ANILAM

6000i CNC Technical Manual

September 2008
Ve 04
627787-21 · 9/2008 · KS · Printed in USA · Subject to change without notice

www.anilam.com

Warranty

ANILAM® is a brand manufactured and sold by Acu-Rite Companies Inc. Acu-Rite Companies Inc. warrants its products to be free from defects in material and workmanship for one (1) year from date of installation. At our option, we will repair or replace any defective product upon prepaid return to our factory.

This warranty applies to all products when used in a normal industrial environment. Any unauthorized tampering, misuse or neglect will make this warranty null and void.

Under no circumstances will Acu-Rite Companies Inc., any affiliate, or related company assume any liability for loss of use or for any direct or consequential damages.

The foregoing warranties are in lieu of all other warranties expressed or implied, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose.

The information in this manual has been thoroughly reviewed and is believed to be accurate. ACU-RITE® Companies Inc. reserves the right to make changes to improve reliability, function, or design without notice. ACU-RITE Companies Inc. assumes no liability arising out of the application or use of the product described herein.

Section 1 - Introduction

General Information ..... **1 – 1**
System Overview ..... **1 – 1**
Product Designations ..... **1 – 2**
Meaning of the Symbols Used in this Manual ..... **1 – 2**
6000i Overview ..... **1 – 3**
Software Update Procedure ..... **1 – 4**

Section 2 - Mounting and Electrical Installation

General Information ..... **2 – 1**
Safety Precautions ..... 2 – 1
Degrees of Protection ..... 2 – 2
Electromagnetic Compatibility ..... 2 – 2
Handling the HDR Hard Disk and SIK ..... **2 – 3**
Shipping Brace of the HDR ..... 2 – 3
Installing/Removing the HDR and SIK ..... 2 – 4
Environmental Conditions ..... **2 – 5**
Heating and Cooling ..... 2 – 5
Humidity ..... 2 – 6
Mechanical Vibration ..... 2 – 6
Mounting Considerations ..... **2 – 7**
MC, CC, Inverter, and Amplifier Power Module ..... 2 – 7
Shipping Brace of the Hard Disk ..... 2 – 8
Installing and Removing the Hard Drive and SIK ..... 2 – 8
Display ..... 2 – 8
Connection Overview ..... **2 – 9**
Connecting MC 400 and CC 600 with Maximum Six Control Loops ..... 2 – 9
Cable and Basic Circuit Overview ..... 2 – 10
MC 400 and CC 600 Pinouts ..... **2 – 10**
Position Control for Encoders ..... 2 – 11
Encoders for Speed Control ..... 2 – 12
Touch Probe ..... 2 – 13
PWM Connection to Axis/Spindle Motors ..... 2 – 22
CNC Power Supply and Control Signals ..... 2 – 23
Control-Is-Ready Signal ..... 2 – 24
Power Supply for PLC Outputs ..... 2 – 24
Buffer Battery ..... 2 – 26
Analog Nominal Value Output ..... 2 – 27
Analog Input ..... 2 – 28
Switching Inputs 24 VDC (PLC) ..... 2 – 30
Switching Outputs 24 VDC (PLC) ..... 2 – 34
Flat Panel Display ..... 2 – 38
Manual Panel ..... 2 – 40
CNC Keyboard ..... 2 – 42
I/O Module Connection ..... 2 – 43
Data Interfaces ..... 2 – 46
USB Interface ..... 2 – 49
Drive Controller Enable ..... 2 – 50
PLC Input/Output Units ..... 2 – 51
I/O Module and I/O Expansion Base Module P/N Summary ..... **2 – 51**
Handwheel Input ..... **2 – 52**
RM 500 Remote Handwheel ..... 2 – 53
RM 300 Panel-Mounted Handwheel ..... 2 – 56

PM 350 Panel-Mounted Handwheel ..... 2 – 57
Console FP 6000i ..... 2 – 60
Manual Panel MP 6000M and MP 6001M ..... 2 – 61

Section 3 - Machine Parameters
General Information ..... 3 – 1
The Configuration Editor ..... 3 – 2
Calling the Configuration Editor ..... 3 – 3
Machine Parameter Screen ..... 3 – 5
Entering and Editing Machine Parameters ..... 3 – 7
Managing Configuration Files ..... 3 – 13
Sort File Content ..... 3 – 13
Table View ..... 3 – 14
Access Rights ..... 3 – 14
Update Rules ..... 3 – 16
Remove Syntax Error ..... 3 – 17
Reset Update Version ..... 3 – 17
Backup of Parameters ..... 3 – 18
Allocation of Configuration Data ..... 3 – 19
Setup of a Parameter File ..... 3 – 22
MP Subfiles ..... 3 – 23
Syntax of MP Subfile ..... 3 – 23
Activating MP Subfiles ..... 3 – 23
Displaying/Editing Data Records in the Configuration Editor ..... 3 – 26
MP Change List in the Configuration Editor ..... 3 – 28
MP Movement Monitoring ..... 3 – 29
MP Programming Station Mode ..... 3 – 29
Read or Change Machine Parameters via a PLC Module ..... 3 – 31
Overview of the Machine Parameters of the 6000i ..... 3 – 35
System ..... 3 – 35
Channels ..... 3 – 87
Axes ..... 3 – 104
KeySynonym ..... 3 – 131

Section 4 - Modules and PLC Operands
Overview of Modules ..... 4 – 1
Overview of the PLC Operands ..... 4 – 6
PLC Operands of the “General Data” Group ..... 4 – 6
PLC Operands of the “Operating Mode Group” Group ..... 4 – 8
PLC Operands of the “Machining Channels” Group ..... 4 – 9
PLC Operands of the “Axis” Group ..... 4 – 12
PLC Operands of the “Spindle” Group ..... 4 – 14

Section 5 - Configuring the Axes and Spindle
Machine Structure ..... 5 – 1
Adapting the Control to Machine Structure ..... 5 – 2
Definition of Axes ..... 5 – 3
Configuration of Machining Channels ..... 5 – 4
Configuring a Machining Channel ..... 5 – 5
Traversing the Reference Marks ..... 5 – 9

Moving to Restore Position ..... 5 – 10

Configuration of Axes ..... 5 – 11
Axis Designations and Coordinates ..... 5 – 11
Programmable Axes ..... 5 – 15
Physical Axes ..... 5 – 17
Virtual Axes ..... 5 – 21

Encoders ..... 5 – 23
Type of Encoder ..... 5 – 23
Distance-Coded Reference Marks ..... 5 – 27
Encoder Connections ..... 5 – 28
Defining the Traverse Direction ..... 5 – 31
Encoder Monitoring ..... 5 – 32

Analog Axes ..... 5 – 35

Reading Axis Information ..... 5 – 36

Traverse Ranges ..... 5 – 42

Lubrication Pulse ..... 5 – 43

Controlling Axes by PLC (PLC Axes) ..... 5 – 44
Stopping/Starting Axes by PLC ..... 5 – 44

Axis Error Compensation ..... 5 – 56
Backlash Compensation ..... 5 – 58
Linear Axis Error Compensation ..... 5 – 60
Nonlinear Axis Error Compensation ..... 5 – 62
Compensation of Thermal Expansion ..... 5 – 70

Machine Kinematics ..... 5 – 72
Configuration of the Machine Kinematics ..... 5 – 72
Definition of the Transformation with Vectors ..... 5 – 79

Reference Marks ..... 5 – 82
Definition ..... 5 – 82
Traversing the Reference Marks ..... 5 – 82
Defining the Process of Traversing the Reference Marks ..... 5 – 85
“Pass Over Reference Point” Mode of Operation ..... 5 – 92

The Control Loop ..... 5 – 95
Relation Between Jerk, Acceleration, Velocity, and Distance ..... 5 – 96
Geometry Filter ..... 5 – 98
Look-Ahead ..... 5 – 100
Interpolator ..... 5 – 107
Filter Before Position Control Loop ..... 5 – 108
Position Controller ..... 5 – 112
Activating and Deactivating Position Control Loops ..... 5 – 120
Feed-Rate Enable ..... 5 – 123
Controller Parameters for Manual Traverse ..... 5 – 124
Controller Parameters for Analog Axes ..... 5 – 125
Switching Parameter Blocks ..... 5 – 133

Monitoring Functions ..... 5 – 136
Monitoring the Drives ..... 5 – 136
Position Monitoring ..... 5 – 138

Movement Monitoring ..... 5 – 141
Standstill Monitoring ..... 5 – 142
Positioning Window ..... 5 – 143
Temperature Monitoring ..... 5 – 146
Read Actual Utilization of Drive Motors ..... 5 – 147
EMERGENCY STOP Monitoring ..... 5 – 149
Spindles ..... 5 – 151
Configuring Spindles ..... 5 – 151
Spindle Position Encoder ..... 5 – 152
Filtering the Acceleration Values ..... 5 – 153
Controlling the Spindle ..... 5 – 154
Stop Spindle at Trip Dog Position ..... 5 – 165
Spindle for Per-Revolution Feed ..... 5 – 166
Gear Shifting ..... 5 – 167
Tapping ..... 5 – 167
Integrated Oscilloscope ..... 5 – 168
Fundamentals ..... 5 – 168
Prepare Recording ..... 5 – 170
Record Signals ..... 5 – 174
Analyze Recording ..... 5 – 176
Saving and Loading Recordings ..... 5 – 179
Configure the Colors of the Oscilloscope Display ..... 5 – 181

Section 6 - Machine Integration
Display and Operation ..... 6 – 1
Position and Status Display ..... 6 – 1
Unit of Measurement for Display and Operation ..... 6 – 3
Decimal Separator ..... 6 – 4
Switching the Control On/Off ..... 6 – 5
Powering Up the Control ..... 6 – 5
Shutting Down the Control ..... 6 – 6
Conversational Language ..... 6 – 11
Control Operation in the Operating Mode Group ..... 6 – 14
Modes of Operation ..... 6 – 14
Control Operation in the Machining Channel ..... 6 – 16
Channel-Specific Settings ..... 6 – 16
NC Program Run ..... 6 – 17
Error Status ..... 6 – 25
Assignments in Manual Modes of Operation ..... 6 – 26
M Functions (M Strobe) ..... 6 – 27
Assigning M Functions to the Machining Channels ..... 6 – 27
Configuration of M Functions ..... 6 – 28
Overview of M Functions of the 6000i ..... 6 – 32
S Function (S Strobe) ..... 6 – 37
Assigning S Functions to the Machining Channels ..... 6 – 37
Configuration of S Function ..... 6 – 38

T Functions (T Strobe) ..... 6 – 41
Assigning T Functions to the Machining Channels ..... 6 – 41
Configuration of T Functions ..... 6 – 42

Alias Functions (Alias Strobe) ..... 6 – 45
Assigning Alias Functions to the Machining Channels ..... 6 – 45
Configuration of Alias Functions ..... 6 – 46

Error Messages and Log Files ..... 6 – 47
Error Window ..... 6 – 48
Error Log ..... 6 – 50
Keystroke Log File ..... 6 – 54
Saving Log Files ..... 6 – 55
PLC Error Messages ..... 6 – 56
Structure of the Error Text File ..... 6 – 58

Keystroke Simulation ..... 6 – 65
Control Keyboard ..... 6 – 65
Machine Operating Panel ..... 6 – 70

Electronic Handwheel ..... 6 – 71
General Handwheel Parameters ..... 6 – 71
Serial Handwheel ..... 6 – 73
Handwheel at Position Encoder Input ..... 6 – 76
Traverse Per Handwheel Revolution ..... 6 – 80
Assigning a Handwheel to an Axis ..... 6 – 81

Override ..... 6 – 86
Override Devices ..... 6 – 86
Compensation for Potentiometers ..... 6 – 88
Override Functions ..... 6 – 89

PLC Inputs/Outputs ..... 6 – 94
Diagnosis of the Programmable Logic (PL) ..... 6 – 95
24 VDC Switching Input/Outputs ..... 6 – 99
Analog Inputs ..... 6 – 101
Analog Outputs ..... 6 – 103

Incremental Jog Positioning ..... 6 – 104

Operating Times and System Times ..... 6 – 107
Measuring Operating Times ..... 6 – 107
System Time ..... 6 – 113

Tool Changer ..... 6 – 115
Tool and Pocket Number ..... 6 – 115

Commissioning ..... 6 – 126
Preparation ..... 6 – 126
Adjusting the Servo Amplifier ..... 6 – 128
Commissioning the Axes ..... 6 – 129

Diagnosis with the On-Line Monitor (OLM) ..... 6 – 145
Introduction ..... 6 – 145
Operation of the OLM ..... 6 – 147
Screen Layout ..... 6 – 151
Selecting Axes and Channels ..... 6 – 153
Group of NC Axes ..... 6 – 154

Group of Spindle Commands .....6 – 166
Group of NC Channels .....6 – 168
Hardware Group .....6 – 173
Auxiliary Group .....6 – 178
PLC Group .....6 – 183
Queue Trace .....6 – 185
END Soft Key .....6 – 187
Frequent Causes of Error .....6 – 187

Section 7- PLC Programming

PLC Functions..... 7 – 2
The Symbolic PLC-API (New Programming Interface) ..... 7 – 3
ANILAM PLC Basic Program ..... 7 – 6
Selecting the PLC Mode ..... 7 – 6
PLC Main Menu ..... 7 – 7
The API DATA Function ..... 7 – 9
The Watch List Function ..... 7 – 10
The Table Function ..... 7 – 12
The Compile Function ..... 7 – 17
The Edit Function ..... 7 – 18

Operands ..... 7 – 19
Operand Overview ..... 7 – 20
Timers ..... 7 – 23
Counter ..... 7 – 28
Fast PLC Inputs ..... 7 – 30

Data Organization ..... 7 – 32
PLC System Files ..... 7 – 32

Tables..... 7 – 36
Creating a New Table Type ..... 7 – 37
Creating a New Table with File Manager ..... 7 – 45
Inserting Additional Columns in an Existing Table ..... 7 – 46
Deleting Columns from an Existing Table ..... 7 – 47
Removing Column Names and Column Descriptions ..... 7 – 47
Symbolic Names for Tables ..... 7 – 48
Editing Tables Via the PLC ..... 7 – 49
Access to Tables Via SQL Commands ..... 7 – 60
Reference for Syntax Elements ..... 7 – 63
PLC Modules for the SQL Statements ..... 7 – 74

Data Transfer NC –> PLC, PLC –> NC..... 7 – 94
Data Transfer of NC Program –> PLC (“FN19: PLC =” or “FN29: PLC =”) ..... 7 – 95
Data Transfer of NC Program –> PLC (FN17: SYSWRITE) ..... 7 – 97
Data Transfer NC –> NC Program (FN18: SYSREAD) ..... 7 – 105
Data Transfer Machine Parameters –> PLC ..... 7 – 117
Interrogate PLC Operands in the NC Program (FN20: WAIT FOR) ..... 7 – 118

Program Creation..... 7 – 119
ASCII Editor ..... 7 – 119
Program Structure ..... 7 – 120

PLC Commands ..... 7 – 121
Overview ..... 7 – 123
LOAD (L) ..... 7 – 126
LOAD NOT (LN) ..... 7 – 128
LOAD TWO'S COMPLEMENT (L–) ..... 7 – 130
LOAD BYTE (LB) ..... 7 – 131
LOAD WORD (LW) ..... 7 – 132
LOAD DOUBLE WORD (LD) ..... 7 – 132
ASSIGN (=) ..... 7 – 133
ASSIGN BYTE (B=) ..... 7 – 135
ASSIGN WORD (W=) ..... 7 – 135
ASSIGN DOUBLE WORD (D=) ..... 7 – 136
ASSIGN NOT (=N) ..... 7 – 136
ASSIGN TWO'S COMPLEMENT (=–) ..... 7 – 136
SET (S) ..... 7 – 137
RESET (R) ..... 7 – 138
SET NOT (SN) ..... 7 – 139
RESET NOT (RN) ..... 7 – 140
AND (A) ..... 7 – 141
AND NOT (AN) ..... 7 – 143
OR (O) ..... 7 – 145
OR NOT (ON) ..... 7 – 147
EXCLUSIVE OR (XO) ..... 7 – 149
EXCLUSIVE OR NOT (XON) ..... 7 – 151
ADDITION (+) ..... 7 – 153
SUBTRACTION (–) ..... 7 – 154
MULTIPLICATION (X) ..... 7 – 155
DIVISION (/) ..... 7 – 156
REMAINDER (MOD) ..... 7 – 157
INCREMENT (INC) ..... 7 – 158
DECREMENT (DEC) ..... 7 – 158
EQUAL TO (==) ..... 7 – 159
LESS THAN (<) ..... 7 – 160
GREATER THAN (>) ..... 7 – 161
LESS THAN OR EQUAL TO (<=) ..... 7 – 162
GREATER THAN OR EQUAL TO (>=) ..... 7 – 163
NOT EQUAL (<>) ..... 7 – 164
AND [ ] (A[ ]) ..... 7 – 165
AND NOT [ ] (AN[ ]) ..... 7 – 166
OR [ ] (O[ ]) ..... 7 – 166
OR NOT [ ] (ON[ ]) ..... 7 – 166
EXCLUSIVE OR [ ] (XO[ ]) ..... 7 – 166
EXCLUSIVE OR NOT [ ] (XON[ ]) ..... 7 – 166
ADDITION [ ] (+[ ]) ..... 7 – 167
SUBTRACTION [ ] (–[ ]) ..... 7 – 168
MULTIPLICATION [ ] (X[ ]) ..... 7 – 168
DIVISION [ ] (/[ ]) ..... 7 – 168
REMAINDER [ ] (MOD[ ]) ..... 7 – 168
EQUAL TO [ ] (==[ ]) ..... 7 – 169
LESS THAN [ ] (<[ ]) ..... 7 – 170

GREATER THAN [ ] (>[ ]) ..... 7 – 170
GREATER THAN OR EQUAL TO [ ] (>=[ ]) ..... 7 – 170
NOT EQUAL [ ] (<>[ ]) ..... 7 – 170
SHIFT LEFT (<<) ..... 7 – 171
SHIFT RIGHT (>>) ..... 7 – 172
BIT SET (BS) ..... 7 – 173
BIT CLEAR (BC) ..... 7 – 174
BIT TEST (BT) ..... 7 – 175
Push Data onto the Data Stack (PS) ..... 7 – 176
Pull Data from the Data Stack (PL) ..... 7 – 177
Push LOGIC ACCUMULATOR onto the Data Stack (PSL) ..... 7 – 178
Push WORD ACCUMULATOR onto the Data Stack (PSW) ..... 7 – 178
Pull LOGIC ACCUMULATOR from the Data Stack (PLL) ..... 7 – 179
Pull WORD ACCUMULATOR from the Data Stack (PLW) ..... 7 – 179
UNCONDITIONAL JUMP (JP) ..... 7 – 179
JUMP IF LOGIC ACCUMULATOR = 1 (JPT) ..... 7 – 180
JUMP IF LOGIC ACCUMULATOR = 0 (JPF) ..... 7 – 180
CALL MODULE (CM) ..... 7 – 181
CALL MODULE IF LOGIC ACCUMULATOR = 1 (CMT) ..... 7 – 181
CALL MODULE IF LOGIC ACCUMULATOR = 0 (CMF) ..... 7 – 182
END OF MODULE, END OF PROGRAM (EM) ..... 7 – 183
END OF MODULE IF LOGIC ACCUMULATOR = 1 (EMT) ..... 7 – 183
END OF MODULE IF LOGIC ACCUMULATOR = 0 (EMF) ..... 7 – 183
LABEL (LBL) ..... 7 – 183

INDEX Register (X Register) ..... 7 – 184

Commands for String Processing ..... 7 – 186
LOAD String (L) ..... 7 – 187
ADD String (+) ..... 7 – 188
STORE a String (=) ..... 7 – 188
OVERWRITE a String (OVWR) ..... 7 – 189
EQUAL TO Command for String Processing (==) ..... 7 – 190
LESS THAN Command for String Processing (<) ..... 7 – 190
GREATER THAN Command for String Processing (>) ..... 7 – 190
LESS THAN OR EQUAL TO Command for String Processing (<=) ..... 7 – 191
GREATER THAN OR EQUAL TO Command for String Processing (>=) ..... 7 – 192
NOT EQUAL Command for String Processing (<>) ..... 7 – 192
Modules for String Processing ..... 7 – 193

Submit Programs ..... 7 – 197
Calling the Submit Program (SUBM) ..... 7 – 198
Interrogating the Status of a Submit Program (RPLY) ..... 7 – 198
Canceling a Submit Program (CAN) ..... 7 – 199

Cooperative Multitasking ..... 7 – 201
Starting a Parallel Process (SPAWN) ..... 7 – 201
Control of Events ..... 7 – 202

Constants Field (KF) ..... 7 – 207

Program Structures ..... 7 – 208
IF ... ELSE ... ENDI Structure ..... 7 – 209
REPEAT ... UNTIL Structure ..... 7 – 209

WHILE ... ENDW Structure ..... 7 – 210
Case Branch ..... 7 – 211

Linking Files ..... 7 – 212
USES Statement (USES) ..... 7 – 213
GLOBAL Statement (GLOBAL) ..... 7 – 214
EXTERN Statement (EXTERN) ..... 7 – 214

PLC Modules ..... 7 – 215
Markers, Bytes, Words, and Double Words ..... 7 – 215
Number Conversion ..... 7 – 253

Section 8 - Data Interfaces

Introduction ..... 8 – 1

The Ethernet Interface ..... 8 – 2

The USB Interface of the Control (USB 1.1) ..... 8 – 3

The Serial Interface of the Control ..... 8 – 6
RS-232-C/V.24 Interface ..... 8 – 6

Configuring the Serial Interface ..... 8 – 9
Control Characters ..... 8 – 9
Configuration of Interfaces ..... 8 – 10

Data Transfer by PLC ..... 8 – 19
PLC Modules ..... 8 – 19

Section 9 - Drawings

Drawings Listed ..... 9 – 1
Figure 9-1, Console ..... 9 – 2
Figure 9-2, MC, CC, and Inverter ..... 9 – 3
Figure 9-3, MP 6000M Manual Panel ..... 9 – 4
Figure 9-4, MP 6001M Manual Panel ..... 9 – 5
Figure 9-5, CC 600 and MC 400 ..... 9 – 6
Figure 9-6, CC 600 and MC 400 Dimensions ..... 9 – 7
Figure 9-7, I/O EXP BASE 4-SLOTS (P/N 624498-01, iIEB 404), 6-SLOTS (P/N 62500-01, IEB 406), 8-SLOTS (P/N 624501-01, IEB 408) ..... 9 – 8
Figure 9-8, I/O EXP BASE 4-SLOTS (P/N 624498-01, iIEB 404) Connector Description ..... 9 – 9
Figure 9-9, Expansion Base Grounding ..... 9 – 10
Figure 9-10, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) Dimensions . 9 – 11
Figure 9-11, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) LEDs and Connectors ..... 9 – 12
Figure 9-12, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Dimensions .... 9 – 13
Figure 9-13, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Connectors ..... 9 – 14
Figure 9-14, Hard Disk Drawer (P/N 574746-51, HDR) Dimensions ..... 9 – 15
Figure 9-15, Hard Disk Drawer (P/N 574746-51, HDR) Minimum Clearances ..... 9 – 16
Figure 9-16, Hard Disk Drawer (P/N 574746-51, HDR) Locking/Unlocking the Drive .. 9 – 17
Figure 9-17, System ID Key (P/N 574744-51, SIK) Installation ..... 9 – 18
Figure 9-18, USB Hub (P/N 624508-01) Dimensions ..... 9 – 19
Figure 9-19, Basic Servo Turn On Circuit ..... 9 – 20
Figure 9-20, RM 500 Remote Handwheel, P/N 34000850 ..... 9 – 23

Figure 9-21, PM 300 Panel-Mounted Handwheel, P/N 34000855 ..... 9 – 24
Figure 9-22, Cable Overview ..... 9 – 25
Figure 9-23, Cable Overview, Modular ..... 9 – 27
Figure 9-24, Basic System Diagram ..... 9 – 29

Index Index-1

Section 1 - Introduction

The following topics are described in this section:

- **General Information**
- **System Overview**
- **Product Designations**
- **Meaning of the Symbols Used in this Manual**
- **6000i Overview**
- **Software Update Procedure**

General Information

This manual was written for machine tool manufacturers. It contains information required to install and connect the 6000i Computer Numerical Control (CNC) and components, which include:

- Main computing (MC) unit
- Digital current controller (CC)
- ANILAM inverter
- Axis and spindle motors
- APM 100A power supply
- I/O module(s)
- Operating panel
- Console
- Accessories and cables

System Overview

The CNC is designed to be used with ANILAM compact and modular inverters. They feature PC chipsets, hard disks, and external Pulse Width Modulation (PWM) connections.

Keyboards feature machine operating panels, feedrate override, and spindle override.

Product Designations

Refer to **Table 1-1**.

Table 1-1, Product Designations

Model Number	Component
6000i	This designation is used when the control is considered as a whole (including accessories, such as machine operating panel, handwheels and touch probes)
MC 400	Stand-alone logic unit
CC 600	Digital current controller
FP 6000i	6000i flat panel displays
MP 600*X*M	Manual panels with (MP 6001M) and without handwheel (MP 6000M)
IEB 404	P/N 624498-01, Exp base module, 4-slots
IEB 406	P/N 624500-01, Exp base module, 6-slots
IEB 408	P/N 624501-01, Exp base module, 8-slots
IEM 16-8D	P/N 624505-01, I/O module, digital 16 inputs/8 outputs
IEM 4-4A	P/N 624506-01, I/O module, analog 4 inputs/4 outputs
	P/N 624507-01, I/O module, blank
APM 100A	Power module
AM *XXX*	Axis (synchronous) motor
SM *XXX*	Spindle (asynchronous) motor)
SA *XXX*	Spindle/Axis amplifier

Meaning of the Symbols Used in this Manual

Danger: Failure to comply with this information could result in most serious up to fatal injuries or in substantial material damage.

Warning: Failure to comply with this information could result in injuries and interruptions of operation up to material damage.

Note: Tips and tricks for operation as well as important information, for example about standards and regulations as well as for better understanding of the document.

6000i Overview

The core of the 6000i is the MC 400. The MC 400 supports the standard 12.1” LCD. The 6000i uses a modular MC and Current Controller (CC). The CC for the 6000i is CC 600. In addition to the MC 400 and CC 600, the 6000i will also use:

- A Hard Disk dRawer (i.e.,HDR). The hard disk drawer makes servicing the MC (standalone logic unit) easier. The hard disk drawer can be pulled out and put into the new MC. All parameters, PLC programs, and user programs are then available in the new MC.
- A System Identification Key (i.e., SIK).

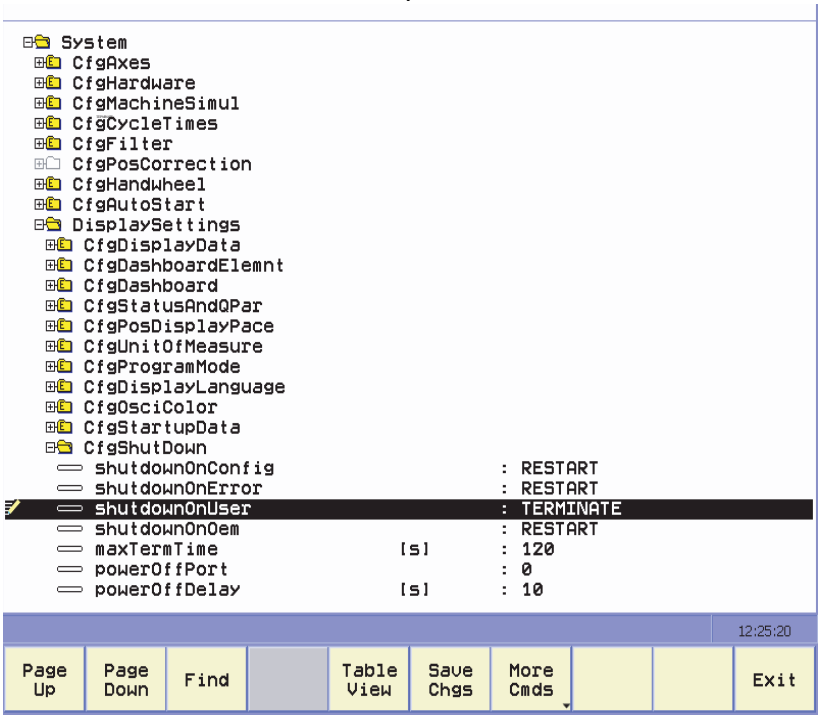
The hard-disk and SIK are assembled into the MC by ANILAM. The hard disk and SIK would normally be accessed by the user only for service reasons.

P/N	Designation	Description
574774-01	MC 400	Main Computer
574746-51	HDR	Hard Disk Drawer
574744-51	SIK	System Identification Key
624513-01	CC 600	Current Controller

Software Update Procedure

To do a software update; you will place the update file “setup.zip” file on a Universal Serial Bus (USB) stick (the setup.zip file is approximately 120 MB and you will need a USB memory stick with at least 500 MB of free disk space).

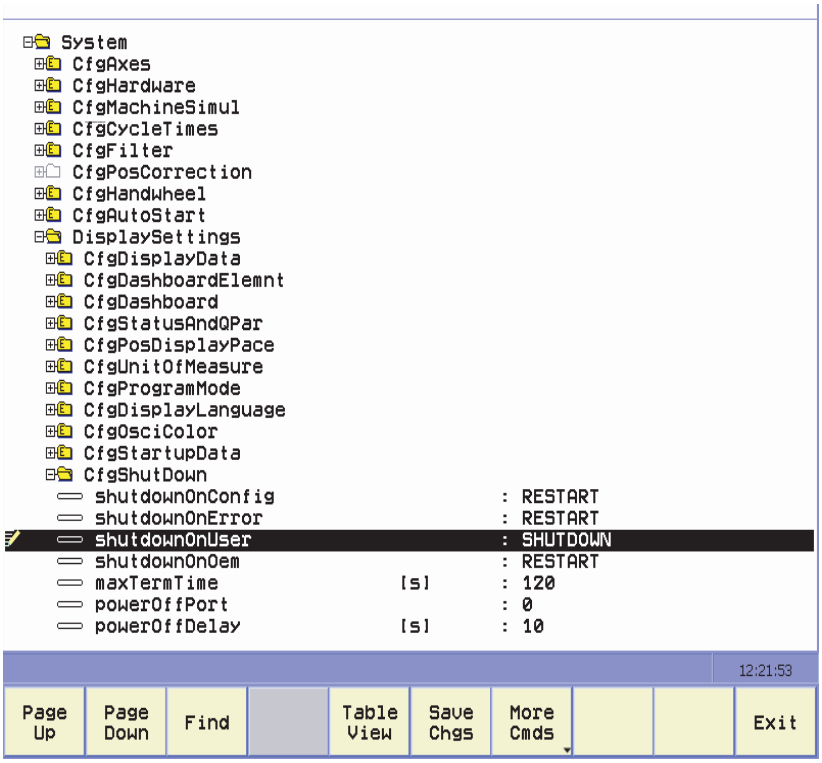
- With the control up and running and the Estop pressed in, plug the USB stick into the control.
- Press **Config** (**SHIFT** + **F3**) and the control displays a prompt for a password. Press **ENTER**.
- Navigate to System > DisplaySettings > CfgShutDown > shutdownOnUser > and select “Terminate” from the drop down menu:



- Press **Exit** (**F10**) and then **F1** to save changes.
- Press **ShutDown** (**SHIFT** + **F10**) and press **ShutDown** (**F1**) to exit the software.
- Once you are out of the 6000i software and at a black screen, type MENU and press **ENTER**.
- The 6000i menu displays on the screen. From the menu arrow down to the 8th item (Update) and press **ENTER**.
- Another menu is displayed, select the first item (Source: USB stick) and press **ENTER**.
- The control will look for the setup.zip file on the USB stick and prompt to press “1” to begin the Update.
- Press “1” and press **ENTER** to start the update.
- The update may take 20 minutes to load. The screen may go into screen saver mode. If the screen blanks out, press any key to watch the update progress.
- When the update is complete, you are prompted to remove the USB stick and press **ENTER** to reboot the control.
- Once the control reboots and comes up, home the machine, then reset the machine parameter shutdownOnUser to SHUTDOWN.

- If they are not already set, set shutdownOnConfig, shutdownOnError, shutdownOnOem, maxTermTime, powerOffPort, and powerOffDelay to the values below.

shutdownOnConfig	RESTART
shutdownOnError	RESTART
shurdownOnUser	SHUTDOWN
shutdownOnOem	RESTART
maxTermTime	120
powerOffPort	0
powerOffDelay	10



- Press **Exit** (**F10**) and then **F1** to save changes; you are ready to run.

Section 2 - Mounting and Electrical Installation

The following topics are described in this section:

- **General Information**
- **Handling the HDR Hard Disk and SIK**
- **Environmental Conditions**
- **Mounting Considerations**
- **Connection Overview**
- **MC 400 and CC 600 Pinouts**
- **I/O Module and I/O Expansion Base Module P/N Summary**
- **Handwheel Input**
- **Console FP 6000i**
- **Manual Panel MP 6000M and MP 6001M**

General Information

Warning: Keep the following in mind during mounting and electrical installation:

- National regulations for power installations
- Interference and noise immunity
- Conditions of operation
- Mounting attitude

The following topics are described:

- **Safety Precautions**
- **Degrees of Protection**
- **Electromagnetic Compatibility**

Safety Precautions

Danger: Ensure that the main switch of the control or machine is switched off when you engage or disengage connecting elements or connection clamps.

Danger: Ensure that the equipment grounding conductor is continuous. Interruptions in the equipment grounding conductor may cause damage to persons or property.

Danger: Incorrect or not optimized input values may lead to malfunction of the machine and may thus cause damage to persons or property. Modifications of the machine configuration should be done with caution and uncontrolled axis motions should be taken into account.

Warning: In order to be able to judge the behavior of an NC controlled machine, you need to have fundamental knowledge about drives, inverters, controls and encoders. Inappropriate use may cause considerable damage to persons or property.
ANILAM does not accept any responsibility for direct or indirect damage caused to persons or property through incorrect use or operation of the machine.

Danger: The interfaces for the PLC inputs/outputs, machine operating panel, and PL connection comply with the requirements for basic insulation in accordance with **IEC 742 EN 50 178**.
Only units that comply with the requirements of **IEC 742 EN 50 178** for basic insulation may be connected; otherwise, damage to persons or property may be caused. The maximum DC voltage mean value of the PLC inputs is 31 V.

Degrees of Protection

The following components fulfill the requirements for IP54 (dust and splash-proof protection).

- MC 400 (when properly installed)
- Machine operating panel (when properly installed)
- Handwheel

Electromagnetic Compatibility

This unit fulfills the requirements for Class A according to EN 55022 and is intended for operation in industrially zoned areas.

Protect your equipment from interference by observing the following rules and recommendations.

The following topics are described:

- **Likely Sources of Interference**
- **Protective Measures**

Likely Sources of Interference

Interference is mainly produced by capacitive and inductive coupling from electrical conductors or from device inputs/outputs, such as:

- Strong magnetic fields from transformers or electric motors
- Relays, contactors, and solenoid valves
- High-frequency equipment, pulse equipment, and stray magnetic fields from switch-mode power supplies
- Power lines and leads to the above equipment

Protective Measures

- Keep a minimum distance of 20 cm from the control and its leads to interfering equipment.
- A minimum distance of 10 cm from the control and its leads to cables that carry interference signals. For cables in metallic ducting, adequate decoupling can be achieved by using a grounded separation shield.
- Shielding according to EN 50 178
- Use potential compensating lines with 6 mm^2 cross-sections
- Use only genuine ANILAM cables, connectors, and couplings

Handling the HDR Hard Disk and the SIK

The following topics are described:

- **Shipping Brace of the HDR**
- **Installing/Removing the HDR and SIK**

Shipping Brace of the HDR

The HDR hard disks of the MC 400 are fitted with a shipping brace. Before putting the 6000i into service, the shipping brace of the hard disk must be removed.

Warning: Do not transport the HDR with the MC 400 after you have installed the HDR. The shipping brace for the hard disk is not required when the machine is being transported.

Should servicing become necessary (i.e. the HDR is being shipped on its own), the hard disk must be secured with the shipping brace. Refer to **Figure 2-1**.

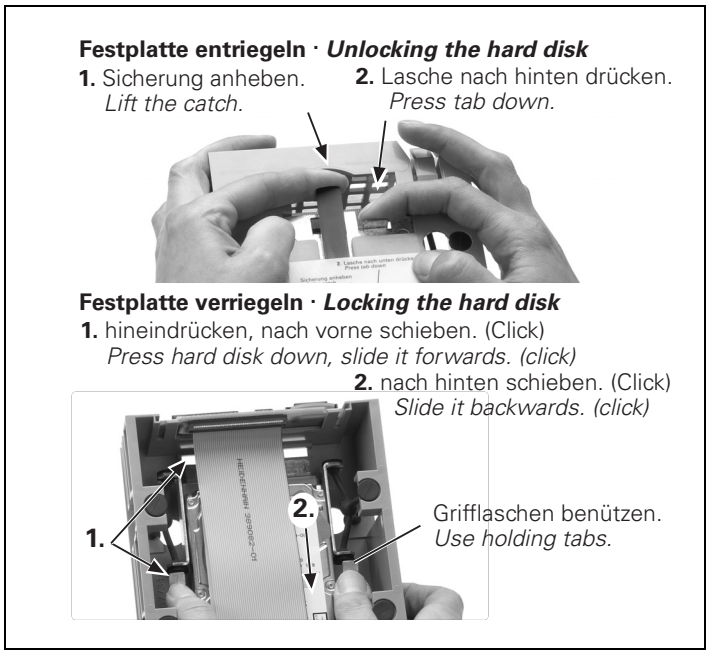
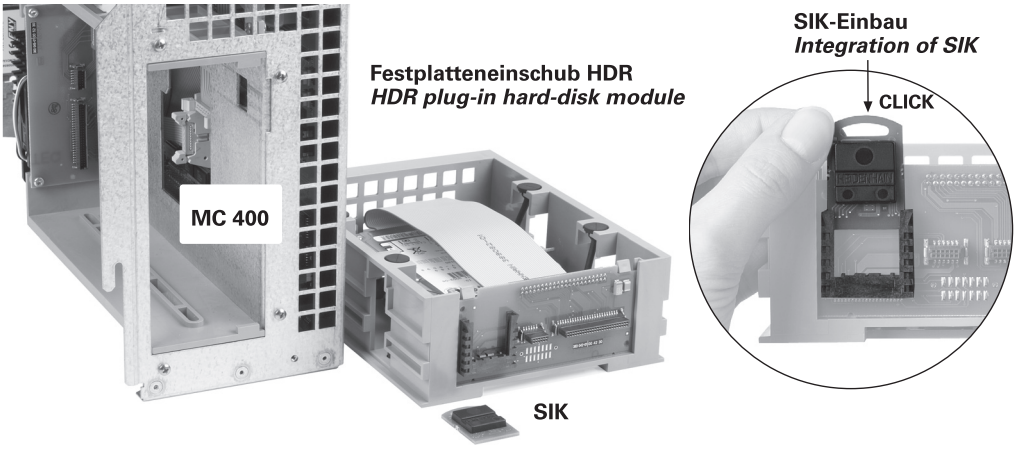


Figure 2-1, Unlocking/Locking the Hard Drive

Installing/Removing the HDR and SIK

Refer to **Figure 2-2**.



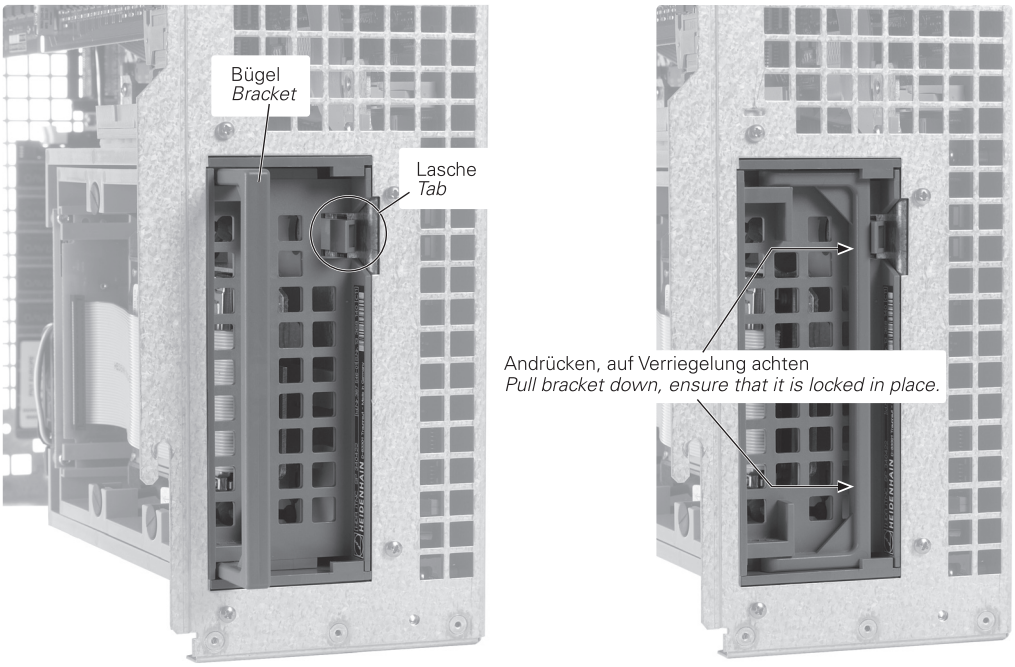
Achtung: Im Servicefall Festplatte verriegeln, SIK entnehmen und aufbewahren.
Note: *Should servicing become necessary, lock the hard disk, remove the SIK and keep it in a safe place.*

HDR-Einbau · *Installation of HDR*

1. SIK einbauen
Integrate SIK.

2. HDR einschieben, auf Verriegelung achten. (Click)
Insert HDR. (click)

3. Bügel umlegen und gegen Lasche andrücken. (Click)
Turn the bracket over and press it against the tab until it "clicks" in place.



HDR-Ausbau in umgekehrter Reihenfolge · *To remove the HDR, proceed in reverse order*

1. Bügel entriegeln (Click) und aufstellen.
Unlock the bracket so that it "unclicks" and pull it up.

2. Lasche drücken und HDR rausziehen.
Press the tab and pull out the HDR.

3. SIK entnehmen.
Remove the SIK.

Figure 2-2, Unlocking/Locking the Hard Drive

Environmental Conditions

The following topics are described:

- **Heating and Cooling**
- **Humidity**
- **Mechanical Vibration**

Heating and Cooling

Danger: The permissible ambient temperature in operation is between 0 °C and 40 °C (32 °F to 104 °F). Any deviation from this will impair the operating safety of the machine.

Ensure adequate cooling as follows:

- Provide sufficient space for air circulation.
- Install in a fan to extract warm air. Do not allow pre-warmed air to be blown into the unit. The warmed air should flow over surfaces such as sheet metal, which enable heat dissipation.
- Where the chassis is a closed steel housing without assisted cooling, the formula for heat conduction is 3 W/m2 of surface per °C air temperature difference between inside and outside.
- Use a heat exchanger with separate internal and external circulation.
- Do not blow external air through the control cabinet to exchange the internal air. Fine dust or vapors could damage electronic assemblies. If no other method of cooling is possible, ensure that the fan draws warm air out of the electrical cabinet and pulls in air that is adequately filtered. Service the filter regularly.

A heat exchanger or a cooling unit is preferable for controlling the internal temperature of the electrical cabinet. Refer to **Figure 2-3, Correct Positioning**.

If filtered air is blown into the electrical cabinet for cooling purposes, the standard EN 50 178 applies, which permits contamination level 2.

Danger: Be sure to take the measures required for preventing dust from entering the electrical cabinet.
Dust depositing inside electrical devices may cause them to fail and impair the safety of the system.

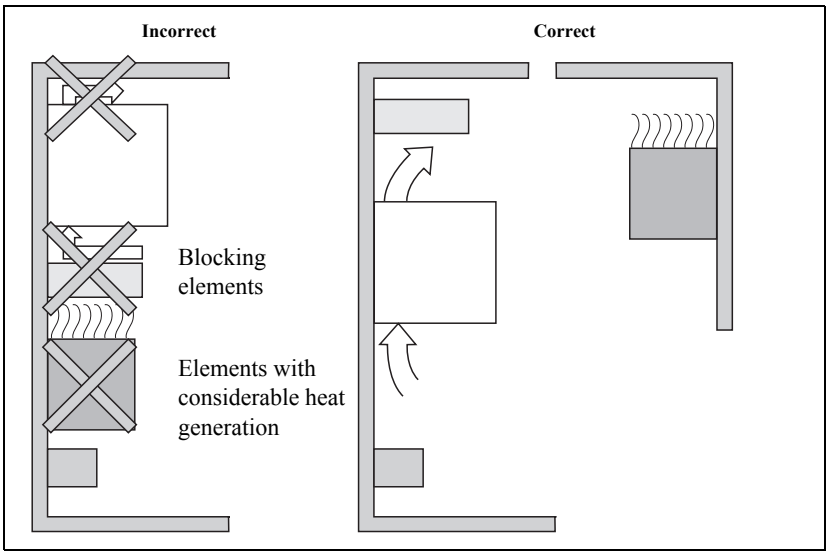


Figure 2-3, Correct Positioning

Humidity

Permissible humidity:

- Maximum 75% in continuous operation
- Maximum 95% for not more than 30 days a year (randomly distributed)

In tropical areas it is recommended that the control not be switched off, so that condensation is avoided on the circuit boards.

Mechanical Vibration

Permissible vibration:	± 0.075 mm, 10 to 41 Hz
	5 m/s^2, 41 Hz to 500 Hz
Permissible shock:	100 m/s^2, 11 ms during operation
	300 m/s^2, 11 ms during transport (with shipping brace for hard disk)

Mounting Considerations

The following topics are described:

- **MC, CC, Inverter, and Amplifier Power Module**
- **Shipping Brace of the Hard Drive**
- **Installing and Removing the Hard Drive and SIK**
- **Display**

MC, CC, Inverter, and Amplifier Power Module

Refer to **Figure 2-4**.

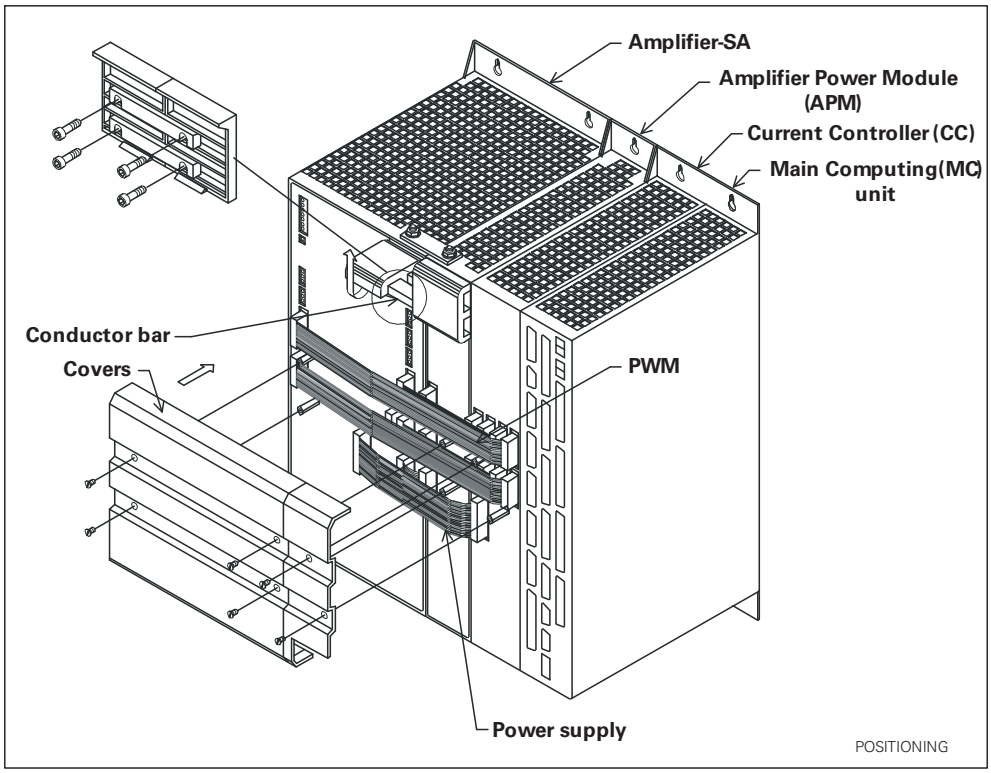


Figure 2-4, Positioning the MC, CC, Inverter, and Power Module

Note: Refer to "**Section 9 - Drawings**" for dimensions and clearances.

Shipping Brace of the Hard Drive

The 6000i hard drive is fitted with a shipping brace. Before putting the 6000i into service, the shipping brace must be removed.

Warning: Do not transport the Hard Drive with the 6000i after you have installed the Hard Drive. The shipping brace for the Hard Drive is not required when the 6000i is being transported.

Should servicing become necessary (that is, the Hard Drive being shipped on its own), the hard disk must be secured with the shipping brace.

To unlock the Hard Drive:

1. Lift the catch
2. Press down tab

To lock the Hard Drive:

1. Press the hard disk down, slide it forwards (click)
2. Using the holding tabs, slide it backwards (click)

Installing and Removing the Hard Drive and SIK

Should servicing the Hard Drive become necessary, lock the Hard Drive, remove the System ID Key (SIK) and keep it in a safe place.

To Install the Hard Drive:

1. Integrate the SIK
2. Insert the Hard Drive
3. Turn the bracket over and press it against the tab until it “clicks” in place

To remove the Hard Drive:

1. Unlock the bracket so that it “unclicks” and pull it up
2. Press the tab and pull out the Hard Drive
3. Remove the SIK

Display

For space requirements, refer to **Figure 9-1, Console**.

Note: The display is sensitive to electromagnetic or magnetic noise. Strong fields can lead to slight distortions of the picture. Ensure a minimum clearance of 0.5 m (1.64 ft).

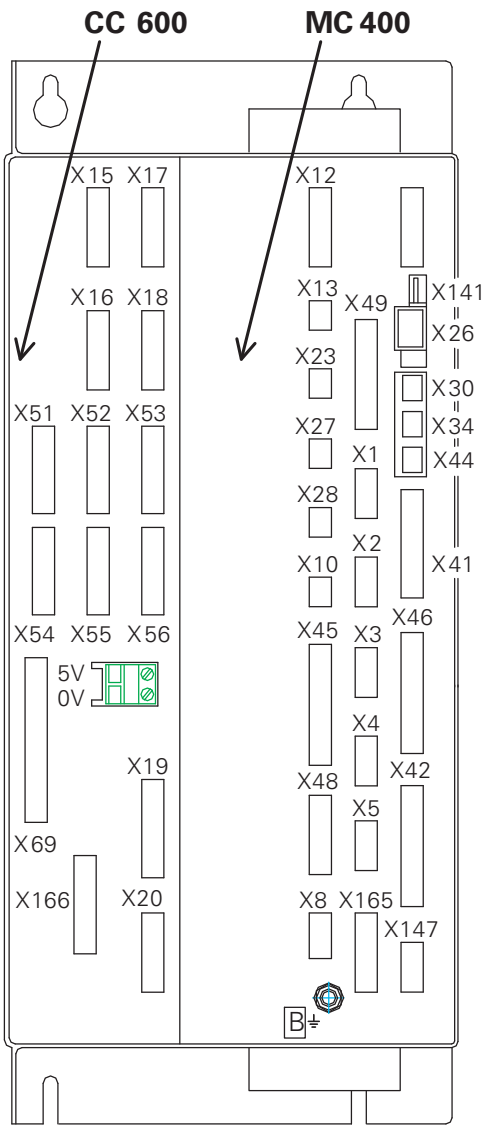
Connection Overview

The following topics are described:

- **Connecting MC 400 and CC 600 with Maximum Six Control Loops**
- **Cable and Basic Circuit Overview**

Connecting MC 400 and CC 600 with Maximum Six Control Loops

Refer to **Figure 2-5**



Conctr.	Description
X1–X5	Axes position encoder 1 Vpp
X15–X20	Axis speed encoder 1 Vpp
X51–X56	PWM output
X8	Nominal value output, analog
X12	Touch probe for workpiece measurement
X13	Touch probe for tool measurement
X23	Handwheel input
X26	Ethernet data interface
X27	COM1 data interface (RS-232)
X28	COM2 data interface (RS-422)
X141, X142	USB interface
X30	Do not use
X34	24 V for control-is-ready output (X41/34)
X41	PLC output
X42	PLC input
X44	24 V PLC power supply
X45	Keyboard
X46	Manual panel
X147	PLC Expansion
X48	PLC analog input
X49	Flat panel display
X69	Power supply
X10	Reserved
X165, X166	Reserved
X150	Axis-specific drive release
B	Signal ground
⏚	Equipment ground (YL/GN)

Figure 2-5, MC 400 and CC 600 Connections

Warning: Do not engage or disengage any connecting elements while the unit is under power.

Cable and Basic Circuit Overview

For a cable overview, refer to **Figure 9-22, Cable Overview** for systems with Spindle Axis (SA xxxx) amplifiers and **Figure 9-23, Cable Overview, Modular** for systems with modular spindle and axis amplifiers.

For basic circuit diagrams, refer to **Figure 9-24, Basic System Diagram** and **Figure 9-19, Basic Servo Turn On Circuit**.

MC 400 and CC 600 Pinouts

Refer to **Figure 2-5, MC 400 and CC 600 Connections**, **Figure 9-5, CC 600 & MC 400**, and **Figure 9-6, CC 600 & MC 400 Dimensions**.

The following topics are described:

- **Position Control for Encoders**
- **Encoders for Speed Control**
- **Touch Probe**
- **PWM Connection to Axis/Spindle Motors**
- **CNC Power Supply and Control Signals**
- **Control-Is-Ready Signal**
- **Power Supply for PLC Outputs**
- **Buffer Battery**
- **Analog Nominal Value Output**
- **Analog Input**
- **Switching Inputs 24 VDC (PLC)**
- **Switching Outputs 24 VDC (PLC)**
- **Flat Panel Display**
- **Manual Panel**
- **CNC Keyboard**
- **I/O Module Connection**
- **Data Interfaces**
- **USB Interface**
- **Drive Controller Enable**
- **PLC Input/Output Units**

Position Control for Encoders

X1–**X4** are linear encoder axis position connections. **X5** is the spindle encoder position connection. ANILAM CNCs are designed to be used with linear encoders or rotary encoders for position control. Refer to **Table 2-1** for linear encoder pinouts.

ANILAM recommends using linear encoders with distance-coded reference marks or with EnDat interface. These encoders reduce significantly the travel distance required to establish absolute position.

Maximum input frequency: 350 kHz.

Table 2-1, X1–X-5: Encoder Position Control Connections (1 V_{PP} amplitude)

Chassis		Encoder Cable	
D-Sub Connection (Male) 15-Pin	**Assignment**	**D-Sub Connector (Female) 15-Pin**	**Color**
1	+5 V (U_P)	1	Brown/Green
2	0 V (U_N)	2	White/Green
3	A+	3	Brown
4	A–	4	Green
5	0 V	5	
6	B+	6	Gray
7	B–	7	Pink
8	0 V	8	
9	+5 V	9	Blue
10	R+	10	Red
11	0 V	11	White
12	R–	12	Black
13	0 V	13	
14	Do not assign	14	Violet
15	Do not assign	15	
Housing	External shield	Housing	External shield

The encoder signals are interpolated 1024-fold.

Encoders for Speed Control

Maximum input frequency: 350 kHz. The speed encoder can also be used for position control. **X15**–**X18** and **X20** are rotary encoder axis speed control connections. **X19** is the spindle encoder speed control connection.

X15–X20 Rotary Encoder (1 V_{PP} amplitude)

Refer to **Table 2-2**.

Table 2-2, X15–X20: Encoder Speed Control Connections - Pinout

MC 400			Adapter Cable		
D-Sub Connctn. (Male) 25-Pin	Assignment		D-Sub Connctr. (Female) 25-Pin	Color(s)	Connctr. (Female) 17-Pin
1	+ 5 V (U_P)		1	Brown/Green	10
2	0 V (U_N)		2	White/Green	7
3	A+		3	Green/Black	1
4	A–		4	Yellow/Black	2
5	0 V		5		
6	B+		6	Blue/Black	11
7	B–		7	Red/Black	12
8	0 V		8	Internal shield	17
9	Not assigned		9		
10	0 V		10		
11	Not assigned		11		
12	Not assigned		12		
13	Temperature +		13	Yellow	8
14	+5 V or not assigned		14	Blue	16
15	Analog output (test)		15		
16	0 V		16	White	15
17	R+		17	Red	3
18	R–		18	Black	13
19	C+		19	Green	5
20	C–		20	Brown	6
21	D+		21	Gray	14
22	D–		22	Pink	4
23	+5 V (test)		23		
24	0 V		24		
25	Temperature–		25	Violet	9
Housing	External shield		Housing	External shield	Housing

Note: The **1 Vpp** signals are interpolated by a factor of 1024.

The rotary encoder to motor cable is ANILAM P/N 342000*XX*, where XX = 10, 15, 20, 25, 30, 35, 40, and 45 feet.

Touch Probe

Use the touch probe for workpiece or tool measurement during machining. Refer to **Table 2-3** and **Table 2-4**.

Table 2-3, X12: Touch Probe Input for Workpiece Measurement

D-Sub Terminal (Female) 15-pin	Assignment
1	0 V
2	Do not use
3	Ready
4	Start
5	+ 15 V ± 10 % (maximum 1000mA)
6	+ 5 V ± 5 % (maximum 100mA)
7	Battery warning (jump to pin 5 for wired probe)
8	0 V (U_N)
9	Trigger signal
10	$\overline{\text{Trigger signal}}$ [1]
11 to 15	Do not use
Housing	External shield

[1] Stylus at rest means logic level High

Table 2-4, X13: Touch Probe Input for Tool Measurement

D-Sub Terminal (Female) 9-pin	Assignment
1	Ready
2	0 V (U_N)
3	Do not use
4	+ 15 V ± 5 % (U_P)
5	Do not use
6	Do not use
7	+ 5 V ± 5 % (U_P)
8	Trigger signal
9	$\overline{\text{Trigger signal}}$ [1]
Housing	External shield

[1] Stylus at rest means logic level High

The 6000i CNC offers two inputs for touch probes:

- X12 connector – probe input for workpiece measurement (3-D probe)
- X13 connector – probe input for tool measurement

Corded **Renishaw® probes have to be interfaced to the control signal levels in either connector X12 and/or X13. The following figures represent the connection of the MP11 3-D corded touch probe (**Figure 2-6**), the TS27R tool probe (**Figure 2-7**), the **BLUM® laser probe (**Figure 2-8**), and the OMP-40 cordless touch probe (**Figure 2-9**) to the 6000i system:

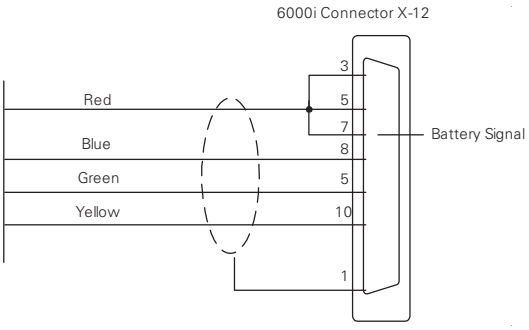


Figure 2-6, Renishaw® MP11 3-D Corded Touch Probe

Renishaw® MP11 curly cable: Blue 8, Green 5, Yellow 10, Red Jump pin 7 to pin 5

Note: An input should be connected through a draw-down resister in such a way that it is pulled low when the probe is connectred so the PLC can disable the spindle and tool changes.

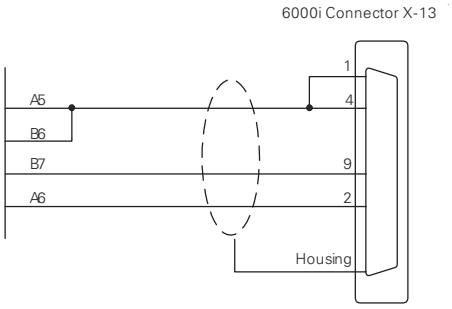


Figure 2-7, TS27R Tool Probe

Renishaw® MI-8 interface for B6 TS27-R tool probe: A5 4,B7 9, A6 2
Switch SW1 in MI-8 interface must be in N/C position

** Renishaw® is a registered trademark of Renishaw plc.

** BLUM® is a registered trademark of BLUM-Novotest GmbH.

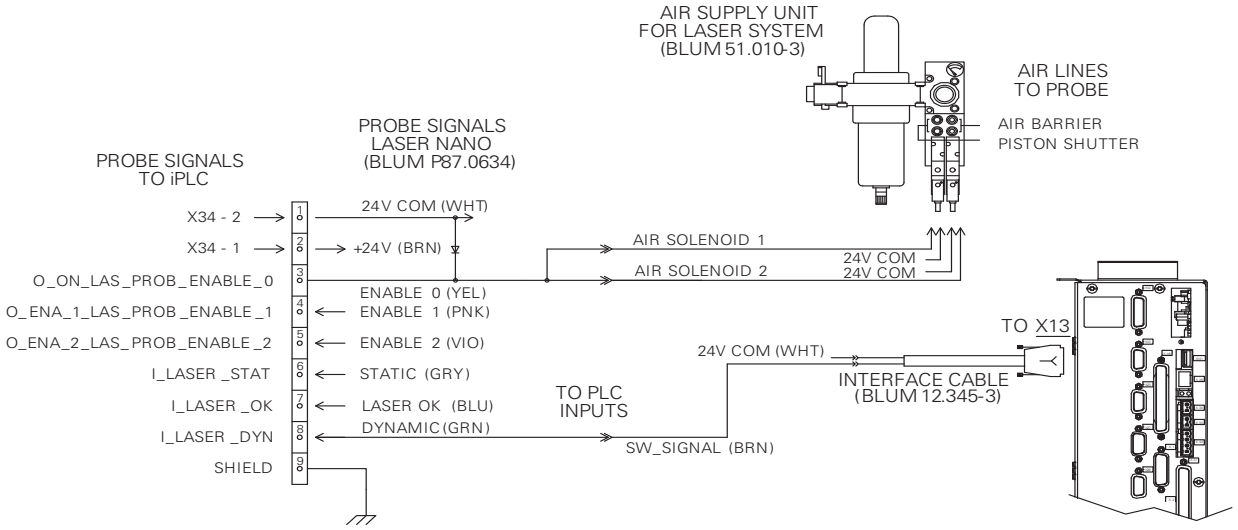


Figure 2-8, BLUM® Laser Touch Probe

The Blum Laser Tool Probe requires three outputs and three inputs to be setup in the PLC as shown above. It is also necessary to create four M-codes from M550 to M555. The M-code numbers cannot be changed as they are hard coded into the tool probing canned cycles.

M550:	Laser Probe On
M551	Laser Probe Off
M552	Enable 1 On
M553	Enable 1 Off
M554	Enable 2 On
M555	Enable 2 Off

Optional Blum Laser M-codes:

M556	Blow Nozzle On
M557	Blow Nozzle Off

The following topics are described:

- **PLC Additions for Blum Laser Probing**
- **Tool Probe Parameter Setup**
- **Spindle Probe Setup**

PLC Additions for Blum Laser Probing

The following topics are described:

- **PLC Input Labels and Connections to Blum Probe**
- **PLC Output Labels and Connections to Blum Probe**

PLC Input Labels and Connections to Blum Probe

I_Laser_OK	->	Blum “Laser OK” (Blue wire)
I_Laser_Dyn	->	Blum “Dynamic” (Green wire)
I_Laser_Stat	->	Blum “Static” (Gray wire)

PLC Output Labels and Connections to Blum Probe

O_On_Las_Prob_Enable_0	->	Blum “Enable 0” (Yellow wire) and also the coils of the Air Barrier and Piston Shutter. (Ensure you install a back surge diode between the solenoid coil and 24V common to protect the output.)
O_Ena_Las_Prob_Enable_1	->	Blum “Enable 1” (Pink wire)
O_Ena_Las_Prob_Enable_2	->	Blum “Enable 2” (Violet wire)
O_Blow_Nozzle_Las_Prop_On	->	Blum optional air blast

Plus 24 VDC connects to Blum +24V (Brown wire).

Common 24 VDC connects to Blum 24V COM (White wire) and also the common side of the solenoid coils mentioned above.

The turning On and Off of the outputs are set up in the PLC module touchpro.scr with six M-codes:

M550 Laser On
M551 Laser Off
M552 Enable 1 On
M553 Enable 1 Off
M554 Enable 2 On
M555 Enable 2 Off

And if the Blow Nozzle is on the probe, you would need to create two additional M-codes for that, using:

M556 Blow Nozzle On
M557 Blow Nozzle Off

This way, the Blum cycles can turn on and off the outputs individually as needed for more flexibility depending on the laser model.

An example for the M-code logic in touchpro.src is listed below and the associated entries for M-code definition in the ANILAM 6000i control will also need to be made in the PLC definition file GLB_PLCCFG.DEF and the machine configuration files of plc.cfg and plc_oem.cfg.

Example of the PLC code in touchpro.src:

```
L     NP_MG_M550_On_Laser_Probe_On
R     O_Ena_2_Las_Prob_Enable_2
S     O_Ena_1_Las_Prob_Enable_1
S     O_On_Las_Prob_Enable_0

L     NP_MG_M551_Off_Laser_Probe_Off
ON    I_control_operational
R     O_Ena_2_Las_Prob_Enable_2
R     O_Ena_1_Las_Prob_Enable_1
R     O_On_Las_Prob_Enable_0

L     O_On_Las_Prob_Enable_0
A     NP_MG_M550_On_Laser_Probe_On
S     PN_MG_quit_M_function

LN    O_On_Las_Prob_Enable_0
A     NP_MG_M551_Off_Laser_Probe_Off
S     PN_MG_quit_M_function

L     NP_MG_M552_Enable_1_On
R     O_Ena_2_Las_Prob_Enable_2
S     O_Ena_1_Las_Prob_Enable_1

L     O_Ena_1_Las_Prob_Enable_1
AN    O_Ena_2_Las_Prob_Enable_2
A     NP_MG_M552_Enable_1_On
S     PN_MG_quit_M_function

L     NP_MG_M554_Enable_2_On
R     O_Ena_1_Las_Prob_Enable_1
S     O_Ena_2_Las_Prob_Enable_2

L     O_Ena_2_Las_Prob_Enable_2
AN    O_Ena_1_Las_Prob_Enable_1
A     NP_MG_M554_Enable_2_On
S     PN_MG_quit_M_function
```

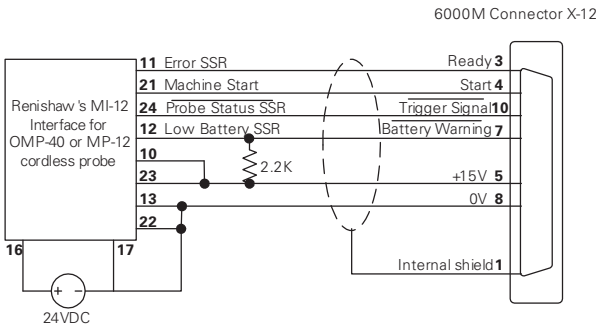


Figure 2-9, OMP-40 Cordless Touch Probe

Probes come in various different transmission systems: corded or cordless (infrared, radio, or inductive). When a corded 3-D touch probe is connected to the control, the spindle and tool changer should be automatically disabled in the PLC by means of an input signal which is hardwired to the probe connector. Also the low battery signal should be jumped to 15VDC when using a corded 3-D touch probe. When a cordless 3-D touch probe is connected to the control, low battery and not ready (e.g. transmission medium obstructed or probe in “sleep mode”) conditions are continuously monitored. Cordless probes usually go into a “sleeping mode” to conserve battery charge. These probes are automatically awakened by the CNC when probing is activated (M9387 X1).

Refer to the probe manufacture’s documentation for further details. Refer to the *6000i CNC User’s Manual*, P/N 627785-2X, for probe usage guidelines.

Tool Probe Parameter Setup

The tool probing parameters (the table probe) can be found on the control by going into the machine configuration as follows (refer to **Figure 2-10**):

1. From the Manual mode, press **SHIFT** then **F3** (**Config**).
2. When asked for a password, simply press the **ENTER** key.
3. Go into System>Probing>CfgToolProbingParameters
4. Remember that all numeric values are in metric.

Figure 2-10, Config Data Parameter Screen Capture

- Set **toolProbeType**: Standard or Laser (Standard is the default).
- Set **useAnilamLaserCycles** to **YES** to utilize the ANILAM laser table probe cycles. These cycles have limited compatibility for the Blum laser probe, therefore if another laser table probe is used, it may be necessary to set this to **NO** and use the cycles provided by the particular probe manufacturer. This parameter is read by the PLC to allow different input/output configuration.
- Set **moveAxisParallelToLaserBeam** (Laser probe only) to **YES** so the tool probing cycles will locate both X and Y axes when calibrating a tool. Set to **NO** if the probe is mounted in such a way that the axis parallel to the laser beam need not be moved.
- Set **nominalProbeStylusDiameter**, the overall nominal probe stylus diameter. For example, 12mm for the Renishaw® probe, or for the Heidenhain probe use 40mm. On a laser style probe this value would normally be zero. This is dependent on the probe style and specifications (refer to your probe documentation).
- Set **maxStrokeFromHome_FirstPick**, represents the distance from machine Z home with the shortest tool or the spindle face to just below the probe stylus top as the maximum stroke for the initial probe pick.
- Set **calibAndToolMeasurementRPM**, the spindle RPM for tool touch. [For example, set to 800 for standard probe type or 4000 for laser probe type tool presetter.]
- Set **probeOrientation**, the proper probe orientation. For example, if set to –1, the probe should be installed on the right side of the table pointing toward the left in the –X direction. See **Table 2-5**.

Table 2-5, Probe Orientation Settings

Probe Orientation Settings	Direction
1	Probe is pointing to the right as you are facing the machine in +X direction
-1	Probe is pointing to the left of the machine in the -X direction.
2	Probe is pointing away from you, toward the back of the machine in the +Y direction
-2	Probe is pointing toward you, toward the front of the machine in the -Y direction

- Set **ZFirstPickFeedRate_Fast**, the Z fast feedrate. [For example, set to 2000 mm per minute (mm/min).]

Warning: When using **G151**, the tool travels down beyond the top of the probe after the probe is tripped. For this reason, make sure that the fast feedrate is not so high as to cause the tool to travel past the probe travel causing damage to the probe. The maximum feedrate that can be used is specific to the machine and may need to be set much lower to prevent damage to the probe.

Note: There is a maximum feedrate setting which is set by your machine tool manufacturer in System > CfgHardware > maxTouchFeed. This will limit the maximum feedrate allowed in these feedrate parameters.

- Set the **ZFirstPickFeedRate_Medium,** the Z medium feedrate. [For example, set to 125 mm/min.]
- Set **ZFirstPickFeedRate_Slow**, the Z slow feedrate for the actual probe pick. [For example, set to 10.0 mm/min.]
- Set **ZRetractAmount**. Amount to retract from top of probe after pick. [For example, set to 4.0 mm.]
- Set **XYRetractAmount**. Amount to retract from side of probe after pick. [For example, set to 4.0 mm.]
- Set **ZRapidToStartPositionFromHome**. Install the longest tool in the spindle and bring the Z-axis to machine home. With a tape measure, measure the distance from the tool tip to within 13 mm above the top of the probe stylus and enter that number into **ZRapidToStartPositionFromHome**. When using **G151**, this causes the tool to rapid to this position in the Z-axis before starting the initial probe touch in the Z-axis. This saves time especially if the Z-feed must be set relatively slow to prevent probe over travel after the probe has been tripped.
- Set **diameterOfToolProbeGauge**. The default gauge diameter of the tool calibration standard. **diameterOfToolProbeGauge** can be overwritten by the **D** word in the **G150** cycle. **diameterOfToolProbeGauge** is used in the **G150** calibration only. [For example, set to 12 mm.]

Spindle Probe Setup

The probing parameters can be found on the control by going into the machine configuration as follows (refer to **Figure 2-11**):

1. From the Manual mode, press SHIFT then **F3** (**Config**).
2. When asked for a password, simply press the **ENTER** key.
3. Go into System>Probing>CfgSpindleProbingParameters
4. Remember that all numeric values are in metric.

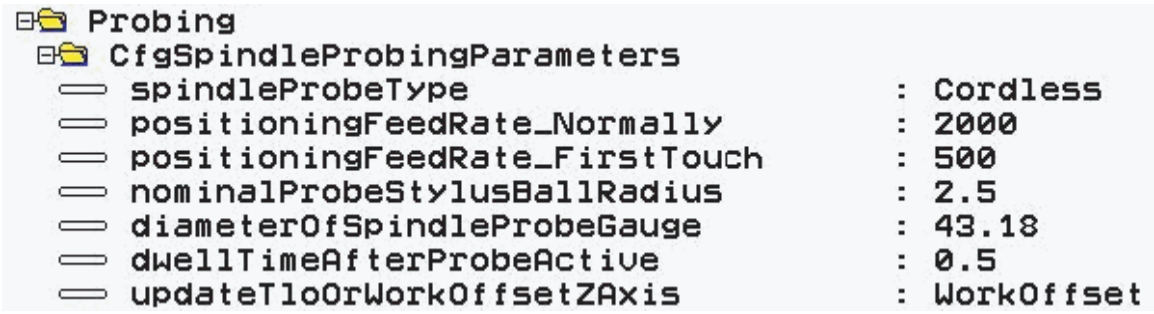


Figure 2-11, Config Data Parameter Screen Capture (Additional Parameters Displayed)

- Set **spindleProbeType**, to Corded, Cordless, depending on the probe style.

- Set **positioningFeedrate_Normally**, the feedrate the control uses while positioning the probe around the part. [For example, set to 1000 mm/min.]
- Set **positioningFeedrate_FirstTouch**, the feedrate the control uses while making its initial touch finding the surface it is measuring. [For example, 125 mm/min.]

Note: There is a maximum feedrate setting which is set by your machine tool manufacturer in System > CfgHardware > maxTouchFeed. This will limit the maximum feedrate allowed in these feedrate parameters.

- Set **nominalProbeStylusBallRadius**, the diameter as measured with a micrometer. [For example, set to 5 mm for a 10mm ball.]
- Set **diameterOfSpindleProbeGauge**, the exact diameter of the ring gauge used to calibrate the spindle probe. [For example, 25.4 mm.]
- Set **dwellTimeAfterProbeActive**, the time recommended by the particular probe manufacturer to wait after the probe is turned on before attempting a probe move.
- Set **updateTloOrWorkOffsetZAxis**, to WorkOffset or TLO. This only affects the **G141** with a **Q4** and **Q5** which will store the result as the difference of the spindle probes TLO and the top of the part in the Z axis work offset or will place the distance from machine home to the surface being probed in the current active tool length offset.

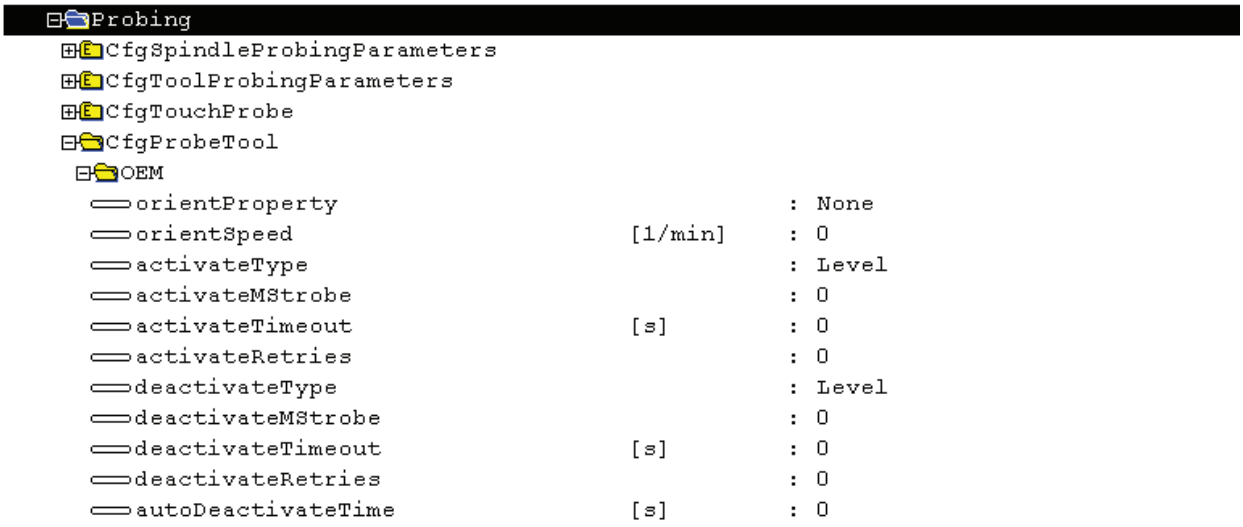


Figure 2-12, Probe Tool Settings Found in System > Probing > CfgProbeTool > OEM

- **orientSpeed**, **activateMStrobe**, **activateRetries**, **deactivateMStrobe**, **deactivateTimeout**, **deactivateRetries**, and **autoDeactivateTime** are not used and should be set to zero (0) or 1 in the case of **deactivateMStrobe**.
- Set **orientProperty** to NONE for a wired spindle probe and MultiOrient for a wireless probe.
- Set **activationType** to Pulse.
- Set **activateTimeout** to 4 seconds.
- Set **deactivateType** to Timeout.

PWM Connection to Axis/Spindle Motors

X51–**X55** are the Pulse Width Modulation (PWM) axes connections. **X56** is the spindle encoder PWM axis connection.

X51–X56 PWM Connection

Refer to **Table 2-6**.

Table 2-6, X51–X56: PWM Connection to Motor

Ribbon Connector	Assignment
1a	PWM U1
1b	0V U1
2a	PWM U_2
2b	0V U_2
3a	PWM U_3
3b	0V U_3
4a	–SH2
4b	0V (–SH2)
5a	–SH1B
5b	0V (–SH1B)
6a	+Iactl 1
6b	–Iactl 1
7a	0 V (analog)
7b	+Iactl $_2$
8a	–Iactl $_2$
8b	0 V (analog)
9a	Do not assign
9b	Do not assign
10a	Temperature warning
10b	Ready

Logic level:	5 V
Analog signals I_{act}:	±7.5 V
Maximum PWM frequency:	10 kHz

Note: The PWM frequency is defined via parameter. Defined via parameter “ampPwmFreq” found in “CfgPowerStage” of the axis which corresponds to the connectors X51–X56.

CNC Power Supply and Control Signals

X69: MC 400 Power Supply

Refer to **Table 2-7** for the ribbon cable pinout.

Table 2-7, X69: Ribbon Cable Connector - Pinout

X69 Ribbon Cable Connector 50-pin	Assignment
1a	+5 V
1b	+5 V
2a	+5 V
2b	+5 V
3a	+5 V
3b	+5 V
4a	+5 V
4b	+5 V
5a	+5 V
5b	+5 V
6a	+12 V
6b	+12 V
7a	+12 V
7b	+12 V
8a	+5 V (separated)
8b	0 V (separated)
9a	+15 V
9b	-15 V
10a	UZAN
10b	0 V
11a	IZAN
11b	0 V
12a	–RES.PS
12b	0 V
13a	–PF.PS
13b	GND

X69 Ribbon Cable Connector 50-pin	Assignment
14a	–ERR.ZU.GR
14b	GND
15a	–ERR.I.GR
15b	GND
16a	–ERR.TMP
16b	GND
17a	RDY.PS
17b	GND
18a	–ERR.ILEAK
18b	GND
19a	Do not assign
19b	GND
20a	Do not assign
20b	GND
21a	Do not assign
21b	GND
22a	Do not assign
22b	GND
23a	Reserved (SDA)
23b	GND
24a	Reserved (–SCL)
24b	GND
25a	–RES.S
25b	GND

(Continued on adjacent table…)

Control-Is-Ready Signal

X34: 24V Control-Is-Ready Signal

Refer to **Table 2-8**.

Table 2-8, X34: 24V Control-is-Ready Signal

Terminal	Assignment
1	+24 V input
2	0V

Power Supply for PLC Outputs

The PLC of the MC 400 as well as the IEB 404 are powered by the 24 V control voltage of the machine (in accordance with VDE 0551).

The control voltage must be smoothed with a minimum 1000 µF at a rated current capacity of 150 µF/A. At a current load of 15 A, for example, this corresponds to a capacity of 2250 µF.

EN 61 131-2:1994 permits:

- 5% alternating voltage component is permissible
- Minimum absolute value: 20.4 VDC
- Maximum absolute value: 28.8 VDC

Warning: Use only original replacement fuses.

The following topics are described:

- **Power Consumption**
- **Nominal Operating Current per Output**
- **X44: 24V Input for PLC Power Supply**
- **Power Supply for IEB 404**
- **IEB 404 Basic Module**
- **IEM 16-8D Input/Output Module**

Power Consumption

If half of the outputs are switched at the same time, the following are the values for power consumption:

MC 400: 115 W
IEB 404: Approx. 385 W

Nominal Operating Current per Output

MC 400: 0.150 A
IEM 16-8D: 2 A
Simultaneity
2 outputs with 2 A each
4 outputs with 1 A each
8 outputs with 0.5 A each
Total current:
Out0 to Out7: ≤ 4 A
Out0 to Out3: ≤ 2 A
Out4 to Out7: ≤ 2 A

X44: 24V Input for PLC Power Supply

Refer to **Table 2-9**.

Table 2-9, X44 I/O Module Supply Voltage - Pinout

Connection Terminal	Assignment	PLC Outputs
1	+24 VDC cannot be switched off via EMERGENCY STOP	Y0:24 – Y0:30 control-is-ready signal
2	+24 VDC can be switched off via EMERGENCY STOP	Y0:16 – Y0:28
3		Y0:0 – Y0:15
4	0 V	

Note: If the +24-V power supply (which cannot be shut off via emergency stop) is missing at X44, the error message **Supply voltage missing at X44** is displayed.

Power Supply for IEB 404

Note: The MC 400 cyclically monitors the supply voltage of the IEB 404.

IEB 404 Basic Module

Refer to **Table 2-10**.

Table 2-10, X3 (Power Supply for Logic Circuit) - Pinout

Terminal	Assignment
1	+24 VDC (20.4 V to 28.8 V)
2	+0 V

IEM 16-8D Input/Output Module

Refer to **Table 2-11**.

Table 2-11, X6 (Power Supply for PLC Outputs) - Pinout

Terminal	Assignment
9	+24 VDC (20.4 V to 28.8 V) for group 1
10	+24 VDC (20.4 V to 28.8 V) for group 2

Buffer Battery

Note: Make a data backup before changing the buffer battery.

Danger: When exchanging the buffer battery, remember:

- Switch off the machine and the 6000i.
- The buffer battery may be exchanged only by trained personnel.

Battery type: 1 lithium battery, type CR 2450N (Renata), Id. Nr. 315 878-01

If the voltage of the buffer battery falls below 2.6 V, the error message **Exchange buffer battery** is displayed. If the voltage does not exceed 2.6 V, the error message is reactivated after 30 minutes. Refer to **Figure 2-13**.

To exchange the battery:

- The buffer battery is located on the rear side of the MC 400.
- Exchange the battery; the new battery can be inserted in only one position.

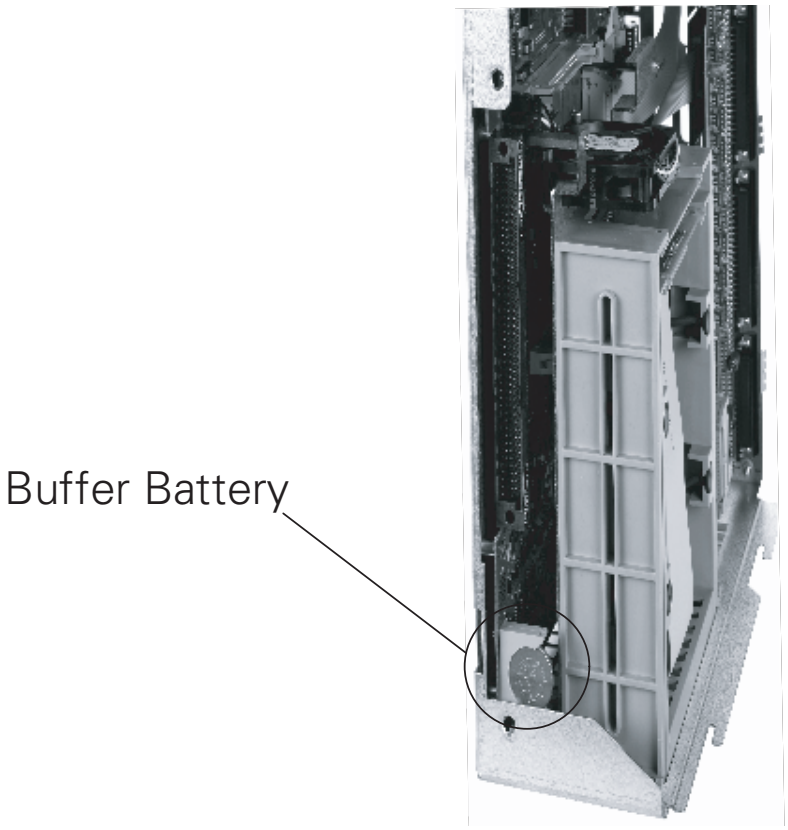


Figure 2-13, Buffer Battery

Analog Nominal Value Output

Output	±10 V
Maximum load of analog outputs	2 mA
Maximum capacity	2 nF

Six analog outputs are available:

Connection X8: Analog output 1 to 6

The following topics are described:

- **Nominal Value Output**
- **Analog Output 1 to 6**

Nominal Value Output

The connecting cable to the nominal value outputs must not have more than one intermediate terminal.

If you must branch to physically separate servo inputs, the connection must be made in a grounded terminal box. The housing of the terminal box must be electrically connected with the frame of the machine.

Analog Output 1 to 6

See **Table 2-12**.

Table 2-12, X8: Analog Output - Pinout

MC 400		Conn. Cable P/N 343007XX	
D-Sub Connector (Female) 15-Pin	**Assignment**	**D-Sub Connector (Female) 15-Pin**	**Color(s)**
1	Analog output 1: ±10 V	1	Brown
2	Do not assign	2	Brown/Green
3	Analog output 2: ±10 V	3	Yellow
4	Analog output 5: ±10 V	4	Red/Blue
5	Analog output 3: ±10 V	5	Pink
6	Analog output 5: 0 V	6	Gray/Pink
7	Analog output 4: ±10 V	7	Red
8	Analog output 6: ±10 V	8	Violet
9	Analog output 1: 0 V	9	White
10	Do not assign	10	White/Gray
11	Analog output 2: 0 V	11	Green
12	Do not assign	12	
13	Analog output 3: 0 V	13	Gray
14	Analog output 4: 0 V	14	Blue
15	Analog output 6: 0 V	15	Black
Shell	External shield	Shell	External shield

Analog Input

The MC 400 and the PLC Input/Output Board have analog inputs and inputs for Pt100 termistors.

The PLC Input/Output Board is available with and without analog inputs.

Voltage range:	–10 V to +10 V
Input resistance:	> 250 kW
Resolution:	10 mV (MC 400)
Internal value range:	–100 to +100, at a resolution of 100 mV

The following topics are described:

- **Inputs for Pt 100 Thermistors**
- **Analog Voltage Cable Characteristics**
- **X48: PLC Analog Input**

Inputs for Pt 100 Thermistors

Constant current:	5 mA
Temperature range:	0 °C (32 °F) to 100 °C (212 °F)
Resolution:	0.1 °C (32.18 °F)
Internal value range:	0 to 200, at a resolution of 0.5 °C (32.9 °F)

Analog Voltage Cable Characteristics

Characteristics of analog voltage connecting cable:

- Shielding
- 2 conductors with 0.14 mm^2 (AWG 7)
- Maximum length 50 meters (164 feet)

X48: PLC Analog Input

Warning: Remember to connect the analog inputs with the correct polarity.

Refer to **Table 2-13**.

Table 2-13, X48: PLC Analog Input - Pinout

S-Sub Connection (Female) 25-Pin	Assignment
1	I_1+ Constant current for Pt 100
2	I_1– Constant current for Pt 100
3	U_1+ Constant current for Pt 100
4	U_1– Constant current for Pt 100
5	I_2+ Constant current for Pt 100
6	I_2– Constant current for Pt 100
7	U_2+ Constant current for Pt 100
8	U_2– Constant current for Pt 100
9	I_3+ Constant current for Pt 100
10	I_3– Constant current for Pt 100
11	U_3+ Constant current for Pt 100
12	U_3– Constant current for Pt 100
13	Do not assign
14	Analog input 1: –10 to +10 V
15	Analog input 1: 0 V (reference potential)
16	Analog input 2: –10 V to +10 V
17	Analog input 2: 0 V (reference potential)
18	Analog input 3: –10 V to +10 V
19	Analog input 3: 0 V (reference potential)
20 to 25	Do not assign
Housing	External shield

Note: The interface complies with the requirements of EN 50 178 for “low electrical separation.”

Switching Inputs 24 VDC (PLC)

Input signals of the switching inputs on the MC 400 and the IEM 16-8D. Refer to **Table 2-14** and **Table 2-15**.

Table 2-14, I/O Module Voltage Requirements

Voltage Ranges:	MC 400	IEM 16-8D I/O Module
“1” signal: Ui	13 V to 30.2 V	13 V to 28.8 V
“0” signal: Ui	–20 V to 3.2 V	–20 V to 2.5 V
Current Ranges:		
“1” signal: Ui	3.8 mA to 8.9 mA	2.5 mA to 5.8 mA
“0” signal: Ii when Ui = 3.2 V	1.0 mA	0.3 mA

Table 2-15, Addresses of the Switching Inputs

Address	Number	Device
I:00 – I:31	31 + Control-Is-Ready Signal	MC 400, X42 (PLC input)
I:128 – I:152	25	MC 400, X46 (machine operating panel)
I:64 to I:127	64	IEB 404 (IEM 16-8D)
I:192 to I:255	64	IEB 404 (IEM 16-8D)
I:256 to I:319	64	IEB 404 (IEM 16-8D)
I:320 to I:383	64	IEB 404 (IEM 16-8D)

The following topics are described:

- **X42: PLC Input on the CNC**
- **PLC Inputs on the IEB 404**

X42: PLC Input on the CNC

The connecting cables are:

P/N 343007*XX* is available in lengths of 3 feet, 10, 15, 20, 30, 45, and 60 feet, and is an open-ended wire-type cable. The last two digits of the P/N indicate the cable length. For example, 34300703 indicates a 3-foot cable. Refer to **Table 2-16**.

Table 2-16, X42: PLC Input to the MC 400 - Pinout

MC 400		Conn. Cable P/N 343007XX	
Female open-ended wires	**Assignment**	**D-Sub Connctr. 37-Pin**	**Color(s)**
1	I:00 **X HOME	1	Black
2	I:01 **Y HOME	2	Red
3	I:02 **Z HOME	3	White
4	I:03 **CNCACK	4	Green
5	I:04 **U HOME	5	Orange
6	I:05 **W HOME	6	Blue
7	I:06	7	Brown
8	I:07	8	Yellow
9	I:08	9	Violet
10	I:09	10	Gray
11	I:10	11	Pink
12	I:11	12	Tan
13	I:12	13	Red/Green
14	I:13	14	Red/Yellow
15	I:14	15	Red/Black
16	I:15	16	White/Black
17	I:16	17	White/Red
18	I:17	18	White/ Green
19	I:18	19	White/Yel-low
20	I:19	20	White/Blue
21	I:20	21	White/ Brown
22	I:21	22	White/ Orange
23	I:22	23	White/Gray
24	I:23	24	White/Violet

** Hard Coded Function

(Continued…)

Table 2-16, X42: PLC Input to the MC 400 - Pinout *(Continued)*

MC 400		Conn. Cable P/N 343007XX	
Female open-ended wires	**Assignment**	**D-Sub Connctr. 37-Pin**	**Color(s)**
25	I:24	25	White/Black/Red
26	I:25	26	White/Black/Green
27	I:26	27	White/Black/Yellow
28	I:27	28	White/Black/Blue
29	I:28	29	White/Black/Brown
30	I:29	30	White/Black/ Orange
31	I:30 **Software ESTOP	31	White/Black/Gray
32	I:31 **Software DRIVE ENABLE	32	White/Black/Violet
33	I:32 **DRIVE ENABLE	33	White/Black/Black
34	Do not assign	34	White/Red/ Black
35	Do not assign	35	White/Red/ Red
36	Do not assign	36	White/Red/ Green
37	Do not assign	37	White/Red/ Blue
Shell	External shield	Shell	External shield

** Hard Coded Function

PLC Inputs on the IEB 404

X4 to X5: PLC Inputs

Refer to **Table 2-17** and **Table 2-18, X5: IEM 16-8D I/O Module - Pinout**.

Note: he 0-V terminals of X4 and X5 of the IEM 16-8D are connected internally. These connections are used for connecting the potential of the electronics and for operating the LEDs. Since only a low current is required.

