAIN ALL NECESSARY functions
650	BOF_PCI_OBJECT_NOT_EXIST_ERROR	The addresses PCI communication object does NOT exist
651	BOF_TRIGGER_EVENT_COUNTER_TO_MUCH_ ERROR	A maximum of 1280 event handles are admissible in the trigger list
652	BOF_INVALID_VISUAL_MOTION_ASCII_ PROTOCOL_ERROR	There is an invalid VISUAL-MOTION ASCII request
653	BOF_INVALID_VISUAL_MOTION_ERROR_TEXT	There is an invalid VISUAL-MOTION ERROR TEXT answer
654	BOF_VISUAL_MOTION_ERROR	There is a VISUAL MOTION ERROR

8.3 Error Codes 1000 to 1999

Code	Error Text	Name and Meaning of Error
1001	BOF_FAULT_FCT	Invalid function code passed (e.g. "CW" for a read function).
1002	BOF_DATA_FAULT	Data is invalid.
1003	BOF_FAULT_PIPE_NR	Incorrect 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 created.
1007	BOF_PIPE_NOT_BREAK	Pipe not running.

Code	Error Text	Name and Meaning of Error
1008	BOF_NO_VALUE	No value string.
1009	BOF_BUFFER_SIZE_TO_SMALL	Buffer is 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 blocked; repeat access.
1014	BOF_NEGATIVE_ACKNOWLEDGE	Negative acknowledge for the FI command executed.
1015	BOF_PARAMETER_INVALID	Invalid parameter details.
1016	BOF_FUNCTION_INVALID	Invalid FI command.
1017	BOF_DEVICE_TIMEOUT	Timeout of NC-Task
1018	BOF_INDEX_DATA_ERROR	Index data from the resultbuf is corrupt.
1019	BOF_UNKNOWN_TOOL_STORE	Unknown type of memory (tool store)!= magazine, spindle, gripper.
1020	BOF_MAX_COUNT_ERROR_FOR_TOOL_DATA	Maximum count error for tool data.
1021	BOF_NO_TOOLMANAGMENT	No tool management.
1022	BOF_NO_TOOLMANAGMENT_FOR_PROCESS	Tool management not available for process.
1025	BOF_RESULT_BUF_TYPE_ERROR	Error result type is incorrect or not supported.
1030	BOF_NC_PACKET_IS_PRESENT	NC package already present in control.
1031	BOF_NC_PARTPROGRAM_IS_NOT_PRESENT	NC program is not present.
1032	BOF_NC_PROGRAM_DIRECTORY_IS_EMPTY	Part-directory or program directory is empty.
1033	BOF_NC_PROGRAM_COMPILER_ERROR	Error flag set by program.
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 can not be read in.
1037	BOF_NC_PARTPROGRAM_DIR_NOT_PRESENT	Program directory does not exist.
1038	BOF_NC_PARTPROGRAM_DIR_READ_ERROR	Program directory can not be read in.
1039	BOF_PIPE_CYCLE_LIST_EMPTY	Pipe request list is empty.
1040	BOF_PIPE_RUN	Pipe already running.
1041	BOF_ITEM_DATA_INVALID	Partial result is invalid.
1042	BOF_FUNC_INVALID_PARAM	Invalid parameter for function
1043	BOF_PIPE_NO_FREE_PIPE	All pipes already assigned.
1044		Communication channel is already running.
1301	Exception	
1302	No "Common" information section present	
1303	No "Common" information key present	
1304	FI job interrupted	
1305	Download control file cannot be read	
1310	No "Package" information section present	
1311	No "Package" information key present	
1312	“Package” key value not permitted	
1315	No ListOfPrograms section present	

Code	Error Text	Name and Meaning of Error
1321	No NC program section present	
1322	Program key value not permitted	
1323	No NC program info present	
1324	Process number does not conform	
1325	Data file cannot be read	
1331	Parameter record not active	
1332	No memory for creating object	
1333	Invalid value passing	
1334	Invalid access mode	
1340	No valid function call	
1341	No COMPILER info section present	
1342	No COMPILER info key present	
1350	No variable/events/D-correction section present	
1351	Variable/event/D-correction data file cannot be read	
1355	No events section present	
1360	No D-correction section present	
1361	No D-correction parameter record present	
1370	Invalid transfer of parameters	
1371	Invalid key	
1372	Invalid PLC variables type	
1373	Telegram delimiter reached	
1374	Configuration file does not exist	
1375	Section "project" does not exist	
1376	Key "project" does not exist	
1377	Section "variables" does not exist	
1378	Key "variables" does not exist	
1379	Section "buffer" does not exist	
1380	Key "buffer" does not exist	
1381	Invalid parameter transfer at "buffer" section	
1382	Invalid parameter transfer at "buffer" section	
1383	Section "trigger\condition" does not exist	
1384	Key "trigger\condition" does not exist	
1390	No I/O tables section	
1391	I/O tables value do not exist	
1392	Short identification is invalid	
1501	BOF_FUNC_NAME_LIMIT150	Name of interface 'B' functions is too large.
1502	EXCEPTION	Internal error.
1503	EXCEPTION	Internal error.
1504	EXCEPTION	Internal error.
1505	EXCEPTION	Internal error.

Code	Error Text	Name and Meaning of Error
1506	EXCEPTION	Internal error.
1507	EXCEPTION	Internal error.
1508	EXCEPTION	Internal error.
1509	EXCEPTION	Internal error.
1510	EXCEPTION	Internal error.
1511	EXCEPTION	Internal error.
1512	BOF_FUNC_EOF_STRING_150	FI command incomplete.
1513		Maximum number of lines has been reached
1514		Maximum number of columns has been reached

8.4 Error Codes 2000 to 2999

Code	Meaning
2001	No channel free.
2002	Channel already open.
2003	Channel cannot be closed.
2004	Channel not open.
2005	Re-initialization error.
2006	Channel cannot be opened.
2007	Version is incompatible to file "LOGINTFC.EXE".
2008	Channel flags are blocked.
2009	Access to controls temporarily blocked due to download.
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	Pass format of routine "GetSysMsg" is faulty.
2018	System message (SysMsg) cannot be issued.
2019	DMA request is still active.
2020	Invalid FI command code.
2021	Invalid result type.
2022	Result too long for receive buffer.
2023	Invalid FI command during group request.
2024	Empty result buffer
2025	Request too long for request buffer.
2026	Faulty input format.
2050	"LOGINTFC.DAT" file cannot be opened.
2051	No channel free.

Code	Meaning
2052	Communication process (COM task) not responding.
2053	"LOGINTFC.EXE" file not found.
2059	Error message from the LOG process.
2060	False syntax in the command string
2081	The addressed controller is off line (DeviceStatus=OFF)
2082	The addresses controller is in monitor mode
2126	Faulty result type
2127	Faulty request on response
2150	"LOGINTFC.DAT" file cannot be opened.
2154	File Version Mismatch.
2155	"LOGINTFC.DAT" file is too large.
2156	Internal configuration error.
2157	Faulty ChFreeList from GetDeviceCommAddrExtend()
2160	Invalid command string.
2161	Telegram code not implemented.
2162	Parameter outside the limit value.
2163	Invalid parameter syntax.
2164	Unknown PLC variable.
2165	Not enough parameters transmitted.
2166	PLC map file cannot be opened.
2167	PLC variable type not implemented.
2168	PLC variable reference error.
2169	Date cannot be edited.
2170	Checksum error.
2171	Undefined telegram code.
2172	Missing processing rule.
2173	Too much data for the response telegram.
2174	Unknown additional diagnostics information.
2175	Unknown unit.
2176	PLC variable is larger than 240 byte.
2177	Device has no PLC
2178	NDC parameter too large
2179	Result buffer too small
2180	No axis defined
2181	The addressed controller is off line (DeviceStatus=OFF)
2182	The addresses controller is in monitor mode
2201	Input string "Date-Time" not in format: "DD.MM.YY hh:mm:ss".
2202	Effective data length of SIS telegram is too large.
2304	Specified file not found.

8.5 Error Codes 4000 to 4999

Code	Error Text	Name and Meaning of Error
4000		An error has been detected in checking the composition of the request of the “BR_NPA1.....” and ff command. (see also FI command: "NPA1_/?") The following error messages in the error window provide additional information regarding the error.
	ERROR : invalid ParNo/value	An incorrect parameter number has been transmitted.
	ERROR : invalid ParNo/value	An error has been detected in checking transmission of the parameter. The possible cause of this is an invalid parameter name or an error in the order in which the entry was made. The first parameter number must be smaller than the second parameter number. (see also FI command: "NPA1_/?")
	invalid Parametervalue or No.:[<ParNr>]	An error has been detected in checking the command. Either a directory number has been selected that is outside of the range of validity or a parameter name is invalid.
	[No.] missing Startparameter	The command has not been passed on in its entirety.
	Illegal start parameter value [wrong value passed]	An incorrect value has been detected for the parameter number.
	ERROR : different Parametertypes	Requesting different types of parameter within one request command is not possible.
	ERROR : Second ParNo before First ParNo	The parameter request must be made starting from the lower number and moving to the higher number. (see also FI command: "NPA1_/?")
	ERROR : Invalid startparameter - ProcNo out of Range	When requesting one or more process parameters, an invalid definition range has been detected. A request is only possible within the NC process numbers [0...6].
	ERROR : Invalid startparameter - AxesNo out of Range	When requesting one or more axis parameters, an invalid definition range has been detected. A request is only possible within 1 to 20 or 32.
4001	ERROR : invalid function	The FI command contains an invalid parameter.
4002	NO_PARAMETER_DATA_FOUND	The requested parameter(s) do not exist. Either parameters have been requested that have not already been defined or the appropriate parameter has been removed. Check all entries and make sure that the corresponding data exists in BOF menu item <F5> (Parameters).
4003	Verz_No_Out_of_Range	An invalid range has been detected when checking the command passed. Check the directory number entries.
4004	BR_NPA_No_Data_File_exists	The attempt to read data from a file could not be executed. Re-check your entries for possible processes or axes on the definition range. Otherwise, try to view the data using BOF menu item <F5> (Parameters). The data may not exist or the installation has not been made correctly . In this case, please contact our customer service department.

Code	Error Text	Name and Meaning of Error
4005	BR_NPA_No_INI_File_exists	Parameter data could not be read from an initialization file. Possible causes are: The file does not exist. There has been an installation error or the file has been deleted accidentally. ⇒ Execute Update/Installation • There is an error in the file. The file has been accidentally edited or illegally copied. Data recognition has thereby been rendered invalid. ⇒ Carry out an update installation or contact our customer service department. • The file has been damaged, either by a system crash or by a defect on the storage media. ⇒ Contact our customer service department.
4006	Device Address out of Range	A system outside the definition range has been selected in the command.
4007	Buffer error detected =[Error Code]	Internal error. The data range set for provision of the results is not large enough. This problem can be remedied as follows: • Request fewer data. Use a group request. • Increase the memory made available for the data range when creating the application yourself. ⇒ Contact our customer service department.
4012	Create_DLL_Error detected!	The result buffer could not be initialized. Contact our customer service department.
4013	Function will not run for DLL-Version-Mode:[DLL-Version]	An attempt has been made to execute a command that is not available in the existing DLL version.
4014	Corrupted Parameter Identification = [Parameterident.]	Initialization of the required data memory is not possible die to an error in parameter recognition. Check to make sure that there is a valid parameter record for all devices. If necessary, re-transmit the parameter(s) to the controls. If the error remains, or the parameter(s) cannot be transmitted, then please contact our customer service department.
4015	Wrong version installed	This error message appears always appears on starting the GUI when the memory could not be initialized based on the version being used. Up to and including version 18, error code 4109 is returned. From version 19, the corrected error code 4015 is returned.
4017	**OK** (none Parameterset in CNC) – finished function FillParamDataInCncDataMap	This text message only appears in the starting-up phase with the setting "/U0" of the start parameter (for TSRPG25I.EXE) if an empty parameter name has been transmitted. This means that no parameter record is as yet in the control. No error is returned.
4100	Couldn‘t open ParameterIndexFile: [File Error=xxxx]	An error has been detected when attempting to open the parameter directory file. Any of the following could trigger this error: • Versions do not conform ⇒ Parameters need to be converted. (refer to Parameters) • The parameter directory file has been accidentally destroyed. • The disk drive is faulty.
4102	ParameterIndexFile has wrong structure	An error has been detected when reading the parameter directory file indicating that the data in the file is not in the correct format. Check this by running Converting Parameters. If the error continues to occur after this then you must contact our customer service department.

Code	Error Text	Name and Meaning of Error
4103	to much Indizies found – File has wrong structure ?:	An error has been detected when creating the directory data. More directories have been recognized than allowed by the definition range. Probably the parameter directory data is from an earlier version. Execute Converting Parameters. If the error persists, please contact our customer service department.
4104	Invalid parameter value detected	An invalid range was detected when initializing (booting up the GUI). Contact our customer service department.
4105	Can't create Parameterindexbuffer: [filename]	No data could be provided in the memory. Close other applications to free up enough memory for the compilation of the data.
4108	Don't found the Parameter: [Parameternummer]	This message text only appears in the starting phase with the setting "/U0" (in case of TSRPG25I.EXE). This error code is only returned when an attempt has been made to request a non-defined parameter.
4109	Didn't get BOF-Version – BOF installed? [error code]	The attempt to determine the GUI version has failed. Contact our customer service department.
4110	Couldn't load Parameters in shared Memory – Error= [ErrorCode]	Initialization failed when starting the GUI. Contact our customer service department.
4111	Invalid parameter value Cxx.053 [Cxx.053 <Value>]	Initialization failed when starting the GUI. An invalid axis meaning has been detected in the current parameter record of a device. Switch the corresponding system to offline and correct the appropriate parameter record. After you have done this, the system should be brought back online and the altered parameter record should then once more be transmitted to the controls. If the problem persists, please contact our customer service department.
4200	Invalid start parameter	This message text only appears in the starting phase with the setting "/U0" (in case of TSRPG25I.EXE). This error code is always returned when a parameter request has been made outside the definition range. Otherwise, please contact our customer service department.
4201	Invalid parameter type	A parameter request has been made with a non-defined parameter type. Check the entry and/or request
4202	Buffer size not enough	The result of the parameter request cannot be transmitted as the transmission range is not large enough. For applications that you have created yourself, increase the size of the transmission range. Otherwise, please contact our customer service department.
4203	Error detect by ReadPar_Value – can't read Data [Error number or directory number]	The requested parameter could not be formed or found. Re-check your request or contact our customer service department.
4204	Could not find direct. entry	No error message is emitted. The error code is always returned when, after a request for a particular parameter directory entry, the **parameter number** has not been found.
4205	Function will not running by InterFace-Version: [Version]	During the command request, the program has detected that it cannot be run on this version. Contact our customer service department.
4220	Invalid Save Order by Save function please test the ParType by Save_Begin;	The "writing parameters" function has been repeatedly started before the previously started command has been completed.
4221	invalid IndexNo by Save[ParameterNumber]	The parameter number is outside the definition range.
4222	co_str_ConWData_Buffer_Size_to_small [defined size 2000]	This message text only appears in the starting phase with the setting "/U0" (for TSRPG25I.EXE). The error code is always returned if the defined memory range in the program is too small. In this case, please contact our customer service department.

Code	Error Text	Name and Meaning of Error
4223	WriteError by Config-SCR-File = [error number]	An error has been detected when writing the configuration parameters. The function has been cancelled. Contact our customer service department.
4224	SaveError – Couldn't rename DAT->old [file name]	This message text only appears in the starting phase with the setting "/U0" (for TSRPG25I.EXE). The attempt to rename the original file could not be executed. Check the properties for the relevant parameter file and also the remaining free space on the storage medium.
4225	SaveError – Couldn't rename tmp → dat ⇒ copy old to DAT[Dateiname]	This message text only appears in the starting phase with the setting "/U0" (for TSRPG25I.EXE). The attempt to recopy the newly created file could not be executed. Check the properties for the relevant parameter file and also the remaining free space on the storage medium.
4226	Missing file = [file name]	The previously created file could not be found or opened. Check the free memory on the storage medium.
4227	Create instance failed - can't save Data	An internal error has occurred. Contact our customer service department.
4228	Can't create file = [file name <additional info>]	The specified file could not be created acc. to the additional info. Check the amount of free memory on the storage medium and the access properties of the directory concerned. Otherwise, please contact our customer service department.
4229	Can't create file = [file name]	Specified file could not be found. Check your entry. Perhaps an incorrect directory number has been entered.
4230	ConWData_Error_by_WPar_Begin	This error code is reported after an error has occurred in transmitting an error code to the function interface.
4231	ConWData_Error_by_WPar_End	This error code is reported after an error has occurred in transmitting an error code to the function interface.
4232	Error detect by WritePar_Value - can't save Data:[file name or parameter value]	This error message is only displayed internally when in debug mode. The value of a parameter could not be transferred to the file specified. Contact our customer service department.
4233	Attention - Return value of process definition undefined	This error message is only displayed internally when in debug mode. An error has been detected in the generation of the process definition. Check the process definitions within the processing of the parameter. Otherwise, please contact our customer service department.
4234	Can't actualize direct.line [Parameter directory line]	The specified parameter directory line could not be updated.
4235	Can't actualize date or length in directory line	The date or length could not be updated when the parameter directory line was being updated. Contact our customer service department.
4236	CreateFiErrorResult_DLL failed	This error message is only displayed internally when in debug mode. The error message could not be transmitted to the FI. Contact our customer service department.
4237	Can't write by undefined parameter number [Parameter number]	An attempt has been made to write a non-defined parameter for this type of parameter. Check your entry. Check, e.g. that parameters exist for the various axis types.
4238	Cxx.083 : more as defined Elements for Cxx.083 found:	An attempt has been made to transmit a larger number of compensation values than is listed as the max. range of a compensation list. Re-check your entry. A maximum of 1000 values can be included in a list.
4239	Installation error - missing file: [File name]	Specified file not found. Re-run the update installation. If the error persists after this then you must contact our customer service department.

Code	Error Text	Name and Meaning of Error
4240	Invalid Parameter value =[parameter line]	An invalid range has been detected in the specified parameter line. Re-check your entry.

Converting Parameters

An update installation of the Rexroth Indramat GUI automatically results in a parameter conversion from version "xx" to the next version "yy". Parameters can be converted by calling the "COPAxxyy.EXE" conversion program in standard installation directory "C:\Programme\Indramat\MTGUI\Bin". Both wildcards "xx" and "yy" represent the directory ID from which version and into which version the conversion is carried out.

Note: In case of an error, you can start the conversion program "COPAxxyy.EXE" with the starting parameter "/?" to receive additional messages.

8.6 Error Codes 5000 to 5999

Code	Name and Meaning of Error
5001	Alias used is not defined (is not yet used)
5002	Invalid device address
5003	Syntax error in the device address
5007	Invalid device address
5010	No request active
5011	Error in request string
5012	No response buffer specified
5013	Wrong mode during cyclic login (internal error)
5014	Invalid receive telegram
5015	Invalid receive telegram
5050	No access to remote PC possible
5051	Remote connection has been canceled
5052	Network interface cannot be initialized
5101	Unexpected general error (internal error)
5102	Memory error
5151	No memory for data to be sent
5152	No memory for data to be received
5153	No memory for telegram to be received
5401	Unspecific error
5402	Invalid parameter passed to function
5403	Transfer timeout, remote PC not ready, or network connection down
5404	Send failed; error sending to a remote PC
5405	Memory shortage; in remote access of the interface
5406	Invalid connection to a remote PC
5407	Service disabled

Code	Name and Meaning of Error
5408	Connection to remote partner aborted
5409	Invalid parameter hook ID; Sys Message Handling
5410	Invalid device number

8.7 Error Codes 6000 to 6999

Code	Error Text	Name and Meaning of Error
6001	BOF_C_TYP_FAULT	Transmitted data type not OK.
6002	BOF_C_LEN_FAULT	Transmitted data length not OK.
6003	BOF_C_DEV_FAULT	Transmitted system number not OK.
6004	BOF_C_PAKNR_FAULT	Transmitted package number not OK.
6005	BOF_C_PROZ_FAULT	Transmitted process number not OK.
6006	BOF_C_PROG_FAULT	Transmitted 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-program directory
6009	(BOF_C_NCPROG_CREATED)	NC program exists (where check =1)
6010	BOF_C_DESCR_FAULT	Identifier, 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 available
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 to 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	Test error after commands to be compiled.
6023	BOF_C_CURS_FILE_ERROR	Handling error in NCCPxx.DAT file.
6024	BOF_C_TOOL_SETUP_LIST_NOT_READ	Error in setup list
6025	BOF_C_TOO_MUCH_TOOLS_IN_LIST	More tools in the setup list than in the parameters.

8.8 Error Codes 7000 to 7999

All error codes – except for error code 7000, which shows a syntax error in the compiled NC program – normally require you to contact Rexroth Indramat for further clarification of their cause. Either this is a software error or files for the GUI have been deleted or corrupted.

Note: As for all error codes, additional information regarding an error that has occurred can be requested via the "General error result line" (see chapter entitled “Error Codes"). The error information informs the user in plain text about the cause of the error.

Code	Meaning and Notes Regarding Diagnosis and Troubleshooting
7000	Syntax error in NC program. The "General Error Result Line" contains further information.
7002	File with incorrect information. The "General Error Result Line" contains the file name and the line.
7005	File not found. The "General Error Result Line" contains the file name.
7006	File cannot be created. The "General Error Result Line" contains the file namen.
7008	File cannot be read. The "General Error Result Line" contains the file name.
7009	Error in connecting the function interface. No connection can be made to the device (control unit) via the function interface.
7015	Too many axes defined. More than 9 Axes used in NC process.
7016	Invalid number of parameters. The number of parameters in the "NCPRG.CFG" file has been exceeded.
7017	Axis name is invalid. The axis name in axis parameter "CXX.001" or "CXX.075" is invalid.
7018	Axis meaning is invalid. The axis meaning in axis parameter "CXX.053" is invalid.
7019	Maximum axis speed is invalid. The value of axis parameter "CXX.016" is invalid.
7020	Maximum axis acceleration is invalid. The value of axis parameter "CXX.018" is invalid.
7021	Lowest run time of an NC record = [2.5...30ms]. The counter value of the parameter "METB" in the NC options of the BOF/GBO is outside the allowed range.
7022	Lowest run time of an NC record is invalid. The counter value of the parameter "METB" in the NC options of the BOF/GBO is invalid.
7023	Only 4 or 5 decimal places are allowed. Process parameter "BXX.002" is invalid.
7024	Invalid counter value. The counter value of parameter "VFBT" or "BBTC" in the NC options of the BOF/GBO is invalid.
7025	Only 0 (mm) or 1 (inch) permitted! The process parameter "BXX.001" is invalid.
7026	Counter value outside the allowed range. Axis parameter "CXX.006" is smaller than 0.1.