Table 2-17, X4: IEM 16-8D I/O Module - Pinout

X4										
Assignment		**Terminal**								
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
First IEB 404	Slot 1	0 V	0 V	I:64	I:65	I:66	I:67	I:68	I:69	I:70	I:71
	Slot 2	0 V	0 V	I:80	I:81	I:82	I:83	I:84	I:85	I:86	I:87
	Slot 3	0 V	0 V	I:96	I:97	I:98	I:99	I:100	I:101	I:102	I:103
	Slot 4	0 V	0 V	I:112	I:113	I:114	I:115	I:116	I:117	I:118	I:119
Second IEB 404	Slot 1	0 V	0 V	I:192	I:193	I:194	I:195	I:196	I:197	I:198	I:199
	Slot 2	0 V	0 V	I:208	I:209	I:210	I:211	I:212	I:213	I:214	I:215
	Slot 3	0 V	0 V	I:224	I:225	I:226	I:227	I:228	I:229	I:230	I:231
	Slot 4	0 V	0 V	I:240	I:241	I:242	I:243	I:244	I:245	I:246	I:247
Third IEB 404	Slot 1	0 V	0 V	I:256	I:257	I:258	I:259	I:260	I:261	I:262	I:263
	Slot 2	0 V	0 V	I:272	I:273	I:274	I:275	I:276	I:277	I:278	I:279
	Slot 3	0 V	0 V	I:288	I:289	I:290	I:291	I:292	I:293	I:294	I:295
	Slot 4	0 V	0 V	I:304	I:305	I:306	I:307	I:308	I:309	I:310	I:311
Fourth IEB 404	Slot 1	0 V	0 V	I:320	I:321	I:322	I:323	I:324	I:325	I:326	I:327
	Slot 2	0 V	0 V	I:336	I:337	I:338	I:339	I:340	I:341	I:342	I:343
	Slot 3	0 V	0 V	I:352	I:353	I:354	I:355	I:356	I:357	I:358	I:359
	Slot 4	0 V	0 V	I:368	I:369	I:370	I:371	I:372	I:373	I:374	I:375

Table 2-18, X5: IEM 16-8D I/O Module - Pinout

X5											
Assignment		**Terminal**									
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
First IEB 404	Slot 1	0 V	0 V	I:72	I:73	I:74	I:75	I:76	I:77	I:78	I:79
	Slot 2	0 V	0 V	I:88	I:89	I:90	I:91	I:92	I:93	I:94	I:95
	Slot 3	0 V	0 V	I:104	I:105	I:106	I:107	I:108	I:109	I:110	I:111
	Slot 4	0 V	0 V	I:120	I:121	I:122	I:123	I:124	I:125	I:126	I:127
Second IEB 404	Slot 1	0 V	0 V	I:200	I:201	I:202	I:203	I:204	I:205	I:206	I:207
	Slot 2	0 V	0 V	I:216	I:217	I:218	I:219	I:220	I:221	I:222	I:223
	Slot 3	0 V	0 V	I:232	I:233	I:234	I:235	I:236	I:237	I:238	I:239
	Slot 4	0 V	0 V	I:248	I:249	I:250	I:251	I:252	I:253	I:254	I:255
Third IEB 404	Slot 1	0 V	0 V	I:264	I:265	I:266	I:267	I:268	I:269	I:270	I:271
	Slot 2	0 V	0 V	I:280	I:281	I:282	I:283	I:284	I:285	I:286	I:287
	Slot 3	0 V	0 V	I:296	I:297	I:298	I:299	I:300	I:301	I:302	I:303
	Slot 4	0 V	0 V	I:312	I:313	I:314	I:315	I:316	I:317	I:318	I:319
Fourth IEB 404	Slot 1	0 V	0 V	I:328	I:329	I:330	I:331	I:332	I:333	I:334	I:335
	Slot 2	0 V	0 V	I:344	I:345	I:346	I:347	I:348	I:349	I:350	I:351
	Slot 3	0 V	0 V	I:360	I:361	I:362	I:363	I:364	I:365	I:366	I:367
	Slot 4	0 V	0 V	I:376	I:377	I:378	I:379	I:380	I:381	I:382	I:383

Switching Outputs 24 VDC (PLC)

The following topics are described:

- **Output Signals and Addresses**
- **X41: PLC Output on the CNC**
- **X6: PLC outputs on the IEB 404**

Output Signals and Addresses

The switching outputs are transistor outputs with current limitation.

Please note:

- Permissible load: Resistive load (ohmic load)—inductive load (e.g., relay, contactor) only with quenching diode parallel to inductance
- MC 400: Short circuiting of **one** output is **permissible. No more than one** output may be short-circuited **at one time.**
- IEM 16-8D: The outputs are short-circuit proof.

Refer to **Table 2-19** and **Table 2-20, Output Addresses**.

Table 2-19, Output Signals

	MC 400, IEM 16-8D
Min. output voltage for “1” signal	3 V below supply voltage

Note: The switching outputs need a minimum load of 5 mA.
They conform to EN 61131-2.

Warning: PLC outputs must neither be connected to a 24-V supply, nor to other PLC outputs with a difference in potential. Otherwise, the voltage present at the PLC outputs is transmitted to the power supply. As a result, the PLC outputs that can be switched off may nevertheless be supplied with this voltage.

Table 2-20, Output Addresses

Address	Number	Device
O:0 to O:30	31	MC 400, X41 (PLC output)
O:0 to O:7	8	MC 400, X46 (machine operating panel)
O:32 to O:62	31	First PLC I/O unit
O:64 to O:94	31	Second PLC I/O unit
O:128 to O:158	31	Third PLC I/O unit
O:160 to O:190	31	Fourth PLC I/O unit

The "control-is-ready" output at X41 can have the same load as a normal PLC output. If a higher current is required for switching a relay, the "control-is-ready" outputs of the IEB 404, 406, 408s can be used in addition. A separate power supply for the IEB 404, 406, 408s is necessary for this.

X41: PLC Output on the CNC

The connecting cables are:

P/N 343001XX is available in lengths of 5 feet to 75 feet in increments of 5 feet, and is terminated with a female connector. The last two digits of the P/N indicate the cable length. For example, 34300115 indicates a 15-foot cable.

P/N 343007XX is available in lengths of 3 feet, 10, 15, 20, 30, 45, and 60 feet, and is an open-ended wire-type cable. The last two digits of the P/N indicate the cable length. For example, 34300703 indicates a 3-foot cable.

Refer to **Table 2-21, X41: PLC Output on the CNC - Pinout**.

Table 2-21, X41: PLC Output on the CNC - Pinout

37-pin	IPI	Assignment	Color
1		Do not assign	BLK
2		Do not assign	RED
3		Do not assign	WHT
4		Do not assign	GRN
5		Do not assign	ORN
6		Do not assign	BLU
7		Do not assign	BRN
8		Do not assign	YEL
9	O:08		VIO
10	O:09		GRAY
11	O:10		PINK
12	O:11		TAN
13	O:12		RED/GRN
14	O:13		RED/YEL
15	O:14		RED/BLK
16	O:15		WHT/BLK
17	O:16		WHT/RED
18	O:17		WHT/GRN
19	O:18		WHT/YEL
20	O:19		WHT/BLU
21	O:20		WHT/BRN
22	O:21		WHT/ORN
23	O:22		WHT/GRAY
24	O:23		WHT/VIOL
25	O:24		WHT/BLK/RED
26	O:25		WHT/BLK/GRN
27	O:26		WHT/BLK/YEL
28	O:27		WHT/BLK/BLU
29	O:28		WHT/BLK/BRN
30	O:29		WHT/BLK/ORN
31	O:30		WHT/BLK/GRAY
32		Do not assign	WHT/BLK/VIOL
33		Do not assign	WHT/BLK/BLK
34		CNC READY SIGNAL	WHT/RED/BLK
35		Do not assign	WHT/RED/RED
36		Do not assign	WHT/RED/GRN
37		Do not assign	WHT/RED/BLU
Shell		EXTERNAL SHIELD	

X6: PLC Outputs on the IEB 404

Refer to **Table 2-22**.

Table 2-22, IEM 16-8D I/O Module - Pinout

X6										
Assignment		**Terminal**								
		1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**
First IEB 404	Slot 1	O:32	O:33	O:34	O:35	O:36	O:37	O:38	O:39[a]	+24 V[b]	+24 V[c]
	Slot 2	O:40	O:41	O:42	O:43	O:44	O:45	O:46	O:47[a]	+24 V[b]	+24 V[c]
	Slot 3	O:48	O:49	O:50	O:51	O:52	O:53	O:54	O:55[a]	+24 V[b]	+24 V[c]
	Slot 4	O:56	O:57	O:58	O:59	O:60	O:61	O:62	-	+24 V[b]	+24 V[c]
Second IEB 404	Slot 1	O:64	O:65	O:66	O:67	O:68	O:69	O:70	O:71[a]	+24 V[b]	+24 V[c]
	Slot 2	O:72	O:73	O:74	O:75	O:76	O:77	O:78	O:79[a]	+24 V[b]	+24 V[c]
	Slot 3	O:80	O:81	O:82	O:83	O:84	O:85	O:86	O:87[a]	+24 V[b]	+24 V[c]
	Slot 4	O:88	O:89	O:90	O:91	O:92	O:93	O:94	-	+24 V[b]	+24 V[c]
Third IEB 404	Slot 1	O:128	O:129	O:130	O:131	O:132	O:133	O:134	O:135[a]	+24 V[b]	+24 V[c]
	Slot 2	O:136	O:137	O:138	O:139	O:140	O:141	O:142	O:143[a]	+24 V[b]	+24 V[c]
	Slot 3	O:144	O:145	O:146	O:147	O:148	O:149	O:150	O:151[a]	+24 V[b]	+24 V[c]
	Slot 4	O:152	O:153	O:154	O:155	O:156	O:157	O:158	-	+24 V[b]	+24 V[c]
Fourth IEB 404	Slot 1	O:160	O:161	O:162	O:163	O:164	O:165	O:166	O:167[a]	+24 V[b]	+24 V[c]
	Slot 2	O:168	O:169	O:170	O:171	O:172	O:173	O:174	O:175[a]	+24 V[b]	+24 V[c]
	Slot 3	O:176	O:177	O:178	O:179	O:180	O:181	O:182	O:183[a]	+24 V[b]	+24 V[c]
	Slot 4	O:184	O:185	O:186	O:187	O:188	O:189	O:190	-	+24 V[b]	+24 V[c]

a. The function of this terminal can be set with a sliding switch on the rear side of the PLD 16-8 I/O module:
   Setting 1: Control-is-ready signal
   Setting 2: PLC output
b. Group 1 (terminals 1 to 4)
c. Group 2 (terminals 5 to 8)

Note: If you use only the outputs at X6 for an IEM 16-8D I/O unit (and no inputs), the 0-V connection for supplying the electronics and for operating the LEDs must be established at X4 or X5.

Note: The 6000i cyclically monitors the PLC outputs of the IEB 404 for a short-circuit.

Flat Panel Display

Refer to **Table 2-23** for the pin layout for the MC 400, the connecting cable, and the flat panel display.

Table 2-23, X49: Flat Panel Display to MC 400 - Pinout

MC 400		Connecting cable P/N 343000XX				FP 6000i
D-sub Connctn. (Female) 62-pin	**Assignment Name**	**D-sub Connector (Male) 62-pin**	**Color of Wire**	**No. of Pair**	**D-sub Connector (Female) 62-pin**	**D-sub Connector (Male) 62-pin**
1 22	0 V 0 V	1 22	BLK RED	1	1 22	1 22
2 23	CLK.P CLP.P*	2 23	BLK WHT	2	2 23	2 23
3 24	HSYNC HSYNC*	3 24	BLK GRN	3	3 24	3 24
4 25	BLANK BLANK	4 25	BLK BLU	4	4 25	4 25
5 26	VSYNC VSYNC*	5 26	BLK YEL	5	5 26	5 26
6 27	0 V 0 V	6 27	BLK BRN	6	6 27	6 27
7 28	R0 R0*	7 28	BLK ORN	7	7 28	7 28
8 29	R1 R1*	8 29	RED WHT	8	8 29	8 29
9 30	R2 R2*	9 30	RED GRN	9	9 30	9 30
10 31	R3 R3*	10 31	RED BLU	10	10 31	10 31
11 32	0 V 0 V	11 32	RED YEL	11	11 32	11 32
12 33	G0 G0*	12 33	REF BRN	12	12 33	12 33
13 34	G1 G1*	13 34	RED ORN	13	13 34	13 34
14 35	G2 G2*	14 35	GRN WHT	14	14 35	14 35
15 36	G3 G3*	15 36	GRN BLU	15	15 36	15 36
16 37	0 V 0 V	16 37	GRN BRN	16	16 37	16 37

(Continued…)

Table 2-23, X49: Flat Panel Display to MC 400 - Pinout (Continued)

MC 400		Connecting cable P/N 343000XX				FP 6000i
D-sub Connctn. (Female) 62-pin	**Assignment Name**	**D-sub Connector (Male) 62-pin**	**Color of Wire**	**No. of Pair**	**D-sub Connector (Female) 62-pin**	**D-sub Connector (Male) 62-pin**
17 38	B0 B0*	17 38	GRN YEL	17	17 38	17 38
18 39	B1 B1*	18 39	GRN ORN	18	18 39	18 39
19 40	B2 B2*	19 40	WHT BLU	19	19 40	19 40
20 41	B3 B3*	20 41	WHT YEL	20	20 41	20 41
43 44	DISP.LOW* DISP.LOW	43 44	WHT BRN	21	43 44	43 44
45 46	DISP.ON* DISP.ON	45 46	WHT ORN	22	45 46	45 46
47 48	C0 C1	47 48	BLU YEL	23	47 48	47 48
49 50	C2 C3	49 50	BLU BRN	24	49 50	49 50
51 52	C4 C5	51 52	BLU ORN	25	51 52	51 52
21	0 V	21	N.C.		21	21
42	0 V	42	N.C.		42	42
53	Free	53	N.C.		53	53
54	Free	54	N.C.		54	54
55	Free	55	N.C.		55	55
56	Free	56	N.C.		56	56
57	Free	57	N.C.		57	57
58	Free	58	N.C.		58	58
59	Free	59	N.C.		59	59
60	Free	60	N.C.		60	60
61	Free	61	N.C.		61	61
62	Free	62	N.C.		62	62
Housing		Housing			Housing	Housing

Manual Panel

Refer to **Table 2-24**.

Table 2-24, X46: Machine Operating Panel - Pinout

MC 400 X46		Connecting Cable P/N 343001XX			Manual Panel X3
D-Sub Connctn. (Female) 37-Pin	Assignment	D-Sub Connctr. (Male) 37-Pin	Color(s)	D-Sub Connctr. (Female) 37-Pin	D-Sub Connctn. (Male) 37-Pin
1	I:128	1	Gray/Red	1	Key 0
2	I:129	2	Brown/Black	2	Key 1
3	I:130	3	White/Black	3	Key 2
4	I:131	4	Green/Black	4	Key 3
5	I:132	5	Brown/Red	5	Key 4
6	I:133	6	White/Red	6	Key 5
7	I:134	7	White/Green	7	Key 6
8	I:135	8	Red/Blue	8	Key 7
9	I:136	9	Yellow/Red	9	Axis 1 – X
10	I:137	10	Gray/Pink	10	Axis 2 – Y
11	I:138	11	Black	11	Axis 3 – Z
12	I:139	12	Pink/Brown	12	FR01
13	I:140	13	Yellow/Blue	13	FR02
14	I:141	14	Green/Blue	14	FR03
15	I:141	15	Yellow	15	FR04
16	I:143	16	Red	16	JOG1X1
17	I:144	17	Gray	17	JOG2X10
18	I:145	18	Blue	18	JOG3X100
19	I:146	19	Pink	19	SS01
20	I:147	20	White/Gray	20	SS02
21	I:148	21	Yellow/Gray	21	SS03
22	I:149	22	Green/Red	22	SS04
23	I:150	23	White/Pink	23	EXIN1
24	I:151	24	Gray/Green	24	EXIN2
25	I:152	25	Yellow/Brown	25	EXIN3
26	O:00	26	Gray/Brown	26	EXOUT1
27	O:01	27	Yellow/Black	27	EXOUT2
28	O:02	28	White/Yellow	28	LED COM

(Continued…)

Table 2-24, X46: Machine Operating Panel - Pinout (Continued)

MC 400 X46		Connecting Cable P/N 343001XX			Manual Panel X3
D-Sub Connctn. (Female) 37-Pin	**Assignment**	**D-Sub Connctr. (Male) 37-Pin**	**Color(s)**	**D-Sub Connctr. (Female) 37-Pin**	**D-Sub Connctn. (Male) 37-Pin**
29	O:03	29	Gray/Blue	29	LEDCLRDY
30	O:04	30	Pink/Blue	30	LEDSFWD
31	O:05	31	Pink/Red	31	LEDSREV
32	O:06	32	Brown/Blue	32	LEDSOFF
33	O:07	33	Pink/Green	33	LEDSVRST
34	0 V (PLC)	34	Brown	34	24VCOM
35	0 V (PLC)	35	Yellow/Pink	35	24VCOM
36	+24 V (PLC)	36	Violet	36	+24V
37	+24 V (PLC)	37	White	37	+24V
Housing	External shield	Housing	External shield	Housing	

CNC Keyboard

The connecting cable, P/N 343001XX is available in lengths of 5 feet to 75 feet in increments of 5 feet. The last two digits of the P/N indicate the cable length. For example, 34300115 indicates a 15-foot cable. Refer to **Table 2-25**.

Table 2-25, X45: CNC Keyboard - Pinout

MC 400 X45	
D-Sub Connctn. (Female) 37-Pin	**Assignment**
1	RL0
2	RL1
3	RL2
4	RL3
5	RL4
6	RL5
7	RL6
8	RL7
9	RL8
10	RL9
11	RL10
12	RL11
13	RL12
14	RL13
15	RL14
16	RL15
17	RL16
18	RL17
19	RL18

MC 400 X45	
D-Sub Connctn. (Female) 37-Pin	**Assignment**
20	SL0
21	SL1
22	SL2
23	SL3
24	SL4
25	SL5
26	SL6
27	SL7
28	RL19
29	RL20
30	Not assigned
31	RL21
32	RL22
33	RL23
34	Spindle override (wiper)
35	Feedrate override (wiper)
36	+5 V override potentiometer
37	0 V override potentiometer
Housing	External shield

(Continued on adjacent table…)

I/O Module Connection

Refer to **Table 2-26** and **Table 2-27, X2: I/O Module - Pinout** for PLC expansion IEB 40X on the IEB 40X

Table 2-26, X147: I/O Module to MC 400 - Pinout

MC 400		Connecting Cable P/N 624517-XX			I/O Module	
D-Sub Connctn. (Male) 26-Pin	**Assignment**	**D-Sub Connctr. (Female) 26-Pin**	**Color(s)**	**D-Sub Connctr. (Male) 26-Pin**	**X1 D-Sub Connctn. (Female) 26-Pin**	**Assignment**
1	0 V	1	Black	1	1	0 V
2	0 V	2	Violet	2	2	0 V
3	0 V	3		3	3	0 V
4	Do not assign	4		4	4	Do not assign
5	Address 6	5	Yellow	5	5	Address 6
6	INTERRUPT	6	Blue	6	6	INTERRUPT
7	RESET	7	Red	7	7	RESET
8	WRITE EXTERN	8	Gray	8	8	WRITE EXTERN
9	WRITE EXTERN	9	Pink	9	9	WRITE EXTERN
10	Address 5	10	Green	10	10	Address 5
11	Address 3	11	White	11	11	Address 3
12	Address 1	12	Brown	12	12	Address 1
13	Do not assign	13		13	13	Do not assign
14	+ 5 V (output)	14	White/Blue	14	14	+5 V
15	+ 5 V (feedback)	15	Brown/Blue	15	15	+5 V
16	PCB identifier 2	16	White/Pink	16	16	PCB identifier 2

(Continued…)

Table 2-26, X147: I/O Module to MC 400 - Pinout (Continued) .

MC 400		Connecting Cable P/N 624517-XX			I/O Module	
D-Sub Connctn. (Male) 26-Pin	**Assignment**	**D-Sub Connector (Female) 26-Pin**	**Color(s)**	**D-Sub Connector (Male) 26-Pin**	**X1 D-Sub Connctn. (Female) 26-Pin**	**Assignment**
17	PCB identifier 1	17	Pink/Brown	17	17	PCB identifier 1
18	Address 7	18	Brown/ Green	18	18	Address 7
19	Serial IN 1	19	White/Gray	19	19	Serial IN
20	EM. STOP	20	Gray/Brown	20	20	EM. STOP
21	Serial OUT	21	White/ Yellow	21	21	Serial OUT
22	Serial OUT	22	Yellow/ Brown	22	22	Serial OUT
23	Address 4	23	White/ Green	23	23	Address 4
24	Address 2	24	Gray/Pink	24	24	Address 2
25	Address 0	25	Red/Blue	25	25	Address 0
26		26			26	
Housing	External shield	Housing	External shield	Housing	Housing	External shield

The connecting cable, P/N 624517-XX is available in lengths of: 2, 3, 6, 9, 15, 20, and 25 feet. The last two digits of the P/N indicate the cable length. For example, 624517-09 indicates a 9-foot cable.

For space requirements, refer to:

- **Figure 9-7, I/O EXP. BASE: 4-SLOTS (P/N 624498-01, IEB 404), 6-SLOTS (P/N 624500-01, IEB 406), 8-SLOTS (P/N 624501-01, IEB 408)**
- **Figure 9-10, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) Dimensions**
- **Figure 9-12, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Dimensions**

Table 2-27, X2: I/O Module - Pinout for PLC expansion IEB 40X on the IEB 40X

IEB 40X		Connecting Cable P/N 624517-XX			IEB 40X on IEB 40X	
X2 D-sub connctn. (male) 26-pin	**Assignment**	**D-sub connctn. (female) 26-pin**		**D-sub connctn. (male) 26-pin**	**X1 D-sub connctn. (female) 26-pin**	**Assignment**
1	0 V	1	Black		1	0 V
2	0 V	2	Violet		2	0 V
3	0 V	3			3	0 V
4	Do not assign	4			4	Do not assign
5	Address 6	5	Yellow		5	Address 6
6	INTERRUPT	6	Blue		6	INTERRUPT
7	RESET	7	Red		7	RESET
8	WRITE EXTERN	8	Gray		8	WRITE EXTERN
9	WRITE EXTERN	9	Pink		9	WRITE EXTERN
10	Address 5	10	Green		10	Address 5
11	Address 3	11	White		11	Address 3
12	Address 1	12	Brown		12	Address 1
13	Do not assign	13			13	Do not assign
14	+5 V (output)	14	White/Blue		14	+5 V (output)
15	+5 V (feedback)	15	Brown/Blue		15	+5 V (feedback)
16	PCB identifier 2	16	White/Pink		16	PCB identifier 2
17	PCB identifier 1	17	Pink/Brown		17	PCB identifier 1
18	Address 7	18	Brown/Green		18	Address 7
19	Serial IN 1	19	White/Gray		19	Serial IN
20	EM. STOP	20	Gray/Brown		20	EM. STOP
21	Serial OUT	21	White/Yellow		21	Serial OUT
22	Serial OUT	22	Yellow/Brown		22	Serial OUT
23	Address 4	23	White/Green		23	Address 4
24	Address 2	24	Gray/Pink		24	Address 2
25	Address 0	25	Red/Blue		25	Address 0
26		26			26	
Housing	External shield	Housing	External shield	Housing	Housing	External shield

Data Interfaces

Three data interfaces are available:

- **RS-232-C/V.24 (COM1)**
- **RS-422/V.11 (COM2)**
- **Ethernet**

X27: RS-232-C/V.24 Data Interface (COM1)

Maximum cable length is 20 m (65.6 ft.). Refer to **Table 2-28**.

Table 2-28, X27: RS-232-C/V.24 Data Interface (COM1) - Pinout

MC 400	
X27 D-Sub Connctn. (Male) 9-Pin	**Assignment**
1	Do not assign
2	RXD
3	TXD
4	DTR
5	Signal GND
6	DSR
7	RTS
8	CTS
9	Do not assign
Housing	External shield

Note: The interface complies with the requirements of EN 50 178 for “low electrical separation.”

X28: RS-422/V.11 Data Interface (COM2)

Maximum cable length is 50 m (164 ft.). Refer to **Table 2-29**.

Table 2-29, X28: RS-422/V.11 Data Interface (COM2) - Pinout

MC 400	
X28 D-Sub Connection (Female) 9-Pin	**Assignment**
1	RTS
2	DTR
3	RxD
4	TxD
5	Signal GND
6	CTS
7	DSR
8	RxD
9	TxD
Housing	External shield

Note: The interface complies with the requirements of EN 50 178 for “low electrical separation.”

X26: Ethernet Interface RJ45 Port

Maximum data transfer rate: Approximately 2 to 5 Mbps (depending on the file type and network utilization).

Maximum cable length (shielded cable): 100 m (328 ft.).

Refer to **Table 2-30**, **Figure 2-14**, and **Table 2-31** for the meaning of the X26 Light Emitting Diodes (LEDs).

Table 2-30, X26: Ethernet Interface RJ45 Port - Pinout

RJ45 Connection. (Female) 8-Pin	Assignment
1	TX+
2	TX–
3	REC+
4	Do not assign
5	Do not assign
6	REC–
7	Do not assign
8	Do not assign
Housing	External shield

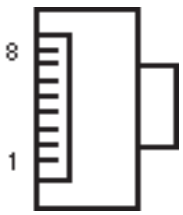


Figure 2-14, RJ45 Connector

Note: The interface complies with the requirements of EN 50 178 for “low electrical separation.”

Table 2-31, X26: Ethernet Interface LEDs

LED	Status	Meaning
Green	Blinks	Interface is active
	Off	Interface is inactive
Yellow	On	100 Mb network
	Off	10 Mb network

USB Interface

Refer to **Table 2-32**.

Table 2-32, X141, X142: USB Interface - Pinout

USB Connection (Female) 4-pin	Assignment
1	+5 V
2	USBP–
3	USBP+
4	GND

The following topic is described:

- **USB Hub**

USB Hub

The power supply for the USB hub must comply with EN 50 178, 5.88 requirements for "low voltage electrical separation."

Connections on the USB hub (368 735-01). Refer to **Table 2-33**.

Table 2-33, USB Hub - Pinout

Connection designation	Function
X1	24 V power supply
X32	5-V output
X140	USB input (to the MC 400)
X141	USB output 1
X142	USB output 2
X143	USB output 3
X144	USB output 4

Drive Controller Enable

A drive controller can be enabled by the NC software only if the controller is enabled with 24 V on X150 and on X42, pin 33. Refer to **Table 2-16, X42: PLC Input to the MC 400 - Pinout**.

The following topic is described:

- **X150: Drive controller enabling for axis groups**

X150: Drive controller enabling for axis groups

The connecting terminal X150 is located on the bottom of CC 600. Refer to **Figure 2-5, MC 400 and CC 600 Connections**.

X150 controls drive enabling for the axis groups on the first controller board (PWM outputs X51 to X56). Refer to **Table 2-34**.

Note: The pin of an axis group must always be wired to the connector on whose PCB the control loop is located.

Table 2-34, X150: Axis-specific Drive Release - Pinout

X150 Terminal	Assignment
1	+24 V**1; drive controller enabling for axis group 1
2	+24 V**1; drive controller enabling for axis group 2
3	+24 V**1; drive controller enabling for axis group 3
4	Not used for CC 600
5	Not used for CC 600
6	Not used for CC 600
7	Reserved, do not assign
8	Reserved, do not assign
9	0 V

**1 Maximum current consumption 10 mA

PLC Input/Output Units

IEB 404

Up to four IEB 404s can be connected to the MC 400.

The IEB 404 basic modules can be fitted with any combination of IEM 16-8D I/O modules and IEM 4-4A analog modules. It is also possible to leave gaps, since not all slots on the IEB 404 must be used.

Refer to **Table 2-26, X147: I/O Module to CNC - Pinout** for PLC expansion on the MC 400 and **Table 2-27, X2: I/O Module - Pinout** for PLC expansion IEB 404 on the IEB 404.

Meaning of the LEDs on the IEM 16-8D

Refer to **Table 2-35**.

Table 2-35, IEM 16-8D LEDs Description

LED	Meaning
Red LED at X4, pin 1	Short circuit of the outputs[a]
Yellow LEDs at X4, X5 and X6	Status of the inputs and outputs
Green LEDs at X6, pin 9 and pin 10	24 V power supply of the outputs

a. An output is reset when a short circuit occurs. The short-circuit monitoring remains in effect, and must therefore be reset with Module 9139.
In order to recognize a short circuit, a current of 20 A must be able to flow for approximately 3 ms. If this is not the case (e.g., the 24-V supply limits the current sooner), the short-circuit monitoring might not become effective.

I/O Module and I/O Expansion Base Module P/N Summary

Refer to **Table 2-36**.

Table 2-36, I/O Module and I/O Expansion Base Module P/N Summary

P/N	Description	
624498-01	I/O EXP BASE MODULE, 4-SLOTS	IEB 404
624500-01	I/O EXP BASE MODULE, 6-SLOTS	IEB 406
624501-01	I/O EXP BASE MODULE, 8-SLOTS	IEB 408
624505-01	I/O MODULE, DIGITAL 16/8	IEM 16-8D
624506-01	I/O MODULE, ANALOG 4/4	IEM 4-4A
624507-01	I/O MODULE, BLANK	

Handwheel Input

One remote RM 500 handwheel or one panel-mounted PM 300 handwheel can be used with a CC 600 Series CNC. Refer to RM 500 parameters for configuration. Refer to **Table 2-37**.

The following topics are described:

- **X23: Handwheel Input**
- **RM 500 Remote Handwheel**
- **Connection Guidelines**
- **Operation Guidelines**
- **PM 300 Panel-Mounted Handwheel**
- **PM 350 Panel-Mounted Handwheel**

X23: Handwheel Input

Table 2-37, X23: Handwheel Input - Pinout

D-Sub Connection (Female) 9-Pin	Assignment
2	0 V
4	+12 V ± 0.6 V (Uv)
6	DTR
7	TxD
8	RxD
9	DSR
1, 3, 5	Not assigned
Housing	External shield

This interface meets requirements per IEC 742 EN 50 178 for low voltage electrical separation.

RM 500 Remote Handwheel

Refer to **Table 2-37, RM 500 Remote Handwheel - Pinout**, **Figure 2-15, RM 500 Internal Wiring of Contacts to Control Buttons and E-STOP**, and **Figure 9-20, RM 500 Remote Handwheel, P/N 34000850**. The RM 500 is a portable electronic handwheel that offers the following features:

- Keys for selection of five axes for handwheel or jogging:
  - X – X-axis
  - Y – Y-axis
  - Z – Z-axis
  - IV – U-axis
  - V – Not used
- Jog traverse direction keys: Jog+ and Jog–
- Jog and handwheel resolution setting. For jogging, the 1, 10, 100 corresponds to 1, 10, 100 times system resolution. For handwheel, the 1, 10, 100 corresponds to pre-set feedrates – Slow, Medium, or Fast
- Actual-position-capture key (not used)
- Spindle control keys – Counter-clockwise (CCW), Stop, and Clockwise (CW)
- Two permissive buttons – Must be pressed to allow operation of Remote Handwheel
- EMERGENCY STOP button
- Holding magnets
- Dummy plug for EMERGENCY STOP circuit (optional), P/N 34000865

Connection Guidelines

Connection layout for the various cables and the handwheel is provided in **Table 2-37, RM 500 Remote Handwheel - Pinout**. The EMERGENCY STOP plug must be connected if the handwheel is not connected. Connect adapter cable or the extension cable (if used) to X23 on the MC 400.

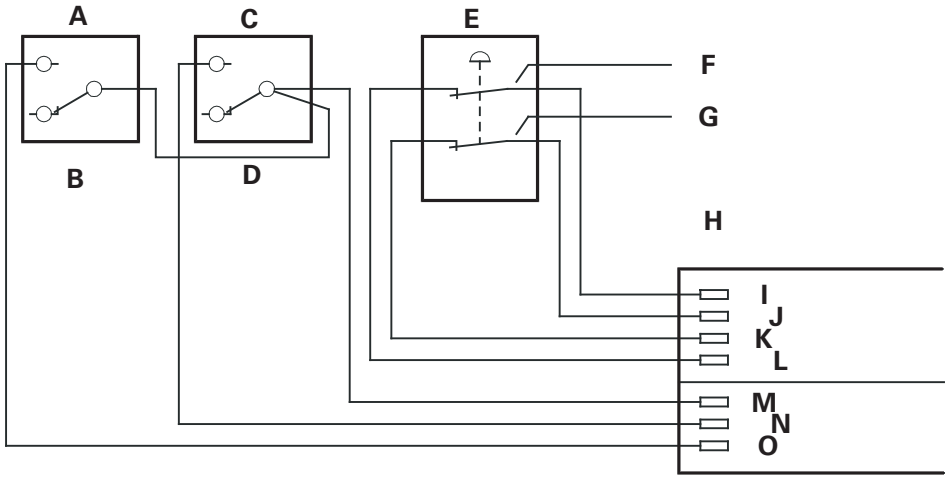


Figure 2-15, RM 500 Internal Wiring of Contacts to Control Buttons and E-STOP

Figure 2-15 callouts:

A – Permissive button 1
B – Right
C – Permissive button 2
D – Left
E – Emergency STOP
F – Contact 2
G – Contact 1
H – Cable adapter
I – Contact 2
J – Contact 1
K – Contact 1
L – Contact 2
M – Contact 1+2
N – Contact 2
O – Contact 1

The adapter cable (P/N 343000415) includes terminal strips for the EMERGENCY STOP contacts and permissive buttons (max load 1.2 A). See **Figure 2-15**.

The terminals are supplied with the adapter cable. If you need replacement terminals, these can be ordered from ANILAM:

Terminal strip 3-pin:	P/N 80900155
Terminal strip 4-pin:	P/N 80900136

Table 2-37, RM 500 Remote Handwheel - Pinout

Extension Cable P/N 343003*XX*			Adapter Cable P/N 34300415			Connecting Cable P/N 34000810			RM 500 P/N 34000850	
D-Sub Connctr. (Male) 9-Pin		**D-Sub Connctr. (Female) 9-Pin**	**D-Sub Connctr. (Male) 9-Pin**		**Coupling on Mtg. Base (Female) 18-Pin**	**Connctr. (Male) 18-Pin**	**Color(s)**	**Connctr. (Female) 18-Pin**	**Connctr. (Male) 18-Pin**	
Housing	Shield	Housing	Housing	Shield	Housing	Housing	Shield	Housing	Housing	Shield
2	White	2	2	White	E	E	White	E	E	
4	Brown	4	4	Brown	D	D	Brown	D	D	
6	Yellow	6	6	Yellow	B	B	Yellow	B	B	
7	Gray	7	7	Gray	A	A	Gray	A	A	
8	Green	8	8	Green	C	C	Green	C	C	
				A	6	6	WH/BK	6	6	
				B	7	7	YL/BK	7	7	
				C	5	5	WH/RD	5	5	
				D	4	4	WH/BL	4	4	
				E	2	2	WH/GN	2	2	
				F	3	3	WH/YL	3	3	
				G	1	1	WH/BN	1	1	

G	WH/BN	3	Contact 1 + 2	
F	WH/YL	2	Contact 2 (left)	Control button
E	WH/GN	1	Contact 1 (right)	
D	WH/BL	1	Contact1	
C	WH/RD	2	Contact1	EMERGENCY STOP
B	YL/BK	3	Contact2	
A	WH/BK	4	Contact2	

Operation Guidelines

The permissive buttons must be pressed for any of the features on the remote handwheel to work. The remote handwheel can, essentially, only be used in Manual mode, and, in general, duplicates the equivalent functionality in the Manual Panel. For details on Manual mode operation of the CNC refer to *6000i CNC User's Manual*, P/N 627785-2X.

PM 300 Panel-Mounted Handwheel

The PM 300, P/N 34000855, does not have axis keys, rapid traverse keys, etc. It is connected to MC 400 directly or via extension cable. Refer to **Table 2-38**. Refer to **Figure 9-21, PM 300 Panel-Mounted Handwheel, P/N 34000855**.

The PM 300 is available with standard 3 ft. cable.

For space requirements, refer to **Figure 9-2, CNC, CC, and Inverter**.

Table 2-38, PM 300 Panel-Mounted Handwheel - Pinout

Extension Cable P/N 343003*XX*			PM 300 P/N 34000855	
D-Sub Connector (Male) 9-Pin		**D-Sub Connector (Female) 9-Pin**	**D-Sub Connector (Male) 9-Pin**	
Housing	Shield	Housing	Housing	Shield
2	White	2	2	White
4	Brown	4	4	Brown
6	Yellow	6	6	Yellow
8	Green	8	8	Green
7	Gray	7		

PM 350 Panel-Mounted Handwheel

With the PMA 310 (P/N 34000870) handwheel adapter, you can connect two or three PM 350 (P/N 34000875) handwheels to the 6000M.

The first two handwheels are assigned to axes X and Y (connected to inputs X1 and X2 of the PMA 310). The third handwheel (X3 input) can be assigned as shown in **Table 2-39**.

Table 2-39, PM 350 Third Handwheel Assignments

MC_4102	Switch Position	Third Handwheel
X Fixed	Any switch position	X-axis
Y Fixed	Any switch position	Y-axis
Z Fixed	Any switch position	Z-axis
MP Switch	0 (MP switch at the left stop - i.e., all the way to the left)	X-axis
	1	Y-axis
	2	Z-axis
	3	U-axis
	4	W-axis
Third Handwheel Adapter	0 (S2 at the left stop)	Z-axis
	1	U-axis
	2	W-axis
	3	Z-axis
	4	Z-axis

An additional switch enables you to select the interpolation factor for the handwheel as shown in **Table 2-40**.

Table 2-40, PM 350 Interpolation Assignments

MC_4101	Switch Position	Resolution
Fixed 1	Any switch position	1
Fixed 10	Any switch position	10
Fixed 100	Any switch position	100
MP Switch	0 (MP switch at the left stop)	1
	1	10
	2	100
	3	100
	4	100
Third Handwheel Adapter	0 (S1 at the left stop)	1
	1	10
	2	100
	3	100
	4	100

Refer to **Figure 2-16, PM 350 and PMA 310 Internal Wiring**, **Table 2-41, X1, X2, X3: Inputs for PM 350 Handwheels - Pinout**, **Table 2-42, X23: Connection to MC 400 - Pinout**, and **Table 2-43, X31: Supply Voltage - Pinout**.

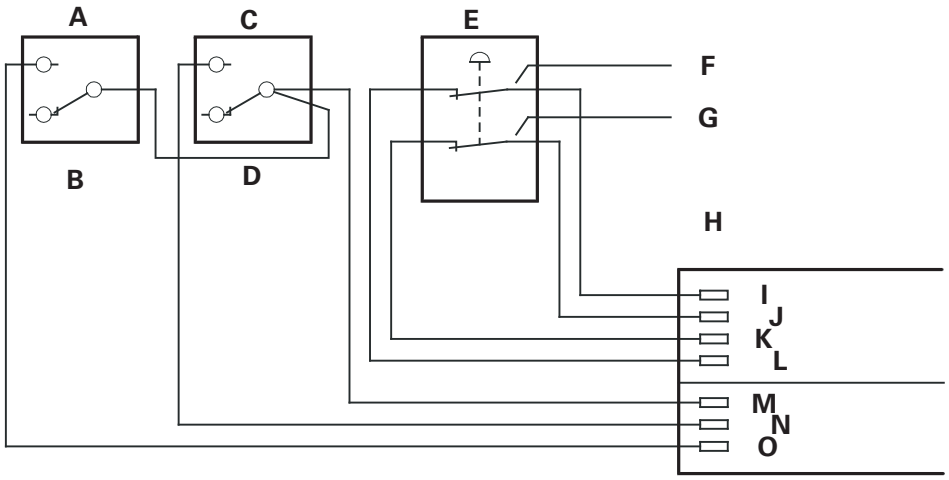


Figure 2-16, PM 350 and PMA 310 Internal Wiring

Figure 2-16 callouts:

A – PM 350, P/N 34000875

B – PMA 310, P/N 34000870

C – Axis Selection, P/N 3400088*X*,
where *X*= 1 (3'), 2 (15'), or 3 (60'). (Optional)

D – Interpolation Factor Selector Knob, P/N 3400088*X*,
where *X*= 1 (3'), 2 (15'), or 3 (60'). (Optional)

E – Maximum 60 feet (20 m). The connecting cable for PM 350 to PMA 310 is P/N 3400089*X*, where *X* represents the available cable lengths:
X= 1 (15'), 2 (20'), 3 (30'), or 4 (60').

F – Maximum 60 feet (20 m). The connecting cable for PMA 310 to MC 400 is P/N 343005*XX*, where *XX* represents the available cable lengths:
XX= 05 (5'), 15 (15'), 20 (20'), 25 (25'), 30 (30'), 35 (35'), 40 (40'), 45 (45'), 50 (50'), 55 (55'), or 60 (60').

For the X1, X2, and X3 pin layout on PMA 310 adapter for inputs to the PM 350 handwheel, see **Table 2-41**.

Table 2-41, X1, X2, X3: Inputs for PM 350 Handwheels - Pinout

PMA 310 Adapter	
Connection (Female) 9-Pin	**Assignment**
1	I_1 +
2	I_1 –
3	+5 V
4	0 V
5	I_2 +

PMA 310 Adapter	
Connection (Female) 9-Pin	**Assignment**
6	I_2 –
7	I_0 –
8	I_0 +
9	Internal shield
Housing	External shield

(Continued on adjacent table...)

For the X23 connection to MC 400 pin layout on the PMA 310 adapter, see **Table 2-42**.

Table 2-42, X23: Connection to MC 400 - Pinout

PMA 310 Adapter	
D-Sub Connection (Female) 9-Pin	**Assignment**
1	RTS
2	0 V
3	CTS
4	+12 V +0.6 V (U_V)
5	Do not assign
6	DSR
7	R_XD
8	T_XD
9	DTR
Housing	External shield

For the X31 supply voltage pin layout on the PMA 310 adapter, see **Table 2-43**.

Warning: The power supply of the I/O must not be used simultaneously for the PMA 310 adapter; otherwise, the isolation of the inputs/outputs would be bridged.

Table 2-43, X31: Supply Voltage - Pinout

PMA 310 Adapter	
Terminal	**Assignment**
1	+24 V - as per EN 50 178 (basic insulation)
2	0 V

Console FP 6000i

Console assembly consists of 12.1" color, active matrix, flat-panel display, and alphanumeric keypad. Refer to **Table 2-44** and **Table 2-45**.

Table 2-44, Console FP 6000i

System	Designation	Console
6000i	FP 6000i	P/N 624514-01

Table 2-45, Hardware Connections to MC 400

Hardware	Connects to MC 400
Flat panel	Via connector X49 with cable P/N 343000*XX*
Alphanumeric keypad	Via connector X45 with cable P/N 343001*XX*

Where XX = 10, 15, 20, 25, 30, 35, 40, and 45 feet.

Power to the flat-panel console is delivered to X1, 2-pin, 5.08 mm Phoenix connector. The mating connector is included with the console. Refer to **Table 2-46**.

Table 2-46, Flat Panel X1 Power Connection - Pinout

Power Connection 2-Pin	Description
1	+24 Volts
2	24 V common

For space requirements, refer to **Figure 9-1, Console**.

Manual Panel MP 6000M and MP 6001M

Manual panel assembly includes commonly used machine operating functions. The manual panel connects to the MC 400 via connector X46 using cable PN 343001XX, where XX = 10, 15, 20, 25, 30, 35, 40, and 45 feet. Two versions of manual panel assemblies are available:

MP 6001M, Manual panel with Manual Pulse Generator (MPG) (P/N 34000701). Use MPG extension cable 343005XX to connect the MPG to the MC 400 using connector X23.

MP 6000M, Manual panel without MPG (34000704)

For space requirements, refer to **Figure 9-3, MP 6000M Manual Panel** and **Figure 9-4, MP 6001M Manual Panel**.

The Manual Panel interface, P4, is an 8-pin, 3.81 mm Phoenix connector. There are extra relay level I/O for special customer functions. Refer to **Table 2-47**.

Table 2-47, P4 Manual Panel Interface - Pinout

Interface Connection 8-pin	Description	Assignment
1	+24 Volts	
2	24 V common	
3	Spare input 1	I:150
4	Spare input 2	I:151
5	Spare input 3	I:152
6	Reserved	
7	Reserved	
8	Reserved	

The Manual Panel interface, P5, is a 2-pin, 3.81 mm Phoenix connector, Servo Reset Key contact. The mating connector is included in the Manual Panel. The contact is rated 1 Amp, 50 VDC. Also, the contact is isolated from logic common and relay common. Refer to **Table 2-48**.

For details on wiring the Servo Reset signal, refer to **Figure 9-19, Basic Servo Turn On Circuit**.

Table 2-48, P5 Manual Panel Interface - Pinout

Interface Connection 2-pin	Description
1	Form A, N/O contact
2	Form A, N/O contact

Section 3 - Machine Parameters

The following topics are described in this section:

- **General Information**
- **The Configuration Editor**
- **Allocation of Configuration Data**
- **Setup of a Parameter File**
- **MP Subfiles**
- **Read or Change Machine Parameters via a PLC Module**
- **Overview of the Machine Parameters of the 6000i**

General Information

A control must have access to specific machine data (e.g., traverse distances, acceleration, speeds) before it can execute its programmed instructions. You define these data in machine parameters. Each machine has its own set of parameters.

The parameter values are entered in the **configuration editor**. The machine parameters are grouped as parameter objects in a tree structure. They are saved in **parameter files** with the extension **.cfg** on the Config directories on the drives: **TNC:\Config** and **PLC:\Config**.
These drives should always be addressed with the system variables %OEM% (PLC:\ ) and %USR% (TNC:\ ).

Note: You should only make changes to the parameter files by using the configuration editor.
Only in exceptional cases should the *.cfg files be edited directly. This could possibly lead to faulty syntax, which would prevent the control from starting up.

Each parameter object (also called data object or object) has a name that gives information about the parameters it contains.
Depending on the function, the parameters are differentiated into system-specific, channel-specific, and axis-specific types.
Each object has a “key” for unique identification. The keyname can have a maximum of 18 characters.

The following applies:

- **System data** (parameters that are valid for the entire system) only occur once. The configuration editor does not require a keyname for these parameters, nor is one entered. These objects are identified with an “empty” key in the *.cfg files.
- **Parameter objects that apply to axes** occur more than once.
  A unique keyname is assigned to each axis. All objects that apply to a certain axis must be identified with this key.
  Example: The keyname “Axis-X” for all objects that belong to the X axis
  The keyname “Axis-S” for all objects that belong to the spindle

- **Parameter objects that apply to channels** occur more than once.
  A unique keyname is assigned to each channel. All objects that apply to a certain channel must be identified with this key.
  Example: The keyname “Channel1” for all objects that belong to Channel1

Note: Keynames should be short and clear, and refer to the function.

The Configuration Editor

The following topics are described:

- **Calling the Configuration Editor**
- **Machine Parameter Screen**
- **Entering and Editing Machine Parameters**
- **Managing Configuration Files**
- **Sort File Content**
- **Table View**
- **Access Rights**
- **Update Rules**
- **Remove Syntax Error**
- **Reset Update Version**
- **Backup of Parameters**

Calling the Configuration Editor

Call the main menu (see figure below) of the configuration editor as described below:

On the Manual screen, press (**SHIFT** + **F3**).

In the Full Access Password: field, type the password and press **ENTER** to display the Configuration screen. Refer to **Figure 3-1**.

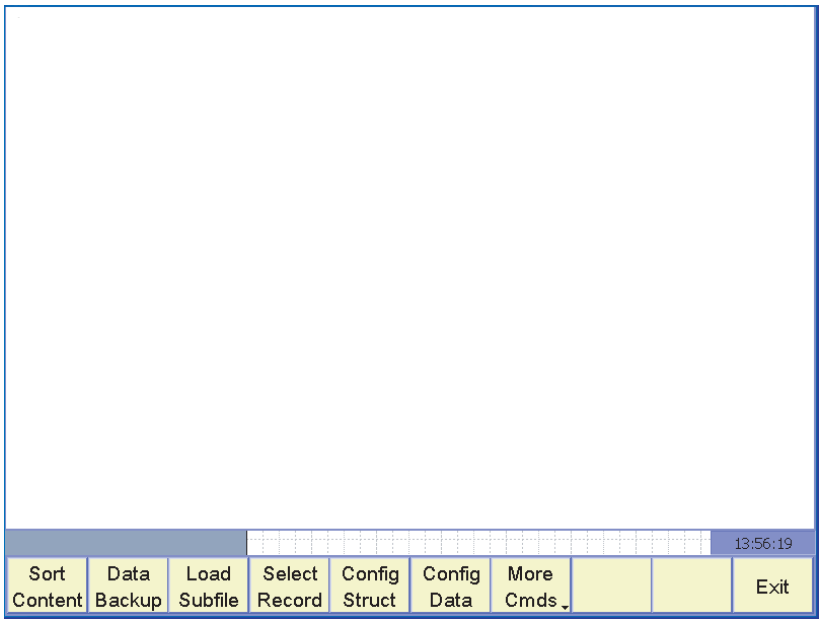


Figure 3-1, Configuration Screen

- After initially openning the Configuration Screen with (**SHIFT** + **F3**) and enter the password, it is not necessary to enter the password again unless the CNC is rebooted. To open this screen again, press (**SHIFT** + **F3**) from Manual.

Meaning of the soft keys on **Figure 3-1**.

Label	Soft Key	Description
Sort Content	**F1**	Sorts the contents of the *.cfg files
Data Backup	**F2**	Backs up the machine parameters
Load Subfile	**F3**	Activates MP subfiles
Select Record	**F4**	Selects various data records
Config Struct	**F5**	Path information for the *.cfg files of the machine configuration
Config Data	**F6**	Opens the configuration editor for editing the machine parameters in tree or table view. Press **Config Data** (**F6**) to display the Machine Parameter screen. Refer to **Figure 3-3, Machine Parameter Screen**.
More Cmds	**F7**	Calls additional information. Refer to **Figure 3-2, More Cmds (F7) Screen from Configuration Screen**.
Exit	**F10**	Exit the Configuration Screen, save changes, and return to the Manual screen.

Press **More Cmds** (**F7**) to display **Figure 3-2** soft keys.

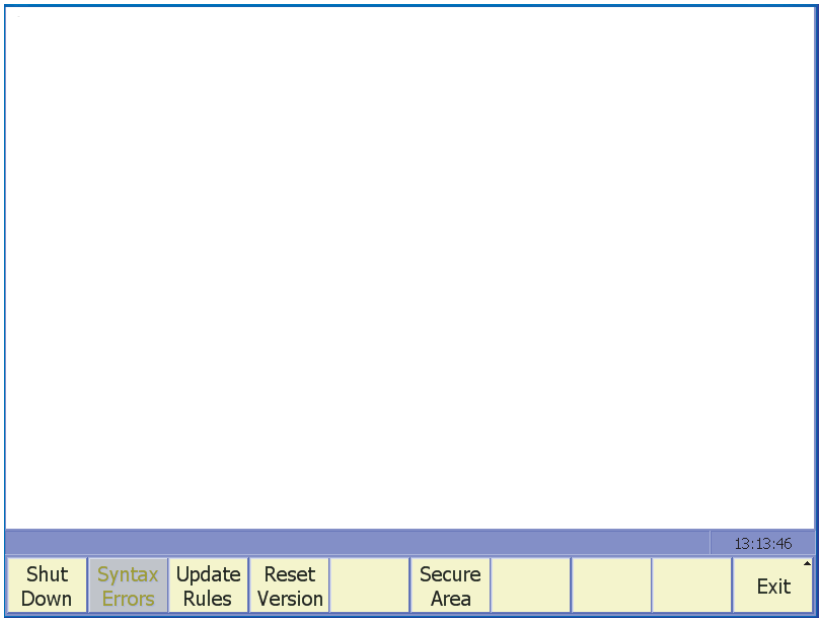


Figure 3-2, More Cmds (F7) Screen from Configuration Screen

Meaning of the soft keys on **Figure 3-2**.

Label	Soft Key	Description
Shut Down	**F1**	Shuts down/ restarts the CNC
Syntax Errors	**F2**	Grayed out during normal operations. Only active after a software update or with faulty configuration data. Used for finding and fixing errors in the machine configuration.
Update Rules	**F3**	Display and edit rules of the software exchange
Reset Version	**F4**	Resets update version
Secure Area	**F6**	Displays access rights, selection lists, limit values, and units of measurement
Exit	**F10**	Exit the **More Cmds** (**F7**) screen, save changes, and return to the Manual screen.

Machine Parameter Screen

Press **Config Data** (**F6**) to display the Machine Parameter screen. Refer to **Figure 3-3**.

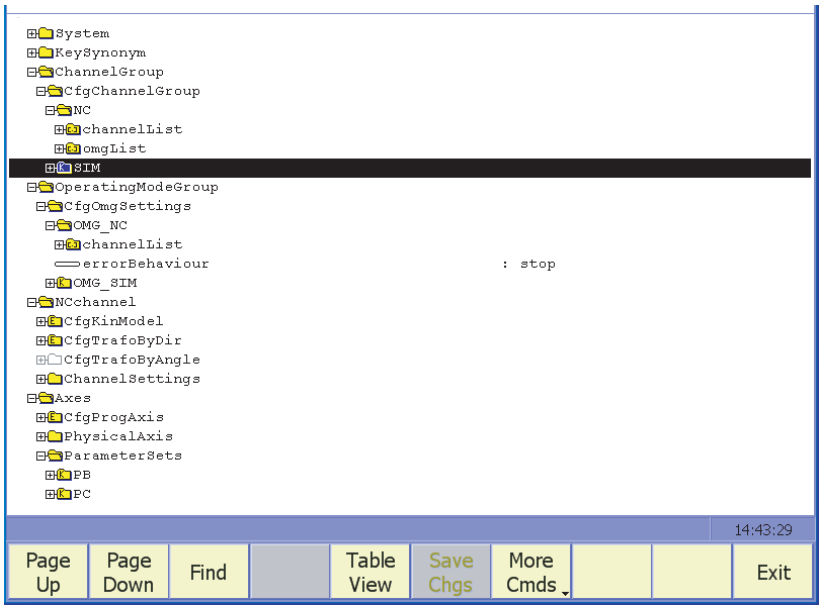


Figure 3-3, Machine Parameter Screen

- On the Machine Parameter Screen, press **More Cmds** (**F7**) to display a More Cmds screen. Refer to **Figure 3-4, More Cmds (F7) Screen from Machine Parameter Screen**.

Meaning of the soft keys in **Figure 3-3**:

Label	Soft Key	Description
Page Up	**F1**	Pages up the Machine Parameters screen
Page Down	**F2**	Pages down the Machine Parameters screen
Find	**F3**	Opens the Find screen. Refer to **Figure 3-8, Find Dialog Box**.
Help	**F4**	Opens the Help screen. Refer to **Figure 3-5, Help Screen**.
Table View	**F5**	Enables you to see the machine parameters for all axes together in a table view. Refer to **Figure 3-11, Tree View (F5) Screen**.
Save Chgs	**F6**	Save the changes
More Cmds	**F7**	Opens the More Cmds screen. Refer to **Figure 3-4, More Cmds (F7) Screen from Machine Parameter Screen**.
Exit	**F10**	Exit the Machine Parameter Screen, save changes, and return to the previous screen.

The following topic is described:

- **More Commands from Machine Parameter Screen**

More Commands from Machine Parameter Screen

Press the **More Cmds** (**F7**) to display the following screen.

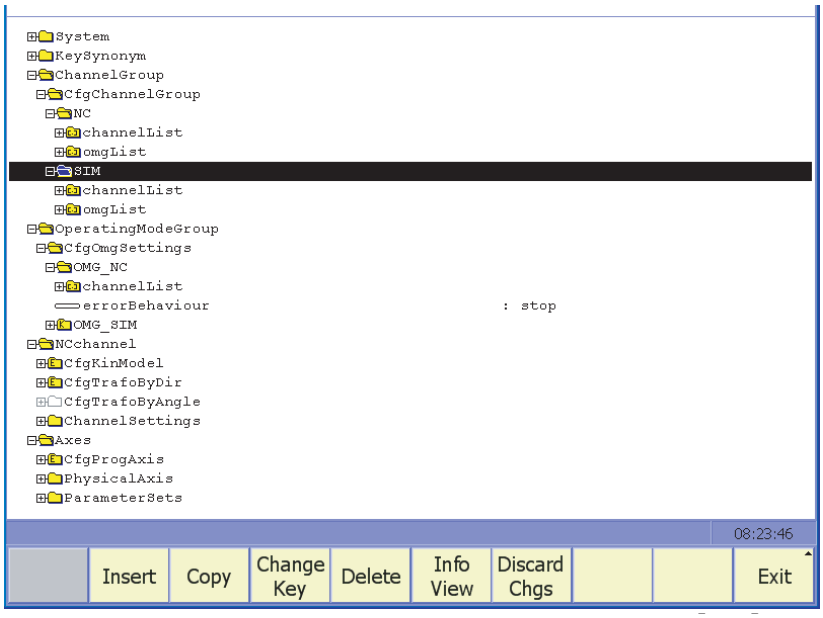


Figure 3-4, More Cmds (F7) Screen from Machine Parameter Screen

Meaning of the soft keys on **Figure 3-4**.

Label	Soft Key	Description
Insert	**F2**	Insert objects in lists (an axis, for example). Refer to "**Inserting and Copying Objects**."
Copy	**F3**	Copy objects in lists (an axis, for example). Refer to "**Inserting and Copying Objects**."
Change Key	**F4**	Change the keyname of an object. Refer to "**Changing Keynames**."
Delete	**F5**	Delete objects or parameters from a list (an axis, for example). Refer to "Deleting objects."
Info View	**F6**	Display the Info View screen (refer to **Figure 3-9, Info View Screen**)."
Discard Chgs	**F7**	Opens the change window gives you an overview of all changed parameters to determine if you want to keep or discard changes. Refer to "**MP Change List in the Configuration Editor**."
Exit	**F10**	Exit the Configuration Screen, save changes, and return to the Manual screen.

Entering and Editing Machine Parameters

After pressing the **Config Data** (**F6**) soft key, the object tree for the machine parameters is displayed. Refer to **Figure 3-3, Machine Parameter Screen**.

The actual machine parameters with their values are located on the lowest level of this tree.

The cursor is positioned within the tree by using the **ARROW** keys.

To open a branch: Press the + key, **ENTER** key, or the **RIGHT ARROW** key.

To close a branch: Press the – key, **ENTER** key, or the **LEFT ARROW** key.

The following topics are described:

- **Icons in the Object Tree**
- **Displaying Help Texts**
- **Entering and Editing Parameters**
- **Limit Values**
- **Deleting Objects**
- **Inserting and Copying Objects**
- **Changing Keynames**
- **Saving Input Values**
- **Find Function**
- **User Commenting**
- **Finish Editing**

Icons in the Object Tree

An icon at the beginning of each line in the parameter tree shows additional information about this line. The icons have the following meanings:

-  Branch exists but is closed
-  Branch is open
-  Empty object, cannot be opened
-  Initialized machine parameter
-  Uninitialized (optional) machine parameter
-  Can be read but not edited
-  Cannot be read or edited
-  Entity (object)
-  Array (list)
-  Key (keyname)

Displaying Help Texts

The **Help** (**F4**) key enables you to call a help text for each parameter object or attribute.

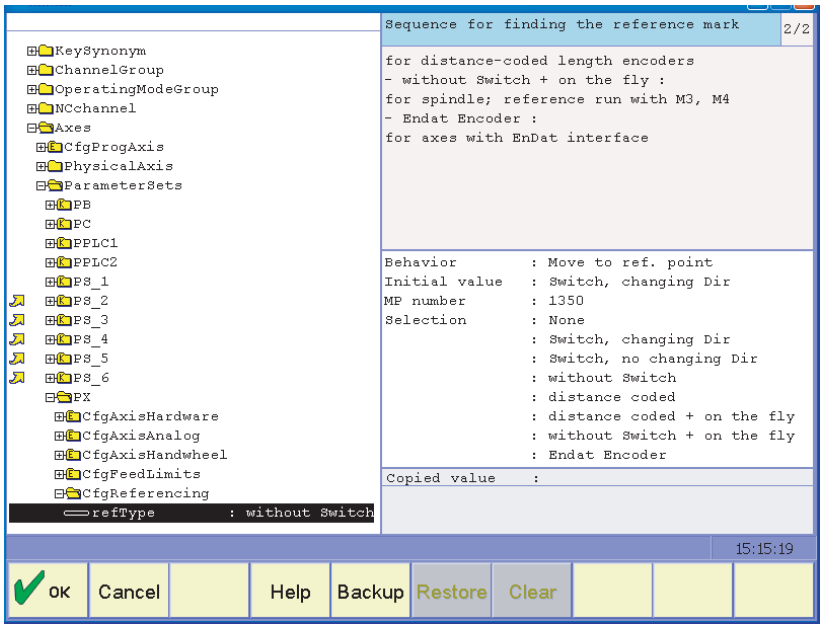


Figure 3-5, Help Screen

If the help text does not fit on one page (1/2 is then displayed at the upper right, for example), press the **Help** (**F4**) soft key to scroll to the second page.

Additional information, such as the unit of measure, the initial value, or a selection list, is also displayed. If the selected machine parameter matches a parameter in the 6000i, the corresponding MP number is shown.

To exit the help text, press the **Help** (**F4**) soft key again.

Entering and Editing Parameters

In order to change the input values, open an input or selection dialog box for the selected parameter by pressing the **ENTER** key.

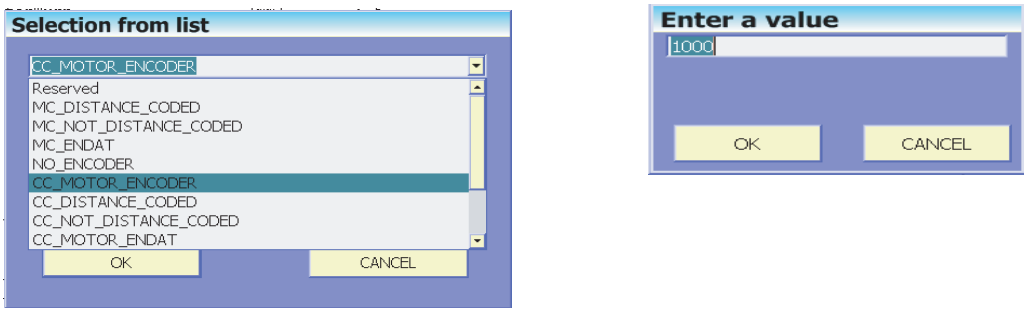


Figure 3-6, Selection from list and Enter a value Dialog Boxes

In selection dialogs, press the **UP/DOWN ARROW** key or the **ENTER** key to select the desired value from the list.

Units of measurement can be defined for numeric machine parameters. The unit of measurement assigned to this parameter is displayed. Enter a value appropriate to this unit.

Limit Values

Limit values are displayed for numeric machine parameters. If you attempt to enter a value outside of these limits, a message is issued and the entry is not accepted.

Deleting Objects

Press the **More Cmds** (**F7**) soft key then the **Delta** soft key to delete objects or parameters from a list (an axis, for example).

Inserting and Copying Objects

Press the **More Cmds** (**F7**) soft key then the **Insert** (**F2**) or **Copy** (**F3**) soft key to insert or copy objects or items in lists (an axis, for example). Items in lists (arrays) are inserted after the cursor. When inserting an object, the object name (keyname) and memory file must be given. The memory file is the *.cfg file in which the inserted object is to be saved. Press the **UP/DOWN ARROW** key or the **ENTER** key to select the file.

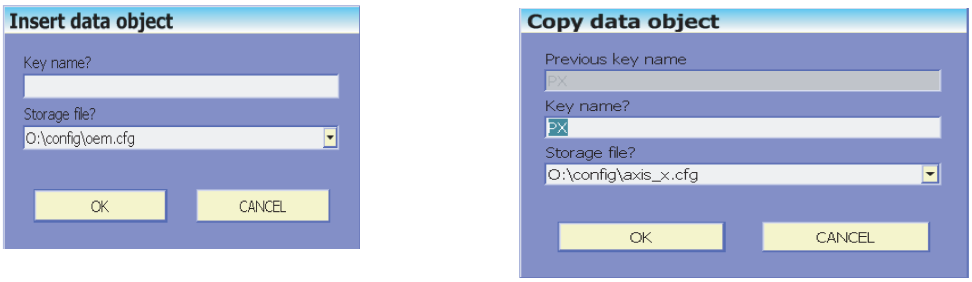
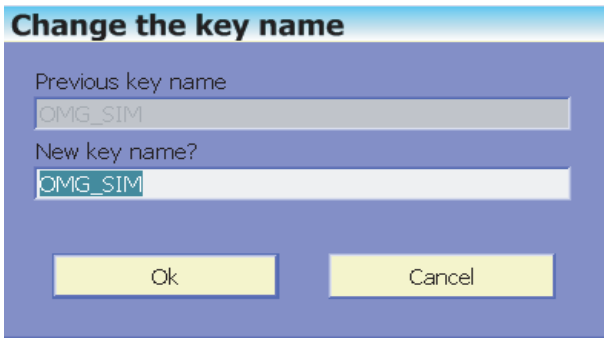


Figure 3-7, Insert and Copy data object Dialog Boxes

Empty objects, lists and parameters are shown with a gray icon. They can be activated with the **Insert** (**F2**) soft key.

Changing Keynames

Press the **More Cmds** (**F7**) soft key then the **Change Key** (**F4**) soft key to display the **Change the key name** dialog box to change the keyname of an object, for example, from Kinem1 to Kinem_XYZ.



Saving Input Values

The input values are buffered with the **OK** soft key. The **Cancel** soft key closes the dialog box without buffering the value.

Press the **Save Chgs** (**F6**) soft key for the value to take effect.

Certain data cannot be stored while an NC program is running. The message "**Cannot change parameter during program run**" is displayed. In this case, the program must be stopped and exited. Then, the data can be saved.

Some data are transferred directly. Others require that the axes be referenced again, or that the system be restarted. This is indicated in a corresponding message.

Find Function

Search for objects and parameters within the configuration editor using a dialog box. Open the dialog box with the **Find** (**F3**) soft key.

Type the string or number that you want to find in the top line. On the second line, use the direction up/down on the far-right to select between **Search forward** and **Search backward** (see Figure below).

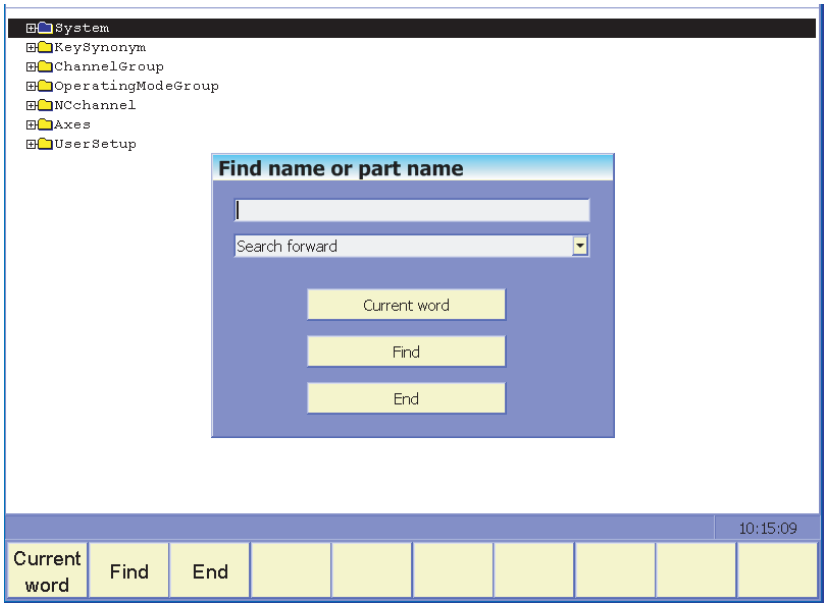


Figure 3-8, Find Dialog Box

The current word or a part of the object or parameter name may be entered as the search term. The search term can be written in large or small letters.

Soft Keys on Find Dialog Box

Meaning of the soft keys in **Figure 3-8**:

Current word	Searches for the current word in the top line
Find	Starts the search for the current word in the top line
End	Exit the Find pop-up screen and return to the Machine Parameter screen

User Commenting

You can comment objects. Press the **More Cmds** (**F7**) soft key then the **Info View** (**F6**) soft key to -. Type the comment in the Comment Area (refer to **Figure 3-9**). A maximum of four comment lines can be entered.

Objects with comments are displayed on the right side of the parameter tree and are identified with the letter “i.” The complete text, including the help text for the object, displays on the Info View screen.

The current value of a parameter can be buffered together with the comment and can, for example, be reactivated later.

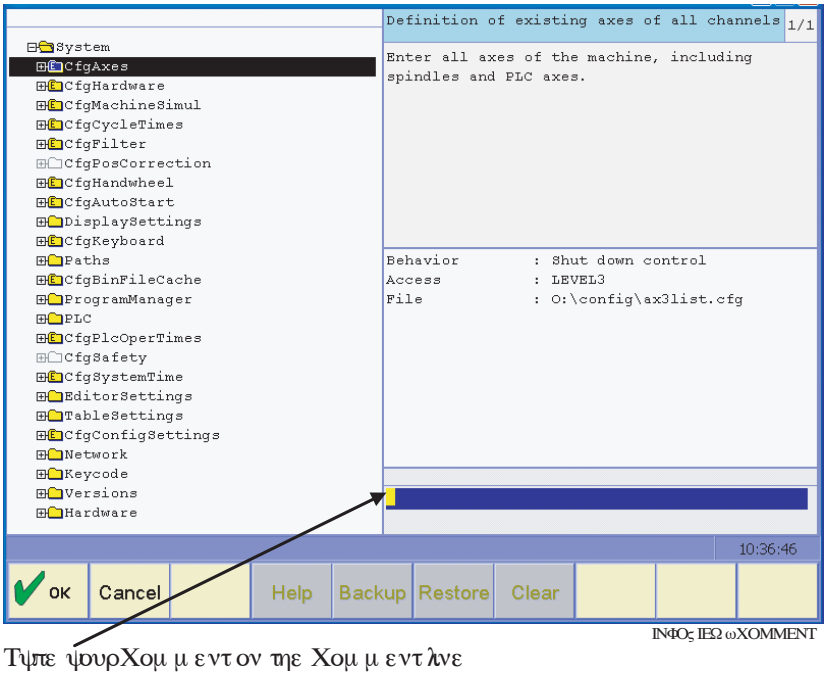


Figure 3-9, Info View Screen

Finish Editing

Press the **OK** (**F1**) or **Cancel** (**F2**) soft key to return to the main menu.
The control asks if the changed data are to be saved or discarded (see Saving input values).

Managing Configuration Files

The configuration data is saved in several files with the extension .cfg. This enables you to establish the correct configuration for different types of machines by selecting the appropriate files from the paths entered.

There are two types of configuration file lists: ANILAM files and OEM files. The ANILAM files are permanently defined and cannot be changed (e.g., **CfgJhConfigDataFiles**).

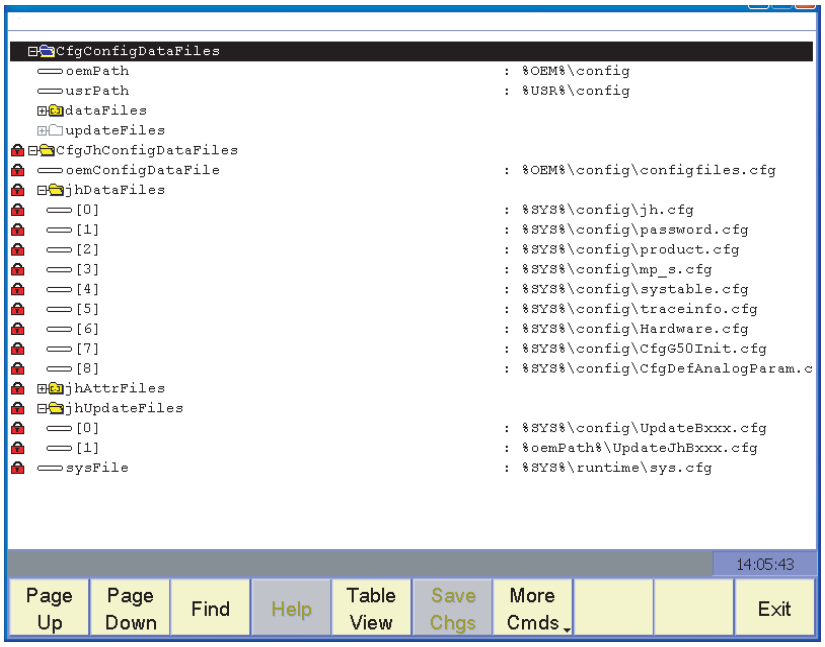


Figure 3-10, Config Struct Screen

The paths and names of OEM files can be changed with the **Config Struct** (**F5**) soft key. New configuration files can also be added (for a new axis, for example). The paths are saved in the file configfiles.cfg.

The paths and names of the configuration files are stored in the **dataFiles** list (see "**Allocation of Configuration Data**"). The control searches for the parameter objects and their parameters in these *.cfg files.

The paths of these files can be changed. Use the **RIGHT ARROW** key to open a dialog box for entering the new path or file name.

Sort File Content

Pressing the **Sort Content** (**F1**) soft key sorts the contents of the *.cfg configuration files so that the objects are in the same order as they are listed in the configuration editor.

However, since the data in the configuration editor comes from multiple files, there is no direct correlation between the display in the configuration editor and the contents of each file.
This sorting makes it easier to compare the contents of similar files (such as *.cfg for axes), since the entries are now in the same order.

Table View

In the configuration editor, you can new switch from the familiar tree view to a new table view. This is especially useful when configuring the parameter blocks, since now the parameters of all axes are visible at a glance.

All editing functions available in the tree view are also available in the table view.

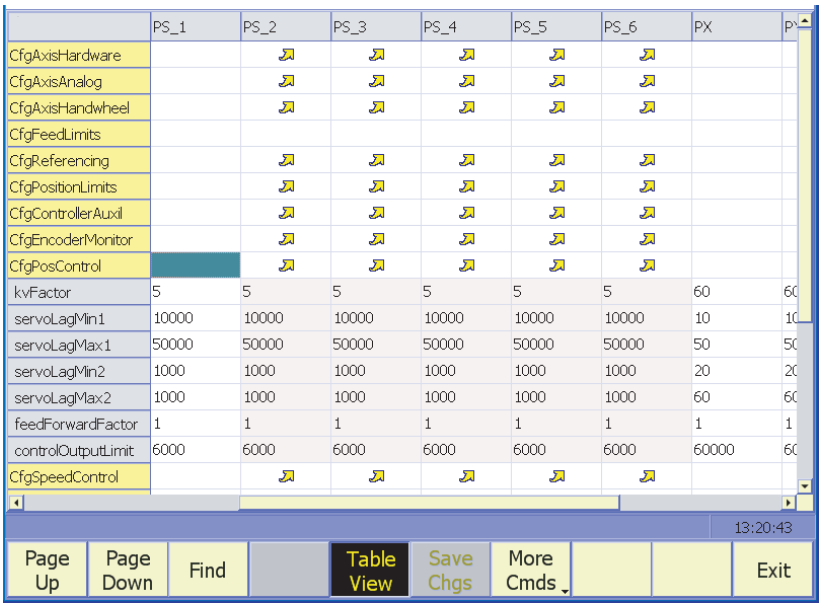


Figure 3-11, Tree View (F5) Screen

Access Rights

Press the **More Cmds** (**F7**) soft key then the **Secure Area** (**F6**) soft key for access rights. Refer to **Figure 3-2, More Cmds (F7) Screen from Configuration Screen**. Entering a code number also grants access rights to the machine parameters.
There is a difference between the four levels, from Level1 to Level4. Level1 grants few rights, whereas Level4 grants all rights.

LEVEL1 access right
Machine parameters on LEVEL1 can be reached and edited using the access password.

LEVEL2 access right
Machine parameters on LEVEL2 can be reached and changed with the access password.

LEVEL3 access right
Machine parameters on LEVEL3 can be accessed by ANILAM. The machine manufacturer can only read them.

LEVEL4 access right
Machine parameters on LEVEL4 can only be accessed by ANILAM**.** The machine manufacturer can only read them.

The following topic is described:

- **Reaction to Change**

Reaction to Change

The following reactions can occur when machine parameters are changed:

- NOTHING
- RUN
- RESET
- REF

Information on which reactions occur for which machine parameters is given later in this Section.

Reaction NOTHING
Data with this reaction can be changed at any time, including during program run.

Reaction RUN
Changes are only possible during a PLC strobe or NC stop.

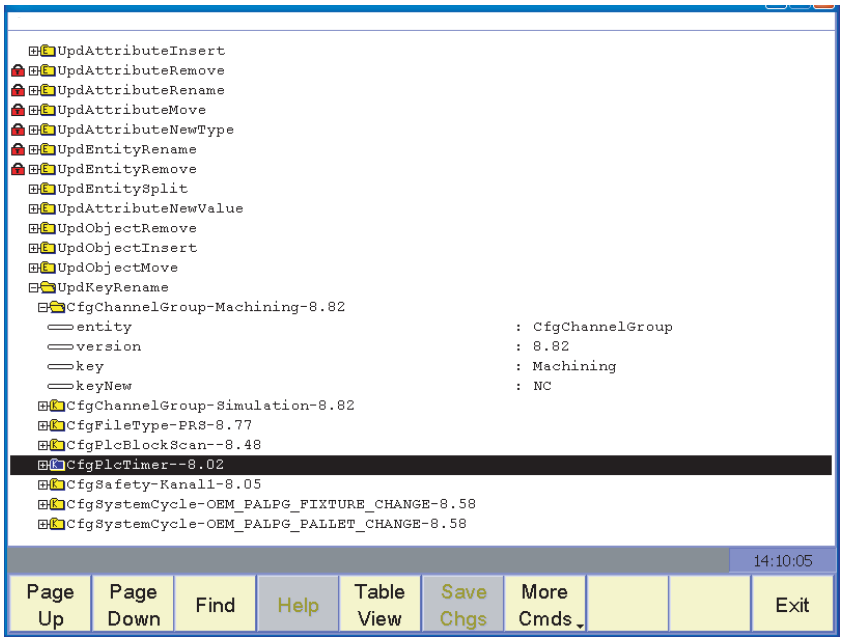
Reaction RESET
After a machine parameter to which the RESET reaction is assigned has been changed, the error message **Machine parameters were changed. Shut down and restart the control** is displayed.
This message cannot be cleared. The machine must be restarted. The control can also be restarted later so that you can make further changes in the configuration editor.
Data objects with this reaction must not be changed during program run.

Reaction REF
After a machine parameter connected to the REF reaction has been changed, the affected axis is set to unreferenced.

A new reference run must be made for this axis.
Data objects with this reaction must not be changed during program run.

Update Rules

Press the **More Cmds** (**F7**) soft key then the **Update Rules** (**F3**) soft key to define rules that are required for updating the machine parameters during a software update.



When new objects are added, or existing ones deleted, the OEM must update the reference to the parameter file here. These rules no longer need to be followed after the software has been exchanged.

Remove Syntax Error

Press the **More Cmds** (**F7**) soft key then the **Syntax Errors** (**F2**) soft key to display syntax errors. The **Syntax Errors** (**F2**) soft key becomes selectable when the configuration data in the *.cfg files is changed manually, or when faulty or incomplete update rules are used during a software update. Pressing it opens the faulty file as well as a text editor so that the file can be corrected manually.

Since this soft key can only be selected in one of the above cases, and these cases do not occur during normal operation, the soft key cannot be selected during normal operation.

The start-up of the control is interrupted if a faulty file is detected. The dialog box for entering a code number is displayed. You must enter the OEM or ANILAM code number for the configuration editor. The main menu of the configuration editor is displayed. The **Syntax Errors** (**F2**) and **END** soft keys can be selected. Pressing the **END** soft key continues start-up. However, this will lead to many error messages, since only faulty or no configuration data is available.

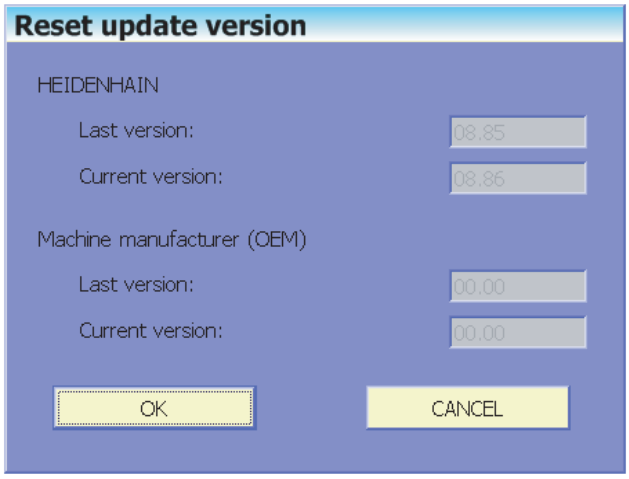
The **END** soft key saves and reloads the file. This can take a moment. If there are still errors, the soft key remains active.

Otherwise, the **Config Data** (**F6**) soft key becomes selectable. You can use it for further corrections in the configuration editor. If the data is now correct, the **END** soft key in the main menu of the configuration editor will continue start-up.

If any other errors are reported, they must be fixed with the configuration editor.

Reset Update Version

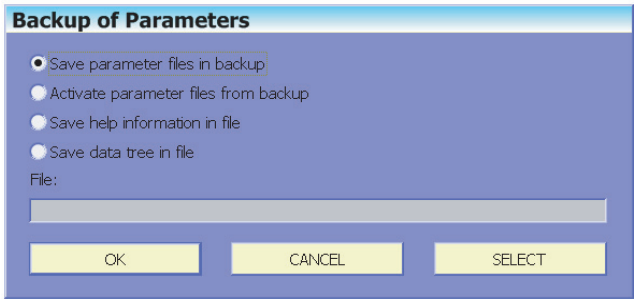
Press the **More Cmds** (**F7**) soft key then the **Reset Version** (**F4**) soft key to enable you to return to the previous software version of machine parameters (configuration data).



If, after a software update, configuration errors occur while the control is starting up, the previous executable version can be reactivated. Then find and correct the error by using the update rules.

Backup of Parameters

The **Data Backup** (**F2**) soft key (refer to **Figure 3-1, Configuration Screen**) enables you to save and restore configuration data as well as to create text files with the current machine parameters:



The following functions are available:

- **Save parameter files in backup**
  The following file name is suggested:
  %OEM%\service\BKUPyear-month-day_.ZIP
  Append meaningful information to this name, for example, the control model, software version, etc.
  All active *.cfg configuration files from %OEM% and %USR% are saved in the selected backup file (e.g., BKUP2005-04-05_tnc320-sw123.ZIP); see %OEM%\config\Configfiles.cfg:
  - All files from %OEM% in the "_Oem_Config_Files_.zip" file
  - All files from %USR% in the "_Usr_Config_Files_.zip" file
  The update files listed in Configfiles.cfg under updateFiles:= are not saved in the backup file.
- **Activate parameter files from backup**
  The *.cfg configuration files from %OEM% and %USR% are retrieved from the selected backup file and activated.
- **Save help information in file**
  The following file name (to be supplemented) is suggested:
  %OEM%\service\HELPyear-month-day_.TXT
  The created text file with the selected name contains the help information about all of the parameter objects and attributes. If a parameter exists more than once, for example several axes, only the information about the first parameter is saved.
- **Save data tree in file**
  The following file name (to be supplemented) is suggested:
  %OEM%\service\TREEyear-month-day_.TXT
  The created text file with the selected name contains the current values of all parameter objects and attributes.

Note: You can also use the PC software TNCbackup for backing up your data in an easy and convenient way. TNCbackup is part of the TNCremoNT and is available free of charge from ANILAM, for example from the file base on the internet (filebase.anilam.de).

Note: A backup should be created after commissioning the machine and every time the machine parameters have been edited.

Allocation of Configuration Data

The configuration data is saved in several files with the extension .cfg (see "**Managing Configuration Files**"). Paths saved in the file **configfiles.cfg** refer to these files. This allocation has already been specified by ANILAM when the software is delivered, but it can be adapted by the OEM to his requirements.

Allocation of configuration data:

- oem.cfg General data for systems and channels.
- axlist.cfg Lists which axes are mounted, and can be displayed and programmed. axlist.cfg also refers to the kinematics "Kinem_1" and "Kinem_2".
- ax3list.cfg Same as axlist.cfg, but for a machine with only the three axes X, Y, and Z.
- kinem*.cfg Description of kinematics.
- kinem_XYZ.cfg Simple three-axis kinematics. (ax3list.cfg refers to this kinematics model.)
- axis_*.cfg Each drive has its own configuration file. This makes it easy to change drives.
- spindle.cfg Configuration of spindle drive.
- plc.cfg Configurations for the PLC.
- oemtable.cfg Configuration for tables
- configfiles.cfg Reference to the above configuration files.

The following topics are described:

- **Multi-axis System**
- **3-Axis System**
- **Multiple Variants**

Multi-axis System

The file configfiles.cfg for a 5-axis system might look like this (see **Data Files** soft key):

```
dataFiles:=
[
  "%OEM%\\config\\oem.cfg",
  "%OEM%\\config\\ax3list.cfg",
  "%OEM%\\config\\kinem_1.cfg",
  "%OEM%\\config\\kinem_2.cfg",
  "%OEM%\\config\\kinem_xyz.cfg",
  "%OEM%\\config\\axis_x.cfg",
  "%OEM%\\config\\axis_y.cfg",
  "%OEM%\\config\\axis_z.cfg",
  "%OEM%\\config\\axis_b.cfg",
  "%OEM%\\config\\axis_c.cfg",
  "%OEM%\\config\\spindle.cfg",
  "%OEM%\\config\\plc.cfg",
  "%OEM%\\config\\oemtable.cfg",
```

```
    "%USR%\\config\\user.cfg"
  ],
```

kinem_xyz is already included in this list, even though the system does not use it. But since a 3-axis system is often used for commissioning and testing, this allows you to quickly reconfigure the system (see below).

These configuration files are preset in the control's factory default setting.

3-Axis System

Working from a 5-axis configuration (see above) prepared on it, the system can easily be reconfigured to a 3-axis system by referring to the file ax3list.cfg.

```
  dataFiles:=
  [
    "%OEM%\\config\\oem.cfg",
   "%OEM%\\config\\ax3list.cfg",
    "%OEM%\\config\\kinem_xyz.cfg",
    "%OEM%\\config\\axis_x.cfg",
    "%OEM%\\config\\axis_y.cfg",
    "%OEM%\\config\\axis_z.cfg",
    "%OEM%\\config\\axis_b.cfg",
    "%OEM%\\config\\axis_c.cfg",
    "%OEM%\\config\\spindle.cfg",
    "%OEM%\\config\\plc.cfg",
    "%OEM%\\config\\oemtable.cfg",
    "%USR%\\config\\user.cfg"
  ],
```

Multiple Variants

As described above, configfiles.cfg may also contain files that are currently not being used. This enables you to support more than one variant without having to change or replace configfiles.cfg. However, instead of this, you can also enter only those files in configfiles.cfg you will really use.

In practice, it is often necessary to support different sets of kinematics, for example, if one customer wants to operate a fork head with the A axis, whereas another customer wants to operate it with the B axis. If configfiles.cfg is prepared properly, simply replace axlist.cfg and kinem_*.cfg to be able to switch between the two variants.

```
  dataFiles:=
  [
    "%OEM%\\config\\oem.cfg",
    "%OEM%\\config\\axlist.cfg",
    "%OEM%\\config\\kinem_1.cfg",
    "%OEM%\\config\\kinem_2.cfg",
    "%OEM%\\config\\kinem_xyz.cfg",
    "%OEM%\\config\\axis_x.cfg",
    "%OEM%\\config\\axis_y.cfg",
    "%OEM%\\config\\axis_z.cfg",
    "%OEM%\\config\\axis_a.cfg",
    "%OEM%\\config\\axis_b.cfg",
    "%OEM%\\config\\axis_c.cfg",
```

```
    "%OEM%\\config\\spindle.cfg",
    "%OEM%\\config\\plc.cfg",
    "%OEM%\\config\\oemtable.cfg",
    "%USR%\\config\\user.cfg"
 ],
```

Please note that both axis_a.cfg and axis_b.cfg are indicated. This prepares you for using both variants.

Setup of a Parameter File

The individual machine parameters are collected into parameter objects in the *.cfg parameter files. A parameter object has a name, of which the first three letters are always “Cfg.” The name is followed by an open parenthesis and a “key” for identifying the parameter object. This is followed by the individual machine parameters. A parameter object must be surrounded by parentheses.

If there are several input values for a parameter (such as separate parameter settings for each axis), then the corresponding parameter objects are addressed via the key, and therefore occur more than once.
The parameter objects in the system files have an “empty” key.

Note: ANILAM recommends changing the parameter values directly in the *.cfg files only in exceptional cases.

The following topic is described:

- **Rules for Entries**

Rules for Entries

If changes are to be made directly in a text editor, the following rules must be observed:

- “Key”: Each parameter object has a key at the beginning, which generally represents the name of the axis or channel, but in certain cases can also be empty. The control assigns this parameter object to the object addressed by the key, for example to the axis “X axis”.
- The characters := must come between the parameter name and the value.
- Individual parameters must be separated by commas. No comma may follow the last parameter.
- Individual components, such as strings in a list or the components in the array must be separated by commas. No comma may follow the last component.
- The different levels in path entries must always be separated by “\\”, for example, “%SYS%\\CONFIG\\AXIS\\...”.
- A list must always be in brackets [ ].
- If data objects with the same names and same identifications (keys) are present, the error message “**Data object already exists in file**” is displayed.
- Comments are text that is ignored during transfer. You can enter two types of comments:
- Comment in one line: After a double hyphen “--” the text until the end of the line is ignored.
- Comments that are on more than one line must be surrounded like this: (*comment*).

- Comments in files that are overwritten by the control (such as files with axis-setting parameters or oscilloscope parameters) are deleted. For this reason you should only add comments to files that are not written to by the control.

Example from the parameter file axlist.cfg:

Parameter object	Description
CfgChannelAxes(	Name of the parameter object with open parenthesis. You cannot change this name.
Key:= "Channel1",	Identification of the parameter object with a string, such as the name of the NC channel or an axis.
progAxis:= [	Data variable from a list.
“Xaxis”, “Yaxis”, “Zaxis”, “Aaxis”, “Baxis”], ...	The individual elements of a list are separated by commas. No comma may follow the last element in a list. A list must always be in brackets [ ].
[...], ...	More parameters follow…
)	No comma may come before the closing parenthesis. Conclusion of the parameter object
-- Comment to the end of the line	The text in the line after “--” (double hyphen) is ignored.
(* Comment distributed over several lines *)	Characters for comment beginning Everything between the comment beginning and end is ignored Characters for comment end

MP Subfiles

Individual data from the configuration files can be taken into the MP subfiles. These subfiles can be superimposed on the machine parameters during run time.
In principle, all files that do not require a system restart can be superimposed. The MP subfiles are usually activated by the PLC, but they can also be activated manually by using the configuration editor.

The following topics are described:

- **Syntax of MP Subfile**
- **Activating MP Subfiles**
- **Displaying/Editing Data Records in the Configuration Editor**
- **MP Change List in the Configuration Editor**
- **MP Movement Monitoring**
- **MP Programming Station Mode**

Syntax of MP Subfile

The syntax of an MP subfile corresponds to that of a basic file.
Subfiles differ from basic files in that only the entities or attributes to be changed must be described.
In basic files a data object (entity) must be described completely. This means that the basic file must contain the “key” and all “attributes” of the entity.

```
entity(
  key:= Key4711,
  attribute1:= value1,
  attribute2:= value2,
  attribute3:= value3,
  attribute4:= value4,
)
```

In subfiles, only the required data need to be indicated. Entity and key, however, must always be indicated.
Please note: MP subfiles must not contain any reset parameters.

Example of MP subfile with a new value for attribute 3:

```
entity(
  key:= Key4711,
  attribute3:= valuex
)
```

Activating MP Subfiles

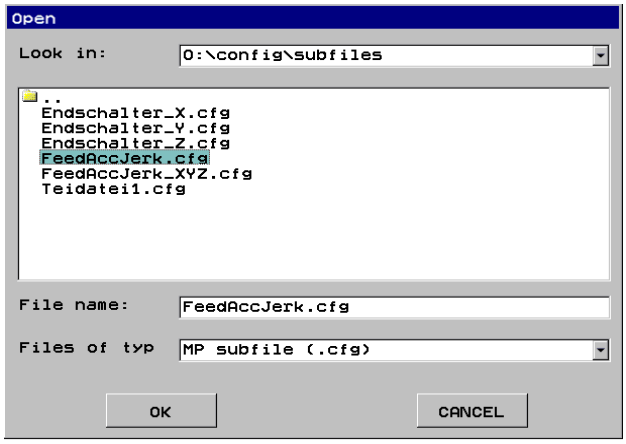
MP subfiles can be activated in the configuration editor or by the PLC.

The following topics are described:

- **Activation in the Configuration Editor**
- **Activation by the PLC**

Activation in the Configuration Editor

Use the **Load Subfile** (**F3**) soft key to activate individual subfiles.



A file will be loaded and activated immediately upon selection.

The PLC marker **NN_GenCycleAfterReConfig** is set upon activation if data relevant to the PLC have changed. Irrelevant data is indicated in the %SYS%\\config\\jh.cfg file as follows:

```
CfgNoNotification (
  key:="plc.QM4174",
  objectNames:=[
    "CfgOsci",
    "CfgOsciFile",
    "CfgOsciColor",
    "CfgOsciSetUp",
    "CfgOsciChannel",
    "CfgOsciTrigger",
    "CfgOsciDisplay",
    "CfgSelectFile",
    "CfgRecentFileList",
    "CfgDisplayData",
    "CfgPosDisplayPace",
    "CfgJogIncrement",
    "CfgInterpretOption",
    "CfgHandWheelFactor",
    "CfgAutoStartData",
    "CfgFeedRate",
    "CfgLayoutData",
    "CfgTablePath",
    "CfgEditor",
    "CfgGeoRotWorkPlane",
    "CfgUserPath",
    "CfgUnitOfMeasure",
    "CfgProgramMode",
    "CfgPassword",
    "CfgFunctionProtection",
    "CfgActualProtection",
```

```
    "CfgJhProtection",
    "CfgModSkText"
    ]
)
```

Activation by the PLC

Subfiles are activated by Module 9034. In this case the symbolic PLC operand **NN_GenCycleAfterReConfig** (M4174) will not be set.