Code	Meaning and Notes Regarding Diagnosis and Troubleshooting
7027	Internal block number is invalid. The block numbers of the NC program file are in the wrong order.
7028	Block number in the file is invalid. The "General Error Result Line" (p. 8-1) contains the names of the file in which the block numbers are incorrect.
7070	Counter value outside the permitted range (1..10). The counter value of parameter "BBTC" in the NC options of the BOF/GBO is outside the permitted range.
7077	Counter value outside the permitted range (1.0.25). The counter value of parameter "VFBT" in the NC options of the BOF/GBO is outside the permitted range.
7083	Invalid parameter. The "General Error Result Line" contains an invalid control parameter.

8.9 Error Codes 8000 to 8999

Code	Error Text	Name and Meaning of Error
8000	OUTOFMEMORY	Heap memory is full
8001	PARAMETER_FAILURE	Error in transmitting parameter (response telegram)
8002	INVALIDARG	Incorrect request string
8003	REQUEST_NOT_FILLED	Internal run error
8004	GET_ATTRIBUT_FAILED	Incorrect attribute contained in response telegram
8005	WALK	Internal run error
8006	EXTRACT_COMMON_INFO_FAILED	Error in transmitting parameter (response telegram)
8007	WRONG_DATA_SIZE	Undefined data length in the response
8008	ELEMENT_UNEXPECTED	Unexpected coding in BW_SPA1
8009	SERCOS_LONG_TO_ASCII	Result conversion error.
8010	VERSION_MISMATCH	Command did not yet exist for set IfDllMode.
8011	ERROR_BYTE_INFO	Error reading error byte information
8012	CANT_OPEN_MODULDEF_INI	The "Moduldef.ini" file cannot be opened.
8013	WRONG_PROFILE_FILENAME	Wrong profile file name
8014	WRONG_SECTION_INFORMATION	Wrong section information in 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., incorrect device or module parameter).
8019	LANGUAGE_NOT_FOUND	Language not supported
8020	Reserved	Reserved
8021	MODUL_NOT_FOUND	Module not found; (e.g., missing keyword module name).
8022	DEVICE_ADDR_GENERAL_NOT_FOUND	No device entry found.
8023	FB_NOT_FOUND	No function component found; (e.g., error or message keyword missing).
8024	DEVICE_ADR_FALSE	Device address not in the valid range.
8025	MODULE_NO_FALSE	Module number not within valid range (0-99)

Code	Error Text	Name and Meaning of Error
8026	KEY_WORD_FALSE	Wrong keyword; (e.g., no ModulY in section name [DeviceAddrX\ModulY])
8027	MODULE_ASSIGN_PROCESS	No module can be found for the specified process.
8028	PROCESS_NO_FALSE	Process number not within valid range (0-31)
8031	RESULT_TYPE_INVALID	Invalid result type.
8032	E_COM_SIS_TEL_TOO_LONG	Transmitted length of telegram exceeds maximum SIS telegram length.
8033	E_COM_SIS_TEL_POS	Telegram position addressed is outside the SIS telegram range.
8034	E_COM_SIS_TEL_NO_LEN	SIS telegram length is "0"
8035	E_COM_OPERATING_SYSTEM_NOT_ SUPPORTED	Operating system is not supported.
8036	SERCOS_ASCII_TO_LONG	Error in converting the value to be written.
8038	PROCESS_NOT_DEFINED	The process addressed does not exist
8039	NO_TOOLMANAGEMENT	The tool management is not activated for the process
8040	WRONG_TOOL_NUMBER	Invalid tool number
8041	WRONG_SPINDLE_NUMBER	Wrong spindle number
8042	WRONG_GRIPPER_NUMBER	Wrong gripper number
8043	UNKNOWN_TOOL_STORE	Unknown tool store (memory)
8044	INVALID_VALUE	Value or element of the value list not correctly formatted
8045	MUTEX_TIMEOUT	The command access control was not quit in time
8046	UNKNOWN_DEVICETYPE	An unknown device type has been detected

8.10 Error Codes 10000 and above

Code	Meaning
10001	The WinHMI component is not installed.
10002	Incorrect WinHMI version
10021	DDS not installed
10022	Incorrect DDS version
10101	Incorrect version of the function interface.
10102	The "CreateGroup" routine has failed.
10103	Error in command string.
10104	Unknown variable requested.
10105	Error in determining the status.
10107	"HMI_Data.DLL" file not found.
10110	WinHMI has not been started in the same process
10111	Error on determination of the ProVi message.
10112	ProVi message type does not exist.
10201	DDS not ready yet.
10210	Unknown error
10211	Unknown function has been requested
10212	Function of this controller is not available.
10221	Internal error.
10222	Invalid module number.
10223	Faulty message ID.
10224	Invalid diagnosis type
10225	Incorrect detail type
10226	Unknown detail
10227	Unknown step.
10228	Unknown sequencer.
10229	Sequencer is not disturbed.
10230	Step is not disturbed.
10231	Unknown variable requested.
10232	Unknown error ID.
10233	Unknown message number.
10234	Retain variable up or download error
10235	Retain variable up or download message

8.11 Error Codes 35000 and above

Code	Error Text	Name and Meaning of Error
35500	PARA_NOT_DEFINED	CMOS parameter not yet defined. Recovery: Describe CMOS parameter.
35501	VALUE_TYPE_INVALID	Invalid coding type.
35502	ERROR_VERSION_MISMATCH	Command does not yet exist for set IfDllMode.

8.12 Error Codes 100000 and above

Code	Error Text	Name and Meaning of Error
100101	FS_NO_TEXT_FILE_ACCESS	Record file is opened 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	An interface object could not be created.
100105	FS_ERROR_SIM5	Without "iMTc" ID
100106	FS_ERROR_FILETYP	Different file type
100107	FS_ERROR_FILEVERSION	Current file version is larger than file version
100108	FS_ERROR_FILELENGTH	Current file length != ID length
100109	FS_ERROR_FILEDATE	File date != ID date
100110	FS_ERROR_FILETIME	File time != ID time
100111	FS_ERROR_FILENAME	File name != ID name
100112	FS_ERROR_CHECKSUM	Checksum is incorrect
100113	FS_ERROR_FILE_NOT_EXIST	File does not exist
100114	FS_ERROR_FILE_MIN_LENGTH	File with ID must be at least 65 byte.
100115	FS_ERROR_T04	Without "iT04" ID
100116	FS_ERROR_FILE_NOT_OPEN	File cannot be opened.
100117	FS_ERROR_NO_SIGN	File has no ID (sign)
100118	FS_ERROR_MMIVERSION	GUI version is smaller than file version.

8.13 Error Codes 110000 and above

Code	Error Text	Name and Meaning of Error
110001	BOF_MAP_VERSION_FUNC_ERROR	Incorrect Dll mode set
110002	BOF_MAP_FILE_VERSION_ERROR	Incorrect file version number
110003	BOF_MAP_LONGID_VERSION_ERROR	If the long ID version is invalid
110004	BOF_MAP_LONGID_INVALID_ERROR	If the long ID is invalid
110005	BOF_MAP_LONGID_PARAM_ERROR	Missing parameter in split long ID
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 test of map file long ID is negative.
110010	BOF_MAP_ID_ERROR	Long ID comparison is 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 parameters for a function.
110014	BOF_MAP_INVALID_DEVICE_NO	Transmitted device number does not agree with the system number in the MAP.
110015	BOF_MAP_INVALID_STATUS	Invalid access status
110016	BOF_MAP_ACCESS_ERROR	Access to a MAP when MAP has not been loaded, incorrect 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_ERROR52	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 $\rightarrow$ variable not found.
110022	BOF_MAP_LONGID_DIFFERENT_TO_MAP12	MAP file long ID is different from PLC long ID
110023	BOF_MAP_INVALID_ARRAY_INDEX50	Invalid array index
110024	BOF_MAP_INVALID_STRING_INDEX51	Invalid string index
110025	BOF_MAP_NO_CREATED_MAP_ACCESS	MAP access has not been generated
110026	BOF_MAP_LONGID_INVALID_NO_MAP_ERROR	Long ID is not valid and no PLC MAP access has yet been initialized.
110027	BOF_MAP_OUTOFMEMORY	No more memory available for creating object
110028	BOF_MAP_STRUCT_ELEMENT_NO_FOUND	Structure element does not exist
110029	BOF_MAP_STRUCT_ELEMENT_NO_FOUND	Global management information has not been created
110030	BOF_MAP_DOWNLOAD_STATUS	Access to MAP during a download
110031	BOF_MAP_NO_DOS_PCL	System is not DOS - PLC
110032	BOF_MAP_NO_WIN_PCL	System is not WIN - PLC
110033	BOF_MAP_EXCEPTION	Exception has been triggered while determining PLC address
110034	BOF_WINPCL_INSTANCE	WinPcl object instance could not be created
110035	BOF_WINPCL_CREATE	Error creating access to WinPcl

Code	Error Text	Name and Meaning of Error
110036	BOF_WINPCL_INIT	Error initializing WinPcl; current PLC long ID differs from database
110037	BOF_WINPCL_ACCESS	WinPcl access object has not been created
110038		Parameter too small
110040		An administration error occurred
110041		Initiialization of the WinPCL address server not possible
110050	BOF_MAP_COMMON_FILETOOL_ERROR	Basic number cErrorGroup_filetool
110100	BOF_MAP_COMMON_MAP_BAS_C_ERROR	Basic number ErrorGroup_map_bas_c
110150	BOF_MAP_COMMON_LKENN_ERROR	Basic number cErrorGroup_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; to clearly identify the error, the error number is added to the BASE.

8.14 Error Codes 210000 and above

Code	Meaning
210917	String is too long
210920	String does not begin with '
210921	String does not end with '
210923	Counter value has been exceeded
210924	Minimum counter value not achieved
210925	Incorrect counter format

8.15 SERCOS Error

Code	Error Messages in Serial Protocol
0x0000	No error in NC/MMI service channel.
0x0001	NC/MMI service channel not opened.
0x0009	Incorrect access to Element 0.
0x0090	The control is currently busy. The request is not possible at the moment. Please try again later.
0x00A0	"invalid request" e.g. access to S-/P parameter in initialization mode.
0x00B0	"invalid element" Only the operating date element is valid for write access.
0x00C0	"invalid drive address" The drive address is larger than allowed or the drive is not active within the SERCOS ring (deactivated or does not exist).
0x00F0	"Fatal software error" An internal CLC error has occurred during parameter transmission (see C-0-0041), that has affected the exchange of data.
0x1001	IDN does not exist.
0x1009	Incorrect access to Element 1.
0x13E8	Transmission error.
0x13E9	Drive does not exist.
0x13EA	Cancellation of data transmission when requested.
0x13EB	Request data channel is closed.
0x13EC	System faults
0x2001	Name does not exist.
0x2002	Name transmitted too short.
0x2003	Name transmitted too long.
0x2004	Name cannot be changed.
0x2005	Name currently write-protected.
0x3002	Attribute transmitted too short.
0x3003	Attribute transmitted too long.
0x3004	Attribute cannot be changed.
0x3005	Attribute currently write-protected.
0x4001	Unit does not exist.
0x4002	Unit transmitted too short.
0x4003	Unit transmitted too long.
0x4004	Unit cannot be changed.
0x4005	Unit currently write-protected.
0x5001	Minimum input value does not exist.
0x5002	Minimum input value transmitted too short.
0x5003	Minimum input value transmitted too long.
0x5004	Minimum input value cannot be changed.
0x5005	Minimum input value currently write-protected.