Module 9034 Load a machine parameter subfile

With this module you load the contents of the given configuration file into the main memory. All configuration data not listed in this file remain unchanged.

The configuration file to be selected is checked. A faulty file is not loaded. If the file contains parameters that require a system reset, the file is not loaded.

The file name is transferred in a string that must contain the complete path, name, and file extension. Further characters, even space characters, are not permitted.

If the PLC program is created externally, ensure that lower-case letters are not used for the file name!

Once the NC program has started, the module operates only during the output of an M/S/T/Q strobe.

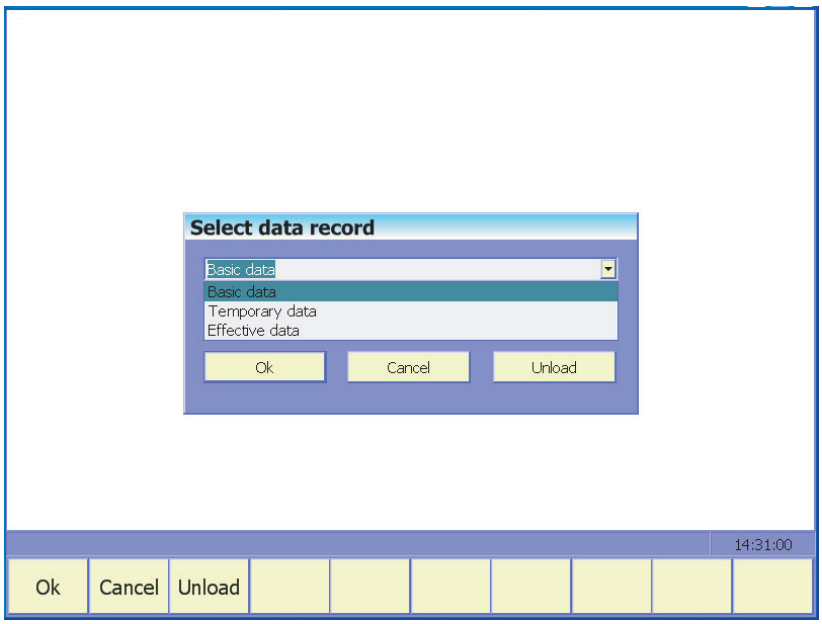
Call only in a submit job.

Call:

PS	B/W/D/K	<>String number> 0 to 99
CM	9034	
PL	B/W/D	<>Error code> 0: No error 1: String does not contain a valid file name, or the name (including the path) is too long 2: File not found 3: File is faulty / contains reset parameters 4: Incorrect string number was transferred (0 to 3) 5: Call was not in a submit job 6: Call during running program without strobe

Displaying/Editing Data Records in the Configuration Editor

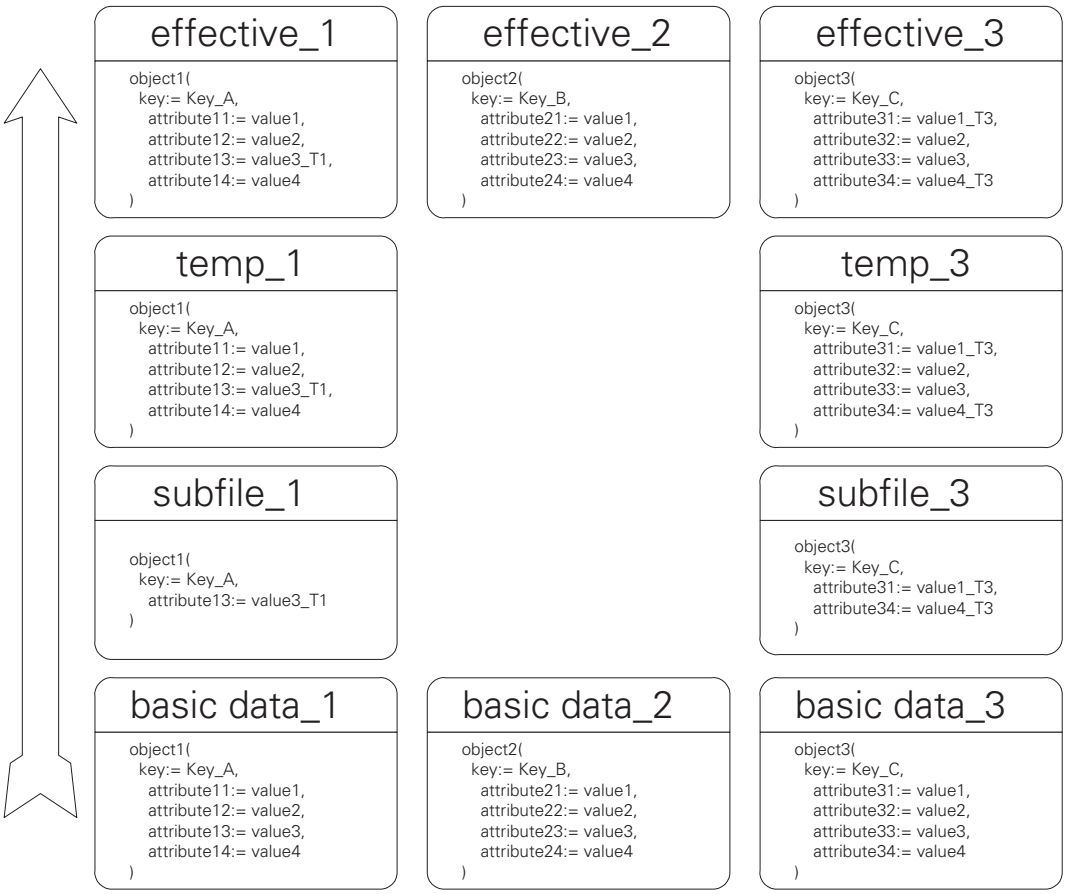
Use the **Select Record** (**F4**) soft key (refer to **Figure 3-1, Configuration Screen**) to choose between the following views:



The following topics are described:

- **Basic Data**
- **Loaded Subfile(s)**
- **Temporary Data**
- **Effective Data**
- **Unload Subfile**

The following overview shows which values of an object (object1, 2, 3) are displayed in the individual views:



Basic Data

This view shows the data imported during system start-up. Any changes will be rewritten to the respective basic files.

Loaded Subfile(s)

You must have loaded a subfile during system start-up (with "CfgPortionFiles") or by soft key for the subfile to be shown. If you select a subfile, only the data of the subfile are displayed in the configuration editor.
The attributes contained in the subfile can be edited and rewritten to the subfile while saving.
The subfile must be reloaded for the edited data to become effective.

Temporary Data

The complete data objects of all loaded subfiles are shown. They can be edited, but they are not rewritten to the file. If a subfile from the PLC is loaded, the data is saved only as temporary data.

Effective Data

The "effective data" view does not permit data to be edited. The displayed data refer to the current data loaded by the PLC or by selecting the data record of basic data or

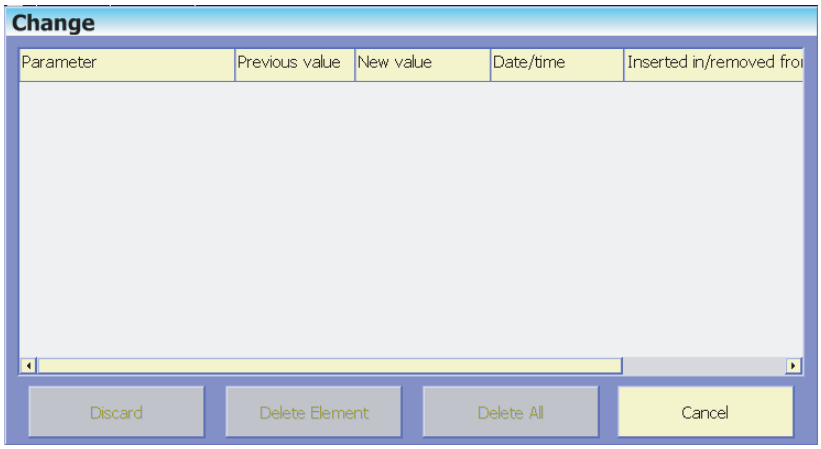
subfiles, and therefore show the data effective in the system.
Exception: Configuration data written with FN17 are not shown in this view.

Unload Subfile

The **UNLOAD** soft key can be used to unload the selected subfile.
This automatically activates the basic data and the subfiles that are still loaded. The result is also shown in “Temporary data” and “Effective data.”

MP Change List in the Configuration Editor

A machine parameter change list is displayed after pressing the **Save** or **OK** soft keys. The change window gives you an overview of all changed parameters. You can Discard, Delete Element, Delete All, or Cancel the changes.



The control also saves a list of the last 20 changes to the configuration data. In this list, you can see all changes performed, and can undo any of them. The change list is maintained upon power off of the control. The change list is reached in the configuration editor via **More Cmds** (**F7**)>**Secure Area** (**F6**)>**More Cmds** (**F7**)>**Discard Chgs** (**F7**).

MP Movement Monitoring

The actions of **MP_movementThreshold** (movement monitoring) were changed.
If the **MP_movementThreshold** parameter is set to a value greater than 0, the manipulated value of the position controller is now totaled in the IPO clock as soon as the threshold configured in the parameter is exceeded.
The control now calculates a nominal path and compares it with the actual path traversed after every 5 mm.
An error message is output if the actual path traversed is…

- less than a quarter of the nominal path

or

- four times greater than the nominal path.

Settings in the configuration editor:	
Axes ParameterSets [Parameter-set key] CfgEncoderMonitor **movementThreshold**	

MP Programming Station Mode

You can switch the control into programming-station mode with **MP_simMode**. The control can then be used as a programming station. No drives are enabled. You can create and test the PLC program and NC programs. The operation of the machine is simulated in the programming station mode. As OEM, you have access to the machine configuration in the programming station mode. This way the control can be adapted to the machine before actual commissioning, for example, or be used for training purposes.

Settings in the configuration editor:	
System CfgMachineSimul **simMode** **skipReferencing** **skipEmStopTest**	

The **MP_simMode** parameter offers three different setting possibilities for the programming station mode:

With **FullOperation** the control starts in normal operation. The programming station mode is deactivated. All drives and the PLC are active.

Choose the **CcOnly** setting in order to simulate the CC controller unit while the PLC is active. In this case all PLC inputs and outputs, as well as the emergency-stop loop (X41/34 and X42/4), must already be connected correctly in order to switch the control on correctly.

Choose **CcAndExt** in order to simulate the CC controller unit and all PLC inputs and outputs. The PLC runs in simulation mode, and the emergency-stop loop and PLC inputs and outputs are not interrogated.

The control must be restarted after changes in **MP_simMode** in order for the new settings to become active.

MP_simMode

Defines the programming station mode

Format: Pull-down selection menu

Selection:

[**FullOperation**]

Programming station mode is switched off, the emergency-stop loop (X41/34 and X52/4) must be complete. The drives are moved.

[**CcOnly**]

Simulation of the controlled drives. All PLC inputs and outputs, as well as the emergency-stop loop, must be connected correctly in order to switch the control on correctly.

[**CcAndExt**]

Simulation of the controlled drives and all PLC inputs and outputs. The emergency-stop loop does not need to be complete. The PLC runs in simulation mode.

Approaching the reference position of the axes can be skipped in programming station mode.

Set the **MP_skipReferencing** parameter to the value **TRUE** in order to set the axes directly on the reference position when the control is started.

MP_skipReferencing

Rapid approaching of the reference position

Format: Pull-down selection menu

Selection:

[**TRUE**]

The axes are set directly on the reference position when the control is started.

[**FALSE**]

The axes are not set on the reference position.

You can suppress the emergency-stop test with the **MP_skipEmStopTest** parameter.

MP_skipEmStopTest

No emergency-stop test is performed

Format: Pull-down selection menu

Selection:

[**TRUE**]

Emergency-stop test is not performed

[**FALSE**]

Emergency-stop test is performed

Read or Change Machine Parameters via a PLC Module

Machine parameters can be read and overwritten via the PLC.

Module 9430 Change the numeric value of a machine parameter

Use this module to enter a numeric value in the machine parameter given. The value of the machine parameter is overwritten in the run-time memory. The machine parameter in the .cfg file is not overwritten. The overwritten parameters are only in effect until the next control start-up.

Call only in a submit job.

Call:

PS	B/W/D/K/S	<>object name>
PS	B/W/D/K/S	<>key name>
PS	B/W/D/K/S	<>MP_ name>
PS	B/W/D/K	<>index>
PS	B/W/D/K	<>factor>
PS	B/W/D/K	<>value>
CM	9430	
PL	B/W/D	<>error>

Error:

0: Module executed correctly

1: Parameter does not exist, cannot be changed, or cannot be changed during program run

3: Fatal error (no connection to config server, etc.)

5: Call during program run without a strobe

6: Not called in submit/spawn

7: Not a numeric parameter

Entries for PS/PL	Description of the module entries
<object name>	Name of the object in which the parameter (attribute name) to be changed is located (for example, “PlcCfgValue”)
<key name>	Key where the object is located (for example, “Channel1”)
<MP name>	Name of the parameter (attribute) that is to be changed (for example, “oemWord768”)
<index>	Index within an array; 0= for parameters without array
<factor>	Conversion factor for real to integer, and vice-versa
<value>	Value of the parameter (for example, 123456)
<string>	String number (0 to 15)
<error>	See module description

Module 9431 Read the numeric value of a machine parameter

Use this module to read the value of the given machine parameter from the run-time memory.

Call only in a submit job.

Call:

PS	B/W/D/K/S	<>object name>
PS	B/W/D/K/S	<>key name>
PS	B/W/D/K/S	<>MP name>
PS	B/W/D/K	<>index>
PS	B/W/D/K	<>factor>
CM	9431	
PL	B/W/D	<>value>
PL	B/W/D	<>error>

Error:

0: Module executed correctly

1: Parameter does not exist, cannot be changed, or cannot be changed during program run

3: Fatal error (no connection to config server, etc.)

6: Not called in submit/spawn

7: Not a numeric parameter

Module 9432 Change the string value of a machine parameter

Use this module to enter a string in the machine parameter given. The value of the machine parameter is overwritten in the run-time memory. The machine parameter in the *.cfg file is not overwritten. The overwritten parameters are only in effect until the next control start-up.

Call only in a submit job.

Call:

PS	B/W/D/K/S	<>object name>
PS	B/W/D/K/S	<>key name>
PS	B/W/D/K/S	<>MP name>
PS	B/W/D/K	<>index>
PS	B/W/D/K/S	<>new string>
CM	9432	
PL	B/W/D	<>error>

Error:

0: Module executed correctly

1: Parameter does not exist, cannot be changed, or cannot be changed during program run

3: Fatal error (no connection to config server, etc.)

5: Call during program run without a strobe

6: Not called in submit/spawn

7: Parameter is not a string

Module 9433 Read the string value of a machine parameter

Use this module to read the value of the given machine parameter from the run-time memory.

Call only in a submit job.

Call:

PS	B/W/D/K/S	<>object name>
PS	B/W/D/K/S	<>key name>
PS	B/W/D/K/S	<>MP name>
PS	B/W/D/K	<>index>
PS	B/W/D/K	<>string number 0–15>
CM	9433	
PL	B/W/D	<>error> Error: 0: Module executed correctly 1: Parameter does not exist, cannot be changed, or cannot be changed during program run 3: Fatal error (no connection to config server, etc.) 6: Not called in submit/spawn 7: Parameter is not a string

Overview of the Machine Parameters of the 6000i

The following topics are described:

- **System**
- **Channels**
- **Axes**
- **KeySynonym**

System

Parameters valid for the entire system:

MP System	Function and Input	Reaction/ Access	Page
CfgAxes Definition of existing axes of all channels in the system; Create entries for all axes of the machine. Also for spindles and PLC axes.			
axisList MP10	**Keynames for all axes on the machine** Enter the axes of all channels of the machine, including spindles and PLC axes. Format: Array [0–9] Input: A string of max. 18 characters (e.g.: Xaxis, Yaxis, spindle1, spindle2, etc.)	RESET/ LEVEL3	
spindleIndices	**Keynames for all spindles on the machine** The spindle keys you define must be contained in 'MP_CfgAxes/axisList'. The list index of a spindle key defines the programmable spindle number used by the PLC to identify the spindle. Format: Array [0–1] Input: A string of max. 18 characters (e.g.: index 0 for spindle1, index 1 for spindle2, etc.)	RESET/ LEVEL3	
specCoordSysList	**Keynames of special axes for the kinematics description (optional)** If in 'MP_CfgAxisPropKin/specKinCoordSys' an axis is defined with 'FixedTransAxis' or 'DefPointTrans' or 'DefPointRot', you must enter the respective axis here. Format: Array [0–9] Input: A string of max. 18 characters (e.g.: C1, C2 etc.)	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgHardware Hardware specification; definition of type of drive controller.			
hardwareType	**Type of drive controller hardware** Input: - automatic : Automatic identification of controller unit - CC422 : CC 422, CC 522 controller unit for conventional axes - CC424 : CC 424 controller unit for direct drives with high control loop requirements (very short cycle times) - NoCC : No controller unit - CC520 In preparation: New CC 520 controller unit Default: NoCC – for analog command interface of the 6000i	RESET/ LEVEL3	
i32stopsMonitoring	**Behavior of input I32 (drive enabling)** Input: - on : If I32=0, all monitoring functions that can be influenced by the PLC are switched off. - off : Input I32 has no effect on the monitoring functions. Default: off	RESET/ LEVEL3	5-124
maxTouchFeed	**Absolute maximum probing feed rate** Limitation of values from touch probe table. Input: 0.000 to 99 960.000 [mm/min] Default: 960.000 [mm/min]	RESET/ LEVEL3	-

MP System	Function and Input	Reaction/ Access	Page
CfgCycleTimes Definition of cycle times for IPO, PLC, and Look Ahead			
ipoCycle MP7600.0	**Cycle time of position controller (interpolation clock pulse)** Only a cycle time of 3 [ms] for the position controller is supported. The value of 3 ms is therefore preset by the system. Input:3 ms Default: 3 ms	RESET/ LEVEL3	5-96
plcCount MP7600.1, MP7602	**PLC cycle time (Look Ahead cycle time)** The cycle time of the PLC and Look Ahead is a multiple of the Ipo clock (interpolation clock pulse). The Look Ahead function is triggered exactly two Ipo clock pulses after the PLC. Input:3 to 10 [ipoCycle] Default: 7 (i.e., the PLC cycle time is 21 ms)	RESET/ LEVEL3	
watchdogTime	**Delayed switch-off of SH1** Program the monoflop time for watchdog 2 here. Input: 1 to 6 [s] Default: 3 [s]	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgFilter Configuration of position command filters, applies to all axes			
typeFilter1	**Type of first nominal position value filter** Global definition of type of first nominal position value filter, valid for all axes. Input: Off: Filter 1 switched off Position:Axis position CutterLocation: For rotary axes Default: off	RESET/ LEVEL3	5-97
orderFilter1 MP1095	**Order of first nominal position value filter** Global definition of order of first nominal position value filter, valid for all axes. Input:1 to 31 Default: 11	RESET/ LEVEL3	5-97
typeFilter2	**Type of second nominal position value filter** Global definition of type of second nominal position value filter, valid for all axes. Input: Off: Filter 2 switched off Position:Axis position CutterLocation: For rotary axes Default: off	RESET/ LEVEL3	5-97
orderFilter2 MP1095	**Order of second nominal position value filter** Global definition of order of second nominal position value filter, valid for all axes. Input:1 to 31 Default: 11	RESET/ LEVEL3	5-97
CfgPosCorrection Parameters for asynchronous position compensation (optional)			
enable	**Switching asynchronous position compensation on/ off** Input: "ON" or "OFF"	RUN/ LEVEL3	
feed	**Speed for asynchronous position compensation** Speed at which the compensation is to be executed. Input: 0 to 99 960 [mm/min] Default: 960 [mm/min]	RUN/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgHandwheel Parameters for configuring the electronic handwheel Configuration of serial handwheel connected to X23.			
type MP7640	**Handwheel model** Indicate the type of connected handwheel. If the 'ENCODER' type is selected, you must enter more information on the connected handwheel in 'MP_CfgAxisHandwheel.' Input: **NONE:** No handwheel connected **HRNAX:** HR 410 with detent connected **HR410:** HR 410 connected **HR332:** HR 332 connected **HR330:** HR 330 connected **ENCODER:** Handwheel at encoder input (Function not available on 6000i)	RUN/ LEVEL3	
initValues MP7645	**Initialization values for handwheel** Format: Array [0–7] Input: 0 to 255 Default: 0	RUN/ LEVEL3	
incrPerRevol (optional)	**Increments per handwheel revolution** Input: 0 to 100 000 [incr.] Default: 0; this corresponds to 20 000 [incr.]	RUN/ LEVEL3	
rasterPerRevol (optional)	**Detent steps per handwheel revolution** Input: 0 to 100 000 Default: 0	RUN/ LEVEL3	
countDir MP7650	**Counting direction for handwheel** Input: Positive: Positive counting direction Negative: Negative counting direction	RUN/ LEVEL3	
sensitivity MP7660	**Sensitivity for electronic handwheel** Shock or vibrations can cause a slight motion at the handwheel. You can define a threshold sensitivity for the handwheel to prevent unintentional axis movements. Input: 0 to 10 000 [pulses] Default: 0	RUN/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
speedFactor MP7670.0 MP7670.1 MP7670.2	**Handwheel transmission ratio** Three transmission ratios in [%]. Distance per handwheel revolution: Transmission ratio defined in 'MP_CfgAxisHandwheel/ distPerRevol'. Format: Array [0–2] 0 = 1st level in [%] 1 = 2nd level in [%] 2 = 3rd level in [%] Default: 1, 10, and 100 [%]	RUN/ LEVEL3	
feedFactor	**Manual feed rates in the Electronic Handwheel mode** Percentage factor for the three feed rates. The effective handwheel feed rate is equal to the feed rate percentage factor multiplied by the maximum handwheel feed rate of the selected axis. Format: Array [0–2] 0 = 1st level in [%] 1 = 2nd level in [%] 2 = 3rd level in [%] Default: 1, 10, and 100 [%]	RUN/ LEVEL3	
crossShortSafety MP7640	**Short-circuit-proofed handwheel** Input: "on" or "off"	RUN/ LEVEL3	
CfgAutoStart AUTOSTART function Settings for automatic program start		NOTHING/ LEVEL1	
autoStartEnabled MP7683 bit 5	**Activate AUTOSTART soft key** Definition of automatic program start at any time. Input: ON: AUTOSTART soft key is displayed OFF: AUTOSTART soft key is not displayed Default: ON	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
DisplaySettings Configuration of the user interface			
CfgDisplayData Settings for screen displays			
axisDisplayOrder MP7291.0	**Sequence of displayed axes** Definition of sequence in which the axes are displayed. Field [0] is the top-most position. Format: Array [0–9] Input: String (see **MP_CfgAxes/axisList**)	NOTHING/ LEVEL1	
axisDisplayOrderRef (optional)	**Sequence of the displayed axes before the reference run** Definition of sequence in which the axes are displayed before the reference run. Field [0] is the top-most position. Format: Array [0–9] Input: String (see MP_CfgAxes/axisList)	NOTHING/ LEVEL1	
positionWinDisplay	**Type of position display in the position window** Input: SOLL: Nominal position IST: Actual position REFIST: Actual position referenced to the machine datum REFSOLL: Nominal position referenced to the machine datum SCHPF: Servo lag (following error) RESTW: Distance-to-go	NOTHING/ LEVEL1	
statusWinDisplay	**Type of position display in the status window** Input: SOLL: Nominal position IST: Actual position REFIST: Actual position referenced to the machine datum REFSOLL: Nominal position referenced to the machine datum SCHPF: Servo lag (following error) RESTW: Distance-to-go	NOTHING/ LEVEL1	
decimalCharacter MP7280	**Definition of decimal separator for position display** Input: “.” or “,” Default: “.”	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
CfgStatusAndQPar Settings for status values, Q parameters and tool data; Definition of behavior at program end and when the operating mode is changed.			
clearMode MP7300	**Reset status values, Q parameters and tool data (DL, DR, DR2 from PGM)** Input: 0 to 7 0: Reset status, Q parameters and tool data. 1: Erase the status display, Q parameters, and tool data if a program is selected and if M02, M30, or END PGM occur. 2: Erase the status display and tool data when a program is selected. 3: Erase the status display and tool data when a program is selected and if M02, M30, or END PGM occur. 4: Erase the status display and Q parameters when a program is selected. 5: Erase the status display and Q parameters when a program is selected and if M02, M30, or END PGM occur. 6: Erase the status display when a program is selected. 7: Erase the status display when a program is selected and if M02, M30, or END PGM occur.	RUN/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgPosDisplayPace Display step for individual axes			
Keys with the axis names of all axes and spindles Enter the axis names from 'MP_CfgAxes/axisList' (e.g., Xaxis, etc., as keys).			
displayPace MP7290.0-8	**Display step for position display in mm or degrees** Input: 0.1 [mm] or [°] 0.05 [mm] or [°] 0.01 [mm] or [°] 0.005 [mm] or [°] 0.001 [mm] or [°] 0.0005 [mm] or [°] 0.0001 [mm] or [°] 0.00005 [mm] or [°] 0.00001 [mm] or [°] Default: 0.001[mm] or [°]	NOTHING/ LEVEL1	
displayPaceInch MP7290.0-8	**Display step for position display in inches** Input: 0.005 inches 0.001 inches 0.0005 inches 0.0001 inches 0.00005 inches 0.00001 inches Default: 0.001[inch]	NOTHING/ LEVEL1	
CfgUnitOfMeasure Definition of the unit of measure valid for the display			
unitOfMeasure	**Unit of measure for display and user interface** Metric system or inches selectable. Input: metric: Metric system inch: Inches Default: metric	RUN/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
CfgProgramMode Format of NC programs and cycle display Definition of type of NC program for MDI (ANILAM conversational or ISO) and definition of type of cycle display.			
programInputMode	**Program entry in ANILAM plain language or in ISO** For positioning with manual data input. Input: ANILAM: Manual data input in ANILAM conversational format ISO: Manual data input using ISO code	RUN/ LEVEL1	
cycleFormat	**Display of cycles** Input: TNC_STD: Standard 6000i format TNC_PARAM:Parameters in one line	RUN/ LEVEL1	
CfgDisplayLanguage Settings for the NC and PLC dialog language			
ncLanguage MP7230.0	**NC conversational language** Input: ENGLISH GERMAN CZECH FRENCH ITALIAN SPANISH PORTUGUESE SWEDISH DANISH FINNISH DUTCH POLISH HUNGARIAN JAPANESE RUSSIAN CHINESE CHINESE_TRAD SLOVENIAN Default: ENGLISH	RUN/ LEVEL1	
plcDialogLanguage MP7230.1	**PLC conversational language** See **ncLanguage**	RUN/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
plcErrorLanguage MP7230.2	**Language for PLC error messages** See **ncLanguage**	RUN/ LEVEL1	
helpLanguage MP7230.3	**Language for online help** See **ncLanguage**	RUN/ LEVEL1	
CfgOsciColor Color settings for the internal oscilloscope			
background MP7365.0	**Background color** Input: black blue light_grey red dark_grey light_green really_light_grey really_dark_grey light_violet dark_green light_blue light_red medium_grey yellow white Default: white	NOTHING/ LEVEL3	5-168
channel1 MP7365.4	**Color for Channel 1** Input: See **background** Default: blue	NOTHING/ LEVEL3	5-168
channel2 MP7365.5	**Color for Channel 2** Input: See **background** Default: light_green	NOTHING/ LEVEL3	5-168
channel3 MP7365.6	**Color for Channel 3** Input: See **background** Default: light_blue	NOTHING/ LEVEL3	5-168
channel4 MP7365.7	**Color for Channel 4** Input: See **background** Default: yellow	NOTHING/ LEVEL3	5-168

MP System	Function and Input	Reaction/ Access	Page
channel5 MP7365.8	**Color for Channel 5** Input: See **background** Default: light_violet	NOTHING/ LEVEL3	5-168
channel6 MP7365.9	**Color for Channel 6** Input: See **background** Default: dark_green	NOTHING/ LEVEL3	5-168
logicTrace	**Color for logic-trace channels** Input: See **background** Default: black	NOTHING/ LEVEL3	5-168
select MP7365.3	**Color for selected channel** Input: See **background** Default: red	NOTHING/ LEVEL3	5-169
grid MP7365.1	**Color for grid** Input: See **background** Default: mediu_grey	NOTHING/ LEVEL3	5-169
cursorText MP7365.2	**Color for the cursor and text** Input: See **background** Default: really_dark_grey	NOTHING/ LEVEL3	5-169
CfgStartupData Behavior during control startup			
powerInterruptMsg	**Acknowledge the "Power interrupted" message.** Input: TRUE: Start-up is not continued until the message has been acknowledged. FALSE: The "**Power interrupted**" message does not display.	NOTHING/ LEVEL1	6-17

MP System	Function and Input	Reaction/ Access	Page
CfgShutDown Behavior when exiting control operation			
shutdownOnConfig MP4040	**Behavior when RESET configuration data are changed** Definition of the control's reaction to a change that requires a RESET. Note: If no value is entered for this parameter (icon is displayed dimmed), no automatic reset will be performed. In this case you start the reset by soft key. Input: RESTART: Control is shut down and restarts. TERMINATE: Control is shut down. SHUTDOWN: Control is shut down. POWEROFF: Shuts down the control. Default: RESTART	NOTHING/ LEVEL1	
shutdownOnError MP4040	**Behavior when RESET errors are acknowledged** Definition of the control's behavior when a RESET error is acknowledged. Input: See **shutdownOnConfig** Default: RESTART	NOTHING/ LEVEL1	
shutdownOnUser MP4040	**Behavior during switch-off by soft key** Definition of the control's behavior when it is shut down by soft key. Input: See **shutdownOnConfig** Default: SHUTDOWN	NOTHING/ LEVEL1	
shutdownOnOem MP4040	**Behavior when PLC module 9279 is called** Definition of the control's behavior when the OEM shuts down the control with PLC module 9279. Input: See **shutdownOnConfig** Default: RESTART	NOTHING/ LEVEL1	
maxTermTime (optional)	**Delay time until control is shut down** Definition of time to elapse before the process is terminated. Input: 0 to 1000 [s] Default: 0 [s]	NOTHING/ LEVEL1	
powerOffPort (optional) MP4041	**PLC output to be set after shutdown** Input: 0 to 31 Default: 0	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
powerOffDelay (optional) MP4042	**Delay time until PLC output is set** Time after shut down of the control until setting the PLC output from 'MP_powerOffPort'. Input: 0 to 1000 [s] Default: 0 [s]	NOTHING/ LEVEL1	
CfgTable Display properties of the table editor			
tableView	**Selection of various table views** The table view can be selected by using the screen layout in the table editor (applies to the first six entries). The last two entries are used in the 'Edit table characteristics' mode or the 'Select from table' mode. Input: UNDEF: No default TABLIST: List TABLISTPICT: List + graphics TABFORM: Form TABFORMPICT: Form + graphics TABFORMTEXT: Form + graphics TABLEPROP: Edit table characteristics TABLESELECT: Select from table	NOTHING/ LEVEL2	
enableNotify	**Switch for table change notification** Switch for prompt notification about external changes in the current table. SQL command option 'FOR NOTIFICATION'. Input: TRUE: Prompt notification about external changes. FALSE: No prompt notification about changes. Default: FALSE	NOTHING/ LEVEL2	
dispComplCol	**Display mode of column at right margin** Input: TRUE: Column is displayed, but its right margin is truncated. FALSE: Incomplete column is hidden completely. Default: FALSE	NOTHING/ LEVEL2	

MP System	Function and Input	Reaction/ Access	Page
CfgKeyboard Assignment of orange axis-address keys I to V Axis designation (see 'MP_CfgProgAxis/Key/axName')			
axisKeyI MP410.0	**Axis designation of Axis I on the operating panel** Input: X	NOTHING/ LEVEL1	
axisKeyII MP410.1	**Axis designation of Axis II on the operating panel** Input: Y	NOTHING/ LEVEL1	
axisKeyIII MP410.2	**Axis designation of Axis III on the operating panel** Input: Z	NOTHING/ LEVEL1	
axisKeyIV MP410.3	**Axis designation of Axis IV on the operating panel** Input: (e.g., C(	NOTHING/ LEVEL1	
axisKeyV MP410.4	**Axis designation of Axis V on the operating panel** Input: (e.g., B)	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
Paths Definition of the paths and file names that are valid for the entire system			
CfgJHPath Paths for ANILAM files; The path entries are write-protected and cannot be edited by the OEM.			
jhTable	**Path for ANILAM tables**	RESET/ LEVEL4	
jhCycle	**Path for ANILAM cycles**	RESET/ LEVEL4	
jhCycleDataFile	**Path/name for the ANILAM cycle file (.CDC)**	RESET/ LEVEL4	
jhCycleTreeFile	**Path/name for the ANILAM cycle file (.CDF)** ANILAM cycles that can be programmed by using predefined soft-key rows.	RESET/ LEVEL4	
sysCycleDataFile	**Path/name for the ANILAM cycle file (.CDC)** ANILAM cycles that cannot be programmed in the editor.	RESET/ LEVEL4	
sysCycleTreeFile	**Path/name for the ANILAM cycle file (.CDF)** ANILAM cycles that cannot be programmed in the editor.	RESET/ LEVEL4	
dspPath	**Path for CCU software (DSP software)** Directory in which the CCU software (controller software) is stored.	RESET/ LEVEL4	
runtimePath	**Path for files that are changed during the run time**	RESET/ LEVEL4	
CfgOemPath Paths for OEM files; Path entries for files that the OEM can create and change.			
oemTable	**Path for OEM tables** Format: String Input: Path of max. 260 characters	RESET/ LEVEL3	
dialogTextfile	**Name of the text file for OEM texts** The path %OEM%\plc\language\en is fixed, whereby the last subdirectory is formed from the configured language (here en = English). Format: String Input: Name of max. 260 characters	NOTHING/ LEVEL2	
cycleMainTreeFile	**Path and name for the OEM cycle file (.CDF)** Format: String Input: Path and name of max. 260 characters	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
cycleSubTreeFiles	**List of paths/names of user cycle files (.CDF)** The soft-key rows of these files are appended to the soft-key rows of the ANILAM or OEM files. Format: Array [0–9] Input: Path and name of max. 260 characters	RESET/ LEVEL3	
oemCycle	**Path for OEM cycles** Format: String Input: Path of max. 260 characters	RESET/ LEVEL3	
ncDir	**List of drives and/or directories** The drives and directories entered here are visible in the file manager, provided that you have the required access rights. The respective paths contain NC programs or tables, for example, floppy disk, HDR, or CFR directories, network drives, etc. Format: Array [0–9] Input: A string of max. 260 characters	RESET/ LEVEL3	
CfgUserPath Paths for the end user; Directories that are to be visible in the file manager. These entries can be edited by the user (machine operator).			
ncDir	**List of drives and/or directories** The drives and directories entered here are visible in the file manager, provided that you have the required access rights. The respective paths contain NC programs or tables, for example, floppy disk, HDR, or CFR directories, network drives, etc. Format: Array [0–9] Input: A string of max. 260 characters	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
CfgPlcPath Path/name of most recently compiled PLC program files			
mainPgm	**Path/name of the PLC main program** Format: String Input: Path and name of max. 260 characters Default: %OEM%\plc\TestPgmMC\MAIN.SRC	NOTHING/ LEVEL2	
pwmPgm	**Path/name of the PLC commissioning program** PLC program for commissioning the current controller. This PLC program is alternately compiled and run if 'MP_CfgHardware/currentControlAdjust' is set to 'on'. Format: String Input: Path and name of max. 260 characters Default: %OEM%\plc\testpgmmc\main_ib.src	NOTHING/ LEVEL2	
errorTable	**Path/name of the PLC error-message table (PET table)** Format: String Input: Path and name of max. 260 characters Default: %OEM%\table\PlcTestPgm.pet	NOTHING/ LEVEL2	
errorText	**Name of the text file for PLC error messages** The path %OEM%\plc\language\en is fixed, whereby the last subdirectory is formed from the configured language (here en=English). Format: String Input: Name of max. 260 characters Default: TestPgm.csv	NOTHING/ LEVEL2	
dialog	**Name of text file for PLC dialogs** The path %OEM%\plc\language\en is fixed, whereby the last subdirectory is formed from the configured language (here en=English). Format: String Input: Name of max. 260 characters	NOTHING/ LEVEL2	
softkeyProject	**Path/name of project file for PLC soft keys** Format: String Input: Path and name of max. 260 characters Default:%OEM%\plc\Test_Base\SOFTKEYS\PLC_softkeys.xml	NOTHING/ LEVEL2	
compErrorTable	**Path/name of error table for PLC compiler** Format: String Input: Path and name of max. 260 characters Default: %SYS%\config\plccomp.ert	NOTHING/ LEVEL2	

MP System	Function and Input	Reaction/ Access	Page
compCfgFile	**Path/name of configuration file for PLC compiler** Format: String Input: Path and name of max. 260 characters Default: %OEM%\plc\Test_Base\PLCCOMP.CFG	NOTHING/ LEVEL2	
events	**Path/name of the event list (SPAWN processes)** If evaluation is not in process, events for SPAWN processes are to be defined over API modules. Format: String Input: Path and name of max. 260 characters	NOTHING/ LEVEL2	–

MP System	Function and Input	Reaction/ Access	Page
CfgTablePath Path for tables Path for tables that can be activated in SQL instructions through the symbolic name (SQL synonym) given as key. Only enter the synonym instead of the complete path and file name; for example, TOOL instead of v:\table\tool.t.			
BASISTRANS **path**	**Symbolic table names for access via SQL commands** Path for tables that can be activated in SQL instructions through the symbolic name (SQL synonym) given as key. These symbolic names are used in cycles or in the PLC. Format: A string of max. 84 characters Input: Path/name consisting of device name, up to 6 directories, file name and extension	NOTHING/ LEVEL1	
KONST **path**	See **BASISTRANS**		
MANUAL **path**	See **BASISTRANS**		
MOTOR **path**	See **BASISTRANS**		
MOTOR_AMP **path**	See **BASISTRANS**		
MOTOR_OEM **path**	See **BASISTRANS**		
MP_MESSEN **path**	See **BASISTRANS**		
MPxxxx **path**	See **BASISTRANS**		
NC_JH_ERROR **path**	See **BASISTRANS**		
NC_OEM_ERROR **path**	See **BASISTRANS**		
PALLET **path**	See **BASISTRANS**		
PO_MERK **path**	See **BASISTRANS**		
SIMBASISTRANS **path**	See **BASISTRANS**		
SIMPALLET **path**	See **BASISTRANS**		
SIMTOOL **path**	See **BASISTRANS**		
SIMTOOL_P **path**	See **BASISTRANS**		
SIMZEROSHIFT **path**	See **BASISTRANS**		

MP System	Function and Input	Reaction/ Access	Page
TABCMA **path**	See **BASISTRANS**		
TCHPROBE **path**	See **BASISTRANS**		
TOOL **path**	See **BASISTRANS**		
TOOL_P **path**	See **BASISTRANS**		
TOOL_TC_PR **path**	See **BASISTRANS**		
ZEROSHIFT **path**	See **BASISTRANS**		
CfgSystemCycle Path for a system cycle The keyname is a symbolic name which will enable you to call the system cycle. This enables you to call machine-specific subcycles from a general system cycle without knowing the path of the respective cycles.			
path	**Path for a system cycle** The keyname is a symbolic name which will enable you to call the system cycle. Format: A string of max. 260 characters	RUN/ LEVEL3	
CfgBinFileCache Save NC programs binary-coded in cache; If a cache is present, the binary files are used for executing NC programs. Then the ASCII code does not have to be interpreted directly. The cache for binary files improves the system performance.			
cachePath	**Directory in which the binary cache files are saved** The cache is inactive if no directory is entered. Format: A string of max. 500 characters Input: Directory for binary files; (the %SYS%\bincache directory should not be changed!)	NOTHING/ LEVEL3	
maxFiles	**Maximum number of cache files in cache directory** If the defined quantity is exceeded, 10% of the cache is erased. Input: 0 to 1000 0= no cache memory available Default: 100	NOTHING/ LEVEL3	
freeSpace	**Minimum reserve memory to be kept free** If the memory reserve falls below 50 [MB], 10% of the cache is erased. Input: 0 to 4000 [Megabytes] Default: 50 [Megabytes]	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
ProgramManager Configuration of the program manager for file management.			
CfgFileType Assignment of the editor to a file type; Depending on the file extension, a standard editor as well as further data required for controlling the editing process are assigned.			
Key for file extensions (tables, programs, …) (e.g.: H (ANILAM programs), I (ISO programs), CMA (table for axis-error compensation), etc.)			
unitOfMeasure	**Unit of measure for length (metric/inch)** Position display, NC programs, tables, etc. Input: UNIT_MM: Input in mm UNIT_INCH: Input in inches UNIT_MMINCH: Input in mm or inches UNIT_INDEPENDENT: Input without unit of measure	NOTHING/ LEVEL3	
standardEditor	**Standard editor used for this file** Input: TEXT-EDITOR PROGRAM-EDITOR TABLE-EDITOR Default: TEXT-EDITOR	NOTHING/ LEVEL3	
fileSize (optional)	**File size above which the alternate editor is used** Files larger than MP_fileSize are no longer converted to binary by the program editor. Input: 0 to 1000 [KB] Default: 100 [KB]	NOTHING/ LEVEL3	
alternateEditor (optional)	**Alternative editor for files larger than MP_fileSize** Input: TEXT-EDITOR PROGRAM-EDITOR TABLE-EDITOR Default: TEXT-EDITOR	NOTHING/ LEVEL3	
CfgPathProtection Access rights to drives and directories			
Key with drive or directory (path name) (e.g., %OEM%\ or %USR%\CONFIG, etc.)			
protection	**Access rights to the drive or directory** Input: LEVEL1: Access via MOD key LEVEL2: Access via code number 123 LEVEL3: Access via code number 95 148	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
PLC Definition of PLC properties			
CfgPlcPeriphery Configuration of PLC peripheral devices Assignment of logical PLs to physical PLs; behavior of override potentiometers; reaction of PT100 inputs, etc.			
plNumber MP4030	**Assignment of logical PL / physical PL** Format: Array [0–3] Input: 0 to 3 0: Physical PL is first logical PL 1: Physical PL is second logical PL 2: Physical PL is third logical PL 3: Physical PL is fourth logical PL	NOTHING/ LEVEL2	
pt100Discrete MP4020 bit 7	**Transfer of PT100 values** Input: TRUE: Transfer value immediately FALSE: Transfer value at 1 K/s Default: TRUE	NOTHING/ LEVEL2	
tempCompensation MP4070	**Compensation of thermal expansion** Speed of lag-tracking axis-error compensation Input: 0 to 359 999.64 [mm/min]	NOTHING/ LEVEL2	
overrideFullRatio (optional)	**Compensation for cable losses of the override potentiometer** Input: 0.5 to 1.0 Default: 0.98	NOTHING/ LEVEL2	
overrideDelta (optional)	**Compensation for thermal noise in override potentiometer** Input: 0.0001 to 0.1 Default: 0.0005	NOTHING/ LEVEL2	
overrideIntegDelta (optional)	**Compensation for thermal noise in override potentiometer** Input: 0.0001 to 1.0 Default: 0.025	NOTHING/ LEVEL2	

MP System	Function and Input	Reaction/ Access	Page
CfgPlcTimer Default values for PLC timers and counters A change does not become effective until the PLC program has been restarted.			
Keys with the names of PLC timers and counters (e.g., T1, T4711, C0, C47, etc.)			
unit	**Unit of measure** Input: SECONDS: Input in seconds PLC_CYCLES:Input in number of PLC cycles	RUN/ LEVEL3	
value MP4110 / 4120	**Default value for PLC timers/counters** Format: Array [1–99] Input: 0 to 1000 [s or PLC cycles] Default: 0 [s or PLC cycles]	RUN/ LEVEL3	
CfgPlcFastInput Configuration of fast PLC inputs; Definition of numbers, operands and edge detection. Keep in mind that the time between two edges must be longer than the time from 'MP_CfgCycleTimes/ ipoCycle'.			
number MP4130	**Numbers of fast PLC inputs** Format: Array [0–4] Input: 0 to 31 Default: 0	RUN/ LEVEL3	
significance MP4131	**Activation criteria for fast PLC inputs** Format: Array [0–4] Input: **lowActive:** Activate at LOW level **highActive:**:Activate at HIGH level **allEdges:** Activate at both levels **disabled:** Switched off	RUN/ LEVEL3	
operand	**PLC operand for fast PLC inputs** Name or number of the operand that is set through the fast PLC input. Format: Array [0–4] Input: A string of max. 24 characters	RUN/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgPlcTechnology Reserved PLC markers for technology data Definition of a range of max. 16 markers with which technology-specific events of the interpolator can be used. (optional)			
operand	**PLC operand for IPO event marker** Definition of a PLC array operand with which the max. 16 IPO event markers for grinding can be used, for example, "PP_GrindIpoEvents". Input: A string of max. 24 characters	RUN/ LEVEL3	
CfgPlcMStrobe Properties of a range of M functions Output of M strobe from NC to PLC: Defines the treatment of the M function during NC program run and the mapping to the PLC markers.			
Keys with the name or numerical code of the M strobe (e.g., Cycle13, FN19, FN29, M3, M4, M5, M300, TouchProbe, etc.)			
Min.	**Number of the first M function** The properties described in this parameter object apply for this M function. Input: 0 to 9 999 Default: 0	RESET/ LEVEL3	
Max. (optional)	**Number of the largest M function** Greatest code for which the following properties apply. A missing attribute means that the properties apply only for the value shown under "min". Input: 0 to 9 999 Default: 0	RESET/ LEVEL3	
signal (optional)	**Symbolic name or the number of a marker** This marker is set during decoding. If the “acknowledge” attribute is not filled, a reset of this marker means the strobe is acknowledged. A missing attribute means that the data connected with the output are saved without synchronization with the PLC program and the output is immediately acknowledged. Input: A string of max. 24 characters	RESET/ LEVEL3	
acknowledge (optional)	**Symbolic name or the number of a marker** This marker is set to acknowledge the strobe. If an attribute is missing, resetting the marker given under “signal” means that the strobe is acknowledged. Input: A string of max. 24 characters	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
code (optional)	**Symbolic name or the number of a word marker** The M code is also copied to this marker during decoding. If an attribute is missing, the value cannot be read as numerical value. If the same word marker is given for two or more functions in one strobe, no more codes are saved in the programmed sequence than correspond to the field size taken from the symbol. If an absolute number is given, only one code is saved. Input: A string of max. 24 characters	RESET/ LEVEL3	
data	**Symbolic name or the number of a word marker** Additionally programmed data are copied to this marker. No more data are saved in the programmed sequence than correspond to the field size taken from the symbol. If an absolute number is given, only one datum is saved. If an attribute is missing, no data can be transferred. Input: A string of max. 24 characters	RESET/ LEVEL3	
revoke (optional)	**M functions whose effect will be canceled** A code cannot be pooled together with other codes in one strobe if their effects cancel each other. If an attribute is missing, the code can be pooled together with all other codes in one strobe. However, see attribute “singular”. S outputs for the same spindle outputs, T outputs and T2 outputs always cancel each other. However, in some cases they may also cancel M functions. M functions cannot cancel the effects of S, T, and T2 outputs. Format: Array [0–19] Input: 0 to 9 999	RESET/ LEVEL3	
singular	**Output only in its own strobe** Singular codes must always be output in their own strobe. Input: TRUE or FALSE	RESET/ LEVEL3	
blockEnd	**M function output at block end** Input: TRUE or FALSE	RESET/ LEVEL3	
blockSearch	**Code output also during mid-program startup** Input: TRUE or FALSE	RESET/ LEVEL3	
sync	**Code output with / without synchronization of the NC** A strobe is output without synchronization if this is permitted for all codes to be output. Input: SYNC_EXEC:Synchronous with program run SYNC_CALC:Synchronous with program calculation ASYNC: Output without synchronization	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
macro	**NC subprogram call instead of the M function** This makes it possible to indicate and run an NC subprogram instead of outputting the M function. Input: A string of max. 80 characters	RESET/ LEVEL3	
CfgPlcSStrobe Properties of the S function for a spindle Output of S strobe from NC to PLC: Defines the treatment of the S function during NC program run and the mapping to the PLC markers.			
Key with the number or name of the spindle (e.g., S etc.)			
signal	**Symbolic name or the number of a marker** This marker is set during decoding. If the “acknowledge” attribute is not filled, a reset of this marker means the strobe is acknowledged. A missing attribute means that the data connected with the output are saved without synchronization with the PLC program and the output is immediately acknowledged. Input: A string of max. 24 characters	RESET/ LEVEL3	
acknowledge	**Symbolic name or the number of a marker** This marker is set to acknowledge the strobe. If an attribute is missing, resetting the marker given under “signal” means that the strobe is acknowledged. Input: A string of max. 24 characters	RESET/ LEVEL3	
spindleSpeed	**Symbolic name or the number of a word marker** The spindle speed is also copied to this marker during decoding. If an attribute is missing, the spindle speed cannot be read as a numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	
badSpeed (optional)	**Symbolic name or the number of a marker** This is set if the programmed spindle speed is outside the permissible range. A missing attribute means that there is no monitoring. Input: A string of max. 24 characters	RESET/ LEVEL3	
spindleMode (optional)	**Symbolic name or the number of a word marker** The mode of the S strobe is also copied to this marker during decoding. If an attribute is missing, the value cannot be read as numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	
gearCode (optional)	**Symbolic name or the number of a word marker** The gear range is also copied to this marker during decoding. If an attribute is missing, the gear range cannot be read as a numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
revoke (optional)	**M functions whose effect will be canceled** A code cannot be pooled together with other codes in one strobe if their effects cancel each other. If an attribute is missing, the code can be pooled together with all other codes in one strobe. However, see attribute “singular”. S outputs for the same spindle outputs, T outputs and T2 outputs always cancel each other. However, in some cases they may also cancel M functions. M functions cannot cancel the effects of S, T, and T2 outputs. Format: Array [0–19] Input: 0 to 9 999	RESET/ LEVEL3	
singular	**Output only in its own strobe** Singular codes must always be output in their own strobe. Input: TRUE or FALSE	RESET/ LEVEL3	
blockSearch	**Code output also during mid-program startup** Input: TRUE or FALSE	RESET/ LEVEL3	
sync	**Code output with / without synchronization of the NC** A strobe is output without synchronization if this is permitted for all codes to be output. Input: SYNC_EXEC:Synchronous with program run SYNC_CALC:Synchronous with program calculation ASYNC: Output without synchronization	RESET/ LEVEL3	
CfgPlcTStrobe Properties of the T function (ToolCall and ToolDef) Output of T strobe from NC to PLC: Defines the treatment of the T function during NC program run and the mapping to the PLC markers.			
Keys with the names of the T strobes (e.g., ToolCall, ToolDef, etc.)			
type	**Type of T strobe** Input: T0: Remove tool from spindle T1: Insert tool in spindle T2: Prepare the next tool change	RESET/ LEVEL3	
signal	**Symbolic name or the number of a marker** This marker is set during decoding. If the “acknowledge” attribute is not filled, a reset of this marker means the strobe is acknowledged. A missing attribute means that the data connected with the output are saved without synchronization with the PLC program and the output is immediately acknowledged. Input: A string of max. 24 characters	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
acknowledge	**Symbolic name or the number of a marker** This marker is set to acknowledge the strobe. If an attribute is missing, resetting the marker given under “signal” means that the strobe is acknowledged. Input: A string of max. 24 characters	RESET/ LEVEL3	
toolNumber	**Symbolic name or the number of a word marker** The tool number is also copied to this marker during decoding. If an attribute is missing, the tool number cannot be read as a numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	
toolIndex	**Symbolic name or the number of a word marker** The tool index is also copied to this marker during decoding. If an attribute is missing, the index cannot be read as a numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	
toolMagazine	**Symbolic name or the number of a word marker** The magazine number is also copied to this marker during decoding. If an attribute is missing, the magazine number cannot be read as a numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	
pocketNumber	**Symbolic name or the number of a word marker** The pocket number is also copied to this marker during decoding. If an attribute is missing, the pocket number cannot be read as a numerical value. Input: A string of max. 24 characters	RESET/ LEVEL3	
revoke (optional)	**M functions whose effect will be canceled** A code cannot be pooled together with other codes in one strobe if their effects cancel each other. If an attribute is missing, the code can be pooled together with all other codes in one strobe. However, see attribute “singular”. S outputs for the same spindle outputs, T outputs and T2 outputs always cancel each other. However, in some cases they may also cancel M functions. M functions cannot cancel the effects of S, T, and T2 outputs. Format: Array [0–19] Input: 0 to 9 999	RESET/ LEVEL3	
singular	**Output only in its own strobe** Singular codes must always be output in their own strobe. Input: TRUE or FALSE	RESET/ LEVEL3	
blockSearch	**Code output also during mid-program startup** Input: TRUE or FALSE	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
sync	**Code output with / without synchronization of the NC** A strobe is output without synchronization if this is permitted for all codes to be output. Input: SYNC_EXEC:Synchronous with program run SYNC_CALC:Synchronous with program calculation ASYNC: Output without synchronization	RESET/ LEVEL3	
CfgPlcStrobeAlias Conversion to M functions Reproduction of control-dependent functions through the transfer of M functions to the PLC program.			
Keys with the names of the converted functions (alias strobes) (e.g., Cycle13, FN19, FN29, TouchProbe, etc.)			
type	**Type of alias strobe** Input: FN19: Data transfer NC prog. -> PLC, 2 values synchronously FN29: Data transfer NC prog. -> PLC, max. 8 values asynchronously CYCLE13: Define spindle position for M19 TCHPROBE:Calling measuring cycles	RESET/ LEVEL3	
mCode	**Number of the M function** Number of the M function for which the control-dependent function is mapped. Input: 0 to 9 999 Default: 0	RESET/ LEVEL3	
mOffset	**Offset (mCode) for the FN19 and FN29 functions** If the attribute has the value TRUE, then the first datum transferred with the FN19 or FN29 function is taken as an offset to the "min" value (first M function), and only the remaining data are saved under "data". Input: TRUE or FALSE	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgPlcOverrideDev Configuration of sources for override values Definition of hardware input and evaluation.			
Keys with the names of the override devices (e.g., potentiometerF, potentiometerS, etc.)			
source	**Selection of configurable source for override values** Input: OVR1, OVR2 or KEY	NOTHING/ LEVEL3	
mode	**Evaluation of override values** Input: DISCRETE, LINEAR or CURVE	NOTHING/ LEVEL3	
values (optional)	**Discrete values or interpolation points for curve** Format: Array [0–63] Input: 0 to 200	NOTHING/ LEVEL3	
CfgPlcOverrideS Configuration of spindle override for this spindle			
Keys with the names of the spindle overrides (e.g., spindle, etc.)			
minimal	**Minimum value for override** Input: 0 to 100 [%] Default: 0	NOTHING/ LEVEL3	
maximal	**Maximum value for override** Input: 0 to 200 [%] Default: 150	NOTHING/ LEVEL3	
source	**Source for override values** Format: A string of max. 18 characters Input: Device name (e.g., PotentiometerS)	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgOemBool User parameters with Boolean data			
Keys with the names of user status values The keynames of these objects are arbitrary and are defined by the OEM. If the data is to be copied into the PLC run-time image, the keynames must correspond to the names of the PLC markers, for example, M4017.			
value	**List of user status values (Boolean)** Format: Array [1– 99] Input: TRUE or FALSE	NOTHING/ LEVEL3	
ignorePlc (optional)	**Do not copy data object into the PLC image** Input: TRUE or FALSE	NOTHING/ LEVEL3	
CfgOemInt User parameters with integer data (whole numbers)			
Keys with the names of user whole-number values The keynames of these objects are arbitrary and are defined by the OEM. If the data is to be copied into the PLC run-time image, the keynames must correspond to the names of the PLC words, for example, W960, W976.			
value	**List of user whole-number values (integers)** Format: Array [1–99] Input: -2 147 483 648 to +2 147 483 647	NOTHING/ LEVEL3	
ignorePlc (optional)	**Do not copy data object into the PLC image** Input: TRUE or FALSE	NOTHING/ LEVEL3	
CfgOemPosition User parameters with fixed-point data (position value)			
Keys with the names of user fixed-point data The keynames of these objects are arbitrary and are defined by the OEM. If the data is to be copied into the PLC run-time image, the keynames must correspond to the names of the PLC double words, for example, D768.			
value	**List of user fixed-point values (position)** Format: Array [1–99] Input: -30 000 to +30 000	NOTHING/ LEVEL3	
ignorePlc (optional)	**Do not copy data object into the PLC image** Input: TRUE or FALSE	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgPlcOperTimes Configuration of machining time display			
displayPlcTimes MP7237.0	**Display PLC operating times** Format: %xxxxxxxx Input: Bits 0 to 7 represent PLC operating times 1 to 8 0: Do not display 1: Display	NOTHING/ LEVEL2	
resetPlcTimes MP7237.1	**Reset PLC operating times with the code number 857 282** Format: %xxxxxxxx Input: Bits 0 to 7 represent PLC operating times 1 to 8 0: Do not reset 1: Reset	NOTHING/ LEVEL2	
resetNcTimes MP7237.2	**Reset NC operating times with the code number 857 282** Format: %xxx Input: Bits 0 to 2 represent NC operating times 1 to 3 Bit 0 – Nonfunctional Bit 1 – "Machine on" operating time Bit 2 – "Program run" operating time 0: Do not reset 1: Reset	NOTHING/ LEVEL2	
textNumber MP7238	**Dialogs for PLC operating times** Format: Array [0–7] Input: Lists 0 to 7, field 0 is the text for PLC time 1 In the text file for PLC dialogs, indicate the line number of the dialog text. This text file is located under %OEM%\plc\language\en, whereby the last subdirectory is formed from the configured language (here en=English).	NOTHING/ LEVEL2	

MP System	Function and Input	Reaction/ Access	Page
CfgPlcBlockScan Beginning and end of strobes to be updated in the block scan Definition of M functions that mark the beginning and end of the block scan tracking for the PLC.			
mFirst	**M function at beginning of strobes to be updated** In the block scan, this M function is output before the strobes to be tracked. At a value of –1 there is no output. Input: -1 to 999 Default: –1 (no output)	RESET/ LEVEL3	
mLast	**M function at end of updated strobes** In the block scan, this M function is output after the tracked strobes. At a value of –1 there is no output. Input: -1 to 999 Default: –1 (no output)	RESET/ LEVEL3	
CfgSystemTime Universal Time (Greenwich Mean Time)			
offsetToUTC MP7235	**Time difference to universal time** This is used in the real-time system (HEROS). Difference from Greenwich Mean Time, for example, +1 for Central Europe Input: -23 to 23 [hours] Default: 1	RESET/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
EditorSettings			
CfgEditorSelect Definition of selection lists for the NC editor			
Keynames for possible selection lists The keyname "1-TOOL" is the ANILAM default.			
elementList	**List of keys for selection in the NC editor** Configuration descriptions of elements are available for these keys. Format: Array [0–7] Input: A string of max. 18 characters 1-TOOL: Tool selection by T number 2-TOOL: Tool selection by name These keys must be available in 'MP_CfgTableSelect'.	NOTHING/ LEVEL4	
CfgEditorSettings Settings for the NC editor			
createBackup	**Generate backup file** After you edit an NC program, make a backup file (*.bak). Input: TRUE or FALSE Default: TRUE	NOTHING/ LEVEL1	
deleteBack	**Behavior of the cursor after deletion of lines** Input: – TRUE: Cursor is on the previous line (same as 6000i) – FALSE: Cursor is on the following line Default: TRUE	NOTHING/ LEVEL1	
cursorAround	**Behavior of the cursor on the first or last line** Input: – TRUE: Cursor jumps to the last or first line – FALSE: Cursor stands ready; at first or last line Default: TRUE	NOTHING/ LEVEL1	
lineBreak MP7281.0	**Line break with multiline blocks** Input: – ALL: Always display all lines – ACT: Only display all lines when active block – NO: Only display all lines when active block is edited	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
stdTNChelp	**Activate filter** Input: TRUE: as 6000i, during input FALSE: Via HELP key	NOTHING/ LEVEL1	
toggleCyclDef (optional)	**Behavior of the soft key row after a cycle entry** Input: – TRUE: The cycle soft-key row remains active after the cycle entry - FALSE: The cycle soft-key row is hidden after the cycle entry	NOTHING/ LEVEL1	

MP System	Function and Input	Reaction/ Access	Page
TableSettings Description of table types, definition of parameters for tables			
CfgTableProperties Assignment of columns to a table type; Defines for a table type: – The columns in the table – The primary and foreign key With this information you can import a table or create a new one.			
Keys with the file extension of all tables For the key, enter the file extension of the table, for example, AMP, T, TCH.			
columnKeys	**List of keynames of the columns used** Keynames of all columns that form the table. The column names must be available under "**Columns**". Format: Array [1–79] Input: Column name (max. 18 characters)	NOTHING/ LEVEL3	
primaryKey	**Primary key (column name) for sorting** Name of the column, based upon which the data is sorted in ascending order. This column name must be available in 'MP_CfgTableProperties/Key/columnKeys'. Input: A string of max. 18 characters	NOTHING/ LEVEL3	
foreignKey (optional)	**Foreign key for this table** Specify a character string of the type <column name> <blank> <referential action> for each list item. Valid items for <referential action> are: - NO ACTION - RESTRICT - SET NULL - SET DEFAULT - INHERIT Format: Array [1–79] Input: A string of max. 40 characters	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
Columns Column description for all columns of all tables			
Keys with the column names of all tables Enter the column name as key. If the column is to be used only in a certain table, you must enter a unique keyname. Enter the table type (table extension) followed by a period and the column name, for example, AMP.F-AC, T.DL, TCH.T. If a column is used in two or more tables, a simple keyname is enough, for example, NAME (without a period).			
CfgColumnDescription Definition of a table column, column description			
width	**Column width** Definition of width for the column made in the table file. At least one character for the column name and one character for spacing from the next column. Input: 2 to 50 (column width of max. 50 characters) Default: 2	NOTHING/ LEVEL3	
unit	**Data type of values in the column** Input: TEXT: Text input SIGN: Algebraic sign + or – BIN: Binary number DEC: Decimal, positive, whole number (cardinal number) HEX: Hexadecimal number INT: Whole number LENGTH: Length FEED: Feed rate (mm/min or 0.1 ipm) IFEED: Feed rate (mm/min or ipm) FLOAT: Floating decimal point BOOL: Logical value INDEX: Index with subindexes	NOTHING/ LEVEL3	
initial (optional)	**Default value of the column** Input: A string of max. 50 characters - NULL: No defined default value. This column can stay empty. - Value: Default value. During insertion of a new line, this value is assigned as default to the column. If a default value other than NULL is entered, a valid value must always be entered in the column.	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
minimum	**Smallest permissible input value** The minimum value is considered only for the columns with numerical values or logical values. It defines the smallest permissible numerical input value or the text representing the logical value FALSE. For a value of data types FLOAT, FEED, IFEED, or LENGTH, the given number of decimal places determines the number of decimal places saved for values in this column (e.g., 0.001 means 3 decimal places). Input: A string of max. 50 characters	NOTHING/ LEVEL3	
maximum	**Largest permissible input value** The maximum value is considered only for the columns with numerical values or logical values. It defines the largest permissible numerical input value or the text representing the logical value TRUE. For a value of data types FLOAT, FEED, IFEED, or LENGTH, the given number of decimal places determines the number of decimal places saved for values in this column (e.g., 300.000 means 3 decimal places). Input: A string of max. 50 characters	NOTHING/ LEVEL3	
charset (optional)	**Permissible number of characters for text columns** The number of permissible characters is evaluated only for text columns. If it is defined, only the characters listed here are allowed; if not, all characters are allowed. Input: A string of max. 224 characters	NOTHING/ LEVEL3	
unique (optional)	**No ambiguous values allowed in the column** If the attribute is set to TRUE, unambiguous values are required in this column. If the attribute is not set or set to FALSE, the same values may display more than once in different lines. Input: TRUE: No unambiguous values FALSE: Values may display more than once	NOTHING/ LEVEL3	
readonly	**Write protection on column entry** If the attribute is set to TRUE, the value assigned when inserting the line cannot be changed. If the attribute is not set or set to FALSE, values may be overwritten. Input: TRUE: Values are write-protected FALSE: Values may be overwritten	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
unitIsInch (optional)	**Column entry in inches** If lengths and feed rates are to be specified in the column in a definite unit of measure, enter TRUE here for values in inches and FALSE for values in mm. If the attribute is not set, the unit of measure is taken from the corresponding table. Input: TRUE: Column entry contains measurement in inches FALSE: Column entry contains measurement in mm	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgColumnText Definition of language-sensitive text of a column Language-sensitive dialog text and selection lists for columns.			
dialogText Language-sensitive dialog text and selection lists for columns. Use the text name from the text resource file (for language-dependent texts), or use texts that are understood in all languages.			
dialogRes	**Name of a text** The text must be available with this name in a text resource file. Leave the attribute empty if the text is not to be language-sensitive. Then enter the text at the “text” attribute. Input: A string of max. 40 characters	NOTHING/ LEVEL3	
text	**Language-sensitive text** This text is loaded from a text resource file and should not be changed here. If the text is not language-sensitive, you must enter it here directly. In this case do not enter anything for the "dialogRes" attribute. Input: A string of max. 60 characters	NOTHING/ LEVEL3	
info	**Language-sensitive additional information** Help text for user parameter or text soft key. This text is loaded from a text resource file and should not be changed here. If the text is not language-sensitive, you must enter it here directly. In this case do not enter anything for the "dialogRes" attribute. Input: A string of max. 60 characters	NOTHING/ LEVEL3	
choice	**Define a selection list for input values** A selection element consists of a value/text pair. The text is displayed. When selected, the value belonging to the text is entered in the table. The value cannot be directly edited, but only changed using the selection. For the associated "text" attribute, the text can consist of two parts separated by a comma. This will have the following effect: - Text in front of the comma is displayed in the selection list - The following part in the table editor Format: Array [2–29]	NOTHING/ LEVEL3	
value	**Value for a selection element** A selection element consists of a value/text pair. When selected, the value belonging to the text is entered in the table. Input: A string of max. 20 characters	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
dialog Text for a selection element. A selection element consists of a value/text pair.			
dialogRes	**Name of a text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
text	**Language-sensitive text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
info	**Language-sensitive additional information** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
lockValue Locked due to certain input values. If the value entered in the column equals the value specified under “value”, the replacement text is displayed. It cannot be edited. Whether the editing is locked thus depends on the value.			
value	**Value for a selection element** A selection element consists of a value/text pair. When selected, the value belonging to the text is entered in the table. Input: A string of max. 20 characters	NOTHING/ LEVEL3	
dialog Text for a selection element. A selection element consists of a value/text pair.			
dialogRes	**Name of a text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
text	**Language-sensitive text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
info	**Language-sensitive additional information** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgTableBinding Predefined interpreter bindings for an SQL table. These bindings can be entered in the respective NC channel under 'MP_CfgSqlProperties/bindings'.			
Keys for all active table bindings Use the names from CfgSqlProperties/bindings as key.			
binds	**Keynames of the individual column bindings** For a list of all column bindings valid for this table see also 'MP_CfgColumnBinding'. Format: Array [1–79] Input: A string of max. 18 characters	RUN/ LEVEL3	
CfgColumnBinding Predefined interpreter binding of a column. Predefined binding of a column in an SQL table with a system datum in the interpreter. Enter the key for all active connections of a table under ’MP_CfgTableBinding’.			
Keys for all active column bindings Use the names from ’MP_CfgTableBinding/binds’ as key.			
column	**Table type and column name of the binding** Designation of the table type and the column name for which this binding is effective. Table type and column name must be linked with a dot (e.g., PRS.ROTA, T.TYPE, etc.). Input: A string of max. 18 characters	RUN/ LEVEL3	
id	**System data ID (group number)** Here id=0 means a binding with Q parameter. If no id is entered, no actual binding is generated, and the interpreter is simply instructed to read in the respective column descriptions during the configuration phase. Thus, this has not to be done during run time. Input: Group number 0 = Binding to Q parameter	RUN/ LEVEL3	
number	**System data no. (NR) or Q parameter no.** Input: System data no. or Q parameter no. when id=0	RUN/ LEVEL3	
index	**System data index (IDX)** Input: Index number	RUN/ LEVEL3	
subIndex	**System data subindex (IDXIDX)** Input: Is not used	RUN/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgTableFilter Filter for limited view of a table type			
Keyname for display filter The keyname must have the following syntax for selectable filters in the table editor (generic soft key): 't'.'f' - t stands for the table type; 1–3 characters - f for the filter; 1–16 characters (e.g., T.DRILL defines filters for the tool table.) Keynames that do not have this syntax are permitted, but they are typically used as reference objects (e.g., 1-TOOL from 'MP_CfgTableSelect/filter').			
softkeyText Language-sensitive description of soft key; use the text name from the text resource file (for language-dependent texts), or use texts that are understood in all languages.			
dialogRes	**Name of a text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
text	**Language-sensitive text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
info	**Language-sensitive additional information** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL3	
softkeyIcon	**Path/name of a soft key icon** The path and file name of an icon may be used instead of a text. Input: Path and name of max. 80 characters	NOTHING/ LEVEL3	
helpColumn	**Column name for displaying help graphics** If a column name is entered, different help graphics can be displayed. This depends on the column values in the "helpValue" attribute. Input: Column name of max. 50 characters	NOTHING/ LEVEL3	
helpValue	**List of column values for assignment to help graphics** A list of values can be specified for the column defined in "helpColumn". If this value is given in the table column, the help graphic defined in "helpPicture" will be displayed. Format: Array [0–49] Input: A string of max. 50 characters	NOTHING/ LEVEL3	
helpPicture	**List of path/file names of help graphics** When this filter is applied, the help graphic defined by path and file name is displayed in the 'FORM + GRAPHICS' view. Depending on the column value, you can display different help graphics if the corresponding entries are contained in "helpColumn" and "helpValue". Format: Array [0–49] Input: A string of max. 80 characters	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
select	**Selected column names of table** If the attribute is not set or empty, all columns are selected (SQL instruction: SELECT*). Input: A string of max. 80 characters	NOTHING/ LEVEL3	
from	**Path/file name or synonym of table** Definition of the desired table in connection with the possibility of selecting a column in the parameter object 'MP_CfgTableSelect'. Input: A string of max. 80 characters	NOTHING/ LEVEL3	
option	**Limitation of data record display** Optional for limiting the data record (SQL instruction option: WHERE and/or ORDER BY). Example: TYPE <> 21 AND L > 80 limits the data record of the table to exclude measuring probes (TYPE=21) and display only tools with a length > 80 mm. Input: A string of max. 80 characters	NOTHING/ LEVEL3	
CfgTablePrototype Path and file name for the prototype of a table type			
Keys with the file extension of all prototype tables For the key, enter the file extension of the prototype table, for example, TCH.			
path	**Path/name for the prototype of a table type** Input: A string of max. 80 characters	NOTHING/ LEVEL3	
enableReset	**Reset current table** Switch as to whether a current table may be overwritten with the prototype. Input: TRUE: Overwrite table FALSE: Do not overwrite table	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
CfgTableSelect Selection option for a column in the table editor. An inquiry can be configured for a column to copy values from one table to another table as foreign key.			
Keyname for selection option The keyname must have the following syntax for the selection option: 't'.'s' - t stands for the table type; 1–3 characters - s for the column name; 1–16 characters Example: TCH.T permits selection in column T (tool number) of the pocket table. Keynames that do not have this syntax are permitted, but they are typically used as reference objects (e.g., 1-TOOL from 'MP_CfgEditorSelect/elementList').			
filter	**Keyname for display filter in table editor** Keyname for a filter defining the table and the view (e.g., SELECT.TOOL for 'MP_CfgTableFilter'). Input: A string of max. 18 characters	NOTHING/ LEVEL3	
column	**Addresses for a value query during selection** Column name or column list, the values of which will be returned to the calling process during selection (e.g., T,NAME). Input: A string of max. 50 characters	NOTHING/ LEVEL3	
target	**Target table addresses for loading values** Column name or column list of the table to which the values are transferred after being returned from selection (e.g., T,TNAME). Input: A string of max. 50 characters	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
Network Description of communication connections			
Serial Configuration of serial interfaces			
CfgSerialPorts Data record belonging to the serial port; The data record for configuring the serial port is stored in 'MP_CfgSerialInterface'.			
interfaceRs232	**Keyname of the data record for the RS-232 interface** It refers to the default for the RS-232 serial interface entered in 'MP_CfgSerialInterface'. Input: A string of max. 18 characters	NOTHING/ LEVEL2	8-17
interfacePlc	**Keyname of the data records for the PLC interface** Refers to the serial interface settings when the interfaces are assigned to the PLC. If a reference is left empty, the default for the interface set in 'MP_CfgSerialInterface' will be selected. Format: Array [0–2] Input: A string of max. 18 characters	NOTHING/ LEVEL2	8-17
baudRateLsv2	**Data transfer rate for LSV2 communication in bauds** Input: BAUD_110 BAUD_150 BAUD_300 BAUD_600 BAUD_1200 BAUD_2400 BAUD_4800 BAUD_9600 BAUD_19200 BAUD_38400 BAUD_57600 BAUD_115200 Default: BAUD_115200	NOTHING/ LEVEL2	8-17

MP System	Function and Input	Reaction/ Access	Page
CfgSerialInterface Definition of data blocks for the serial ports; Under each keyname, the properties of a serial port are defined. The data block to be active is specified in the parameter 'MP_CfgSerialPorts'.			
baudRate MP5040	**Data transfer rate in bauds** Input: BAUD_110 BAUD_150 BAUD_300 BAUD_600 BAUD_1200 BAUD_2400 BAUD_4800 BAUD_9600 BAUD_19200 BAUD_38400 BAUD_57600 BAUD_115200 Default: BAUD_9600	NOTHING/ LEVEL2	8-17
protocol MP5030	**Data transfer protocol** Input: STANDARD:Standard data transfer BLOCKWISE:Blockwise transfer RAW_DATA:Transfer without protocol Default: STANDARD	NOTHING/ LEVEL2	8-19
dataBits MP5020 bit 0	**Data bits in each transferred character** Input: 7 bits: 7 data bits 8 bits: 8 data bits Default: 8 bits	NOTHING/ LEVEL2	8-19
parity MP5020 bit 4/5	**Type of parity checking** Input: NONE: Character parity not desired EVEN: Even character parity ODD: Odd character parity Default: NONE	NOTHING/ LEVEL2	8-19
stopBits MP5020 bit 6/7	**Data bits in each transferred character** Input: 1 stop bit or 2 stop bits Default: 1 stop bit	NOTHING/ LEVEL2	8-20

MP System	Function and Input	Reaction/ Access	Page
flowControl MP5020 bit 2/3	**Type of data-flow checking (handshake)** Input: NONE: No data-flow checking RTS_CTS: Transmission stop with RTS XON_XOFF:Transmission stop with DC3 (XOFF) Default: RTS_CTS	NOTHING/ LEVEL2	8-22
fileSystem	**File system for file operation via serial interface** Input: FEX: Magnetic-tape file system to be used for non-ANILAM devices, such as printer, punch, or PC with other data transfer software FE1: Floppy-disk file system to be used for ANILAM Floppy Disk Unit Default: FEX	NOTHING/ LEVEL2	8-23
bccAvoidCrtlChar (optional) MP5020 bit 1	**Block Check Character (BCC) is not a control character** Set to "TRUE" if you want to make sure that the check sum does not correspond to a control character. Input: TRUE or FALSE Default: FALSE	NOTHING/ LEVEL2	8-24
rtsLow (optional) MP5020 bit 8	**Status of the RTS line** Set to "TRUE" if the RTS line is at “low” level when idle. Input: TRUE or FALSE Default: FALSE	NOTHING/ LEVEL2	
noEotAfterEtx (optional) MP5020 bit 9	**Define behavior after receipt of ETX** Set to "TRUE" if no EOT character is to be sent upon receipt of the ETX character. Input: TRUE or FALSE Default: FALSE	NOTHING/ LEVEL2	

MP System	Function and Input	Reaction/ Access	Page
Keycode Define code numbers for OEM			
CfgOemPassword Password definition for OEM Use the password definition for enabling functions. Examples: - Activating the configuration editor with a specific layout - Evaluating the password in the PLC user program - Displaying soft keys in the MOD dialog Enter the password as keyname. If you want to make soft keys always available in the MOD dialog, enter “default” as keyname.			
funcList	**List of function names** Name of the function that is activated by entering the password. Enter the name in 'MP_CfgModOemSoftkey' and 'MP_CfgCfgEditActivate' as keyname. Format: Array [0–199] Input: Function name (max. 18 characters)	RESET/ LEVEL3	
CfgModOemSoftkey Soft key in the MOD dialog; Define a soft key in the MOD dialog (e.g., for activating the configuration editor with a specific layout). The keyname must be entered in 'MP_CfgOemPassword/funcList'.			
activation	**Foreground application** Definition whether the activated function is a foreground application. Input: TRUE: Foreground application (e.g., configuration editor) FALSE: Function performed in the background Default: FALSE	RESET/ LEVEL3	
skPos	**Soft-key space** Location for displaying the soft key (0=first soft key at left). Menu rows 1 and 2 are reserved for ANILAM. Leave empty if no soft key is to be displayed. Input: 0 to 7	RESET/ LEVEL3	
buttonText Soft-key text; Enter either a language-neutral text or the name of a text from a language-sensitive text file. Leave empty if you enter a graphic in the “buttonImage” attribute.			
dialogRes	**Name of a text** See entry under **CfgColumnText/dialogText**	RESET/ LEVEL3	
text	**Language-sensitive text** See entry under **CfgColumnText/dialogText**	RESET/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
info	**Language-sensitive additional information** See entry under **CfgColumnText/dialogText**	RESET/ LEVEL3	
buttonImage	**Soft-key graphic** Specify the directory and name of a graphic for the soft key. Leave empty if you enter a text in the “buttonText” attribute. Input: A string of max. 260 characters	RESET/ LEVEL3	
funcKey	**Function name** Name of the function that is activated by entering the password. To be specified only if the name of the function to be activated does not correspond to the keyname. Input: A string of max. 18 characters	RESET/ LEVEL3	
CfgCfgEditActivate Configuration editor layout; The OEM may configure the tree structure and attribute display according to his requirements.			
Keys with the function names Use the function name entered in 'MP_CfgOemPassword/funcList' or ’MP_CfgModOemSoftkey/funcKey' as keyname. "CFGEDIT-USER123" and "CFGEDIT-USERPARAM" are the ANILAM defaults.			
layoutFile	**Path/Name of layout file** The layout file defines the tree structure and the attributes to be displayed in the configuration editor. Input: A string of max. 80 characters	NOTHING/ LEVEL3	
dispLangText	**Language-sensitive names in the configuration editor** Input: TRUE: Display language-sensitive names FALSE: Display system names	NOTHING/ LEVEL3	
readOnly	**Start the configuration editor in read-only mode** In this mode, you can only view values, but not change them. Input: TRUE or FALSE	NOTHING/ LEVEL3	

MP System	Function and Input	Reaction/ Access	Page
Versions Version of the PLC, type and version of the NC			
CfgPlcVersion PLC program version			
plcVersion	**PLC software version** Format: String Input: Software version (max. 32 characters)	NOTHING/ LEVEL2	
versionText*-+ Language-sensitive description for PLC version. Use the text name from the text resource file (for language-dependent texts), or use texts that are understood in all languages.			
dialogRes	**Name of a text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL2	
text	**Language-sensitive text** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL2	
info	**Language-sensitive additional information** See entry under **CfgColumnText/dialogText**	NOTHING/ LEVEL2	
CfgNcVersion Version of the control; The version is entered by ANILAM when creating the system.			
Keyname for more detailed control description "KERNEL" and "PRODUCT" are the ANILAM defaults.			
ncType	**Control model** Format: A string of max. 16 characters Input: ATEK M	NOTHING/ LEVEL4	-
ncVersion	**NC software version** Format: A string of max. 16 characters Input: Software version	NOTHING/ LEVEL4	-

Channels

Settings for channel-reference parameters:

MP NCchannel	Function and input	Reaction/ Access	Page
CfgKinModel Description of kinematics models			
Keys with the names of the kinematics models Several kinematics models can be defined, such as Kinem_3Achs, Kinem_Gabel_cb. All kinematics models available are listed in **MP_CfgChanneAxes/ kinModels.** For multiple entries, the last one is valid			
axesToolSide	**Keys of the axes that lie on the tool side** The axes are listed in the correct (physical) sequence from the machine bed system to the tool. The axis names can be taken from 'MP_CfgAxes/ axisList' (e.g., Y axis, B axis). Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
trafoToolSide	**Coordinate transformations on the tool side** List of the keys for all coordinate transformations on the tool side. Defined in the same sequence as in 'MP_CfgKinModel/Key/axesToolSide'. Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
trafoDirToolSide	**Coordinate transformations defined by direction vectors** List of keys for the coordinate transformations on the tool side, where the orientation is defined by direction vectors. Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
trafoAngleToolSide	**Coordinate transformations defined by angle** List of keys for the coordinate transformations on the tool side, where the orientation is defined by angles. Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
toolCoordSys	**Key of the tool coordinate system** Indicates the end of the kinematics chain from the machine bed system to the tool. Input: A string of max. 18 characters	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
axesWpSide	**Keys for the axes on the workpiece side** The axes are listed in the correct (physical) sequence from the machine bed system to the workpiece. The axis names can be taken from 'MP_CfgAxes/ axisList' (e.g., X axis, C axis). Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
trafoWpSide	**Coordinate transformations on the workpiece side** List of the keys for all coordinate transformations on the workpiece side. Defined in the same sequence as in 'MP_CfgKinModel/Key/axesWpSide'. Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
trafoDirWpSide	**Coordinate transformations defined by direction vectors** List of keys for the coordinate transformations on the workpiece side, where the orientation is defined by direction vectors. Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
trafoAngleWpSide	**Coordinate transformations defined by angle** List of keys for the coordinate transformations on the workpiece side, where the orientation is defined by angles. Format: Array [1–9] Input: A string of max. 18 characters	RESET/ LEVEL3	
machineTableSys	**Key of the machine-table coordinate system** Indicates the end of the kinematics chain from the machine bed system to the workpiece. Input: A string of max. 18 characters	RESET/ LEVEL3	
activeSpindle	**Key of the active spindle of the kinematics model** The spindle name can be taken from 'MP_CfgAxes/spindleIndices' (e.g., spindle1). Input: A string of max. 18 characters	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgTrafoByDir Description of the coordinate transformations; Definition by direction vectors.			
location	**Origin of this coordinate system in the previous system** Position of the coordinate origin of the transformed system relative to the previous coordinate system. Format: Array [0.1.2] Input: - 100 000 to + 100 000 [mm]	RESET/ LEVEL3	
zDir	**Z-basis vector expressed in the previous coordinate system** Z-basis vector of the transformed system relative to the previous coordinate system Note: Translation axes move in this direction and rotation axes rotate around this vector. Format: Array [0.1.2] Input: - 1, 0, 1	RESET/ LEVEL3	
xDir	**X-basis vector expressed in the previous coordinate system** X-basis vector of the transformed system relative to the previous coordinate system Format: Array [0.1.2] Input: - 1, 0, 1	RESET/ LEVEL3	
CfgTrafoByAngle Description of the coordinate transformations; Definition by angles (Cardan, RollPitchYaw or Euler).			
location	**Origin of this coordinate system in the previous system** Position of the coordinate origin of the transformed system relative to the previous coordinate system. Format: Array [0.1.2] Input: - 100 000 to + 100 000 [mm]	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
angleDef	**Specify the interpretation of the angles** Input: Cardan: Orientation by Cardan angles RollPitchYaw: Orientation by rotation around fixed axes Euler: Orientation by Eulerian angles	RESET/ LEVEL3	
angle1	**Angle 1** Meaning corresponds to the "angleDef" attribute. Input: - 360 to +360 [°] Default: 0	RESET/ LEVEL3	
angle2	**Angle 2** Meaning corresponds to the "angleDef" attribute. Input: - 360 to +360 [°] Default: 0	RESET/ LEVEL3	
angle3	**Angle 3** Meaning corresponds to the "angleDef" attribute. Input: - 360 to +360 [°] Default: 0	RESET/ LEVEL3	
ChannelSettings Description of channel-referenced parameters			
Keynames of machining channels Two channels are permanently defined: **Ch-Nc** : Machining channel **Ch-Sim** : Simulation channel			
CfgChannelAxes Definition of this channel’s axes and axis names			
progAxis	**Programmable axes** Programmable axis names and axis names for the position display in the workpiece system. The axis name can be retrieved from 'MP_CfgAxes/axisList' (e.g., X axis, Y axis). Format: Array [0–7] Input: A string of max. 18 characters	RESET/ LEVEL3	
grindAxis (reserved)	**Axes of the grinding generator** Subset of ’MP_progAxis’, random order. Format: Array [0–5] Input: A string of max. 18 characters	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
refAxis (optional)	**Axes that are to be run over the reference point** Definition of the axes that are to be referenced and the reference order. Format: Array [0–5] Input: A string of max. 18 characters	RESET/ LEVEL3	
refAllAxes	**Home all axes in succession after an NC start** Input: TRUE or FALSE	RESET/ LEVEL3	
restoreAxis	**Sequence for returning to the contour** Definition of sequence in which the axes are returned to the contour after an NC stop or during a block scan. After an NC stop, the axes are moved to the stop position. During a block scan, they are moved to the calculated restore position. Format: Array [0–5] Input: A string of max. 18 characters	RESET/ LEVEL3	
kinModels	**Keys for the kinematics models available to this channel** For multiple entries, the last one is valid. The kinematics models are defined in 'MP_CfgKinModel'. Format: Array [0–15] Input: A string of max. 18 characters	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgChannelFile Definition of the initialization files for this channel			
geoChainInit	**Path/name of the file for initializing the geometry chain** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoIniProgram	**Path/name of the lead program for program run** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoIniBlock	**Path/name of the lead program for MDI mode** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoIniCycle	**Path/name of the lead program for manual cycles** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoCycleEnd	**Path/name of the trailing program for program end** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoCancelCycle	**Path/name of the trailing program if program aborts** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoTCallCycPath	**Path/name for the TOOL CALL cycle** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoTDefCycPath	**Path/name for the TOOL DEF cycle** Input: A string of max. 260 characters	RESET/ LEVEL3	
geoAutoTCallSycle	**Path/name for automatic TOOL CALL cycle** TOOL CALL after expiration of tool life. Input: A string of max. 260 characters	RESET/ LEVEL3	
geoPalletCtrlCycle	**Path/name of the pallet control cycle** Input: A string of max. 260 characters	RESET/ LEVEL3	
plcSetPresetCycle	**Path/name of the preset cycle for PLC module 9090/9281** Input: A string of max. 260 characters	RESET/ LEVEL3	
progSelectCycle	**Path/name of the program selection cycle** Input: A string of max. 260 characters	RESET/ LEVEL3	
afterMdiCycle	**Path/name of the trailing program when leaving MDI** This program is called when you leave the "Positioning with MDI” mode. Input: A string of max. 260 characters	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgSqlProperties Table bindings for this channel; The keys listed under "tables" identify the tables, and the keys listed under "bindings" identify the corresponding column bindings. A corresponding binding at the same list position under “bindings” belongs to every list entry under "tables".			
tables	**Keynames for tables** Format: Array [0–39] Input: A string of max. 18 characters	RUN/ LEVEL3	
bindings	**Keynames for table column bindings** Enter predefined bindings from 'MP_CfgTableBinding'. Format: Array [0–39] Input: A string of max. 18 characters	RUN/ LEVEL3	
CfgNcErrorReaction Behavior of programmable errors FN14:ERROR; FN14 errors are triggered only if according to the PET table the warning level of the error is maximally as high as the warning level set here. Note that errors with warning level 0 are always triggered and errors with warning level 5 are never triggered.			
warningLevel	**Warning level of channel** Input:0 to 4 Input: 0: FN14 errors with warning level = 0 are triggered 1: FN14 errors with warning level </= 1 are triggered 2: FN14 errors with warning level </= 2 are triggered 3: FN14 errors with warning level </= 3 are triggered 4: FN14 errors with warning level </= 4 are triggered Default: 0	RUN/ LEVEL1	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgNcPgmParState Specification for storage of Q/QS parameters Definition of persistent storage and name of the current Q/QS parameter block			
persistent	**Persistent storage of Q/QS parameters** Input: – TRUE: Storage of Q/QS parameters in the current parameter block at program end, see 'MP_currentSet' - FALSE: No storage Default: FALSE	RUN/ LEVEL1	
currentSet	**Name of the current Q/QS parameter block** If no name is indicated, the name of the machining channel is used for storage. Input: A string of max. 18 characters	RUN/ LEVEL1	
CfgGeoTolerance Geometry tolerances			
circleDeviation MP7431	**Permissible deviation from radius** Permissible deviation of circle radius at the circle end point from the radius at the circle starting point. Input: 0.0001 to 0.016 [mm] Default: 0.005 [mm]	RUN/ LEVEL1	
CfgGeoCycle Configuration of geometry cycles			
pocketOverlap MP7430	**Overlap factor for pocket milling** Input: 0.001 to 1.414 Default: 1.000	RUN/ LEVEL1	
suppressSpindleErr MP7441 Bit0	**Error message “Spindle ?” is suppressed.** Input: on or off Default: off	RUN/ LEVEL1	
suppressDepthErr MP7441 bit 2	**Error message “Enter depth as negative” is suppressed.** Input: on or off Default: off	RUN/ LEVEL1	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgRestorePosition Configuration of the restore position			
distance	**Safety clearance when moving to the restore position** Input: 0.0 to 500.0 [mm]	RUN/ LEVEL1	
CfgLiftOff Configuration of lift-off parameters for NC stop			
on	**Switching on/off lift-off movements during NC stop** Input: on: Lift-off movements active off: Lift-off movements not active	RUN/ LEVEL3	
distance	**Maximum retraction height for NC stop** Input: 0.0 to 2.0 [mm] Default: 0.0 [mm]	RUN/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgLaPath Parameters for calculation of path feed-rate profile; Definition of feed rate minimum within one segment (path segment) and for transition between two segments (corners). The feed rate minimum is violated only if programmed. Maximum values for jerk and yank are defined for acceleration on the path. A tolerance value is included. The feed rate for corners and curvatures is limited so that the filter error does not exceed this value.			
minPathFeed	**Minimum feed rate on the path** This feed rate within a segment is only violated if a lower feed rate is programmed. Input: 0.0 to 600 000.0 [mm/min] Default: 60.0 [mm/min]	RUN/ LEVEL3	
minCornerFeed	**Minimum feed rate at corners** This feed rate between two segments is only violated if a lower feed rate is programmed. Input: 0.0 to 600 000.0 [mm/min] Default: 30.0 [mm/min]	RUN/ LEVEL3	
maxG1Feed	**Maximum machining feed rate** For feed rates higher than the maximum machining feed rate, the same MPs ("maxPathJerkHi" or "pathToleranceHi") as for rapid traverse apply. Input: 0.0 to 99 999.0 [mm/min] Default: 99 999.0 [mm/min]	RUN/ LEVEL3	
maxPathJerk MP1090	**Maximum jerk on the path** This value applies for machining feed rates up to "maxG1Feed". Input: 0.0 to 1 000 000.0 [m/s^3] Default: 40.0 [m/s^3]	RUN/ LEVEL3	
maxPathJerkHi	**Maximum jerk on the path at rapid traverse** This value also applies for feed rates greater than "maxG1Feed". Input: 0.0 to 1 000 000.0 [m/s^3] Default: 40.0 [m/s^3]	RUN/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
pathTolerance MP1096	**Path tolerance for contour transitions after the filter** The feed rate for corners and curvatures is limited so that the filter error does not exceed this value. The path tolerance can be changed in Cycle 32. Input: 0.0001 to 10.000 [mm] Default: 0.010 [mm]	RUN/ LEVEL3	
pathToleranceHi	**Path tolerance after the filter at rapid traverse** This value also applies for feed rates greater than "maxG1Feed". Input: 0.0001 to 10.000 [mm] Default: 0.010 [mm]	RUN/ LEVEL3	
maxPathYank	**Maximum yank on the path (dj/dt)** Input: 0.0 to 1 000 000.0 [mm/s^4] Default: 4 000.0 [mm/s^4]	RUN/ LEVEL3	
curveTolFactor	**Factor for path tolerance in circles** This makes it possible to separately set the tolerance for corners (MP_pathTolerance effective) and arcs (MP_curveTolFactor * MP_pathTolerance effective). Input: 0.5 to 100.0 Default: 1.0	RUN/ LEVEL3	
curveJerkFactor	**Factor for feed rate reduction at curvature changes** If oscillations occur, for example, at line-to-arc transitions, the feed rate can be reduced by MP_curveJerkFactor (<1.0). Input: 0.0 to 100.0 Input 0: No further feed rate reduction Default: 1.0	RUN/ LEVEL3	
angleTolerance (optional)	**Tolerance for rotary axes with M128 (changeable in Cycle 32)** Input: 0.0001 to 10.0 [mm] Default: 0.01 [mm]	RUN/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
angleToleranceHi (optional)	**Tolerance for rotary axes with M128 and rapid traverse** This value also applies for feed rates greater than "maxG1Feed". Input: 0.0001 to 10.0 [mm] Default: 0.01 [mm]	RUN/ LEVEL3	
CfgPlcStrobes Definition of the M, S and T strobes for this channel			
mStrobes	**List of M strobe descriptions in this channel** Format: Array [0–99] Input: A string of max. 18 characters	RESET/ LEVEL3	
sStrobe	**Description of the S strobe in this channel** Input: A string of max. 18 characters	RESET/ LEVEL3	
tStrobes	**List of T strobe descriptions in this channel** Format: Array [0–2] Input: A string of max. 18 characters	RESET/ LEVEL3	
aliasStrobes	**List of the converted strobes in this channel** Reproduction of control-dependent functions on uniform M-function transfer to the PLC program. Format: Array [0–3] Input: A string of max. 18 characters	RESET/ LEVEL3	
unitOfMeasure (optional)	**Symbolic name or the number of a marker** This is set if the NC program causing the strobe is programmed in inches. If an attribute is missing, the unit of measure (metric/inch) is not available. Input: A string of max. 24 characters	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgPlcOverrideF Configuration of the feed rate override for this channel			
minimal	**Minimum value for override** Input: 0 to 100 [%] Default: 0 [%]	NOTHING/ LEVEL3	
maximal	**Maximum value for override** Input: 0 to 200 [%] Default: 150 [%]	NOTHING/ LEVEL3	
source	**Source for override values** Input: A string of max. 18 characters	NOTHING/ LEVEL3	
CfgPlcOverrideR Configuration of rapid traverse override for this channel (optional)			
minimal	**Minimum value for override** Input: 0 to 100 [%] Default: 0 [%]	NOTHING/ LEVEL3	
maximal	**Maximum value for override** Input: 0 to 200 [%] Default: 150 [%]	NOTHING/ LEVEL3	
source	**Source for override values** Input: A string of max. 18 characters	NOTHING/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgToolBehaviour Behavior of the tool change cycles			
varPocketCoding	**Tool pocket table with flexible tool-pocket coding** Input: TRUE: Flexible pocket coding FALSE: Fixed pocket coding Default: FALSE	RESET/ LEVEL1	
toolDefMode	**Behavior of the tool change cycles in TOOL DEF** Input: 0 to 2 0: TOOL DEF serves to call the tool (L or R not allowed) 1: TOOL DEF is for tool definition (L and/or R must be available). 2: TOOL DEF is for tool definition, but, unlike 1, also allows redefinition. Default: 0	RESET/ LEVEL1	
cleanToolTblAtRun	**Empty the tool table during program start** Input: TRUE: Empty the tool table, advisable if toolDefMode = 1 or 2 FALSE: Do not empty the tool table, advisable if toolDefMode = 0 or 2 Special case 1: Empty/modify the complete table if modifyToolTblFrom = 0 und modifyToolTblTo = 0 Special case 2: Emptying/modifying disabled for range “from” to “to” if modifyToolTblFrom < modifyToolTblTo Default: FALSE	RESET/ LEVEL1	

MP NCchannel	Function and input	Reaction/ Access	Page
modifyToolTblFrom	**Minimum possible pocket number for tool table modification** Modification, including this pocket number Input: - 0 to n (clear): The tool table is cleared starting from the specified pocket number, advisable if cleanToolTableAtRun = TRUE and toolDefMode = 1 or 2. - 0 to n (modify): The tool table is modified starting from the specified pocket number, advisable if cleanToolTableAtRun = FALSE and toolDefMode = 2.	RESET/ LEVEL1	
modifyToolTblTo	**Maximum possible pocket number for tool table modification** Modification, including this pocket number Input: - 0 to n (clear): The tool table is cleared up to the specified pocket number, advisable if cleanToolTableAtRun = TRUE and toolDefMode = 1 or 2. - 0 to n (modify): The tool table is modified up to the specified pocket number, advisable if cleanToolTableAtRun = FALSE and toolDefMode = 2.	RESET/ LEVEL1	

MP NCchannel	Function and input	Reaction/ Access	Page
CfgPrefForPolarKin Settings for polar kinematics (optional)			
kindOfPref	**Behavior of polar kinematics at radius 0** If the tool center lies exactly on the polar axis (C axis, radius 0) on machines with polar kinematics, there are two possibilities for a linear positioning block: r,phi or -r,phi+180. This attribute can be used to influence the behavior of the control if the tool center path crosses the polar axis. Input: - RadiusPositive: After crossover, the tool center is in a position with positive radius. - RadiusNegative: After crossover, the tool center is in a position with negative radius. - MinimalAngle: The crossover path is executed through a C-axis movement that is as small as possible. - NoChangeOfRadius: After crossover, the radius has the same algebraic sign as before crossover. - Radius0NotAllowed: A tool center path that crosses the polar axis is not allowed.	RUN/ LEVEL1	
CfgSafety Configuration of the safety packet for this channel; Definition of axis-independent safety parameters for machines with integrated operator safety system (SG).			
timeToEmStopTest MP511	**Time for switch-off test** Maximum time until the next test of the cutout channels is due. Input: 1 to 1 440 [min] Default: 1 440 [min]	RESET/ LEVEL3	
timeToAxGrpStop1 MP520	**Stopping time of axis group A for SH** Input: 0.000 to 2.000 [s] Default: 1.000 [s]	RESET/ LEVEL3	

MP NCchannel	Function and input	Reaction/ Access	Page
timeToSpGrpStop1 MP521	**Stopping time of axis group S for SH** Input: 0.000 to 10.000 [s] Default: 1.000 [s]	RESET/ LEVEL3	
timeToAxGrpStop23 MP523	**Stopping time of axis group A for SBH** Input: 0.000 to 2.000 [s] Default: 1.000 [s]	RESET/ LEVEL3	
timeToSpGrpStop23 MP524	**Stopping time of axis group S for SBH** Input: 0.000 to 10.000 [s] Default: 1.000 [s]	RESET/ LEVEL3	
specialModeOn MP526	**Enable special mode** Input: on: Special mode active off: Special mode not active Default: off	RESET/ LEVEL3	
inpNoAxSwitchPos MP580	**Input numbers of machine-panel axis keys +** Format: Array [0.1.2.3.4] Input: - 1 to +152	RESET/ LEVEL3	
inpNoAxSwitchNeg MP581	**Input numbers of machine-panel axis keys -** Format: Array [0.1.2.3.4] Input: - 1 to +152	RESET/ LEVEL3	
inpNoMachine MP582	**Input numbers of remaining machine-panel axis keys** Format: Array [0.1.2.3.4.5.6.7.8.9.10] Input: - 1 to +152	RESET/ LEVEL3	
inpNoHandwheel MP583	**Input numbers of handwheel keys** Format: Array [0.1.2.3.4.5.6] Input: - 1 to +175	RESET/ LEVEL3	

Axes

Axis-specific parameters:

MP Axes	Function and input	Reaction/ Access	Page
CfgProgAxis Settings for programmable / displayed axes If you want to be able to program, display and/or edit axis names, you must enter the corresponding keyname of the axis here.			
Keynames of axes They can be taken from **MP_CfgAxes/axisList** (e.g., X axis). Other names may be assigned to axes whose names cannot be programmed until a specific kinematics model is activated.			
axName	**Designation of the axis for position display** This axis name is also valid for programming/ editing. Format: String Input: Programmable axis name, such as A, B, C, U, V, W, X, Y, Z	RESET/ LEVEL3	
dir	**Spatial orientation of the axis or center of rotation** Input: XAxis: Motion / rotary axis in X direction YAxis: Motion / rotary axis in Y direction ZAxis: Motion / rotary axis in Z direction SpecAxis: Free/undefined spatial orientation (e.g., for spindle)	RESET/ LEVEL3	
progKind	**Type of axis** Input: MainLinCoord: Primary coordinate, always linear (X, Y, Z) ParallelLinCoord: Secondary linear coordinate (U, V, W) ParallelAngCoord: Parallel rotary coordinate (A, B, C) SatelliteLinCoord: Other linear coordinate ??? SatelliteAngCoord: Other rotary coordinate ??? Spindle Spindle	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
index	**Index for SYSREAD and SYSWRITE commands** Input: 0 to 999 Default index for axes X=1, Y=2, Z=3, A=4, B=5, C=6, U=7, V=8, W=9 and spindle S=10.	RESET/ LEVEL3	
relatedAxis (optional)	**Assigned physical axis** The axis name must be specified only if the keyname of the programmable axis does not correspond to the keyname of the physical axis. Input: A string of max. 18 characters	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
PhysicalAxis Physical description of the axes			
Keynames of axes They can be taken from 'MP_CfgAxes/axisList' (e.g., Xaxis).			
CfgAxis General description of an axis; The parameter object "CfgAxis" must be configured for each axis or spindle.			
isAng	**Rotary axis** Note: Rotary axes are not allowed as principal axes. Input: TRUE: Rotary axis FALSE: Linear axis (no rotary axis) Default: FALSE	RESET/ LEVEL3	
isModulo	**Modulo display** Modulo limit of 360 degrees for the position display of rotary axes. Input: TRUE: Modulo display from 0 to 360 [°] FALSE: No modulo display Default: FALSE	RESET/ LEVEL3	
axisHw	**Hardware to which the axis is connected** Input: None: No hardware CC: CC controller unit (e.g., CC 600, CC 424) Analog: Analog interface Default: CC	RESET/ LEVEL3	
axisMode MP10 (expanded)	**Operational mode of the axis** Input: NotAllowed: Axis not allowed NotActive: Axis not present Active: Axis physically present Virtual: Virtual axis for superimposed movements Display: Axis is only displayed (without motor) Default: active	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
testMode	**Axis in test mode** Input: TRUE: Test mode for commissioning (i.e., the axis needs not be connected) FALSE: Axis must be electrically connected if MP_axisMode = Active. Default: FALSE	RESET/ LEVEL3	
parList	**List of all parameter blocks of this axis** Multiple parameter blocks can be created for one axis. The keyname is used to select the desired block. Example: ParSatzX-0 (parameter block for X axis). Format: Array [0–9] Input: Keyname of max. 18 characters	RESET/ LEVEL3	
realAxis	**Keyname of the associated real axis** The axis name must be entered only for virtual axes. Input: A string of max. 18 characters	RESET/ LEVEL3	
CfgAxisPropKin Description of special axis properties; Definition of different properties that are important for kinematics and are not defined in **MP_CfgAxis.**			
specKinCoordSys	**Type of special coordinate system** Indicates whether the assigned coordinate transformation is used for defining a fixed translation axis or a datum (DefPoint). Enter this "axis" in 'MP_CfgAxes/ specCoordSysList'. Input: - FixedTransAxis: Translation axis for which no physical axis exists (the Y axis of a grinding machine, for example, is represented by using the X and Z axes). - DefPointTrans: Coordinate system in the kinematics model to which no physical axis is assigned (e.g., for defining auxiliary coordinate systems). - DefPointRot: Same as DefPointTrans, but for rotation axes.	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
kindOfRotAxis	**Type of rotational axis, only for rotary axes** Input: RollOver: Axis can rotate completely NotRollOver:Axis has limited angle of rotation	RESET/ LEVEL3	
presetToAlignAxis	**Controls the treatment of the preset for rotation axes** If the attribute is not set or set to TRUE (default), the offset from the preset is subtracted from the axis value before the kinematics calculation. If the attribute is set to FALSE, the offset only affects the position display of the axis. Input: TRUE: Offset is subtracted FALSE: Offset only affects the display Default: TRUE	RESET/ LEVEL3	
hasSpecAxisData	**Special axis data available, only for special axes** Input: TRUE: Data are available FALSE: No special axis data available Default: FALSE	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgRollOver Configuration of a rollover axis; Rotary axes that are able to execute more than one rotation—ideally as many as required—are configured as rollover axes.			
shortestDistance MP7682	**Rollover axis moves by shortest angle to the target position** The "shortestDistance" mode has the effect of the rotary axis moving by the shortest angle to the target position if non-incremental programming is used (<= 180°). Input: on: Shortest path off: Usual behavior (like linear axis) Default: off	RUN/ LEVEL1	
startPosToModulo	**Rollover axis moves start angle into the range of 0–360°** The "startPosToModulo" is only effective if "shortestDistance" is switched off. It causes the position of the rotary axis to be limited to the range from 0 to 360° at the beginning of each positioning block. Input: on: Start position within 0–360° off: Not active Default: off	RUN/ LEVEL1	
CfgMachDatumExtra Definition of a fixed machine reference point Datum for positioning blocks with M92 (e.g., for tool-change position).			
distFromMachDatum MP950	**Position of the machine reference point for M92** Distance between the machine reference point and the machine datum Input: -100 000.0 to + 100 000.0 [mm]	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
ParameterSets Parameter blocks for axes			
Keynames of the parameter blocks They can be taken from 'MP_CfgAxis/parList' (e.g., ParSatzX-0).			
CfgAxisHardware Parameters of the axis hardware; Configuration of wiring and encoder connections.			
signCorrActualVal MP210, MP3140	**Reversal of the algebraic sign of the position encoder signal** Reverse counting direction of the actual value Input: on: Reverse the counting direction off: No sign reversal Default: off	RESET/ LEVEL3	
signCorrNominalVal MP1040, MP3110	**Sign reversal of the speed command signal** Reverse the counting direction of the speed command signal Input: on or off Default: off	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
posEncoderType	**Position measurement by position encoder or motor encoder** Evaluation of the encoder type. Input: - MC distance-coded: Distance-coded position encoder on the MC (X01 to X06 and X35 to X38) - MC not distance-coded: No distance-coded linear encoder on the MC - MC EnDat: Position encoder with EnDat interface on the MC - no encoder: No position measurement - CC motor encoder: Motor encoder on the CC (X15 to X20 and X80 to X83 or X201 to X210) - CC distance-coded: Distance-coded position encoder on the CC - CC not distance-coded: No distance-coded linear encoder on the CC - CC motor EnDat: Motor encoder with EnDat interface on the CC - CC extern EnDat: External EnDat encoder on the CC	RESET/ LEVEL3	
distPerMotorTurn MP1054	**Travel of one motor revolution** Input: 0.001 to 1 000.000 [mm] or [°] Default: 5.000 [mm] **On the 6000i this parameter has no effect! This parameter should be set to 1.**	RESET/ LEVEL3	
posEncoderDist MP331	**Distance for number of signal periods from MP_posEncoderIncr** Enter 360° for spindles. Entry for multiturn encoders with EnDat interface: Distance per encoder revolution Input: 0.001 to 1 000 000 [mm] or [°] Default: 5.000 [mm]	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
posEncoderIncr MP332, MP3142	**Number of signal periods for distance from MP_posEncoderDist** Number of increments of external encoder. For spindles, enter the line count of the encoder used. Entry for multiturn encoders with EnDat interface: Increments per encoder revolution Input: 1 to 100 000 [incr.] Default: 2 048 [increments]	RESET/ LEVEL3	
posEncoderRefDist MP334	**Nominal increment between two fixed reference marks** Number of signal periods between the reference marks. The attribute only applies for encoders with distance-coded reference marks. Input: 1 to 65 535 [incr.] Default: 1 000 [increments]	RESET/ LEVEL3	
posEncoderInput MP110, MP111	**Assignment of a position encoder input to the axis** Input: none, X01 to X06 and X35 to X38 on the CC 600, X201 to X210 on the CC 424 Default: none	RESET/ LEVEL3	
posEncoderSignal MP115.0, MP116.0	**Signal amplitude at position encoder input** Input: 1 Vpp or 11 µA Default: 1 Vpp	RESET/ LEVEL3	
posEncoderFreq MP115.2, MP116.2	**Input frequency of position encoder inputs** Input: fast: 350 kHz slow: 33 kHz Default: Fast	RESET/ LEVEL3	
posEncoderResistor MP115.1, MP116.1	**Terminating resistor at position encoder input** Input: without: Without resistor 120 ohms: With resistor Default: without	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
driveOffGroup (optional)	**Circuit of X150/X151** Note: This parameter is required only for axes with digital control (CC). Format: Array [0–7] Input: None or Group 1 to Group 8	RESET/ LEVEL3	
checkPhiFieldRef (optional)	**Adjustment for non-adjusted EnDat encoders** EnDat encoders must be adjusted during commissioning. Note: This parameter is required only for axes with digital control (CC). Input: on: Adjustment off: No adjustment required Default: off	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgPosControl Position control parameters			
kvFactor MP1510, MP1810, MP3440	**Kv factor (proportional component of position controller)** Input: 0.000 to 1 000.000 [1/s] Default: 0.000 [1/s]	RUN/ LEVEL3	
servoLagMin1 MP1410	**Minimum for following-error monitoring (clearable)** Position monitoring in operation with velocity feedforward control This value applies for constant feed rates and clearable error messages. Input: 0.0010 to 30.0000 [mm] Default: 1.000 [mm]	RUN/ LEVEL3	
servoLagMax1 MP1420	**Maximum for following-error monitoring (clearable)** Position monitoring in operation with velocity feedforward control This value applies during changes in feed rate and clearable error messages. Input: 0.0010 to 30.0000 [mm] Default: 5.000 [mm]	RUN/ LEVEL3	
servoLagMin2	**Minimum for following-error monitoring (emergency stop)** Position monitoring in operation with velocity feedforward control This value applies for constant feed rates and emergency-stop error messages. Input: 0.0010 to 30.0000 [mm] Default: 1.000 [mm]	RUN/ LEVEL3	
servoLagMax2 MP1420	**Maximum for following-error monitoring (emergency stop)** Position monitoring in operation with velocity feedforward control This value applies during changes in feed rate and emergency-stop error messages. Input: 0.0010 to 30.0000 [mm] Default: 5.000 [mm]	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
feedForwardFactor MP1396, MP1391 MP1392	**Factor for velocity feedforward control** Configuration of velocity feedforward in all modes of operation. Vout = Kv*following error + Vnominal*fFF Input: 0.000 to 1.500 Default: 1.000	RUN/ LEVEL3	
controlOutputLimit	**Control-variable limit for the position controller** Used only during switch-on of position control without actual-to-nominal value transfer. Input: 0.000 to 1 666.000 [mm/min] Default: 0.000 [mm/min]	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgAxisAnalog Description of an analog axis or spindle			
analogOutput MP120, MP121	**Speed command output of axis or spindle** Input: - **none:** No servo-controlled axis - **Analog Output 1** to **Analog Output 6** : Analog output at terminal X8 Default: **Analog Output 1**	RUN/ LEVEL3	
analogOffset MP1080	**Offset on analog axis** Offset that is only effective for analog axes. Input: 0.0 to 1.0 [V] Default: 0.0 [V]	RUN/ LEVEL3	
kvFactor2 (optional)	**Proportional component of position controller above MP_kvSpeedLimit** Parameter for analog axes only. Default: 0.0 **Note:** The unit of measurement of the kv factor differs from the one used by iTNC controls! Unit of measurement:1/s iTNC unit of measurement:m/(min · mm) Therefore: iTNC kv factor · 16.66 = 6000i kv factor	RUN/ LEVEL3	
kvSpeedLimit (optional)	**Limit velocity for MP_kvFactor2** Parameter for analog axes only. Input: [mm/min] Default: 0.0 [mm/min]	RUN/ LEVEL3	
maxFeedAt9V	**Velocity at 9 volts** Input: [mm/min] Default: 4 999.98 [mm/min]	RUN/ LEVEL3	
accForwardFactor	**Factor for acceleration feedforward control** Parameter for analog axes only. Default: 0.0	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
compStrength (optional)	**Strength of the compensation** Specify the surface of the compensation curve that is superimposed on the nominal speed command during a reversal of the traverse direction. Input: [mm] Algebraic sign: - 0 : No compensation - Positive: Compensation is effective in the direction of acceleration. - Negative: Compensation is effective in the direction opposite to acceleration.	RUN/ LEVEL3	
compWidth (optional)	**Duration of the compensation** Specify (with respect to **MP_compTimeOffset** = 0) the distance to the reversal point within which the compensation curve is superimposed on the nominal speed command. Input: [mm] Default: 0.001	RUN/ LEVEL3	
compTimeOffset (optional)	**Time offset of the compensation** Specify the velocity of the axis at which the compensation curve reaches its maximum. Algebraic sign: - 0 : The compensation curve reaches its maximum at the time of direction reversal. - Positive: The compensation curve is delayed and therefore reaches its maximum after the direction reversal. - Negative: The compensation curve is output earlier and therefore reaches its maximum before the actual direction reversal.	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
compFFAdjust (optional)	**Acceleration compensation** This parameter takes into account that the required compensation strength depends on acceleration during direction reversal. The value indicates by how much the compensation strength defined in **MP_compStrength** is corrected during negligibly low acceleration. Input: [mm] Algebraic sign: - 0 : The compensation strength is constant over all acceleration values. - Positive: The compensation strength is increased during low acceleration. - Negative: The compensation strength is decreased during low acceleration.	RUN/ LEVEL3	
compRefAcc (optional)	**Reference acceleration** The value entered is used for adjusting the acceleration (**MP_compFFAdjust**). The compensation strength entered in **MP_compStrength** is used for the given acceleration. Input: [m/s^2]	RUN/ LEVEL3	
noOffsetAdjust (optional)	**Exclude axis from offset adjustment** Analog axes for which this parameter is set to TRUE are excluded from the automatic offset adjustment by code number. Input: – TRUE Axis is excluded from offset adjustment. – FALSE Offset of this axis is adjusted.	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgAxisHandwheel Configuration of the handwheel for this axis			
input	**Handwheel connector at encoder input** Input: off, or X01 to X06 Default: off	RESET/ LEVEL3	
distPerRevol	**Traverse per handwheel revolution** Input: [mm] Default: 1.0 [mm]	RESET/ LEVEL3	
incrPerRevol	**Increments per revolution of handwheel at encoder input** Input: Number of increments Default: 1 024	RESET/ LEVEL3	
rasterPerRevol	**Detent steps per revolution of handwheel at encoder input** Input: Number of detent steps Default: 100	RESET/ LEVEL3	
encoderSignal	**Signal amplitude at position encoder input for handwheel** Input: 1 Vpp, 11 µA or TTL Default: 1 Vpp	RESET/ LEVEL3	
encoderFreq	**Input frequency of position encoder input for handwheel** Input: Fast or slow Default: Fast	RESET/ LEVEL3	
encoderResistor	**Terminating resistor of position-encoder input for handwheel** Input: Without or 120 ohms Default: without	RESET/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgFeedLimits Definition of axis velocities and acceleration; For rotary axes and spindles, the velocity is specified in [°/min] and the acceleration in [1000°/s^2].			
minFeed MP3240.1	**Minimum axis feed rate** For rotary axes and spindles, the velocity is specified in [°/min]. Input: 0.0 to 36 000 000.0 [mm/min] Default: 0 [mm/min]	RUN/ LEVEL3	
maxFeed MP1010, MP3515	**Maximum axis feed rate (rapid traverse)** For rotary axes and spindles, the velocity is specified in [°/min]. Input: 0.0 to 36 000 000.0 [mm/min] Default: 16 000.2 [mm/min]	RUN/ LEVEL3	
rapidFeed MP1010	**Rapid traverse in manual mode** Maximum axis feed rate in manual mode, using the rapid traverse key. For rotary axes and spindles, the velocity is specified in [°/min]. Input: 0.0 to 36 000 000.0 [mm/min] Default: 4 999.98 [mm/min]	RUN/ LEVEL3	
manualFeed MP1020	**Maximum manual feed rate** In the Electronic Handwheel mode, this feed rate is multiplied by MP_CfgHandwheel/feedFactor. Input: 0.0 to 36 000 000.0 [mm/min] Default: 4 999.98 [mm/min]	RUN/ LEVEL3	
maxAcceleration MP1060, MP3411	**Maximal permissible axis acceleration** Input: 0.0 to 1 000.0 [m/s^2] Default: 3.0 [m/s^2] In **maxAcceleration** and **maxDeceleration**, the same value must be entered!	RUN/ LEVEL3	
maxDeceleration	**Brake ramp for handwheel movement to MP_swLimitSwitch** Input: 0.0 to 1 000.0 [m/s^2] Default: 3.0 [m/s^2] In **maxAcceleration** and **maxDeceleration**, the same value must be entered!	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
M19MaxSpeed MP3520.1	**Maximum rotational speed limit for M19** Input: 1 000 to 20 000 [1/min] Default: 1 000 [1/min]	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgReferencing Axis parameters for the reference run; For rotary axes and spindles, the velocity is specified in [°/min].			
refType MP1350	**Sequence for finding the reference mark** Input: - **Switch, changing Dir**: For linear axes with motor encoder; reference run with NC start - **Switch, no changing Dir**: For linear axes with motor encoder; reference run with NC start - **without Switch**: For spindle, rotary table with angle encoder; reference run with NC start - **distance coded**: For distance-coded linear encoders; reference run with NC start - **distance coded + on the fly**: For distance-coded linear encoders; reference run with axis-direction keys - **without Switch + on the fly**: For spindle; reference run with M3, M4, or NC start - **EnDat Encoder**: For axes with EnDat interface; reference-mark traverse not necessary Default: **Switch, changing Dir**	REF/ LEVEL3	
refPosition MP960, MP3430	**Position of machine datum** Position is defined relative to the scale reference point. For encoders with distance-coded reference marks, relative to the zero reference mark. For EnDat encoders, relative to the absolute encoder datum. Input: -100 000.0 to + 100 000.0 [mm] Default: 100.0 [mm]	REF/ LEVEL3	
refSwitchActive	**Active level of the trip dog for reference end position** Input: **high** or **low** Default: **high**	REF/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
endatDiff	**Permissible difference of EnDat encoders during switch-on** Input: -100 000.0 to + 100 000.0 [mm] Default: 0.0 [mm]	REF/ LEVEL3	
refFeedLow MP1330	**Low speed when finding the reference mark** Depending on MP_refType, the low velocity is used for finding the reference mark. Input: 10.0 to 36 000 000.0 [mm/min] Default: 600.0 [mm/min]	REF/ LEVEL3	
refFeedHigh MP1330	**High speed when finding the reference mark** Depending on 'MP_refType', the high velocity is used for finding the reference mark. Input: 80.0 to 36 000 000.0 [mm/min] Default: 1200.0 [mm/min]	REF/ LEVEL3	
refDirection MP1320	**Traversing direction for finding the reference mark** Input: **Positive** or **negative** Default: **Negative**	REF/ LEVEL3	
moveAfterRef	**Activate movement after finding the reference mark** Activate positioning after reference-mark traverse Input: **On** or **off** Default: **Off**	REF/ LEVEL3	
moveAfterRefAbs	**Absolute movement after finding the reference mark** Absolute or incremental positioning after finding the reference mark Input: **Absolute:** Absolute positioning **Relative:** Incremental positioning Default: **Absolute**	REF/ LEVEL3	
moveAfterRefPos	**Position for positioning after finding the reference mark** Input: -100 000.0 to + 100 000.0 [mm] Default: 0.0 [mm]	REF/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
moveAfterRefFeed	**Feed rate for positioning after finding the reference mark** Input: 10.0 to 36 000 000.0 [mm/min] Default: 6000.0 [mm/min]	REF/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgPositionLimits Axis parameters for setting the traverse ranges; Only one range of traverse is supported at present.			
swLimitSwitchPos MP910	**Positive software limit switch** If the positive and negative software limit switches both = 0, monitoring is switched off. Input: -100 000.0 to + 100 000.0 [mm] Default: 0.0 [mm]	RUN/ LEVEL3	
swLimitSwitchNeg MP920	**Negative software limit switch** If the positive and negative software limit switches both = 0, monitoring is switched off. Input: -100 000.0 to + 100 000.0 [mm] Default: 0.0 [mm]	RUN/ LEVEL3	
lubricationDist MP4050	**Path-dependent lubrication of axis** Input: 0.000 to + 100 000.000 [mm] 0 = no output of lubrication pulse to PLC Default: 100.0 [mm]	RUN/ LEVEL3	
CfgControllerAuxil Miscellaneous parameters for the position controller			
driveOffLagMonitor	**Following-error monitoring with drive switched off** Only for hanging axes; the value from MP_servoLagMax2 is monitored Input: on or off Default: off	RUN/ LEVEL3	
checkPosStandstill MP1110	**Standstill monitoring** (gross positioning error x D) Input: 0.000 to 100 000.000 [mm] Default: 10 000.000 [mm]	RUN/ LEVEL3	
checkPosDiff	**Calculation of position difference between position/speed encoders for oscilloscope** Input: on or off Default: off	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgEncoderMonitor Hardware monitoring functions of the position encoders			
checkAbsolutPos MP20.0	**Monitoring the absolute position with distance-coded encoder** Monitoring the absolute position for position encoders with distance-coded reference marks Input: **On** or **off** Default: **Off**	RESET/ LEVEL3	
checkSignalLevel MP20.1	**Monitoring the encoder amplitude** Monitoring the amplitudes of the position encoders. Input: **On** or **off** Default: off	RESET/ LEVEL3	
checkFrequency MP20.2	**Monitoring the edge separation of the position encoders** Input: **On** or **off** Default: off	RESET/ LEVEL3	
checkRefDistance MP2221	**Monitoring the reference mark of the spindle speed encoder** Monitoring the datum-datum distance of the spindle Input: **On** or **off** Default: **Off**	RESET/ LEVEL3	
movementThreshold MP1140	**Threshold above which the motion monitoring functions** Input: 0.0 to 36 000 000.0 [mm/min] 0 = monitoring switched off Default: 600 000.0 [mm/min]	RESET/ LEVEL3	
lagTolerance (optional)	**Tolerance at and above which the following error is included** Input: Tolerance in [mm] Default: 0		

MP Axes	Function and input	Reaction/ Access	Page
CfgControllerTol Position and speed tolerances in the servo control			
posTolerance MP1030, MP3420	**Positioning window** Control window for message **IN POSITION**. Input: 0.0000 to 360.0000 [mm] Default: 0.0050 [mm]	RUN/ LEVEL3	
timePosOK	**Hysteresis time reached for positioning window** Input: 0.000 to 20.000 [s] Default: 0.010 [s]	RUN/ LEVEL3	
speedTolerance	**Rotational speed (feed rate) window** Control window for message **SPEED REACHED**. Input: [mm/min] Default: 3 000.0 [mm/min]	RUN/ LEVEL3	
timeSpeedOK	**Hysteresis time for monitoring the speed deviation** Input: [s] Default: 0.010 [s]	RUN/ LEVEL3	
syncTolerance	**Angle tolerance for spindle synchronism** Input: [mm] Default: 0.01 [mm]	RUN/ LEVEL3	
timeSyncOK	**Hysteresis time for spindle synchronism** Input: [s] Default: 0.010 [s]	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgSpindle Special spindle parameters To be entered only for axis that is configured as a spindle.			
fastInputType	**Treatment of the fast input for the spindle** Input: - none: No function - forStopping: With M19 without encoder, the spindle is positioned to this input signal. - forReferencing: (Not yet supported) Default: none	RUN/ LEVEL3	
fastInput	**Number of the fast input for the spindle on X42** PLC input (I0 to I31) which is used for M19 or referencing a spindle. Input: 0 to 31	RUN/ LEVEL3	
zeroPosEdge (optional)	**Edge evaluation** Cam edge indicating the position of 0° for positive direction of spindle rotation. Input: zeroOne: Transition from zero to one zeroOne: Transition from one to zero	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgPositionFilter Configuration of this axis's position command filter; Define the shape of the filters and the cutoff frequencies.			
filter1Shape	**Form of the first nominal position value filter** Input: Off: Switched off Average: Mean value Triangle: Triangle HSC: High Speed Cutting Default: off	RUN/ LEVEL3	
filter1LimitFreq	**Limit frequency of the first nominal position value filter** Input: 10 to 100 [Hz] Default: 66 [Hz]	RUN/ LEVEL3	
filter2Shape	**Form of the second nominal position value filter** Input: Off, average, triangle or HSC Off: Switched off Average: Mean value Triangle: Triangle HSC: High Speed Cutting Default: HSC	RUN/ LEVEL3	
filter2LimitFreq	**Limit frequency of the second nominal position value filter** Input: 10 to 100 [Hz] Default: 66 [Hz]	RUN/ LEVEL3	
manualFilterOrder	**Order of mean-value filter in Manual mode** This position command filter is used in the "Manual" and "Electronic Handwheel" modes. Input: 1 to 51 Default: 11	RUN/ LEVEL3	

MP Axes	Function and input	Reaction/ Access	Page
CfgLaAxis Axis-dependent parameters for look-ahead; Please keep in mind that the axis jerk is added to the path jerk (which may also act in the same direction). Use 'MP_axFilterErrWeight' to consider the behavior of rotary axes with a large effective radius.			
axJerk MP1097	**Maximum axis jerk** Input: 0.0 to 1 000 000.0 [m/s^3] Default: 0.1 [m/s^3]	RUN/ LEVEL3	
axFilterErrWeight	**Factor for filter error (for rotary axes)** Input: 0.01 to 100.00 [factor] Default: 1.00 (for linear axes)	RUN/ LEVEL3	
CfgAxisComp Parameters for axis compensations; Note: Backlash compensation type 2 is not supported.			
active	**(Switch all axis compensations on/off)** Input: on: Active off: Not active	RUN/ LEVEL3	
backLash MP710	**Backlash compensation; Backlash outside of the control loop** Input: -1.0000 to +1.0000 [mm]	RUN/ LEVEL3	
linearCompValue MP720	**Linear axis error compensation** Input: -1.000 to +1.000 [mm/m]	RUN/ LEVEL3	
compType MP730	**Selection of linear/nonlinear axis error compensation** Input: Linear or non-linear	RUN/ LEVEL3	

KeySynonym

Definition of synonym names:

MP KeySynonym	Function and input	Reaction/ Access	Page
CfgKeySynonym Definition of a synonym name If parameter objects with the same content but different names are needed, define a synonym name. The keyname of this object is the synonym name. Examples: - Parameter objects for simulation channel - Further parameter blocks for axes - etc.			
Synonym names Example: K1Sim for the simulation channel			
relatedTo	**Reference to keyname** The synonym name refers to the keyname specified here. The data of the parameter object with the keyname are used for the parameter object with the synonym name. Input: A string of max. 18 characters	RESET/ LEVEL3	

Section 4 - Modules and PLC Operands

The following topics are described in this section:

- **Overview of Modules**
- **Overview of the PLC Operands**

Overview of Modules

Module	Function	Page
9000/ 9001	Copy in the marker or word range	7–216
9002	Read inputs of PLC input/output unit	6–99
9004	Read the edges of PLC inputs	6–100
9005	Set the outputs of PLC input/output unit	6–100
9006	Set and start PLC timer	7–26
9007	Read the diagnosis information of a PLC input/output unit	6–95
9010/ 9011/ 9012	Read in the word range	7–217
9019	Size of the processing stack	7–120
9020/ 9021/ 9022	Write in the word range	7–218
9025	Writing a value as a BCD code to eight successive markers	7–219
9034	Load a machine parameter subfile	3–25
9035	Read NC status information • Function 9: Read assigned handwheel axis • Function 19: Display active line of the *.CMA file • Function 26: Read jog increment	 • 6–83 • 5–67 • 6–106
9036	Write NC status information • Function 6: Select the handwheel axis • Function 7: Set the handwheel transmission ratio • Function 10: Limit value for jog increment	 • 6–81 • 6–81 • 6–105
9038	Read status information of axes	5–37
9040	Reading of axis coordinates by the PLC in the format 1/1000 (0.001) mm	5–39, 7–220
9041	Reading of axis coordinates by the PLC in the format 1/10000 (0.0001) mm	5–40, 7–221
9042	Reading of spindle coordinates by the PLC in the format 1/1000 (0.001) degrees	7–222
9044	Reading of spindle coordinates by the PLC in the format 1/10000 (0.0001) degrees	7–223
9050	Conversion of binary numbers –> ASCII	7–253

Module	Function	Page
9051	Conversion of binary numbers –> ASCII	7–254
9052	Conversion of ASCII numbers –> binary	7–255
9053	Conversion from binary –> ASCII/hexadecimal	7–256
9054	Conversion from ASCII/hexadecimal –> binary	7–257
9055	Convert time (binary) to formatted string	6–114
9060	M function status	6–35
9061	Status of non-modal M functions	6–35
9070	Copy a number from a string	7–193
9071	Find the string length	7–194
9072	Copy a byte block into a string	7–195
9073	Copy a string into a byte block	7–196
9084	Display PLC error messages with additional data	6–59
9085	Display PLC error messages	6–60
9086	Delete PLC error messages	6–61
9087	Status of PLC error messages	6–62
9088	Status display of M functions	6–36
9091	Find the line number of a tool in the tool table	6–125
9092	Search for an entry in the tables selected for execution (.T/.D/ .TCH)	6–119
9093	Read data from tables selected for program (.T/.D/.TCH)	6–121
9094	Write data into a tool and datum table	6–122
9095	Activate axis error compensation	5–66
9096	Delete a line in the tool table	6–123
9100	Assign data interface	8–20
9101	Release data interface	8–21
9102	Status of data interface	8–22
9103	Transmit string through data interface	8–23
9104	Receive string through data interface	8–24
9105	Transmit binary data through data interface	8–25
9106	Receive binary data through data interface	8–26
9107	Read from receiving buffer	8–27
9111	Receive a message via LSV2	7–224
9112	Transmit ASCII characters via data interface	8–28
9113	Receive ASCII characters via data interface	8–29
9120	Position PLC axis	5–44
9121	Stop PLC axis	5–46
9122	Status of PLC axis	5–47
9123	Traverse the reference marks of PLC axes	5–48
9124	Feed rate override for PLC axis	5–49

Module	Function	Page
9125	Stop PLC axis at next Firth grid position	5-50
9130	Output the analog voltage	6–103
9133	Interrogate the values, internal ADCs	5–146
9137	Read diagnostic information of the IEB 404	6–96
9138	Read analog input of IEB 404	6–101
9139	Monitoring functions for the IEB 404 PLC input/output units	6–98
9140	Set axis-specific feed-rate limit	7–225
9141	Read axis-specific feed-rate (status)	7–226
9145	Actual-to-nominal value transfer	5–122
9147	Assigning a reference value to an axis	7–227
9166	Read current utilization of drive motor	5–148
9171	Start of a spindle orientation with adjustable parameters	7–228
9180	Keystroke simulation	6–66
9181	Disable NC key by PLC	6–66
9182	Re-enable NC key by PLC	6–67
9183	Disable NC key group by PLC	6–67
9184	Enable locked NC keys groups by PLC	6–68
9186	Call a soft key function	6–69
9187	Status query of a soft key call	6–69
9189	Shut down the control	6–9
9190	Start the PLC operating hours counter	6–109
9191	Stop the PLC operating hours counter	6–110
9192	Transfer the operating hours counter	6–110
9193	Set the operating hours counter	6–111
9194	Alarm when operating time exceeded	6–112
9195	Transfer the real-time clock	6–114
9196	Find the PLC cycle time	7–2
9197	Start the cycle timer	7–27
9220	Traverse the reference marks	5–83
9221	Start PLC positioning movement	5–54
9222	Interrogate PLC positioning status	5–55
9223	Free rotation (not for NC axes during program run)	6–2
9224	Stop PLC positioning movements	5–55
9231	Compensation of thermal expansion	5–71
9240	Open a file	7–49
9241	Close a file	7–50
9242	Positioning in a file	7–51
9243	Reading from an ASCII file line by line	7–52
9244	Writing to an ASCII file line by line	7–53

Module	Function	Page
9245	Read a field in a table	7–54
9246	Write to a field in a table	7–56
9247	Search for a condition in a table	7–58
9248	Copy, rename, or delete file	7–230
9249	Read and reset variable “errno”	7–59
9255	Read a data field in a table	7–55
9256	Write to a data field in a table	7–57
9260	Receiving events and waiting for events	7–202
9261	Sending events	7–204
9262	Context change between spawn processes	7–205
9263	Interrupting a spawn process for a defined time	7–205
9264	Wait for a condition	7–206
9270	Read OEM-defined string value	7–231
9271	Write OEM-defined string value	7–232
9275	Write ASCII data into the log file	6–63
9276	Write operand contents into error log file	6 –64
9277	Writing data into the OEM log	7–233
9279	Shut down control (configurable)	6–10
9291	Starting an NC macro	7–234
9300	Locking and releasing the pocket table	7–236
9301	Find the number of an entry in the pocket table	7–237
9302	Search for a vacant pocket in the tool magazine	7–237
9304	Copying OEM values from the pocket table	7–238
9305	Moving tools in the pocket table	7–239
9306	Moving tools between magazines	7–240
9321	Find the current block number	6–23
9322	Information of the current NC program	7–241
9340	Searching for a pocket depending on magazine rules	7–242
9341	Editing a pocket table depending on magazine rules	7–243
9342	Find magazine and pocket number	7–244
9343	Compilation and activation of magazine rules	7–245
9350	Read data from the tool table	7–245
9351	Write data to a tool table	7–247
9404	Start movement when an NC stroke is present	6–24
9405	Convert a symbolic operand into a numerical PLC operand	7–21
9407	Give default tool number for an NC channel	7–248
9410	Read spindle status	5–156
9411	Read the actual spindle values (speed, coordinates)	5–41, 7–249

Module	Function	Page
9412	Stop the spindle	5–157
9413	Move the spindle	5–158
9414	Position the spindle	5–160
9416	Select gear range and assigned settings for spindle	7–250
9417	Set default shaft speed for spindle	7–251
9418	Set status for spindle	7–252
9430	Change numeric value of a machine parameter	3–31
9431	Read numeric value of a machine parameter	3–32
9432	Change string value of a machine parameters	3–33
9433	Read string value of a machine parameters	3–34
9434	Select parameter block	5–134
9435	Status of the parameter block of an axis	5–135
9440	SQL: Open a transaction	7–75
9441	SQL: Conclude and close a transaction	7–76
9442	SQL: Seek a record in the result set	7–77
9443	SQL: Fetch a record from the result set	7–78
9444	SQL: Change a record in the result set	7–79
9445	SQL: Read a single value from a table	7–80
9447	SQL: Delete record from result set	7–81
9448	SQL: Load a column description	7–82
9449	SQL: Extract a value from a comma separated list	7–83
9450	SQL: Execute an SQL statement	7–84
9451	SQL: Roll back and close a transaction	7–85
9452	SQL: Seek next record in the result set of a query	7–86
9453	SQL: Fetch binary data from the result set of a query	7–87
9454	SQL: Update binary data in the result set of a query	7–88
9455	SQL: Read a single numeric value from a table	7–89
9458	SQL: Unload a column description	7–90
9459	SQL: Change or insert a value in a comma separated list	7–91

Overview of the PLC Operands

The following topics are described:

- **PLC Operands of the “General Data” Group**
- **PLC Operands of the “Operating Mode Group” Group**
- **PLC Operands of the “Machining Channels” Group**
- **PLC Operands of the “Axis” Group**
- **PLC Operands of the “Spindle” Group**

PLC Operands of the “General Data” Group

	Marker	Description	Page
	General – Control configuration		
D	NN_GenOmgCount	Number of configured groups of operating modes	–
D	NN_GenChnCount	Number of configured machining channels	–
D	NN_GenAxCount	Number of configured logical axes (including spindles)	5–4
D	NN_GenSpiCount	Number of configured spindles	5–151
	General – Control status		
D	NN_GenOmgManual	Selected operating mode for manual operation	6–26
D	NN_GenChnManual	Selected machining channel for manual operation	6–26
D	NN_GenSpiManual	Selected spindle for manual operation	6–26
M	NN_GenCycleAfterPowerOn (M4172)	1st PLC scan after power on	6–5
M	NN_GenCycleAfterPlcStop (M4173)	1st PLC scan after interruption of the PLC program	6–5
M	NN_GenCycleAfterReConfig (M4174)	1st PLC scan after changing of the configuration data	6–5
M	NN_GenNcInitialized (M4184)	Control is being initialized (after start-up cycles)	6–5
M	NN_GenNcEmergencyStop (M4178)	Control in “external emergency stop” state	5–149
	General – Error handling		
M	NN_GenApiModuleError (M4203)	An error occurred during use of an API module.	5–38
D	NN_GenApiModuleErrorCode (W1022)	Error code that occurred during use of an API module.	5–38

	Marker	Description	Page
	General – Key information		
D	NP_GenKeyCode (W274)	Code of the depressed key	7–4
M	PP_AxHandwheelLocked (M4576)	Disable handwheel motion	6–72
D	NP_GenSoftkeyHori (W302)	Code of the horizontal soft key last pressed	–
D	NP_GenSoftkeyVert (W304)	Code of the vertical soft key last pressed	–
	General – Touch probe		
M	NN_GenTchProbeReady (M4050)	Touch probe: Ready (hardware signal)	–
M	NN_GenTchProbeDeflected (M4052)	Touch probe: Stylus deflected (hardware signal)	–
M	NN_GenTchProbeBatteryLow (M4054)	Touch probe: Battery low (hardware signal)	–
M	NN_GenTchProbeX13 (M4060)	Touch probe: X13 active (if 0, X12 is active)	–
	General – Safe control		
M	NN_GenSafetyInputs (M4264)	Safety oriented: Inputs 0 to 15 bit-encoded	–
M	NN_GenSafetyStopActive (M4280)	Safety oriented: Stop is activated	–
M	NN_GenSafetyPermButtonsPressed (M4281)	Safety oriented: Both permissive buttons are pressed	–
M	NN_GenSafetyStartCutoutChannel (M4282)	Safety oriented: Beginning of the test channel	–
M	NN_GenSafetyCutout (M4283)	Safety oriented: Switch off, since no switch-off test was performed	–
M	NN_GenSafetyFeedLimitActive (M4284)	Safety oriented: Limitation of the feed rate or spindle infeed active	–

PLC Operands of the “Operating Mode Group” Group

	Marker	Description	Page
	OMG – Configuration		
D	NN_OmgChnCount	Number of machining channels for this group of operating modes	–
D	NN_OmgChn	Array of the machining channels for this group of operating modes	–
	OMG – Operating modes		
M	NN_OmgManual (M4150)	Manual Operation operating mode	6–14
M	NN_OmgHandwheel (M4151)	Electronic Handwheel operating mode	6–14
M	NN_OmgMdi (M4152)	Positioning with Manual Data Input operating mode	6–14
M	NN_OmgProgramSingle (M4153)	Program Run, Single Block operating mode	6–14
M	NN_OmgProgramRun (M4154)	Program Run, Full Sequence operating mode	6–14
M	NN_OmgProgramReference (M4155)	Reference operating mode	6–14
M	NN_OmgProgramDiagnosis	Diagnosis operating mode	6–14
M	NN_OmgJogIncrement (M4579)	Jog Positioning operating mode	6–14
	OMG – Program run		
M	PP_OmgNcStart	NC start for all machining channels of this group of operating modes	6–15
M	PP_OmgNCStop	NC stop for all machining channels of this group of operating modes	6–15

PLC Operands of the “Machining Channels” Group

	Marker	Description	Page
	Channel – Configuration		
D	NN_ChnAxisCount	Number of axes of this machining channel	5–7
D	NN_ChnAxis	Array of the axes of this machining channel	5–7
	Channel – Error handling		
M	NN_ChnErrorWarning	Error or warning occurred	6–25
M	NN_ChnErrorFStop (M4220)	Feed rate stopped because of an error	6–25
M	NN_ChnErrorNCStop (M4221)	NC stop because of an error	6–25
M	NN_ChnErrorCancel (M4223)	Program canceled because of an error	6–25
M	NN_ChnErrorEmergencyStop (M4222)	Emergency stop because of an error	6–25
M	NN_ChnErrorReset	Reset because of an error	6–25
	Channel – Program run		
M	PP_ChnNcStart (M4564)	NC start or cycle on	6–18
M	PP_ChnNCStop (M4560)	NC stop or cycle off	6–21
M	PP_ChnNcStartExternRequest	External request for an NC start	6–18
M	PP_ChnNCStopExtern	NC stop or cycle off	6–21
M	NN_ChnControlInOperation (M4176)	Control is in operation	6–22
M	NN_ChnProgStoppedAsync	Asynchronous NC program interruption	6–21
M	NN_ChnProgStopped	NC program interruption	6–21
M	NN_ChnProgCancel	NC program cancellation	6–23
M	NN_ChnProgEnd	End of NC program reached	6–20
M	NN_ChnAutostart (M4182)	Autostart function: Request for program start	6–20
M	NN_ChnAutostartTimeExpired	Autostart function – Request for program start	6–20
M	PN_ChnAutostartEnable (M4586)	Enabling the autostart	6–19
M	NN_ChnBlockScan (M4158)	Mid-program startup (or block scan) active	6–23

	Marker	Description	Page
M	NN_ChnBlockScanStrobeTransfer (M4161)	Restore status at block scan (M/S/T/Q transfer)	6–23
M	NN_ChnProgManTraverse (M4156)	Manual traverse of the axes active (for lathes: Inspection operation)	6–22
M	NN_ChnProgReturnContour (M4157)	Return to contour active (after manual traverse or block scan)	6–22
	Channel – Touch probe		
M	NN_ChnTchProbeCycle (M4053)	Touch probe: Touch probe cycle active	–
M	PP_ChnTchProbeMonitor (M4055)	Touch probe: Enabling the probing process	–
	Channel – Feed rate		
D	NN_ChnProgFeedMinute (D360)	Programmed feed rate per minute [mm/min]	5–119
D	NN_ChnProgFeedRevolution	Programmed feed rate per revolution [mm/rev]	5–119
D	NN_ChnProgFeedThread	Programmed thread feed rate [mm/rev]	5–119
M	NN_ChnProgMinuteActive	Feed per minute active	5–119
M	NN_ChnProgRevolutionActive	Feed rate per revolution active	5–119
M	NN_ChnProgThreadActive	Thread feed rate is active	5–119
D	NN_ChnContourFeed (D388)	Current contouring feed rate [mm/min]	5–119
D	PP_ChnContourFeedMax (D596)	Max. feed rate from PLC [mm/min]	5–119
D	NN_ChnFeedOverrideInput (W494)	Feed-rate override set [%]	6–92
D	PP_ChnFeedOverride (W766)	Feed-rate override set by the PLC [%]	6–92
D	NN_ChnRapidFeedOverrideInput (W496)	Rapid traverse override set [%]	6–93
D	PP_ChnRapidFeedOverride (W752)	Rapid traverse override set by the PLC [%]	6–93
D	PP_ChnConfigOverride	Configurable override (e.g., rapid traverse)	–
M	PP_ChnFeedEnable	Feed-rate enable for all axes	5–123
M	PP_ChnWorkFeedEnable	Rapid traverse enable for all axes	5–123

	Marker	Description	Page
	Channel – Status		
M	NN_ChnToolLifeExpired (M4543)	Tool life 1 expired	6–124
M	PP_ChnEnableAxisKeyLatch	Enabling latching of axis-direction key	6–70
M	PP_ChnRapidTraverseKey (M4561)	Rapid-traverse key	6–70

PLC Operands of the “Axis” Group

	Marker	Description	Page
	Axis – Configuration		
D	NN_AxLogNumber	Logical axis number (corresponds to the axis number from “Axes of the machining channel”)	5–4
	Axis – Drive		
M	NN_AxDriveReady	Axis drive is ready	5–120
M	PP_AxDriveOnRequest (CM9161)	Switch axis drive on	5–120
M	NN_AxDriveOn (CM9162)	Axis drive is on (and at least speed-controlled)	5–120
M	PP_AxPosControlRequest (W1040)	Position-control the axis	5–120
M	NN_AxPosControl (W1024)	Axis is position-looped	5–121
M	PP_AxValueActToNominal	Actual-to-nominal value transfer	5–122
M	NN_AxCorrectingLagError	Following error eliminated	5–122
M	PP_AxClampModeRequest (W1038)	Prepare to open the position control loop	5–121
	Axis – Control		
M	NN_AxReferenceAvailable (W1032)	Reference mark not yet traversed	5–92
M	PP_AxReferenceEndPosition (W1054)	Reference end position	5–92
D	PP_AxManualFeedMax	Maximum manual axis feed rate [mm/rev]	5–119
M	PP_AxFeedEnable (W1060)	Axis-specific feed-rate enable	5–123
M	NN_AxInMotion (W1028)	Axes in motion	5–145
M	NN_AxInPosition (W1026)	Axes in position	5–121
M	PP_AxTraversePos (W1046)	Manual traverse in positive direction	5–119
M	PP_AxTraverseNeg (W1048)	Manual traverse in negative direction	5–119

	Marker	Description	Page
M	PP_AxHandwheelLocked (W1062)	Disable handwheel motion for specific axes	6–72
M	PP_AxDeactivateMonitoring (W1042)	Deactivate monitoring functions	5–137
M	NN_AxLubricationPulse (W1056)	Lubrication pulse: Value from MP_lubricationDist exceeded	5–43
M	PP_AxLubricationDistReset (W1058)	Reset the accumulated distance	5–43

PLC Operands of the “Spindle” Group

	Marker	Description	Page
	Spindle – Configuration		
D	NN_SpiLogNumber	Logical axis number of the spindle	5–151
	Spindle – Drive		
M	NN_SpiDriveReady	Spindle drive is ready	5–155
M	PP_SpiDriveOnRequest (CM9161)	Switch spindle drive on	5–155
M	NN_SpiDriveOn (CM9162)	Spindle drive is on (and at least speed-controlled)	5–155
	Spindle – Control		
M	NN_SpiReferenceAvailable (M4018)	Reference position found	5–152
M	PP_SpiReferenceMarkSignal	Trip dog for reference end position	5–152
D	PP_SpiSpeedMax (D604)	Maximum speed of the spindle	5–155
M	PP_SpiEnable (M4008)	Spindle enabling	5–155
M	NN_SpiInMotion (M4002)	Spindle in motion	5–155
M	NN_SpiSpeedOK (M4001)	Spindle speed reached	5–159
M	NN_SpiControl	Spindle position-looped	5–151
M	NN_SpiControlInPos (M4000)	Spindle in position	5–151
M	NN_SpiSyncSpeed	Speed synchronism active	–
M	NN_SpiSyncAngle	Angle synchronism active	–
M	NN_SpiSyncReached	Synchronous operation reached	–
D	NN_SpiOverrideInput (W492)	Speed override set [%]	6–90
D	PP_SpiOverride (W764)	Speed override set by the PLC [%]	6–90
M	NN_SpiTapping (M4030)	Tapping active	5–161
M	NN_SpiRigidTapping (M4031)	Tapping with spindle interpolated with Z axis active	5–161

Section 5 - Configuring the Axes and Spindle

The following topics are described in this section:

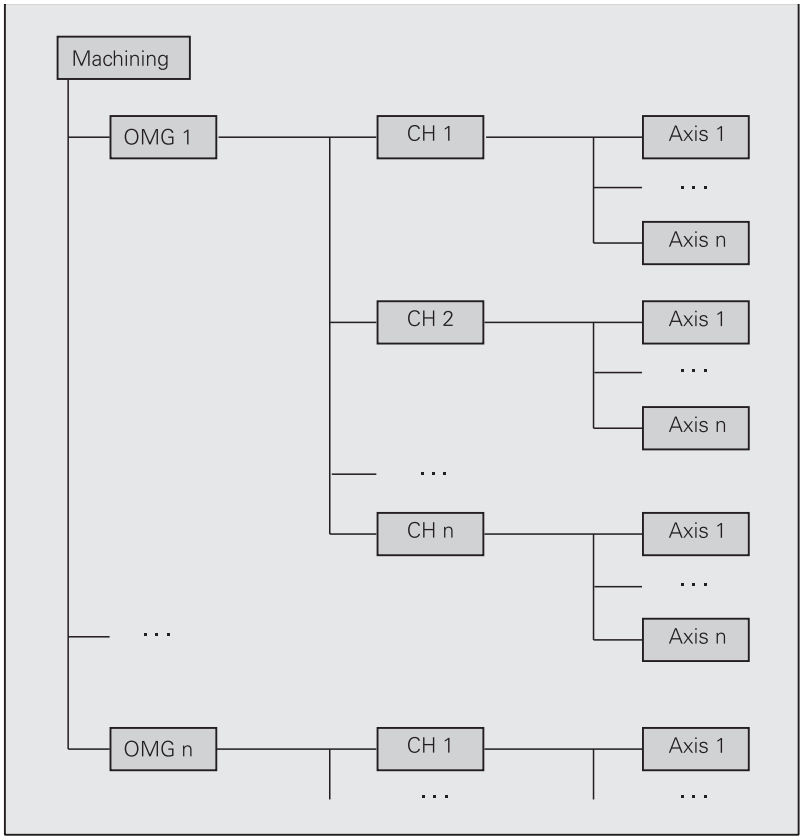
- **Machine Structure**
- Configuration of Machining Channels
- Configuration of Axes
- Encoders
- Analog Axes
- Reading Axis Information
- Traverse Ranges
- Lubrication Pulse
- Controlling Axes by PLC (PLC Axes)
- Axis Error Compensation
- Machine Kinematics
- Reference Marks
- The Control Loop
- Monitoring Functions
- Spindles
- Integrated Oscilloscope

Machine Structure

The following topics are described:

- Adapting the Control to Machine Structure
- Definition of Axes

Adapting the Control to Machine Structure



Legend:

- OMG: Operating mode group (OperatingModeGroup)
- CH: Machining channel (NC channel)
- Axis: Axis or spindle

Use the following organizational structure to configure the machine structure in the control:

- The machine consists of operating mode groups (OperatingModeGroups).
- Every operating mode group manages machining channels.
- Every machining channel manages axes.

The operating mode groups of a machine as well as the machining channels of an operating mode group operate independently of each other.

All machines have this organizational structure – even if a simple machine requires only one OMG and one channel.

A structure for simulation corresponding to the corresponding to the physical structure of the machine must be configured. As a rule, the machine structure and the structure for simulation have the same configuration.

Definition of Axes

Settings in the configuration editor:	
System CfgAxes **axisList** **specCoordSysList**	

The following topics are described:

- **Defining the Logical Axes**
- **Spindles**
- **Special Kinematics Axes**

Defining the Logical Axes

Within the geometry and interpolator processing, a unique identifier (=logical axis number) must be assigned to each axis. The identifier corresponds to the index in **MP_axisList.** Logical axes are defined by sequential numbering starting from the index [0].

The definition of the logical axes is independent of their assignment to the machining channels.

Enter the axes of all channels of the machine, including spindles and PLC axes.

MP_axisList

Key names for all axes on the machine

Format: Array [0–9] (# = logical axis number)

Input: String of max. 18 characters

Default:
[0]: X (logical axis number 0)
[1]: Y (logical axis number 1)
[2]: Z (logical axis number 2)
[3]: S (logical axis number 3)
[4]: A (logical axis number 4)
[5]: B (logical axis number 5)
[6]: U (logical axis number 6)

The PLC indicates the number of configured logical axes in **NN_GenAxCount**. **NN_AxLogNumber** contains the logical axis number for using general modules (such as Module I 9165). The axis number corresponds to the index from **NN_ChnAxis** (see "**Configuring a Machining Channel**").

PLC operand	Type
NN_GenAxCount **Number of configured logical axes (including spindle)**	D
NN_AxLogNumber **Logical axis number (identical to axis number of "axes of the machining channel")**	D

Spindles

In the control software, spindles and axes are treated in largely the same way. Spindles are considered a special kind of axis and are defined as a logical axis.

For parameters, PLC operands and spindle functions, see "**Spindles**".

Special Kinematics Axes

Axes that are used in the kinematics model but are not entered in **MP_CfgAxes/axisList** are defined in **MP_specCoordList.**

In MP_specCoordList, enter the axes for which one of the following attributes is defined in **MP_CfgAxisPropKin/specKinCoordSys** (see "**Virtual Axis, Kinematics Properties of Axes**"):

- FixedTransAxis
- DefPointTrans
- DefPointRot

The special kinematics axes are indicated by sequential numbering starting with the index [0].

MP_specCoordSysList

Key names of special axes for the kinematics description

Format: Array [0–9]

Input: String of max. 18 characters

Configuration of Machining Channels

The following topics are described:

- **Configuring a Machining Channel**
- **Traversing the Reference Marks**
- **Moving to Restore Position**

Configuring a Machining Channel

Settings in the configuration editor:	
Channels ChannelSettings Key for channel CfgChannelAxes **progAxis** **grindAxis** **kinModels** CfgChannelType **type** CfgNcErrorReaction **warningLevel** CfgNcPgmParState **persistent** **currentSet**	

With the 6000i, two channels are permanently defined in “Key for channel”:

Ch-Nc
Machining channel

Ch-Sim
Simulation channel

This setting cannot be changed.

The following topics are described:

- **Type of NC Channel**
- **Axes of Machining Channel**
- **Kinematics of Machining Channel**
- **Error Behavior of Machining Channel**
- **Saving Q/QS Parameters**

Type of NC Channel

Machining channels that are assigned physical axes are defined by the **Main** type. Machining channels without physical axes are defined by the **Internal** type. These channels are normally used for special tasks (e.g., for the calculation of superimposed contours).

MP_type

Type of machining channel

Format: Drop-down selection menu

Selection:

[Main]

Normal channel

[Internal]

Channel for special applications, such as noncylindrical grinding

Axes of Machining Channel

In the parameter object **CfgChannelAxes,** you specify the axes of the machining channel (NC channel) and define the behavior of the axes during reference run.

In **MP_progAxis,** enter the axes which can be used within the NC program. Axes that are **not** included are, for example, slave axes in master-slave operation or axes that are for display only.

MP_progAxis

Programmable axes

Format: Array [0–7]

Input: Key names from MP_CfgAxes/axisList

Default:
[0]: Axis-X1
[1]: Axis-Y1
[2]: Axis-Z1
[3]: Axis-A1
[4]: Axis-C1

MP_grindAxis

Axes of the grinding generator

Format: Array [0–5]

Input: Subset of MP_progAxis in random order

In **NN_ChnAxisCount,** the NC informs the PLC of the number of axes assigned to this machining channel. The axes assigned to this machining channel are indicated in the array **NN_ChnAxis**.

PLC operand	Type
NN_ChnAxisCount **Number of axes of machining channel**	D
NN_ChnAxis **Array of axes of machining channel** Only the axes (and not the spindles) are entered. The order of the entries has no meaning.	D

Kinematics of Machining Channel

In **MP_kinModels,** enter the kinematics models of the machining channel. After control start-up, the last entry will be activated.

MP_kinModels

Keys for the kinematics models available to this channel

Format: Array [0–15]

Input: Key names from MP_CfgKinModels

Error Behavior of Machining Channel

The parameter **MP_warningLevel** specifies the behavior when FN14 errors occur. Errors are triggered only if according to the PET table the warning level of the error is maximally as high as the warning level set here. Note that errors with warning level 0 are always triggered and errors with warning level 5 are never triggered.

MP_warningLevel

Warning level of channel

Format: Numerical value

Input: 0 to 4

0: FN14 errors with warning level = 0 are triggered

1: FN14 errors with warning level </= 1 are triggered

2: FN14 errors with warning level </= 2 are triggered

3: FN14 errors with warning level </= 3 are triggered

4: FN14 errors with warning level </= 4 are triggered

Default: 0

Saving Q/QS Parameters

In the parameter object **CfgNcPgmParState,** you specify whether and where Q/QS parameters are to be stored persistently. If **MP_currentSet** is not defined, the name of the machining channel is used as name for the Q/QS parameter block.

MP_persistent

Define the storage of Q/QS parameters

Format: Drop-down selection menu

Selection:

[True]
Q/QS parameters are stored persistently

[False]
Q/QS parameters are not stored

Default: False

MP_currentSet

Name of Q/QS parameter block

Format: String

Input: Name of active Q/QS parameter block

Traversing the Reference Marks

Settings in the configuration editor:	
Channels ChannelSettings Key for channel CfgChannelAxes **refAxis** **refAllAxis**	

In **MP_refAllAxis,** you specify whether all axes are to be referenced in the sequence defined in **MP_refAxis**, or whether the reference point in these axes is to be traversed by pressing the axis-direction keys.

The automatic or MDI mode of operation cannot be used until all axes entered in **MP_refAxes** have been referenced.

MP_refAllAxes

Home all axes in succession after an NcStart

Format: Drop-down selection menu

Selection:

[True]
All axes are referenced after an NC start. The sequence is defined in MP_refAxes.

[False]
The individual axes are referenced by using the axis-direction keys. This also defines the sequence of the reference run.

MP_refAxis specifies the axes to be referenced. The sequence of the reference run is determined by the index.
Index [0] = First axis

MP_refAxes

Axes that are to be run over the reference point

Format: Array [0–5]

Input: Key names from MP_CfgAxes/axisList

Default:
[0]: Zaxis
[1]: Yaxis
[2]: Xaxis
[3]: Aaxis
[4]: Caxis

Moving to Restore Position

Settings in the configuration editor:	
Channels ChannelSettings Key for channel CfgChannelAxes **restoreAxis** CfgRestorePosition **distance**	

Moving the axes:

- After an NC stop, the axes are moved to the last interpolated position (stop position).
- During a block scan, they are moved to the calculated restore position.

To move to the restore position:

- In **MP_restoreAxis,** specify the sequence in which the axes are to move.
- In **MP_distance,** define the safety clearance.

MP_restoreAxis

Sequence for returning to the contour

Format: Array [0–5]

Input: Key names from MP_CfgAxes/axisList

MP_distance

Safety clearance when approaching the restore position

Format: Numerical value

Input: 0.000000 to 500.000000 [mm]

Default: 25.0 [mm]

Configuration of Axes

The following topics are described:

- **Axis Designations and Coordinates**
- **Programmable Axes**
- **Physical Axes**
- **Virtual Axes**

Terms used in connection with axes:

- **Logical axes** are
  - Axes transmitting signals to an encoder input.
  - Axes that are operated by transmitting signals to the PWM outputs
  - Axes that are operated by transmitting signals to analog outputs
  - Virtual axes whose output signals superimpose the signals of other axes
  - Inactive axes
- Axes whose movements are performed by other logical axes are **not logical axes.** Example: Programmed C axis whose movements are performed by the spindle.
- **Physical axes:** The parameter objects in **PhysicalAxis** describe the physical structure of each logical axis.
- Axes that can be addressed in an NC program are referred to as **programmable axes.**

Axis Designations and Coordinates

The following topics are described:

- **Properties of the Principle Axes X, Y, Z**
- **Algebraic Signs of the Axes**
- **Properties of the Rotary Axes A, B, C**
- **Properties of the Linear Axes U, V, W**
- **Standard Coordinates**

Principal, parallel, and rotary axes are distinguished.

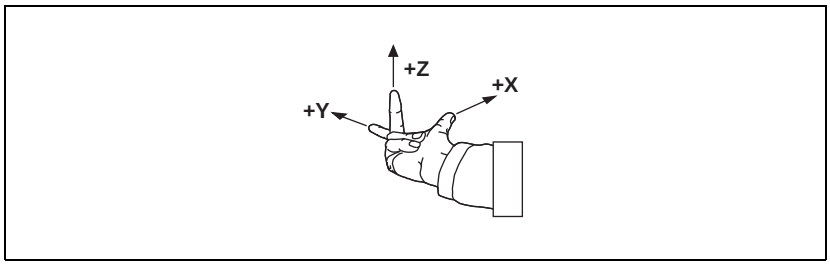
Properties of the Principal Axes X, Y, Z

X, Y, and Z axes are principal axes. These axes have a defined spatial orientation in a coordinate system model, and are always linear.

It is of no importance to the editor if the current coordinate system is that of the machine bed system, or if it is aligned otherwise.

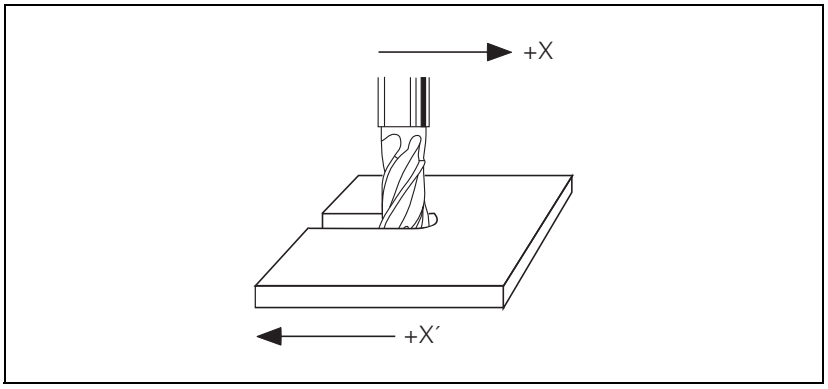
They are the principal coordinates for programming in the editor.

An easy way to remember this system is to use the “right-hand rule”:



Algebraic Signs of the Axes

When the programmer writes an NC program, he always assumes that the tool (not the workpiece) is in motion. If the machine moves its workpiece-holding element (table) in a particular axis instead of the tool, then the direction of actual motion is opposite to the direction of axis motion. In this case the direction of motion is designated with the same algebraic sign as the axis direction, but with an apostrophe: +X´, +Y´, and +Z´:

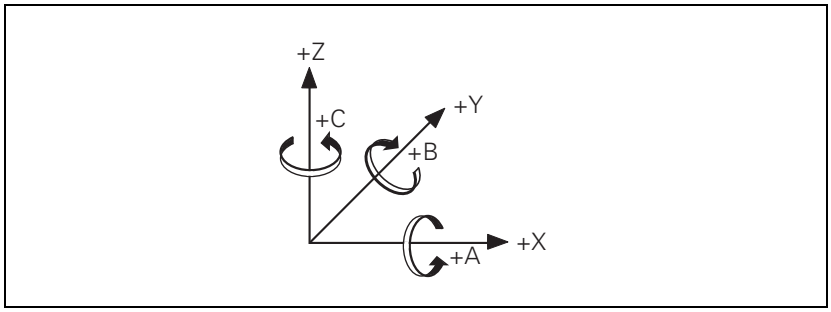


Properties of the Rotary Axes A, B, C

For rotary axes, the turning axis is in the direction of a principal coordinate. A, B, and C axes are “parallel” rotary axes.

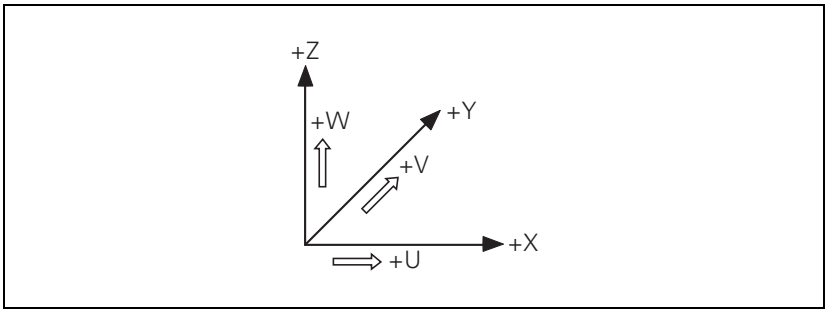
They are parallel coordinates for programming in the editor.

The directions of the rotary axes A, B and C follow the “right-fist rule.” The fingers of the closed right hand point in the proper rotation direction of an axis when the thumb points in the direction of the associated linear axis.



Properties of the Linear Axes U, V, W

The additional axes U, V, and W are parallel linear axes.
They are parallel coordinates for programming in the editor.



Standard Coordinates

The meanings of the coordinates X, Y, Z, A, B, C, U, V, and W are specified in **ISO 841**.

The control views coordinates whose **MP_axName** begins with X, Y, Z, A, B, C, U, V, or W as standardized coordinates.

For these standardized coordinates, the parameters under **CfgProgAxis** must obey the following rules:

First letter of parameter MP_axName	Parameter MP_dir	Parameter MP_progKind
X	XAxis	MainLinCoord
Y	YAxis	MainLinCoord
Z	ZAxis	MainLinCoord
U	XAxis	ParallelLinCoord
V	YAxis	ParallelLinCoord
W	ZAxis	ParallelLinCoord
A	XAxis	ParallelAngCoord
B	YAxis	ParallelAngCoord
C	ZAxis	ParallelAngCoord

Programmable Axes

Settings in the configuration editor:	
Axes CfgProgAxis Key of axis **axName** **dir** **progKind** **index** **relatedAxis**	

In the parameter object **CfgProgAxis,** define all axes that are programmable and/or are displayed. This description is independent of the assignment to NC channels.

Information about programmable axes: see “**Standard Coordinates**”.

The name entered in **MP_axName** is used for programming and in the position display.

MP_axName

Designation of the axis for position display

Format: String

Input: (e.g., A, B, C, U, V, W, X, Y, Z)

MP_dir

Spatial orientation of the axis or center of rotation

Format: Drop-down selection menu

Selection:

[XAxis]
Motion / rotary axis in X direction
[YAxis]
Motion / rotation in Y direction
[ZAxis]
Motion / rotation in Z direction
[SpecAxis]
Free/undefined spatial orientation (e.g., for spindle)

MP_progKind

Type of axis

Format: Drop-down selection menu

Selection:

[MainLinCoord]
Main coordinate, always linear (X, Y, Z)
[ParallelLinCoord]
Parallel coordinate, linear (U,V,W)
[ParallelAngCoord]
Parallel coordinate, angular (A, B, C)
[SatteliteLinCoord]
Minor coordinate, linear: Not used at present.
[SatteliteAngCoord]
Minor coordinate, angular: Not used at present.
[Spindle]
Spindle

The following topics are described:

- **Index for SysRead and SysWrite**
- **Axis Without a Separate Drive Motor**

Index for SysRead and SysWrite

In **MP_index,** you define the index for SysRead/SysWrite commands. Default index for axes X=1, Y=2, Z=3, A=4, B=5, C=6, U=7, V=8, W=9 and spindle S=10.

MP_index

Index for SYSREAD and SYSWRITE commands

Format: Numerical value

Input: 0 to 999

Axis Without a Separate Drive Motor

Enter the axis name of the assigned physical axis in **MP_relatedAxis** if the key name of the programmable axis does not correspond to the key name of the physical axis. This links the axis with the physical axis. The axes concerned are usually axes that do not have a separate drive motor.

Example: If the spindle drive is used for the C axis, you link the C axis with the “spindle” physical axis.

MP_relatedAxis

Assigned physical axis

Format: String

Input: String of max. 16 characters;
Key name of physical axis

Physical Axes

Settings in the configuration editor:	
Axes PhysicalAxis Key of axis CfgAxis **isAng** **isModulo** **axisHw** **axisMode** **testMode** **parList** **realAxis**	

In the parameter object **PhysicalAxis,** define all axes that can be instructed by the interpolator to execute a command. The description in **PhysicalAxis** is independent of the assignment to NC channels.

In the parameter object **CfgAxis,** you specify the axis type and drive interface, you assign a parameter block to the axis and define the operating mode of the axis.

MP_isAng

Rotary axis

Format: Drop-down selection menu

Selection:

[True]
This is a rotary axis.
[False]
Linear axis (no rotary axis).

In **MP_axisHw,** you define the drive interface.

With the 6000i, the default is None.

MP_axisHw

Hardware to which the axis is connected

Format: Drop-down selection menu

Selection:

[None]
No hardware connection
[InOutCC]
Connection to controller unit.
[AnalogMC]
Analog drive interface

In MP_isModulo, you define whether the modulo limit of 360 degrees applies to the position display of rotary axes.

MP_isModulo

Modulo display

Format: Drop-down selection menu

Selection:

[True]
Position display for rotary axes: modulo 360°
[False]
Position display is not a modulo display.

Note: Spindles must be configured as modulo axes: **MP_isModulo=True**.

The following topics are described:

- **Activate Axis**
- **Assigning Parameter Blocks**

Activate Axis

An axis can only be moved in a closed loop after it has been activated with MP_axisMode.

For commissioning, use MP_testMode to switch the axes to test mode. In this operation, the interpolator views the axis as a fully-functional axis, but the nominal values are not passed on to the drive motor. There is only an internal nominal-to-actual transfer. Such an axis does not need to have all the hardware connections made (position and speed input, PWM output), but can be “positioned”.

MP_axisMode

Axis operating mode

Format: Drop-down selection menu

Selection:

[Active]
Axis physically present
[Not active]
Axis does not exist
[Virtual]
Virtual axis for superimposed movements
[Display]
Axis is only displayed (without motor)

Default: Active

MP_testMode

Axis in test mode

Format: Drop-down selection menu

Selection:

[True]
Test mode for commissioning (i.e., the axis needs not be connected)
[False]
Axis must be electrically connected if MP_axisMode = Active.

Default: False

Note: Hanging axes cannot be supported in test mode. The PLC must ensure that these axes are braked in test mode.

Assigning Parameter Blocks

In MP_parList, enter the key name of the parameter block that is assigned to this axis. The parameter block describes the axis control response, the encoder connection, the encoder signals, etc.

You can create more than one parameter block for one axis. This enables you to define different controller settings, for example.

Examples:

a) You define different controller settings to ensure appropriate control response depending on the load.

b) The spindle and the C axis are realized by using a physical axis. This enables you to define separate parameter blocks for the spindle and the C axis.

MP_parList

List of all parameter blocks of this axis

Format: Array [0–9]

Input: Key name of max. 18 characters

(e.g., ParSatzX-0 (parameter block for the X axis))

Note: The first parameter block must be fully defined. In **KeySynonym/ CfgKeySynonym,** you can relate the other parameter blocks to the first one. Then you only have to define the parameters that differ from the ones of the parameter block to which you have related the present parameter block.

Virtual Axes

For virtual axes, enter the key name of the associated real axis in MP_realAxis.

Virtual axes: see “**Configuration of Axes**”.

MP_realAxis

Key name of the associated real axis

Format: String

Input: Key names from MP_CfgAxes/axisList

The following topic is described:

- **Kinematics Properties of Axes**

Kinematics Properties of Axes

Settings in the configuration editor:	
Axes PhysicalAxis Key of axis CfgAxisPropKin **specKinCoordSys** **kindOfRotAxis** **presetToAlignAxis** **hasSpecAxisData**	

In the parameter object **CfgAxisPropKin,** specify the properties important for kinematics.

In MP_specCoordSys, define whether the assigned coordinate transformation is used for defining a fixed translation axis or a datum (DefPoint).

MP_specKinCoordSys

Type of special coordinate system

Format: Drop-down selection menu

Selection:

[FixedTransAxis]
Translation axis for which no physical axis exists.

[DefPointTrans]
Coordinate system in the kinematics model of a translation axis to which no physical axis is assigned.

[DefPointRot]
Coordinate system in the kinematics model of a rotation axis to which no physical axis is assigned.

For rotary axes, specify in MP_kindOfRotAxis whether the axis concerned can rotate completely or has a limited angle of rotation.

MP_kindOfRotAxis

Type of rotation axis (only for rotary axes)

Format: Drop-down selection menu

Selection:

[RollOver]

Axis can rotate completely

[NotRollOver]

Axis has limited angle of rotation

MP_presetToAlignAxis controls the treatment of presets for rotation axes. If the attribute is set to TRUE, the offset from the preset is subtracted from the axis value before the kinematics calculation. If set to FALSE, the offset is only effective for the position display of the axis.

MP_presentToAlignAxis

Controls the treatment of the preset for rotation axes

Format: Drop-down selection menu

Selection:

[True]

Offset is subtracted.

[False]

Offset is only effective for the display.

Default: True

MP_hasSpecAxisData is only for special axis data – the parameter is not used at present.

MP_hasSpecAxisData

Special axis data available (only for special axes)

Format: Drop-down selection menu

Selection:

[True]

Special axis data available

[False]

No special axis data

Default: False

Encoders

Encoders transmit positions and movements of the machine to the control. ANILAM contouring controls operate with incremental and absolute encoders with EnDat interface.

In the parameter object **CfgAxisHardware,** define the connections of the encoders, the type of encoder, the type of signals, etc.

The following topics are described:

- **Type of Encoder**
- **Distance-Coded Reference Marks**
- **Encoder Connections**
- **Defining the Traverse Direction**
- **Encoder Monitoring**

Type of Encoder

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisHardware **posEncoderType** **distPerMotorTurn** **posEncoderDist** **posEncoderIncr**	

The parameter object CfgAxisHardware is not required for:

- Virtual axes (MP_axisMode=Virtual)

- In **MP_posEncoderType,** define the type of position measurement and the type of position encoder or speed encoder.

- In **MP_distPerMotorTurn,** define the traverse distance per motor revolution.

MP_posEncoderType

Position measurement by position encoder or motor encoder

Format: Drop-down selection menu

Selection:

[MC distance coded]

Distance-coded position encoder on the MC (X01 to X06 and X35 to X38)

[MC not distance coded]

No distance-coded position encoder on the MC (X01 to X06 and X35 to X38)

[MC EnDat]

EnDat position encoder on the MC (X01 to X06 and X35 to X38)

[no encoder]

No position measurement

[CC motor encoder]

Position measurement by speed encoder on the CC (X15 to X20 and X80 to X83)

[CC distance coded]

Distance-coded position encoder on the CC (X201 to X210)

No distance-coded position encoder on the CC (X201 to X210)

[CC Motor EnDat]

Position measurement by EnDat speed encoder on the CC (X15 to X20 and X80 to X83)

[CC extern EnDat]

External EnDat encoder on the CC (X201 to X210)

Default: CC motor encoder

MP_distPerMotorTurn

Distance of one motor revolution

Format: Numerical value

Input: 0.000 000 001 to 1 000 [mm] or [°]

Default: 5 [mm] or [°]

Note: The parameter **MP_distPerMotorTurn** is not effective for analog axes. However, you should not enter 0 for this parameter. ANILAM recommends that you enter the suggested default value.

The following topic is described:

- **Signal Period**

Signal Period

For any given distance the position encoder supplies a fixed number of signal periods. The signal is subdivided 1024 times.

For **EnDat rotary encoders,** enter the following values for calculating the signal period:

- **MP_posEncoderDist:** Distance per encoder revolution
- **MP_posEncoderIncr:** Increments per encoder revolution

Note: Ensure that the line count per rotary encoder revolution specified by the manufacturer is entered in the **MP_posEncoderIncr** parameter. This value is used for the plausibility check of the measured value. If you enter a line count that differs from the one specified by the manufacturer, the control displays an error message.
Note:
Linear encoders with EnDat interface are excepted from the plausibility check.

- For **other encoders** (not rotary encoders with EnDat interface), enter the following values for calculating the signal period:
- **MP_posEncoderDist:** Linear distance for the number of signal periods from **MP_posEncoderIncr**
- **MP_posEncoderIncr:** Number of signal periods for the distance from **MP_posEncoderDist**

For **spindles,** enter the following values for calculating the signal period:

- MP_posEncoderDist: 360°

- MP_posEncoderIncr: Line count of the encoder used

The control calculates the quotient:

$$\text{Signal period} = \frac{\text{MP\_posEncoderDist}}{\text{MP\_posEncoderIncr}}$$

MP_posEncoderDist

Distance for number of signal periods from MP_posEncoderIncr

Format: Numerical value

Input: 0.000 000 001 to 100 000 [mm] or [°]
For spindles: 360°

Default: 5 [mm] or [°]

MP_posEncoderIncr

Number of signal periods for distance from MP_posEncoderDist

Format: Numerical value

Input: 1 to 100 000 [incr.]
For spindles: Line count of the encoder used

Default: 2048 [incr.]

Distance-Coded Reference Marks

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisHardware **posEncoderRefDist**	

The parameter object CfgAxisHardware is not required for:

- Virtual axes (MP_axisMode=Virtual)

ANILAM offers linear encoders with **distance-coded reference marks**. The nominal increment between two fixed reference marks depends on the encoder being used.

- For encoders with distance-coded reference marks, enter for each axis the nominal increment between two fixed reference marks in **MP_posEncoderRefDist**.

Example:
LS 486C: Incremental linear encoder with distance-coded reference marks
Grating period 20 µm (= one signal period covers 0.02 mm), nominal increment between reference marks is 20 mm.

MP_posEncoderDist = 0.02
MP_posEncoderIncr = 1

MP_encoderDistance = $\frac{20\ \text{mm}}{0{,}02\ \text{mm}}$ =1000 [signal periods]

MP_posEncoderRefDist
Nominal distance between two fixed reference marks
Format: Numerical value
Input: 1 to 65 535 [signal periods]
Default: 1 000 [signal periods]

Encoder Connections

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisHardware **posEncoderInput** **posEncoderSignal** **posEncoderFreq** **posEncoderResistor** **speedEncoderInput** **driveOffGroup**	

The parameter object CfgAxisHardware is not required for:

Virtual axes (MP_axisMode=Virtual)

Position encoder input

The following topics are described:

- **MP_posEncoderInput**
- **Position Encoder Signal**
- **MP_posEncoderSignal**

MP_posEncoderInput

Assignment of position-encoder input to axis

Format: Drop-down selection menu

Selection:

[None]
No position encoder connected.
[X01–X06]
Position encoder inputs are X01 to X06 (on the MC).
[X35–X38]
Position encoder inputs are X35 to X38 (on the MC).
[X201–X210]
Reserved

Position Encoder Signal

Position encoders supply 1-V_{PP} or 11-μA_{PP} signals. Define the type of signal, input frequency and terminating resistance in the following machine parameters.

- MP_posEncoderSignal: 1-V_{PP} or 11-μA_{PP} signal
- MP_posEncoderFreq: Maximum input frequency
- MP_posEncoderResistor: Terminating resistor

Note: The incremental track data must be entered for the corresponding position encoder inputs for encoders with EnDat interfaces.

The parameter MP_driveOffGroup is optional and is not effective for analog axis feedback control.

MP_driveOffGroup

Circuit of X150/X151

Format: Numerical value

Selection:

[None]
Axis not assigned (disabling only through I32).
[Group1]
Axis is assigned to X150 pin 1.
[Group2]
Axis is assigned to X150 pin 2.
[Group3]
Axis is assigned to X150 pin 3.
[Group4]
Axis is assigned to X150 pin 4.
[Group5]
Axis is assigned to X151 pin 1.
[Group6]
Axis is assigned to X151 pin 2.
[Group7]
Axis is assigned to X151 pin 3.
[Group8]
Axis is assigned to X151 pin 4.

MP_posEncoderSignal

Signal amplitude at position encoder input

Format: Drop-down selection menu

Selection:

[1 Vpp]
Input signal of encoder is 1 Vpp.
[11 µA]
Input signal of encoder is 11 µA.

Default: 1 Vpp

MP_posEncoderFreq

Input frequency of position encoder inputs

Format: Drop-down selection menu

Selection:

[Fast]
Input frequency
With 1 V_{PP}: 350 kHz
With 11 μA_{PP}: 150 kHz

[Slow]
Input frequency of 33 kHz

In **MP_posEncoderResistor,** you define whether a terminating resistor is required. (120 ohms)

This parameter may be required if the encoder signals are looped through a servo drive, or if Y cables are used. It is usually sufficient to set the parameter to **Without.**

MP_posEncoderResistor

Terminating resistor at position encoder input

Format: Drop-down selection menu

Selection:

[Without]
Without resistor.
[120 Ohm]
With resistor.

Default: Without

Defining the Traverse Direction

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisHardware **signCorrActualVal** **signCorrNominalVal**	

The parameter object CfgAxisHardware is not required for:

- Virtual axes (MP_axisMode=Virtual)

For analog axes, define whether the sign of the nominal position value and/or the sign of the nominal speed value are/is to be reversed, depending on the mounting position of the encoders. Both parameters must be inverted if the traverse direction of the axis is to be reversed.

MP_signCorrActualVal

Reversal of the algebraic sign of the position encoder signal

Format: Drop-down selection menu

Selection:

[On]
Reverse the sign of the position encoder signal.
[OFF]
Do not reverse the sign of the position encoder signal.

Default: Off

MP_signCorrNominalVal

Reversal of the algebraic sign of the nominal speed value

Format: Drop-down selection menu

Selection:

[On]
Reverse the sign of the speed encoder signal.
[OFF]
Do not reverse the sign of the speed encoder signal.

Default: Off

Encoder Monitoring

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgEncoderMonitor **checkAbsolutPos** **checkSignalLevel** **checkFrequency** **checkRefDistance**	

The parameter object CfgEncoderMonitor is not required for:

- Virtual axes (MP_axisMode=Virtual)

ANILAM contouring controls monitor the signal transmission from the encoders.

The following topics are described:

- **Position Encoder**
- **Monitoring of Encoders with EnDat Interface**

Position Encoder

Activate the following monitoring functions for the position encoders:

- MP_checkAbsolutPos: Monitor the absolute position.
- MP_checkSignalLevel: Monitor the encoder amplitude.
- MP_checkFrequency: Monitor the frequency of the encoder signals.

The interpolator calculates the absolute position when a reference mark of a distance-coded encoder is crossed over. If MP_checkAbsolutPos is active, the nominal values are compared to the actual values. If deviations are found, an error message is displayed and an Emergency Stop is initiated.

MP_checkAbsolutPos

Monitoring the absolute position of distance-coded encoder

Format: Drop-down selection menu

Selection:

[On]
Monitor the absolute position.
[OFF]
No monitoring.

Default: Off

MP_checkSignalLevel

Monitoring the encoder amplitude

Format: Drop-down selection menu

Selection:

[On]
Monitor the encoder amplitude.
[OFF]
No monitoring.

Default: Off

MP_checkFrequency

Monitoring the edge separation of the position encoders

Format: Drop-down selection menu

Selection:

[On]
Monitor the encoder frequency.
[OFF]
No monitoring

Criterion	Error message
Absolute position with distance-coded reference marks	**Encoder <AXIS> DEFECTIVE**
Amplitude of encoder signals	**Encoder AMPLITUDE TOO LOW <AXIS>**
Edge separation of encoder signals	**Encoder <AXIS>: FREQUENCY TOO HIGH**

Monitoring of Encoders with EnDat Interface

In the event of a disturbance, the error message **EnDat defective <error code> <axis>** will display.

The error code is shown in hexadecimal notation. Error codes may also display combined, in which case they add themselves together.

There are two possible types of errors:

- The encoder reports an error.
- Access to the encoder via the EnDat interface is faulty.

Codes for errors reported by the encoder:

Error code	Meaning
0x00000001	Light source defective
0x00000002	Signal amplitude too small
0x00000004	Incorrect position value
0x00000008	Overvoltage
0x00000010	Undervoltage
0x00000020	Overcurrent
0x00000040	Replace battery
0x00000080	Reserved
0x00000100	Reserved
0x00000200	Reserved
0x00000400	Reserved
0x00000800	Reserved
0x00001000	Reserved
0x00002000	Reserved
0x00004000	Reserved
0x00008000	Reserved

Error codes if the access to the encoder via the EnDat interface is faulty:

Error code	Meaning
0x80010000	Delete the alarm bit
0x80020000	Read the alarm status
0x80040000	Read the number of pulses
0x80080000	Read the number of signal periods
0x80100000	Read the number of differentiable revolutions
0x80200000	Read the measuring steps
0x80400000	Read the series number
0x80800000	Read the type of encoder
0x81000000	Read the position value
0x82000000	Reserved
0x84000000	Reserved
0x88000000	Read the checksum
0x90000000	Alarm bit remains set
0xA0000000	Timeout while waiting for data - signal “high”
0xC0000000	Timeout while waiting for data - signal “low”
0x80000000	Error during access to EnDat interface

Analog Axes

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisAnalog **analogOutput** **maxFeedAt9V**	

The Parameter object

- CfgAxisHardware is not required for:
  - Virtual axes (MP_axisMode=Virtual)
- CfgAxisAnalog is not required for:
  - Virtual axes (MP_axisMode=Virtual)
  - Axes that are for display only (MP_axisMode=Display)
  - Digital axes (MP_axisHw=CC or None)

In **MP_CfgAxis/axisHw=Analog** analog closed-loop axes are defined as such and are described in the parameter object **CfgAxisAnalog** (see "**The Control Loop, Controller Parameters for Analog Axes**").

The following topics are described:

- **Analog Output**
- **Rapid Traverse for Analog Axes**

Analog Output

- In **MP_analogOutput,** define the number of the analog nominal-value output at connector X8.

MP_analogOutput

Nominal speed value output of axis or spindle

Format: Drop-down selection menu

Selection:

[None]
No analog output assigned
[Analog Output 1–6]
Analog outputs 1 to 6 (connector X8)
[Analog Output 7–12]
Analog outputs 7 to 12 (connector X9)

Rapid Traverse for Analog Axes

- In **MP_maxFeedAt9V,** enter the rapid-traverse rate to be reached at an analog voltage of 9 V (e.g., for drives reaching the rapid traverse rate at 6 V, the corresponding value at 9 V must be calculated by linear calculation).
- Adjust the rapid traverse feed rate (v_{max}) with the analog voltage at the servo amplifier.

MP_maxFeedAt9v

Velocity at 9 V

Format: Numerical value

Input: 0.0 to 36 000 000.0 [mm/min or °/min]

Default: 4 999.98

Reading Axis Information

The following topics are described:

- **Module 9038 Read status information of axes**
- **Reading the Axis Coordinates**
- **Module 9040 Reading of axis coordinates by the PLC in the format 1/1000 (0.001) mm**
- **Module 9041 Reading of axis coordinates by the PLC in the format 1/10000 (0.0001) mm**
- **Module 9411 Read the actual spindle values (speed, coordinates)**
- **Reading the Actual Spindle Values**

Module 9038 Read status information of axes

With Module 9038 you can interrogate the general status information of the axes. You can interrogate the status of a specific axis or of all axes at once.

For bit-coded information, the status request for a specific axis returns code 0 or 1. The meaning of the return codes is explained in the table below.

The status request for all axes only returns bit-coded information. The information is then passed on in the bit corresponding to the axis.

Status information	Bit information	Meaning
0	x	0: Axis is not active (MP_axisMode not equal to “Active” or no encoder) 1: Axis is active
1	x	Axis in interpolation context? 0: Axis is presently in the interpolation context or not active. 1: Axis is presently not in the interpolation context, or axis is a spindle. (Spindles are not in interpolation context.)
2	x	0: Open-loop axis 1: Closed-loop axis (MP_axisMode=Active)
3	–	Maximum temperature of the motor [°C]
4	x	0: No Hirth axis 1: Hirth axis
5	–	Hirth grid [1/10 µm]
6	–	Should not be used.
7	x	0: Linear axis or not active 1: Rotary axis (MP_isAng=True)
8	x	0: Analog axis (MP_axisHw=Analog) or not active 1: Digital axis

Call:

PS	B/W/D/K	<>Axis> Axis-specific: Index from MP_CfgAxes/axisList For all axes: –1
PS	B/W/D/K	<>Status information> See table above
CM	9038	
PL	B/W/D	<>Information> Axis-specific: Status information according to table For all axes: Bit-coded (Bit 0 corresponds to logic axis 0, etc.)