Code	Error Messages in Serial Protocol
0x6001	Maximum input value does not exist.
0x6002	Maximum input value transmitted too short.
0x6003	Maximum input value transmitted too long.
0x6004	Maximum input value cannot be changed.
0x6005	Maximum input value currently write-protected.
0x7002	Date transmitted too short.
0x7003	Date transmitted too long.
0x7004	Date can not be changed.
0x7005	Date currently write-protected.
0x7006	Date smaller than min. input value.
0x7007	Date larger than max. input value.
0x7008	Incorrect date.
0x7009	Date is write-protected by password.
0x700A	The operating date is currently write-protected as it has been configured cyclically (IDN is configured with MDT or AT; therefore, writing via the service channel is not allowed)
0x700B	Invalid list element IDN is not supported, value outside of the input limits).
0x700C	Operating data write-protected at the moment because of other settings (e.g. parameter, operating mode, drive release, drive ON, etc.).
0x700D	"Length of date cannot currently be changed" The length of the date cannot be changed in the current mode.
0x700E	"Length of the date cannot currently be changed" The length of the date is permanently write-protected.
0x7010	Command already active
0x7011	Command cannot be interrupted
0x7012	Command cannot be executed at present (e.g. command cannot be activated in this phase)
0x7013	Command cannot be executed (invalid or incorrect parameters)
0x710C	Date outside of figure range
0x710D	Length of date cannot be changed at present
0x710E	Length of date can not be changed.
0x8001	"Service channel is currently assigned (BUSY)" The required access is not currently possible as the service channel is assigned. Data transmission is not executed.
0x8002	"Fault in service channel" Access to the required drive is not currently possible.
0x800B	Transmission has been cancelled by the control unit as it must currently communicate with the same drive (higher priority).
0x800C	Unauthorized access (service channel is still active); last transmission has not yet been completed and a new request has been started.

8.16 Global SERCANS Error

The global SERCANS errors are not directly related to the message transmitted. These are fatal communication errors that result in the breakdown of communication with one or more drives.

The following global SERCANS error codes have been defined:

Code	Error Messages in Serial Protocol
0x8006	HS timeout
0x8007	Double AT breakdown.
0x8008	Optical waveguide ring not closed.
0x8009	Optical waveguide ring interrupted.
0x800A	"Test operation: zero bit current or continuous light". Test operation is set on the SERCANS assembly in order to check the optical transmission route on the SERCOS interface.
0xC001	Invalid command control word.
0xC002	IDN is not a command.
0xC003	Command channel cannot currently be activated.
0xD001	Drive error (status class 1, S-0-0011).
0xD004	Command cannot be executed in drive.
0xF001	"Configuration error". An error occurred on configuration of the setpoint or actual value channel: a) too many setpoint or actual values have been configured, or the setpoint or actual values are not supported.
0xF002	"Error in calculating time slot" a) Telegram configured is too long b) Communication cycle time is too short
0xF003	Incorrect phase details from the NC
0xF004	"Error in life counter". The control no longer accesses the DPR of SERCANS cyclically.
0xF005	SERCANS: internal error
0xF006	"Copy times too long". The copy times of the command values and actual values taken together are larger than the time between the end of the last ATs and the beginning of the MDTs.
0xF007	Checksum error (Y parameter).
0xF008	Breakdown of input signal SYNCIN
0xF009	Error in storing the system parameter or the system parameter has been changed. A check of the min/max values failed.
0xF00A	Parameter is write-protected.

8.17 Structure of Error File after Download

If an error is generated during downloading, this error is recorded in an error file and the *DownloadError* key is set to the value "YES" in the download file. The error file corresponds to the download file with the ending ".ERR".

An error is assigned to the section in which it occurs. The following error keys are set.

XX indicates a serial number

Key: ErrorLine_XX
Line in which the error has occurred (optional).

Key: ErrorLink_XX
Shows the connection with the error.

Key: ErrorSection_XX
Identifies the section in which the error has occurred.

Key: ErrorText_XX
Error text generated during the program flow.

Key: ErrorToken_XX
Error number

Key: TextFileName_XX
Name of the text file used (without language extension).

Examples for the MWCX Device Group

NC Cycle Download: CCA

Example: Download file

[Common]

ErrorSection_01= ListOfCycPrograms

ErrorToken_01 = 1014

[ListOfCycPrograms]

ErrorLink_01= K:\Program Files\Indramat\Mtgui\Project_000\Cyc-Prg-00-01.dat

ErrorToken_01 = 1014

Beispiel: NC cycle program file

[Data]

ErrorColumn_01 = 7

ErrorLine_01 = 6

ErrorText_01 = Format error in the NC program

ErrorToken_01 = 1014

NC D-Correction Download: DCA

Example: Download file

[Common]

;Errors of a general kind. Cannot be assigned to a specific section. E.g., Compiler not found

ErrorToken_01 = 1234

ErrorText_01=."Download error"

ErrorSection_01= DCorrection_1

[DCorrectionPackage_Info]

ErrorToken_01=5678

ErrorText_01=Missing info value

[DCorrection_1]
ErrorToken_01=1014
ErrorText_01=Transmission error

NC Program Download: NCA

Example: Download file
[Common]
ErrorSection_01= ListOfNCPrograms
ErrorToken_01 = 1014

[ListOfNCPrograms]
ErrorLink_01= K:\Program Files\Indramat\Mtgui\Project_000\\NC-PRG-00-01.Dat
ErrorToken_01 = 1014

Example: NC program file
[Data]
ErrorColumn_01 = 7
ErrorLine_01 = 6
ErrorText_01 = Format error in the NC program
ErrorToken_01 = 1014

NC Events Download: NEA

Example: Download file
[Common]
;Errors of a general kind. Cannot be assigned to a specific section. E.g., Compiler not found
ErrorToken_01 = 1234
ErrorText_01=."Download error"
ErrorSection_01= NCEvents_1

[NCEventsPackage_Info]
ErrorToken_01=5678
ErrorText_01=Missing info value

[NCEvents_1]
ErrorToken_01=1014
ErrorText_01=Transmission error

NC Zero Point Download: NUA

Example: Download file
[Common]
;Errors of a general kind. Cannot be assigned to a specific section. E.g., Compiler not found
ErrorToken_01 = 1234
ErrorText_01=."Download error"
ErrorSection_01= OffsetDataPackage_Info

[OffsetDataPackage_Info]
ErrorToken_01=5678
ErrorText_01=Missing info value

[OffsetData_0\0\0] ; process 0, zero point data base 0, axis X

ErrorToken_01=1014

ErrorText_01=Transmission error

NC Variables Download: NVA

Example: Download file

[Common]

;Errors of a general kind. Cannot be assigned to a specific section. E.g., Compiler not found

ErrorToken_01 = 1234

ErrorText_01=."Download error"

ErrorSection_01= NCVariables_1

[NCVariablesPackage _Info]

ErrorToken_01=5678

ErrorText_01=Missing info value

[NCVariables_1]

ErrorToken_01=1014

ErrorText_01=Transmission error

9 Answers to Frequently Asked Questions

9.1 Function Interface FAQs

In this chapter, you'll find a collection of **F**requently **A**sked **Q**uestions from our customers' feedback on the Rexroth Indramat Function Interface.

Question 1 A message box appears when starting my application. Has the message box been issued by the function interface?

Note: As message boxes are entered in the Windows NT Task Manager as "applications", it is easy to see what has actually issued the message box.

Answer To do this, open Windows NT Task Manager e.g. using key combination: <Strg>+<Shift>+<Esc>

Highlight the message box entry in the "applications" tab page and click with the right-hand mouse button.

Note: The key combination <Ctrl>+<F10> does not work here for the right mouse button!

Select the “Switch to Process” command in the context menu that opens for the highlighted object.

If one of the following processes is displayed

- LOGINTFC.exe
- BOFINTFC.exe
- COMINTFC.exe

then this is a basic process of the function interface.

Question 2 Can group requests also be issued via the ”DataTransfer” routine?

Answer No, the "DataTransfer" routine is only for issuing single read or write requests. Group requests are issued via the routines for cyclic reading via pipes.

Question 3 Why does the login procedure for my application to the function interface take so long?

Answer During the function interface initialization phase numerous safety checks are carried out.

9.2 Windows NT FAQs

This chapter contains FAQs regarding Windows NT from customer feedback.

Question 1 How can I log in with my name and password automatically (AutoLogin)?

Answer You must make the following entries in the Windows NT registry using the registry editor "REGEDT32" under key

HKEY_LOCAL_MACHINE\ Software Microsoft\ Windows NT\ Current Version Winlogon:

Value	**Type**	**Content**	**Info**
AutoAdminLogon	REG_SZ	1	Switch AutoLogin on/off
DefaultUserName	REG_SZ	<user name>	User login name
DefaultPassword	REG_SZ	<password>	User password (a password must exist)
DefaultDomainName	REG_SZ	<domain name>	Login must be carried out on another computer

Note: No further message box will appear. If you want to log in using another name then you must keep the <Shift> key pressed during the starting procedure. You will now be prompted to enter your name and password.

If no password is entered in the registry then AutoLogin will only function once and Windows will then reset ”AutoAdminLogon” to "0". Entering the password is absolutely essential. Please note that the password is then visible in the registry for anyone to see!

10 Reference to Literature

10.1 Information in Rexroth Indramat Literature

[1] More detailed information regarding acceleration value and value range is contained in the Rexroth Indramat documentation:

NC Programming Instructions, chapter entitled “Interpolation Requirements/ Programmable Acceleration ACC”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[2] More detailed information regarding the arguments of the trigonometric functions is contained in the Rexroth Indramat documentation:

NC Programming Instructions, chapter „Angle Dimension for Trigonometrical Functions RAD, DEG“, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[3] More detailed information regarding the axis speeds is contained in the Rexroth Indramat documentation:

NC Programming Instructions, chapter entitled “Interpolation Functions/ Linear Interpolation, Rapid Traverse Rate G00”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE

Parameter Description, chapter entitled “Maximum Track Speed”, DOK-MT*CNC-PAR*DES*Vxx-AW0x-DE.