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Information was read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Status information not available
	2	Axis does not exist

Reading the Axis Coordinates

- Read the axis coordinates with Module 9041.

The values are saved in double words beginning at the given address.

The values for all axes are read in, regardless of whether individual axes are excluded through the machine configuration. Values for excluded axes are undefined.

The coordinate value of an axis remains undefined until the reference point of an axis has been traversed.

Module 9040 Reading of axis coordinates by the PLC in the format 1/1000 (0.001) mm

Module 9040 loads the axis coordinates from the control loop for all NC axes. The actual values in the reference system, the servo lag, the distance-to-go and the deflection of a triggering touch probe can be loaded.

The values are saved in 10 double words in the format 1/1000 mm, beginning at the given target address.

The module is only supported if you use the 6000i compatible programming interface (API 1.0).

Note: This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls. This module is not supported if the symbolic programming interface is used. Use Module 9041 instead.

Possible errors:

- The argument for the type of coordinate is outside the permitted range (2).
- The specified target address is not a double word address (4).
- The double word block cannot be written to the specified target address (4)
- You are using the symbolic programming interface.

Call:

PS	K/B/W/D	<Target address Dxxxx>
PS	K/B/W/D	<Type of coordinate> 2: Actual values in the reference system 3: Following error 4: Distance-to-go 5: Deflection (measuring touch probe) 6: Actual values in the datum system
CM	9040	

Error recognition:

Marker	Value	Meaning
M4203	0	Data was read
	1	Faulty call data

Module 9041 Reading of axis coordinates by the PLC in the format 1/10000 (0.0001) mm

Module 9041 loads the axis coordinates from the control loop for all NC axes. The actual values in the reference system, the servo lag, the distance-to-go and the deflection of a triggering touch probe can be loaded.

The values are saved in 10 double words in the format 1/10000 mm, beginning at the given target address.

Possible errors:

- The argument for the type of coordinate is outside the permitted range (2).
- The specified target address is not a double word address (4).
- The double word block cannot be written to the specified target address (4)

Call:

PS	K/B/W/D	<Target address Dxxxx>
PS	K/B/W/D	<Type of coordinate> 2: Actual values in the reference system 3: Following error 4: Distance-to-go 5: Deflection (measuring touch probe) 6: Actual values in the datum system
CM	9041	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data was read
	1	Faulty call data

Module 9411 Read the actual spindle values (speed, coordinates)

The position and speed values of the spindle are transferred.

This module is supported only by the new symbolic programming interface (API). If you are using the 6000i compatible storage interface, the module returns the error code 99.

Call:

PS	K/B/W/D	<>logic spindle number> (e.g., 0 for spindle 1, 1 for spindle 2)
PS	K/B/W/D	<>desired spindle information> 1: Actual position 2: Nominal position 3: Actual position in the reference system 4: Lag error 5: Distance-to-go 10: Actual speed 11: Nominal speed
CM	9411	
PL	D	<>spindle information> About 1 to 5: Value in 0.0001° About 10 to 11: Value in 0.0001 rpm

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Actual spindle value has been read.
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Requested spindle number or spindle information is invalid.
	99	Module is not supported (control operates with 6000i compatible API).

Reading the Actual Spindle Values

Read the spindle coordinates with Module 9411.

Note: You can use this module ony if you are working with the new symbolic API, see “**Definition of Axes**”.

Traverse Ranges

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgPositionLimits **swLimitSwitchPos** **swLimitSwitchNeg**	

The parameter object CfgPositionLimits is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

Define the software limit switches in the parameter object **CfgPositionLimits:**

- The datum is the machine datum (MP_refPosition).
- If the geometry detects that a software limit switch will be traversed, the traverse path concerned will not be executed and an error message will be displayed.
- If a software limit switch has been traversed, the control passes into the Emergency Stop state.
- The software limit switches can usually be overwritten from the NC program (see below).
- Limit-switch monitoring can be deactivated by entering 0 for positive and negative limit values.

MP_swLimitSwitchPos
Positive software limit switches
Format: Numerical value
Input: –100 000.000 000 000 to +100 000 [mm] or [°]
Default: 0 [mm] or [°]

MP_swLimitSwitchNeg
Negative software limit switches
Format: Numerical value
Input: –100 000.000 000 000 to +100 000 [mm] or [°]
Default: 0 [mm] or [°]

Note: If the positive and negative software limit switches both = 0, monitoring is switched off.

Software limit switches can be overwritten with **FN17:SYSWRITE** (e.g., for automatic tool change. This function is effective only until the next GOTO command (GOTO key or FN9 to FN12) or the end of the program).

Lubrication Pulse

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgPositionLimits **lubricationDist**	

The parameter object CfgPositionLimits is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

- In **MP_lubricationDist,** you define the traverse distance at which the lubrication pulse for the axis guideways is to be output. The NC reports in **NN_AxLubricationPulse** when the entered distance in an axis has been exceeded.
- Reset **PP_AxLubricationDistReset** after lubrication. This resets the distance counter to 0.

Note: After the control has been reset, the accumulated distance is reset.

MP_lubricationDist

Path-dependent lubrication of axis

Format: Numerical value

Input: 0.000 000 000 to + 100 000 [mm] or [°]
0= no output of lubrication pulse to PLC

Default: 100 [mm] or [°]

PLC operand	Type
NN_AxLubricationPulse **Lubrication pulse: Value in MP_lubricationDist exceeded** 0: Value not exceeded 1: Value exceeded	M
PP_AxLubricationDistReset **Resetting the accumulated distance** 0: Do not reset accumulated distance 1: Reset accumulated distance	M

Controlling Axes by PLC (PLC Axes)

Axes that are **not** in an interpolation context, can be used by the PLC as required. The PLC can start more than one axis simultaneously, but they are not interpolated with each other.

Note: The axis interpolation context can by changed by the PLC at any time. Example: By activating another kinematics model.

The following topic is described:

- **Stopping/Starting Axes by PLC**

Stopping/Starting Axes by PLC

The following topics are described:

- **Module 9120 Position PLC axis**
- **Module 9121 Stop PLC axis**
- **Module 9122 Status of PLC axis**
- **Module 9123 Traverse the reference marks of PLC axes**
- **Module 9124 Feed rate override for PLC axis**
- **Positioning of Axes by PLC**
- **Module 9125 Stop PLC axis at next Hirth grid position**
- **Tool-change Sequences are Defined via Scripts**
- **Spindle-gear Switching**
- **Module 9221 Start PLC positioning movement**
- **Module 9222 Interrogate PLC positioning status**
- **Module 9224 Stop PLC positioning movements**

Module 9120 Position PLC axis

The module positions an axis. The target position and feed rate are transferred in the module call.

The axis is positioned regardless of any other processes in the control. In particular, there is no interpolation with other axes.

Constraints:

- The axis must **not** be in an interpolation context.
- The parameter values for rapid traverse, acceleration, etc. must be set correctly.
- Rotary axes are positioned in the direction of the shortest path, except if the target position was transferred as an incremental value.
- Software limit switches are not effective.
- The axis must be stationary. Any positioning movement must be aborted beforehand with Module 9121.
- Feed-rate override is disabled.
- If no reference mark has been traversed, the positioning process builds on the counter value as it was upon switch-on.
- If Modules 9120, 9121, and 9122 are called more than once for the same axis during one PLC scan, only the last command is transferred.

- The “Positioning error” status set in this axis is cleared. The status must be evaluated by Module 9122.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Target position> Input unit: [0.0001 mm]
PS	B/W/D/K	<>Feed rate> Input unit: [mm/min]
PS	B/W/D/K	<>Mode> Bit 0 – Definition of the target position: 0: Absolute (i.e., relative to the machine datum) 1: Incremental
CM	9120	
PL	B/W/D	<>Error code> 0: No error. Positioning was started. 1: Axis does not exist 2: Axis is still in interpolation context 3: Axis is already being positioned 4: Absolute position is outside of modulo range 5: Programmed axis not in closed loop

Module 9121 Stop PLC axis

The module stops a positioning movement that has been started by Module 9120 or 9123.

Constraints:

- If Modules 9120, 9121, and 9122 are called more than once for the same axis during one PLC scan, only the last command is transferred.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
CM	9121	
PL	B/W/D	<>Error code> 0: Positioning is canceled 1: Axis does not exist 2: Axis is still in interpolation context 3: Axis was already stationary

Module 9122 Status of PLC axis

The module provides information on the present operating status of the axis.

Constraints:

- Status changes through a PLC positioning command (Modules 9120, 9121, 9123) are not detected until the next PLC scan.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
CM	9122	
PL	B/W/D	<>Status> Bit 0 – Axis in interpolation context ? 0: Axis does not exist or in interpolation context 1: Axis is not in interpolation context Bit 1 – Reference mark 0: Reference mark not yet traversed 1: Reference mark traversed Bit 2 – Positioning 0: Positioning inactive 1: Positioning active Bit 3 – Direction of motion 0: Positive direction of motion 1: Negative direction of motion Bit 4 – Positioning error 0: No positioning errors occurred 1: Positioning error Bit 5 – Closed-loop or open-loop axis 0: Closed-loop axis 1: Open-loop axis Bit 6 – Target position reached? 0: Target position not yet reached 1: Target position reached

Module 9123 Traverse the reference marks of PLC axes

The module starts a positioning movement in a defined direction. The positioning movement is continued until a reference mark is found or until the positioning movement is canceled by Module 9121.

Note: Use Module 9123 only if no conventional procedure for traversing the reference marks is possible.

Constraints:

- The axis must **not** be in an interpolation context.
- The parameter values for rapid traverse, acceleration, etc. must be set correctly.
- Software limit switches are not active.
- The axis must be stationary. Any positioning movement must be interrupted beforehand with Module 9121.
- Feed-rate override is disabled.
- If Modules 9120, 9121, and 9122 are called more than once for the same axis during one PLC scan, only the last command is transferred.
- The “Positioning error” status set in this axis is deleted.
- The “Find reference point” status is set for the axis.
- Any pre-existing reference point in this axis is cleared, but the numerical axis value remains. It will not be reinitialized until the reference point is found.
- The positioning movement is interrupted as soon as the reference point is found. However, due to the braking distance, the axis comes to a standstill somewhat beyond the reference mark.

Call:

PS	B/W/D/K	<>Axis>
		Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Feed rate>
		Input unit: [mm/min]
PS	B/W/D/K	<>Mode>
		Bit 0: Direction of traverse
		0: Positive
		1: Negative
CM	9123	
PL	B/W/D	<>Error code>
		0: No error. Positioning was started.
		1: Axis does not exist
		2: Axis is still in interpolation context
		3: Axis is already being positioned
		5: Programmed axis not in closed loop

Module 9124 Feed rate override for PLC axis

The override value set in this module influences the traversing speed of an axis traversed by the PLC with Module 9120 or 9123.

Constraints:

- The axis must **not** be in an interpolation context.
- The override value is transferred as an integer (0 to 10 000), which may be in the range from 0% to 100.00% (resolution 0.01%).
- The last transmitted override value is accounted for at the beginning of movement.
- After a reset or interruption of the PLC program the override value is set to 100.00%.
- The override value can also be changed during a positioning movement.
- The module can be called in addition to a module from the group (9120/9121/9123) during the same PLC scan.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Override> Input unit: 0 to 10 000, corresponds to 0 to 100% in 0.01% steps.
CM	9124	
PL	B/W/D	<>Error code> 0: No error, override value was set 1: Axis does not exist 2: Not a PLC axis 3: Override value incorrect

Positioning of Axes by PLC

You start a PLC positioning movement with Module 9221, and you can interrogate the status with Module 9222.

The following conditions apply to a PLC positioning command:

- Tool compensation is not included. Before a PLC positioning command you must end any tool compensation.
- A PLC positioning movement is not displayed in the test graphics.

The NC cancels a PLC positioning movement under the following conditions:

- If in the Manual or Handwheel modes, there is an NC STOPP
- If in the automatic operating modes, there is an NC STOPP and “internal stop”
- An EMERGENCY STOP
- An error message that results in a STOPP

Module 9125 Stop PLC axis at next Hirth grid position

Stop an already started PLC-positioning of an axis at the next Hirth grid position.

Call:

PS	B/W/D/K	<Axis> 0 to 8 represent axes 1 to 9
CM	9125	
PL	B/W/D	<Error code> 0: Positioning is canceled 1: Axis does not exist 2: Not a PLC axis 3: Axis was already stationary 4: Axis is not a Hirth axis

Tool-change Sequences are Defined via Scripts

Effective immediately, new channel-specific machine parameters are available for configuration of the tool-change sequences. You must enter the names of the script files in these machine parameters. The script defines the tool-change sequence for each type of changer. ANILAM has already made four pre-defined script files available, which you must enter in the optional machine parameters under **CfgPlcToolChange** as required:

Settings in the configuration editor:	
Channels ChannelSettings Ch-Nc CfgPlcToolChange **sequT0Text** **sequT0Tint** **sequT0TintS** **sequTextT0** **sequTextText** **sequTextTint** **sequTextTintS** **sequTintT0** **sequTintT0S** **sequTintText** **sequTintTextS** **sequTintTint** **sequTintTintS** **sequTintTintF** **sequTintTintFS** **followUpS** **followUptT** **updateTable**	

Explanations to the new optional machine parameters for configuring the tool-change sequences:

Parameters	Description
sequT0Text	Name of a script file given under CfgOemScript for loading an external tool.
sequT0Tint	Name of a script file given under CfgOemScript for loading an internal tool.
sequT0TintS	Name of a script file given under CfgOemScript for loading an internal tool that requires special handling.
sequTextT0	Name of a script file given under CfgOemScript for unloading an external tool.
sequTextText	Name of a script file given under CfgOemScript for unloading and loading an external tool.
sequTextTint	Name of a script file given under CfgOemScript for unloading an external tool and loading an internal tool.
sequTextTintS	Name of a script file given under CfgOemScript for unloading an external tool and loading an internal tool that requires special handling.
sequTintT0	Name of a script file given under CfgOemScript for unloading an internal tool.
sequTintT0S	Name of a script file given under CfgOemScript for unloading an internal tool that requires special handling.
sequTintText	Name of a script file given under CfgOemScript for unloading an internal tool and loading an external tool.
sequTintTextS	Name of a script file given under CfgOemScript for unloading an internal tool that requires special handling and loading an external tool.
sequTintTint	Name of a script file given under CfgOemScript for unloading an internal tool and loading an external tool. Neither has a fixed pocket in the tool magazine.
sequTintTintS	Name of a script file given under CfgOemScript for unloading an internal tool and loading another internal tool. Neither has a fixed pocket in the tool magazine, but at least one requires special handling.
sequTintTintF	Name of a script file given under CfgOemScript for unloading an internal and loading another internal tool. At least one tool has a fixed pocket in the tool magazine.
sequTintTintFS	Name of a script file given under CfgOemScript for unloading an internal tool and loading another internal tool. At least one tool has a fixed pocket in the tool magazine, and at least one tool requires special handling.
followUpS	Name of a symbolic PLC operand or the number of a logical marker: the marker is set if an other strobe with an S code follows in the tool-change sequence.
followUpT	Name of a symbolic PLC operand or the number of a logical marker: the marker is set if an other strobe with a T code follows in the tool-change sequence.
updateTable	Name of a symbolic PLC operand or the number of a logical marker: use the PLC program to set the marker if the control should not automatically update the pocket table. If the parameter is empty, the pocket table is automatically updated by the control.

The following pre-defined scripts for configuring the tool-change sequences are available under the CfgOemScript configuration object:

Script key	Description
T0_TNC	Unload tool, transfer tool number 0 and data from the pocket table
T1_TNC	Only unload tool and transfer data from the pocket table
T0T1_TNC	First unload tool, and transfer tool number 0 and data from the pocket table. Then load tool and transfer data from the pocket table.
T1T0_TNC	First load new tool and transfer data from the pocket table. Then unload tool and transfer tool number 0 and data from the pocket table.

Spindle-gear Switching
Effective immediately, the NC software supports management of spindle-gear switching.

Settings in the configuration editor:	
System PLC CfgPlcSStrobe [Key for S strobe] **gearSpeed0** Axes ParameterSets [Key for the axis] CfgFeedLimits **nominalSpeed**	

Configure a separate spindle parameter set for each gear stage.

Note: Use the “KeySynonym” function to create new parameter sets rapidly and easily. (in the configuration editor under **KeySynonym** -> **CfgKeySynonym**). Only the first parameter set must be fully defined. All further parameters sets are “gated” to the first set **(MP_relatedTo),** so you only have to describe the differing parameters.

You must enter the names of the spindle parameter sets under **MP_gearSpeed0** in the list parameters. The list must be sorted in ascending order, with the smallest gear shaft speed at the top. Gear stages are not supported if the list is missing or empty.

Use the **MP_CfgFeedLimits** configuration datum of a parameter set to define the minimum and maximum spindle shaft speed for each gear stage.

The new **MP_nominalSpeed** machine parameter specifies the rated shaft speed for each gear stage. The control selects the necessary gear stage based on this shaft speed. The minimum and maximum shaft speeds of the individual gear stages (**MP_minFeed** and **MP_maxFeed** parameters) may overlap.

MP_gearSpeed0

List of parameter sets for spindle gear stages

Format: Array

Input: Keys of spindle parameter sets for each gear stage in wye operation. The parameter sets must be entered in ascending order, with the smallest gear shaft speed at the top.

MP_nominalSpeed

Rated shaft speed for the gear stage

Format: Numerical value

Input: Shaft speed in [rpm]
Enter the greatest programmable shaft speed at which this spindle parameter set is to be used. If a shaft speed greater than the given one is programmed, the next higher gear stage is switched to.

Module 9221 Start PLC positioning movement

The module positions an axis. The target position and feed rate are transferred in the module call. Limit switch interrogation can be activated in a separate transfer parameter.

The axis is positioned regardless of any other processes in the control. In particular, there is no interpolation with other axes.

Constraints:

- The module can only be called if no NC program is running, or if there is an M/G/S/T/T2/Q strobe. No axis direction key may be pressed in the Manual operating mode.
- For rotary axes with transition to zero, positioning is by the shortest path.
- If you wish to change a parameter (e.g., target position, feed rate) of a positioning command already in progress, you must first abort positioning, then change the parameter and start again.
- A simultaneous PLC positioning movement of several axes is interpolated. If you start an additional axis while already positioning another, the first movement is aborted, and then all the programmed axes (e.g., X, Y, and Z) are positioned together.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Target position> Input unit: 0.0001 mm
PS	B/W/D/K	<>Feed rate> Input unit: mm/min
PS	B/W/D/K	<>Mode> Bit 0 – Definition of the target position: 0: Absolute (i.e., relative to the machine datum) 1: Incremental Bit 1 – Software limit switch: 0: Inactive 1: Active
CM	9221	
PL	B/W/D	<>Error code> 0: Positioning is being started 1: Axis is not in a closed loop or is an auxiliary axis 2: Inadmissible values for the feed rate 3: Axis has not traversed the reference mark 4: No M/S/T/Q strobe during running program 5: Programmed axis not in closed loop 6: Positioning already started.

Module 9222 Interrogate PLC positioning status

The module provides the PLC positioning status.

Constraints:

- The status of all axes is interrogated simultaneously in bit-coded form.

Call:

PS	B/W/DK	<>Axis> Index from MP_CfgAxes/axisList
CM	9222	
PL	B/W/D	<>Status> 0: No PLC positioning was started 1: Target position reached 2: PLC positioning was started 3: Due to cancellation, target position not reached 4: Target position is outside of traverse range 5: Positioning not possible (e.g., due to “free rotation”)

Module 9224 Stop PLC positioning movements

The module stops the positioning movement of an NC axis that has been started by Module 9220 or 9221.

With this module you can interrogate the status of a PLC positioning movement.

Call:

PS	B/W/DK	<>Axis> Index from MP_CfgAxes/axisList
PS	B/W/DK	<>Mode> 0: Mode is not used at present.
CM	9224	
PL	B/W/D	<>Error code> 0: Stop PLC positioning 1: Invalid axis number 2: Invalid axis type 3: Axis is not in motion 4: Axis is controlled by NC 5: Invalid mode

Axis Error Compensation

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisComp **active**	

The parameter object CfgAxisComp is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

Axis errors are compensated for by changing the command variables for the position.

The control compensates the following mechanically-caused axis errors:

- Backlash
- Linear axis errors
- Nonlinear axis errors (direction-dependent)
- Screw pitch error
  - Axis sag
  - Thermal expansion
- Stiction (compensation is carried out in the speed controller)

You can activate either linear or nonlinear axis-error compensation per axis.

Backlash compensation can be activated in addition to linear axis-error compensation. If non-linear axis-error compensation is active, backlash compensation is not available. Backlash is taken into account in the compensation value table.

All other types of compensation are nonexclusive.
In MP_active, you switch all compensations (except stiction) on or off.

MP_active

Switch all axis compensations on/off

Format: Drop-down selection menu

Selection:

[On]
Backlash compensation, linear or nonlinear compensation, reversal-error compensation, and thermal compensation are all active

[OFF]
Axis-error compensation is not active

Default: Off

The following topics are described:

- **Backlash Compensation**
- **Linear Axis Error Compensation**
- **Nonlinear Axis Error Compensation**
- **Compensation of Thermal Expansion**

Backlash Compensation

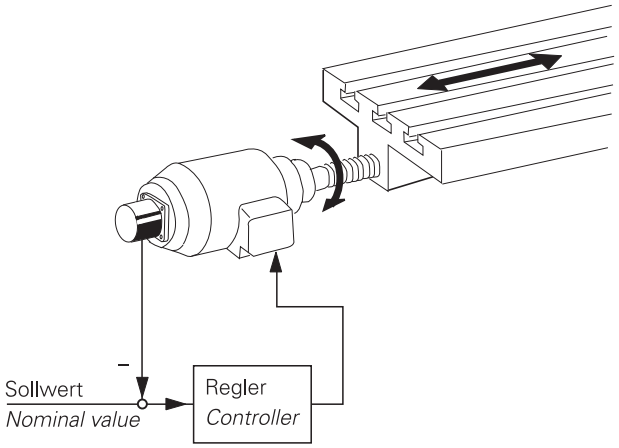
Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisComp **backLash**	

The parameter object CfgAxisComp is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

During a reversal in axis direction, there is often a little play between the rotary encoder and table. This play is referred to as backlash.

If the distance is measured using a speed encoder, the backlash compensation compensates the play between the rotary encoder and the table.



The following topic is described:

- **Compensation:**

Compensation:

- Enter the backlash in **MP_backLash.**

The value of the backlash is added to the position value at every reversal of direction (even if it results from a nonlinear axis-error compensation, for example) and considered by the position controller. The value of the k_V factor therefore influences the settling time for backlash compensation.

MP_backLash

Backlash compensation; backlash outside of the control loop

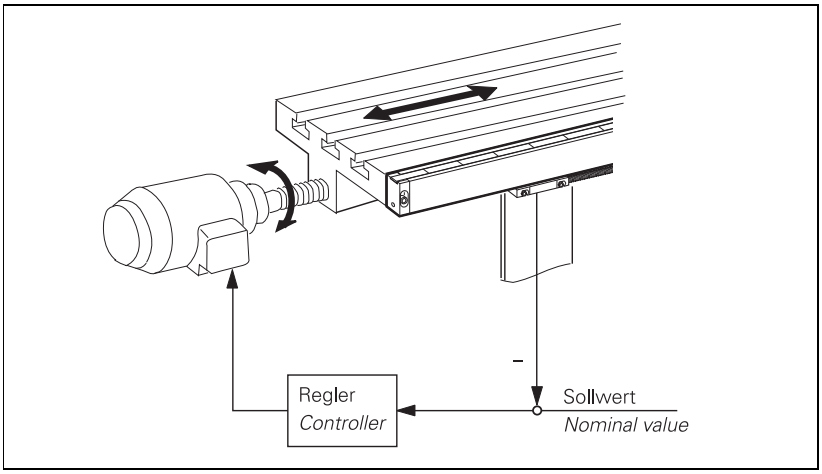
Format: Numerical value

Input: –1.000 000 000 to +1 [mm] or [°]

Default: 0

Note: If **nonlinear** axis-error compensation is active (MP_compType = non-linear), the backlash compensation is not available (see "**Linear Axis Error Compensation**")

For closed-loop (or direct) measurement with position encoders, the backlash compensation is usually not required.



Linear Axis Error Compensation

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisComp **linearCompValue** **compType**	

The parameter object CfgAxisComp is not required for:

Virtual axes (MP_axisMode=Virtual)

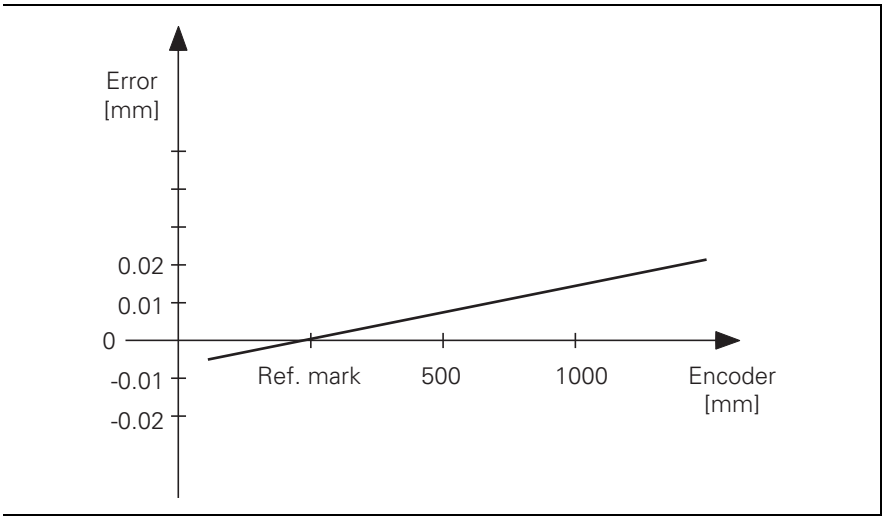
Axes that are for display only (MP_axisMode=Display)

Note: Linear axis error compensation is not available for rotary axes!

For every linear axis you can compensate a linear axis error.

Positive linear axis error: The table moves too far.

Negative linear axis error: The table moves too short a distance.



The following topic is described:

- **Compensation:**

Compensation:

- In MP_linearCompValue, enter the axis error in [mm/m].
- In MP_compType, activate the linear axis error compensation.

MP_linearCompValue

Linear axis error compensation

Format: Numerical value

Input: –1.000 000 000 to +1 [mm/m]

Default: 0

MP_compType

Selection of linear/nonlinear axis error compensation

Format: Drop-down selection menu

Selection:

[Linear]
Linear axis error compensation is active
[non-linear]
Nonlinear axis error compensation is active

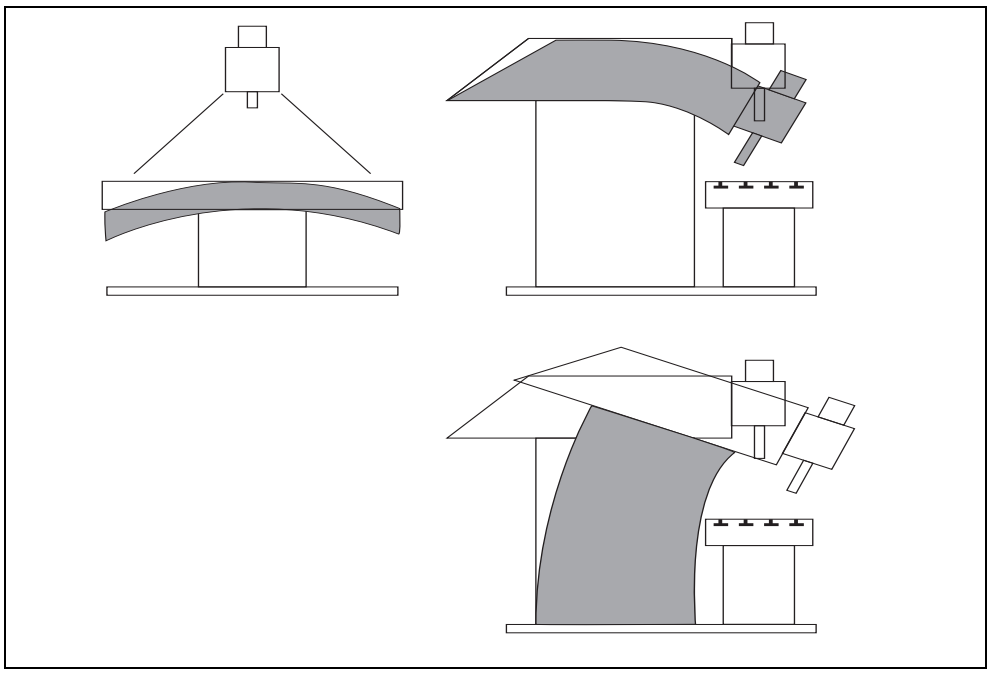
Default: Linear

Note: If **nonlinear** axis-error compensation is active (MP_compType = non-linear), the linear axis-error compensation is not available.

Nonlinear Axis Error Compensation

Depending on the design of the machine, production tolerances, or external factors (e.g., temperature), a non-linear axis-error can occur. Typical errors are screw-pitch errors and axis sag.

These graphics show typical nonlinear axis errors:



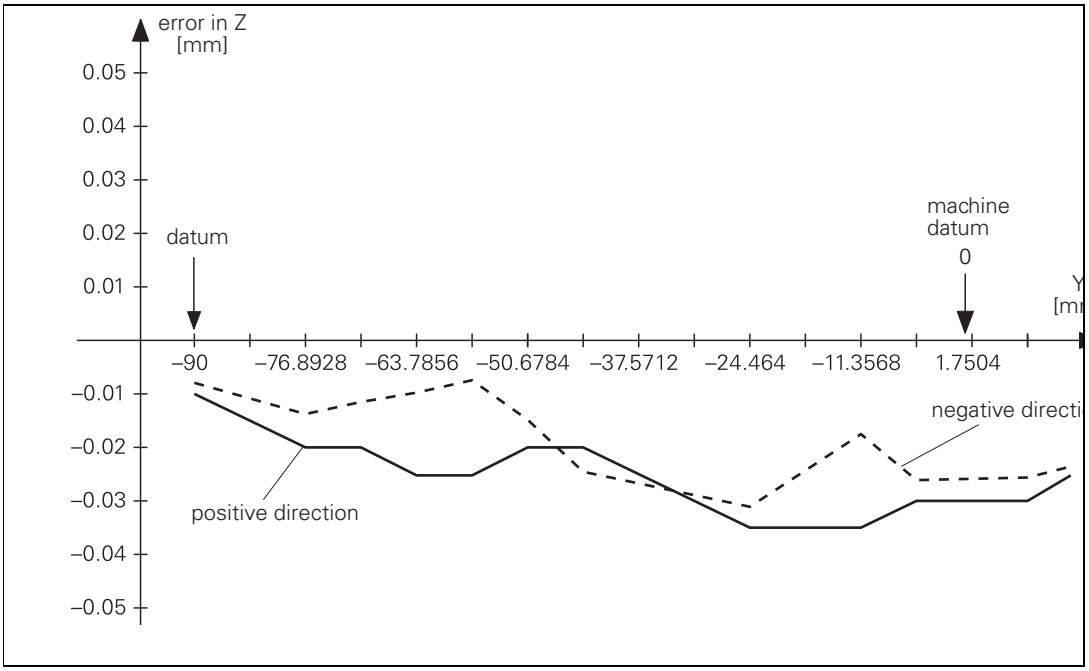
The best way to measure nonlinear axis error is with a comparator measuring system such as the ANILAM VM 101.

Note: The control can compensate screw-pitch error and axis sag simultaneously.

Nonlinear axis error compensation is also effective for an open loop. In this case the compensation value is considered when the control loop is closed.

Nonlinear axis error compensation supports one compensation value group for each the positive and negative direction of traverse.

The following graphic shows the trace of an axis sag error in the Z axis as a function of Y (Z = f(Y)):



The following topics are described:

- **Compensation Value Tables**
- **Entering Compensation Values**
- **Number of Compensation Points:**
- **How to Access the Tables:**
- **Assigning the Compensation Value Tables to the Axes**
- **Activate Error Compensation**
- **Module 9095 Activate axis-error compensation**
- **Module 9035 Read NC status information**
- **A Rotary Axis is a Special Case**
- **Master-slave Axes are Special Cases**

Compensation Value Tables

The compensation values for nonlinear axis error compensation are stored in the following tables:

- The ***.COM** tables contain the compensation values for max. 256 compensation points. A *.COM table is required for each axis and spindle. It consists of the following columns:
  - **AXISPOS:** Compensation points that are assigned compensation values. The compensation points are given with respect to the machine datum.
  - **BACKLASH:** Compensation values for screw pitch errors in negative direction of traverse. The BACKLASH column is defined for the axis for which this compensation-value table is created. This way the backlash can be compensated directly via the compensation-value table.
  - **Axis * – axis to which the table refers:** Compensation values for screw pitch errors in positive direction of traverse.
  - **Axis * – adjoining axis:** Compensation values for sag errors with respect to the adjoining axis.
  - **Spindle:** The compensation values for a spindle are entered in this column.
- In the ***.CMA** table, the *.COM tables are assigned to the error-causing axis.
  - **ACTIVE:** The character * activates the compensation value tables.
  - *** axis:** File name of the *.COM file with the compensation values of this axis.
  - **Spindle:** File name of the *.COM file with the compensation values of this spindle.

You will find the path of the *.CMA tables in the parameter object **System/Paths/CfgTablePath/TABCMA** (standard name of the file: config.cma). The *.CMA file contains the file names of the *.COM files. The directory path of the *.COM tables is entered in the parameter object **System/Paths/CfgOemPath/oemTable**.

Entering Compensation Values

The following information must be entered in the <*.COM> tables:

- In the AXISPOS column, enter the compensation points for the compensation values. The positions are given with respect the machine datum (MP_refPosition).
- (If required), enter the compensation values measured in the negative direction of traverse in the BACKLASH column.
- Enter the compensation values to which the compensation points belong in the column of the axis for which a dependency relationship exists. The name of the column is the name from MP_CfgAxes/axisList (see Table Format).

Example: The following dependencies apply for the Y axis and Z axis:

- Ballscrew pitch error in Z and Y: Z = F(Z) and Y = F(Y)
- Axis sag in Z depending on Y
- Traverse range: Z axis = 800 mm, Y axis = 500 mm
- Start point for compensation values: Z = –200 mm, Y = –90 mm
- Desired interval of compensation points: 5 mm

Number of Compensation Points:

$\frac{500\ mm}{5\ mm}$ = 100 compensation points in Y-axis

$\frac{800\ mm}{5\ mm}$ = 160 compensation points in Z-axis

How to Access the Tables:

- Switch to the **Programming** operating mode.
- Press the MOD key.
- Enter the code number 95148.
- Change to the program manager (PLC drive: becomes visible).
- Open the tables *.COM and *.CMA under PLC:/table.

Assigning the Compensation Value Tables to the Axes

General relationship for *.CMA tables:
[Axis in column from *.com] = F(Axis in column from *.cma, in which *.com is entered)

Enter the compensation-value tables in <*.CMA> table (standard name config.cma).
(for table formatting, see the chapter Tables):

- Enter a column for each axis to be compensated. The column names must match the axis keys from MP_CfgAxes/axisList.
- Enter the names of the compensation-value tables (*.COM) line-by-line in the appropriate axis columns.
  You can assign more than one compensation value table to each axis, however only one table can be active.
- Activate the compensations with an * in the column ACTIVE, which can be entered via the table editor or via the PLC (SQL server).
  All compensations in this line become active.

Example:
Z axis = F(Y axis); axis sag compensation
Y axis = F(axis); Nonlinear compensation
The first line is active.

Activate Error Compensation

Three requirements must be fulfilled for activating nonlinear axis error compensation:

- Activate the general compensation procedures with **MP_active = ON.**
- Activate the axis-specific nonlinear axis error compensation with **MP_compType=non-linear** (see "**Axis Error Compensation, Linear Axis Error Compensation**").
- In the config.cma file, activate a line with an ***** in the ACTIVE column or with Module 9095. The active line can be interrogated using Module 9035.

Note: Compensation is not available for axis and spindle positioning by PLC.

Module 9095 Activate axis-error compensation

Module 9095 activates a line in the selected file (*.CMA) and assigns the arguments for the compensation value tables (*.COM). Multiple measurement series (e.g., x=f(), y=f()...) can be stored in the compensation value tables. After the module has been executed, the argument is assigned. In this way the screw pitch error x=f(x) and axis sag x=f(y) can be compensated simultaneously, for example.

Constraints:

- The transferred line remains selected as the active line even after a control reset.
- Once the NC program has started, the module operates only during the output of an M/G/S/T/T2/Q strobe.
- The axis nominal values may change slightly when the compensation value table is switched over.

Call:

PS	B/W/D/K	<>Active line>
CM	9095	
PL	B/W/D	<>Error code> 0: Compensation was selected 1: Line was not found in the *.CMA table 2: Compensation value table (*.COM) is missing 3: Compensation value table > 256 entries 4: Maximum total number of compensation points exceeded 5: Too many compensation value tables (>10) 6: *.CMA file does not exist 7: Call was not from a submit job 8: Call during running program without strobe 10: *.CMA file is protected

Module 9035 Read NC status information

Module 9035 reads status information. A function number specifying the desired status information is transferred.

Transferred number		Return code
8	Selected machine axis (for actual-position-capture)	0: X axis 1: Y axis 2: Z axis 3: IV axis 4: V axis 5: VI axis 6: VII axis etc.
9	Handwheel axis	Finds the axis which is assigned to the handwheel connected to connector X23 of the MC. –1: None or more than one 0: X axis 1: Y axis 2: Z axis 3: IV axis 4: V axis 5: VI axis 6: VII axis etc.
10	Handwheel axis, bit-encoded	Bit 0: X axis Bit 1: Y axis Bit 2: Z axis Bit 3: IV axis Bits 4 to 13: Axes 5 to 14 (only available for PLC programs that work with API 1.0)
	Handwheel interpolation factor	
11	X key	0 to 10
12	Y key	
13	Z key	
14	IV key	
15	V key	
19	Active line in the *.CMA file	>=0: Line number –1: No *.CMA file
	Handwheel interpolation factor	
31	Axis 1	0 to 10
32	Axis 2	
33	Axis 3	
34	Axis 4	
35	Axis 5	
36	Axis 6	
37	Axis 7	
38	Axis 8	
39	Axis 9	
20	HR 410 speed	0: Slow 1: Medium 2: Fast

Transferred number		Return code
21	Control model	0: TNC 310 1: TNC 370 2: TNC 410 3: TNC 426 CA/PA 4: TNC 426 CB/PB/M or TNC 430 CA/PA/M 5: iTNC 530 6: iTNC 530 (with Windows®**1 2000) 20: **NC-Kernel based control** (TNC 320)
23	Handwheel superimposition with M118	0: M118 not active Bits 0 to 13: Axes 1 to 14
26	Jog increment	
	Handwheel interpolation factor	0 to 10
31	Axis 1	
32	Axis 2	
33	Axis 3	
34	Axis 4	
35	Axis 5	
36	Axis 6	
37	Axis 7	
38	Axis 8	
39	Axis 9	
100	Number of the tool axis	Only available for PLC programs that work with API 1.0
	Cycle counter	
500	Interpolator	0.. Counting of this value starts beginning at “0” and is incremented every cycle.
501	Cycle time of the PLC	
502	Current utilization	
503	Maximum utilization	
1001	Pallet table (only in a spawn job or submit job)	>= 0: Active line in the pallet table –1: Pallet table not active

Constraint:

Status information no. 19 (read active line in the *.CMA file) is displayed even if the active line does not contain any *.COR file.

Call:

PS	B/W/D/K	<Number of the desired status information>
CM	9035	
PL	B/W/D	<Status information>

**1 Windows® is a registered trademark of Microsoft Corporation.

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Status information invalid
	20	Call was not in a submit or spawn job

A Rotary Axis is a Special Case

For a rotary axis, only the compensation values for the entries of 0° to +60° are effective, relative to the machine datum. Therefore, the datum for the nonlinear compensation must lie within the 0° to +360° range. To compensate a full circle, set the compensation value datum to the machine datum.

Master-slave Axes are Special Cases

Separate compensation tables can be created for master axes and slave axes.

Compensation of Thermal Expansion

Settings in the configuration editor:	
System PLC CfgPlcPeriphery **tempCompensation** Axes ParameterSets Key for parameter set CfgAxisComp **active**	

The parameter object CfgAxisComp is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

To compensate thermal expansion, exact measurements of machine thermal behavior as a function of temperature (e.g., the center of axis expansion, the amount of the expansion) are necessary. Since the thermal expansion of the axes is largely proportional to the temperature, you can determine the amount of expansion by multiplying the temperature value by a certain factor.

The temperature values measured by the Pt100 thermistors are transferred using Module 9003. Module 9231 activates the compensation for thermal expansion according to the lag tracking method.

Compensation:

- Activate the general compensation procedures with **MP_active=ON**.
- Transfer the distance to be compensated to module 9231. At the same time, “lag tracking” becomes active. This means that the actual position is offset by a certain value per PLC cycle until the complete value is compensated.
- Define the amount of compensation per PLC cycle for lagged-tracking axis error compensation in MP_tempCompensation.

For gantry axes, the compensation value must be transferred separately for each axis.

Thermal compensation when using tilting axes is compensated via machine parameters.

MP_tempCompensation

Compensation of thermal expansion

Format: Numerical value

Input: 0.0000 to 359999.6400 [mm/min]
0 = Compensation switched off

Default: 0

The following topic is described:

- **Module 9231 Compensation of thermal expansion**

Module 9231 Compensation of thermal expansion

Thermal expansion is compensated by Module 9231. The axis number and the compensation value are transferred.

The module activates lag tracking. This means that the actual position is offset by a certain value per PLC cycle until the complete value is compensated. The increment of change per PLC cycle must be defined in MP_tempCompensation.

This does not change the value in the actual position display.

The module functions only in the cyclic PLC program.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Compensation value> Range: –30000 to +30000 [1/10 µm]
CM	9231	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	No error
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid axis number
	3	Invalid compensation value
	24	The module was called in a spawn job or submit job

Machine Kinematics

In the control the machine kinematics is described by machine parameters. It is a precondition that the kinematics models consist of translation axes and rotation axes which are linked to each other. This structure can also be used for configuring axes that are not perpendicular with respect to each other.

Multiple sets of kinematics can be configured for one machining channel. Multiple sets of kinematics are needed, for example, if different spindle heads are used on a milling machine, or if the spindle and C axis on a lathe are driven by the same motor.

After control startup, the last kinematics model entered in **CfgKinModel** will be active. If required, activate another kinematics model.

The following topics are described:

- **Configuration of the Machine Kinematics**
- **Definition of the Transformation with Vectors**
- **Definition of the Transformation with Angles**

Configuration of the Machine Kinematics

The following topics are described:

- **Overview of Machine Parameters for the Kinematics Configuration**
- **Important Rule of Thumb**
- **Preconfigured Kinematics in the Factory Default Setting of the NC Software**
- **Transformations on the Tool Side**
- **Transformations on the Workpiece Side**

Overview of Machine Parameters for the Kinematics Configuration

Settings in the configuration editor:	
NCchannel CfgKinModel [Key for kinematics model] **axesToolSide** **trafoToolSide** **trafoDirToolSide** **trafoAngleToolSide** **toolCoordSys** **axesWpSide** **trafoWpSide** **trafoDirWpSide** **trafoAngleWpSide** **machineTableSys**	

Describe the kinematics models in the parameter object **CfgKinModel/Key for kinematics model.**

Two transformation sequences are defined based on a **machine bed system C_{mb}**:

- Transformation sequence on the **tool side**
- Transformation sequence on the **workpiece side**

Each axis on the machine is represented by a coordinate system in one of the two sequences.

The **Z axis** for these coordinate systems is always defined as the **direction of movement** (for translation axes) or the **rotary axis** (for rotation axes) (convention for internal kinematics model).

The Z-axis of **translation axes** always indicates the positive direction of movement for the tool, and for the workpiece the negative direction of motion of the assigned physical axis. Positive direction of motion means that the REF display increases when the axis moves in this direction. On the other hand, a negative direction of motion means that the REF display decreases when the axis moves in this direction.

This means that the Z axis of a coordinate system indicates the positive direction of rotation for the tool, and for the workpiece the negative direction of rotation of the assigned physical axis for a **rotational axis** Positive direction of rotation for rotational axes means that the REF display increases when the axis rotates in this direction. On the other hand, a negative direction of rotation means that the REF display decreases when the axis rotates in this direction.

For machines with mutually perpendicular axes, the following results from this convention:
The machine bed coordinate system C_{mb} must be selected so that its axes are parallel to the physical axes of the machine.

If the algebraic sign of an axis is defined oppositely on the machine, then the coordinate system of the affected axis must be rotated in the transformation so that its Z axis points in the opposite direction.

Important Rule of Thumb

The position (location and orientation) of a coordinate system is always expressed in the coordinates of the **previous** coordinate system.

Example:

Position of C_Y in coordinates of C_{mb}

Position of C_x in coordinates of C_y

Position of C_{mt} (machine table) in coordinates of C_X

etc.

The following are defined as well:

- **Tool system** (C_{tool}) in **MP_toolCoordSystem** – in addition to the transformation sequence on the tool side
- **Machine table system** (C_{mt}) in **MP_machineTableSys** – in addition to the transformation sequence on the workpiece side

The transformation sequence can also contain other systems, such as the coordinate system of a 45× rotary axis (for horizontal/vertical spindles).

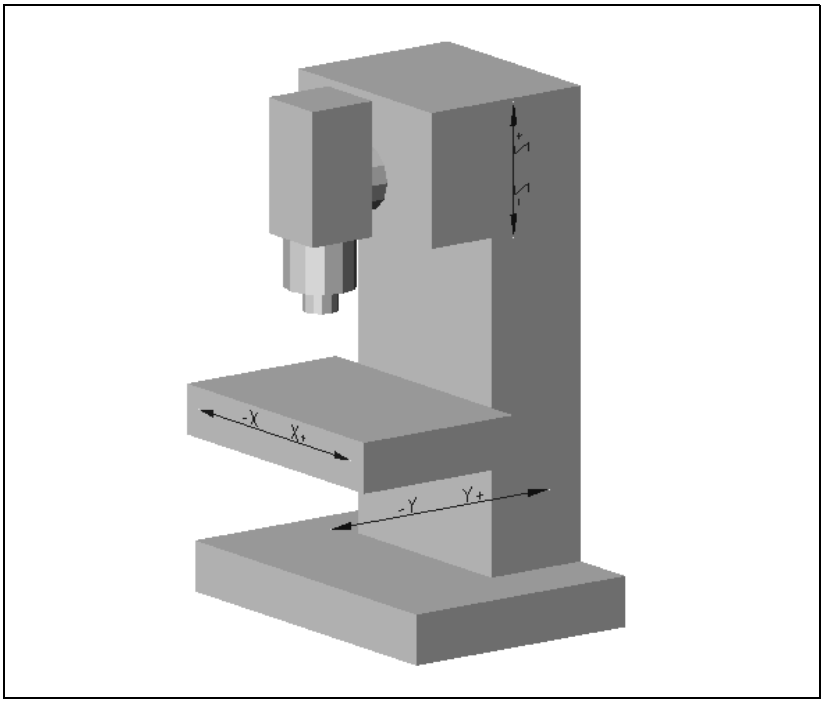
This system also represents an axis that can have the axis values 0× (vertical position) and 180× (horizontal position). Such an axis is moved to the appropriate position manually, via the PLC or an NC linear block.

Other coordinate systems can be auxiliary systems, which do not represent axes, but are only used to enter the values of the relevant factors in the kinematics chain. These systems are described as DefPoint systems. No axis values can be assigned to these DefPoint systems (as a default, the axis values are always null).

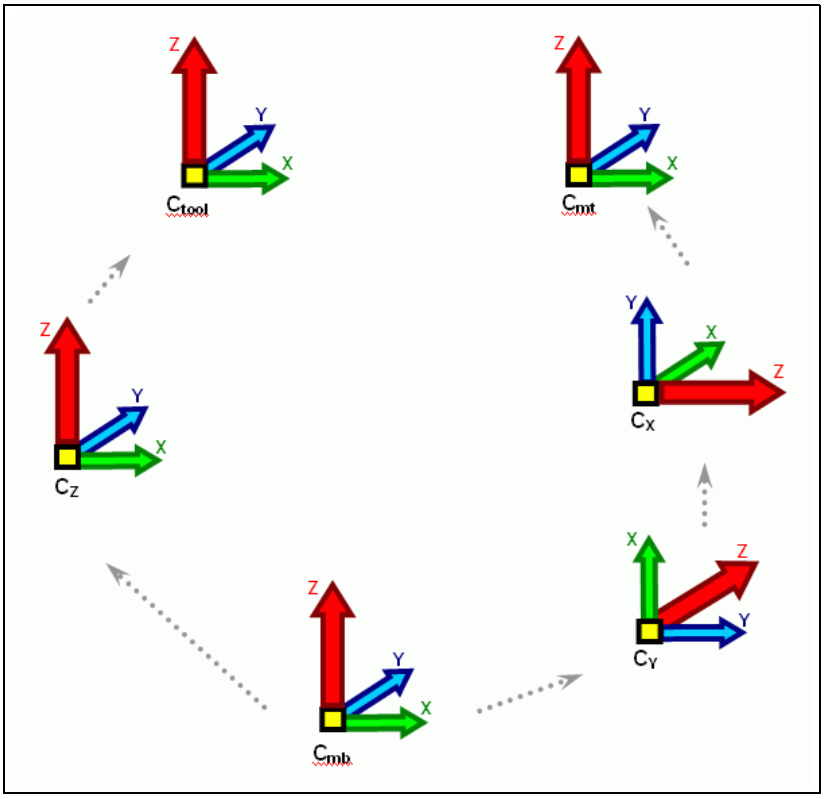
All axes defined in the kinematics chain must be entered in the parameter object **CfgAxis**. The axes with axis values are entered in **MP_CfgAxes/axisList,** and those without axis values (DefPoint systems) are entered in **MP_specCoordSysList.**

Preconfigured Kinematics in the Factory Default Setting of the NC Software

Below is a sketch of a simple machine with three linear axes X, Y, and Z. The kinematics properties of the machine have already been configured for you in the factory default setting of the NC software. Starting from the machine bed, the Z axis is on the tool side, and the Y axis followed by the X axis are on the machine bed side.



The kinematics chain for the example machine illustrated on the previous page is as follows:



Note: For display purposes, the origins of the coordinate systems are drawn distinct from each other even when they are at the same location.

In the machine configuration of the control, the kinematic chain shown above is described with vectors. The following pages will inform you of all machine parameters necessary for this. In the following table you can see how the kinematics configuration of the simple example machine is represented in the machine configuration:

Kinematics configuration of a machine with 3 linear axes X, Y, Z						
Machine bed						
						Coordinate system
C_{mb}: Kinematics of the machine bed						
Transformation sequence on the tool side						
C_Z: Kinematics Z axis	location		zDir		xDir	Coordinate system
	[0]	0	[0]	0	[0]	1
	[1]	0	[1]	0	[1]	0
	[2]	0	[2]	1	[2]	0
C_{tool}: Kinematics of tool system	location		zDir		xDir	Coordinate system
	[0]	0	[0]	0	[0]	1
	[1]	0	[1]	0	[1]	0
	[2]	0	[2]	1	[2]	0
Transformation sequence on the workpiece side						
C_Y: Kinematics Y axis	location		zDir		xDir	Coordinate system
	[0]	0	[0]	0	[0]	0
	[1]	0	[1]	1	[1]	0
	[2]	0	[2]	0	[2]	1
C_X: Kinematics X axis	location		zDir		xDir	Coordinate system
	[0]	0	[0]	0	[0]	0
	[1]	0	[1]	1	[1]	0
	[2]	0	[2]	0	[2]	1
C_{mt}: Kinematics of machine table	location		zDir		xDir	Coordinate system
	[0]	0	[0]	0	[0]	0
	[1]	0	[1]	1	[1]	0
	[2]	0	[2]	0	[2]	1

Transformations on the Tool Side

The machine kinematics (i.e., the coordinate transformations are described in the following parameters).

The key names of all axes on the tool side are entered in **MP_axesToolSide**. The sequence of the entries reflects the physical arrangement of the axes. Enter the axis on which the other axes are based at position [0].

The control assumes a three-dimensional kinematics model. If the NC channel does not have all of the three principal axes, replace the missing principal axes by dummy axes.

MP_axesToolSide

Key names of the axes that lie on the tool side

Format: Array [0–9]

Input: Key names from MP_CfgAxes/axisList (for dummy axes from MP_CfgAxes/specCoordSysList)

Enter the key names of the coordinate transformations on the tool side in **MP_trafoToolSide**. The sequence must correspond to the axes entered in **MP_axesToolSide**.

MP_trafoToolSide

Coordinate transformations on the tool side

Format: Array [0–9]

Input: Key names for coordinate transformations

Key names of the coordinate transformations defined by direction vectors are entered in **MP_trafoDirToolSide.** They must also be entered in **MP_trafoToolSide**, but cannot be entered in **MP_trafoAngleToolSide**.

MP_trafoDirToolSide

Coordinate transformations defined by direction vectors

Format: Array [0–9]

Input: Key names for coordinate transformations

Key names of transformations defined by angles are entered in **MP_trafoAngleToolSide**. They must also be entered in **MP_trafoToolSide,** but cannot be entered in **MP_trafoDirToolSide**.

MP_trafoAngleToolSide

Coordinate transformations defined by angle

Format: Array [0–9]

Input: Key names for coordinate transformations

MP_toolCoordSys is the end of the kinematics chain on the tool side.

MP_toolCoordSys

Key name of the tool coordinate system

Format: String

Input: Key names

Transformations on the Workpiece Side

The key names of all axes on the workpiece side are entered in **MP_axesWpSide**. The sequence of the entries reflects the physical arrangement of the axes. Enter the axis on which the other axes are based at position [0].

MP_axesWpSide

Keys for the axes on the workpiece side

Format: Array [0–9]

Input: Key names from MP_CfgAxes/axisList (for dummy axes from MP_CfgAxes/specCoordSysList)

Enter the key names of the coordinate transformations on the workpiece side in **MP_trafoWpSide**. The sequence must correspond to the axes entered in **MP_axesWpSide**.

MP_trafoWpSide

Coordinate transformations on the workpiece side

Format: Array [0–9]

Input: Key names for coordinate transformations

Key names of the coordinate transformations defined by direction vectors are entered in **MP_trafoDirWpSide**. They must also be entered in **MP_trafoWpSide,** but cannot be entered in **MP_trafoAngleWpSide**.

MP_trafoDirWpSide

Coordinate transformations defined by direction vectors

Format: Array [0–9]

Input: Key names for coordinate transformations

Key names of transformations defined by angles are entered in **MP_trafoAngleWpSide**. They must also be entered in **MP_trafoWpSide**, but cannot be entered in **MP_trafoDirWpSide**.

MP_trafoAngleWpSide

Coordinate transformations defined by angle

Format: Array [0–9]

Input: Key names for coordinate transformations

MP_machineTableSys is the end of the kinematics chain on the workpiece side.

MP_machineTableSys

Key of the machine-table coordinate system

Format: String

Input: Key names

Each coordinate transformation on the workpiece or tool side is defined with direction vectors or angles.

Definition of the Transformation with Vectors

Settings in the configuration editor:	
NCchannel CfgTrafoByDir [Key for transformation description] **location** **zDir** **xDir** CfgTrafoByAngle [Key for transformation description] **location** **angleDef** **angle1** **angle2** **angle3**	

A coordinate transformation is defined by the description of the position of a coordinate system in the previous coordinate system. This type of position is described by a position vector **(MP_location)** and an orientation. The two principle possibilities for describing the orientation are described below:

The following topic is described:

- **Defining Transformations by Using Direction Vectors**

Defining Transformations by Using Direction Vectors

MP_location defines the position of the coordinate origin of the transformed system relative to the previous coordinate system.

MP_location

Origin of this coordinate system in the previous system

Format: Array [2]

Input: -100 000.00000 to +100 000.00000 [mm]

In **MP_zDir** you define the Z direction of the current coordinate system using the previous coordinate system. For more information about the position of the vector **MP_zDir,** see the table **Kinematics configuration of a machine with 3 linear axes X, Y, Z**.

Z-basis vector expressed in the previous coordinate system

Format: Array [2]

Input: –1 to +1

In **MP_xDir** you define the X direction of the current coordinate system using the previous coordinate system. For more information about the position of the vector **MP_xDir,** see the table **Kinematics configuration of a machine with 3 linear axes X, Y, Z**.

MP_xDir

X-basis vector expressed in the previous coordinate system

Format: Array [2]

Input: –1 to +1

Definition of the Transformation with Angles

MP_location defines the position of the coordinate origin of the transformed system relative to the previous coordinate system.

MP_location

Origin of this coordinate system in the previous system

Format: Array [2]

Input: -100 000.00000 to +100 000.00000 [mm]

MP_angleDef specifies the interpretation of the angles.

MP_angleDef

Specifies the interpretation of the angles

Format: Drop-down selection menu

Selection:

[Cardan]
Orientation by Cardan angles
[RollPitchYaw]
Orientation by rotation around fixed axes
[Euler]
Orientation by Eulerian angles

MP_angle1

Angle 1

Format: Numerical value

Input: –360.0000 to +360.0000 [°]

MP_angle2

Angle 2

Format: Numerical value

Input: –360.0000 to +360.0000 [°]

MP_angle3

Angle 3

Format: Numerical value

Input: –360.0000 to +360.0000 [°]

Reference Marks

The following topics are described:

- **Definition**
- **Traversing the Reference Marks**
- **Defining the Process of Traversing the Reference Marks**
- **"Pass Over Reference Point" Mode of Operation**

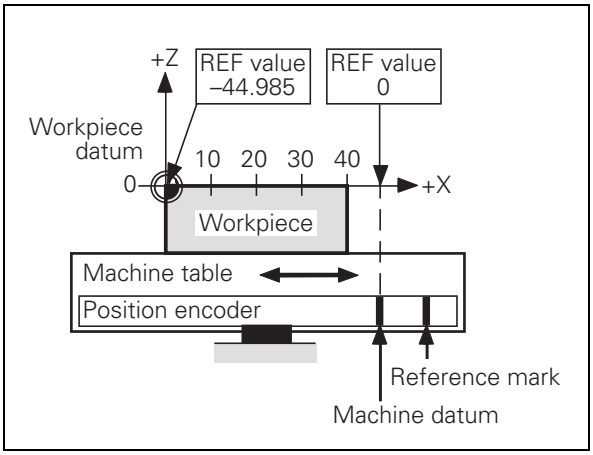
Definition

The position value (the coordinates) of an axis position is defined with respect to a freely selectable datum. When the axes are moved, the ACTUAL position is calculated incrementally. An interruption in power causes the reference between the axis position and the position value to be lost.

ANILAM linear encoders are designed with one or more reference marks. The reference marks identify an axis position at a known distance from the machine datum. The position of the freely selectable datum is defined with respect to the machine datum.

The datum and the actual position can be reproduced as soon as the reference marks are traversed.

ANILAM recommends position encoders with distance-coded reference marks. With distance-coded reference marks, the position value can be reestablished after traverse of a short distance over any two reference marks.



Traversing the Reference Marks

The reference marks must be traversed after any interruption in power. Specify which axes are homed, and in which sequence, in **MP_refAllAxes** or **MP_refAxis.**

- Press the machine **START** button: The reference marks are automatically traversed (MP_refAllAxes=True).

or:

- Press the machine axis-direction button: The user determines the sequence of the axes (MP_refAllAxes=False).
- After the reference marks have been traversed:
- The software limit switches are activated.
- The most recently set datum or workpiece datum and the machine datum are reproduced.

The following topics are described:

- **Distance Between the Scale Reference Point and the Machine Datum**
- **Encoders with EnDat Interface**
- **Renewed Traversing of the Reference Marks**

Distance Between the Scale Reference Point and the Machine Datum

For position encoders with distance-coded reference marks, the machine datum is defined with respect to the scale reference point, which is the first reference mark after the beginning of the measuring length. On angle encoders, the scale reference point is marked:

- In MP_refPosition, enter the distance between the scale reference point and the machine datum.

For position encoders without distance-coded reference marks but with more than one reference mark, every reference mark to be traversed must be evaluated.

- For each reference mark to be traversed, create another parameter block, and enter in MP_refPosition the distance between the scale reference point and the reference mark.
- Activate the parameter block that corresponds to the traversed reference mark.

Encoders with EnDat Interface

Position encoders and speed encoders with EnDat interface can be connected to the control. With these encoders there is no need to traverse the reference marks. The position value is only read when the control is switched on. It cannot be read again.

When connecting a position encoder with EnDat interface, or a speed encoder with EnDat interface as a position encoder:

- Enter **MP_refType=EndatEncoder**.

Note: f use of multiturn encoders with EnDat interfaces results in overflows, the corresponding information is stored temporarily. If the control is exchanged, **MP_refPosition** must be re-adjusted.

Renewed Traversing of the Reference Marks

Module 9220 Traverse reference mark

The module starts the reference mark traverse in an axis or servo-controlled spindle. If the reference mark has already been evaluated, it can be evaluated again by this module. The module can be called in all operating modes.

Constraints:

- Software limit switches are not effective.
- The sequence of functions is determined by **MP_refType.**
- The velocity and the direction for traversing the reference marks are either taken from **MP_CfgReferencing/refFeedHigh** and **MP_CfgReferencing/refDirection**, or they are defined in the module.
- An axis cannot be started for referencing until all other axes are in position.
- If an axis is started for reference point traverse although the reference mark has already been traversed, NN_AxReferenceAvailable is reset and the reference mark is

evaluated again. The same constraints apply as when traversing the reference mark for the first time.

- If the spindle is started for reference point traverse, marker NN_SpiReferenceAvailable is set.
- The spindle must be started from a standstill to traverse the reference mark.

Note: The direction of traverse should be defined in the module only in exceptional cases. Since the reference end position is not considered in this case, the limits of the traverse range may be violated.

Call:

PS	B/W/D/K	<>Axis/spindle> Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Feed rate/shaft speed> 0: Feed rate/rpm from MP_CfgReferencing/refFeedHigh >0: Feed rate in mm/min or shaft speed in 1/1000 rpm
PS	B/W/D/K	<>Direction of traverse> –1: Negative direction 0: Direction from MP_CfgReferencing/refDirection 1: Positive direction
CM	9220	
PL	B/W/D	<>Error code> 0: Reference mark traverse is commanded 1: Axis does not exist, or not a servo-controlled spindle 2: Inadmissible values for the feed rate / direction 3: Incorrect operating mode 4: Reference traverse already started 5: Axis is already being positioned or the spindle is in motion 6: Other axis is already being positioned 8: Programmed axis not in closed loop

Defining the Process of Traversing the Reference Marks

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgReferencing **refType** **refSwitchActive** **refFeedLow** **refFeedHigh** **refDirection**	

The parameter object CfgReferencing is not required for:

- Virtual axes (MP_axisMode=Virtual)

You define the process of traversing the reference marks in the following machine parameters:

- In MP_refDirection and MP_refFeedHigh (for rotary encoders also in MP_refFeedLow), you define the direction and velocity for traversing the reference marks.
- In MP_refAxis, you define the sequence of axes for traversing the reference marks.
- In MP_refType you select the type of reference marks.

MP_refType

Functional sequence for traversing the reference mark

Format: Drop-down selection menu

Selection:

[Switch, changing Dir]

For linear axes with speed encoder; reference run with NC start

[Switch, no changing Dir]

For linear axes with speed encoder; reference run with NC start

[without Switch]

For spindle, rotary table with angle encoder; reference run with NC start

[distance coded]

For distance-coded linear encoders; reference run with NC start

[distance coded + on the fly]

For distance-coded linear encoders; reference run with axis-direction keys or NC start

[without switch + on the fly]

For spindle; reference run with M3, M4

[Endat Encoder]

For axes with EnDat encoder; reference-mark traverse not necessary

The following topics are described:

- **Direction and Velocity**
- **Position Encoder with Distance-coded Reference Marks**
- **Position Encoder with One Reference Mark**
- **Linear Measurement Through Rotary Encoder**

Direction and Velocity

In MP_refDirection you specify the direction of traverse. If the axis traverses the reference-end-position trip dog, and PP_AxReferenceEndPosition is set, the direction of traverse is reversed.

In MP_refFeedHigh and MP_refFeedLow, define the velocity for traversing the reference marks.

It depends on the entry in MP_refType whether the low or high reference run velocity is used.

MP_refDirection

Direction for traversing the reference marks

Format: Drop-down selection menu

Selection:

[Positive]
Positive direction of traverse
[Negative]
Negative direction of traverse

Default: Negative

MP_refFeedLow

Low velocity for traversing the reference mark

Format: Numerical value

Input: 10.000 000 000 to 36 000 000 [mm/min]

Default: 600[mm/min]

MP_refFeedHigh

High velocity for traversing the reference mark

Format: Numerical value

Input: 80.000 000 000 to 36 000 000 [mm/min]

Default: 1 200 [mm/min]

MP_refSwitchActive defines the status of the trip dog for reference end position.

MP_refSwitchActive

Active level of the trip dog for reference end position

Format: Drop-down selection menu

Selection:

[High]

Reference-end-position trip dog is active at high level

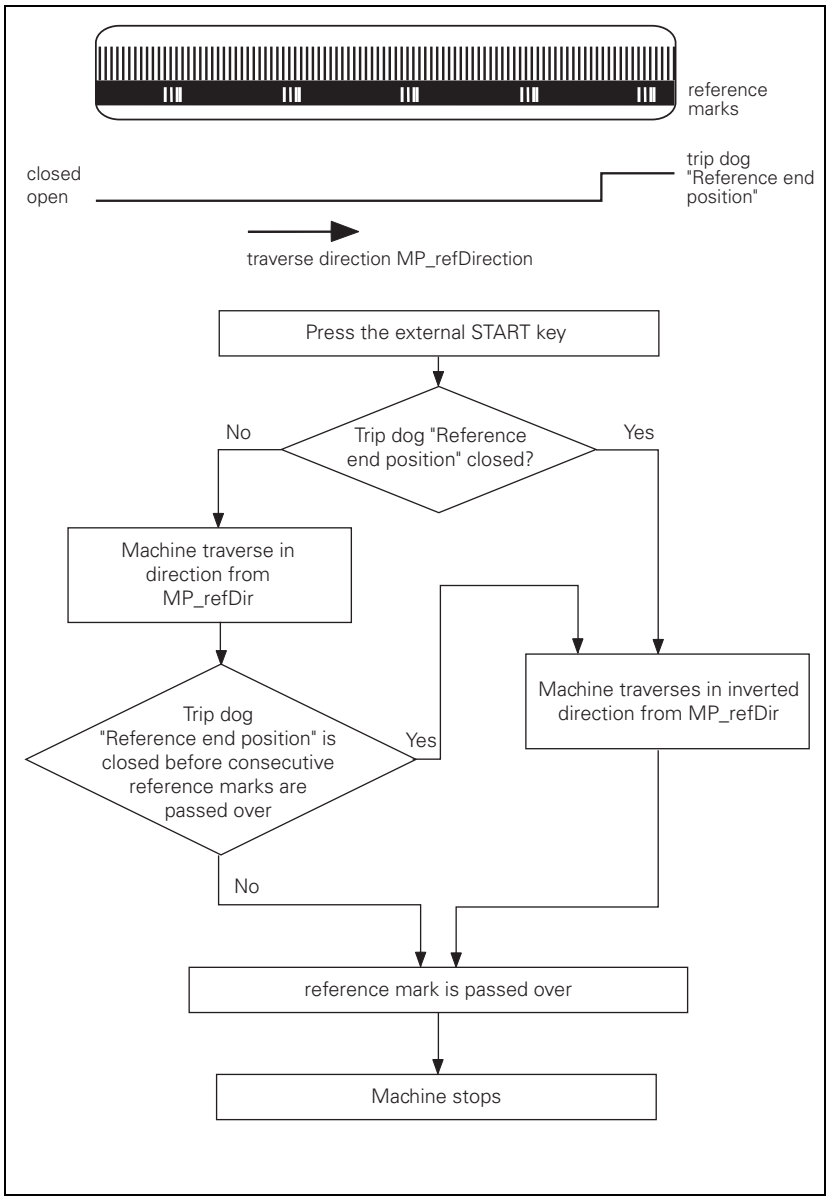
[Low]

Reference-end-position trip dog is active at low level

Default: High

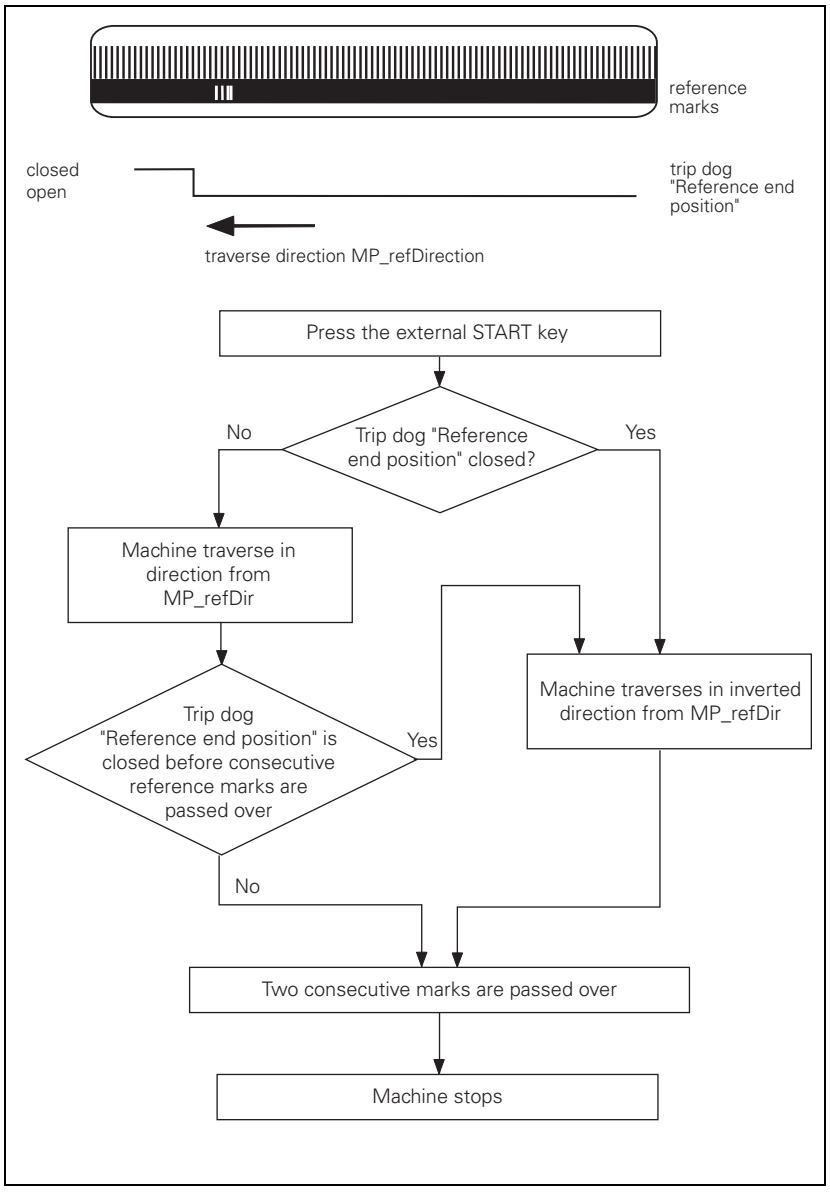
Position Encoder with Distance-coded Reference Marks

Functional sequence when **MP_refType=distance coded**



Position Encoder with One Reference Mark

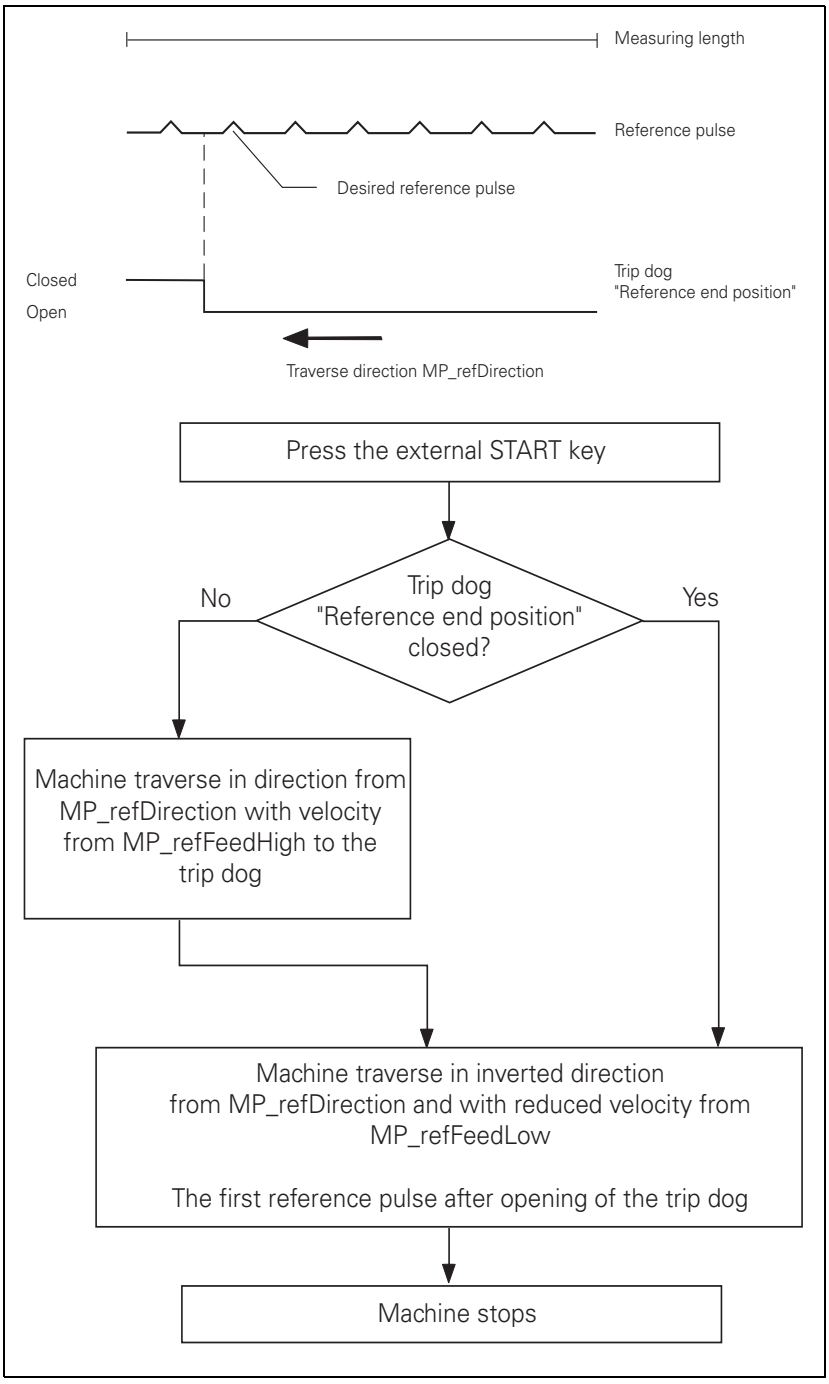
Functional sequence when **MP_refType=without Switch**



Linear Measurement Through Rotary Encoder

Functional sequence when **MP_refType=Switch, changing Dir**

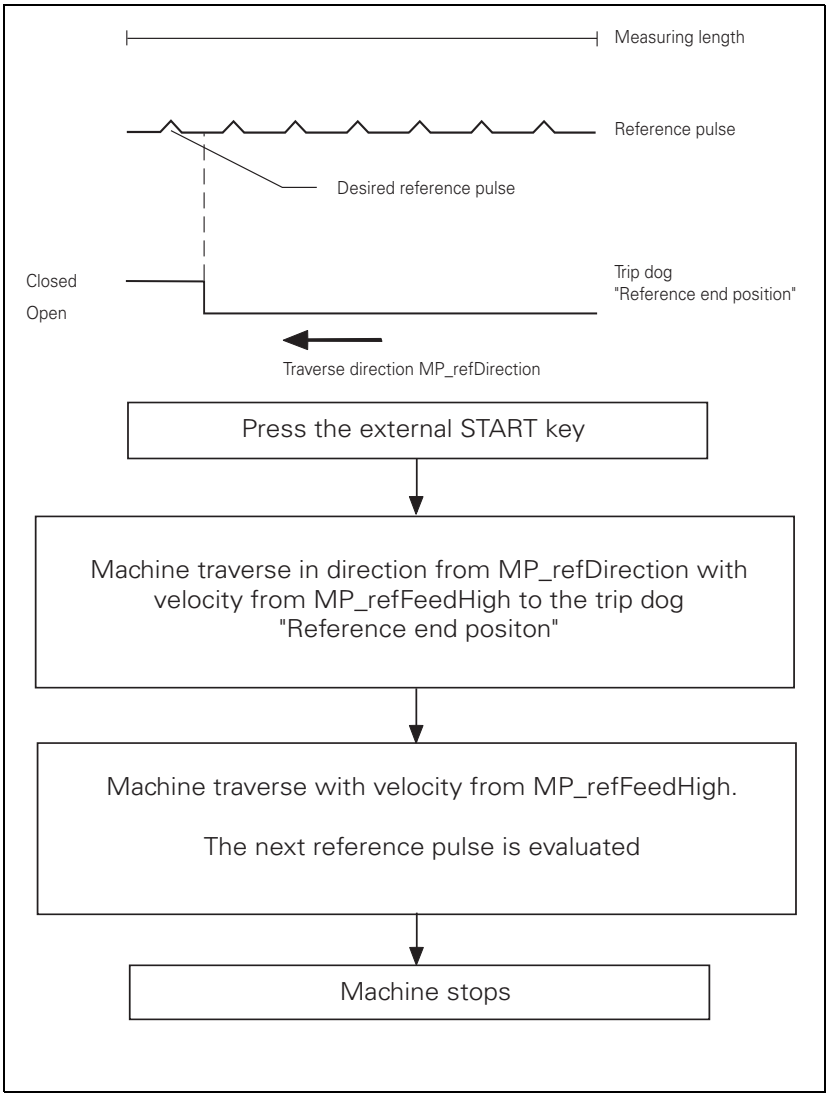
For linear measurement using a rotary encoder, a reference pulse is produced at each revolution of the encoder. Ensure that during referencing the same reference pulse is always evaluated. This can be realized with the trip dog for reference end position.



Functional sequence when **MP_refType=Switch, no changing Dir**

For linear measurement using a rotary encoder, a reference pulse is produced at each revolution of the encoder. During the reference run the first reference pulse traversed after the trip dog for reference end position is closed is evaluated. This ensures that the same reference pulse is always evaluated.

For linear measurement using a rotary encoder, ANILAM recommends using the **MP_refType=Switch, changing dir** method.



“Pass Over Reference Point” Mode of Operation

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgReferencing **refPosition** **moveAfterRef** **moveAfterRefAbs** **moveAfterRefPos** **moveAfterRefFeed**	

The parameter object CfgReferencing is not required for: reference marks:

- Virtual axes (MP_axisMode=Virtual)

In NN_OmgReference, the NC informs the PLC of the “Pass Over Reference Point” operating mode. In NN_AxReferenceAvailable, the NC reports whether the reference marks of this axis were traversed.

If you switch the operating mode before all reference marks are traversed, the control identifies this state and prompts you to traverse the remaining reference marks.

The following topics are described:

- **Reference End Position**
- **Machine Datum**
- **Positioning after Reference Mark Traverse**

Reference End Position

To prevent the axes from violating their traverse limits when traversing the reference marks, each axis requires a trip dog (at the reference end position). The trip dogs must be installed by the machine tool builder at the ends of the traverse range. The switch signals from the trip dogs are sent to free PLC inputs. The PLC program must gate these PLC inputs with **PP_AxReferenceEndPosition** for “reference end position”. Setting the reference end position causes a reversal of the traverse direction from MP_refDirection.

PLC operand	Type
NN_AxReferenceAvailable **Reference mark has not yet been traversed** 0: Reference mark not traversed 1: Reference mark traversed	M
PP_AxReferenceEndPosition **Reference end position** 0: Trip dog not triggered 1: Trip dog triggered	M

Machine Datum

MP_refPosition defines the position of the machine datum relative to the reference point of the scale. For encoders with distance-coded reference marks, the position is relative to the scale reference point; for encoders with EnDat interface, relative to the absolute encoder datum.

MP_refPosition

Position of machine datum

Format: Numerical value

Input: –100 000.000 000 000 to +100 000 [mm] or [°]

Default: 100

Positioning after Reference Mark Traverse

The axis can automatically be moved to a certain position after reference mark traverse is completed. This behavior is activated with MP_moveAfterRef.

Define the following information for positioning after reference mark traverse:

- In MP_moveAfterRefPos, the end position.
- In MP_moveAfterRefFeed, the feed rate.
- In MP_moveAfterRefAbs, absolute or incremental positioning.

Note: This function is mainly intended for positioning rotary tables.
Ensure that no collision occurs a result of this positioning.
The software limit switches are already active.

MP_moveAfterRef

Activate motion after reference-mark traverse

Format: Drop-down selection menu

Selection:

[On]

Activate positioning after reference-mark traverse

[OFF]

No positioning after reference-mark traverse

Default: Off

MP_moveAfterRefAbs

Absolute movement after finding the reference mark

Format: Drop-down selection menu

Selection:

[Absolute]
Absolute positioning
[Relative]
I Incremental positioning

Default: Absolute

MP_moveAfterRefPos

Position for positioning after reference-mark traverse

Format: Numerical value

Input: –100 000.000 000 000 to + 100 000 [mm] or [°]

Default: 0

MP_moveAfterRefFeed

Feed rate for positioning after reference-mark traverse

Format: Numerical value

Input: 10.000 000 000 to 36 000 000 [mm/min] or [°/min]

Default: 6 000

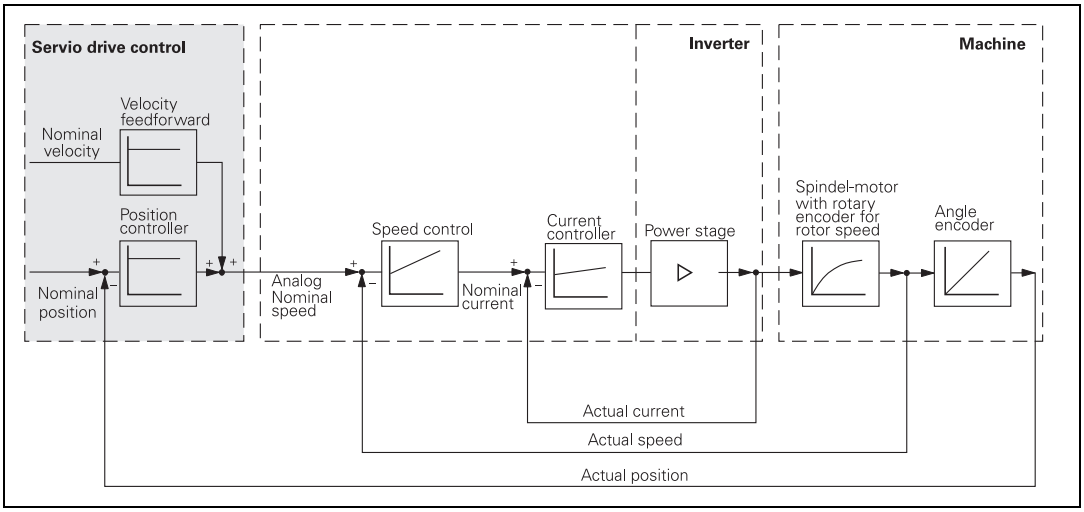
The Control Loop

Machine tools normally function on the principle of cascade control. Here the position control loop is prior to the speed and current control loops.

Benefits of cascade control::

- Transparent structure of the individual control loops.
- Disturbances can be compensated through the subsequent controllers. This relieves the prior controller.
- The respective outer control loop protects the inner control loop by limiting the command variable.
- Individual commissioning of each control loop, starting with the innermost loop.

The position controller is integrated in the 6000i. The speed controller, current controller and power module are located in the servo amplifier. The speed command signal is sent by the control to the servo amplifier through an analog interface.



The following topics are described:

- **Relationship Between Jerk, Acceleration, Velocity, and Distance**
- **Geometry Filter**
- **Look-Ahead**
- **Interpolator**
- **Filter Before Position Control Loop**
- **Position Controller**
- **Activating and Deactivating Position Control Loops**
- **Feed-Rate Enable**
- **Controller Parameters for Manual Traverse**
- **Controller Parameters for Analog Axes**
- **Switching Parameter Blocks**

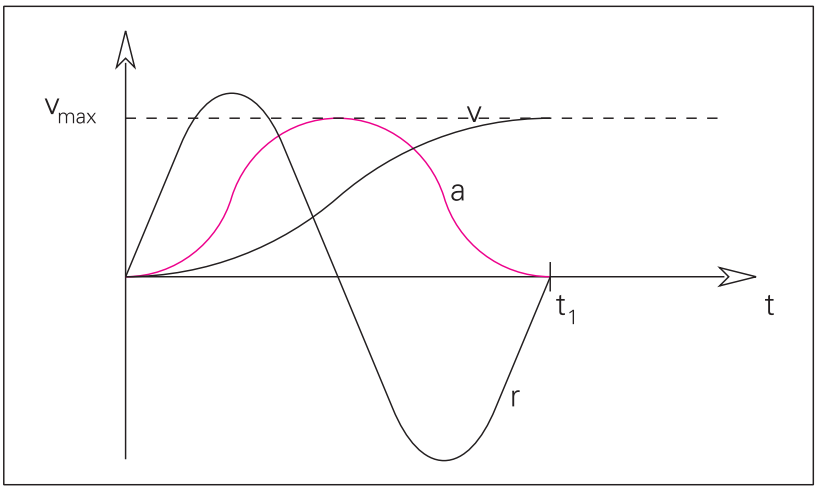
Relation Between Jerk, Acceleration, Velocity, and Distance

The following topics are described:

- **Acceleration and Jerk**
- **Distance**

Acceleration and Jerk

Taking into account the motor and the power module, the machine should be specified in such a way that acceleration during the acceleration phase is as constant as possible. This ensures maximum utilization of the drive current. On the other hand, the machine should also be designed to fulfill the dynamic requirements. The jerk should be kept to a minimum and the jerk phase should be maximized in order to prevent the machine from oscillating. This results in a bell-shaped acceleration curve (see figure).



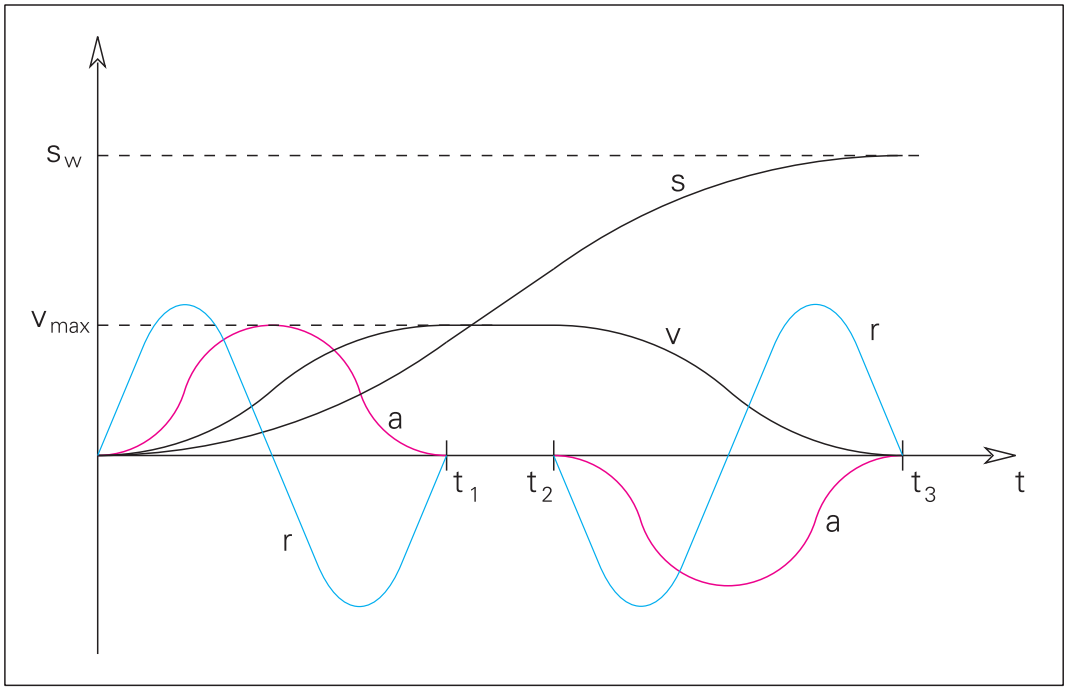
Legend:

- *v*: Velocity (5th-order curve)
- *a*: Acceleration (4th-order curve)
- *r*: Jerk (3rd-order curve)
- v_{max}: Maximum velocity

Distance

To attain the maximum velocity, a minimum distance must be traversed. This also applies to the braking phase.

If the traverse distance is greater than the distance covered during the acceleration and braking phases, a movement at constant (maximum) speed is inserted (see period of time from t_1 to t_2 in the figure below).



Legend:

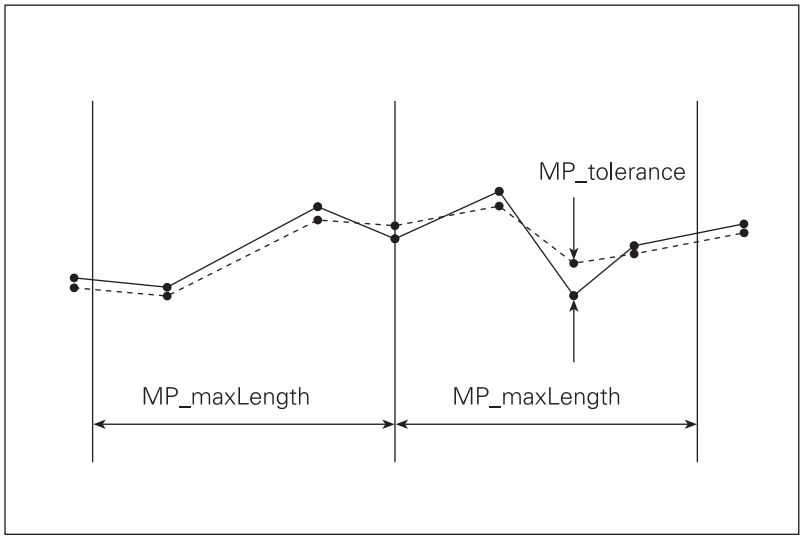
- *v*: Velocity
- *a*: Acceleration
- *r*: Jerk
- *s*: Distance
- v_{max}: Maximum velocity
- S_w: Traverse distance
- t_1: End of acceleration phase
- t_2: Start of braking phase
- t_3: End of traverse distance

Geometry Filter

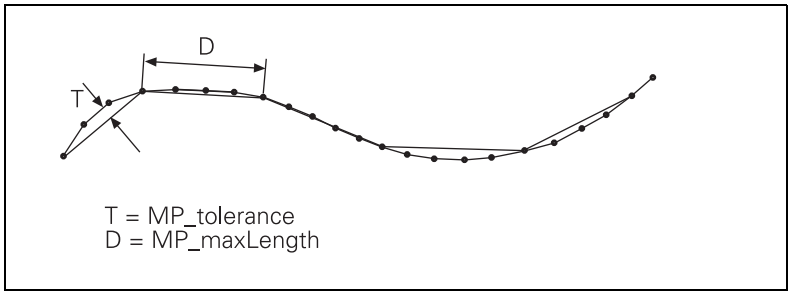
Settings in the configuration editor:	
NCchannel channelSettings Key for channel CfgStretchFilter **filterType** **tolerance** **maxLength**	

The function of the filter, which takes effect before look-ahead in the geometry chain, depends on the setting in **MP_filterType**.

- **Average:** The geometry filter smoothes corners. This method moves the contour points in such a way that the change in direction is less distinct (see figure below). This enables you to machine corners at a higher feed rate, resulting in an improved workpiece surface quality.
  Use the following machine parameters to influence this function:
  - MP_tolerance: defines the maximum shift of a contour point
  - MP_maxLength: Defines the filter width (i.e., the range within which the geometry filter is effective)



- **ShortCut:** The geometry filter filters out very close NC blocks (unnecessary linear blocks) ahead of time, resulting in fewer NC blocks, which speeds the program processing (see figure below).
  Use the following machine parameters to influence this function:
  - MP_tolerance: defines the maximum tolerance
  - MP_maxLength: Defines the maximum length of the new linear path as a result of filtering



MP_filterType

Type of stretch filter

Format: Drop-down selection menu

Selection:

[OFF]
Filter is off.

[ShortCut]
Omit individual points on polygon; if the connecting line from the middle point of three subsequent points on a polygon to the previous point or to the next point is within the tolerance band, the middle point will be omitted.

[Average]
The geometry filter smoothes corners.

MP_tolerance

Maximum distance of the filtered to the unfiltered contour

Format: Numerical value

Input: 0.000000 to 10.000000 [mm]
0 = Stretch filter is off

Default: 0

MP_maxLength

Maximum length of the path as a result of filtering

Format: Numerical value

Input: 0.0000 to 1 000.0000 [mm]
0 = Stretch filter is off

Default: 0

Look-Ahead

Look-Ahead receives the NC blocks from the geometry filter. Under consideration of certain limit values, look-ahead cyclically calculates the maximum possible contouring speed approx. 300 blocks in advance.

The calculated values are transferred to the **interpolator** in feed-rate profiles. The interpolator calculates axis-specific nominal values from the position polynomials and feed-rate profile.

The programmed contouring feed rate, maximum axis accelerations, permissible axis/path jerk, filter parameters and tolerances are taken into account in these feed-rate profiles. The feed-rate profiles are also influenced by changes to the override potentiometer, and by whether SINGLE BLOCK or FULL SEQUENCE is active.

Small variations in the feed rate, which appear during calculation of the feed-rate profile, are suppressed in order to achieve a smooth feed rate.

The following topics are described:

- **Contour Smoothing**
- **Path-Specific Limit Values**
- **Axis-Specific Limit Values**
- **Tolerance for Corners and Arcs**
- **Tolerance for Rotary Axes**

Contour Smoothing

In order to achieve smooth machining surfaces with a minimum of machining time, the following must be kept in mind:

- Each jerk (da/dt), which is caused by a change in direction on the contour, or by a change in the acceleration or in the feed rate, excites vibrations in the machine. Therefore, the jerk must be limited to a permissible size.
- For feed rates above the machining feed rate, an increased jerk and increased tolerance are both permissible, since they no longer have any effect on the machining quality.
- The tool may go to the limits of the adjustable path tolerance (deviation from the contour), but must not go outside the tolerance.
- Each machine axis is programmed for a certain feed rate, and has a specified capability for acceleration. For interpolating axes, the acceleration of the slowest axis is decisive.
- Feed rates must not fall beneath the minimum value.

- In MP_minPathFeed, define the minimum feed rate within a segment (on the path except at corners), which will not be violated by look-ahead. The feed rate will be violated by look-ahead only if programmed in the NC block, or if defined by feed-rate override.
- In MP_minCornerFeed, define the minimum feed rate at corners, which will not be violated by look-ahead. The feed rate will be violated by look-ahead only if programmed in the NC block, or if defined by feed-rate override.
- In MP_maxG1Feed, define the maximum machining feed rate.

  - The tolerance from MP_pathTolerance and the jerk from MP_maxPathJerk are effective for feed rates that do not exceed the feed rate from MP_maxG1Feed.
  - The higher tolerance from MP_pathToleranceHi and the higher jerk from MP_maxPathJerkHi are effective for higher feed rates as well as for rapid traverse movements.
- Independent of the feed rate, define the maximum yank (dj/dt) in MP_maxPathYank.

MP_pathTolerance includes errors caused by the filter before the position control loop (see "**Function of the Filters Before the Position Control Loop**"). The feed-rate override does not affect which jerk or tolerance is in effect.

Path-Specific Limit Values

Settings in the configuration editor:	
NCchannel channelSettings Key for channel CfgLaPath **minPathFeed** **minCornerFeed** **maxG1Feed** **maxPathJerk** **maxPathJerkHi** **pathTolerance** **pathToleranceHi** **maxPathYank**	

- Define the path-specific limit values for feed rate, acceleration and jerk.

The feed rate minimum defined in MP_minPathFeed is only violated within a segment if a lower feed rate is programmed.

MP_minPathFeed

Minimum feed rate on the path

Format: Numerical value

Input: 0.000 000 000 to 600 000 [mm/min]

Default: 60 [mm/min]

The feed rate minimum defined in MP_minCornerFeed is only violated between two segments if a lower feed rate is programmed.

MP_minCornerFeed

Minimum feed rate at corners

Format: Numerical value

Input: 0.000 000 000 to 600 000 [mm/min]

Default: 30 [mm/min]

MP_pathTolerance limits the feed rate at corners and curvatures. This keeps the filter errors within certain limits.

TNC control: In the NC program, Cycle 32 can be used to change the value in MP_pathTolerance.

MP_pathTolerance

Path tolerance for contour transitions after the filter

Format: Numerical value

Input: 0.0001 to 10.000 000 000 [mm]

Default: 0.01 [mm]

The tolerance defined in MP_pathToleranceHi is effective for feed rates greater than those defined in MP_maxG1Feed.

MP_pathToleranceHi

Path tolerance after the filter at rapid traverse

Format: Numerical value

Input: 0.0001 to 10.000 000 000 [mm]

Default: 0.01 [mm]

MP_maxG1Feed

Maximum machining feed rate

Format: Numerical value

Input: 0.000 000 000 to 99 999 [mm/min]

Default: 99 999 [mm/min]

The jerk defined in MP_maxPathJerk is effective for machining feed rates that do not exceed the maximum machining feed rate from MP_maxG1Feed.

MP_maxPathJerk

Maximum jerk on the path

Format: Numerical value

Input: 0.000 to 1 000 000.000 [m/sec^3]

Default: 40 [m/sec^3]

The jerk defined in MP_maxPathJerkHi is effective for feed rates greater than the feed rate defined in MP_maxG1Feed.

This value also applies for feed rates greater than "maxG1Feed".

MP_maxPathJerkHi

Maximum jerk on the path at rapid traverse

Format: Numerical value

Input: 0.000 to 1 000 000.000 [m/sec^3]

Default: 40 [m/sec^3]

MP_maxPathYank

Maximum yank on the path (dj/dt)

Format: Numerical value

Input: 0.000 000 000 to 1 000 000 [mm/sec^4]

Default: 4 000 [mm/sec^4]

Axis-Specific Limit Values

Settings in the configuration editor:	
Axis ParameterSets Key for parameter set CfgFeedLimits **minFeed** **maxFeed** **rapidFeed** **manualFeed** **maxAcceleration** CfgLaAxis **axJerk**	

The parameter objects **CfgFeedLimits** and **CfgLaAxis** are not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

- Define the axis-specific limit values for feed rate, acceleration and jerk.

MP_minFeed

Minimum axis feed rate or minimum spindle speed

Format: Numerical value

Input: 0.000 000 000 to 36 000 000 [mm/min] or [°/min]

Default: 0

MP_maxFeed

Maximum axis feed rate (rapid traverse) or maximum spindle speed

Format: Numerical value

Input: 0.000 000 000 to 36 000 000 [mm/min] or [°/min]

Default: 16 000

MP_rapidFeed is used as the maximum axis feed rate in the **Manual Operation** and **El. Handwheel** modes in conjunction with the rapid traverse key.

MP_rapidFeed

Rapid traverse in Manual Operation

Format: Numerical value

Input: 0.0 to 36 000 000.0 [mm/min or °/min]

Default: 4 999.98

In the **El. Handwheel** mode, the value entered in **MP_manualFeed** is multiplied by the value entered in **MP_CfgHandwheel/feedFactor.**

MP_manualFeed

Maximum manual feed rate

Format: Numerical value

Input: 0.0 to 36 000 000.0 [mm/min or °/min]

Default: 4 999.98

MP_maxAcceleration defines the axis-specific acceleration. The value entered also applies to braking.

MP_maxAcceleration

Maximum permissible axis acceleration

Format: Numerical value

Input: 0.000 000 000 to 1000 [m/sec^2] or [1000°/sec^2]

Default: 3 [m/sec^2] or [1000°/sec^2]

In **MP_axJerk,**you limit the axis-specific jerk.

MP_axJerk

Maximum axis jerk

Format: Numerical value

Input: 0.000 000 000 to 1 000 000 [m/sec^3]

Default: 0.1 [m/sec^3]

Tolerance for Corners and Arcs

Settings in the configuration editor:	
NCchannel channelSettings Key for channel CfgLaPath **curveTolFactor** **curveJerkFactor**	

A distinction is made between corners and arcs when estimating the filter error. Therefore, the tolerance at corners (MP_pathTolerance effective) and on arcs (MP_curveTolFactor * MP_pathTolerance effective) can be set separately. This results in different maximum velocities.

If, based on the contour geometry, a progression of straight lines is recognized as an arc, the tolerance from MP_pathTolerance is multiplied by the value in MP_curveTolFactor. This enables the velocity to be increased (with lower contour accuracy) on curved contours, but not at corners.

Curvature changes cause a jerk resulting in a reduction of speed. If the value in MP_curveJerkFactor does not equal 0, the reduced speed is multiplied by this value.

For example, if oscillations occur on the workpiece at line-to-arc transitions, the speed can be reduced by MP_curveJerkFactor <1.

MP_curveTolFactor and MP_curveJerkFactor are evaluated for smoothing the contour. Function of the filters: see "**The Control Loop, Filter Before Position Control Loop**".

MP_curveTolFactor
Factor for path tolerance in circles
Format: Numerical value
Input: 0.5 to 100.000 000 000
Default: 1

MP_curveJerkFactor
Factor for feed rate reduction at curvature changes
Format: Numerical value
Input: 0 to 100.000 000 000
0: No further feed rate reduction
Default: 1

Tolerance for Rotary Axes

Settings in the configuration editor:	
Axis ParameterSets Key for parameter set CfgLaAxis **axFilterErrWeight**	

The parameter object CfgLaAxis is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

The filter error for a rotary axis with a large radius can be multiplied by a factor. Weighting the factor smoothes the feed-rate profile for the rotary axis.

Enter MP_axFilterErrWeight=1 for linear axes.

MP_axFilterErrWeight is evaluated for smoothing the contour. Function of the filters: see "**The Control Loop, Filter Before Position Control Loop**".

MP_axFilterErrWeight
Factor for filter errors (for rotary axes)
Format: Numerical value
Input: 0.010 000 000 to 100.0
For linear axes: 1
Default: 1

Interpolator

Settings in the configuration editor:	
System CfgCycleTimes **ipoCycle**	

The interpolator operates at the clock rate defined in MP_ipoCycle. The axis-specific nominal position values are calculated from the feed rate profiles transferred by look-ahead using this cycle time.

MP_ipoCycle

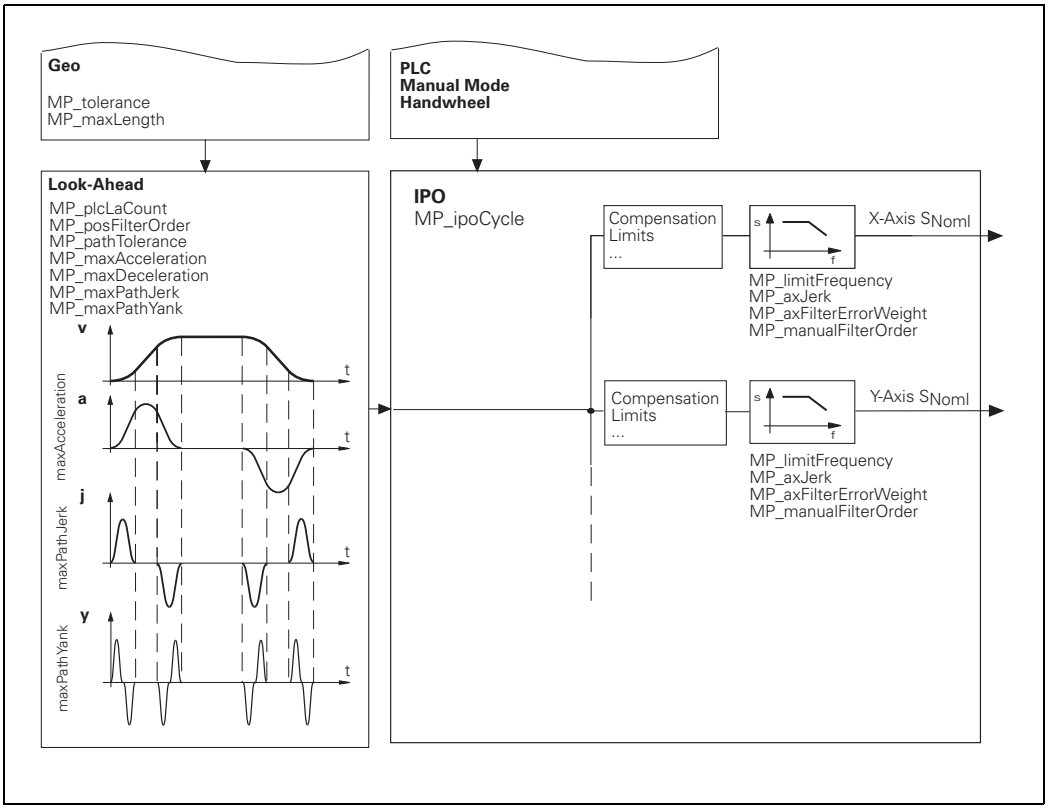
Cycle time of position controller (interpolation clock pulse)

Format: Numerical value

Input: 3 [ms]

Default: 3 [ms]

Schematic of the Interpolator:



Filter Before Position Control Loop

Settings in the configuration editor:	
System CfgFilter **typeFilter1** **orderFilter1** **typeFilter2** **orderFilter2** Axis ParameterSets Key for parameter set CfgPositionFilter **filter1Shape** **filter1LimitFreq** **filter2Shape** **filter2LimitFreq**	

The following topics are described:

- **Configuration of Filters**
- **Function of the Filters Before the Position Control Loop**

Configuration of Filters

Two filters are located before the position control loop to prevent the machine from oscillating. In MP_typeFilter1/2, you assign a filter to the linear axes or the rotary axes, and in MP_orderFilter1/2 you define the global filter order. Then you define for each axis whether a filter is used and which filter is used for optimizing the axis. In addition, the frequency of the filter is defined axis-specifically.

The smoothing function of these filters causes contour errors. The velocity profile is adjusted by the look-ahead function so that the contour error does not exceed the given tolerance (see "**The Control Loop, Look-Ahead**").

Note: Filters delay the processing time of NC blocks by the control, since multiple NC blocks must be considered for the filter functions.

Take the machine setup into account when you configure filters. It is decisive whether TCPM (Tool Center Point Management) executes compensating movements for rotary axes.

ANILAM recommends that you configure the filters of standard machines (TCPM is not used) as follows:

- Enter MP_typeFilter1=Off.
- Enter MP_typeFilter2=Position and define the filter order in MP_orderFilter2.

- For **rotary and linear axes,** enter MP_filter2Shape=HSC and enter the cutoff frequency of the machine in MP_filter2LimitFreq.

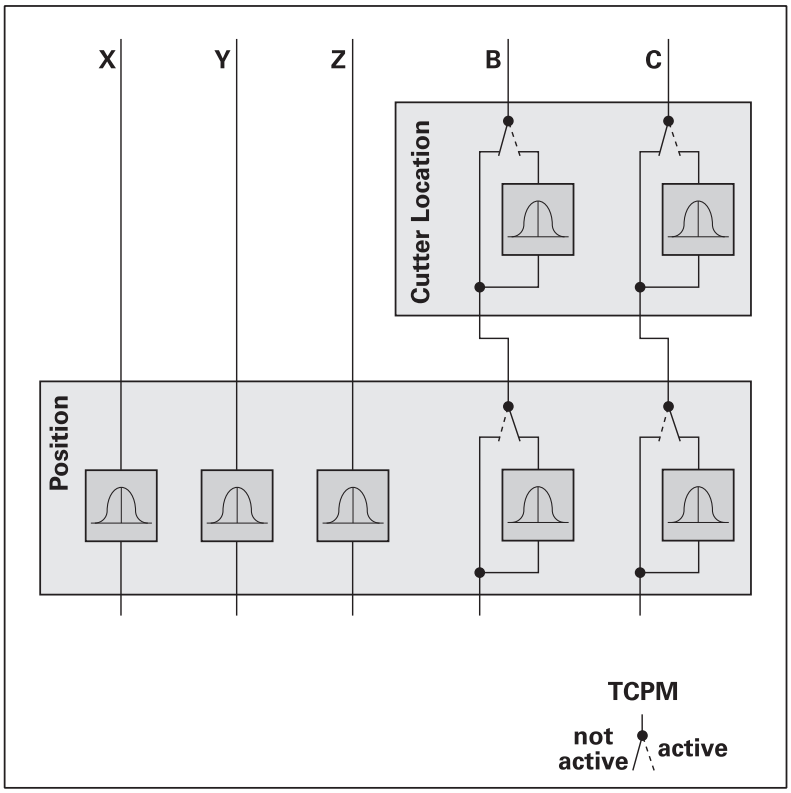
ANILAM recommends that you configure the filters of complex machines (TCPM is used) as follows:

- Enter MP_typeFilter1=Cutter Location and define the filter order in MP_orderFilter1.
- Enter MP_typeFilter2=Position and define the filter order in MP_orderFilter2.
- For **rotary axes,** enter MP_filter1Shape=HSC and enter the cutoff frequency of the machine in MP_filter1LimitFreq.
- For **rotary and linear axes,** enter MP_filter2Shape=HSC and enter the cutoff frequency of the machine in MP_filter2LimitFreq.

Function of the Filters Before the Position Control Loop

The function of the filters for rotary axes depends on TCPM (see figure):

- TCPM is not active:
  - The Cutter Location filters are not active.
  - The Position filters are active for all axes.
- TCPM is active:
  - The Cutter Location filters are active.
  - The Position filters are not active for rotary axes.



This method provides the following advantage:

- At first, the cutter location point for rotary axes is optimized by the HSC filter.

- The values optimized by the cutter-location-point filter are used as the basis for the filter for linear axes. This means that the rotary axis values that have already been corrected are evaluated by the linear axis filter.

MP_typeFilter1

Type of first nominal position value filter

Format: Drop-down selection menu

Selection:

[OFF]

Filter 1 is not active.

[Position]

Axis position (for linear and rotary axes)

[CutterLocation]

For rotary axes

Default: Off

MP_orderFilter1

Order of first nominal position value filter

Format: Numerical value

Input: 1 to 31

Default: 11

MP_typeFilter2

Type of second nominal position value filter

Format: Drop-down selection menu

Selection:

[OFF]

Filter 2 is not active.

[Position]

Axis position (for linear and rotary axes)

[CutterLocation]

For rotary axes

Default: Position

MP_orderFilter2

Order of second nominal position value filter

Format: Numerical value

Input: 1 to 31

Default: 11

MP_filter1Shape

Shape of first nominal position value filter

Format: Drop-down selection menu

Selection:

[OFF]
Not active
[Average]
Not yet implemented
[Triangle]
Not yet implemented
[HSC]
High Speed Cutting

Default: Off

MP_filter1LimitFreq
Cutoff frequency of first nominal position value filter
Format: Numerical value
Input: 10.000 000 000 to 100 [Hz]
Default: 66 [Hz]

MP_filter2Shape
Shape of second nominal position value filter
Format: Drop-down selection menu
Selection:

[OFF]
Not active
[Average]
Not yet implemented
[Triangle]
Not yet implemented
[HSC]
High Speed Cutting

Default: Off

MP_filter2LimitFreq
Cutoff frequency of second nominal position value filter
Format: Numerical value
Input: 10.000 000 000 to 100 [Hz]
Default: 66 [Hz]

Note:
- For the Position filter, the cutoff frequencies of the axes should not differ significantly.
- The rotary-axis cutoff frequencies for the Cutter Location filter are usually smaller than the cutoff frequencies for the Position filter.

Position Controller

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgPosControl **kvFactor** **feedForwardFactor** **controlOutputLimit**	

The parameter object **CfgPosControl** is not required for:

- Virtual axes (**MP_axisMode**=Virtual)
- Axes that are for display only (**MP_axisMode**=Display)

The position controller uses the axis-specific nominal position values transferred by the interpolator. The nominal speed values are determined and transferred to the speed controller.

The following topics are described:

- **Feedback Control**
- **Feedback Control with Following Error**
- **Interrelation of k_v Factor, Feed Rate, and Following Error**
- **Limiting of Controller Output**
- **Feedback Control with Velocity Feedforward**
- **Rapid Traverse and Feed Rate Limitation**
- **Feed Rate Values in PLC Operands**

Feedback Control

The control operates with following error (servo lag) or with velocity feedforward. It is defined in **MP_feedForwardFactor**.

- If MP_feedForward=0, operation with 100% servo lag is in effect.
- MP_feedForward>0 activates velocity semifeedforward control (for analog axes).
- If MP_feedForward=1, machining will be carried out, using 100% velocity feedforward (for digital axes).

Calculation:

U_{out} = **MP_kvFactor** · P_{err} + V_{nom} · **MP_feedForwardFactor**

U_{out} = output voltage (analog nominal value in volts at connector X8)
P_{err} = following error (mm)
V_{nom} = nominal velocity (mm/min)

MP_feedForwardFactor

Factor for velocity feedforward

Format: Numerical value

Input: 0.000 000 000 to 1.5

0: Feedback control with following error

>0,<1: Feedback control with velocity semifeedforward

1: Feedback control with velocity feedforward

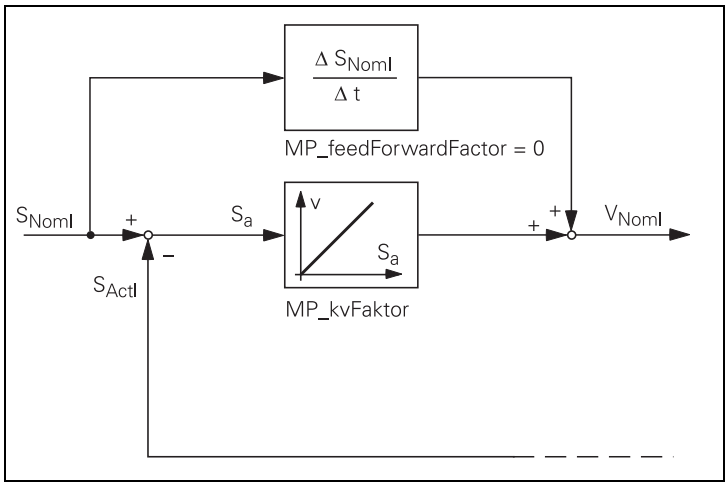
Default: 1

Note: For axes that are interpolated with each other, the k_v factor and the factor for velocity feedforward must be equal. In this case the smaller k_V factor determines the input value for all axes.

Feedback Control with Following Error

Following error (also known as servo lag) is a gap that remains between the nominal position commanded by the NC and the actual position of the axis.

Simplified representation:



The nominal position value s_{nomI} for a given axis is compared with the actual position value s_{actI} and the resulting difference is the following error s_a:

$s_a = s_{NomI} – s_{ActI}$

s_a = following error
s_{NomI} = nominal position value
s_{ActI} = actual position value

The following error is multiplied by the k_v factor and passed on as nominal velocity value:

$v = k_v \cdot s_a$

v_{nomI} = nominal velocity value

The control loop gain, known as the k_v factor, defines the amplification of the position control loop. You must find the optimum k_v factor by trial and error.

If you choose a k_v factor that is too large, the following error will become very small. However, this can lead to oscillations.

If you choose too small a k_v factor, the axis will move to a new position too slowly.

For axes that are interpolated with each other, the k_v factors must be equal to prevent contour deviations.

- Define the k_v factor in MP_kvFactor.

Interrelation of k_V Factor, Feed Rate, and Following Error

The following formula shows the interrelation of k_V factor, feed rate, and following error. :

$$k_v = \frac{v_e}{s_a} \quad \text{or} \quad s_a = \frac{v_e}{k_v}$$

k_V = kv factor [(mm/sec)/mm]
v_e = rapid traverse [mm/sec]
s_a = following error [mm]

Note: The unit for the kv factor of the 6000i differs from the one used for the other TNC contouring controls, such as the iTNC 530.
Unit for the kv factor of the 6000i:mm / (mm · sec)
Unit for the kv factor of the iTNC 530:m / (mm · min)
Therefore:
iTNC 530 kv factor · 1000 / 60 = 6000i kv factor

MP_kvFactor
kv factor (proportional component of the position controller)
Format: Numerical value
Input: 0.000 000 000 to 1000 [1/sec]

Limiting of Controller Output

The controller output limit **MP_controlOutputLimit** is used only during switch-on of position control without actual-to-nominal value transfer. Example:

Clamped or hanging axes cause a following error when the position control loop is open. When closing the control loop without actual-to-nominal value transfer, this difference in the position is corrected by the control. The deviation is corrected at the maximum feed rate entered in **MP_controlOutputLimit**.

Note: The axis parameters entered for jerk and acceleration have no effect. Enter only values that are non-critical to the axis.
ANILAM recommends:
In **MP_controlOutputLimit,** enter a value that is approximately
0.1 · **MP_manualFeed.**

If **MP_controlOutputLimit** = 0, the resulting following error is not corrected until the next positioning block.

MP_controlOutputLimit

Controller output limit for the position controller

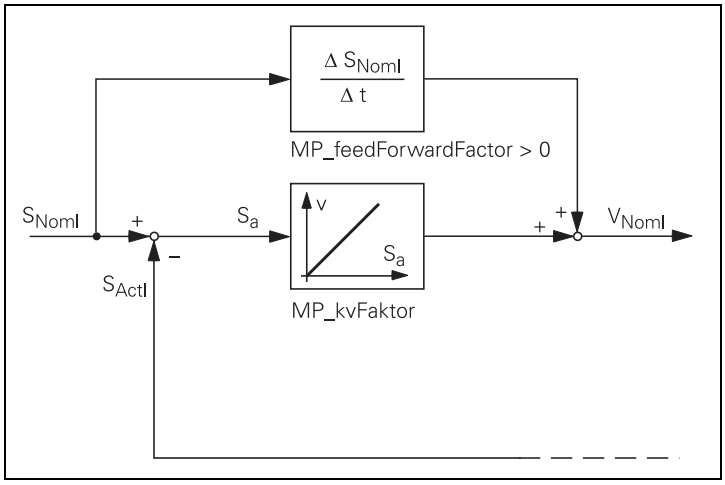
Format:	Numerical value
Input:	0.000 000 000 to 1666 [mm/min]
Default:	0 [mm/min]

Feedback Control with Velocity Feedforward

For feedback control with velocity feedforward, the nominal velocity value consists of an open-loop and a closed-loop component.

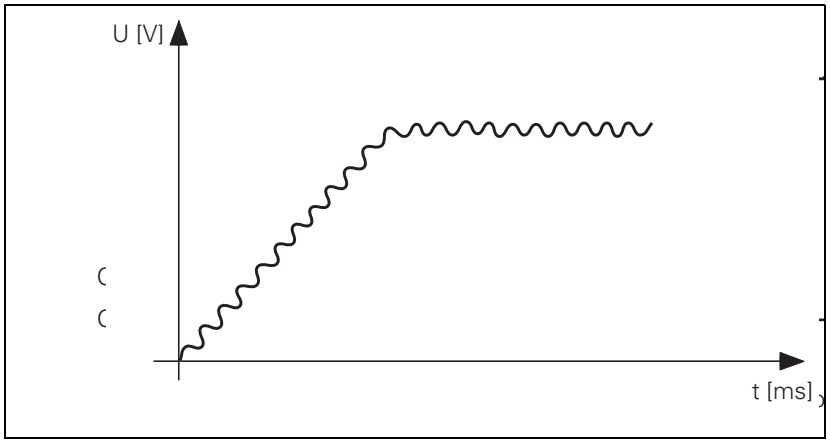
The machine-adjusted nominal velocity value is the open-loop controlled component. The closed-loop velocity component is calculated through the following error. The following error is small.

In most cases, machines are controlled with velocity feedforward, since it makes it possible to machine exact contours even at high speeds.



You can influence the control of the forward-fed velocity with the k_v factor:

- Enter a k_v factor in **MP_kvFactor.**

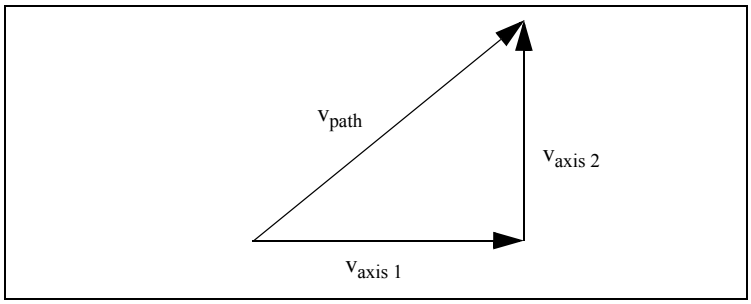


Warning: If the k_v factor that you select is too large, the system will oscillate around the forward-fed nominal velocity value.

Unlike operation with following error, you must enter the optimum k_v factor for each axis when operating with interpolated axes (see "**The Control Loop, Position Controller, Interrelation of k_v Factor, Feed Rate, and Following Error**").

Rapid Traverse and Feed Rate Limitation

If more than one axis is moved simultaneously, the rapid traverse on the path v_{path} is formed from the appropriate axis components (see "**Axis-Specific Limit Values**").



- In MP_maxFeed, define the maximum rapid traverse for this axis.
- Feed rate and rapid traverse are significantly lower for **Manual Operation:**
- Define the feed rate for manual mode in MP_manualFeed.
- Define the feed rate for rapid traverse in MP_rapidFeed.

If the value in PP_ChnContourFeedMax is greater than the value in MP_maxFeed, the parameter value applies. After the control is switched on, or after an interruption of the PLC run, PP_ChnContourFeedMax is assigned the value 300 000 so that MP_maxFeed becomes effective.

Note: The absolute maximum velocity of this axis is defined in MP_maxFeed. The value is not exceeded.

The maximum possible feed rate depends on the encoder being used.

v_{max} [mm/min] = P [mm] · f_i [kHz] · 60

v_{max} = Maximum traverse speed
P = Signal period of the encoder
f_i = Input frequency of the encoder input (see "**Section 2 - Mounting and Electrical Installation**")

Analog axes: The rapid traverse rate at an analog voltage of 9 V is defined in MP_maxFeedAt9V (see "**Analog Axes**").

Feed Rate Values in PLC Operands

The feed rate values are indicated by PLC operands (see the following tables).

The PLC can influence the following values:

- **PP_ChnContourFeedMax:** Maximum feed rate
- **PP_AxManualFeedMax:** Maximum manual axis feed rate

The following PLC operands contain channel-specific feed rate values.

PLC operand	Type
NN_ChnProgFeedMinute **Programmed feed per minute [mm/rev]**	D
NN_ChnProgFeedRevolution **Programmed feed per revolution [mm/min]**	D
NN_ChnProgFeedThread **Programmed thread feed rate [mm/rev]**	D
NN_ChnProgMinuteActive **Feed per minute is active** 0: Per-minute feed rate is not active 1: Per-minute feed rate is active	M
NN_ChnProgRevolutionActive **Feed per revolution is active** 0: Per-revolution feed rate is not active 1: Per-revolution feed rate is active	M
NN_ChnProgThreadActive **Thread feed rate is active** 0: Thread feed rate is not active 1: Thread feed rate is active	M
NN_ChnContourFeed **Current contouring feed rate [mm/min]** In the manual operating modes, the highest axis feed of all axes is stored in this operand.	D
PP_ChnContourFeedMax **Max. feed rate from PLC [mm/min]**	D

The following PLC operands contain axis-specific feed rate values.

PLC operand	Type
PP_AxManualFeedMax **Maximum axis feed rate in manual mode [mm/rev]**	D
PP_AxTraversePos **Manual traverse in positive direction** 0: Do not move axis 1: Move axis	M
PP_AxTraverseNeg **Manual traverse in negative direction** 0: Do not move axis 1: Move axis	M

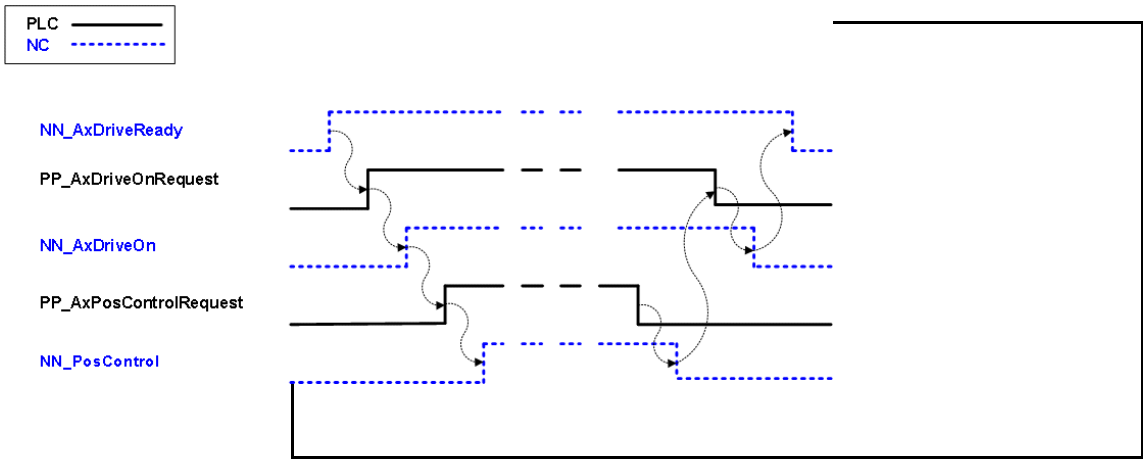
Activating and Deactivating Position Control Loops

The following topics are described:

- **Opening the Position Control Loop**
- **Clamping the Axes**
- **Actual-to-Nominal Value Transfer**

Opening the Position Control Loop

The following figure shows the procedure for switching on the drive motor and activating the position control loop as well as the procedure for opening the position control loop and deactivating the drive.



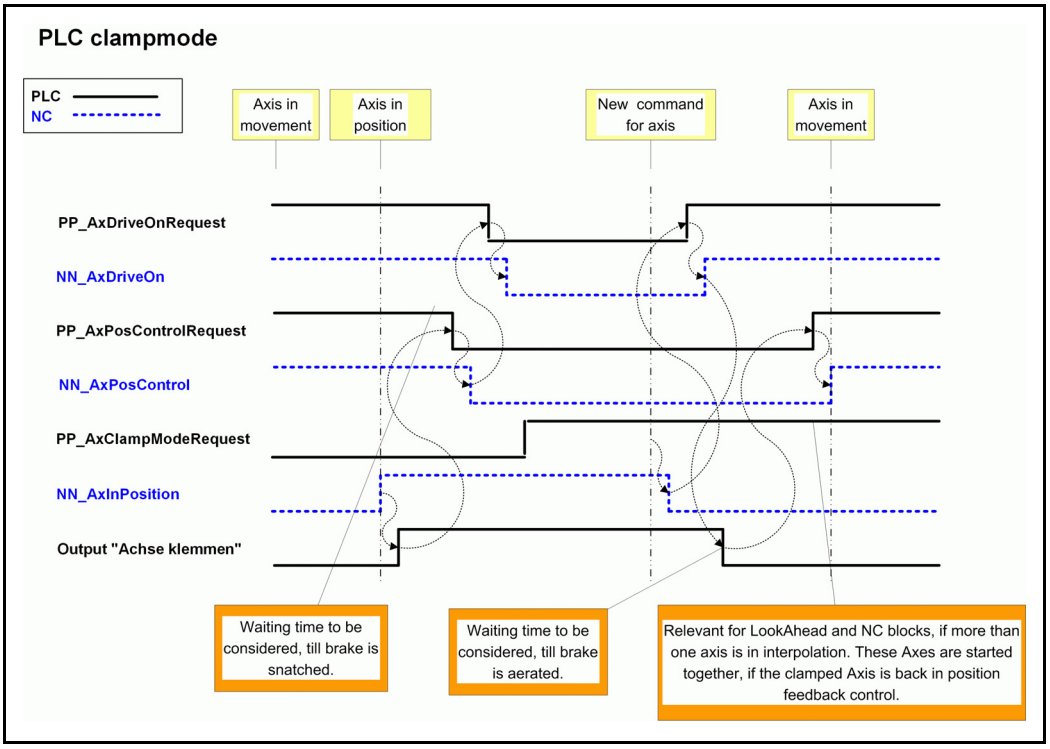
PLC operand	Type
NN_AxDriveReady **Axis drive is ready** 0: Drive not ready for operation 1: Drive ready for operation	M
PP_AxDriveOnRequest **Switch on the axis drive** 0: Do not activate the drive 1: Switch on the drive	M
NN_AxDriveOn **Axis drive is switched on (and is at least speed-controlled)** 0: Drive is off 1: Drive is on	M
PP_AxPosControlRequest **Request for position control of axis** 0: No position feedback control for axis 1: Position feedback control for axis	M

PLC operand	Type
NN_AxPosControl **Axis is position looped** 0: Axis is not position looped 1: Axis is position looped	M

Clamping the Axes

After running an NC block you can clamp the axes.

The following figure shows the procedure for clamping the axes as well as the procedure for unclamping the axis.



PLC operand	Type
PP_AxClampModeRequest **Preparing opening of the position control loop** 0: Not active 1: Active	M
NN_AxInPosition **Axis in position** 0: Axis not in position 1: Axis in position	M

Actual-to-Nominal Value Transfer

During actual-to-nominal value transfer, the current position is saved as the nominal position value. This becomes necessary, for example, if the axis has been moved when the position control loop is open.

There are two ways to turn the actual position into the nominal position:

- Place the request for actual-to-nominal value transfer in the MANUAL OPERATION and EL. HANDWHEEL modes in PP_AxValueActToNominal and check the elimination of the following error in NN_CorrectingLagError.
- To transfer the actual position in all operating modes, use Module 9145.

PLC operand	Type
PP_AxValueActToNominal **Actual-to-nominal value transfer** (Request to eliminate following error) 0: Request to eliminate following error 1: No request to eliminate following error	M
NN_AxCorrectingLagError **Following error eliminated** 0: Following error is not eliminated 1: Following error is eliminated	M

Module 9145 Actual-to-nominal value transfer

Module 9145 is used for an actual-to-nominal value transfer for the axes entered.

Constraints:

- The module functions only in the cyclic PLC program.
- An actual-to-nominal value transfer is possible only if the control is not active (NN_ChnControlInOperation=0) or if there is an M/S/T/T2/G strobe. Actual-to-nominal value transfer can always be performed for axes that are not in an interpolation context.

Call:

PS	B/W/D/K	<>Axes bit-encoded> (Bit 0 corresponds to logic axis 0, etc.)
CM	9145	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Actual-to-nominal value transfer performed
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid axis number
	21	Missing M/S/T/T2/G strobe in NN_ChnControlInOperation=1
	24	Module was called in a spawn job or submit job

Feed-Rate Enable

To move the axes, you must first enable the feed rate through the PLC. Until “feed-rate enable” is set, the nominal velocity value zero is output. The status display shows that the feed rate enable is set / not set.

You can set the feed rate enable for all axes of the NC channel or for specific axes. The PLC run-time system combines PP_ChnFeedEnable and the corresponding axis-dependent feed rate enable PP_AxFeedEnable with an OR gate.

Feed-rate enable for all axes of an NC channel:

- Set PP_ChnFeedEnable.

Axis-specific feed-rate enable:

- Reset PP_ChnFeedEnable.
- Set PP_AxFeedEnable.

PLC operand	Type
PP_ChnFeedEnable **Feed-rate enable for all axes** 0: No feed-rate enable 1: Feed-rate enable	M
PP_AxFeedEnable **Axis-specific feed-rate enable** 0: No feed-rate enable 1: Feed-rate enable	M

The PLC sets PP_ChnWorkFeedEnable if rapid traverse movements are allowed. This marker is set, for example during a tool change or turret actuation, in order to use the time for rapid-traverse movements.

PLC operand	Type
PP_ChnWorkFeedEnable **Rapid traverse enable for all axes** 0: No rapid traverse enable 1: G0 movements are allowed (rapid traverse enable)	M

Controller Parameters for Manual Traverse

Settings in the configuration editor:	
Axis ParameterSets Key for parameter set CfgPosition Filter **manualFilterOrder**	

The following topic is described:

- **Filter Before Position Control Loop**

Filter Before Position Control Loop

The MP_manualFilterOrder parameter differentiates between axes and spindles. In the MANUAL and EL. HANDWHEEL operating modes, or if axes are moved by PLC, a mean-value filter is used as a nominal position value filter for axes.

MP_manualFilterOrder for spindles: see "**Spindles, Filtering the Acceleration Values**".

- Define the order of the mean-value filter for axes in MP_manualFilterOrder.

MP_manualFilterOrder
Order of mean-value filter in Manual mode
Format: Numerical value
Input: 1 to 51
Default: 11

Controller Parameters for Analog Axes

Settings in the configuration editor:	
Axes ParameterSets [Key for parameter block] CfgPosControl **kvFactor** **feedForwardFactor** CfgAxisAnalog **analogOffset** **kvFactor2** **kvSpeedLimit** **accForwardFactor** **compStrength** **compWidth** **compTimeOffset** **compFFAdjust** **compRefAcc** **noOffsetAdjust**	

The following topics are described:

- **General Information**
- **Characteristic Curve Kink Point (Only for Analog Axes)**
- **Acceleration Feedforward Control for Analog Axes**
- **Compensation of Reversal Peaks for Analog Axes**
- **Compensation of Reversal Peaks**
- **Analog Offset**
- **Position Loop Resolution for Analog Axes**

General Information

Analog axis feedback control is based on the following formula:

$$U_{out} = (P_{err} \cdot \mathbf{kvFactor} + V_{nom} \cdot \mathbf{feedForwardFactor} + A_{nom} \cdot \mathbf{accForwardFactor}) \cdot \frac{9V}{\mathbf{maxFeedAt9V}}$$

Value, parameter	Unit	Description
U_{out}	Volt	Output voltage (analog nominal speed value)
P_{err}	mm	Following error (servo lag)
kvFactor	1/s	Kv factor (proportional component of position controller)
V_{nom}	mm/min	Nominal velocity
feedForwardFactor		Factor for velocity feedforward control
A_{nom}	m/s^2	Nominal acceleration
accForwardFactor		Factor for acceleration feedforward control
maxFeedAt9V	mm/min	Assumed velocity of the axis at 9 V

The parameter object **CfgAxisAnalog** is not required for:

- Virtual axes (**MP_axisMode**=Virtual)
- Axes that are for display only (**MP_axisMode**=Display)
- Digital axes (**MP_axisHW**=CC or None)

Characteristic Curve Kink Point (Only for Analog Axes)

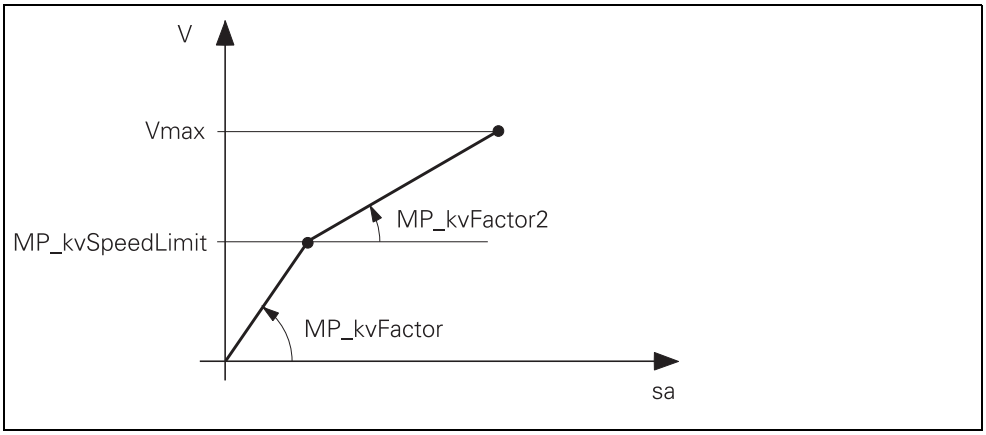
For machines with high rapid traverse, you can not increase the k_v factor enough for an optimum control response to result over the entire velocity range (from standstill to rapid traverse).

In this case, define a characteristic curve kink point, which has the following advantages:

- High k_v factor in the low range
- Low k_v factor in the upper range (beyond the machining velocity range)

- Define the position of the kink point in **MP_kvSpeedLimit.**

- In **MP_kvFactor2,** enter the k_v factor for the upper range.



sa: Following error

The characteristic curve kink point must lie above the tool feed rate!

MP_kvFactor2

Proportional component of position controller above MP_kvSpeedLimit

Format: Numerical value
Input: 0.000 000 000 to 1000 [1/s]
Default: 0 [1/s]

MP_kvSpeedLimit

Limit velocity for MP_kvFactor2

Format: Numerical value
Input: 0.000 000 000 to 36 000 000 [mm/min]
Default: 0.0 [mm/min]

Acceleration Feedforward Control for Analog Axes

MP_accForwardFactor allows you to influence acceleration feedforward control for analog axes.

MP_accForwardFactor

Factor for acceleration feedforward control

Format: Numerical value
Input: 0.000 000 000 to 0.01
Default: 0

Compensation of Reversal Peaks for Analog Axes

The compensation of the reversal peaks affects the nominal speed value, which is output on the analog nominal value output of the control (X8). If an axis reverses its direction of movement, a time-dependent compensation curve is superimposed on the nominal speed value.

You set the compensation of the reversal peaks in the following optional machine parameters:

MP_compStrength

Strength of the compensation
Algebraic sign:
0: No compensation
Positive: Compensation is effective in the direction of acceleration
Negative: Compensation is effective in the direction opposite to acceleration

Format: Numerical value

Input: –999 999 999.999 999 999 to +999 999 999.999 999 999 [mm]

Default: 0

MP_compWidth

Specify, with respect to MP_compTimeOffset=0, the distance from the reversal point at which compensation is to begin.

Format: Numerical value

Input: 0 to +999 999 999.999 999 999 [mm], only positive values

Default: 0.001

MP_compTimeOffset

This parameter merely shifts the compensation curve along the time axis. The width is not changed. The velocity of the axis at which the compensation function is to reach its maximum is defined. This means the higher the acceleration at the reversal point, the closer the maximum will be to the reversal point at the time of direction reversal.
Algebraic sign:
0: Compensation parabola reaches its maximum at the time of direction reversal
Positive: The compensation curve is shifted along the time axis to a later time, which means that the maximum will not be reached until after the direction reversal.
Negative: The compensation curve is shifted along the time axis to an earlier time, which means that the maximum will be reached before the direction reversal.

Format: Numerical value

Input: –999 999 999.999 999 999 to +999 999 999.999 999 999 [mm/min]

Default: 0

Use machine parameter **MP_compFFAdjust** to adjust the surface below the compensation curve as a function of the velocity at the reversal point. The surface entered in the machine parameter **MP_compStrength** is valid for the acceleration entered in **MP_compRefAcc.** The compensation strength is increased or decreased during accelerations that differ from the acceleration in **MP_compRefAcc.**

MP_compFFAdjust

Additive correction of the compensation strength to the machine parameter MP_compStrength
Algebraic sign:
0: The compensation is constant over all acceleration values and is equal to the value in MP_compStrength.
> 0: The surface under the compensation function becomes larger for low accelerations.
< 0: The surface under the compensation function becomes smaller for low accelerations.

Format: Numerical value

Input: –999 999 999.999 999 999 to +999 999 999.999 999 999 [mm]

Default: 0

MP_compRefAcc

For the acceleration given here, the surface under to compensation function is set equal to the value entered in MP_compStrength.

Format: Numerical value

Input: –999 999 999.999 999 999 to +999 999 999.999 999 999 [m/s^2]

Default: 0.03

Compensation of Reversal Peaks

For adjusting the compensation of the reversal peaks, proceed as follows:

- Set machine parameter **MP_compFFAdjust** = 0 and **MP_compRefAcc** = 0 to ensure that the compensation strength is constant over all feed-rate values.
- Now select a typical machining speed and adjust the **MP_compStrength, MP_compWidth** and **MP_compTimeOffset** parameters for the selected speed.
- **MP_compWidth** specifies the duration of compensation and should contain values in the range of a few microns.
- In **MP_compStrength** specify the distance (in [mm]) which the axis will travel if it ideally complies with the transferred nominal speed value. The reversal peak height resulting without compensation is a suitable starting value for the compensation.
- In **MP_compWidth,** enter the duration of compensation. The value entered should lie in the range of a few microns.
- Set the parameter **MP_compTimeOffset** = 0.
- Run a circular interpolation test.
- If the path traversed in the circular test deviates at the reversal point from the nominal path, first toward the inside and then toward the outside, the compensation is performed too early. In this case, you must increase the value in **MP_compTimeOffset.**
- If the path traversed deviates at the reversal point from the nominal path, first toward the outside and then toward the inside, the compensation is performed too late. In this case, you must decrease the value in **MP_compTimeOffset.**

With the following formula you can estimate the magnitude of useful values for the **MP_compTimeOffset** parameter:

$$compTimeOffset = \sqrt{2 \cdot [Beschleunigung] \cdot compWidth} \cdot 6$$

It is easier to adjust the **MP_compWidth** and **MP_compTimeOffset** parameters at low feed rates. ANILAM recommends:

- First roughly adjust **MP_compStrength**
- Then select a low feed rate
- Then set **MP_compWidth** and **MP_compTimeOffset**
- Return to the original feed rate and optimize the value for **MP_compStrength**
- You use these three parameters **(MP_compStrength, MP_compWidth, MP_compTimeOffset)** to adjust the compensation of reversal peaks for a specific feed rate and radius. Under certain circumstances, however, it can be necessary to correct the compensation strength based on the feed rate. Proceed as follows:
- Switch to the **Oscilloscope** mode of operation.
- Set the following value in the oscilloscope by pressing the **SELECTION** soft key:
  Display mode: YT
  Sampling time: IPO clock
  Channel 1: A nom
  Trigger: Free run
- Press the **OSCI** soft key to switch the curve representation.
- Now get the nominal acceleration for the axis **(A nom)** at the reversal point for the feed rate that you have selected for the adjustment up to now

- Press the **START** soft key to start recording.
- Press the axis-direction key of each axis
- Press the **STOP** soft key to stop recording.
- Enter the determined acceleration in the parameter **MP_compRefAcc**

Note: The oscilloscope shows mm/s^2, but the unit of the parameter is actually m/s^2. This means that you must divide the value by 1000.

- Now use the parameter **MP_compFFAdjust** to correct the compensation strength for lower or higher feed rates

Analog Offset

The offset voltage required for analog axes is stored in **MP_analogOffset.** You either enter these values manually or determine them by using the **offset adjustment** function.

During the offset adjustment, the control receives the axis offset values determined by the IPO and enters them in the parameters **MP_analogOffset.**

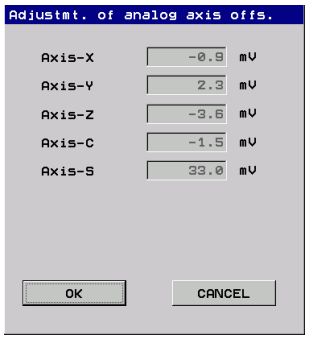
For the offset adjustment, the axes must be in position feedback control.

To adjust the offset:

•Press the MOD key.

•Enter the code number **75 368.**

The control opens the **Adjustment of analog axis offset** dialog box and displays the values determined.



- Press the **OK** button to transfer the offset values to the parameters **MP_analogOffset.**

The maximum permissible offset voltage in the control is +/– 1 V. If this voltage is exceeded, the **offset adjustment** function limits the value.

MP_analogOffset

Offset on analog axis

Format: Numerical value

Input: –1 to +1 [V]

Default: 0 [V]

MP_noOffsetAdjust

Exclude axis from automatic offset adjustment

Format: Array

Selection:

[TRUE]

Axis will be excluded from offset adjustment.

[FALSE]

Offset of the axis is adjusted.

Default: Optional parameter

Position Loop Resolution for Analog Axes

The analog voltage is subdivided 65536-fold with a 16-bit D/A converter. This results in a smallest voltage step of 0.15 mV
This results in the voltage DU per position error or following error s_a.

The control outputs one voltage per position error.

$$\Delta U = \frac{10\,000\ [mV]}{S_a[\mu m]}$$

If DU is divided by the smallest possible voltage step (0.15 mV), the result is the number *n* of the possible voltage steps per position error.

Switching Parameter Blocks

If there is more than one parameter block for one axis, use Module 9434 to select the desired parameter block and Module 9435 to check the currently active parameter block.

Module 9434 responds immediately after execution and reports in the result whether the parameter block could be selected (result=0). It may take some time to switch to the selected parameter block. Use Module 9435 to check which parameter block is active.

Module 9434 does **not** switch off the drive controller. The PLC program can switch the drive controller off using NN_AxDriveOn and reactivate it after the parameter block has been switched.

If the axis is assigned to a machining channel, the execution of the an NC program must be synchronized with the module call.

The following topics are described:

- **Module 9434 Select parameter block**
- **Module 9435 Status of the parameter block of an axis**

Module 9434 Select parameter block

The module activates the parameter block programmed for the drive.

Call:

PS	B/W/D/K	<>Control loop> Index from MP_CfgAxes/axisList
PS	B/W/D/K	<>Parameter block index> 0: Parameter block index 0 1: Parameter block index 1 Etc.
CM	9434	
PL	B/W/D	<>Result> 0: New parameter block selected. 1: Addressed control loop does not exist. 2: Addressed parameter block does not exist. 3: Module was not executed because the axis is active in an NC program. 4: Module was not executed because another command is being performed for this control loop.

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	No error
	1	Error

Module 9435 Status of the parameter block of an axis

The module returns the active parameter block and (if present) the selected parameter block of the control loop.

Call:

PS	B/W/D/K	<>Control loop> Index from MP_CfgAxes/axisList
CM	9435	
PL	B/W/D	<>Index of active parameter block>
PL	B/W/D	<>Index of selected parameter block> –1: No selected parameter block present

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	No error
	1	Error

Monitoring Functions

The following topics are described:

- **Monitoring the Drives**
- **Position Monitoring**
- **Movement Monitoring**
- **Standstill Monitoring**
- **Positioning Window**
- **Temperature Monitoring**
- **Read Actual Utilization of Drive Motors**
- **EMERGENCY STOP Monitoring**

Monitoring the Drives

Settings in the configuration editor:	
System CfgHardware **I32stopsMonitoring**	

The NC monitors the dynamic response of the machine by using the following monitoring functions:

- Position monitoring
- Standstill monitoring
- Movement monitoring

If the fixed values are exceeded, it displays an error message and stops the machine.

You can switch off the monitoring functions for individual axes or for all axes (globally) if the drive enabling is canceled (I32 = 0).

Warning: Safe machine operation is not possible if the monitoring functions are switched off. Uncontrolled axis movements are not detected.

The following topics are described:

- **Switching Off Monitoring Functions Globally**
- **Switching Off Monitoring Functions for Individual Axes**

Switching Off Monitoring Functions Globally

The monitoring functions for all drives are switched off if I32=0 and MP_I32stopsMonitoring=On:

MP_I32stopsMonitoring

Behavior of input I32 (drive enabling)

Format: Drop-down selection menu

Selection:

[On]

If I32=0, all monitoring functions that can be influenced by the PLC are switched off.

[OFF]

Input I32 has no effect on the monitoring functions.

Default: Off

Switching Off Monitoring Functions for Individual Axes

Set PP_AxDeactivateMonitoring to switch off monitoring for individual axes.

PLC operand	Type
PP_AxDeactivateMonitoring **Deactivate monitoring functions** 0: Monitoring functions active 1: Monitoring functions inactive	M

The following table shows the status of monitoring as a function of the axis-specific PLC operand PP_AxDeactivateMonitoring and drive enabling (I32) if MP_I32stopsMonitoring=On.

Monitoring functions for		PP_AxDeactivateMonitoring=	MP_I32stopsMonitoring=On; I32=
Individual servo drive	All servo drives		
Inactive	Inactive	0	0
Active	Active	0	1
Inactive	Inactive	1	0
Inactive	Active	1	1

Position Monitoring

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgPosControl **servoLagMin1** **servoLagMax1** **servoLagMin2** **servoLagMax2** CfgControllerAuxil **driveOffLagMonitor** CfgReferencing **endatDiff**	

The Parameter object

•CfgControllerAuxil is not required for:
- Virtual axes (**MP_axisMode**=**Virtual**)
- Axes that are for display only (**MP_axisMode**=**Display**)

•CfgReferencing is not required for:
- Virtual axes (**MP_axisMode**=**Virtual**)

The axis positions are monitored by the control as long as the control loop is closed.

The input values for position monitoring depend on the maximum possible following error (servo lag). Therefore the input ranges for operation with following error and velocity feedforward are separate.

For both modes of operation there are two range limits for position monitoring.

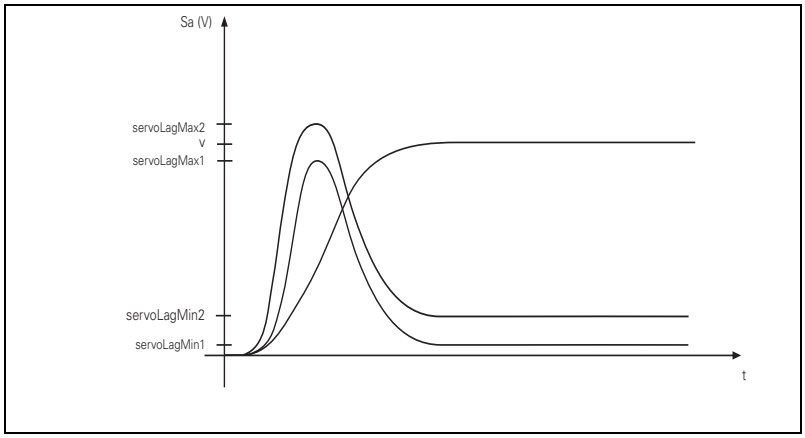
If the first limit (MP_servoLagMin1, MP_servoLagMax1) is exceeded, the warning **Excessive servo lag in [axis]** is displayed. The machine stops.

This message can be cleared. An actual-to-nominal value transfer is then executed for the respective axes.

If the second limit (MP_servoLagMin2, MP_servoLagMax2) is exceeded, the emergency-stop error message **Excessive servo lag in [axis]** is displayed.

The control-is-ready signal output is reset. The machine stops. You cannot clear this message. You must restart the control to correct the error.

- In the machine parameters given below, define two range limits for position monitoring.
- Adjust the input values to the machine dynamics.



MP_servoLagMin1/2 applies for constant feed rates, MP_servoLagMax1/2 during changes in feed rate.

MP_servoLagMin1

Minimum value for following-error monitoring (clearable)

Format: Numerical value

Input: 0.000 000 000 to 100 [mm] or [°]

Default: 1 [mm] or [°]

MP_servoLagMax1

Maximum value for following-error monitoring (clearable)

Format: Numerical value

Input: 0.000 000 000 to 100 [mm] or [°]

Default: 5 [mm] or [°]

MP_servoLagMin2

Minimum value for following-error monitoring (emergency stop)

Format: Numerical value

Input: 0.000 000 000 to 100 [mm] or [°]

Default: 1 [mm] or [°]

MP_servoLagMax2

Maximum value for following-error monitoring (emergency stop)

Format: Numerical value

Input: 0.000 000 000 to 100 [mm] or [°]

Default: 5 [mm] or [°]

The following topic is described:

- **Clamped Axes, Hanging Axes**

Clamped Axes, Hanging Axes

Clamped axes or hanging axes are monitored when the drive motor is switched off if **MP_driveOffLagMonitor** is activated. The value from **MP_servoLagMax2** is monitored.

MP_driveOffLagMonitor

Servo-lag monitoring when drive motor switched off

Format: Drop-down selection menu

Selection:

[On]

Monitoring of hanging axes is active

[OFF]

Monitoring of hanging axes is not active

Default: Off

Note: The monitoring functions for hanging axes can **not** be switched off by using **MP_I32stopsMonitoring**.

Difference between position at switch-on and shutdown

When the control is switched off, the actual position of the axes is saved with an absolute encoder. During switch-on it is compared with the position values read by the encoder.

If the positions differ by more than the difference defined in MP_endatDiff, a pop-up window is displayed with both positions. The new position must be confirmed with a soft key. If it is not confirmed, the error message **Check the position encoder <axis>** is displayed.

If the pop-up window is displayed although the motor is located at the correct position, you can acknowledge the message.

Note: The cause for one of the above listed messages can also be a defect in the encoder or control.

MP_endatDiff

Permissible difference of EnDat encoders during switch-on

Format: Numerical value

Input: –100 000.000 000 000 to +100 000 [mm] or [°]

0 = Off

Default: 0

Movement Monitoring

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgEncoderMonitor **movementThreshold** **lagTolerance**	

Movement monitoring is possible during operation both with velocity feedforward and with following error.

When the nominal movement is completed in the position controller, following errors greater than **MP_lagTolerance** are summated at each interpolator clock pulse. If this sum exceeds the limit defined in **MP_movementThreshold** at a later time, the error message **MOVEMENT MONITORING <AXIS>** is displayed. The greater the following error, the less time it will take until an error message is issued.

An error message is issued if the following condition is fulfilled:

$$\text{MP\_movementThreshold} < \sum \frac{\text{following error}}{\text{time}}$$

- In **MP_movementThreshold,** enter the distance per minute from which movement monitoring should go into effect.
- Enter the tolerance value in **MP_lagTolerance.** Following errors smaller than MP_lagTolerance are ignored.

Warning: If **MP_movementThreshold = 0,** movement monitoring is not active. Safe machine operation is not possible without the movement monitoring function.

MP_movementThreshold

Threshold from which movement monitoring is effective

Format: Numerical value

Input: 0.000 000 000 to 36 000 000 [mm/min] or [°/min]
0 = Monitoring is off

Default: 600 000 [mm/min] or [°/min]

MP_lagTolerance

Tolerance at and above which the following error is included

Format: Numerical value

Input: 0.000 000 000 to 36 000 000 [mm] or [°]

Default: 0 [mm] or [°]

Standstill Monitoring

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgControllerAuxil **checkPosStandstill**	

Standstill monitoring is effective during operation both with velocity feedforward and with following error, as soon as the axes have reached the positioning window.

If the position difference is greater than the value defined in MP_checkPosStandstill, the blinking error message **Standstill monitoring in [axis]** is displayed. The message also is displayed if, while moving to a position, an overshoot occurs that is larger than the input value in **MP_checkPosStandstill,** or if the axis moves in the opposite direction when beginning a positioning movement:

In **MP_checkPosStandstill,** enter a threshold from which the standstill monitoring should go into effect.

MP_checkPosStandstill
Standstill monitoring
Format: Numerical value
Input: 0.001 000 000 to 10 000 [mm]
Default: 10 000 [mm]

Positioning Window

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgControllerTol **posTolerance** **timePosOK**	

The parameter object CfgControllerTol is not required for:

- Virtual axes (**MP_axisMode**=Virtual)
- Axes that are for display only (**MP_axisMode**=Display)

If the axes have reached the positioning window after a movement, the status is shown in NN_AxInPosition. This also applies to the status after the machine control voltage is switched on.

The NC resets NN_AxInPosition as soon as you start a positioning movement or traverse the reference marks.

In the EL. HANDWHEEL mode of operation NN_AxInPosition for the current handwheel axis is reset.

On contours that can be machined with constant surface speed, NN_AxInPosition is not set.

Danger: **incompatibility with earlier TNC controls!**
When an NC program is run, the 6000i does not set the operand **NN_AxInPosition** until the beginning of the following NC block. It is not set at the end of the current block! Example of **Program Run, Single Block** mode of operation:
Program:
.....
N5 L W+30
N6 L X+10
N7 L W+60
.....
If the block N5 is run in Single Block mode,

- the program cursor is located on block N6.
- the W axis is located at +30.
- the status of the operand is:
  NN_AxInPosition = 0
  In this example the PLC clamps the W axis and the Marker NN_AxInPosition is set. But the marker is not set until the next block N6 is run. (With earlier TNC controls, the marker is already set at the end of block N5).

Take this status into account in your PLC program.

The following topics are described:

- **Axes in Position**
- **Axes in Motion**

Axes in Position

The control reports that the axis is in position (NN_AxInPosition) when the axis has remained within the positioning window MP_posTolerance for the period of time defined in MP_timePosOK. After the position has been reached, the control begins running the next block. The position controller can correct a disturbance inside this window without activating the “Return to the Contour” function.

- In MP_posTolerance, define the size of the positioning window.
- In MP_timePosOK, define the period of time which the axis is to remain within the positioning window.

MP_posTolerance

Positioning window

Format: Numerical value

Input: 0.001 000 000 to 2 [mm]

Default: 0.005

MP_timePosOK

Hysteresis time for positioning window reached

Format: Numerical value

Input: 0.000 000 000 to 20 [s]

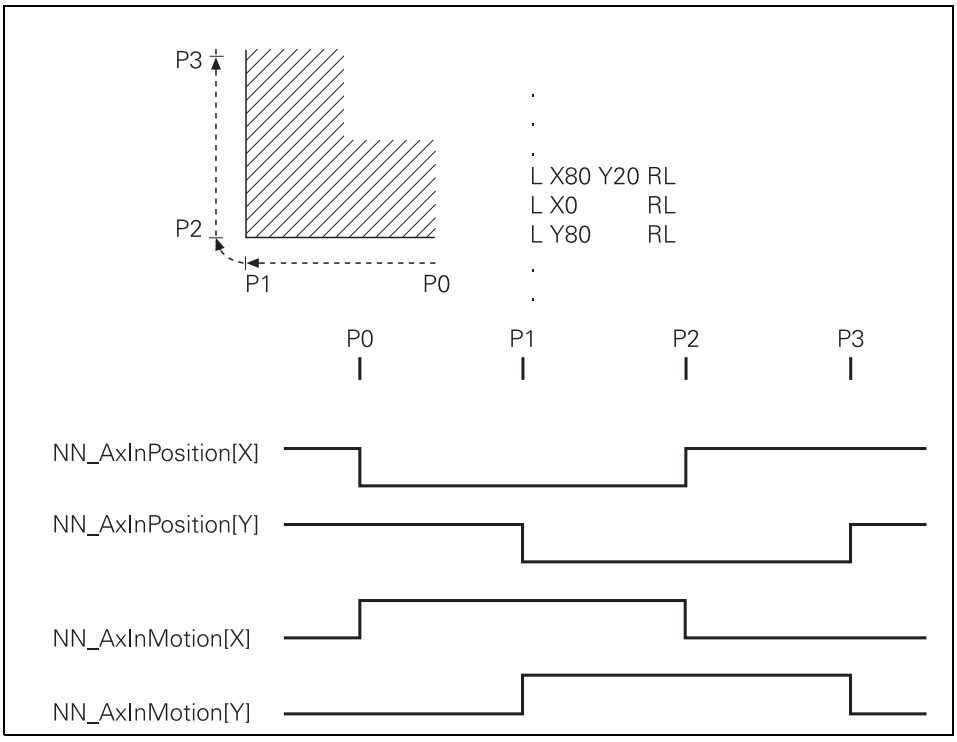
Default: 0.01 [s]

PLC operand	Type
NN_AxInPosition **Axes in position** 0: Axis not in positioning window 1: Axis in positioning window	M

Axes in Motion

During an axis movement, the NC sets NN_AxInMotion.

PLC operand	Type
NN_AxInMotion **Axes in motion** 0: Axis not in motion 1: Axis in motion	M



Temperature Monitoring

The following topics are described:

- **Temperature of the MC**
- **Interrogate the Values of the Internal ADCs**

Temperature of the MC

The internal temperature of the MC is continuously monitored. At about 55 °C, the early warning **Temperature warning** is displayed. If the temperature does not fall below 55 °C, the warning is reactivated after two minutes. Beginning at about 60 °C the error message **Temperature too high <temperature> °C** is displayed and an emergency stop is triggered. If the machine is switched on again and the temperature does not go below 60 °C, the error message is reactivated after about 10 to 20 seconds.

Interrogate the Values of the Internal ADCs

Module 9133 allows you to interrogate the internal values of the analog-to-digital converter of the MC.

Module 9133 Interrogate the values of the internal ADCs

Call:

PS	B/W/D/K	<>Code> 0: Internal temperature sensor in [°C] 1: Temperature CPU1 (basic PCB) in [°C] 2: Temperature CPU2 (additional PCB) in [°C] 3: Voltage of buffer battery in [mV] 4: 5 V supply voltage of the main board 5: 3.3 V supply voltage of the main board 6: Rotational speed of the housing fan
CM	9133	
PL	B/W/D	<>Value>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Value ascertained
	1	Value could not be determined; error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid code programmed
	8	No second processor present (for number 2)

Read Actual Utilization of Drive Motors

Module 9166 provides the momentary utilization of the given drive motor as a percentage value.

Utilization means:

Speed range	n_{actl} < rated speed	n_{actl} rated speed
Asynchronous motor	$\frac{\lvert M \rvert}{\lvert M_{Nenn} \rvert}$	$\frac{\lvert P \rvert}{\lvert P_{Nenn} \rvert}$
Synchronous motor	$\frac{\lvert M \rvert}{\lvert M_{Nenn} \rvert}$	–

Instead of the drive torque, one uses the effective component I_q of the current, which is proportional to the torque.

I_{qMean} is formed as mean value of the individual current values I_{qx} of the last 20 ms:

$$I_{qMittel} = \frac{\sum (I_{q1} .. I_{qn})}{n}$$

$$Auslastung = 1000 \cdot \frac{I_{qMittel}}{I_{qNenn}}$$

For Asynchronous Motors:

$$I_{qNenn} = \sqrt{I_N^{\,2} - I_{mag}^{\,2}}$$

I_N: Rated current of motor
I_{mag}: Magnetizing current

For Synchronous Motors:

I_{qRated} = <Rated current of motor>

The utilization display of synchronous motors is with respect to the rated torque (M/M_{rated}).

Module 9166 Read current utilization of drive motor

The module provides the momentary utilization of a drive motor controlled by the integral current controller in percent (%) of the rated utilization.

The value 0 is read for axes that are not connected to the servo drive.

Call:

PS	B/W/D/K	<>Axis> Index from MP_CfgAxes/axisList
CM	9166	
PL	B/W/D	<>Utilization of the drive in %>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error
	1	Control has no current controller

EMERGENCY STOP Monitoring

On the control there is a PLC output (X41/34) with the designation control-is-ready, and a PLC input (X42/4) with the designation control-is-ready-acknowledgement for the EMERGENCY STOP loop.

If a functional error is detected in the control, the control switches the control-is-ready output off. An error messages is displayed and the PLC program is stopped. You can **not** clear this error message:

- Correct the error and restart the switch-on routine.

If the control-is-ready-acknowledgement input is switched off by a process external to the control, the error message **EXTERNAL EMERGENCY STOP** is displayed. The NC sets **NN_GenNcEmergencyStop.** The nominal speed value 0 is output and the drives are switched off. You can clear this error message after switching the machine control voltage back on.

The “control-is-ready signal acknowledgment” input is passed directly onto the NC; it can **not** be manipulated by the PLC (I3).

PLC operand	Type
NN_GenNcEmergencyStop **Control in external emergency stop** 0: Control is not in external emergency stop 1: Control is in external emergency stop	M

The following topics are described:

- **Connection Diagram**
- **Flowcharts**

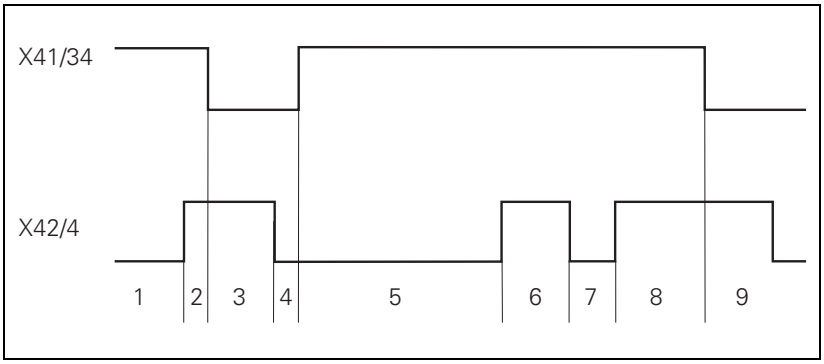
Connection Diagram

In the event of an error, the control-is-ready output must trigger an emergency stop. The control therefore checks this output every time that line power is switched on.

Note: The circuitry recommended by ANILAM is illustrated in the **Figure 9-24, Basic System Diagram**.
Ensure that the control-is-ready acknowledgment (input X42/4) occurs within 1 second.

Flowcharts

Flow chart for emergency stop test:



Step	Function	Screen display
1	Waiting for machine control voltage	**RELAY EXTERNAL DC VOLTAGE MISSING**
2	Recognition of the machine control voltage on input X42/4 and switch-off of the control-is-ready signal on X41/34 by host computer (t < 66 ms)	
3	Maximum time within which the control-is-ready acknowledgment on input X42/4 must go to zero (t < 380 ms)	If exceeded **EMERGENCY STOP DEFECTIVE**
4	Recognition of the acknowledgment and setting of output X41/34	
5	Waiting for machine control voltage	**RELAY EXTERNAL DC VOLTAGE MISSING**
6	Normal control operation. Control-is-ready output and acknowledgment are high.	
7	Control voltage is switched off externally.	**EMERGENCY STOP**
8	After switching on again, the machine control voltage can be switched off, and then the control operates normally.	
9	After detecting a fault, the control switches off the control-is-ready output (X41/34).	**Error message** (reset error)

Spindles

The following topics are described:

- **Configuring Spindles**
- **Spindle Position Encoder**
- **Filtering the Acceleration Values**
- **Controlling the Spindle**
- **Stop Spindle at Trip Dog Position**
- **Spindle for Per-Revolution Feed**
- **Gear Shifting**
- **Tapping**

Configuring Spindles

Settings in the configuration editor:	
System CfgAxes **spindleIndices**	

The list index of a spindle key defines the programmable spindle number used by the PLC to identify the spindle. Spindlesare indicated by sequential numbering starting from the index [0].

The spindle keys you define must be contained in **MP_CfgAxes/axisList.**

MP_spindleIndices
Key names for all spindles on the machine
Format: Array [0–n] (# = logical spindle number)
Input: Key names from MP_CfgAxes/axisList

The PLC indicates the number of configured spindles and the logical spindle number in the following PLC operands.

PLC operand	**Type**
NN_GenSpiCount **Number of configured spindles**	D
NN_SpiLogNumber **Logic axis number of spindle** 0–n: Logical axis number –1: Spindle does not exist –2: Spindle deactivated (example: alternation between C axis and spindle)	D

Spindle Position Encoder

Analog and digital spindles can be driven in a closed position control loop. In this case the spindle needs its own position encoder, or you use the speed encoder to measure the position of a digital spindle.

Due to the higher required accuracy, the position encoder should be mounted directly on the spindle.

If the position encoder cannot be mounted to the spindle, the encoder will output several reference pulses per revolution. For example, with a transmission of 4:1 (motor to spindle), you will receive four reference pulses (every 90°) per spindle revolution.

Evaluate the reference mark with Module 9220 (see "**Renewed Traversing of the Reference Marks**").

In NN_SpiReferenceAvailable, the NC reports whether the reference position of the spindle has been determined.

PLC operand	Type
NN_SpiReferenceAvailable **Reference position determined** 0: Reference position not determined 1: Reference position determined	M
PP_SpiReferenceMarkSignal **Trip dog for reference end position** 0: Trip dog not triggered 1: Trip dog triggered	M

Filtering the Acceleration Values

Settings in the configuration editor:	
Axis ParameterSets Key for parameter set CfgPositionFilter **manualFilterOrder**	

The parameter object CfgPositionFilter is not required for:

- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

The MP_manualFilterOrder parameter differentiates between axes and spindles. Linear acceleration is used for operating spindles. MP_manualFilterOrder allows you to filter the acceleration values.

MP_manualFilterOrder for axes: see "**Filter Before Position Control Loop**".

- Define the order of the mean-value filter for spindles in MP_manualFilterOrder.

MP_manualFilterOrder
Order of mean-value filter in Manual mode
Format: Numerical value
Input: 1 to 51
Default: 11

Controlling the Spindle

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgControllerTol **speedTolerance** **timeSpeedOK**	

The parameter object CfgControllerTol is not required for:

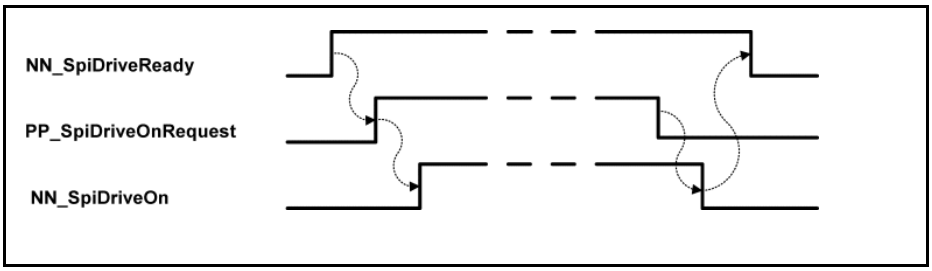
- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

The following topics are described:

- **Switching the Spindle Drive On/Off**
- **Spindle Control by PLC**
- **Tolerances for “Rotational Speed Reached”**
- **Positioning the Spindle (M19/Trip dog position)**
- **Tapping**
- **Oriented Spindle Stop (Spindle Point Stop)**

Switching the Spindle Drive On/Off

The following figure shows the procedure for switching the spindle drive on/off.



The NC or PLC provides the information on controlling the spindle in the following PLC operands.

PLC operand	Type
NN_SpiDriveReady **Spindle drive is ready** 0: Drive not ready for operation 1: Drive ready for operation	M
PP_SpiDriveOnRequest **Switch on the spindle drive** 0: Do not activate the drive 1: Switch on the drive	M
NN_SpiDriveOn **Spindle drive is switched on (and is at least speed-controlled)** 0: Drive is off 1: Drive is on	M
PP_SpiSpeedMax **Maximum spindle speed**	D
PP_SpiEnable **Spindle enabling** 0: Spindle not enabled 1: Spindle enabled	M
NN_SpiInMotion **Spindle in motion** 0: Spindle not in motion 1: Spindle in motion	M

Spindle Control by PLC

The PLC controls the spindle by using the following modules:

- **Module 9410: Read spindle status**
- **Module 9412: Stop the spindle**
- **Module 9413: Rotate the spindle**
- **Module 9414: Position the spindle (M19)**

Status request by

Module 9410: Read spindle status

Note: **PP_SpiEnable=1** must be set for a spindle movement to be executed.

- Module 9410Read spindle status

The module reads the status of the specified spindle.

Call:

PS	B/W/D/K	<>logic spindle number>
CM	9410	
PL	D	<>Spindle status>
		1: No job active – Last job was OK
		2: No job active – Last job was faulty
		3: Job is being executed
PL	D	<>Spindle mode>
		1: Spindle at standstill
		2: Spindle turning clockwise
		3: Spindle turning counterclockwise
		4: Spindle is position-looped (M19)

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status determined
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid spindle number

Module 9412 Stop the spindle

The module stops the specified spindle (M5 status).

Call:

PS	B/W/D/K	<>logic spindle number>
CM	9412	
PL	D	<>Error code>

1: Incorrect module call

2: No permission for module call (example: NC is cutting a thread at the time it is instructed to execute a command)

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status has been read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid spindle number
	9	Module call not possible at this time

Module 9413 Move the spindle

The module rotates the specified spindle in CW/CCW direction (M3/M4) at a constant rotational speed or at a constant cutting speed.

Call:

PS	B/W/D/K	<>logic spindle number>
PS	B/W/D/K	<>Mode> Bit 0–1: Direction of rotation 01 = Rotation CW (M3) 10 = Rotation CCW (M4) Bit 2–3: Type of rotation 01 = Constant cutting speed 10 = Constant rotational speed
PS	D/K	<>Rotational speed or cutting speed> Constant cutting speed in [m/min] Constant rotational speed in [rpm]
CM	9413	
PL	D	<>Error code> 1: Incorrect module call 2: No permission for module call 3: Rotational speed not allowed 4: Mode not allowed

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status has been read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid spindle number
	9	Module call not possible at this time

Tolerances for “Rotational Speed Reached”

The control reports that the rotational speed is reached if the spindle rotation remains within the control window of MP_speedTolerance for the period of time defined in MP_timeSpeedOK.

- In MP_speedTolerance, define the size of the control window.
- In MP_timeSpeedOK, define the period of time which the rotational speed (feed rate) is to remain within the control window.

MP_speedTolerance

Rotational speed (feed rate) window

Format: Numerical value

Input: 0.000 000 000 to 1000 [mm/min] or [°/min]

Default: 3000 [mm/min] or [°/min]

MP_timeSpeedOK

Hysteresis time for monitoring the speed deviation

Format: Numerical value

Input: 0.000 000 000 to 20 [s]

Default: 0.01 [s]

The NC provides the information “Rotational speed reached” in NN_SpiSpeedOK.

PLC operand	Type
NN_SpiSpeedOK **Spindle speed reached** 0: Spindle speed not reached 1: Spindle speed reached	M

Positioning the Spindle (M19/Trip dog position)

The PLC command defined in:

- Module 9414 instructs the NC to activate spindle positioning.
- Module 9412 instructs the NC to deactivate spindle positioning.

Use Module 9414 to switch on position feedback control. Position feedback control is effective until it is switched off by Module 9412.

Module 9414 Position the spindle

The module is used for the following functions:

- Position the spindle (M19): The spindle is stopped at the specified position. The mode defines the direction of rotation.
- Stop the spindle at the trip dog position (mode bit 3=1): The spindle is positioned to the trip dog at the specified rotational speed (see "**Stop Spindle at Trip Dog Position**").

Call:

PS	B/W/D/K	<>logic spindle number>
PS	B/W/D/K	<>Mode> Bit 0–2: Direction of rotation 000 = Shortest direction of rotation, absolute position entry 001 = Rotation CW, absolute position entry 010 = Rotation CCW, absolute position entry 100 = Relative position entry, rotational direction from algebraic sign of position entry Bit 3: Select the function 0 = "Position the spindle" function (M19)" 1 = "Stop spindle at trip dog position" function
PS	D/K	<>Absolute or relative position in [0.0001°]>
PS	D/K	<>Rotational speed in [0.0001 rpm]>
CM	9414	
PL	D	<>Error code> 1: Incorrect module call 2: No permission for module call 3: Rotational speed not allowed 4: Mode not allowed

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status has been read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid spindle number
	9	Module call not possible at this time

The NC provides the status information on spindle positioning in the following PLC operands.

PLC operand	Type
NN_SpiControl **Spindle in position control loop** 0: Spindle is not in position control loop 1: Spindle is in position control loop	M
NN_SpiControlInPos **Spindle in position** 0: Spindle is not in position 1: Spindle is in position	M

Tapping

The NC puts the spindle in the position control loop during tapping and thread cutting (with Cycle 18 for TNC controls). The job is not transmitted by the PLC.

The NC shows in the following PLC operands that a tapping operation is currently being executed. (Both markers were set simultaneously.)

PLC operand	Type
NN_SpiTapping **Tapping active** 0: Tapping not active 1: Tapping active	M
NN_SpiRigidTapping **Tapping with spindle interpolated with Z axis active** 0: Tapping not active 1: Tapping active	M

Oriented Spindle Stop (Spindle Point Stop)

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgSpindle **M19MaxSpeed**	

Note: The spindle position must be measured by an encoder before an oriented spindle stop can be executed. If the parameter MP_CfgAxisHardware/ posEncoderType is set to **no encoder,** an oriented spindle stop is not possible.

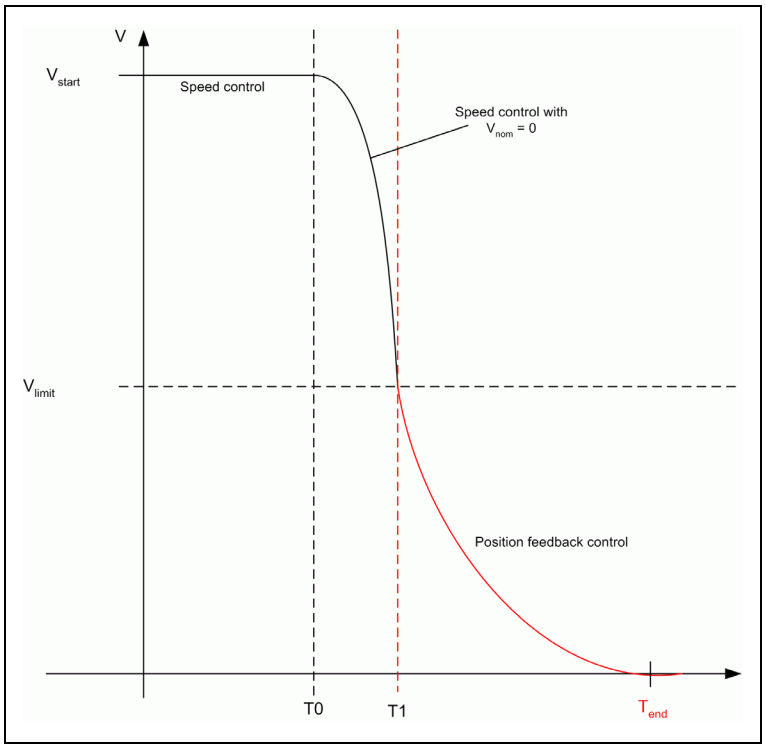
The following topics are described:

- **Oriented Spindle Stop with Rotating Spindle**
- **Oriented Spindle Stop with Stationary Spindle**

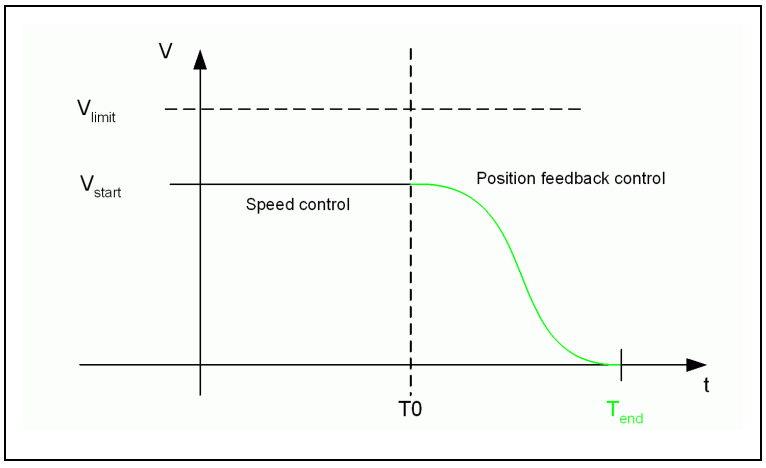
Oriented Spindle Stop with Rotating Spindle

An oriented spindle stop (spindle point stop) with **rotating spindle** and at high speeds is executed in two steps if you set **MP_M19MaxSpeed** accordingly:

- Rotational speed > 120 % of **MP_M19MaxSpeed** (phase 1):
  The spindle is braked at the limit of current until the speed limit is reached (speed control with V_{nom} = 0).
- Rotational speed < 120 % of **MP_M19MaxSpeed** (phase 2):
  The position controller is switched on at the rotational speed limit. The actual position, actual speed and actual acceleration at the time of the transition of phase 1 to 2 are the initial values for positioning under position feedback control. This results in a continuous movement, speed, and acceleration until the target position is reached. The jerk is limited during deceleration and positioning, whereby the maximum jerk can be set.



The oriented spindle stop with **rotating spindle** and at a speed < 120% of MP_M19MaxSpeed is executed under position feedback control.



The rotational speed limit must not be too low because the position controller cycle time would then make it impossible to generate jerk-limited positioning commands. The rotational speed limit of 3 000 rpm is therefore not violated. If you enter smaller parameter values, the limit is increased to 3 000.

MP_M19MaxSpeed

Maximum rotational speed limit for M19

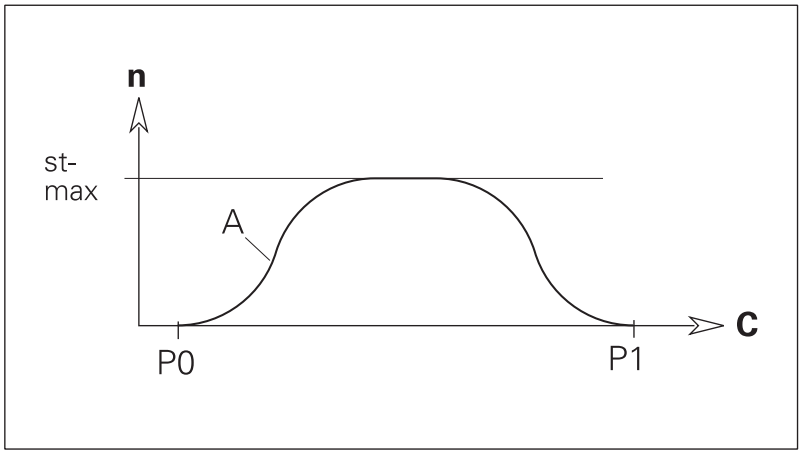
Format: Numerical value

Input: 1000 to 20 000 [1/min]

Default: 1000 [1/min]

Oriented Spindle Stop with Stationary Spindle

A ramp algorithm determines the acceleration with a **stationary spindle**. MP_maxAcceleration determines the steepness of the ramp and MP_maxFeed or the rotational speed from Module 9414 limits the ramp.



Legend:

- **P0:** Initial position
- **P1:** Target position
- **st-max:** MP_maxFeed or the rotational speed from 9414
- **A:** MP_maxAcceleration

Stop Spindle at Trip Dog Position

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgSpindle **fastInputType** **fastInput** **zeroPosEdge** **maxDeceleration**	

The parameter object CfgSpindle is not required for:

- Axes that are **not** defined as spindles (not entered in MP_spindleIndices)
- Virtual axes (MP_axisMode=Virtual)
- Axes that are for display only (MP_axisMode=Display)

A spindle that is not driven in a position control loop can also be stopped at a defined position (trip dog position). It is a prerquesite that this position is reported to a PLC input (X42).

During a spindle stop, the PLC transfers the rotational speed for “waiting for the input signal” (see “**Module 9414 Position the spindle**”) in Module 9414.

The spindle is stopped in three steps.

1 The spindle is decelerated until the defined rotational speed is reached.

2 The spindle continues rotating at the defined rotational speed.

3 The spindle is stopped immediately (without ramp) as soon as the input signal of the spindle trip dog is detected.

MP_fastInputType

Treatment of the fast input for the spindle

Format: Drop-down selection menu

Selection:

[None]
Spindle does not supply any signal or the signal is not evaluated.

[ForStopping]
Spindle stops at trip dog position.

[ForReferencing]
Not yet supported.

Default: None

MP_fastInput

Number of the fast input for the spindle on X42

Format: Numerical value

Input: 0 to 31 (for I0 to I31)

Default: 0

MP_zeroPosEdge defines the trip dog edge, which defines the spindle stop position in positive direction of rotation.

MP_zeroPosEdge

Edge evaluation

Format: Drop-down selection menu

Selection:

[zeroOne]

Zero-one transition is evaluated.

[oneZero]

One-zero transition is evaluated.

MP_maxDeceleration

Define a brake ramp for the spindle different from the acceleration ramp. You usually enter the same value as in MP_maxAcceleration.

Format: Numerical value

Input: 0.000 000 000 to 1000 [m/s^2] or [$1000°/s^2$]

Default: 3 [m/s^2] or [$1000°/s^2$]

The following topic is described:

- **Spindle for Per-Revolution Feed**

Spindle for Per-Revolution Feed

Settings in the configuration editor:	
NCchannel CfgKinModel Key for kinematics model **activeSpindle**	

- In MP_activeSpindle, define the spindle to be used for calculating the feed rate per revolution [mm/rev].

MP_activeSpindle

Key of the active spindle of this kinematics model

Format: String

Input: Key names from MP_CfgAxes/axisList

Gear Shifting

The PLC is responsible for gear shifting. The PLC also manages the parameters that are required for gear shifting.

A separate parameter block can be created for every gear range.

The PLC analyzes the rotational speed and switches to the gear range defined for this rotational speed.

Tapping

Tapping is executed with position feedback control. The spindle and the tool axis interpolate with each other. A floating tap holder is not required.

An oriented spindle stop is executed before tapping. In this way, each axis position is assigned to a certain spindle position. This synchronization makes it possible to cut the same thread more than once. The NC orients the spindle.

The feed-rate override can be changed during tapping. The control automatically adjusts the rotational speed to the changed feed rate. The speed override has no function during tapping.

- Define another parameter block and switch to this parameter block if you want to achieve a specific control response for tapping.

Integrated Oscilloscope

The following topics are described:

- **Fundamentals**
- **Prepare Recording**
- **Record Signals**
- **Analyze Recording**
- **Saving and Loading Recordings**
- **Configure the Colors of the Oscilloscope Display**

Fundamentals

The control features an integrated oscilloscope. This oscilloscope features 6 channels for recording analog signals and 16 channels for recording digital signals (see the following tables).

The following topics are described:

- **Overview of Signals**
- **Sampling Rate**

Overview of signals

Analog signals:	Meaning	CC signal
Saved	The signal last recorded on this channel is “frozen.”	–
A act	Current axis acceleration value [m/s^2] or [°/s^2]. Calculated from position encoder.	–
A nom	Nominal axis acceleration value [m/s^2] or [°/s^2]	–
V act	Actual value of the axis feed rate [mm/min] or [°/min]. Calculated from position encoder.	–
V nom	Nominal value of the axis feed rate [mm/min] or [°/min]. Axis feed rate calculated from the difference from the nominal position values. The following error is not included.	–
Feed rate F	Contouring feed rate [mm/min] or [°/min]	–
P act	Actual position [mm] or [°]	–
P nom	Nominal position [mm] or [°]	–
P err	Following error of the position controller [mm] or [°]	–
P diff	Difference between position and speed encoder [mm] or [°]	–
Position: A	Signal A of the position encoder	–
Position: B	Signal B of the position encoder	–
J act	Current jerk value [m/s^3]. Calculated from position encoder.	–
J nom	Nominal value of the jerk [m/s^3]	–
V (N act)	Shaft speed actual value [mm/min]. Calculated from speed encoder	CC
V (N nom)	Nominal velocity value [mm/min]. Output quantity of the position controller	CC
I (N int)	Integral-action component of the nominal current value [A]; CC 600: peak value, CC 424: effective value	CC

Analog signals:	Meaning	CC signal
I (nom)	Nominal current value [A] that determines torque; CC 600: peak value, CC 424: effective value	CC
PLCPrePgm	The PLC operands (B, W, D, I, O, T, C) are recorded **before** the PLC program run. For types B, W, and D the contents are recorded, and for the other types the logical state of the operands are recorded.	–
PLCPostPgm	The PLC operands (B, W, D, I, O, T, C) are recorded **after** the PLC program run. For types B, W, and D the contents are recorded, and for the other types the logical state of the operands are recorded.	–
Analog	Analog axis/spindle: Analog voltage = nominal velocity value [mV]	–
OFF	No recording for this channel	–

Digital signals	Meaning
M	PLC markers
I	Input
O	Output
T	Timer
C	Counter
X	Reserved

Note: The PLC operands are addressed with numbers in the oscilloscope. You get the numerical addresses from the PLC diagnostic function **Watch List**.