[4] More detailed information regarding the structure of an NC block is contained in the Rexroth Indramat documentation:

NC Programming Instructions, chapter entitled “Elements of an NC Block”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[5] Additional information regarding the reference spindle as well as NC programming of the cutting speed is contained in the Rexroth Indramat documentation:

NC Programming Instructions, chapter entitled “Spindle Speed, Constant Cutting Speed G96 / Selection of Reference Spindle SPF”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[6] More detailed information regarding D-corrections is contained in the Rexroth Indramat documentation:

”NC Programming Instructions Vxx", chapter entitled “D-Corrections”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[7] More detailed information regarding events and their treatment is contained in the Rexroth Indramat documentation:

”NC Programming Instructions Vxx", chapter entitled “Events”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[8] More detailed information regarding tool management is contained in the Rexroth Indramat documentation:

”NC Programming Instructions Vxx", chapter “Commands for Tool Management / Cutter Selection E”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[9] More detailed information regarding feedrate override is contained in the Rexroth Indramat documentation:

"CNC/PLC Interface Description xxVRS",
chapter entitled "Feedrate and Spindle Override PxxCSOVRD",
DOK-MTC200-SPS*GWY*Vxx-AW0x-DE.

[10] More detailed information regarding the feedrate is contained in the Rexroth Indramat documentation:

"CNC NC Programming Instructions Vxx", chapter entitled "Feedrate",
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[11] More detailed information regarding the mode of operation of the G functions, as well as classification of the G-code groups, is contained in the Rexroth Indramat documentation:

"NC Programming Instructions Vxx",
chapter entitled "Table of G-Code Groups",
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[12] More detailed information regarding the mode of operation of the M functions, as well as classification of the M function groups, is contained in the Rexroth Indramat documentation:

"NC Programming Instrucitons Vxx",
chapter entitled "Table of M Function Groups",
DOK-MTC200-NC**PRO*Vxx-AW01x-DE.

[13] More detailed information regarding the diagnostics system and the accompanying types of message is contained in the Rexroth Indramat documentation:

”xxVRS GUI”, Application Description, Chapter 3 "Diagnostics", DOK-MTC200-GBO*GEN*Vxx-AW0x-DE.

[14] More detailed information regarding the machine parameters and their classification within the system, process, axis and APR-SERCOS parameters can be found in the Rexroth Indramat documentation:

"MTC200/MT-CNC MCI Operating Instructions xxVRS",
chapter entitled "Machine Parameters",
DOK-MTC200-GBO*MCI*Vxx-AW0x-DE

"Parameter Description",
DOK-MT*CNC-PAR*DES*Vxx-AW0x-DE.

[15] More detailed information regarding the elements of an NC record and the note is contained in the Rexroth Indramat documentation:

”CNC NC Programming Instructions Vxx", chapter entitled “NC Word”,
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[16] More detailed information regarding the display possibilities within user interfaces, as well as the definition of axis data, is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC xxVRS GUI", chapter "Survey of Axis Data", DOK-MTC200-GBO*GEN*Vxx-AW0x-DE.

[17] More detailed information regarding the NC data structure is contained in the Rexroth Indramat documentation:

”NC Programming Instructions Vxx", chapter entitled “Program and Data Organization”, DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[18] More detailed information regarding the rapid override is contained in the Rexroth Indramat documentation:

"CNC/PLC Interface Description xxVRS",
chapter entitled "Feedrate and Spindle Override"; "Rapid Override PxxCSOVRD",
DOK-MTC200-SPS*GWY*Vxx-AW0x-DE.

[19] Additional information regarding the selection of the reference spindle in the NC program is contained in the Rexroth Indramat documentation:

"NC NC Programming Instructions Vxx", Application Description,
chapter entitled "Spindle Speed", "Selecting the Reference Spindle SPF"
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[20] Additional information regarding the selection of the spindle speed in the NC program is contained in the Rexroth Indramat documentation:

"NC Programming Instructions Vxx",
chapter entitled "Additional Functions M" / "Switching Gear",
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[21] More detailed information regarding the spindle override is contained in the Rexroth Indramat documentation:

"CNC/PLC Interface Description xxVRS",
chapter entitled "Feedrate and Spindle Override PxxCSOVRD",
DOK-MTC200-SPS*GWY*Vxx-AW0x-DE.

[22] More detailed information regarding the axis speeds is contained in the Rexroth Indramat documentation:

"MT-CNC Numeric Control for Multiple Axis, Multiple Process Applications", chapter entitled "Maximum Track Acceleration",
DOK-MT*CNC-PAR*DES*Vxx-AW0x-DE.

[23] More detailed information regarding the structure and elements of the tool data is contained in the Rexroth Indramat documentation:

”NC NC Programming Instructions Vxx, Application Description",
chapter entitled “Access to Tool Data by NC Program TLD”,
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[24] More detailed information regarding the use of zero-point offsets and zero offset tables is contained in the Rexroth Indramat documentation:

"NC NC Programming Instructions Vxx", Application Description,
chapter "Zero-Point Offsets, Zero Offest Tables O",
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[25] Additional information regarding the display of the axis position in the GBO is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC xxVRS GUI",
DOK-MTC200-GBO*GEN*Vxx-AW0x-DE.

[26] More detailed information on resetting the device is contained in the Rexroth Indramat documentation:

"SPS Programming Instructions xxVRS", Chapter "Putting into Operation / Start", DOK-CONTRL-SPS*PRO*Vxx-AW0x-DE.

[27] More detailed information regarding the configuration of the device axes is contained in the Rexroth Indramat documentation:

"Parameter Description", chapter "System Parameters" as well as chapter "Axis Parameters", DOK-MT*CNC-PAR*DES*Vxx-AW0x-DE.

[28] Additional information regarding process parameters and their functions as well as value ranges is contained in the Rexroth Indramat documentation:

"CNC/SPS Interface Description xxVRS Application Description, chapter "External Mechanisms", DOK-MTC200-SPS*GWY*Vxx-AW0x-DE.

[29] Additional information regarding the function of the NC parameters and the structure of the NC parameter records is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC Parameter Description xxVRS", DOK-MTC200-PAR*DES*Vxx-AW0x-DE.

[30] More detailed information concerning the PLC Programming System is contained in the Rexroth Indramat documentation:

"PLC Programming Instructions xxVRS Application Description" DOK-CONTRL-SPS*PRO*Vxx-AW0x-DE.

[31] More detailed information regarding the structure of NC packages is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC NC Programming Instructions xxVRS", chapter „Sub-Programs", DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[32] More detailed information regarding the structure of tool lists is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC xxVRS GUI", chapter "Tool Data Handling BOF" and chapter "Tool Data Handling GBO", DOK-MTC200-GBO*GEN*Vxx-AW0x-DE.

[33] More detailed information regarding the use of machine data is contained in the Rexroth Indramat documentation:

"CNC Machine Data xxVRS Application Description" DOK-MT*CNC-MAS*DAT*Vxx-AW0x-DE.

[34] Additional information regarding process parameters and their functions as well as value ranges is contained in the Rexroth Indramat documentation:

"Parameter Description" chapter "Process Parameters" DOK-MT*CNC-PAR*DES*Vxx-AW0x-DE.

[35] Additional information regarding process parameters and their functions as well as value ranges is contained in the Rexroth Indramat documentation:

"MT-CNC Numeric Control for Multiple Axes, Multiple Process Applications", Chapter 2 "Process Parameters", DOK-MT*CNC-PAR*DES*V15-ANW1-DE-E.

[36] Additional information regarding module configuration and the structure of the "Moduldef.ini" file is contained in the following Rexroth Indramat documentation:

"Diagnostics and Message System for HMI System ProVi", chapter "Configure Moduldef.ini", DOK-MTC200-DIAG*PROVI*-AW0x-DE.

[37] More detailed information on selecting the NC program and the NC memory is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC xxVRS GUI", chapter entitled "Operation Survey of the Administration of NC Programs", DOK-MTC200-GBO*GEN*Vxx-AW0x-DE.

[38] More detailed information regarding the contents of parameter records is contained in the Rexroth Indramat documentation:

"MTC200/MT-CNC Parameter Description xxVRS",
chapter entitled "Processing / Displaying Contents of Parameter Records",
DOK-MTC200-PAR*DES*Vxx-AW0x-DE.

[39] More detailed information regarding NC variables is contained in the Rexroth Indramat documentation:

"NC Programming Instructions Vxx",
chapter entitled "Assigning Variables and Mathematical Functions",
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[40] More detailed information regarding servo lag is contained in the Rexroth Indramat documentation:

"NC Programming Instructions Vxx",
chapter entitled "Movement Records and Interpolation Requirements",
DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[41] Additional information regarding the function of the standard and product-specific SERCOS parameters (S and P) is contained in the Rexroth Indramat Documentation:

"DIAX04 Drive with Servo Functions", Appendix A Description of Parameters, DOK-DIAX04-SSE-02VRS**-FKB1-DE-P.

[42] More detailed information regarding the communication phases is contained in the Rexroth Indramat documentation:

"DIAX04 Drive with Servo Functions", General Instructions on Putting into Operation, DOK-DIAX04-SSE-xxVRS**-FKBx-DE.

[43] More detailed information regarding tool management is contained in the Rexroth Indramat documentation:

"MT-CNC Numeric Control for Multiple Axes, Multi-Process Applications, Tool Data Handling",
DOK-MT*CNC-BOF*WZH*Vxx-AW0x-DE.

[44] More detailed information regarding zero offsets is contained in the Rexroth Indramat documentation:

MTC200/MT-CNC NC Programming Instructions xxVRS",

chapter entitled "Zero Offset", and chapter entitled "Reading and Writing of the Zero Offset Data from the NC Program OTD",

DOK-MTC200-NC**PRO*Vxx-AW0x-DE.

[45] More detailed information regarding SERCANS errors is contained in the Rexroth Indramat documentation:

"SERCANS /SERCVME SERCOS Interface Assemblies with Universal µP Interface or VMEbus", Application Description, System Structure and Axis Structure.

[46] Additional information regarding the function of the SERCANS System Parameters (Y) is contained in the Rexroth Indramat Documentation:

"SERCANS SERCOS Interface Assemblies", Chapter 10 "Description of Parameters",
DOK-SERCAN-SER-VxxVRS**-AW0x-DE.

11 Glossary

ANSI

American **N**ational **S**tandards **I**nstitute, American standards institute which developed the ANSI emulation (refer also to: ANSI code).

ANSI code

Standard code standardized by ANSI making it possible to generate pictures, animation and texts plus sounds from the PC speaker from sequences of ANSI control frequencies. Method primarily used in mailboxes to design a GUI. The ASCII Code is generally also referred to as ANSI Code. These characters are generated in a document by pressing the <AltGr> key together with inputting the relevant code.

ASCII

American **S**tandard **C**ode for **I**nformation **I**nterchange, widely used code, especially on domestic and personal computers, for displaying numbers, letters and special characters; designed as a 7-bit code with a character set of 128 characters or as an 8-bit code with a character set of 256 characters including upper and lower case letters. The unassigned (free) eighth and ninth bits, formatted in bytes, are used as parity bits (check bits).

Operating date

The operating date is data block element 7 of a parameter. The value of the parameter is stored in it.

BTV20

The BTV20 is a machine user terminal in which one or more NC control units with PLC or one or more stand-alone PLCs can be integrated. The number of units that can be integrated depends on their configuration. In contrast to the BTV30, the BTV20 offers an application-oriented function keyboard with the following features:

- Front panel made of 4mm aluminum with camfered edges.
- Fully flush, chemical-resistant polyester foil with embossed stamping.
- Integrated EMC-compatible glass plate to protect the display.
- Integrated machine keys with intermediate plate to avoid double entries and accidental operation.
- Key switch to lock security functions.

BTV30

The BTV30 has all the functions and operating elements of a complete industrial PC. In addition to the 10.4" flat colour display and a complete ASCII keyboard with cursor block, keyboard mouse, there is a standard disk drive and the connection for an external keyboard located behind a lockable hinged cover. Ten PC function keys are positioned under the display, while eight machine function keys are located to the right of the display. These are either fed to the outside via a socket or, in the case of an integrated PLC unit, connected directly to the PLC. Genuine key elements are embedded in the stable PVC/ABS front panel, allowing fatigue-free programming even over prolonged periods of time. The display is protected by a stable, EMC-compatible glass plate. When the hinged disk flap is closed the BTV30 complies with IP65 type of protection and is resistant to all known coolants and lubricants.

Client

A client is a computer system or a process that requests the services of another computer system or process. The term also indicates a workplace computer that can use the services and resources (e.g., printer, scanner, plotter) of a server or also of other clients. Generally speaking it has significantly fewer privileges than the server.

DDE

Dynamic **D**ata **E**xchange is a standard defined by Microsoft for data exchange between programs under MS-Windows from Version 3.0. DDE permits files or parts of files to be linked between two applications that support the DDE standard. A distinction is made between a source application (server) and a target application (client), wherein the target application maps a complete or partial copy of the server file. If the data in the source file is changed, then this information is transferred to the target application via the link and dynamically updated there. However, a DDE communication can also simply be used to exchange commands and instructions between two programs.

DLL

Dynamic **L**ink **L**ibrary, is a library linked to a program during its run-time. DLLs are special files for Windows from which, for example, functions, dialog boxes or symbols are loaded by applications. They simplify programming and save hard disk space if, for example, the same functions are required by several user programs. A dynamic link library offers various advantages: It need only be loaded when required and until then it does not use any RAM.

Dual port RAM

A Dual port RAM is a memory area between two connected users: the actual controllers, and the GUI (PC). This memory area, used by both users, permits only limited data traffic.

If, for example, the control unit wants to send a message to the user interface (GUI/PC), it sends it first to the dual port RAM. A cyclic mechanism running on the PC side detects and fetches the new information, acknowledges it for the control unit and passes it on for processing (display on the GUI).

If the situation is reversed and the GUI wants to send a message to the control unit, it is likewise sent first to the dual port RAM. Just as with the control unit, a cyclic mechanism detects and fetches this new information, acknowledges it for the PC and passes it on for processing.

This ensures that both users only exercise controlled data trafficking and otherwise work in their own, separate memory areas.

Remote device

This term depends on the point of view (refer also to Local Device). Where a user program (client) is concerned, the device is **not** at the PC on which the client is running, but at a PC within the PC network.

Device

A device is the control hardware, drive device or I/O device.

Device Type

The device type indicates which Rexroth Indramat device is meant e.g., MTC200-P-G2, MTVNC, ISP200-R-G2.

FarDevice

Configuring the PC network requires, in addition to the list of PCs, a list of FarDevices. The FarDevice address has been introduced for clear, linear addressing of devices in the PC network. This means that every device available in the PC network receives an additional address (FarDevice address). Any device that is available in the PC network has a FarDevice address and is referred to as a FarDevice. The list of FarDevices is produced on the basis of devices previously connected to each PC.

Local device

This term depends on the point of view (refer also to Remote Device). As far as the user program is concerned the device is at the PC on which the client is running.

MPI

(**MPI** = **M**ulti-**P**rotocol-**I**nterface). The Rexroth Indramat MPI provides a standardized user interface for the communication interfaces Profibus-FMS, MMS-Ethernet (MAP), TCP/IP and FIPWAY on PC units under the Windows NT 4.0 operating system. The MPI provides the interfaces for implementing client and server user programs. The MPI communication driver makes the connection between the MPI and the function interface. This in turn provides the connection with all protocols supported by MPI to Rexroth Indramat devices that can be configured via the function interface.

MTC200

MTC200 is a PC-based generation of control units. The MTC200 system integrates all the functions of an NC and PLC control unit including the entire drive technology. Components of this system include MTC-P, MTC-R, MTS-P, MTS-R.

Up to seven independent NC processes can be controlled with the MTC200. The seven NC processes can be divided to a max. of 32 axes. This means that the MTC is both multi-axis and multi-process capable.

MTC-P

The MTC-P is a powerful CNC control unit in ISA bus plug-in card format for insertion into an industrial PC; it is part of the MTC200 range. It consists of a basic unit with the processor system of an NC and an integrated axis processor to which a maximum of eight drives can be attached via a SERCOS interface. Expansion by a maximum of up to three axis processor modules allows 32 drives to be controlled at the highest level of expansion. These can be divided between 7 processes. Together with the MTS-P01.1 PLC control unit, this unit forms a compact and flexible solution for tool machine control.

MTC200-P-G2

See MTC-P.

MTC-R

The RECO-based NC unit, MTC-R, contains a complete MTC-P-compatible NC processor and an axis processor module for controlling up to 8 digital drives via a SERCOS interface. Up to three further axis processor modules can be plugged in via the PC/104 bus to achieve the maximum capacity of 32 drives. While there is room for one additional PC/104 module in a unit of normal width (for controlling up to 16 drives), a double-width housing is available for additional axis processors.

The MTC-R cannot function on its own; it always requires an MTS-R as an offshore adaptive control. Both units are connected via a local bus for communication between the MTC-R and MTS-R. Both units are then slotted together into an RMB02.2 or RMB02.4 rack. If required, as described for the MTS-R, additional I/O modules can be used to supplement the local I/O level.

MTC200-R-G2

See MTC-R.

MTS-R

The RECO-PLC is a powerful but small PLC that is compatible with the PLC in the MTC200 control system. The housing is designed in accordance with IP20. It can be used as a stand-alone PLC and together with an MTC-R as a slave PLC. To facilitate networking several PLC control units and connecting to a programming device or PC there is an RS 232/RS 485 programming interface available. For connection to a printer, read/write memory, or screen a free serial interface (RS 232/RS 422) is available.

The MTS-R01.1 occupies one module slot in the RMB02 rack, and the MTS-R02.1 occupies two slots. This means that the ISP200-R is capable of driving the bus for up to 15 further I/O modules. An internal local bus communicates with the MTC-R NC control unit that is part of the system by means of an adapter board.

As an option, the MTS-R01 and the MTS-R02 can be equipped with the open fieldbus interfaces INTERBUS or PROFIBUS-DP. Distributed I/O peripheral units, each with up to 4096 inputs and 4096 outputs, can be connected via these optional interfaces. In addition, the MTS-R02 can be equipped with a serial interface (2 x RS 232 and 2 x RS 422).

PC network

The PC network is made up of several PCs connected at the level of the function interface. The PC network comprises the PCs that are used to control a machine (primarily graphics, operation and programming).

Process

The process is the combination of functions and axes relevant to the control work carried out by the MTC200-P-G2 control unit; this combination is processed in a processing unit within the control unit. Each MTC-P / MTC-R (MTC200-P-G2, MTC200-R-G2) has a maximum of 7 processes.

RECO

RECO is a modular I/O system for rapid signal exchange with the PLC. The racks for two or four I/O modules can be mounted on a standardized top-hat rail. There is a choice between analog and digital inputs and outputs as well as serial interfaces.

Registry

See Registry Database.

Registry database

In Windows NT the registry database replaces most of the Win3.x INI files (these files still exist in Windows NT but are generally only used by 16-bit programs). Information concerning configuration is entered in the registry both by Windows NT and by all 32-bit programs.

Registry editor

Entries in the registry database "Registry" are changed using the registry editor. The editor is in the Windows system directory and is called REGEDT32.EXE (→ "Start" → "Run" REGEDT32).

RS232

Serial interface with 9-pin or 25-pin connection conforming to the V.24 standard and developed by the EIA for communication with devices; maximum 115,200 bit/s; often used for connection between computers and modems.

Server

A server is a computer that holds ready applications and documents for connected computers (clients) to access. The term also indicates a program that provides certain services that can be accessed using programs specially adapted to the server.

Shared memory

An area in the computer’s RAM that can be accessed by several processes (applications).

System200

The Rexroth Indramat System200 is a comprehensive scalable control and drive program for the entire machine and system structure. There is a choice of various software packages (WIN-HMI, MPI, function interface etc.,) for one single PC hardware platform (MTC200), plus various screens (BTV20, BTV30, etc.), application-optimized drives (DIAX04, ECODRIVE, etc.) and periphery connections (Profibus-DP, Profibus-FMS, SERCOS interface, etc.).

Thread

Threads are objects within processes which carry out programming instructions. They permit various simultaneous actions within a process and enable a process to execute different parts of a program simultaneously on different processors.

WinHMI

(**WinHMI** = **WIN**dows based - **H**uman **M**achine **I**nterface). The WinHMI software package is a standard GUI for automated production.

12 List of Figures

13 Index

A

B

C

Rexroth Indramat

D

E

F

G

I

K

L

M

N

O

P

R

S

T

W

Z

14 Service & Support

14.1 Helpdesk

Unser Kundendienst-Helpdesk im Hauptwerk Lohr am Main steht Ihnen mit Rat und Tat zur Seite. Sie erreichen uns

- telefonisch: **+49 (0) 9352 40 50 60**
  über Service Call Entry Center Mo-Fr 07:00-18:00
- per Fax: **+49 (0) 9352 40 49 41**
- per e-Mail: **service@indramat.de**

Our service helpdesk at our headquarters in Lohr am Main, Germany can assist you in all kinds of inquiries. Contact us

- by phone: **+49 (0) 9352 40 50 60**
  via Service Call Entry Center Mo-Fr 7:00 am - 6:00 pm
- by fax: **+49 (0) 9352 40 49 41**
- by e-mail: **service@indramat.de**

14.2 Service-Hotline

Außerhalb der Helpdesk-Zeiten ist der Service direkt ansprechbar unter

+49 (0) 171 333 88 26
oder **+49 (0) 172 660 04 06**

After helpdesk hours, contact our service department directly at

+49 (0) 171 333 88 26
or **+49 (0) 172 660 04 06**

14.3 Internet

Unter **www.boschrexroth.de** finden Sie ergänzende Hinweise zu Service, Reparatur und Training sowie die **aktuellen** Adressen *) unserer auf den folgenden Seiten aufgeführten Vertriebs- und Servicebüros.

☐ Verkaufsniederlassungen
☐ Niederlassungen mit Kundendienst

Außerhalb Deutschlands nehmen Sie bitte zuerst Kontakt mit unserem für Sie nächstgelegenen Ansprechpartner auf.

*) Die Angaben in der vorliegenden Dokumentation können seit Drucklegung überholt sein.

At **www.boschrexroth.de** you may find additional notes about service, repairs and training in the Internet, as well as the **actual** addresses *) of our sales- and service facilities figuring on the following pages.

☐ sales agencies
☐ offices providing service

Please contact our sales / service office in your area first.

*) Data in the present documentation may have become obsolete since printing.

14.4 Vor der Kontaktaufnahme... - Before contacting us...

Wir können Ihnen schnell und effizient helfen wenn Sie folgende Informationen bereithalten:

detaillierte Beschreibung der Störung und der Umstände.