Sampling Rate

The resolution of the internal oscilloscope is at most 600 µs.

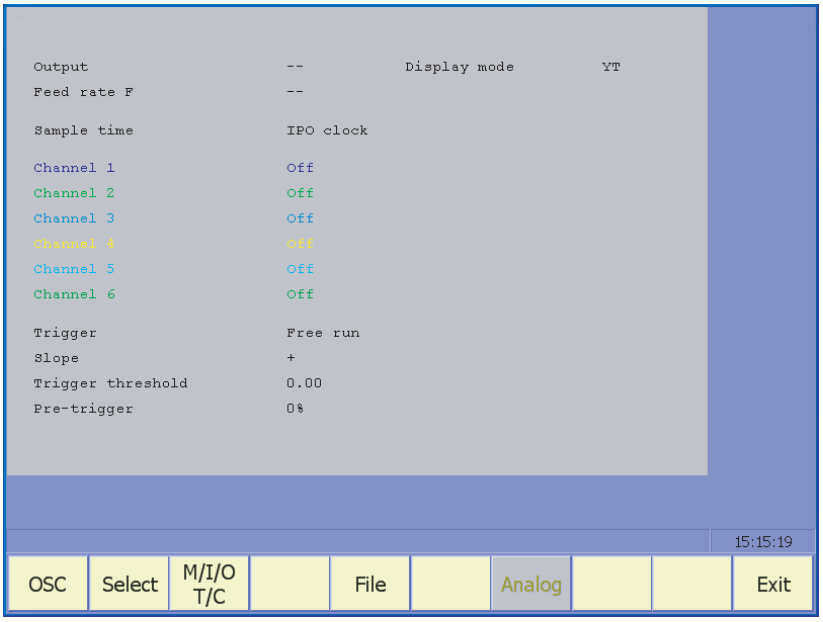
Prepare Recording

The following topics are described:

- **Start Oscilloscope**
- **Setup for Analog Signals**
- **Setup for Digital Signals**

Start Oscilloscope

- Press the **OSC** (**SHIFT** + **F7**) soft key, enter the password, and the control displays the Oscilloscope main screen:

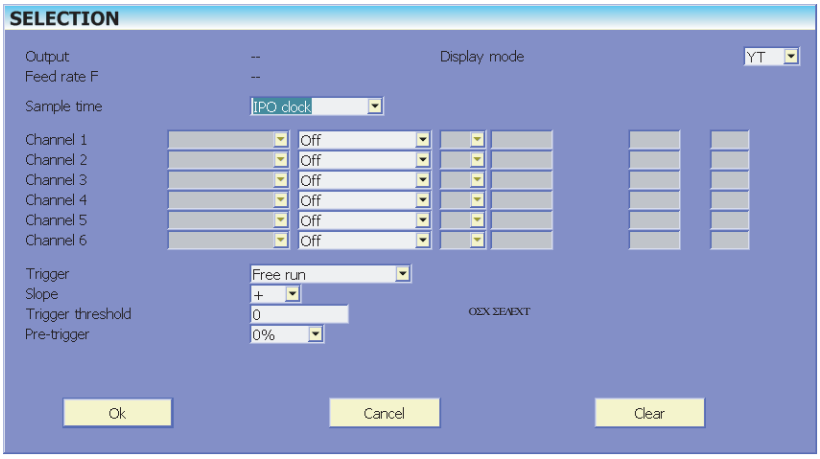


Setup for Analog Signals

Call the “Selection” dialog box:

- Press the **Select** (**F2**) soft key.

In the dialog box, set:



- **Type of display:** Set the time interval for recording the signals.
  - YT: Chronological depiction of the channels
  - YX: Graph of two channels
- **Sampling time:** Set the time interval for recording the signals.
  - CC clock: Time interval = 0.6 ms
  - IPO clock: Time interval = IPO clock (from MP_System/CfgCycleTimes/ipoCycle)
  - PLC clock: Time interval = PLC clock (results from MP_System/CfgCycleTimes/plcCount * Ipo-Takt)

3000 grid points (events) are stored. The time grid determines the duration of recording. Examples:

- 0.6 ms x 3000 = 1.8 s
- 3 ms x 3000 = 9 s
- 21 ms x 3000 = 63 s

Channel 1 to channel 6

Specify the signals to be recorded:

- Assign the channels of the analog signals to be recorded to the respective axes.
- Specify the operand type (B,W,D,I,O,T,C) and the address for the recording of PLC operands
- Use the **SAVED** setting to “freeze” the signal last recorded for this channel. This means that the recorded values remain available on the display. For example, you can use them to record a reference curve for use in future measurements.

Trigger conditions:
Specify the trigger conditions in the following input fields:

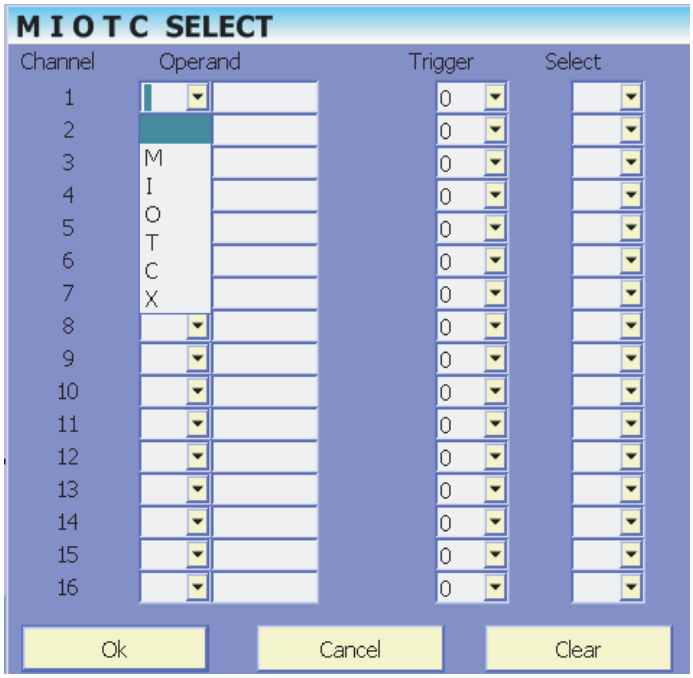
- **Trigger:** Set the trigger condition.
  - **Single shot:** After pressing the soft key, the next 3000 events are stored.
  - **Free run:** The recording is started and ended by soft key. If you press the STOP soft key, the last 3000 events (at most) are stored.
  - **Channel 1** to **6:** Recording begins when the trigger condition of the selected channel is fulfilled.
  - **Channel 1 + L** to **channel 6 + L:** Recording begins when the trigger condition of the channel selected here as well as the trigger conditions of the digital signals (trigger condition “logic”) are fulfilled. The trigger conditions are AND-gated.
  - **Logic:** Recording begins when the trigger condition of the digital signals is fulfilled (trigger condition “logic”).
- **Edge:** Set when triggering is to occur:
  - **+**: Trigger at rising edge
  - **–**: Trigger at falling edge
- **Trigger threshold:** Enter the trigger threshold (you will find the appropriate units in the “**Overview of Signals**”).
- **Pre-Trigger:** Select a value from the selection box.
  Recording begins at a time preceding the trigger time point by the value entered here

The **Output** and **Feed rate F** fields are reserved.

Setup for Digital Signals

Call the “M I O T C SELECT” dialog box:

- Press the **M/I/O/T/C** (**F3**) soft key.



Set:

- **Operand:** Type and number of the PLC operand
  - M: Marker
  - I: Input
  - O: Output
  - T: Timer
  - C: Counter
- Trigger:
  - X: No trigger
  - 0: Trigger at 0-level
  - 1: Trigger at 1-level

The trigger is only taken into consideration if **Selection=X** is set.

Selection:

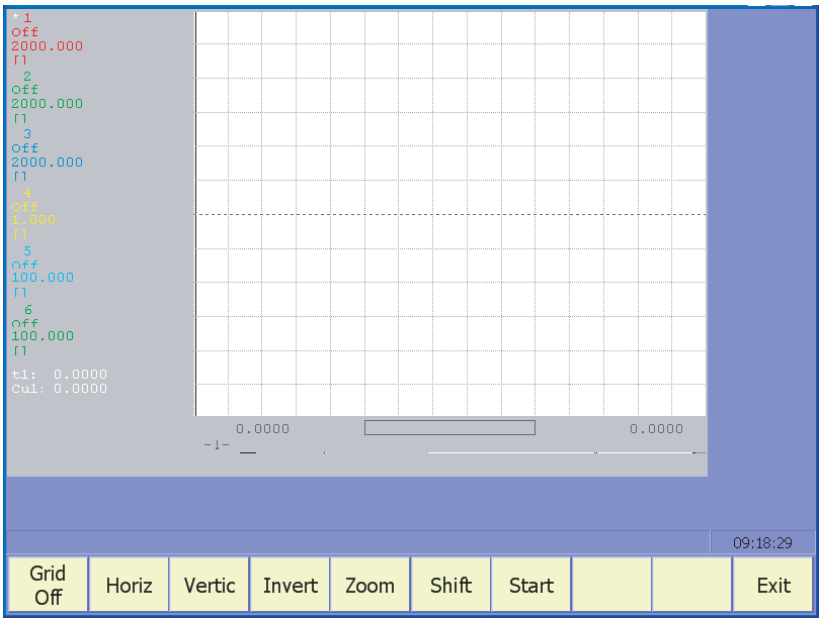
- X: Signal is displayed and considered as trigger
- Empty field: Signal is not displayed and not considered as trigger

Note:
- You define the general trigger conditions (“Trigger” input field) and the pre-trigger in the setup for analog signals.
- The trigger condition “logic” is fulfilled when all triggers set in “M I O T C selection” are fulfilled (AND-gating).

Record Signals

Change to the Recording operating mode:

- Press the **OSC** (**F1**) soft key.



For every channel, the type and resolution of the analog signal are shown in the left status field. The operand type and address are listed for digital signals.

The following topics are described:

- **Starting and Stopping the Recording**
- **Trigger Conditions**
- **Hide/Show Gridlines**

Starting and Stopping the Recording

To start recording:

- Press the **Start** (**F7**) soft key.

To stop recording:

- Press the **STOP** (**F6**) soft key.

Stop display:

- Press the **DISPLAY STOP** (**F2**) soft key.

Trigger Conditions

Trigger and pre-trigger conditions:

- Trigger=Single shot: 3000 events beginning from the start are recorded.
- Trigger=Free run: At most the last 3000 events before the STOP soft key is pressed are recorded.
- Trigger condition defined: The time when recording ends depends on the setting of the pre-trigger.
  - Pre-trigger=0%: 3000 events beginning from the fulfilled trigger condition are recorded.
  - Pre-trigger=25% (or 50%, or 75%): 75% (or 50% or 25%) of the 3000 events beginning from the fulfilled trigger condition are recorded.
  - Pre-rrigger=100%: Recording is stopped. The last 3000 events before the fulfilled trigger condition are recorded.

Note: If the trigger condition is fulfilled **before** the corresponding number of events have been stored when the pre-trigger is set to 25, 50, 75 or 100%, then correspondingly fewer events are recorded.

During recording, the selected signals are continuously displayed. You can freeze the display of the signals with the DISPLAY STOP soft key. This does not affect the recording of the signals.

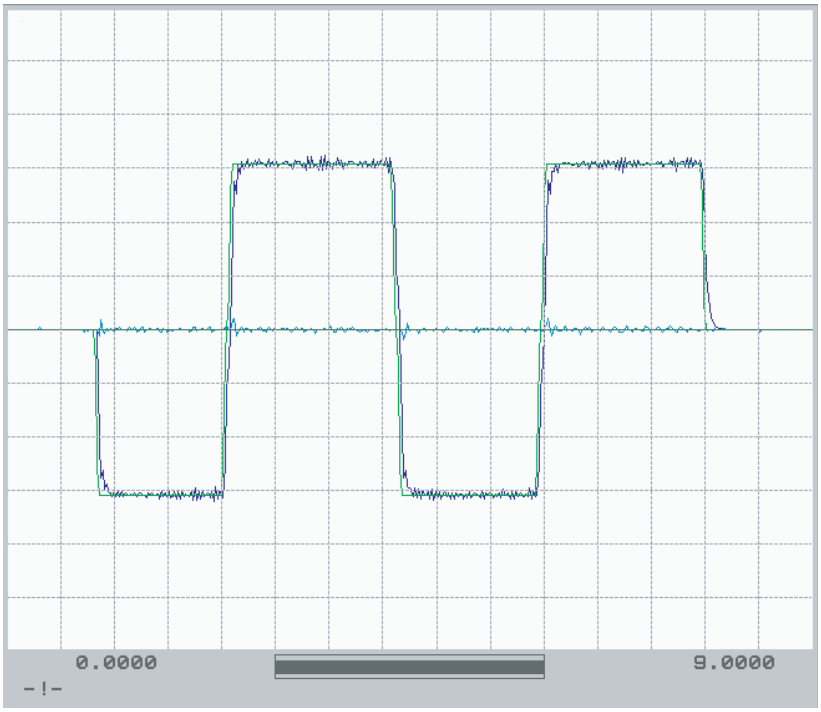
The recorded data remain stored until you start recording again or activate another graphic function.

A fulfilled trigger condition is indicated with a “T” in the status field at right below the display area.

Hide/Show Gridlines

- Press the **Grid Off** (**F1**) soft key.

Analyze Recording



The following topics are described:

- **Recording Completed**
- **Changing the Display**
- **Analyze an Individual Analog Signal**
- **Influence the Signal Display**
- **Second Cursor**

Recording Completed

After recording has completed, the oscilloscope shows the memory contents. The information in the status field below the display area refers to the displayed events. It has the following meanings (see figure below):

Left number: Time the “leftmost” event was recorded

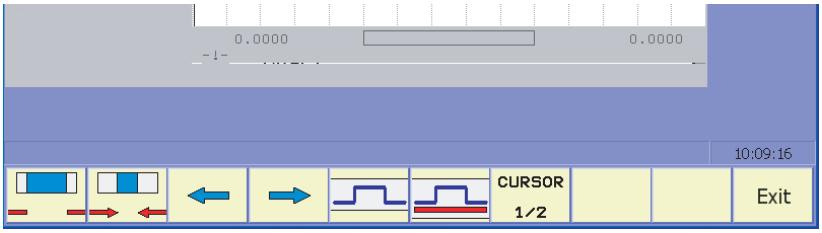
Right number: Time the “rightmost” event was recorded

The bar symbolizes the displayed range relative to the memory content.

The fulfillment of the trigger condition is designated as “t0” (t=0). Events that were recorded before the trigger condition was fulfilled are given a negative time. If no trigger condition was defined, the beginning of the recording is designated as “t0.

Changing the Display

Press **Horiz** (**F1**), the soft keys are:



The following soft keys influence the entire display range (all signals):

- Shft the display range to the left (**F3**)

- Shift the display range to the right (**F4**)

- Decrease the horizontal resolution (**F2**)

- Increase the horizontal resolution (**F1**)

Analyze an Individual Analog Signal

- Use the **ARROW UP** and **ARROW DOWN** keys to select the channel to be analyzed.

The selected channel is indicated with an "*." At the same time, the cursor is activated and placed on the selected channel.

Shown in the status field (bottom left) are (see "**Cursor Information**" figure):

- Code "t1: .....": Cursor position in [s], referenced to t0
- Code "Cu1: .....": Signal size at the cursor position

Shifting the Cursor

- Shift the cursor with the **ARROW LEFT** and **ARROW RIGHT** keys

Influence the Signal Display

Press **Vertic** (**F3**), the soft keys are:

The following soft keys influence the display display:

- Shift the signal downward (**F4**)

- Shift the signal upward (**F3**)

- Decrease the vertical resolution (**F1**)

- Increase the vertical resolution (**F2**)

- Optimum vertical resolution (**F5**). The signal is centered on the zero line and always remains in the display area.

- Undoes vertical shifts (**F6**)

- Invert the signal (multiply by –1) [**OSC** (**F1**) soft key screen, **Invert** (**F4**)]

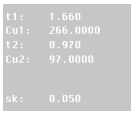
Second Cursor

- Activate/deactivate second cursor

The information for the second cursor is **relative to the first cursor.** It is shown in the status field (see "**Cursor Information**" figure):

- Code "t2: .....": Cursor position in [s], referenced to the first cursor
- Code "Cu2: .....": Signal referenced to the first cursor position

Cursor Information



Saving and Loading Recordings

You can load the recorded signals and all settings in one file. The file must have the extension SCO (oscilloscope trace file).

You can load and evaluate saved SCO files in the oscilloscope. ANILAM also makes the PC program **TNCscopeNT** available for evaluating SCO files.

The following topic is described:

- **Saving and Loading Oscilloscope Recordings**

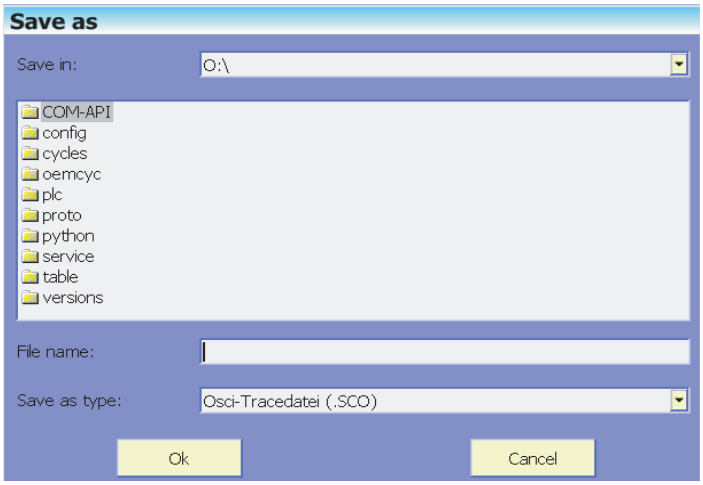
Saving and Loading Oscilloscope Recordings

The following topic is described:

- **Save the Oscilloscope Trace File**
- **Load the Oscilloscope Trace File**

Save the Oscilloscope Trace File

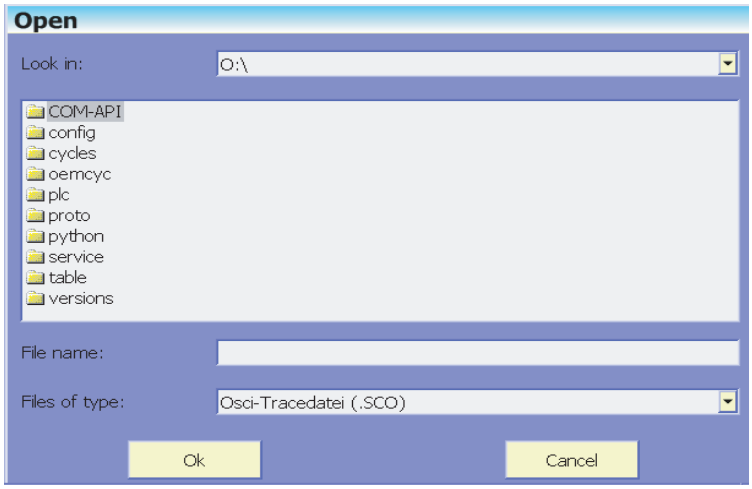
On the **OSC** (**F1**) soft key screen, press the **Save** (**F7**) soft key:



- Enter the path in the “Save as” dialog box
- Press the **Ok** soft key or button

Load the Oscilloscope Trace File

On the **OSC** (**F1**) soft key screen, press the **Load** (**F6**) soft key:



- Enter the path in the “Save as” dialog box
- Press the **Ok** soft key or button

Configure the Colors of the Oscilloscope Display

Settings in the configuration editor:	
System DisplaySettings CfgOsciColor **background** **channel1** **channel2** **channel3** **channel4** **channel5** **channel6** **logicTrace** **select** **grid** **cursorText**	

- In the parameter object CfgOsciColor, define the colors for the oscilloscope.

MP_background

Background color

Format: Drop-down selection menu

Selection:

[black]
[blue]
[light_grey]
[red]
[dark_grey]
[light_green]
[really_light_grey]
[really_dark_grey]
[light_violet]
[light_blue]
[light_red]
[medium_grey]
[yellow]
[white]

Default: black

The colors defined in MP_channel1–6 are used for display of the status information of this channel and the path of the curve. As soon as a channel is selected, the color defined in MP_select is switched to.

MP_channel1

Color for channel 1

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_green

MP_channel2

Color for channel 2

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_violet

MP_channel3

Color for channel 3

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_blue

MP_channel4

Color for channel 4

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_red

MP_channel5

Color for channel 5

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_blue

MP_channel6

Color for channel 6

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_red

The color defined in MP_logicTrace is used for the display of the digital signals.

MP_logicTrace

Color for logic-trace channels

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: yellow

MP_select

Color for selected channel

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: white

MP_grid

Color for gridlines

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: light_grey

MP_cursorText

Color for gridlines

Format: Drop-down selection menu

Selection:

[ ]

See MP_background

Default: dark_grey

Section 6 - Machine Integration

The following topics are described in this section:

- **Display and Operation**
- **Switching the Control On/Off**
- **Control Operation in the Operating Mode Group**
- **Control Operation in the Machining Group**
- **M Functions (M Strobe)**
- **S Functions (S Strobe)**
- **T Functions (T Strobe)**
- **Alias Functions (Alias Strobe)**
- **Error Messages and Log Files**
- **Keystroke Simulation**
- **Electronic Handwheel**
- **Override**
- **PLC inputs/Outputs**
- **Incremental Jog Positioning**
- **Operating Times and System Times**
- **Tool Changer**
- **Commissioning**
- **Diagnosis with the On-Line Monitor (OLM)**

Display and Operation

The display screen of the control is divided into separate windows. The user can select the operating functions by soft key. (Refer to the *6000i CNC User's Manual,* P/N 627785-2X.)

The following topics are described:

- **Position and Status Display**
- **Unit of Measurement for Display and Operations**
- **Decimal Separator**

Position and Status Display

The status display shows the status of the control.

With a soft key you can activate an additional status display in the graphic window instead of the graphic.

This includes:

- Axis positions
- Tools
- Feed rate
- M functions

The following topics are described:

- **Free Rotation**
- **Free Rotation with Module 9223**

Free Rotation

Free rotation means that the rotary axis rotates as often as required (with a display range of 0 to 360°) without being affected by software limit switches. Use Module 9223 to define the Free Rotation function.

The maximum feed rate is 300 000 °/min. The feed rate is not shown in the status window.

Free Rotation with Module 9223

If a program has been started, the module may be called only in conjunction with an M/S/T/Q strobe.

Module 9223 Free rotation

The feed-rate override (**NN_ChnFeedOverrideInput**) remains in effect.

Call:

PS	B/W/D/K	<>Axis>
		0 –max
PS	B/W/D/K	<>Feed rate [°/min]>
PS	B/W/D/K	<>Mode>
		0: Stop
		+1: Start in positive direction
		–1: Start in negative direction
CM	9223	
PL	B/W/D	<>Error code>
		0: No error: Positioning is started/stopped
		1: No rotary axis transferred
		2: Impermissible feed rate
		3: Axis has not traversed the reference mark
		4: No M/S/T/Q strobe during running program
		5: Programmed axis not in closed loop

Unit of Measurement for Display and Operation

Settings in the configuration editor:	
System DisplaySettings CfgUnitOfMeasure **unitOfMeasure**	

MP_unitOfMeasure is evaluated by the following functions or modes of operation:

- Machine display
- Entries in the **Manual Operation**, **El. Handwheel**, and **Positioning with MDI** operating modes
- Entries in the configuration editor

NC programs have a specific code for the unit of measurement.

- In **MP_unitOfMeasure,** you define whether the display or operation is in metric or inch mode.

Input or display	**metric**	**inch**
Coordinates, linear dimensions, compensation values, etc.	mm	inch
Feed rate (fee rate per minute, feed rate per revolution)	mm/min; mm/rev	inch/min; inch/rev
Cutting speed	mm/min	ft/min

Number of decimal places	**metric**	**inch**
Coordinates, linear dimensions, etc.	3	4
Compensation values	3	5

MP_unitOfMeasure

Unit of measure for display and user interface

Format: Pull-down selection menu

Selection:

[metric]
Metric system
[inch]
Inches

Default: metric

Decimal Separator

Settings in the configuration editor:	
System DisplaySettings CfgDisplayData **decimalCharacter**	

At present, only the decimal point may be used as a decimal separator.

MP_decimalCharacter

Decimal separator for position display

Format: Pull-down selection menu

Selection:

[.]

At present, only the decimal point is allowed.

Default: . (Decimal point)

Switching the Control On/Off

The following topics are described:

- **Powering Up the Control**
- **Shutting Down the Control**
- **Conversational Language**

Powering Up the Control

Settings in the configuration editor:	
System DisplaySettings CfgStartupData **powerInterruptMsg**	

Message for Power Interruption

In **MP_powerInterruptMsg,** you define the behavior during control start-up. You can choose between the following start-up procedures.

- The message **Power interrupted** is displayed during control start-up. Run-up is only continued after the message has been acknowledged.
- Run-up is not interrupted. The message **Power interrupted** does not display.

MP_powerInterruptMsg

Acknowledge the **Power interrupted** message

Format: Pull-down selection menu

Selection:

[True]

Run-up is only continued after the message has been acknowledged.

Pull-down selection menu

The **Power interrupted** message does not display.

Default: False

PLC operand	**Type**
NN_GenCycleAfterPowerOn **1. PLC scan after power on**	M
NN_GenCycleAfterPlcStop **1. PLC scan after PLC interruption**	M
NN_GenCycleAfterReConfig **1. PLC scan after change of configuration data**	M
NN_GenNcInitialized **Control is being initialized (after start-up cycles)**	M

Shutting Down the Control

Settings in the configuration editor:	
System DisplaySettings CfgShutDown **shutdownOnConfig** **shutdownOnError** **shutdownOnUser** **shutdownOnOem** **maxTermTime** **powerOffPort** **powerOffDelay**	

The control must be shut down before it can be switched off. The ready signal of the servo drives is removed and the memory card (or hard disk) is put into sleep mode. The shutdown can be delayed.

There are various causes for shutdown. In the parameter object **CfgShutDown,** you define the behavior when the control is shut down, depending on the cause. A distinction is made between the following causes:

- **shutdownOnConfig:** Configuration data that caused a reset were changed.
- **shutdownOnError:** A severe error occurred.
- **shutdownOnUser:** The machine operator terminates control operation (by soft key).
- **shutdownOnOem:** The PLC program terminates control operation with Module 9279 or 9189.

After shutting down the control, you can set a PLC output (example: to switch off the machine). The following prerequisites are to be met:

- Use **PowerOff** to define the shutdown, or shut down the control with Module 9279 – mode 2.
- In **MP_powerOffPort,** a PLC output is defined (I0 to I31).

The setting of the PLC output can be delayed.

Defining the shutdown behavior:

- In the parameter object **CfgShutDown,** you define the behavior when the control shuts down.
- In **MP_maxTermTime,** you enter the time by which the shutdown of the control is delayed.
- In **MP_powerOffPort,** you define whether a PLC output is to be set:
- Set PLC output: Define the PLC output to be set.
- Do not set PLC output: No entry in **MP_powerOffPort.**
- In **MP_powerOffDelay,** enter the delay time until the PLC output is set.

MP_shutdownOnConfig

Behavior when RESET configuration data are changed

Format: Pull-down selection menu

Selection:

[Restart]

The control is shut down and then restarted.

[Terminate]

The control is shut down, but the operating system remains active.

[Shutdown]

The control and the operating system are shut down.

[PowerOff]

The control and the operating system are shut down. If a PLC output is entered in MP_powerOffPort, it will be set after shutdown.

Default: Restart

MP_shutdownOnError

Behavior when RESET errors are acknowledged

Format: Pull-down selection menu

Selection:

[Restart]

The control is shut down and then restarted.

[Terminate]

The control is shut down, but the operating system remains active.

[Shutdown]

The control and the operating system are shut down.

[PowerOff]

The control and the operating system are shut down. If a PLC output is entered in MP_powerOffPort, it will be set after shutdown.

Default: Restart

MP_shutdownOnUser

Behavior during switch-off by soft key

Format: Pull-down selection menu

Selection:

[Restart]

The control is shut down and then restarted.

[Terminate]

The control is shut down, but the operating system remains active.

[Shutdown]

The control and the operating system are shut down.

[PowerOff]

The control and the operating system are shut down. If a PLC output is entered in MP_powerOffPort, it will be set after shutdown.

Default: Terminate

MP_shutdownOnOem

Behavior when PLC module 9279 is called

Format: Pull-down selection menu

Selection:

[Restart]

The control is shut down and then restarted.

[Terminate]

The control is shut down, but the operating system remains active.

[Shutdown]

The control and the operating system are shut down.

[PowerOff]

The control and the operating system are shut down. If a PLC output is entered in MP_powerOffPort, it will be set after shutdown.

Default: Terminate

After shutdown has been initiated, the control waits for the time defined in **MP_maxTermTime** before starting the shutdown.

MP_maxTermTime

Delay time until control is shut down

Format: Numerical value

Input: 0 to 1000 [s]

Default: 0

The entry in **MP_powerOffPort** has the following meaning:

- Entry 0–31: The corresponding PLC output is set if the requirements described above are fulfilled (shutdown with **PowerOff** or shutdown with Module 9279 – mode 2).
- No entry: No PLC output is set.

MP_powerOffPort
PLC output to be set after shutdown
Format: Numerical value
Input: 0–31: Corresponds to PLC outputs I0 to I31
No entry: Do not set PLC output.

After shutdown, the control waits for the time defined in **MP_powerOffDelay** before setting the PLC output.

MP_powerOffDelay
Delay time until PLC output is set
Format: Numerical value
Input: 0 to 1000 [s]
Default: 0

The following topics are described:

- **Module 9189 Shut down the control**
- **Module 9279 Shut down control (configurable)**

Module 9189 Shut down the control

Module 9189 shuts down the control. The PLC is not executable after shut down. The message windows, which is displayed during shutdown via soft key, do not display.

Call:

CM 9189

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Control was shut down
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	20	Module was not called in a spawn job or submit job

Module 9279 Shut down control (configurable)

The module terminates control operation. The behavior during shutdown of the control is defined in a transfer parameter.

Constraints:

- After the module has been called, the file system updates the data on the memory card (or hard disk) and closes all files.
- The module cannot be called in the cyclic PLC program since all accesses to the hard disk by the PLC must be implemented in a submit or spawn job.
- The PLC is not executable after shut down.
- Module 1 triggers a control reset control immediately after shutdown.
- The module call does not result in any outputs on the screen.

Call:

PS B/W/D/K <>Mode>
0: Shut down the control
1: Shut down and restart the control
2: Shut down the control; then set the PLC output from MP_powerOffPort (if defined).

CM 9279

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Control reset was carried out
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	20	Module was not called in a spawn job or submit job

Conversational Language

Settings in the configuration editor:	
System DisplaySettings CfgDisplayLanguage **ncLanguage** **plcDialogLanguage** **plcErrorLanguage** **helpLanguage** (reserved)	

The control distinguishes between conversational languages for the following areas:

- NC operation
- PLC operation
- PLC error messages
- On-line help

The **path** for the dialog text files is permanently defined. The language abbreviation is at the end of the path. You define the language abbreviation in the parameters of the **CfgDisplayLanguage** object.

- In the parameter object **CfgDisplayLanguage,** you define the languages you want to use.

The directories are:

%OEM%\PLC\LANGUAGE\ cs(Czech)
da(Danish)
nl(Dutch)
en(English)
fi(Finnish)
fr(French)
de(German)
it(Italian)
pl(Polish)
pt(Portuguese)
es(Spanish)
sv(Swedish)
hu(Hungarian)
ru(Russian)

If the dialog text files for the selected language are not on the PLC partition, the error message **LANGUAGE LOAD ERROR** is displayed. The control will then try to open the dialog text file in the directory “...\en” (English).

The **file names** of the dialog text file are the same for all languages. The file names are usually defined in parameters.

- Store the dialog texts you created under the same file names in permanently defined directories.

In **MP_ncLanguage,** you define the end of the path indicating the NC conversational language (language abbreviation).

MP_ncLanguage

NC conversational language

Format: Pull-down selection menu

Selection:

[ENGLISH]
[GERMAN]
[CZECH]
[FRENCH]
[ITALIAN]
[SPANISH]
[PORTUGUESE]
[SWEDISH]
[DANISH]
[FINNISH]
[DUTCH]
[POLISH]
[HUNGARIAN]
[JAPANESE]
[RUSSIAN]
[CHINESE]
[CHINESE_TRAD]
[SLOVENIAN]

Default: ENGLISH

In **MP_plcDialogLanguage**you define the end of the path indicating the PLC conversational language (language abbreviation).

MP_plcDialogLanguage

PLC conversational language

Format: Pull-down selection menu

Input: See MP_ncLanguage

In **MP_plcErrorLanguage,** you define the end of the path indicating the language for the PLC error messages (language abbreviation).

MP_plcErrorLanguage

Language for PLC error messages

Format: Pull-down selection menu

Input: See MP_ncLanguage

MP_helpLanguage is used to define the end of the path of the help texts (language abbreviation).

MP_helpLanguage

Language for on-line help

Format: Pull-down selection menu

Input: See MP_ncLanguage

Control Operation in the Operating Mode Group

The following topic is described:

- **Modes of Operation**

Modes of Operation

Note: All machining channels of an operating mode group have the same operating mode.

In the following PLC operands, the NC informs the PLC of the current operating mode of an operating mode group:

PLC operand	Type
NN_OmgManual **Manual operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgHandwheel **Electronic handwheel operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgMdi **Positioning with manual data input operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgProgramSingle **Program run, single block operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgProgramRun **Program Run Full Sequence operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgReference **Reference operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgDiagnosis **Diagnostic operating mode** 0: Operating mode not active 1: Operating mode active	M
NN_OmgJogIncrement **Incremental jog positioning mode of operation** 0: Operating mode not active 1: Operating mode active	M

The following topics are described:

- **Start/Stop of the Machining Channels**
- **Reaction to Errors**

Start/Stop of the Machining Channels

In the following PLC operands, the PLC informs the NC of the start or stop status:

PLC operand	Type
PP_OmgNcStart **NC start for all machining channels of this operating mode group** 0: NC start not active 1: NC start active	M
PP_OmgNCStop **NC stop for all machining channels of this operating mode group** 0: NC stop not active 1: NC stop active	M

Reaction to Errors

In **MP_errorBehavior,** you define the behavior of an operating mode if an error occurs.

MP_errorBehavior

Behavior of the operating mode group in error recovery

Format: Pull-down selection menu

Selection:

[Stop]

All other channels of the OMG are stopped.

[Cont]

All other channels continue operating if possible.

Default: Stop

Control Operation in the Machining Channel

The following topics are described:

- **Channel-Specific Settings**
- **NC Program Run**
- **Error Status**
- **Assignments in Manual Modes of Operation**

Channel-Specific Settings

The following topics are described:

- **Arc End-Point Tolerance**
- **Retract Tool at NC Stop**

Arc End-Point Tolerance

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgGeoTolerance **circleDeviation**	

The control uses the entered NC data to calculate the deviation of the arc radius between the beginning and end of the arc:

- Enter a tolerance value in MP_circleDeviation. If the entered tolerance is exceeded, the error message **CIRCLE END POS. INCORRECT** is displayed.

MP_circleDeviation
Permissible deviation from radius
Format: Numerical value
Input: 0.00001 to 0.01600 [mm]
Default: 0.005

Retract Tool at NC Stop

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgLiftOff **on** **distance**	

In the parameter object **CfgLiftOff,** you define the reaction of the tool to NC stop:

- In **MP_on,** you define whether the tool is to lift off at NC stop.
- In **MP_distance,** you define the retraction height.

MP_on

Switching on/off lift-off movements during NC stop

Format: Pull-down selection menu

Selection:

[On]
Lift-off movements active

[Off]
Lift-off movements not active

Default: Off

MP_distance

Maximum retraction height for NC stop

Format: Numerical value

Input: 0.000000000 to 2.000000000 [mm]
Default: 0.0 [mm]

NC Program Run

Settings in the configuration editor:	
NCchannel ChannelSettings CfgChannelFile **geoIniProgram** **geoCycleEnd** **geoCancelCycle** System Paths CfgSystemCycle Key for OEM system cycle **path**	

The following topics are described:

- **Starting an NC Program**
- **Automatic NC Program Start**
- **Terminating the NC Program**
- **Interrupting an NC Program**
- **Moving the Axes During Program Interruption**
- **NC Program Cancellation**
- **Block Scan (Start Block Search)**
- **Finding the NC Program and Block Number**
- **Control in Operation**
- **M, S, or T Function in Parallel with Traverse Motion**

Starting an NC Program

The PLC executes an NC start with **PP_ChnNcStart**.

PLC operand	Type
PP_ChnNcStart **NC start or Cycle on** 0: NC start not active 1: NC start active	M

With **NN_ChnNcStartExternRequest.** the NC requests the PLC to initiate an NC start. The PLC then uses **PP_ChnNcStart** to activate the NC start.

PLC operand	Type
NN_ChnNcStartExternRequest **External request for NC start** 0: External NC start not requested 1: External NC start requested	M

Before running the actual NC program, the NC first starts the program defined in **MP_geoIniProgram**, and then the OEM program defined in **MP_Path**.

The NC program is executed immediately after the lead programs.

MP_geoIniProgram
Path and name of the lead program
Format: String
Input: Path and name of the lead program
No entry: No lead program is executed.

MP_path
Path and name of the OEM lead program
Format: String
Input: Path and name of the OEM lead program
No entry: No OEM lead program is executed.

Automatic NC Program Start

NC programs can be started by the control automatically at a date and time set by the user.

- The following requirements must be fulfilled to be able to enter an autostart:
- You have to enable the Autostart operation in **MP_autoStartEnabled.**

The PLC must enable Autostart operation with **PN_ChnAutostartEnable=1.**

Danger: Caution, danger to life!

The function must not be used for **open** machines **without** enclosed working space (including protective door)! It must be ensured that the machine operator will not be able to activate the “Automatic NC program start function” while the protective door is open. It is absolutely necessary that you take this into account in the PLC program of the machine. Disable the function for open machines through the PLC by setting the operand **PN_ChnAutostartEnable** permanently to the value 0. Be sure to link the operand with protective-door monitoring in your PLC program.

MP_autoStartEnabled

Activate the AUTOSTART soft key

Format: Pull-down selection menu

Selection:

[On]

Permit autostart operation

[Off]

Suppress autostart operation

PLC operand	Type
PN_ChnAutostartEnable **Enable the autostart function** 0: Autostart disabled 1: Autostart enabled	M

The NC informs the PLC of the current autostart status in two markers:

- **NN_ChnAutostart** indicates whether autostart is active.
- **NN_ChnAutostartTimeExpired** indicates whether the time programmed by the user has expired. After expiration of the time, the PLC an activates NC start with **PP_ChnNcStart**.

PLC operand	Type
NN_ChnAutostart **Autostart active** 0: Autostart not active 1: Autostart active	M
NN_ChnAutostartTimeExpired **Autostart: Time has expired, request for program start** 0: Autostart time not expired 1: Autostart time expired	M

Terminating the NC Program

In **NN_ChnProgEnd,** the NC informs the PLC that an NC stop was executed because the program end has been reached.

PLC operand	Type
NN_ChnProgEnd **NC program end is reached** A “program end” command was executed (END-PGM, M02, or M30). 0: NC program end not reached 1: NC program end reached	M

After the NC program has been run, the NC starts the program defined in **MP_geoCycleEnd.** The trailer program is executed immediately after the NC program.

MP_geoCycleEnd
Path/name of the trailer program for program end
Format: String
Input: Path and name of the trailing program
No entry: No trailing program is executed.

Interrupting an NC Program

The PLC or NC can stop execution of the NC program. After interruption, the NC program is continued.

During program interruption, the axes can be traversed manually.

PLC stops NC program run:

The PLC executes an NC stop with **PP_ChnNcStop.**

PLC operand	Type
PP_ChnNCStop **NC stop or Cycle off** 0: NC stop not active 1: NC stop active	M

NC stops NC Program Run:

The NC uses the following markers to inform the PLC of NC program interruption and the reason for the interruption:

- **NN_ChnStopExtern:** The program was interrupted because of an external request (e.g., Stop key).
- **NN_ChnProgStopped:** The program was interrupted because of a program stop (M0), the end of a block in Single block mode, etc.
- **NN_ChnProgStoppedAsync:** The program interruption was caused by an error, etc.

PLC operand	Type
NN_ChnNcStopExtern **NC stop or Cycle off** NC stop is executed by the NC. 0: NC stop not executed 1: NC stop was executed by the NC	M
NN_ChnProgStopped **NC program interruption** The NC reports an asynchronous program interruption, such as at the end of a block in Single Block mode, M0, etc. 0: No asynchronous NC program interruption 1: NC program interruption	M
NN_ChnProgStoppedAsync **Asynchronous NC program interruption** The NC reports an asynchronous program interruption, for example because of an error, etc. 0: No asynchronous NC program interruption 1: Asynchronous NC program interruption	M

Moving the Axes During Program Interruption

During program interruption, the NC distinguishes between “manual traverse of the axes” and “returning to the contour.” The NC indicates the status in the following markers:

PLC operand	Type
NN_ChnProgManTraverse **Manual traverse of axes is active (with lathes: inspection mode)** 0: Manual traverse not active 1: Manual traverse active	M
NN_ChnProgReturnContour **Return to the contour is active (after manual traverse or for block scan)** 0: Return to contour is not active 1: Return to contour is active	M

NC Program Cancellation

The NC uses **NN_ChnProgCancel** to inform the PLC of program cancellation.

PLC operand	Type
NN_ChnProgCancel **NC program cancellation** NC program cancellation due to an internal stop 0: No NC program cancellation 1: NC program cancellation	M

After the NC program has been canceled, the NC starts the program defined in MP_geoCancleCycle. The trailer program is executed immediately after NC program cancellation.

MP_geoCancelCycle

Path/name of the trailer program for program cancellation

Format: String

Input: Path and name of the trailing program

No entry: No trailing program is executed.

Block Scan (Start Block Search)

PLC operand	Type
NN_ChnBlockScan **Block scan (or start block search) is active** 0: Block scan not active 1: Block scan active	M
NN_ChnBlockScanStrobeTransfer **Restore status at block scan (M/S/T/Q transfer)** 0: Status not restored 1: Status restored	M

Finding the NC Program and Block Number

Module 9321 Find the current block number

Module 9321 finds the name of the NC program and the current block number. The program name and path are stored in the given string.

Call:

PS	B/W/D/K	<>String number>
CM	9321	
PL	B/W/D	<>Current block number>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	NC program and block number found
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid string number

Control in Operation

In the **Positioning with Manual Data Input, Program Run, Single Block**, and **Program Run, Full Sequence** operating modes, the NC uses **NN_ChnControlInOperation** to inform the PLC that the control is in operation. The status “control in operation” applies when the NC executes a program, an M function, or an axis movement.

PLC operand	Type
NN_ChnControlInOperation **Control in operation** 0: Control not in operation 1: Control in operation	M

M, S, or T Function in Parallel with Traverse Motion

The PLC can execute M, S, or T functions in parallel with the movement programmed in the same NC block.

Module 9404 Start movement when an NC strobe is present

The module starts the movement programmed in an NC block when a strobe that is effective at the beginning of the same NC block is still present.

Call:

PS	B/W/D/K	<>Channel number>
CM	9404	
PL	B/W/D	<>Error number>
		0: Successful
		1: Invalid channel number
		2: Execution of NC part program is not synchronized
		15: Module was called in a spawn job or submit job

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Movement successfully started
	1	Process not possible

Error Status

The NC informs the PLC of errors occurring in this machining channel. A distinction is made between the reactions to errors in the PLC operands (see "**PET Table (PLC Error Table)**").

PLC operand	Type
NN_ChnErrorWarning **Error or warning occurred** 0: No error or warning occurred 1: NC error or warning occurred	M
NN_ChnErrorFStop **Feed stop due to an error** 0: No feed stop triggered 1: Feed stop triggered	M
NN_ChnErrorNCStop **NC stop due to an error** 0: No NC stop triggered 1: NC stop triggered	M
NN_ChnErrorCancel **Program cancellation due to an error** 0: No program cancellation 1: Program cancellation triggered	M
NN_ChnErrorEmergencyStop **Emergency stop due to an error** 0: No emergency stop triggered 1: Emergency stop triggered	M
NN_ChnErrorReset **Reset due to an error** 0: No reset 1: Reset triggered	M

NN_ChnErrorReset is not used at present because the PLC program stops when a reset error occurs.

Assignments in Manual Modes of Operation

In the manual operating modes, the peripheral user devices, such as monitor or keyboard unit, are assigned to a machining channel and a spindle. The machining channel is specified in the PLC operands **NN_GenOmgManual** and **NN_GenChnManual**, the spindle in **NN_GenSpiManual.**

PLC operand	Type
NN_GenOmgManual **Selected operating mode group in manual operation**	D
NN_GenChnManual **Selected machining channel in manual operation**	D
NN_GenSpiManual **Selected spindle in manual operation**	D

M Functions (M Strobe)

In the control you can program miscellaneous functions, also known as M functions. The code of an M function is transferred to the PLC before or after execution of the NC block.

Certain M functions are reserved for the NC or have fixed meanings for the NC. The other M functions are freely available and are evaluated by the PLC.

M functions are channel-sensitive. M functions are configured in two steps:

- In the channel-sensitive parameter **MP_mStrobes,** you define a key name for each M function used in this machining channel. In this way, you assign the M functions to the machining channel.
- In the parameter object **System/PLC/CfgPlcMStrobe,** you configure the M function.

The following topics are described:

- **Assigning M Functions to the Machining Channels**
- **Configuration of M Functions**
- **Overview of M Functions of the 6000i**

Assigning M Functions to the Machining Channels

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgPlcStrobes **mStrobes** **unitOfMeasure**	

MP_mStrobes

M strobe descriptions of this machining channel

Format: Array [0–99]

Input: Key name of the M strobes used in this machining channel

In the marker defined in MP_unitOfMeasure, the PLC is informed of the unit of measure used in the NC program when M, S, T, or alias strobes are executed.

MP_unitOfMeasure

Symbolic name or number of the PLC marker for the unit of measure of the NC program

Format: String

Input: Name of the PLC marker, which informs the PLC of the unit of measure of the NC program to be run.

PLC marker=True: Inches

PLC marker=False: Metric mode

No input: The PLC is not informed of the unit of measure

Configuration of M Functions

Settings in the configuration editor:	
System PLC CfgPlcMStrobe Key for MStrobe **min** **max** **signal** **acknowledge** **code** **data** **revoke** **singular** **blockEnd** **blockSearch** **sync** **macro**	

The following topics are described:

- **Transfer of the M Function**
- **Execution at the Beginning or End of Block**
- **Transfer and Acknowledgment of M Function**
- **Calling an NC Macro with an M Function**
- **Executing NC Macros**

Transfer of the M Function

If there is more than one M function number with the same transfer parameters, the M function numbers can be defined in a parameter object.

- Multiple M codes in a parameter object:
  - **MP_min:** First M function number of the group
  - **MP_max:** Highest M function number of the group
- One M function number in a parameter object:
  - **MP_min:** M function number of the parameter object
  - **MP_max:** No entry

If **MP_code** is defined, the NC transmits the programmed M code in the PLC word defined in **MP_code** and other data of the M function in the PLC word defined in MP_data.

MP_min

Number of the first M function

Format: Numerical value

Input: First M function number described in the parameter object

MP_max

Number of the highest M function

Format: Numerical value

Input: Highest M function number described in the parameter object

No entry: The parameter object only applies for the M function defined in MP_min.

MP_code

Symbolic name or number of the PLC word for the M code

Format: String

Input: Name of the PLC word in which the M code is transmitted to the PLC

MP_data

Symbolic name or number of the PLC word for additional data

Format: String

Input: Name of the PLC word in which the additional data of the M function is transmitted to the PLC

In MP_singular, M codes are defined, which must be output in a separate strobe.

MP_singular

M function is output in a separate strobe

Format: Pull-down selection menu

Selection:

[True]

M function must be output in a separate strobe

[False]

M function can be combined with other M functions

Execution at the Beginning or End of Block

In MP_blockEnd, you define whether the M function is to be executed at the beginning or end of block.

MP_blockEnd

M function output at block end

Format: Pull-down selection menu

Selection:

[True]

M function is output at block end

[False]

M function is output at beginning of block

Default: False

Transfer and Acknowledgment of M Function

In the **Program Run, Full Sequence** and **Program Run, Single Block** operating modes, the next NC block is not run until the PLC has acknowledged execution of the M function:

There are two possibilities for transferring the M strobe to the PLC and for acknowledgment by the PLC:

- Transfer and acknowledgment with MP_signal
  - Transfer of the M strobe: The PLC marker defined in **MP_signal** is set.
  - Acknowledgment of the M strobe: The PLC marker defined in **MP_signal** is reset.
- Transfer with **MP_signal** and acknowledgment with **MP_acknowledge:**
  - Transfer of the M strobe: The PLC marker defined in **MP_signal** is set.
  - Acknowledgment of the M strobe: The PLC marker defined in **MP_acknowledge** is set.

ANILAM recommends that you only use **MP_signal** for transfer and acknowledgment.

If **MP_signal** and **MP_acknowledge** are not defined, the data of the M strobe are saved without synchronization with the PLC program. The output is immediately acknowledged.

MP_signal

Symbolic name or number of the PLC marker for M strobe

Format: String

Input: Name of the PLC marker which informs the PLC of the M function

No entry in MP_acknowledge: The M strobe is acknowledged by resetting the marker defined in MP_signal.

MP_acknowledge

Symbolic name or number of the PLC marker for acknowledgment

Format: String

Input: Name of the PLC marker in which the PLC acknowledges the M function

MP_revoke

M functions whose effects cancel each other

Format: Array

Input: M functions canceling the M function defined here

MP_blockSearch

M function output during block scan

Format: Pull-down selection menu

Selection:

[True]

M function is output during the block scan

[False]

M function is not output during the block scan

MP_sync defines the synchronization of the M function with NC program run.

MP_sync

Synchronization between M function and NC

Format: Pull-down selection menu

Selection:

[Sync_Exec]

The M function is synchronized with the program run. The output of movement by the interpolator is stopped; then the M function is executed.

[Sync_Calc]

The M function is synchronized with program interpretation. The interpretation of the NC program is stopped and the geometry chain is executed; then the M function is executed.

[Async]

M function is output without synchronization.

Calling an NC Macro with an M Function

An NC subprogram can be executed instead of transferring an M function to the PLC. The path and name of the NC subprogram are entered in **MP_macro**.

M functions that call an NC subprogram are not sent to the PLC.

MP_macro

NC subprogram call

Format: String

Input: Path and name of the NC subprogram

Note: A maximum of six NC programs can be nested (subprograms, cycles, macros).

Executing NC Macros

With **FN17: SYSWRITE ID420 NR0 IDX0 = 0,** all coordinate transformations (e.g., cycles 7, 8, 10, 11, 19) performed in the NC macro become globally effective. Without this block, they remain locally effective (only in the NC macro).

Overview of M Functions of the 6000i

Settings in the configuration editor:	
System DisplaySettings CfgStatusAndQPar **clearMode**	

The following topics are described:

- **Influencing the Execution of M Functions**
- **Overview of M Functions**
- **Program Stop with M6**
- **Status of M Functions**

Influencing the Execution of M Functions

In **MP_clearMode,** you define when status values, Q parameters, and tool data (DL, DR, DR2) are reset.

MP_clearMode

Reset status values, Q parameters, and tool data (DL, DR, DR2 from PGM)

Format: Pull-down selection menu

Selection:

[0]
Reset status, Q parameters, and tool data when a program is selected.

[1]
Erase the status display, Q parameters, and tool data if a program is selected and in the event of M02, M30, and END PGM.

[2]
Erase the status display and tool data when a program is selected.

[3]
Erase the status display and tool data when a program is selected and in the event of M02, M30, END PGM.

[4]
Erase the status display and Q parameters when a program is selected.

[5]
Erase the status display and Q parameters when a program is selected and in the event of M02, M30, END PGM.

[6]
Erase the status display when a program is selected.

[7]
Erase the status display when a program is selected and in the event of M02, M30, END PGM.

Default: 0

Overview of M Functions

M89 to M299 are reserved for the NC, and several M functions between M00 and M88 have fixed meanings for the NC. The other M functions are freely available.

Effective at A = beginning of block
E = end of block

M function	Meaning	Effectiveness
M00 or M0	Program STOP/Spindle STOP/Coolant OFF	E
M01 or M1	Optional program STOP	E
M02 or M2	End of Program	E
M03 or M3	Spindle ON FWD (clockwise)	A
M04 or M4	Spindle ON REV (counterclockwise)	A
M05 or M5	Spindle STOP	E
M06 or M6	Tool change/Program STOP/Spindle STOP	E
M08 or M8	Coolant ON	A
M09 or M9	Coolant OFF	E
M30	Jump to New Program	E
M98	Call SubProgram	A
M99	End SubProgram	E
M105	Dry Run, all axes	A
M106	Dry Run, NO Z-axis	A
M107	Dry Run, OFF (cancels M105 or M106)	A

Program Stop with M6

According to ISO 6983 Part 2, the M function M06 means “tool change.” After the program stop and the tool change, the NC program must be restarted through an NC start or by the PLC.

Status of M Functions

With Module 9060 you can ascertain the status of M functions M100 to M199.

With Module 9061 the status of the non-modal M functions M94, M142, M143, and M146 can be ascertained.

Note: The function of Modules 9060 and 9061 is restricted when used in conjunction with the new symbolic API and a multiple-channel control structure: M functions can be active on different channels. This channel dependency is not taken into account.

The following topics are described:

- **Module 9060 M function status**
- **Module 9061 Status of non-modal M functions**
- **Module 9088 Status display of M functions**

Module 9060 M function status

Module 9060 can determine whether an M function between M100 and M199 is active.

Call:

PS	B/W/D/K	<>Number of M function (100 to 199)>
CM	9060	
PL	B/W/D	<>Status>
		0: M function was not active
		1: M function was active

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status was found
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid number of M function

Module 9061 Status of non-modal M functions

With module 9061 the status of the non-modal M functions M94, M142, M143, and M146 can be interrogated. The status of the interrogated M function remains until the module is called again, even if the NC program has finished.

Call:

PS	B/W/D/K	<>Number of M function (90 to 199)>
CM	9061	
PL	B/W/D	<>Status>
		0: M function was not active
		1: M function was active

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status was found
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid number of M function

Module 9088 Status display of M functions

Use this module to display an M function in the status display for M function, or to delete it from the status display. The number of the M function and the mode setting must be given to the module.

A maximum of 50 M functions can be displayed.

First the modal M functions are displayed. Then the M functions to be handled by the PLC are appended to the list.

The module only becomes effective if the status has changed since the last call.

Call:

PS	B/W/D/K	<>Number of soft-key function>
PS	B/W/D/K	<>Mode> –1: Delete all M functions 0: Delete M function < number of M function> 1: Display M function <number of M function>
CM	9088	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	M function displayed or deleted
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid M-function number
	2	Invalid mode number

S Function (S Strobe)

The S function is channel-sensitive. S functions are configured in two steps:

- In the channel-sensitive parameter **MP_sStrobes,** you define a key name for the S function. In this way, you assign the S functions to a machining channel.
- In the parameter object **System/PLC/CfgPlcSStrobe,** you configure the S function.

Note: If the rotational speed is programmed within a G function, the S strobe will not be used. Then the rotational speed or constant cutting speed will be transmitted to the PLC in an M strobe.

The following topics are described:

- **Assigning S Functions to the Machining Channels**
- **Configuration of S Function**

Assigning S Functions to the Machining Channels

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgPlcStrobes **sStrobe** **unitOfMeasure**	

MP_sStrobe

S strobe description of this machining channel

Format: String [0–18]

Input: Key name of the S strobe used in this machining channel

In the marker defined in **MP_unitOfMeasure,** the PLC is informed of the unit of measure used in the NC program when M, S, T, or alias strobes are executed (see “**MP_unitOfMeasure**”).

Configuration of S Function

Settings in the configuration editor:	
System PLC CfgPlcSStrobe Key for SStrobe **signal** **acknowledge** **spindleSpeed** **badSpeed** **spindleMode** **gearCode** **revoke** **singular** **blockSearch** **sync**	

The following topics are described:

- **Transfer of the S Function**
- **Transfer and Acknowledgment of S Function**

Transfer of the S Function

The NC transmits the programmed spindle speed in the PLC word defined in **MP_spindleSpeed.** The PLC checks the programmed spindle speed. The result of this check is saved in the PLC marker defined in **MP_badSpeed.** If the marker is set, the spindle speed is outside the permissible range. If **MP_badSpeed** is not defined, the spindle speed is not checked.

MP_spindleSpeed

Symbolic name or number of the PLC word for transferring the spindle speed

Format: String

Input: Name of the PLC word in which the spindle speed is transmitted to the PLC

MP_badSpeed

Symbolic name or number of the PLC marker for impermissible spindle speeds

Format: String

Input: Name of the PLC marker which is set if the spindle speed is outside the permissible range.

No entry: The spindle speed is not monitored.

MP_spindleMode is reserved for the transfer of different spindle speed modes (example: constant spindle speed or constant cutting speed).

MP_spindleMode

Symbolic name or number of the PLC word for spindle speed modes

Format: String

Input: Name of the PLC word in which the spindle speed mode is transmitted to the PLC
No entry: No spindle speed mode is transmitted

MP_gearCode is reserved for transferring the gear range.

MP_gearCode

Symbolic name or number of the PLC marker for the gear range

Format: String

Input: Name of the PLC word in which the gear range is transmitted to the PLC
No entry: No gear range is transmitted

MP_singular specifies whether the S strobe must be output in a separate strobe.

MP_singular

S function is output in a separate strobe

Format: Pull-down selection menu

Selection:

[True]
S function must be output in a separate strobe

[False]
S function can be combined with other functions

Transfer and Acknowledgment of S Function

There are two possibilities for transferring the S strobe to the PLC and for acknowledgment by the PLC:

- Transfer and acknowledgment with **MP_signal**
  - Transfer of the S strobe: The PLC marker defined in **MP_signal** is set.
  - Acknowledgment of the S strobe: The PLC marker defined in **MP_signal** is reset.
- Transfer with **MP_signal** and acknowledgment with **MP_acknowledge:**
  - Transfer of the S strobe: The PLC marker defined in **MP_signal** is set.
  - Acknowledgment of the S strobe: The PLC marker defined in **MP_acknowledge** is set.

ANILAM recommends that you only use **MP_signal** for transfer and acknowledgment.

If **MP_signal** and **MP_acknowledge** are not defined, the data of the S strobe are saved without synchronization with the PLC program. The output is immediately acknowledged.

MP_signal

Symbolic name or number of the PLC marker for the S strobe

Format: String

Input: Name of the PLC marker which informs the PLC of the S function
No entry in MP_acknowledge: The S strobe is acknowledged by resetting the marker defined in MP_signal.

MP_acknowledge

Symbolic name or number of the PLC marker for acknowledgment

Format: String

Input: Name of the PLC marker in which the PLC acknowledges the S function
No entry: The S strobe is acknowledged by resetting the marker defined in MP_signal.

MP_revoke

(Optional)
Functions canceling the S function
Format: Array

Input: Functions canceling the S function defined here

MP_blockSearch

S function output during block scan

Format: Pull-down selection menu

Selection:

[True]
S function is output during the block scan

[False]
S function is not output during the block scan

T Functions (T Strobe)

T functions are channel-sensitive. T functions are configured in two steps:

- In the channel-sensitive parameter **MP_tStrobes,** you define a key name for the T functions. In this way, you assign the T functions to the machining channel.
- In the parameter object **System/PLC/CfgPlcTStrobe,** you configure the T function.

The following topics are described:

- **Assigning T Functions to the Machining Channels**
- **Configuration of T Functions**

Assigning T Functions to the Machining Channels

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgPlcStrobes **tStrobes** **unitOfMeasure**	

MP_tStrobes

T strobe description of this machining channel

Format: String [0–18]

Input: Key name of the T strobes used in this machining channel

In the marker defined in **MP_unitOfMeasure,** the PLC is informed of the unit of measure used in the NC program when M, S, T, or alias strobes are executed (see "**MP_unitOfMeasure**").

Configuration of T Functions

Settings in the configuration editor:	
System PLC CfgPlcSStrobe Key for TStrobe **type** **signal** **acknowledge** **toolNumber** **toolIndex** **toolMagazine** **pocketNumber** **revoke** **singular** **blockSearch** **sync**	

The following topics are described:

- **Transfer of the T Function**
- **Transfer and Acknowledgment of T Function**

Transfer of the T Function

MP_type specifies the type of tool call.

The NC transfers the other data for the tool call in the PLC words defined in the following parameters:

- **MP_toolNumber:** Tool number
- **MP_toolIndex:** Tool index
- **MP_toolMagazine:** Magazine number
- **MP_pocketNumber:** Pocket number

MP_type

Type of T function

Format: Pull-down selection menu

Selection:

[T0]
Remove tool from spindle

[T1]
Insert tool in spindle

[T2]
Prepare the next tool change

MP_toolNumber

Symbolic name or number of the PLC word for transferring the tool number

Format: String

Input: Name of the PLC word in which the tool number is transmitted to the PLC

MP_toolIndex

Symbolic name or number of the PLC word for transferring the tool index

Format: String

Input: Name of the PLC word in which the tool index is transmitted to the PLC

MP_toolMagazine

Symbolic name or number of the PLC word for transferring the magazine number of the tool

Format: String

Input: Name of the PLC word in which the magazine number of the tool is transmitted to the PLC

MP_pocketNumber

Symbolic name or number of the PLC word for transferring the pocket number of the tool

Format: String

Input: Name of the PLC word in which the pocket number of the tool is transmitted to the PLC

In MP_singular, T codes are defined, which must be output in a separate strobe.

MP_singular

T function is output in a separate strobe

Format: Pull-down selection menu

Selection:

[True]

T function must be output in a separate strobe

[False]

T function can be combined with other functions

Transfer and Acknowledgment of T Function

There are two possibilities for transferring the T strobe to the PLC and for acknowledgment by the PLC:

- Transfer and acknowledgment with **MP_signal**
  - Transfer of the T strobe: The PLC marker defined in **MP_signal** is set.
  - Acknowledgment of the T strobe: The PLC marker defined in **MP_signal** is reset.
- Transfer with **MP_signal** and acknowledgment with **MP_acknowledge:**
  - Transfer of the T strobe: The PLC marker defined in **MP_signal** is set.
  - Acknowledgment of the T strobe: The PLC marker defined in **MP_acknowledge** is set.

ANILAM recommends that you only use **MP_signal** for transfer and acknowledgment.

If **MP_signal** and **MP_acknowledge** are not defined, the data of the T strobe are saved without synchronization with the PLC program. The output is immediately acknowledged.

MP_signal

Symbolic name or number of the PLC marker for the T strobe

Format: String

Input: Name of the PLC marker which informs the PLC of the T function
No entry in MP_acknowledge: The T strobe is acknowledged by resetting the marker defined in MP_signal.

MP_acknowledge

Symbolic name or number of the PLC marker for acknowledgment

Format: String

Input: Name of the PLC marker in which the PLC acknowledges the T function
No entry: The T strobe is acknowledged by resetting the marker defined in MP_signal.

MP_revoke

Functions canceling the T function

Format: Array

Input: Functions canceling the T function defined here

MP_blockSearch

T function output during block scan

Format: Pull-down selection menu

Selection:

[True]
T function is output during the block scan

[False]
T function is not output during the block scan

Alias Functions (Alias Strobe)

Alias functions are used to map control-specific functions on M functions.

In **CfgPlcStrobeAlias,** you can define channel-sensitive, but control-specific, functions. Alias functions are configured in two steps:

- In the channel-sensitive parameter **MP_aliasStrobes,** you define key names for the functions. In this way, you assign the functions to the machining channel.
- In the parameter object **System/PLC/CfgPlcStrobeAlias,** you configure the functions.

Note: The alias functions are mapped on M functions. In the parameter object **CfgPlcMStrobe,** you define the M functions used.

The following topics are described:

- **Assigning Alias Functions to the Machining Channels**
- **Configuration of Alias Functions**

Assigning Alias Functions to the Machining Channels

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgPlcStrobes **aliasStrobes** **unitOfMeasure**	

MP_aliasStrobes .

Alias strobe description of this machining channel

Format: String [0–18]

Input: Key names of the alias strobes used in this machining channel

In the marker defined in **MP_unitOfMeasure,** the PLC is informed of the unit of measure used in the NC program when M, S, T, or alias strobes are executed (see "**MP_unitOfMeasure**").

Configuration of Alias Functions

Settings in the configuration editor:	
System PLC CfgPlcStrobeAlias Key for alias strobe **type** **mCode** **mOffset**	

MP_type specifies the type of call. In **MP_mCode,** you specify the M function on which the alias function is to be mapped.

MP_type

Type of alias function

Format: Pull-down selection menu

Selection:

[FN19]

Two values are transferred synchronously from the NC program to the PLC.

[FN29]

Max. eight values are transferred asynchronously from the NC program to the PLC.

[CYCLE13]

Define spindle position for M19

[TCHPROBE]

Call measuring cycles

MP_mCode

Number of the M function

Format: Numerical value

Input: Number of the M function for which the control-dependent function is mapped.

Use **MP_mOffset** to define whether the transferred M code applies directly, or whether the first datum transferred in an FN19 or FN29 function is added as an offset to the first M function (defined in **MP_min**).

MP_mOffset

Transferred M code is offset

Format: Pull-down selection menu

Selection:

[True]

The first datum transferred in FN19 or FN29 is added as an offset to the first M function (defined in MP_min).

[False]

The defined M code is transferred.

Error Messages and Log Files

The control displays errors in the header of the screen. Long error messages or error messages extending over more than one line are abbreviated. The complete information on all pending error messages is given in the error window.

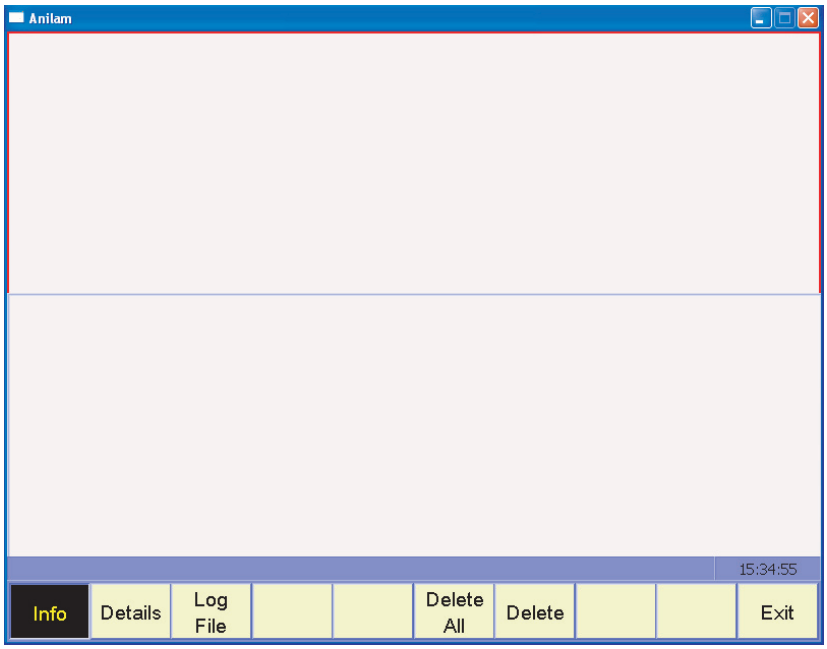
Errors and system information (system start, system end, etc.) are entered in the error log file. The control saves every keystroke and the mouse events in the keystroke log file.

The following topics are described:

- **Error Window**
- **Error Log**
- **Keystroke Log File**
- **Saving Log Files**
- **PLC Error Messages**
- **Structure of the Error Text File**

Error Window

From the Manual screen press (**SHIFT** + **F1**) **Msgs** to display the Error Window.



Meaning of the soft keys in the Error Window.

Info	Shows the top-half of the screen with the general error information
Details	Shows the bottom-half of the screen with specific error information
Log File	Access to the Log File screen
Delete All	Deletes all error information
Delete	Deletes the selected errors that you have highlighted
Exit	Exit the Error Window and return to the Manual screen.

The error window contains the details of all error that have occurred (see bottom-half of the screen - press **Details** (**F2**).

To obtain information on the cause of error and the corrective action, proceed as follows:

- Press the **INFO** (**F1**) soft key.

For further details regarding the error, such as date, time, event class, line of the NC program, control program reporting the error, etc., proceed as follows (see figure):

- Press the **DETAILS** (**F2**) soft key.

The following topic is described:

- **Deleting Errors**

Deleting Errors

To delete an individual error:

- Position the cursor on the entry to be deleted.
- Press the **DELETE** soft key.

To delete all errors contained in the error window:

- Press the **DELETE ALL** soft key.

Information provided by the **error message:**

- Error number: Assigned by ANILAM or the machine tool builder
- Error class: Defines the control reaction to this error (see table)

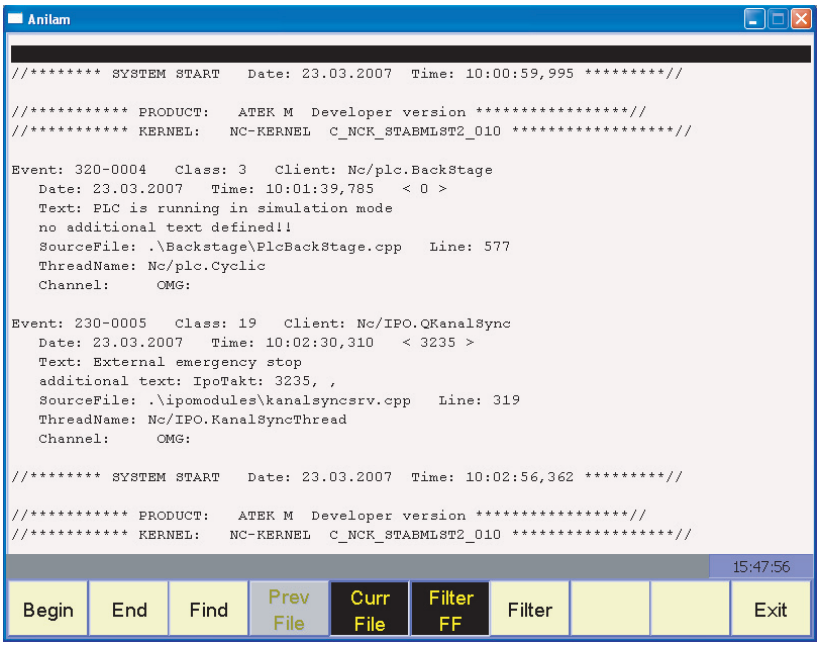
Press **Log File** (**F3**) to display the Log File screen. Meaning of the soft keys in the Log File screen:

Error Log	Opens the Error Log file which includes errors from the previous runs. See the Error Log screen.
Key Log	Opens the file with the history of keys pressed by the user
Service Files	Creates a ZIP file containing the error log files for service purposes
Exit	Exit the Log File Window and return to the Error Window.

Error Log

All errors that occurred and the error information, including all details in the error log file, are stored by the control (see figure):

From the Log File screen press (**F1**) **Error Log** to display the error log screen.



Meaning of the soft keys in the Error Log screen.

Begin	Shows the top of the log file
End	Shows the bottom of the log file
Find	Provides a pop-up window to specify your search
Prev File	Opens the previous run error file
Curr File	Enables you to toggle between the previous run and current run error file
Filter FF	Enables you to filter the types of errors that are displayed.
Filter	Provides new soft keys to select specific errors to display
Exit	Exit the Error Log screen and return to the Log File screen.

- Error text: Describes the error (in one or more lines) If the error occurs while an NC program is being run, the line of the NC program will also be indicated.

Overview of error handling in the control:

Error class	Reaction	Display	Log file entry	Acknowledgment	Error group
Ev_class_2	None	x			Warning
Ev_class_3	None		x		Warning
Ev_class_4	None	x	x		Warning
Ev_class_5	None	x	x	x	Error
Ev_class_6	Feed stop	x	x	x	Error
Ev_class_7	Program abortion	x	x	x	Error
Ev_class_8	Program aborts at stable position	x	x	x	Error
Ev_class_9	Emergency stop	x	x	x	Error
Ev_class_10	Reset	x	x	x	System error
Ev_class_11	NC stop	x	x		Error
Ev_class_12	NC stop	x	x	x	Error
Ev_class_13	Program abortion	x	x		Error
Ev_class_14	Reset – without output of error text (“Processor check error”)	x	x	x	System error
Ev_class_15	Feed stop	x	x		Error
Ev_class_16	Emergency stop	x	x		Error
Ev_class_17	Display informational text	x	x		Info
Ev_class_18	None	x	x		Warning
Ev_class_19	Program abortion	x	x	x	Error

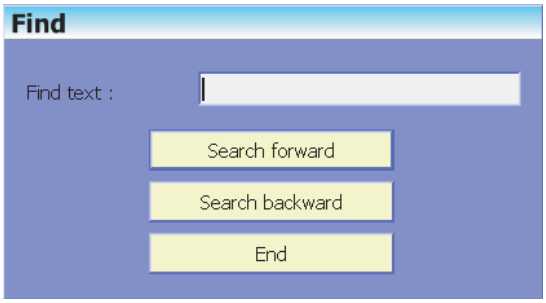
Ev_class_18 is used to report that service files were saved.

The following topics are described:

- **To Find a Log File Entry:**
- **Current and Previous Error Log File**
- **Filter**

To Find a Log File Entry:

- Press the **FIND** (**F3**) soft key.



- Enter the search string.
- Define the search direction.

Current and Previous Error Log File

The error log file uses two files, the **current file** and the **previous file**.

If the current file is full, the control switches the files. After converting the current file to the previous file, the control creates a new current file.

To switch from the current to the previous error log file:

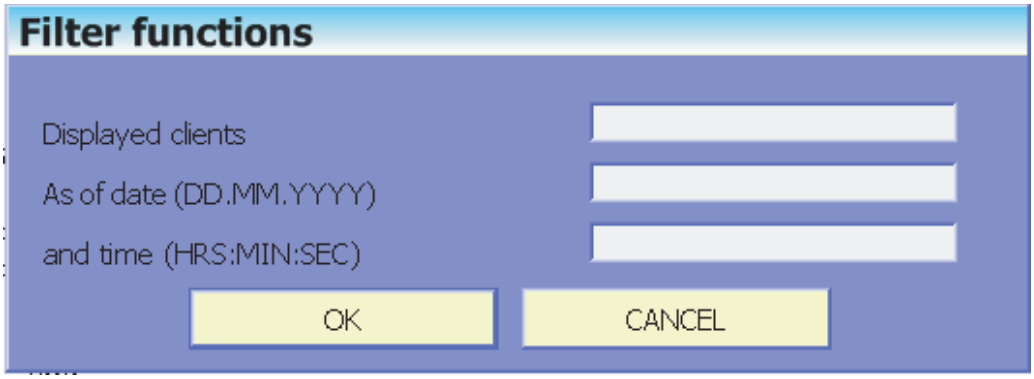
- Press the **PREVIOUS FILE** soft key.
- Press the **CURRENT FILE** soft key.

Filter

On the Error Log screen, press **Filter** (**F7**) to display the Filter screen. Use the filter to limit the log file display to the following error groups:

- **Info** (**F1**)
- **Warn** (**F2**)
- **Error** (**F3**)
- **System Error** (**F4**)

Use the **Func Filter** (**F5**) to select the following information:



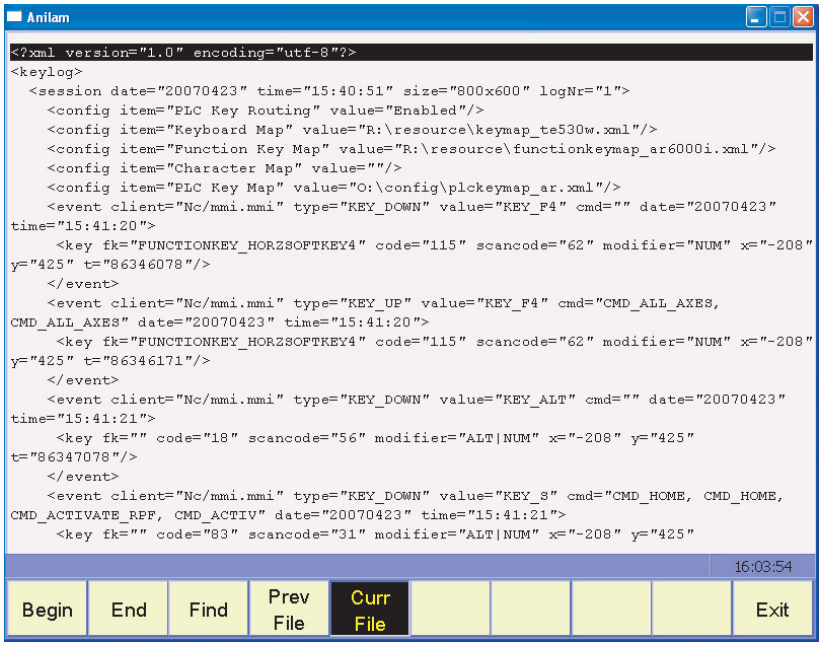
- Date and time from which you want the log file contents to be displayed.
- Clients whose errors and error information are to be considered in the log file display.

Highlight the filter (**F1**–**F4**) that you want to display (Filter On). Then select **Activate Filter** (**F7**) to display only those filters that are highlighted.

Keystroke Log File

On the **Msgs** (**F1**) screen, select **Log File** (**F3**), then press **Key Log** (**F2**) to display the Key Log screen All keystrokes and mouse events that occurred are stored by the control (see figure):

Call the keystroke log file from within the error system:



The following topics are described:

- **To Move Within the Log File:**
- **To Find a Log File Entry:**
- **Current and Previous Keystroke Log File**

To Move Within the Log File:

To move to the oldest entry:

- Press the **Begin** (**F1**) soft key.

To move to the most recent entry:

- Press the **End** (**F2**) soft key.

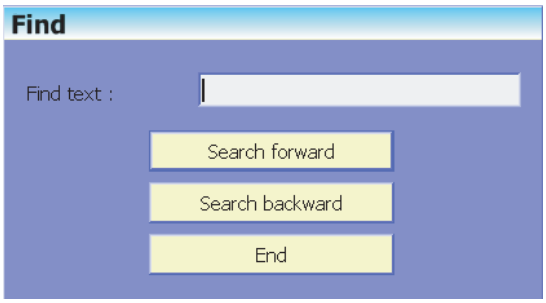
To view other log file entries:

- Move using the arrow keys (**UP ARROW**, **DOWN ARROW**, **PAGE UP**, **PAGE DOWN**).

To Find a Log File Entry:

Call the “Find” dialog box (this is the same as “To find a log file entry:”):

- Press the **Find** (**F3**) soft key.



- Enter the search string.
- Define the search direction.

Current and Previous Keystroke Log File

The keystroke log file uses two files, the **current file** and the **previous file**.

If the current file is full, the control switches the files. The current file is converted to the previous file and the previous file to the current file. The contents of the previous file are deleted before new entries are made.

To switch from the current to the previous error log file:

- Press the **Prev File** (**F4**) soft key.
- Press the **Currr File** (**F5**) soft key.

Saving Log Files

You can save the error log file, keystroke log file, the log files of the IPO and PLC as well as various other information on the memory card of the control.

This function is particularly interesting if servicing becomes necessary and you want to transmit the log files and the control configuration to the ANILAM Service department.

The control automatically packs the data and saves them in a *.ZIP file.
Path: **TNC:\service*.zip**

To save log files:

- Call the error system by pressing the **Msgs** (**F1**) soft key**.**
- Press the **Log File** (**F3**) soft key.
- Press the **Services Files** (**F7**) soft key.
- The control automatically creates the *.ZIP file **TNC:\service*.zip.**
  * = number; if more than one service *.ZIP files are contained on the control, the files are numbered in increasing order from old files to new files.

PLC Error Messages

Settings in the configuration editor:	
System Path CfgPlcPath **errorTable** **errorText**	

PLC error messages are defined in the PET table (PLC Error Table). If the PLC detects an error, it is transferred to the error system by Module 9084, Module 9085, or by activating a marker defined in the PET table. The error system ensures that the error is displayed and handled properly.

With Module 9086 you can delete PLC error messages, and with Module 9087 you can interrogate the current status of the error message.

The following topics are described:

- **PET Table (PLC Error Table)**
- **Error Text File**
- **Structure of the PET Table**

PET Table (PLC Error Table)

- Enter the path and file name of the PET table in **MP_errorTable**.
- Enter the file name of the text file for PLC error messages in **MP_errorText**.

Note: A *.PET table is absolutely mandatory, since without it the PLC program cannot be compiled or activated.
Use the program “PLC-Text” to enter data in the PET table.
If a *.PET table contains more than 999 error messages, the excessive messages are ignored and the error message **PET table: Too many lines** is displayed.

MP_errorTable
PLC error message table
Format: String
Input: Path and file name of the PET table

Error Text File

Error text are defined directly in the PET table (max. 32 characters; not language-dependent) or in the error text file. In the error text file, you define the error text to be displayed as well as the information on the cause of error and corrective action.

Error text files are language-dependent. The path for the error text file is permanently defined: %OEM%\plc\language\en (or another language abbreviation).

In MP_System/DisplaySettings/CfgDisplayLanguage/plcErrorLanguage, you define the language to be used.

You define the name of the error text files in MP_errorText.

MP_errorText
: Text file for PLC error messages

Format: String

Input: Path and file name of the PET table

Structure of the PET Table

The PLC error message table (*.PET) consists of the following columns, to which you can assign special attributes:

- **NR**
  Line number in the table. The modules select the PLC error message by assigning the line number.
- **ERROR**
  There are the following ways to specify the error text:
  - Direct entry of the error text (max. 32 characters)
  - Line number of the PLC error text file (# <line no.>) defined in MP_errorText.
- **MARKER**
  The PLC error message can be activated without module call by setting the marker defined here. The marker is also set if the error message was activated through Module 9085. Enter the symbolic name of the marker to be set.
  Entry 0: No error marker
- **Error class:** The error class is defined in the following columns (see “**Error Status**”). If none of these error classes is set in the PET table, NN_ChnErrorWarning is set.
- **RESET**
  0: No NC reset upon activation of the error message (no system error).
  1: NC reset upon activation of the error message (system error). The PLC program stops.
- **NC_STOP**
  0: No NC stop upon activation of the error message
  1: NC stop upon activation of the error message (NN_ChnErrorNcStop is set).
- **NC_CANCEL**
  0: No NC stop with subsequent INTERNAL STOP upon activation of the error message
  1: NC stop with subsequent INTERNAL STOP upon activation of the error message (NN_ChnErrorCancel is set).

- **F_STOP**
  0: Feed-rate enable is not influenced
  1: Feed rate-enable is reset upon activation of the error message (NN_ChnErrorFStop is set).
- **EMER_STOP**
  0: No EMERGENCY STOP upon activation of the error message
  1: EMERGENCY STOP upon activation of the error message (NN_ChnErrorEmergencyStop is set).
- **CE**
  0: Error message can be deleted by the user.
  1: Error message cannot be deleted by the user.
- **PRIO**
  A priority of 0 to 2 can be entered for the error message, with priority 0 being the highest priority. If the PLC triggers more than one error at the same time, the errors with the highest priority are the first to be sent to the event server (error system).
- **WARN_LVL:** Not evaluated.

Structure of the Error Text File

In the error text file, there are four columns with the following meanings:

- **Reference number:** This reference is used in the PET table (“Error” column).
- **Error text:** Displayed error text.
- **Cause of error:** Text that is displayed under “Cause” after you have pressed the **INFO** soft key.
- **Corrective action:** Text that the error system displays under “Action” after you have pressed the **INFO** soft key.

The following topics are described:

- **Module 9084 Display PLC error messages with additional data**
- **Module 9085 Display PLC error messages**
- **Module 9086 Delete PLC error message**
- **Module 9087 Status of PLC error message**
- **Entering Data in Log Files**

Module 9084 Display PLC error messages with additional data

The Module displays PLC error messages with additional data. You can insert place holders (%s, %d, %f) at any position of the error messages or language-sensitive texts from the ***.csv** are also supported. The place holders are assigned the data from the module at run time. Only those place holders that are defined in the PLC error message will be replaced. %s is replaced by the string or the string content. The first occurrence of %d or %f in the PLC error message is replaced by the content of variable 1, and the second occurrence of %d or %f is replaced by the content of variable 2. %d is an integer, %f is a floating point number with three decimal places. Alternately, you can define the number of decimal places with %.1f to %.6f.

If the module is called several times with the same line number of the *.PET table, the error message is entered only once in the queue. A maximum of 32 PLC error messages can be entered in the queue.

If an error marker is assigned in the PET table, it is set.

If the *.PET table or the line number is not found, the error message **PLC ERROR <line number>** is displayed.

Call:

PS	B/W/D/K	<>line number of the *.PET table>
		0 to 999: Line number
PS	B/W/D/K/S	<>Data for %s>
PS	B/W/D/K	<>Data for %d or %f; variable 1>
PS	B/W/D/K	<>Data for %d or %f; variable 2>
CM	9084	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	PLC error message with additional data displayed
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Line number not available
	8	Incorrect operating mode, compatibility error marker set
	23	Overflow of PLC error message queue

Module 9085 Display PLC error messages

The Module transfers PLC error messages to the error system. The error message texts come directly from the compiled error table (.PET) or from the selected text file for PLC error messages. PLC error messages (except reset errors) can be deleted by Module 9086 or by the user. However, deletion can be disabled in the error table (“CE” column).

Up to 32 error messages can be placed in the queue.

If an error marker is assigned to the error, it is set.

System error: Is displayed without entry in the queue.

Error number –1: System error message **EMERGENCY STOP PLC** is displayed. This error message also occurs if no *.PET table was defined.

Error number not equal to –1 and no *.PET table selected:
System error message **PLC: NO ERROR TABLE SELECTED**

Call:

PS B/W/D/K <>Line number of the *.PET table>
0 to 999: Line number
–1: System error message **EMERGENCY STOP PLC**

CM 9085

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Error message displayed or in queue
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Line number not available
	8	Incorrect operating mode, compatibility error marker set
	23	Overflow of PLC error message queue, or too many error messages from string memory

Module 9086 Delete PLC error message

Use this module to erase all set PLC error messages or a specific error message. System errors cannot be deleted.

Call:

PS B/W/D/K <>Line number of the *.PET table>
0 to 999: Line number
–1: Erase all PLC error messages

CM 9086

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Error message displayed or in queue
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Line number not available
	8	Incorrect operating mode, compatibility error marker set

Module 9087 Status of PLC error message

The module interrogates the status of a PLC error message or in general the PLC error status. In addition, the number of the error message active on the screen and the total number of PLC error messages in the error list can be interrogated.

Call:

PS B/W/D/K <>Line number of the *.PET table, status code>
0 to 999: Line number
–1: PLC error message, general
–2: Number of the active PLC error message
–3: Number of error messages in the *.PET table

CM 9087

PL B/W/D <>Status/error code>
For code 0 to 999:
0: No error message with the number, or message deleted
–1: Line number does not exist
Bit 0 – PLC error message is displayed
Bit 1 – PLC error message in queue

For code –1:
0: No PLC error message
2: PLC error message in queue

For code –2:
Š 0: Number of the displayed error
–1: No error in the *.PET table
For code –3:
Š 0: Number of errors in the *.PET table

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Status information was read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid line number of status code

Entering Data in Log Files

The error log file can be used by the PLC for diagnostic purposes.

Enter **PLC** data in the error log file:

- Use Module 9275 to write ASCII data into the error log file.
- Use Module 9276 to write the contents of operands into the error log file.

Note: Do not use Modules 9275 and 9276 in the PLC program as shipped. Instead, use them only for debugging. Otherwise the processing times could be increased and the hard disk could be written to unnecessarily.

The following topics are described:

- **Module 9275 Write ASCII data into the log**
- **Module 9276 Write operand contents into the log**

Module 9275 Write ASCII data into the log

The module writes a character string from a PLC string or an immediate string into the error log file. The entry can be given a special identifier for fast finding or later editing.

A buffer of approx. 210 bytes is available for the data to be written (including the entry identification).

Call:

PS	B/W/D/K/S	<>Log entry> –1: No entry
PS	B/W/D/K/S	<>Log identifier> –1: No entry
PS	B/W/D/K	<>Priority> 0: Information 1: Warning 2: Error
CM	9275	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Entry was written
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid priority
	2	Invalid string number or invalid immediate string
	12	No string end identifier
	20	Module was not called in a spawn job or submit job

Module 9276 Write operand contents into the log

The module writes the contents of PLC operands into the error log file. The entry can be given a special identifier for fast finding or later editing.

A buffer of approx. 210 bytes is available for the data to be written (including the entry identification).

The operands M/I/O/C/T are stored in binary format (e.g.,110101), the operands B/W/D in hexadecimal format.

Call:

PS	B/W/D/K	<>Identifier operand name> 0: M (marker) 1: I (input) 2: O (output) 3: C (counter) 4: T (timer) 5: B (byte) 6: W (word) 7: D (double word)
PS	B/W/D/K	<>Address of the first operand>
PS	B/W/D/K	<>Number of operands>
PS	B/W/D/K/S	<>Log identifier> –1: No entry
PS	B/W/D/K	<>Priority> 0: Information 1: Warning 2: Error
CM	9276	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Entry was written
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid priority
	2	Invalid identifier for operand name
	3	Invalid first operand address
	4	Sum of first operand address and number of operands invalid
	5	Address is not a word/double-word address
	12	No string end identifier
	20	Module was not called in a spawn job or submit job
	36	Entry in the log was shortened to 210 characters

Keystroke Simulation

The 6000i features two operating panels:

- Integrated control keyboard
- MB 420 machine operating panel

The machine operating panel is connected to X46 of the MC 400.

The key code of the control keyboard unit is evaluated directly by the NC.

PLC inputs and outputs for the machine control panel are available on connector X46. You must evaluate the inputs and outputs in the PLC.

The following topics are described:

- **Control Keyboard**
- **Machine Operating Panel**

Control Keyboard

The key code is displayed in **NP_GenKeyCode** while a key of the control keyboard is being pressed. .

The following modules can influence keys and soft keys:

- **Module 9180 Simulation of NC keys**
- **Module 9181 Disabling of individual keys**
- **Module 9182 Re-enabling of individual keys**
- **Module 9183 Disabling groups of NC keys**
- **Module 9184 Re-enabling of groups of NC keys**
- **Module 9186 Calling a soft-key function**
- **Module 9187 Status of a soft-key function call**

PLC operand	Type
NP_GenKeyCode **Code of the depressed key**	D

Module 9180 Keystroke simulation

The module simulates NC keys and soft keys. You transfer the code of the desired key.

If you transfer the code value zero, the number of occupied elements in the keystroke queue is returned. In this case there is no keystroke simulation.

Call:

PS	B/W/D/K	<>Key code>
CM	9180	
PL	B/W/D	<>Number of occupied elements / error status> 0: Key code was transferred, key queue is empty 1 to 16 : Key code was not yet simulated, max. 16 entries in the keystroke queue are possible –1: For error see NN_GenApiModuleErrorCode

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	NC key was simulated
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Transferred parameter > maximum value
	2	Transferred parameter invalid
	22	Keystroke queue overflow

Module 9181 Disable NC key by PLC

The module disables individual NC keys.

Call:

PS	B/W/D/K	<>Key code>
CM	9181	
PL	B/W/D	<>Error status> 0: NC key disabled –1: For error see NN_GenApiModuleErrorCode

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	NC key was disabled
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Transferred parameter > maximum value
	2	Transferred parameter invalid

Module 9182 Re-enable NC key by PLC

The module cancels the effect of Module 9181.

Call:

PS	B/W/D/K	<>Key code>
CM	9182	
PL	B/W/D	<>Error status>
		0: NC key enabled
		–1: For error see NN_GenApiModuleErrorCode

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Disabling was canceled
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Transferred parameter > maximum value
	2	Transferred parameter invalid

Module 9183 Disable NC key groups by PLC

The module disables the specified key group. The table at the end of this chapter contains the assignment of the keys to the key groups.

The key-group codes are:

- 0: All keys
- 1: ASCII
- 2: Soft keys, Page Up/Down
- 3: Cursor, ENT, NOENT, DEL, END, GOTO
- 4: Numbers, algebraic signs, decimal point, actual position capture
- 5: Operating modes
- 6: Block opening keys

Call:

PS	B/W/D/K	<>Key-group code>
CM	9183	
PL	B/W/D	<>Error status>
		0: Group of NC keys disabled
		–1: Transferred value> maximum value

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	The group of NC keys was disabled
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Transferred parameter invalid

Module 9184 Enable locked NC key groups by PLC

The module cancels the effect of Module 9183 for the given key group. The table at the end of this chapter contains the assignment of the keys to the key groups.

The key-group codes are:

- 0: All keys
- 1: ASCII
- 2: Soft keys, Page Up/Down
- 3: Cursor, ENT, NOENT, DEL, END, GOTO
- 4: Numbers, algebraic signs, decimal point, actual position capture
- 5: Operating modes
- 6: Block opening keys

Call:

PS	B/W/D/K	<>Key-group code>
CM	9184	
PL	B/W/D	<>Error status> 0: Group of NC keys enabled –1: Transferred value> maximum value

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Disabling was canceled
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Transferred parameter invalid

Module 9186 Call a soft key function

The module calls certain soft-key functions in the machine operating modes.