Angaben auf dem Typenschild der betreffenden Produkte, insbesondere Typenschlüssel und Seriennummern.

Tel.-/Faxnummern und e-Mail-Adresse, unter denen Sie für Rückfragen zu erreichen sind.

For quick and efficient help, please have the following information ready:

1. Detailed description of the failure and circumstances.
2. Information on the type plate of the affected products, especially type codes and serial numbers.
3. Your phone/fax numbers and e-mail address, so we can contact you in case of questions.

14.5 Kundenbetreuungsstellen - Sales & Service Facilities

Deutschland – Germany

vom Ausland: (0) nach Landeskennziffer weglassen!
from abroad: don't dial (0) after country code!

Vertriebsgebiet Mitte Germany Centre Rexroth Indramat GmbH Bgm.-Dr.-Nebel-Str. 2 / Postf. 1357 97816 Lohr am Main / 97803 Lohr **Kompetenz-Zentrum Europa** Tel.: +49 (0)9352 40-0 Fax: +49 (0)9352 40-4885	**SERVICE** **CALL ENTRY CENTER** **MO – FR** **von 07:00 - 18:00 Uhr** **from 7 am – 6 pm** **Tel. +49 (0) 9352 40 50 60** service@indramat.de	**SERVICE** **HOTLINE** **MO – FR** **von 17:00 - 07:00 Uhr** **from 5 pm - 7 am** **+ SA / SO** **Tel.: +49 (0)172 660 04 06** **oder / or** **Tel.: +49 (0)171 333 88 26**	**SERVICE** **ERSATZTEILE / SPARES** verlängerte Ansprechzeit - extended office time - ♦ nur an Werktagen - only on working days - ♦ von 07:00 - 18:00 Uhr - from 7 am - 6 pm - **Tel. +49 (0) 9352 40 42 22**
Vertriebsgebiet Süd Germany South Rexroth Indramat GmbH Landshuter Allee 8-10 80637 München Tel.: +49 (0)89 127 14-0 Fax: +49 (0)89 127 14-490	Vertriebsgebiet West Germany West Bosch Rexroth AG Regionalzentrum West Borsigstrasse 15 40880 Ratingen Tel.: +49 (0)2102 409-0 Fax: +49 (0)2102 409-406	Gebiet Südwest Germany South-West Bosch Rexroth AG Service-Regionalzentrum Süd-West Siemensstr.1 70736 Fellbach Tel.: +49 (0)711 51046–0 Fax: +49 (0)711 51046–248	Gebiet Südwest Germany South-West Bosch Rexroth AG Regionalzentrum Südwest Ringstrasse 70 / Postfach 1144 70736 Fellbach / 70701 Fellbach Tel.: +49 (0)711 57 61–100 Fax: +49 (0)711 57 61–125
Vertriebsgebiet Nord Germany North Bosch Rexroth AG Walsroder Str. 93 30853 Langenhagen Tel.: +49 (0) 511 72 66 57-0 Service: +49 (0) 511 72 66 57-256 Fax: +49 (0) 511 72 66 57-93 Service: +49 (0) 511 72 66 57-95	Vertriebsgebiet Mitte Germany Centre Bosch Rexroth AG Regionalzentrum Mitte Waldecker Straße 13 64546 Mörfelden-Walldorf Tel.: +49 (0) 61 05 702-3 Fax: +49 (0) 61 05 702-444	Vertriebsgebiet Ost Germany East Bosch Rexroth AG Beckerstraße 31 09120 Chemnitz Tel.: +49 (0)371 35 55-0 Fax: +49 (0)371 35 55-333	Vertriebsgebiet Ost Germany East Bosch Rexroth AG Regionalzentrum Ost Walter-Köhn-Str. 4d 04356 Leipzig Tel.: +49 (0)341 25 61-0 Fax: +49 (0)341 25 61-111

Europa (West) - Europe (West)

vom Ausland: (0) nach Landeskennziffer weglassen, Italien: 0 nach Landeskennziffer mitwählen
from abroad: don't dial (0) after country code, Italy: dial 0 after country code

Austria - Österreich	Austria – Österreich	Belgium - Belgien	Denmark - Dänemark
Bosch Rexroth GmbH Bereich Indramat Stachegasse 13 1120 Wien Tel.: +43 (0)1 985 25 40 Fax: +43 (0)1 985 25 40-93	Bosch Rexroth GmbH Gesch.ber. Rexroth Indramat Industriepark 18 4061 Pasching Tel.: +43 (0)7221 605-0 Fax: +43 (0)7221 605-21	Bosch Rexroth AG Electric Drives & Controls Industrielaan 8 1740 Ternat Tel.: +32 (0)2 5830719 - service: +32 (0)2 5830717 Fax: +32 (0)2 5830731 indramat@boschrexroth.be	BEC A/S Zinkvej 6 8900 Randers Tel.: +45 (0)87 11 90 60 Fax: +45 (0)87 11 90 61
Great Britain – Großbritannien	Finland - Finnland	France - Frankreich	France - Frankreich
Bosch Rexroth Ltd. Rexroth Indramat Division Broadway Lane, South Cerney Cirencester, Glos GL7 5UH Tel.: +44 (0)1285 863000 Fax: +44 (0)1285 863030 sales@boschrexroth.co.uk service@boschrexroth.co.uk	Bosch Rexroth Oy Rexroth Indramat division Ansatie 6 017 40 Vantaa Tel.: +358 (0)9 84 91-11 Fax: +358 (0)9 84 91-13 60	Bosch Rexroth S.A. Division Rexroth Indramat Avenue de la Trentaine (BP. 74) 77503 Chelles Cedex Tel.: +33 (0)164 72-70 00 Fax: +33 (0)164 72-63 00 **Hotline:** +33 (0)608 33 43 28	Bosch Rexroth S.A. Division Rexroth Indramat ZI de Thibaud, 20 bd. Thibaud (BP. 1751) 31084 Toulouse Tel.: +33 (0)5 61 43 61 87 Fax: +33 (0)5 61 43 94 12
France - Frankreich	Italy - Italien	Italy - Italien	Italy - Italien
Bosch Rexroth S.A. Division Rexroth Indramat 91, Bd. Irène Joliot-Curie 69634 Vénissieux – Cedex Tel.: +33 (0)4 78 78 53 65 Fax: +33 (0)4 78 78 53 62	Bosch Rexroth S.p.A. Via G. Di Vittoria, 1 20063 Cernusco S/N.MI Tel.: +39 02 92 365 1 +39 02 92 365 326 Fax: +39 02 92 365 500 +39 02 92 365 516378	Bosch Rexroth S.p.A. Via Paolo Veronesi, 250 10148 Torino Tel.: +39 011 224 88 11 Fax: +39 011 224 88 30	Bosch Rexroth S.p.A. Via del Progresso, 16 (Zona Ind.) 35020 Padova Tel.: +39 049 8 70 13 70 Fax: +39 049 8 70 13 77
Italy - Italien	Italy - Italien	Netherlands – Niederlande/Holland	Netherlands - Niederlande/Holland
Bosch Rexroth S.p.A. Via Mascia, 1 80053 Castellamare di Stabia NA Tel.: +39 081 8 71 57 00 Fax: +39 081 8 71 68 85	Bosch Rexroth S.p.A. Viale Oriani, 38/A 40137 Bologna Tel.: +39 051 34 14 14 Fax: +39 051 34 14 22	Bosch Rexroth B.V. Kruisbroeksestraat 1 (P.O. Box 32) 5281 RV Boxtel Tel.: +31 (0)411 65 19 51 Fax: +31 (0)411 65 14 83 www.boschrexroth.nl	Bosch Rexroth Services B.V. Technical Services Kruisbroeksestraat 1 (P.O. Box 32) 5281 RV Boxtel Tel.: +31 (0)411 65 19 51 Fax: +31 (0)411 67 78 14 services@boschrexroth.nl
Norway - Norwegen	Spain - Spanien	Spain – Spanien	Sweden - Schweden
Bosch Rexroth AS Rexroth Indramat Division Berghagan 1 or: Box 3007 1405 Ski-Langhus 1402 Ski Tel.: +47 (0)64 86 41 00 Fax: +47 (0)64 86 90 62 jul.ruud@rexroth.no	Bosch Rexroth S.A. Divisiòn Rexroth Indramat Centro Industrial Santiga Obradors s/n 08130 Santa Perpetua de Mogoda Barcelona Tel.: +34 9 37 47 94 00 Fax: +34 9 37 47 94 01	Goimendi S.A. División Rexroth Indramat Parque Empresarial Zuatzu C/ Francisco Grandmontagne no.2 20018 San Sebastian Tel.: +34 9 43 31 84 21 - service: +34 9 43 31 84 56 Fax: +34 9 43 31 84 27 - service: +34 9 43 31 84 60 sat.indramat@goimendi.es	Rexroth Mecman Svenska AB Rexroth Indramat Division - Varuvägen 7 (Service: Konsumentvägen 4, Älfsjö) 125 81 Stockholm Tel.: +46 (0)8 727 92 00 Fax: +46 (0)8 647 32 77
Sweden - Schweden	Switzerland West - Schweiz West	Switzerland East - Schweiz Ost	
Rexroth Mecman Svenska AB Indramat Support Ekvändan 7 254 67 Helsingborg Tel.: +46 (0) 42 38 88 -50 Fax: +46 (0) 42 38 88 -74	Bosch Rexroth Suisse SA Département Rexroth Indramat Rue du village 1 1020 Renens Tel.: +41 (0)21 632 84 20 Fax: +41 (0)21 632 84 21	Bosch Rexroth Schweiz AG Geschäftsbereich Indramat Hemrietstrasse 2 8863 Buttikon Tel. +41 (0) 55 46 46 111 Fax +41 (0) 55 46 46 222	

Rexroth Indramat

Europa (Ost) - Europe (East)

vom Ausland: (0) nach Landeskennziffer weglassen
from abroad: don't dial (0) after country code

Czech Republic - Tschechien Bosch -Rexroth, spol.s.r.o. Hviezdoslavova 5 627 00 Brno Tel.: +420 (0)5 48 126 358 Fax: +420 (0)5 48 126 112	Czech Republic - Tschechien DEL a.s. Strojírenská 38 591 01 Zdar nad Sázavou Tel.: +420 566 64 3144 Fax: +420 566 62 1657	Hungary - Ungarn Bosch Rexroth Kft. Angol utca 34 1149 Budapest Tel.: +36 (1) 42 23 200 Fax: +36 (1) 42 23 201	Poland – Polen Bosch Rexroth Sp.zo.o. ul. Staszica 1 05-800 Pruszków Tel.: +48 22 738 18 00 – service: +48 22 738 18 46 Fax: +48 22 758 87 35 – service: +48 22 738 18 42
Poland – Polen Bosch Rexroth Sp.zo.o. Biuro Poznan ul. Dabrowskiego 81/85 60-529 Poznan Tel.: +48 061 847 64 62 /-63 Fax: +48 061 847 64 02	Rumania - Rumänien Bosch Rexroth Sp.zo.o. Str. Drobety nr. 4-10, app. 14 70258 Bucuresti, Sector 2 Tel.: +40 (0)1 210 48 25 +40 (0)1 210 29 50 Fax: +40 (0)1 210 29 52	Russia - Russland Bosch Rexroth OOO Wjatskaja ul. 27/15 127015 Moskau Tel.: +7-095-785 74 78 +7-095 785 74 79 Fax: +7 095 785 74 77 laura.kanina@boschrexroth.ru	Russia - Russland ELMIS 10, Internationalnaya 246640 Gomel, Belarus Tel.: +375/ 232 53 42 70 +375/ 232 53 21 69 Fax: +375/ 232 53 37 69 elmis_ltd@yahoo.com
Turkey - Türkei Bosch Rexroth Otomasyon San & Tic. A..S. Fevzi Cakmak Cad No. 3 34630 Sefaköy Istanbul Tel.: +90 212 541 60 70 Fax: +90 212 599 34 07	Slowenia - Slowenien DOMEL Otoki 21 64 228 Zelezniki Tel.: +386 5 5117 152 Fax: +386 5 5117 225 brane.ozebek@domel.si		

Africa, Asia, Australia – incl. Pacific Rim

Australia - Australien AIMS - Australian Industrial Machinery Services Pty. Ltd. 28 Westside Drive Laverton North Vic 3026 Melbourne Tel.: +61 3 93 243 321 Fax: +61 3 93 243 329 Hotline: +61 4 19 369 195 terryobrien@aimservices.com.au	Australia - Australien Bosch Rexroth Pty. Ltd. No. 7, Endeavour Way Braeside Victoria, 31 95 Melbourne Tel.: +61 3 95 80 39 33 Fax: +61 3 95 80 17 33 mel@rexroth.com.au	China Shanghai Bosch Rexroth Hydraulics & Automation Ltd. Waigaoqiao, Free Trade Zone No.122, Fu Te Dong Yi Road Shanghai 200131 - P.R.China Tel.: +86 21 58 66 30 30 Fax: +86 21 58 66 55 23 gf.zhu_sh@boschrexroth.com.cn	China Bosch Rexroth China Ltd. 15/F China World Trade Center 1, Jianguomenwai Avenue Beijing 100004, P.R.China Tel.: +86 10 65 05 03 80 Fax: +86 10 65 05 03 79
China Bosch Rexroth China Ltd. Guangzhou Repres. Office Room 1014-1016, Metro Plaza, Tian He District, 183 Tian He Bei Rd Guangzhou 510075, P.R.China Tel.: +86 20 8755-0030 +86 20 8755-0011 Fax: +86 20 8755-2387	China Bosch Rexroth (China) Ltd. A-5F., 123 Lian Shan Street Sha He Kou District Dalian 116 023, P.R.China Tel.: +86 411 46 78 930 Fax: +86 411 46 78 932	China Melchers GmbH BRC-SE, Tightening & Press-fit 13 Floor Est Ocean Centre No.588 Yanan Rd. East 65 Yanan Rd. West Shanghai 200001 Tel.: +86 21 6352 8848 Fax: +86 21 6351 3138	Hongkong Bosch Rexroth (China) Ltd. 6th Floor, Yeung Yiu Chung No.6 Ind Bldg. 19 Cheung Shun Street Cheung Sha Wan, Kowloon, Hongkong Tel.: +852 22 62 51 00 Fax: +852 27 41 33 44 alexis.siu@boschrexroth.com.hk
India - Indien Bosch Rexroth (India) Ltd. Rexroth Indramat Division Plot. A-58, TTC Industrial Area Thane Turbhe Midc Road Mahape Village Navi Mumbai - 400 701 Tel.: +91 22 7 61 46 22 Fax: +91 22 7 68 15 31	India - Indien Bosch Rexroth (India) Ltd. Rexroth Indramat Division Plot. 96, Phase III Peenya Industrial Area Bangalore - 560058 Tel.: +91 80 41 70 211 Fax: +91 80 83 94 345 mohanvelu.t@boschrexroth.co.in	India - Indien Bosch Rexroth (India) Ltd. 1st Floor, S-10 Green Park ext. Market New Delhi – 110016 Tel.: +91 1 16 56 68 88 Fax: +91 1 16 56 68 87	Indonesia - Indonesien PT. Rexroth Wijayakusuma Building # 202, Cilandak Commercial Estate Jl. Cilandak KKO, Jakarta 12560 Tel.: +62 21 7891169 (5 lines) Fax: +62 21 7891170 - 71
Japan Bosch Rexroth Automation Corp. Service Center Japan Yutakagaoka 1810, Meito-ku, NAGOYA 465-0035, Japan Tel.: +81 52 777 88 41 +81 52 777 88 53 +81 52 777 88 79 Fax: +81 52 777 89 01	Japan Bosch Rexroth Automation Corp. Rexroth Indramat Division 1F, I.R. Building Nakamachidai 4-26-44, Tsuzuki-ku YOKOHAMA 224-0041, Japan Tel.: +81 45 942 72 10 Fax: +81 45 942 03 41	Korea Bosch Rexroth-Korea Ltd. Electric Drives and Controls Bongwoo Bldg. 7FL, 31-7, 1Ga Jangchoong-dong, Jung-gu Seoul, 100-391 Tel.: +82 234 061 813 Fax: +82 222 641 295	Korea Bosch Rexroth-Korea Ltd. 1515-14 Dadae-Dong, Saha-Ku Rexroth Indramat Division Pusan Metropolitan City, 604-050 Tel.: +82 51 26 00 741 Fax: +82 51 26 00 747 gyhan@rexrothkorea.co.kr
Malaysia Bosch Rexroth Sdn.Bhd. 11, Jalan U8/82, Seksyen U8 40150 Shah Alam Selangor, Malaysia Tel.: +60 3 78 44 80 00 Fax: +60 3 78 45 48 00 hockhwa@hotmail.com rexroth1@tm.net.my	Singapore - Singapur Bosch Rexroth Pte Ltd 15D Tuas Road Singapore 638520 Tel.: +65 68 61 87 33 Fax: +65 68 61 18 25 sanjay.nemade @boschrexroth.com.sg	South Africa - Südafrika TECTRA Automation (Pty) Ltd. 71 Watt Street, Meadowdale Edenvale 1609 Tel.: +27 11 971 94 00 Fax: +27 11 971 94 40 Hotline: +27 82 903 29 23 georgv@tectra.co.za	Taiwan Rexroth Uchida Co., Ltd. No.17, Alley 24, Lane 737 Cheng Bei 1 Rd., Yungkang Tainan Hsien Tel.: +886 6 25 36 565 Fax: +886 6 25 34 754 indra.charlie@msa.hinet.net
Thailand NC Advance Technology Co. Ltd. 59/76 Moo 9 Ramintra road 34 Tharang, Bangkhen, Bangkok 10230 Tel.: +66 2 943 70 62 +66 2 943 71 21 Fax: +66 2 509 23 62 sonkawin@hotmail.com			

Nordamerika – North America

USA Headquarters - Hauptniederlassung	USA Central Region - Mitte	USA Southeast Region - Südwest	USA SERVICE-HOTLINE
Bosch Rexroth Corporation Rexroth Indramat Division 5150 Prairie Stone Parkway Hoffman Estates, IL 60192-3707 Tel.: +1 847 6 45 36 00 Fax: +1 847 6 45 62 01 servicebrc@boschrexroth-us.com repairbrc@boschrexroth-us.com	Bosch Rexroth Corporation Rexroth Indramat Division Central Region Technical Center 1701 Harmon Road Auburn Hills, MI 48326 Tel.: +1 248 3 93 33 30 Fax: +1 248 3 93 29 06	Bosch Rexroth Corporation Rexroth Indramat Division Southeastern Technical Center 3625 Swiftwater Park Drive Suwanee, Georgia 30124 Tel.: +1 770 9 32 32 00 Fax: +1 770 9 32 19 03	- 7 days x 24hrs - **+1-800-860-1055**
USA East Region – Ost	**USA Northeast Region – Nordost**	**USA West Region – West**	
Bosch Rexroth Corporation Rexroth Indramat Division Charlotte Regional Sales Office 14001 South Lakes Drive Charlotte, North Carolina 28273 Tel.: +1 704 5 83 97 62 +1 704 5 83 14 86	Bosch Rexroth Corporation Rexroth Indramat Division Northeastern Technical Center 99 Rainbow Road East Granby, Connecticut 06026 Tel.: +1 860 8 44 83 77 Fax: +1 860 8 44 85 95	Bosch Rexroth Corporation 7901 Stoneridge Drive, Suite 220 Pleasant Hill, California 94588 Tel.: +1 925 227 10 84 Fax: +1 925 227 10 81	
Canada East - Kanada Ost	**Canada West - Kanada West**	**Mexico**	**Mexico**
Bosch Rexroth Canada Corporation Burlington Division 3426 Mainway Drive Burlington, Ontario Canada L7M 1A8 Tel.: +1 905 335 55 11 Fax: +1 905 335-41 84 michael.moro@boschrexroth.ca	Bosch Rexroth Canada Corporation 5345 Goring St. Burnaby, British Columbia Canada V7J 1R1 Tel. +1 604 205-5777 Fax +1 604 205-6944 david.gunby@boschrexroth.ca	Bosch Rexroth Mexico S.A. de C.V. Calle Neptuno 72 Unidad Ind. Vallejo 07700 Mexico, D.F. Tel.: +52 5 754 17 11 +52 5 754 36 84 +52 5 754 12 60 Fax: +52 5 754 50 73 +52 5 752 59 43 mariofelipe.hernandez@boschrexroth.com.mx	Bosch Rexroth S.A. de C.V. Calle Argentina No 3913 Fracc. las Torres 64930 Monterrey, N.L. Tel.: +52 8 333 88 34...36 +52 8 349 80 91...93 Fax: +52 8 346 78 71 mario.quiroga@boschrexroth.com.mx

Südamerika – South America

Argentina - Argentinien	Argentina - Argentinien	Brazil - Brasilien	Brazil - Brasilien
Bosch Rexroth S.A.I.C. "The Drive & Control Company" Acassusso 48 41/47 1605 Munro Provincia de Buenos Aires Tel.: +54 11 4756 01 40 Fax: +54 11 4756 01 36 victor.jabif@boschrexroth.com.ar	NAKASE Servicio Tecnico CNC Calle 49, No. 5764/66 B1653AOX Villa Balester Provincia de Buenos Aires Tel.: +54 11 4768 36 43 Fax: +54 11 4768 24 13 nakase@usa.net nakase@nakase.com gerencia@nakase.com (Service)	Bosch Rexroth Ltda. Av. Tégula, 888 Ponte Alta, Atibaia SP CEP 12942-440 Tel.: +55 11 4414 56 92 +55 11 4414 56 84 Fax sales: +55 11 4414 57 07 Fax serv.: +55 11 4414 56 86 alexandre.wittwer@rexroth.com.br	Bosch Rexroth Ltda. R. Dr.Humberto Pinheiro Vieira, 100 Distrito Industrial [Caixa Postal 1273] 89220-390 Joinville - SC Tel./Fax: +55 47 473 58 33 Mobil: +55 47 9974 6645 prochnow@zaz.com.br
Columbia - Kolumbien			
Reflutec de Colombia Ltda. Calle 37 No. 22-31 Santafé de Bogotá, D.C. Colombia Tel.: +57 1 368 82 67 +57 1 368 02 59 Fax: +57 1 268 97 37 reflutec@neutel.com.co reflutec@007mundo.com			

Notes

293958

Printed in Germany

Rexroth
Indramat