Do not call a new function until the previous one is completed. Use Module 9187 to interrogate the status of the soft key function call.

For a soft-key function to be simulated it must be displayed either in the foreground or background operating mode. Otherwise the module has no effect. Module 9187 reports the error.

The following codes apply for the 6000i soft keys:

- 0: INTERNAL STOP
- 1: M output
- 2: S output
- 3: PROBE FUNCTION
- 4: PASS OVER REFERENCE MARK
- 5: RESTORE POSITION
- 6: INCREMENTAL JOG
- 7: Feed-rate limitation F MAX

Call:

PS	B/W/D/K	<>Number of the soft-key function>
CM	9186	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Soft-key function was called
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Parameter out of value range
	28	Previous call not ended

Module 9187 Status query of a soft-key call

Immediately after Module 9186 is called, the status 1 (soft-key function not yet completed) is set — regardless of whether the function can be run in the current operating mode. Module 9186 cannot be called again until status 0 or 2 is set. The error status 2 is erased if Module 9186 is called or if power is switched on.

Call:

CM	9187	
PL	B/W/D	<>Status> 0: Soft-key function completed or none called 1: Soft-key function not yet completed 2: Error: Soft-key function cannot be completed because soft key is not available or operating mode is incorrect

Machine Operating Panel

On socket X46 there are 25 PLC inputs (I128 to I152) and eight PLC outputs (O0 to O7) for evaluating the keys on the machine operating panel.

You can activate specific functions by linking the PLC inputs with the corresponding markers and words.

Save the depressed axis-direction key:

If you enable the memory function with **PP_ChnEnableAxisKeyLatch,** the axis will move until an NC STOP is triggered.

PLC operand	Type
PP_AxTraversePos **Manual traverse in positive direction** 0: Do not move axis 1: Move axis	M
PP_AxTraverseNeg **Manual traverse in negative direction** 0: Do not move axis 1: Move axis	M

PLC operand	Type
PP_ChnEnableAxisKeyLatch **Saving of axis-direction key is enabled** 0: Saving not enabled 1: Saving enabled	M
PP_ChnRapidTraverseKey **Rapid traverse key** 0: Rapid-traverse key not pressed 1: Rapid-traverse key pressed	M

Electronic Handwheel

The following topics are described:

- **General Handwheel Parameters**
- **Serial Handwheel**
- **Handwheel at Position Encoder Input**
- **Traverse Per Handwheel Revolution**
- **Assigning a Handwheel to an Axis**

General Handwheel Parameters

The following topics are described:

- **Type of Handwheel**
- **Threshold Sensitivity**
- **Locking the Handwheel**

Type of Handwheel

Settings in the configuration editor:	
System CfgHandwheel **type** **sensitivity**	

The control supports:

- Handwheels that are connected to the serial handwheel input X23 of the MC.
- Handwheels with position encoders, which are connected to the position encoder inputs of the MC.

For information about mounting and electrical connection, see “**Section 2, Handwheel Input**”.

You can connect the following handwheels to the handwheel input (X23) of your control:

- One panel-mounted HR 130 handwheel, or
- One RM 500 portable handwheel

As an alternative or in addition, you can connect rotary encoders to the position encoder inputs of the MC and use them as handwheels.

- Define the type of handwheel, or the connection of the handwheel to the control in **MP_type.** If the handwheel is connected to a position encoder input (MP_type=ENCODER), the handwheel connection is described in **MP_CfgAxisHandwheel.**

MP_type

Handwheel

Format: Pull-down selection menu

Selection:

[NONE]
No handwheel connected.
[HRNAX]
HRA 110 handwheel adapter connected to X23
[HR410]
RM 500 connected to X23
[HR332]
HR 332 connected to X23
[HR330]
HR 330 or HR 130 connected to X23
[ENCODER]
Handwheel(s) connected to position encoder input

Threshold Sensitivity

- Shock or vibrations can cause a slight motion at the handwheel and produce an unintentional axis movement.
- In **MP_sensitivity,** enter a threshold sensitivity, in order to avoid unintentional movements.

MP_sensitivity

Sensitivity for electronic handwheel

Format: Numerical value

Input: 0 to 10 000 [pulses]

Default: 10 [pulses]

Locking the Handwheel

Disable handwheel pulses for specific axes with **PP_AxHandwheelLocked.** Disable the handwheel pulses for all axes with **PP_GenHandwheelLocked.** If **PP_GenHandwheelLocked** is reset, axis-specific disabling of **PP_AxHandwheelLocked** is effective again.

PLC operand	Type
PP_GenHandwheelLocked **Disable handwheel motions** 0: Enable handwheel movements for all axes 1: Disable handwheel movements for all axes	M
PP_AxHandwheelLocked **Disable handwheel motion for specific axes** 0: Enable handwheel movements for this axis 1: Disable handwheel movements for this axis	M

Serial Handwheel

Settings in the configuration editor:	
System CfgHandwheel **initValues** **incrPerRevol** **rasterPerRevol** **countDir** **feedFactor** **crossShortSafety**	

The following topics are described:

- **Handwheel Initialization**
- **Evaluation of Handwheel Pulses**
- **Handwheel with Manual Axis-Direction Keys**

Handwheel Initialization

You enter initialization values for serial handwheels in MP_initValues. During startup, the control transmits the initialization values to the handwheel.

Information about the initialization values:

- RM 500:

MP_initValues

Initialization values for handwheel

Format:	Array	[7]
Input:	0 to 255	
Default:	0	

Evaluation of Handwheel Pulses

Define the evaluation of the handwheel pulses in the following parameters:

- **MP_incrPerRevol:** Number of increments per handwheel revolution
- **MP_rasterPerRevol:** Detent steps per handwheel revolution (only for handwheel with detent)
- **MP_countDir:** Direction for handwheel input

MP_incrPerRevol

Increments per handwheel revolution

Format: Numerical value

Input: 0 to max. value

0: Standard value for ANILAM handwheel (20 000 increments)

Default: 0

MP_rasterPerRevol

Detent steps per handwheel revolution

Format: Numerical value

Input: 0 to max. value

0: Handwheel without detent

Default: 0

MP_countDir

Counting direction for handwheel

Format: Pull-down selection menu

Selection:

[Positive]

Positive counting direction

[Negative]

Negative counting direction

Default: Positive

Handwheel with Manual Axis-Direction Keys

For handwheels with axis-direction keys (e.g., RM 500), you define three feed rates in MP_feedFactor. These entries refer to the feed rate entered in **MP_CfgFeedLimits/ manualFeed**.

MP_feedFactor

Manual feed rates in the Electronic Handwheel mode

Format: Array [0–2]

Input: 0 to 100 [%]

[0] = First feed rate in [%]
[1] = Second feed rate in [%]
[2] = Third feed rate in [%]

Default: 1, 10, and 100 [%]

Note: **MP_feedFactor** only applies in the EL. HANDWHEEL mode, but then for all manual axis-direction keys including the manual axis-direction keys of the operating panel.

In MP_crossShortSafety, you define whether a short-circuit test is to be performed on the permissive buttons.

MP_crossShortSafety

Short-circuit-proofed handwheel

Format: Pull-down selection menu

Selection:

[On]
Short-circuit test on

[Off]
Short-circuit test off

Default: Off

Handwheel at Position Encoder Input

Settings in the configuration editor:	
Axes ParameterSets Key for parameter set CfgAxisHandwheel **input** **incrPerRevol** **rasterPerRevol** **encoderSignal** **encoderFreq** **encoderResistor**	

The following topics are described:

- **Handwheel Connection**
- **MP_EncoderSignal**
- **Evaluation of Handwheel Pulses**

Handwheel Connection

Note: You define the handwheel parameters within the parameter block of an axis. The handwheel is **permanently** assigned to this axis.

Define the handwheel connection in the following parameters:

- MP_input: Assignment of handwheel to position encoder input
- MP_EncoderSignal: 1 V_{PP} or 11 μA_{PP} signal
- MP_EncoderFreq: Maximum input frequency
- MP_EncoderResistor: Terminating resistor

MP_input

Handwheel connector at encoder input

Format: Pull-down selection menu

Selection:

[Off]
No handwheel connected to position encoder input
[X01]
Handwheel connected to X01 of the MC
[X02]
Handwheel connected to X02 of the MC
[X03]
Handwheel connected to X03 of the MC
[X04]
Handwheel connected to X04 of the MC
[X05]
Handwheel connected to X05 of the MC
[X06]
Handwheel connected to X06 of the MC

Default: Off

MP_EncoderSignal

Signal amplitude at position encoder input for handwheel

Format: Pull-down selection menu

Selection:

[1 Vpp]
Input signal of encoder is 1 Vpp signal.

[11 µA]
Input signal of encoder is 11 µA signal.

[TTL]
Input signal of encoder is TTL signal.

MP_EncoderFreq

Input frequency of position encoder input for handwheel

Format: Pull-down selection menu

Selection:

[Fast]
Input frequency of 350 kHz

[Slow]
Input frequency of 50 kHz

MP_EncoderResistor

Terminating resistor of position-encoder input for handwheel

Format: Pull-down selection menu

Selection:

[Without]
Without terminating resistor

[120 Ohm]
With terminating resistor

Default: Without

Evaluation of Handwheel Pulses

Define the evaluation of the handwheel pulses in the following parameters:

- MP_incrPerRevol: Number of increments per handwheel revolution
- MP_rasterPerRevol: Detent steps per handwheel revolution (only for handwheel with detent)

MP_incrPerRevol

Increments per revolution of handwheel at encoder input

Format: Numerical value

Input: 0 to max. value

0: Standard value for ANILAM handwheel (20 000 increments)

Default: 1024

MP_rasterPerRevol

Detent steps per revolution of handwheel at encoder input

Format: Numerical value

Input: 0 to max. value

0: Handwheel without detent

Default: 100

Traverse Per Handwheel Revolution

Settings in the configuration editor:	
System CfgHandwheel **speedFactor** Axes ParameterSets Key for parameter set CfgAxisHandwheel **distPerRevol**	

The distance covered by an axis per handwheel revolution depends on the traverse per handwheel revolution and the handwheel transmission ratio.

- In **MP_distPerRevol,** you define the distance traversed per handwheel revolution for each axis moved by handwheel.
- In **MP_speedFactor,** you define the handwheel transmission for three transmission ratios.

MP_distPerRevol

Traverse per handwheel revolution

Format: Numerical value

Input: 0.000 000 001 to max. value [mm]

Default: 1 [mm]

MP_speedFactor

Handwheel transmission ratio

Format: Array [0–2]

Input: 0 to 100 [%]
[0] = First ratio in [%]
[1] = Second ratio in [%]
[2] = Third ratio in [%]

Default: 1, 10, and 100 [%]

The transmission ratio is set either at the control panel or directly on the handwheel.

The distance traversed by the axis during handwheel operation is calculated according to the following formula:

$$W = \frac{distPerRevol \times speedFactor}{100}$$

Legend:

- *W*: Traverse distance
- distPerRevol: Parameter **MP_distPerRevol**
- speedFactor: Ratio defined for the parameter **MP_speedFactor**

Assigning a Handwheel to an Axis

Module 9036 allows you to assign a handwheel connected to connector X23 to an axis. The module also defines the transmission ratio.

The following topics are described:

- **Module 9036 Write NC status information**
- **Module 9035 Read NC status information**
- **A Rotary Axis is a Special Case**
- **Master-slave Axes are Special Cases**

Use Module 9035 to find the axis to which the handwheel is assigned.

Module 9036 Write NC status information

The module modifies status information from the NC. The status information to be modified is transferred by function number.

- **Select the handwheel axis** function: The handwheel connected to connector X23 of the MC is assigned to an axis.
- **Set the handwheel transmission ratio:** The handwheel transmission ratio is defined.

The call parameters are listed in the following table. .

Number of the function	Function	Value
0	Handwheel interpolation X	0–10: Interpolation factors
1	Handwheel interpolation Y	0–10: Interpolation factors
2	Handwheel interpolation Z	0–10: Interpolation factors
3	Handwheel interpolation IV	0–10: Interpolation factors
4	Handwheel interpolation V	0–10: Interpolation factors
5	Handwheel interpolation for all axes	0–10: Interpolation factors
6	Select the handwheel axis	Index from MP_axisList or -1: Deselect all axes
7	Set the handwheel transmission ratio	n 0: Slow n 1: Normal n 2: Fast
8	Reserved	–
9	Reserved	–
10	Limit value for jog increment	0.0001–50 mm or –1: Cancel the limiting 2: New jog increment = minimum (programmed jog increment, previous limit value), and cancel limitation
11–19	Handwheel interpolation of axes 1 to 9	0–10: Interpolation factors

Constraints:

- Handwheel interpolation factors are limited to the smallest possible value in accordance with the rapid traverse of the corresponding axis. This does not result in an error message, however.
- Call codes 0–4 refer to the five axes that are assigned to the axis keys X/Y/Z/IV/V. Call codes 11–19 refer to the internal axis numeration.
- The value for the jog increment limitation is given in [1/10000 mm].
- If a jog limitation is entered in the inch mode, the limitation value [degrees] for rotary axes is calculated from the limitation value [mm] / 24.5.

Possible errors:

- The input parameter <number of the function> does not refer to any overwritable status information in this software version.
- The transferred value is outside of the range valid for this status information.
- Entry of this status information is disabled (e.g., via the machine configuration).

Call:

PS	B/W/D/K	<Number of the function>
PS	B/W/D/K	<Value to be written>
CM	9036	
PL	B/W/D	<Error code>
		0: Status written
		1: Incorrect status code
		2: Transferred value is out of range
		3: Input disabled

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Status information was written
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Transferred value is out of range
	2	Incorrect number of the status information
	3	Transferred value is out of range
	6	Input disabled

Module 9035 Read NC status information

Module 9035 reads status information. A function number specifying the desired status information is transferred.

Transferred number		Return code
8	Selected machine axis (for actual-position-capture)	0: X axis 1: Y axis 2: Z axis 3: IV axis 4: V axis 5: VI axis 6: VII axis etc.
9	Handwheel axis	Finds the axis which is assigned to the handwheel connected to connector X23 of the MC. –1: None or more than one 0: X axis 1: Y axis 2: Z axis 3: IV axis 4: V axis 5: VI axis 6: VII axis etc.
10	Handwheel axis, bit-encoded	Bit 0: X axis Bit 1: Y axis Bit 2: Z axis Bit 3: IV axis Bits 4 to 13: Axes 5 to 14 (only available for PLC programs that work with API 1.0)
	Handwheel interpolation factor	
11	X key	0 to 10
12	Y key	
13	Z key	
14	IV key	
15	V key	
19	Active line in the *.CMA file	>=0: Line number –1: No *.CMA file
	Handwheel interpolation factor	
31	Axis 1	0 to 10
32	Axis 2	
33	Axis 3	
34	Axis 4	
35	Axis 5	
36	Axis 6	
37	Axis 7	
38	Axis 8	
39	Axis 9	
20	HR 410 speed	0: Slow 1: Medium 2: Fast

Transferred number		Return code
21	Control model	0: TNC 310 1: TNC 370 2: TNC 410 3: TNC 426 CA/PA 4: TNC 426 CB/PB/M or TNC 430 CA/PA/M 5: iTNC 530 6: iTNC 530 (with Windows®**1 2000) 20: **NC-Kernel based control** (TNC 320)
23	Handwheel superimposition with M118	0: M118 not active Bits 0 to 13: Axes 1 to 14
26	Jog increment	
	Handwheel interpolation factor	0 to 10
31	Axis 1	
32	Axis 2	
33	Axis 3	
34	Axis 4	
35	Axis 5	
36	Axis 6	
37	Axis 7	
38	Axis 8	
39	Axis 9	
100	Number of the tool axis	Only available for PLC programs that work with API 1.0
	Cycle counter	
500	Interpolator	0–
501	Cycle time of the PLC	
502	Current utilization	
503	Maximum utilization	
1001	Pallet table (only in a spawn job or submit job)	>= 0: Active line in the pallet table –1: Pallet table not active

Constraint:

Status information no. 19 (read active line in the *.CMA file) is displayed even if the active line does not contain any *.COR file.

Call:

PS	B/W/D/K	<Number of the desired status information>
CM	9035	
PL	B/W/D	<Status information>

**1 Windows® is a registered trademark of Microsoft Corporation.

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Status information invalid
	20	Call was not in a submit or spawn job

A Rotary Axis is a Special Case

For a rotary axis, only the compensation values for the entries of 0° to +60° are effective, relative to the machine datum. Therefore, the datum for the nonlinear compensation must lie within the 0° to +360° range. To compensate a full circle, set the compensation value datum to the machine datum.

Master-slave Axes are Special Cases

Separate compensation tables can be created for master axes and slave axes.

Override

The following topics are described:

- **Override Devices**
- **Compensation for Potentiometers**
- **Override Functions**

Override Devices

Settings in the configuration editor:	
System PLC CfgPlcOverrideDev Key for override device **source** **mode** **values**	

The control supports the following override devices:

- Up to two potentiometers
- Group of keys (You can use two machine operating keys for setting the override value.)

The input values of the override devices (potentiometer setting or key input) are evaluated as follows:

- DISCRETE: The control converts the input values into a maximum of 64 discrete override values.
- LINEAR: The control interpolates the input values linearly between the minimum and maximum override value.
- CURVE: The control uses a curve to convert the lower input values into override values. In this way you can achieve a finer resolution of the lower override values. Higher input values are again interpolated linearly.

Define the override devices as follows:

- Define the override device in MP_source.
- In MP_mode, define the evaluation of the override input.
- In MP_values, define the discrete override values or the interpolation points of the curve.

MP_source

Selection of configurable source for override values

Format: Pull-down selection menu

Selection:

[OVR1]

Potentiometer 1

[OVR2]

Potentiometer 2

[KEY]

Group of keys

MP_mode

Evaluation of override values

Format: Pull-down selection menu

Selection:

[DISCRETE]

The key inputs or potentiometer settings are converted in up to 25 discrete override values from MP_values.

[LINEAR]

The input values of the override device are interpolated linearly between the minimum and maximum override value.

[CURVE]

The input values of the override device is converted using the curve defined in MP_values. You define the curve with up to 64 interpolation points in MP_values. The input values of the override device are again interpolated linearly above the last interpolation point specified.

If no curve is defined in MP_values, the control uses a standard curve.

MP_values

Discrete values or interpolation points for curve

Format: Array [0–63]

Input: 0.000 to 200.000 [%]

The meaning of the parameter depends on MP_mode:

MP_mode=DISCRETE: Enter override values for a maximum of 64 key inputs or potentiometer settings.

MP_mode=LINEAR: MP_values has no function.

MP_mode=CURVE: Define a curve with up to 64 interpolation points. The override values are taken from the curve. Linear interpolation is again effective above the last interpolation point specified.

Compensation for Potentiometers

Settings in the configuration editor:	
System PLC CfgPlcPeriphery **overrideFullRatio** **overrideDelta** **overrideIntegDelta**	

MP_overrideFullRatio allows you to compensate voltage losses.

MP_overrideFullRatio
Compensation for cable losses of the override potentiometer
Format: Numerical value
Input: 0.5000 to 1.0000
Default: 0.98

Use **MP_overrideDelta** and **MP_overrideIntegDelta** to influence the sensitivity of the override potentiometers. **MP_overrideDelta** suppresses short-term fluctuations and **MP_overrideIntegDelta** compensates the signal drift.

MP_overrideDelta
Compensation for thermal noise in override potentiometer
Format: Numerical value
Input: 0.00010 to 0.10000
Default: 0.0005

MP_overrideIntegDelta
Compensation for thermal noise in override potentiometer
Format: Numerical value
Input: 0.00010 to 1.00000
Default: 0.025

Override Functions

Settings in the configuration editor:	
System PLC CfgPlcOverrideS Key name of spindle **minimal** **maximal** **source**	

The following topics are described:

- **Speed Override**
- **Feedrate Override**
- **Rapid Traverse Override**

Speed Override

In the parameter object **CfgPlcOverrideS,** create a parameter block for each spindle to which a spindle speed override is to apply.

MP_minimal
Minimum value for override
Format: Numerical value
Input: 0.000 to 100.000 [%]
Default: 0

MP_maximal
Maximum value for override
Format: Numerical value
Input: 0.000 to 200.000 [%]
Default: 150

MP_source
Source for override values
Format: String
Input: String of max. 16 characters
Key name for override device from CfgPlcOverrideDev

The percentage adjusted with the spindle speed override is entered by the NC in **NN_SpiOverrideInput** and **PP_SpiOverride.**

You can change the percentage through the PLC:

- Enter the desired percentage in **PP_SpiOverride.** The NC immediately takes over the new value.

PLC operand	Type
NN_SpiOverrideInput **Speed override entered [%]**	D
PP_SpiOverride **Speed override entered by the PLC [%]**	D

Feedrate Override

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgPlcOverrideF **minimal** **maximal** **source**	

In the channel-sensitive parameter object **CfgPlcOverrideF,** create a parameter block for each machining channel (slide) to which a feed rate override is to apply.

MP_minimal
Minimum value for override
Format: Numerical value
Input: 0 to 100 [%]
Default: 0

MP_maximal
Maximum value for override
Format: Numerical value
Input: 0 to 200 [%]
Default: 150

MP_source
Source for override values
Format: String
Input: Key name for override device from CfgPlcOverrideDev

Note: The feed rate override also applies to rapid traverse if the rapid traverse override is not active.

The percentage adjusted with the feed rate override is entered by the NC in **NN_ChnFeedOverrideInput** and **PP_ChnFeedOverride.**

You can change the percentage through the PLC:

- Enter the desired percentage in **PP_ChnFeedOverride.** The NC immediately uses the new value.

PLC operand	Type
NN_ChnFeedOverrideInput **Feed rate override entered [%]**	D
PP_ChnFeedOverride **Feed rate override entered by PLC [%]**	D

Rapid Traverse Override

Settings in the configuration editor:	
NCchannel ChannelSettings Key for channel CfgPlcOverrideR **minimal** **maximal** **source**	

In the channel-sensitive parameter object **CfgPlcOverrideR,** create a parameter block for each machining channel (slide) to which a rapid traverse override is to apply.

MP_minimal
Minimum value for override
Format: Numerical value
Input: 0 to 100 [%]
Default: 0

MP_maximal
Maximum value for override
Format: Numerical value
Input: 0 to 200 [%]
Default: 150

MP_source
Source for override values
Format: String
Input: Key name for override device from **CfgPlcOverrideDev**

You can change the percentage through the PLC:

- Enter the desired percentage in **PP_ChnRapidFeedOverride.** The NC immediately uses the new value.

PLC operand	Type
NN_ChnRapidFeedOverrideInput **Rapid traverse override defined [%]**	D
PP_ChnRapidFeedOverride **Rapid traverse override entered by the PLC [%]**	D

PLC Inputs/Outputs

The MC 400 provides you with switching inputs/outputs for the PLC. If the available number of I/O is not enough, you can add up to two IEB 404 (I/O Exp Base Module, 4-Slots) PLC input/output units.

	MC 400		
	X8	**X41**	**X42**
Switching inputs 24 Vdc	–	–	56
Switching outputs 24 Vdc	–	31	–
Analog nominal value outputs 10 V–	12[a])	–	–
Control-is-ready output	–	2	–
Control-is-ready input	–	–	2

a. You need one analog output for each analog axis.

	IEB 404 PLC Input/Output Unit	
	IEM 16-8	**IEM 4-4A**
Switching inputs 24 Vdc	16	–
Switching outputs 24 Vdc	8	–
Analog inputs 10 Vdc	–	4
Inputs for Pt 100 thermistors	–	4
Analog outputs 10 Vdc	–	–
Control-is-ready output	(1)	–
Control-is-ready input	–	–

To interrogate and set the inputs and outputs of the PLC I/O unit, you need PLC modules.

The following topics are described:

- **Diagnosis of the Programmable Logic (PL)**
- **24 VDC Switching Input/Outputs**
- **Analog Inputs**
- **Analog Outputs**

Diagnosis of the Programmable Logic (PL)

The following topics are described:

- **Module 9007 Read the diagnostic information of a PLC input/output unit**
- **Module 9137 Read diagnostic information of the IEB 404**
- **Module 9139 Monitoring functions for the IEB 404 PLC input/output units**

Module 9007 Read the diagnostic information of a PLC input/output unit

The module provides diagnostic information about the IEB 404. To save computing time, refrain from continuously calling this module.

Call:

PS	B/W/D/K	<>Number of the PLB 510 basic module (0 to 3)>
PS	B/W/D/K	<>Number of the information> 0: Reserved 1: Reserved 2: Reserved 3: Total number of IEB 404 on this MC 4: Reserved 5: Reserved
CM	9007	
PL	B/W/D/K	<>Diagnostic information> 0 to 4: Number of IEB 404

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Diagnostic information was read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid code
	2	Invalid PL module
	24	Module was called in a spawn job or submit job
	51	Diagnostic function cannot be read because IEB 404 system is running in modular mode.

Module 9137 Read diagnostic information of the IEB 404

The module provides diagnostic information about the IEB 404. To save computing time, refrain from continuously calling this module.

Call:

PS	B/W/D/K	<>Number of the PLB 510 basic module (0 to 3)>
PS	B/W/D/K	<>Number of the socket (0 to 3)>
PS	B/W/D/K	<>Number of the information> 0: Possible mode of operation (PL modular mode) 1: Active mode of operation 2: Reserved 3: Reserved 4: Basic module code 5: Status of the basic module 6: Module model in the socket 7: Reserved 8: Reserved 9: Status of the module in the socket 10: Logical status of the outputs of a IEM 16-8 11: Short-circuit of the outputs of a IEM 16-8 12: Number of connected IEB 404
CM	9137	
PL	W/D	<>Diagnosis information> Information no. 0: 0: “IEB 404” operating mode not possible operating mode, without new functions of the IEB 404 1: “IEB 404” operating mode possible Information no. 1: 0: “PL 4xxB” operating mode active (without new functions of the IEB 404 1: “IEB 404” operating mode active Information no. 2: 0: No PLB 510 1: PLB 510 Information no. 3: 0–15: Hardware version Information no. 4: 0–15: Hardware code Information no. 5: Bit 0=1: Power supply of the PLB 510 is OK Bits 1 to 15: Reserved Information no. 6: 0: No module in the slot 1: Reserved

2: IEM 16-8
3: IEM 4-4A in the slot

Information no. 7:
0–15: Hardware version (identifies the function status of the module)

Information no. 8:
0–15: Hardware code (identifies a hardware change state)

Information no. 9:
IEM 16-8 (module type 2):
Bit 0=1: Power supply outputs 0 to 3 are OK
Bit 1=1: Power supply outputs 4 to 7 are OK
Bit 2=1: Short circuit at an output
Bit 3 = 1: No load at least one output (< 300 mA)
Bits 4 to 6: No meaning
Bit 7=1: Output 7 is a programmable output (otherwise “control is ready”)
Bits 8 to 31: No meaning
IEM 4-4A (module type 3):
Bit 0=1: Power supply of the inputs is OK
Bits 1 to 31: No meaning

Information no. 10:
Bit 0: Status of output 0 (IEM 16-8)
to
Bit 7: Status of output 7 (IEM 16-8)

Information no. 11:
Bit 0: Short circuit at output 0 (IEM 16-8)
to
Bit 7: Short circuit at output 7 (IEM 16-8)
Bit 8: No load (< 300 mA) at output 0 (IEM 16-8)
to
Bit 15: No load (< 300 mA) at output 7 (IEM 16-8)

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Diagnostic information was read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid code
	2	Invalid basic module number or slot number
	24	Module was called in a spawn job or submit job
	51	Function not possible since no IEB 404 system is connected or the I/O module type is incorrect for the desired type of information.

Explanations:

- **Hardware version** – identifies the function status of the module. When a function that affects the software is changed, the code number identifying the version is increased by one. Modules with low version numbers cannot be replaced by modules with higher version numbers.
- **Hardware code** – identifies the hardware change state. The hardware changes do not affect the functions. It is not necessary to take the hardware code into account when a module is replaced.
- **Short circuit:** Short circuit codes (both the group signal as well as the output-specific messages) are modal. They are indicated by the error LED on the I/O module. In the event of a short circuit, the affected output is automatically reset. With Module 9139 you can withdraw the short-circuit code and then drive the output again.
- **No-load operation:** The limit values for no-load detection are 20 mA (minimum) and 300 mA (maximum).

Module 9139 Monitoring functions for the IEB 404 PLC input/output units

The short circuit of an output of the IEM 16-8 is shown by an LED and the output is reset. Short-circuit monitoring remains in place, and must therefore be reset with Module 9139.

To save computing time, refrain from repeatedly calling this module.

Call:

PS	B/W/D/K	<>Function>
		0: Reserved
		1: Reserved
		2: Reset short-circuit monitoring
CM	9139	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Short-circuit monitoring was reset
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid code
	2	Invalid basic module number or slot number
	24	Module was called in a spawn job or submit job
	51	Function not possible or not a IEB 404

24 VDC Switching Input/Outputs

In PLC addresses you can find the current conditions of the switching inputs and outputs.

For the current states of the inputs/outputs of the PLC:

- Read all inputs with Module 9002.
- Update all outputs with Module 9005.
- With Module 9004 you can evaluate the rising or falling edge of the PLC inputs.

Note: Before the PLC program is converted, the PLC outputs are reset. In addition, the memory of the PLC outputs is reset.
During a loss of power (power fail), the control tries to reset the PLC outputs.

The following topics are described:

- **Module 9002 Read the inputs of a PLC input/output unit**
- **Module 9005 Set the outputs of PLC input/output unit**
- **Module 9004 Read the edges of PLC inputs**

Module 9002 Read the inputs of a PLC input/output unit

The module downloads the current states of the PLC input/output unit. In the PLC addresses (process image) you can read the current states of the PLC input/output unit. The contents of the PLC addresses remain unchanged until you call this module again.

For the IEB 404, inputs of empty sockets are not read.

The program can be called only in the cyclic PLC program.

Call:

PS	B/W/D/K	<>Number of the PLB 510 basic module (0 to 3)>
CM	9002	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Inputs were read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid PL number
	24	Module was called in a spawn job or submit job

Module 9005 Set the outputs of PLC input/output unit

The module overwrites the outputs of the PLC input/output unit with the values from the PLC addresses. The outputs are set or reset immediately at the time of module execution and remain in their condition until they are set or reset again by this module.

For the IEB 404, the outputs of empty sockets are not overwritten with values from the PLC addresses.

The program can be called only in the cyclic PLC program.

Call:

PS	B/W/D/K	<>Number of the PLB 510 basic module (0 to 3)>
CM	9005	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Outputs were set
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid PL number
	24	Module was called in a spawn job or submit job

Module 9004 Read the edges of PLC inputs

The module sets, upon falling or rising edges of the PLC inputs, specified end markers or bits in the specified byte range. Changes in the inputs are recognized only if a change also occurs in the PLC addresses (see **Module 9002**).

Ensure that the specified edge markers or edge bytes are in an unoccupied area. The edge bytes are written beginning with the least significant bit. Superfluous bits are erased.

Call:

PS	B/W/D/K	<>Number of the first PLC input>
PS	B/W/D/K	<>Number of the first edge marker or edge byte>
PS	B/W/D/K	<>Number of PLC inputs>
PS	B/W/D/K	<>Edge evaluation> 0: Rising edge. Entry in edge marker 1: Falling edge. Entry in edge marker 2: Rising edge. Entry in edge byte 3: Falling edge. Entry in edge byte
CM	9004	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Outputs were set
	1	Invalid transfer parameter

Analog Inputs

Settings in the configuration editor:	
System PLC CfgPlcPeriphery **pt100Discrete**	

The following topic is described:

- **Transferring the Analog Inputs of the IEB 404**

Transferring the Analog Inputs of the IEB 404

Module 9138 Read analog input of the IEB 404

The module transfers the current value of the given analog input of the PLB 510.

Value range ±10 Vdc input: –10 V to +10 V, at a resolution of 0.01 V
Value range Pt 100 input: 0 to 100 °C, at a resolution of 0.01 °C

To save computing time, refrain from repeatedly calling this module. The module can only be called in the cyclic PLC program.

Call:

PS	B/W/D/K	<>Number of the PLB 510 basic module (0 to 3)>
PS	B/W/D/K	<>Number of the socket (0 to 3)>
PS	B/W/D/K	<>Number of the analog input (0 to 7)>
CM	9138	
PL	B/W	<>Analog value> Analog inputs 0 to 3: Natural number (–1 000 to +1 000) in steps of 0.01 °C Analog inputs 4 to 7: Natural number (0 to 10 000) in steps of 0.01 °C

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Input was read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid basic module number or slot number
	24	Module was called in a spawn job or submit job
	51	Function not possible or not a IEB 404 or IEM 4-4A analog module

In MP_pt100Discrete, you define whether the values of the Pt 100 inputs are transferred immediately or with a change rate of 1 K/s. The disadvantage of transfer with a change rate of 1 K/s is that at high change rates it may take some time until the correct temperature has been transferred. For example, it would take 30 seconds to correctly read a temperature change of 30 K. An advantage of this, however, is a low sensitivity to disturbance: the temperature display will not jump back and forth between two values:

- If you wish to transfer the values of the Pt 100 inputs immediately, set MP_pt100Discrete=True.
- If you wish to work with a change rate of 1 K/s, set MP_pt100Discrete=False.

MP_pt100Discrete

Transfer of PT100 values

Format: Pull-down selection menu

Selection:

[True]

Transfer value immediately

[False]

Transfer value at 1 K/s

Default: True

Analog Outputs

You can drive analog outputs 1 to 12 at sockets X8 and X9.

Note: Every analog axis or analog spindle needs an analog output. These outputs are no longer available to the PLC.

The following topic is described:

- **Module 9130 Output analog voltage**

Module 9130 Output analog voltage

The module places an analog voltage on an analog output. The voltage is output with a slight delay after the end of the PLC scan.

Call the module only once for each output per PLC scan!

Format: 1 mV
Voltages greater than +10 V or less than –10 V are limited to the respective maximum value.

Call:

PS	B/W/D/K	<>Number of the analog output>
		1 to 6: Analog outputs 1 to 6 (X8)
PS	B/W/D/K	<>Analog voltage in mV>
CM	9130	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Analog voltage was output
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	1	Invalid analog output
	2	Disabled analog output

Incremental Jog Positioning

- The “incremental jog positioning” function is switched on and off with the **INCREMENT OFF/ON** soft key.
- To position with incremental jog, press the axis-direction keys (PP_AxTraversePos/ PP_AxTraverseNeg).

With marker NN_OmgJogIncrement you can interrogate the current state (see “**Modes of Operation**”).

Use Module 9036 (function 10) to limit the jog increment.

You can ascertain the current jog increment with Module 9035 (function 26).

With Module 9186 you can switch the incremental jog function on and off through the PLC (see “**Module 9186 Call a soft-key function**”).

The following topics are described:

- **Module 9036 Write NC status information**
- **Module 9035 Read NC status information**

Module 9036 Write NC status information

The module modifies status information from the NC. The status information to be modified is transferred by function number.

Call parameter:

Number of the function	Function	Value
10	Limit value for jog increment	0.0001–50 mm or –1: Cancel the limiting 2: New jog increment = minimum (programmed jog increment, previous limit value), and cancel limitation

Call:

PS	B/W/D/K	<Number of the function>
PS	B/W/D/K	<Value to be written>
CM	9036	
PL	B/W/D	<Error code>
		0: Status written
		1: Incorrect status code
		2: Transferred value is out of range
		3: Input disabled

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Status information was written
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Transferred value is out of range
	2	Incorrect number of the status information
	3	Transferred value is out of range
	6	Input disabled

Module 9035 Read NC status information

The module reads status information from the NC. The status information to be read is transferred by function number.

- The Read assigned handwheel axis function finds the axis which is assigned to the handwheel connected to connector X23 of the MC.

Call parameter:

Number of the function	Function
9	Read assigned handwheel axis

Call:

PS	B/W/D/K	<>Number of the function>
CM	9035	
PL	B/W/D	<>Jog increment> -1: None or more than one axis assigned 0: X axis 1: Y axis 2: Z axis 3: IV axis 4: V axis 5: VI axis 6: VII axis etc.

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Status information invalid
	20	Call was not in a submit or spawn job

Operating Times and System Times

The following topics are described:

- **Measuring Operating Times**
- **System Time**

Measuring Operating Times

Settings in the configuration editor:	
System CfgPlcOperTimes **displayPlcTimes** **resetPlcTimes** **resetNcTimes** **textNumber**	

The control can measure up to 11 operating times (operating hours counter) and store them in a file:

Operating time	Meaning
TNCTIME	Control on
MACHINETIME	Machine on (NC operating time)
PROGTIME	Program run (NC operating time)
PLCTIME0 to PLCTIME7	Freely definable PLC operating time

For all operating modes except **Programming** and **Test Run:**

- Press the MOD key and press the **MACHINE TIME** soft key.
- In MP_resetPlcTimes, specify the PLC operating times you can reset with the code number 857282.
- In MP_resetNcTimes, specify the NC operating times you can reset with the code number 857282.
- In MP_displayPlcTimes, define the operating times you want to display.
- In MP_textNumber, define the dialog messages to be displayed for the individual PLC operating times.

The time is measured in seconds and is updated every minute during the run time. When the control is switched off, no more than one minute is lost.

The NC starts and stops the NC operating hours counter (**Control on, Machine on**, and **Program run**).

PLC operating hours counters 1 to 8:

- Start with Module 9190.
- Stop with Module 9191.
- All operating times are saved during a hard-disk backup.
- Use the following modules to evaluate or change the operating times
- Module 9190 Start the PLC operating hours counter
- Module 9191 Stop the PLC operating hours counter

- Module 9192 Transfer the operating hours counter
- Module 9193 Set the operating hours counter
- Module 9194 Alarm when operating times are exceeded

Note: When measuring the operating time in the Program Run mode, the different operating mode groups are currently not distinguished.

MP_displayPlcTimes
Display PLC operating times
Format: Numerical value
Input: Binary value
Bits 0 to 7 represent PLC operating times 1 to 8
0: Do not display
1: Display
Default: %11111111

MP_resetPlcTimes
Reset PLC operating times with the code number
Format: Numerical value
Input: Binary value
Bits 0 to 7 represent PLC operating times 1 to 8
0: Do not reset
1: Reset
Default: %00000000

MP_resetNcTimes
Reset NC operating times with the code number
Format: Numerical value
Input: Binary value
Bit 0: “Control on” operating time
Bit 1: “Control on” operating time
Bit 2: “Program run” operating time
0: Do not reset
1: Reset
Default: %000

The dialog texts for the PLC operating hours counter are defined in the text file %OEM%\plc\language\en (or languages other than English).

MP_textNumber
Dialogs for PLC operating times
Format: Array [7]
Input: [0] corresponds to the text for PLC time 1, etc.
In the text file for PLC dialogs, indicate the line number of the dialog text (%OEM%\plc\language\en (or language other than English)).

The following topics are described:

- **Module 9190 Start the PLC operating hours counter**
- **Module 9191 Stop the PLC operating hours counter**
- **Module 9192 Transfer the operating hours counter**
- **Module 9193 Set the operating hours counter**
- **Module 9194 Alarm when operating time exceeded**

Module 9190 Start the PLC operating hours counter

The module activates one or more operating hours counters. The control measures the time elapsing after activation. Use Module 9191 to cancel activation. Bits 0 to 7 of the transfer value represent the PLC operating hours counters 1 to 8. For every bit set, the associated counter is started.

Call:

PS	B/W/D/K	<>PLC operating time> Bits 0 to 7 represent PLC operating times 1 to 8
CM	9190	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	PLC operating time started
	1	Incorrect parameter

Module 9191 Stop the PLC operating hours counter

The module stops one or more PLC operating hours counters. The time measured by the control is saved. Bits 0 to 7 of the transfer value represent the PLC operating hours counters 1 to 8. For every bit set, the associated counter is stopped.

Call:

PS	B/W/D/K	<>PLC operating time> Bits 0 to 7 represent PLC operating times 1 to 8
CM	9191	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	PLC operating time stopped
	1	Incorrect parameter

Module 9192 Transfer the operating hours counter

The module reads the given PLC or NC operating hours counter.

The current value is transferred in seconds. If the value is greater than 2 147 483 648 (approx. 69 years), a negative number will be transferred.

Call:

PS	B/W/D/K	<>Number of the operating time> –3: **Control on** –2: **Machine on** –1: **Program run** 0 to 7: PLC operating hours counters 1 to 8
CM	9192	
PL	B/W/D	<>Current time [s]> –1: Error

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	PLC operating time was read
	1	Incorrect transfer value, or module was not called in a spawn job or submit job

Module 9193 Set the operating hours counter

The module overwrites the given PLC or NC operating hours counter. The old value is lost irretrievably.

The value of the NC operating hours counters may only be changed in exceptional cases (e.g., when the control is exchanged).

The time for **Control on** cannot be overwritten.

Transfer all values greater than 2 147 483 648 (approx. 69 years), as negative numbers.

Call:

PS	B/W/D/K	<>Number of the operating time> –2: **Machine on** –1: **Program run** 0 to 7: PLC operating times 1 to 8
PS	B/W/D/K	<>New time [s]>
CM	9193	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Operating time was overwritten
	1	Incorrect transfer value, or module was not called in a spawn job or submit job

Module 9194 Alarm when operating time exceeded

The module activates a monitoring function in the NC, which sets a PLC marker when the given maximum time for a PLC or NC operating hours counter is exceeded. The marker is set the first time the maximum time is exceeded, and then cyclically once per minute.

The marker can be delayed by max. 59 s the first time it is set. All values greater than 2 147 483 648 (approx. 69 years) must be transferred as negative numbers.

If you enter the value zero as the alarm threshold, the function is deactivated.

Call only in a submit job or spawn job.

Call:

PS	B/W/D/K	<>Number of the operating time> –3: **Control on** –2: **Machine on** –1: **Program run** 0 to 7: PLC operating times 1 to 8
PS	B/W/D/K	<>Alarm threshold [s]>
PS	B/W/D/K	<>Number of the alarm markers>
CM	9194	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Alarm function activated
	1	Incorrect transfer value, or module was not called in a spawn job or submit job

System Time

Settings in the configuration editor:	
System CfgSystemTime **offsetToUTC**	

System time management varies depending on the control system:

- Single-processor systems internally operate with UNIX system time. This is the number of seconds since 0:00 hours on January 1, 1970. The parameter MP_offsetToUTC defines the time difference between Universal Time (Greenwich time) and local time. It is the user's task to adjust between Daylight Saving Time and Standard Time. Daylight Saving Time or Standard Time can be set in MP_offsetToUTC.
- For dual-processor systems, the Windows operating system provides the system time. Windows®**1 automatically adjusts for Daylight Saving Time or Standard Time.

Use the following modules to transfer the system time:

- Module 9195 Transfer the real-time clock (UNIX system time)
- Module 9055 Convert time (binary) to formatted string (and consider MP_offsetToUTC)

- **Single-processor systems:** In MP_offsetToUTC, define the time difference between Universal Time and local time, taking into account Daylight Saving Time or Standard Time.
- **Dual-processor systems:** Set MP_offsetToUTC=0.

MP_offsetToUTC

Time difference to universal time

Format: Numerical value

Input: –23 to +23 [hours]
0: Universal Time (Greenwich Mean Time)
1: Central European Time (CET)
2: Central European Daylight Time (CEDT)

Default: 1

The following topics are described:

- **Module 9195 Transfer the real-time clock**
- **Module 9055 Convert time (binary) to formatted string**

**1 Windows® is a registered trademark of Microsoft Corporation.

Module 9195 Transfer the real-time clock

The module reads the time of the real-time clock. A double word is returned, which contains the number of seconds accumulated since 0:00 hours on January 1, 1970 (UNIX system time).

Call:

CM	9195	
PL	D	<>System time>
		Number of seconds since 0:00 hours on January 1, 1970.

Module 9055 Convert time (binary) to formatted string

The module provides the date and time (local time) in an ASCII string with configurable format.

The module converts the binary UNIX system time (number of seconds since 0:00 hours on January 1, 1970) to an ASCII string, taking into account the time difference between local time and Universal Time (Greenwich time) defined in MP_offsetToUTC.

Call:

PS	B/W/D/K	<>System time>

Number of seconds since 0:00 hours on January 1, 1970.

PS	B/W/D/K	<>String number for the result>
PS	B/W/D/K	<>Format>

0: DD.MM.YYYY hh:mm:ss
1: D.MM.YYYY h:mm:ss
2: D.MM.YYYY h:mm
3: D.MM.YY h:mm
4: YYYY-MM-DD- hh:mm:ss
5: YYYY-MM-DD- hh:mm
6: YYYY-MM-DD h:mm
7: YY-MM-DD- h:mm
8: DD.MM.YYYY
9: D.MM.YYYY
10: D.MM.YY
11: YYYY-MM-DD
12: YY-MM-DD
13: hh:mm:ss
14: h:mm:ss
15: h:mm

CM 9055

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	String was created
	1	Incorrect transfer value

Tool Changer

You control the tool changer through PLC outputs.

If the tool changer is to be driven by controlled axes, use PLC axes see “**Section 5, Controlling Axes by PLC (PLC Axes)**”. You can also control the tool changer through proximity switches:

- Save the information about the tool in the tool table and the information about the tool changer in the pocket table.

The following topic is described:

- **Tool and Pocket Number**

Tool and Pocket Number

You can edit the tool table in the machining modes of operation:

- Press the **TOOL TABLE** soft key.

From the tool table you can call the pocket table (see the *6000I CNC User’s Manual,* P/N 627785-2X):

- Press the **POCKET TABLE** soft key.

Usually, the current tool table is TOOL.T, the pocket table is TOOL_P.TCH. Both files are saved in the root directory TNC:\TABLE\.

Definition of the tool and pocket table:

- In the machine configuration, you define the columns to be contained in the tool table and pocket table. You also define the sequence in which the columns are to be displayed in the tables.
- Example of tool table: First, you define the table columns to be contained in the tool table.
- In the machine configuration, you define the names and properties of each individual table column. Under
  System/TableSettings/Columns/T.
  ...every column of the tool table is displayed as a separate subfolder. The name of the subfolder determines the column name. It is preceded by the file extension (.T) followed by a period.

Settings in the configuration editor:	
System TableSettings Columns T. **T.ANGLE** CfgColumnDescription CfgColumnText **T.CUR_TIME** CfgColumnDescription CfgColumnText **T.CUT** CfgColumnDescription CfgColumnText . . .	

- Then you have to define the sequence in which the individual table columns you have created should display in the tool table. You define the settings in: **System/TableSettings/CfgTableProperties/T/columnKeys.**

Settings in the configuration editor:	
System TableSettings CfgTableProperties T columnKeys **T.T** **T.NAME** **T.L** **T.R** **T.R** . . . primaryKey	

- The machine parameter **MP_primaryKey** indicates the primary key of the table. The data is sorted in ascending order according to the column specified here. Here, you must enter T.T for the tool table (for column “T”, tool number).

The PLC chapter “**Section 7, Tables**” describes in detail how you can configure tables, add or remove columns, and define the names and sequence of columns.

The following topics are described:

- **Elements of the Tool Table**
- **Elements of the Pocket Table**

Elements of the Tool Table

In the basic configuration of the control (factory default setting), the tool table contains the following columns:

Column name	Meaning	Column width
T	Tool number	8
NAME	16-character alphanumeric tool name	16
L	Tool length	12
R	Tool radius	12
R2	Tool radius 2 for toroidal cutter	12
DL	Oversize in tool length	9
DR	Oversize in tool radius	9
DR2	Oversize in tool radius 2	9
TL	Locked tool?	3
RT	Replacement tool	8
TIME1	Maximum tool age	6
TIME2	Maximum tool age TOOL CALL	6
CUR_TIME	Current tool age	9
TYPE	Tool type (MILL=cutter/DRILL=drill)	5
DOC	Comment on the tool	16
PLC	Additional information for PLC (Module 9093)	10
LCUTS	Tooth length	12
ANGLE	Plunge angle	8
CUT	Number of tool teeth	4
LTOL	Wear tolerance for tool length	7
RTOL	Wear tolerance for tool radius	7
DIRECT	Cutting direction of the tool	7
R–OFFS	Tool offset: Radius	12
L–OFFS	Tool offset: Length	12
LBREAK	Breakage tolerance for tool length	7
RBREAK	Breakage tolerance for tool radius	7
LIFTOFF	Retract tool	8

Elements of the Pocket Table

In the basic configuration of the control (factory default setting), the pocket table contains the following columns:

Column name	Meaning	Column width
P	Pocket number	8
T	Tool number	8
TNAME	Tool name	16
ST	Special tool	3
F	Fixed pocket	3
L	Pocket locked	3
PLC	PLC status	10

With Modules 9092, 9093, 9094, and 9096 you can read the tool and pocket tables and overwrite them.

If an input field is open in the editor at the time the modules are called, this field is closed automatically.

In the status display, press the **STATUS TOOL** soft key to display the active tool data.

The following topics are described:

- **Module 9092 Search for an entry in the tables selected for execution (.T/.D/.TCH)**
- **Module 9093 Read data from tables selected for program (.T/.D/.TCH)**
- **Module 9094 Write data into a tool and datum table**
- **Module 9096 Delete a line in the tool table**
- **Special Tools**
- **Tool Life, Replacement Tool**
- **Indexed Tools**

Module 9092 Search for an entry in the tables selected for execution (.T/.D/.TCH)

Prerequisite for table: M status must be set.

The entry or value sought is given as a natural number, shifted by the number of decimal places that can be entered.

As return code the function replies with the number of the line in which the value was found.

It is possible, for example, to look for the vacant pocket (corresponds to T0) in the pocket table.

If you wish to look for more occurrences of the same value, you must enter the line number of the last occurrence plus one as the starting line.

Call:

PS	B/W/D/K	<>File type>
		0: *.T file (tool table)
		1: *.D file (datum table)
		2: *.TCH file (pocket table)
PS	B/W/D/K	<>Element value>
PS	B/W/D/K	<>Element number>
		***.T file**
		0: Tool length (L)
		1: Tool radius (R)
		2: Reserved
		3: Replacement tool (RT); (–1= not defined)
		4: Reserved
		5: TIME 1
		6: TIME 2
		7: CURRENT TIME
		8: Tool radius 2 (R2)
		9: Oversize for tool length (DL)
		10: Oversize for tool radius (DR)
		11: Oversize for tool radius 2 (DR2)
		12: Tool locked (TL); (0: No, 1: Yes)
		13: Number of the tool teeth (CUT)
		14: Wear tolerance for tool length (LTOL)
		15: Wear tolerance for tool radius (RTOL)
		16: Cutting direction of the tool (DIRECT); (0:+; 1: –)
		17: PLC status (PLC)
		18: Tool offset for tool length (TT:LOFFS)
		19: Tool offset for radius (TT:ROFFS); ($7FFF FFFF = R)
		20: Breakage tolerance for tool length (LBREAK)
		21: Breakage tolerance for tool radius (RBREAK)
		22: Tooth length (LCUTS)
		23: Plunge angle (ANGLE)

		24: Tool number 25: Reserved 26: PLC value (PLC-VAL) 27: Probe center offset in reference axis (CAL-OF1) 28: Probe center offset in minor axis (CAL-OF1) 29: Spindle angle during calibration (CAL-ANG) ***.D file:** 0: Shift in axis 1 1: Shift in axis 2 2: Shift in axis 3 3: Shift in axis 4 4: Shift in axis 5 5: Shift in axis 6 6: Shift in axis 7 7: Shift in axis 8 8: Shift in axis 9 ***.TCH file:** 0: Tool number (T); (–1, if no tool is entered) 1: Special tool (ST); (0: no, 1 = yes) 2: Fixed pocket (F); (0: no, 1 = yes) 3: Locked pocket (L); (0: no, 1 = yes) 4: PLC status (PLC)
PS	B/W/D/K	<>Line number for beginning of search>
CM	9092	
PL	B/W/D	<>Line number (in case of error –1)>
PL	B/W/D	<>Error number> 0: No error. Element was found. 1: Call was not in a submit or spawn job 2: File type does not exist 3: No file of the entered type was found with M status 4: Line number not in file 5: Incorrect element number 6: Element value not found

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	No errors
	1	See above for errors

Module 9093 Read data from tables selected for program (.T/.D/.TCH)

Prerequisite for table: M status must be set.

You transfer the line number (i.e., tool number for *.T, vector number for *.D or pocket number for *.TCH) and the number of the element to be read.

The value is given as a natural number, shifted by the number of decimal places that can be entered.

The module must be called in a submit job or spawn job.

Call:

PS	B/W/D/K	<>File type (see **Module 9092**)>
PS	B/W/D/K	<>Line number>
PS	B/W/D/K	<>Element number (see **Module 9092**)>
CM	9093	
PS	B/W/D	<>Element value>
PL	B/W/D	<>Error number>
		0: No error
		1: Call was not in a submit job
		2: File type does not exist
		3: No file of the entered type was found with M status
		4: Line number not in file
		5: Incorrect element number

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	No errors
	1	See above for errors

Module 9094 Write data into a tool and datum table

Prerequisite for table: M status must be set.

You transfer the line number and the element number of the element to be overwritten.

The value is given as a natural number, shifted by the number of decimal places that can be entered.

The execution of Module 9094 reinitializes the geometry.

The module must be called in a submit job or spawn job.

Call:

PS	B/W/D/K	<>File type (see **Module 9092**)>
PS	B/W/D/K	<>Line number>
PS	B/W/D/K	<>Element number (see **Module 9092**)>
PS	B/W/D/K	<>Element value>
CM	9094	
PL	B/W/D	<>Error number> 0: No error. Element was written. 1: Call was not in a submit or spawn job 2: File type does not exist 3: No file of the entered type was found with M status 4: Line number not in file 5: Incorrect element number 6: Element value is outside the permissible range

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	No errors
	1	See above for errors

Module 9096 Delete a line in the tool table

You remove a line from the tool table and cancel any link with a replacement tool.

The module must be called in a submit job or spawn job.

Call:

PS	B/W/D/K	<>Tool number / pocket number>
PS	B/W/D/K	<>Mode> Bit 0: Delete entries in pocket table 0: Pocket table remains unchanged 1: Tool number in pocket table is deleted Bit 1: Tool or pocket number 0: Transferred value = tool number 1: Transferred value = pocket number
CM	9096	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Line was deleted
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid pocket or tool number
	20	Module was not called in a submit job or spawn job
	36	File error

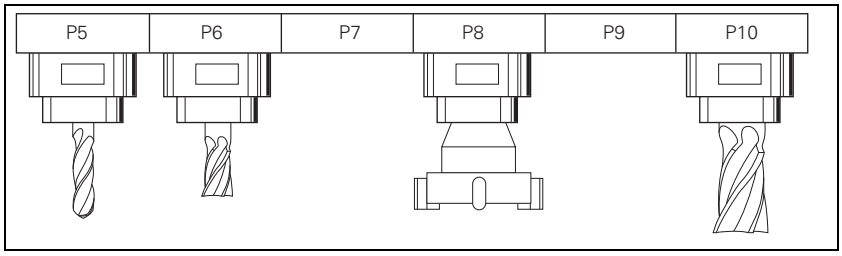
Special Tools

In the pocket table:

In the column **ST** you define tools as special tools.

For oversized special tools:

- Leave a pocket free in the tool magazine on both sides of the pocket (see illustration).
- In the column **L** you lock pockets that are to remain empty.



In the column **F** (fixed pocket), you can define individual tools which are to be returned to their original pockets, although variable pocket coding was selected.

Tool Life, Replacement Tool

You can enter two tool life values (**TIME1** and **TIME2**) and one replacement tool (**RT**) for each tool in the tool table.

For the **TOOL CALL:**

- **CUR.TIME** (current tool age) > **TIME1:** The NC sets the marker **NN_ChnToolLifeExpired**.

You decide in the PLC what should happen when **NN_ChnToolLifeExpired** is set (e.g., display a PLC error message).

With M101, activate the automatic insertion of the replacement tool after expiration of the tool age (**TIME1** or **TIME2**). With M102, deactivate the insertion. The tool is not changed immediately after expiration of the tool life, but rather it varies depending on the processor load.

Note: In standard NC programs (NC block with **RR, RL**, or **R0**), the same radius must be defined for the replacement tool as for the original tool.

No radius compensation is given in NC blocks with normal vectors. One delta value for tool length and radius (**DR** and **DL**) can be entered for each tool in the tool table. These delta values are taken into account by the control.

If the radius of the replacement tool differs from the original tool, you must define this in the **DR** column. The delta value must always be negative. If you enter a positive delta value, the error message **Tool radius too large** is displayed.
You can suppress this error message with the M function M107, and reactivate it with M108.

PLC operand	Type
NN_ChnToolLifeExpired **Tool life 1 has expired**	M

The current tool age is calculated in the **Program Run, Single Block** and **Program Run, Full Sequence** operating modes if the following conditions are fulfilled.

- Spindle ON
- No F MAX
- F enable
- Control in operation (*)

After program interruption with the **INTERNAL STOP** soft key, M02, M30, or **END PGM,** the tool age counter is stopped.

The tool age counter does not run in the **Manual, Electronic Handwheel**, and **Positioning with MDI** operating modes.

The user can reset the current tool age by entering zero.

Indexed Tools

You can also work with indexed tools in the tool table (e.g., when you use a stepped drill with more than one length compensation value). For indexed tools, the tool number is given an index (e.g., 1.1).

If you are working with indexed tools and wish to use Modules 9092, 9093, or 9094, you must first find the line number of the tool, since these modules will need it.

- Use Module 9091 to determine the line number of a tool in the tool table.

Module 9091 Find the line number of a tool in the tool table

Call:

PS	B/W/D/K	<>Tool number>
PS	B/W/D/K	<>Tool index>
CM	9091	
PL	B/W/D	<>Line number>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Line number was found
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode	2	Invalid value for tool or tool index number
	20	Module was not called in a submit job or spawn job
	29	Tool table (TOOL.T) not found
	30	Tool number not found
	32	Tool index number not found

Commissioning

The following topics are described:

- **Preparation**
- **Adjusting the Servo Amplifier**
- **Commissioning the Axes**

Preparation

Proceed as follows:

- Check the wiring against the grounding diagram
- Check the control-is-ready function (see "**Section 5, EMERGENCY STOP Monitoring**")
- Check the EMERGENCY STOP circuit by pressing the EMERGENCY STOP buttons and the EMERGENCY STOP limit switch.
- Determine machine configuration using the documentation on hand.
  The machine configuration must contain a parameter block for every axis. The machine parameters are preassigned with initial values before commissioning.
- Create a PLC program for interfacing the control to the machine (use the PLC development software **PLCdesignNT**). A PLC basic program is available for the control. ANILAM recommends using the PLC basic program.
- Ensure that all drives are enabled by the PLC. Use for example OLM (see "**Actual Status 1 of the Axes (Ipo Act State 1)**" to check this before putting the machine into service).
- Before putting the machine into service, get familiar with the machine and the mechanical data of the individual axes.
- Ensure that the axes are located at noncritical positions and that they can be moved safely during adjustment.

The following topic is described:

- **Temporary Input Value**

Temporary Input Values

- Enter the following temporary input values when you begin commissioning:

Machine parameters in the configuration editor	Temporary input value	Meaning
Channels		
ChannelSettings		
Ch-Nc		
CfgLaPath		
maxPathJerk	1	Maximum jerk on the path
maxG1Feed	99.999	Max. machining feed rate
pathTolerance	0.01	Tolerance for contour transitions
Axes		
ParameterSets		
[Key for parameter block]		Parameter block of axis
CfgAxisHardware		
signCorrAcutalVal	off	Reverse counting direction of actual value
signCorrNominalVal	off	Reverse counting direction of nominal value
CfgPosControl		
kvFactor	0	k_v factor
servoLagMin1	20	Following error limit
servoLagMax1	20	Following error limit
servoLagMin2	20	Following error limit
servoLagMax2	20	Following error limit
feedForwardFactor	1	100% feedforward control
controlOutputLimit	1000	Control-variable limit for position controller
CfgFeedLimits		
maxAcceleration	0.5	Maximum permissible acceleration
CfgControllerAuxil		
driveOffLagMonitor	off	Following-error monitoring
checkPosStandstill	2	Standstill monitoring
CfgEncoderMonitor		
checkAbsolutPos	off	Distance code monitoring
checkSignalLevel	on	Movement monitoring threshold
movementThreshold	0	Tolerance for following error
CfgControllerTol		
posTolerance	0.01	Positioning window
CfgPositionFilter		
filter2Shape	HSC	Shape of 2nd nominal position value filter
manualFilterOrder	11	Order of the mean-value filter

Adjusting the Servo Amplifier

Please note:

Note: Adjust the servo amplifier before optimizing the position controller. For instructions on adjustment, refer to the information given by the manufacturer of your servo amplifier.

- Adjust the offset according to the information given by the drive manufacturer.
- Adjust the proportional (P) component and the integral-action (I) component of the speed controller at the servo amplifier.
- Check the polarity of the tachometer signal of the drive by using a battery box, for example.

ANILAM recommends:

Use a voltage of 9 V for rapid traverse to ensure optimum utilization of the voltage range of +/– 10 V and to attain optimum control loop performance for the axis. The axis velocity to be expected (in [mm/min]) is defined in machine parameter **MP_maxFeedAt9V**. Enter the rapid traverse rate in the machine parameter.

Note: A servo amplifier that has been adjusted according to the information given by the manufacturer is the basic prerequisite for putting the machine into service.

Commissioning the Axes

General information:

Analog axis feedback control is based on the following formula:

$$U_{out} = (P_{err} \cdot \mathbf{kvFactor} + \frac{V_{nom}}{60} \cdot \mathbf{feedForwardFactor} + 1000 \cdot A_{nom} \cdot \mathbf{accForwardFactor}) \cdot \frac{9V \cdot 60}{\mathbf{maxFeedAt9V}}$$

Value, parameter	Unit	Description
U_{out}	Volt	Output voltage (analog nominal speed value)
P_{err}	mm	Following error (servo lag)
kvFactor	1/s	Kv factor (proportional component of position controller)
V_{nom}	mm/min	Nominal velocity
feedForwardFactor		Factor for velocity feedforward control
A_{nom}	m/s^2	Nominal acceleration
accForwardFactor	(s)	Factor for acceleration feedforward control
maxFeedAt9V	mm/min	Assumed velocity of the axis at 9 V

The temporary input values result in the following reduced formula for the output voltage:

$$U_{out} = (P_{err} \cdot 0 + V_{nom} \cdot 1 + A_{nom} \cdot 0) \cdot \frac{9V}{\mathbf{maxFeedAt9V}}$$

Therefore:

$$U_{out} = V_{nom} \cdot \frac{9V}{\mathbf{maxFeedAt9V}}$$

Note: The temporary input values result in the axis only being speed feedback-controlled, and not position feedback-controlled. The resulting servo lag is not eliminated. For this reason, larger values were defined with the temporary input values in the **MP_servoLagMin1, MP_servoLagMin2,** MP_servoLagMax1, and MP_servoLagMax2 parameters.

Danger: Due to the temporary machine parameters, the position control loop is open at the beginning of commissioning!
Hanging axes require a 100% compensation for weight.
Make sure that hanging axes are adequately supported.

The following topics are described:

- **Check the Counting Direction**
- **Reversal of Traverse Direction**
- **Speed Adjustment**
- **Determining the Acceleration**
- **Determining the k_v Factor**
- **Determining the Jerk**
- **Determining Acceleration Feedforward Control**
- **Setting the Traverse Range**
- **Activating Monitoring Functions**
- **Hysteresis / Static Friction**

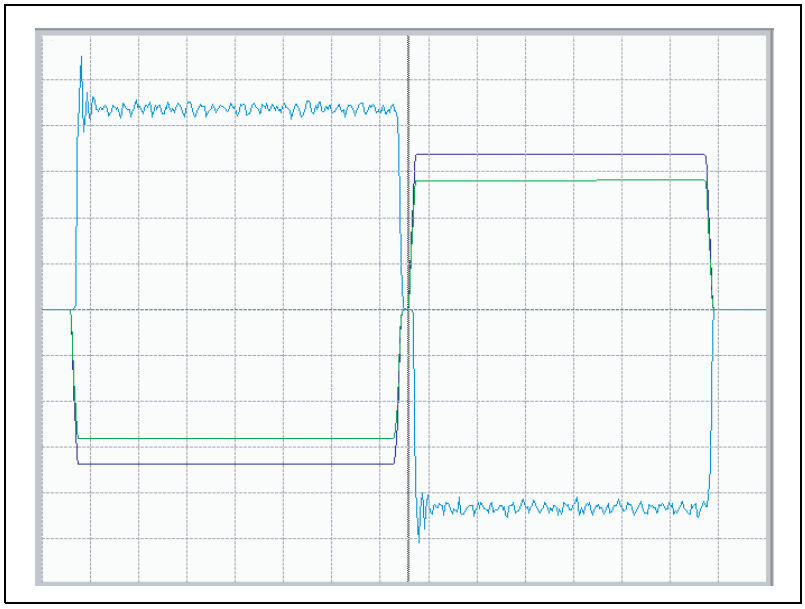
Check the Counting Direction

Settings in the configuration editor:	
Axes ParameterSets [Key for parameter block] CfgAxisHardware **signCorrActualVal** **signCorrNominalVal**	

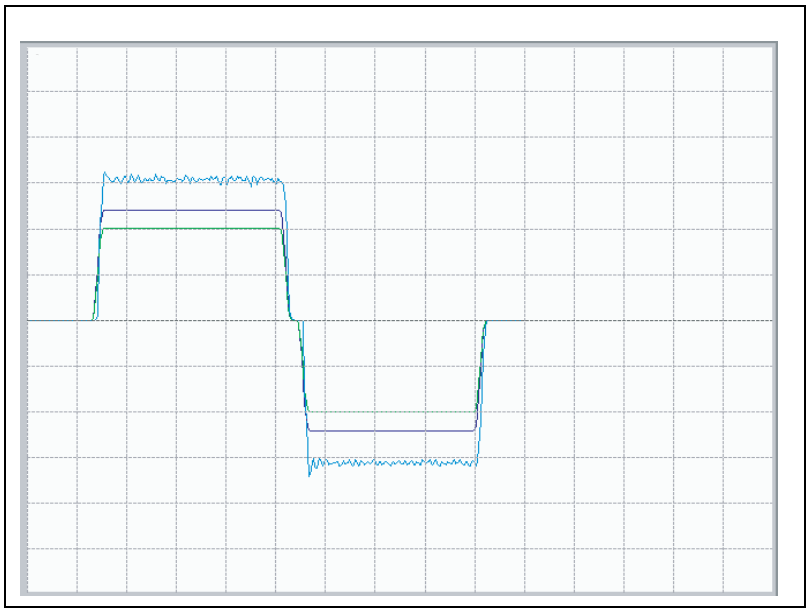
To check the counting direction of the position measuring system, proceed as follows:

- Switch on the machine.
- Select the following operating mode without crossing over the reference marks:
  Manual Operation
- Switch to the **Oscilloscope** mode of operation.
- Set the following values in the oscilloscope by pressing the **SELECTION** soft key:
  Display mode: YT
  Sampling time: IPO clock
  Channel 1: Analog
  Channel 2: V noml
  Channel 3: V actl
  Trigger: Free run
- Press the **OSCI** soft key to switch the curve representation.
- Press the **START** soft key to start recording.
- Press the axis-direction key of each axis to be checked.

- Press the **STOP** soft key to stop recording.



Incorrect Sign



Correct sign

- If **V nomI** and **V actI** do not lie in the same direction on the oscilloscope, you must change either **MP_signCorrActualVal** or **MP_signCorrNominalVal.**

Reversal of Traverse Direction

If the axis does not move in the expected direction after you have pressed the respective axis-direction key (e.g., X axis moves in negative direction although you have pressed the X+ key), you can reverse the traversing direction.

- Invert the two values entered in the parameters**MP_signCorrActualVal** and **MP_signCorrNominalVal.**

Speed Adjustment

Settings in the configuration editor:	
Axes ParameterSets [Key for parameter block] CfgAxisAnalog **maxFeedAt9V**	

The aim of speed adjustment is to achieve that the output nominal speed value is equal to the really measured actual speed value ($V_{nom} = V_{act}$).

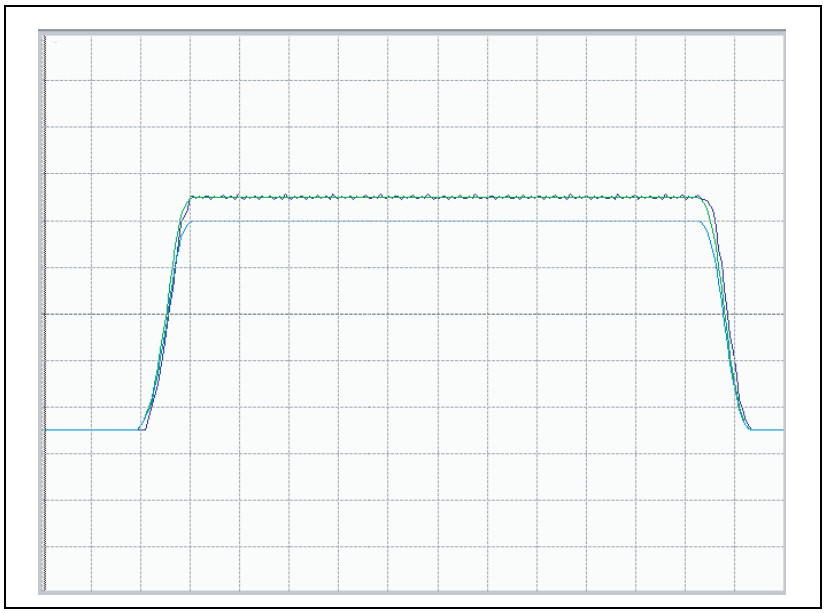
Determine whether the nominal speed value (V_{nom}) differs from the actual value (V_{act}) on the machine. Proceed as follows:

- Switch on the machine.
- Select the following operating mode without crossing over the reference marks: **Manual Operation**
- Switch to the **Oscilloscope** mode of operation (code number **688379**).
- Set the following values in the oscilloscope by pressing the **SELECTION** soft key:
  Display mode: YT
  Sampling time: IPO clock
  Channel 1: Analog
  Channel 2: V nom
  Channel 3: V act
  Trigger: Free run

Note: In the internal oscilloscope, the **analog** signal corresponds to the output voltage U_{out} (= analog speed command signal) at connector X8.

- Press the **OSCI** soft key to switch the curve representation.
- Press the **START** soft key to start recording.
- Press the axis-direction key of each axis to be checked.
- Press the **STOP** soft key to stop recording.
- Compare the values measured for V_{nom} and V_{act} with each other.

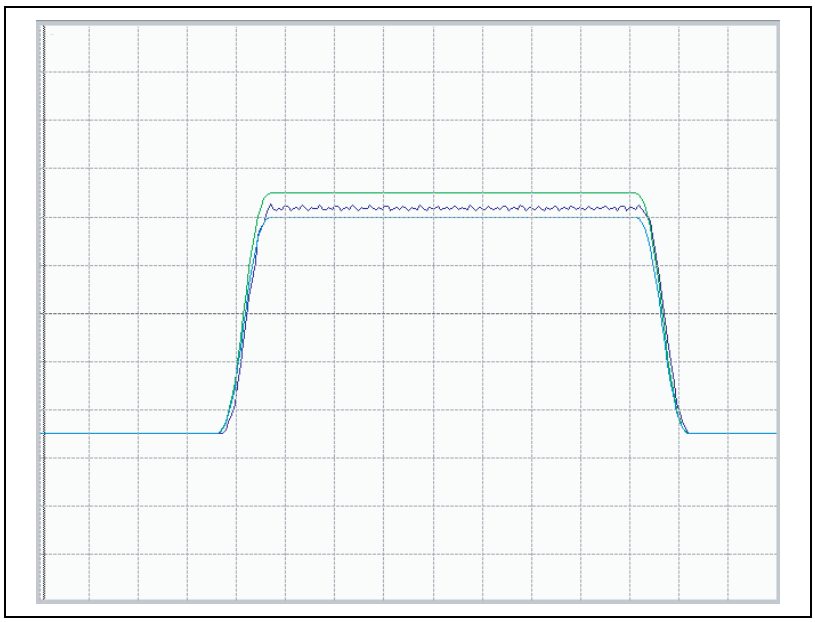
- Ideally, your oscilloscope measurement should look similar to this:



$V_{nom} = V_{act}$, Speed Adjusted Correctly

However, it may occur that the nominal value differs from the actual value.

:



V_{nom} ¼ V_{act}, Speed Adjusted Incorrectly

In this case, you should at first try to eliminate the difference by using the possible settings of the servo amplifier (please note the information given by the manufacturer). If this fails, refer to the information given below on how to adjust the value in **MP_maxFeedAt9V.**

Due to the temporary input values, the following formula applies for the nominal output voltage at connector X8:

$$U_{out} = V_{nom} \cdot \frac{9V}{\boldsymbol{maxFeedAt9V}}$$

Therefore, **MP_maxFeedAt9V** is determined as follows:

$$\boldsymbol{maxFeedAt9V} = V_{act} \cdot \frac{9V}{U_{out}}$$

Perform a measurement with the internal oscilloscope to determine the current difference between **MP_maxFeedAt9V** and the connected drive. Use the parameter formula described above to determine the correct value.

Proceed as follows:

- Switch to the **Oscilloscope** mode of operation.
- Set the following values in the oscilloscope by pressing the **SELECTION** soft key:
  Display mode: YT
  Sampling time: IPO clock
  Channel 1: V actl
  Channel 2: V noml
  Channel 3: Analog
  Trigger: Free run
- Press the **OSCI** soft key to switch the curve representation.
- Press the **START** soft key to start recording.
- Press the axis-direction key of each axis to be checked.
- Press the **STOP** soft key to stop recording.
- Select the values **V noml** and **Analog** by using the arrow keys and write down the measured values, which are displayed at the left side of the oscilloscope (**Cu1:**).
- Enter the displayed values in the formula for parameter **MP_maxFeedAt9V.**

Example:

The internal oscilloscope measured the following values on the machine:

- **Analog** = 1.21 V
- **V actl** = 1517 mm/min
- **V noml** = 1008 mm/min

This measurement makes clear that the actual speed value **V act** differs from the nominal speed value **V nom**. The difference can be eliminated by using the formula mentioned above:

$$\boldsymbol{maxFeedAt9V} = 1517\frac{mm}{min} \cdot \frac{9V}{1.21V} = 11283\frac{mm}{min}$$

- Enter the calculated value in the parameter **MP_maxFeedAt9V** and check the calculated value by performing a measurement with the internal oscilloscope.

Determining the Acceleration

- Clamp an object of maximum permissible weight on the machine table.

Note: Write down the current input values set in **MP_CfgPositionFilter**. You will need to enter these values again after the acceleration has been optimized.

- Now enter the temporary machine parameters listed in the table.:

Goal of the temporary input values: A jump in the nominal value is input to the axis.

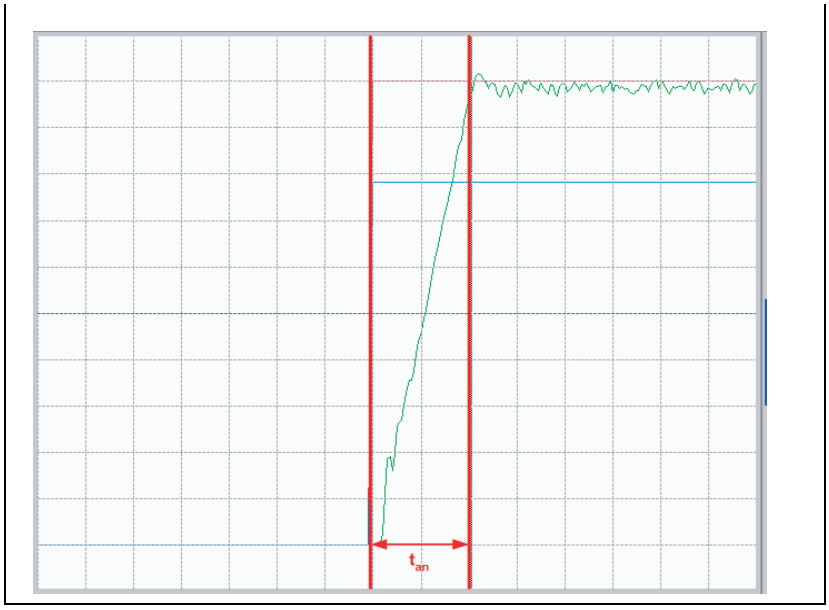
Machine parameters in the configuration editor	Temporary input value	Meaning
Axes		
ParameterSets		
[Key for parameter block]		
CfgPosControl		
kvFactor	0	k_V factor
CfgFeedLimits		
rapidFeed	Specific	Rapid traverse in manual operation
maxAcceleration	999999	Maximum acceleration
CfgPositionFilter		
filter2Shape	Off	Shape of 2nd nominal position value filter
manualFilterOrder	1	Order of mean-value filter in manual mode

Warning: Ensure that the transmitted nominal-value step does not cause any damage to the machine mechanics. It may be necessary to determine the acceleration by approximation.

- Switch to the **Oscilloscope** mode of operation.
- Set the following values in the oscilloscope by pressing the **SELECTION** soft key:
  Display mode: YT
  Sampling time: IPO clock
  Channel 1: V actl
  Channel 2: V noml
  Channel 3: Analog
  Trigger: Free run
- Press the **OSCI** soft key.
- Press the **START** soft key to start recording.
- Press the rapid traverse key together with the axis-direction key to output the maximum possible feed rate.
- Press the **STOP** soft key to stop recording.
- From the step response of the actual velocity (**V actl**) you determine the maximum possible acceleration (incl. 10 % safety margin).

$$a = \frac{Vnom}{t_{an} \cdot 66\,000}$$

Value, parameter	Unit	Description
a	m/s^2	Acceleration
Vnom	mm/min	Nominal velocity
t_{on}	s	Rise time



Example:

The internal oscilloscope measured the following rise-time value on the machine:

t_{on} = 0.125 s

The nominal value V noml is a machine-specific parameter and is determined using the internal oscilloscope with 5000 mm/min in this example.

Calculation of acceleration:

$$a = \frac{5000 \frac{mm}{min}}{0.125\ s \cdot 66\,000} = 0{,}61 \frac{m}{s^2}$$

- Enter the calculated value in the parameter **MP_maxAcceleration** and check the calculated value by performing a measurement with the internal oscilloscope.
- Now reset the temporary input values in **CfgPositionFilter** to the initial values before continuing commissioning.

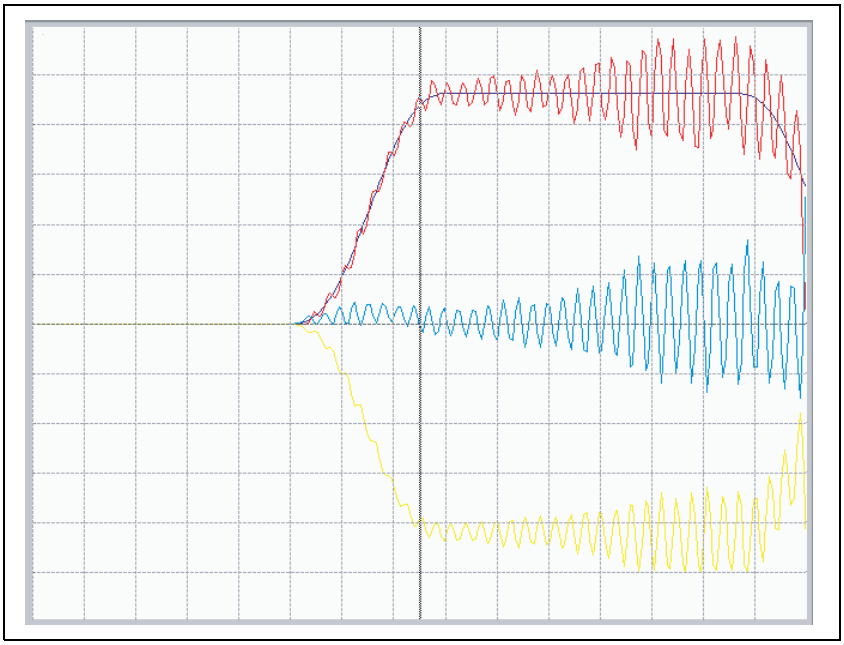
Determining the k_V Factor

Machine parameters in the configuration editor	Temporary input value	Meaning
Axes ParameterSets [Key for parameter block] CfgPosControl **kvFactor**	15	k_V factor

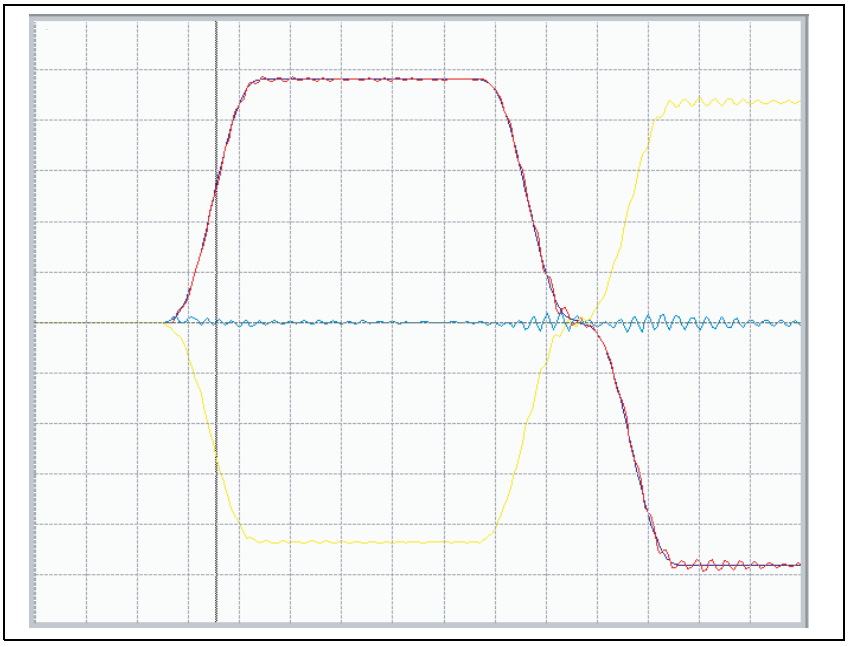
Note: If the value entered causes the control loop to oscillate, the value must be reduced.

- Enter the following test program:
  0 BEGIN PGM TEST MM
  1 LBL 1
  2 L X+100 R0 F2000
  3 L X0 F2000
  4 CALL LBL1 REP 10/10
  5 END PGM TEST MM
  The test program should be structured so that the axis reaches the nominal velocity.
- Run the program at high speed (feed-rate override = 100%).
- With the integrated oscilloscope, record the nominal feed rate **(V noml),** the actual feed rate **(V actl),** and if desired, the servo lag **(P error)** as well.
- Perform the first measurement with the temporary k_V factor (15).
- Increase the k_V factor until the oscillation limit is reached.
- Calculate the starting value of the **MP_kvFactor** with the following formula:
  MP_kvFactor = <determined value of the oscillation limit> · 0.5

See the following oscilloscope illustrations for axis adjustment.



k_V Factor Too High, Axis Oscillates



Axis Has Reached the Oscillation Limit

Determining the Jerk

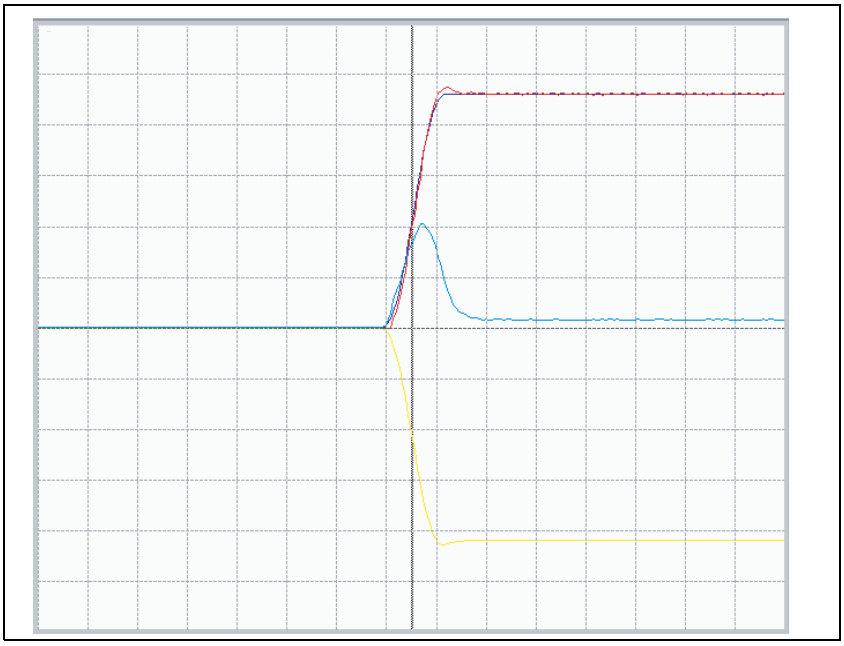
Settings in the configuration editor:	
Channels ChannelSettings Ch-Nc CfgLaPath **maxPathJerk**	

Note: Depending on the position of the axis slide on the ball screw, the axis can have different mechanical properties. Therefore you should repeatedly perform the following measurement several times in a row at different positions within the traverse range.

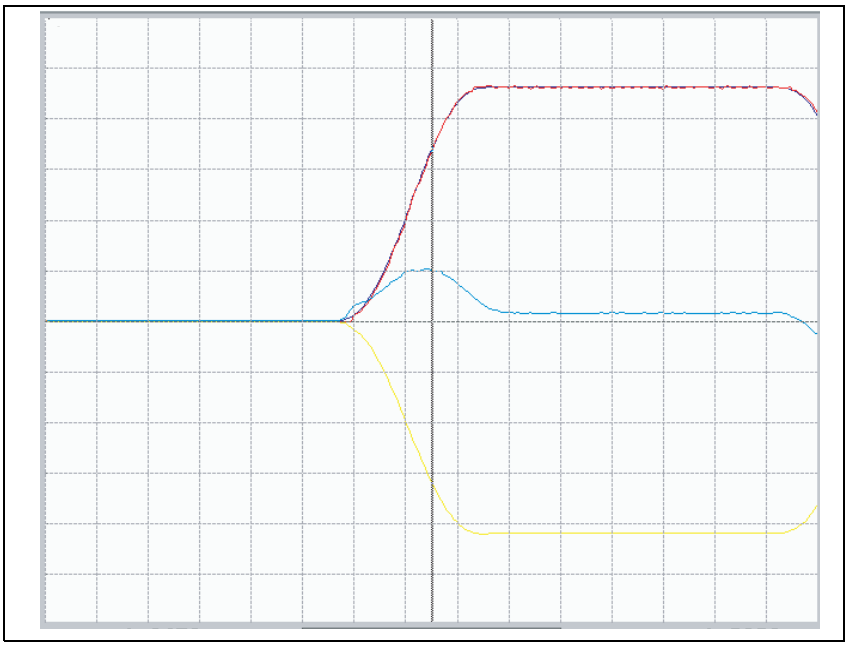
- Use the test program entered earlier for "**Determining the k_V Factor**" for this commissioning step as well.
- Run the program at high speed (feed-rate override = 100%).
- With the integrated oscilloscope, record the actual feed rate **(V act)** and if desired, the servo lag **(P err)** as well.
- Increase the parameter **MP_maxPathJerk** until the overshoot disappears.

Note: The **MP_maxPathJerk** parameter is globally effective for all axes. Therefore, sequentially determine the jerk for each axis individually. In the parameter you then enter the jerk of the interpolating axis with the smallest determined jerk value. The specific jerk values determined for each axis are then entered in **MP_axJerk**.

See the following oscilloscope illustrations for the Jerk adjustment.



Value in **MP_maxPathJerk** Too High



Value in **MP_maxPathJerk** Adjusted Correctly

Determining Acceleration Feedforward Control

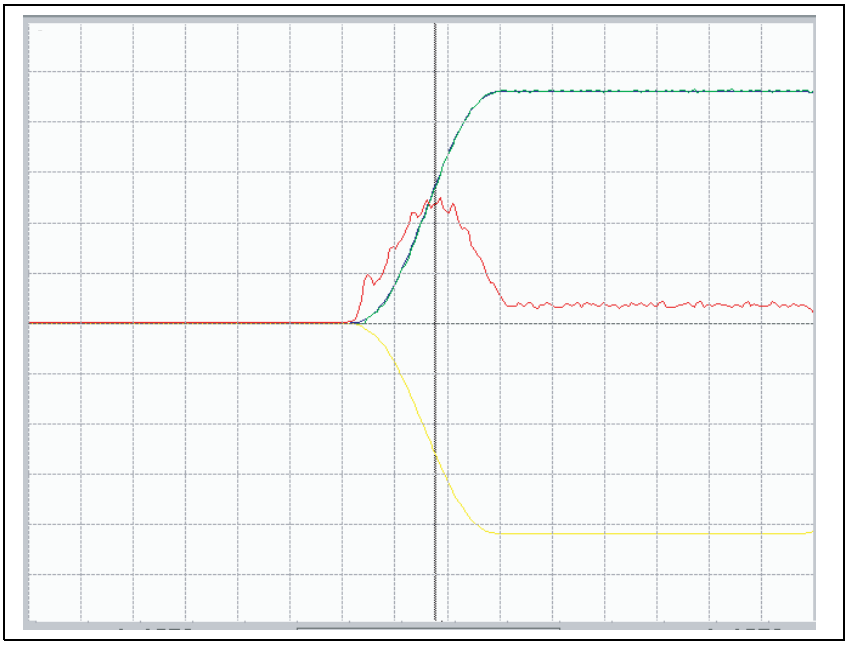
Goal: The following error (servo lag) is to be set as small as possible during the acceleration phase.

Machine parameters in the configuration editor	Temporary starting value	Meaning
Axes ParameterSets [Key for parameter block] CfgAxisAnalog **accForwardFactor**	0.005	k_V factor

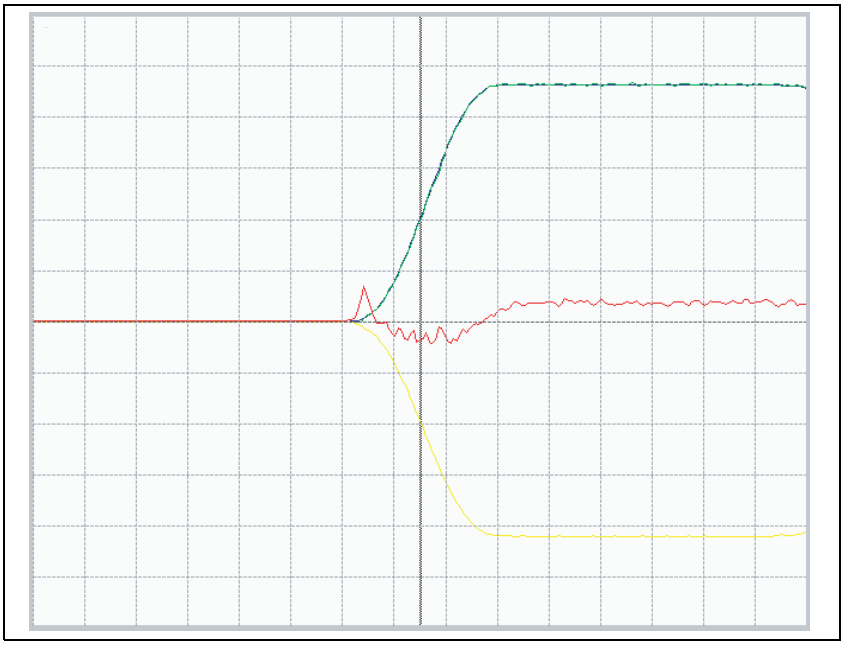
This parameter is determined via step-by-step approximation to the ideal value:

- Carefully increase the temporary starting value to determine the optimum setting for **MP_accForwardFactor.** Use the value 0.01 as a test value in the next step.
- Determine the value just before an **undershoot** forms with the measured following error **(P err).**

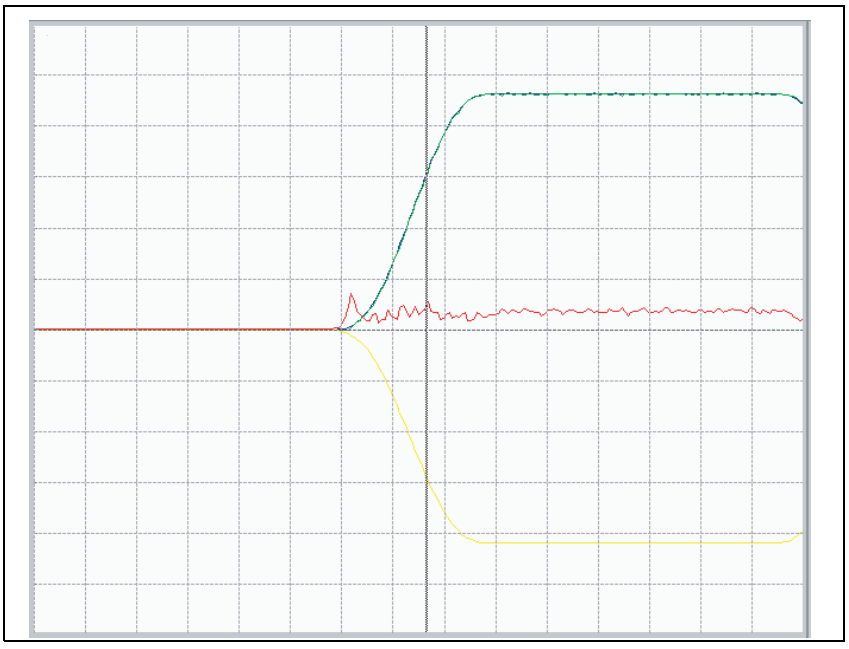
See the following oscilloscope illustrations for the Acceleration Feedforward adjustment.



MP_accForwardFactor Not Adjusted



MP_accForwardFactor Adjusted Too High



Correctly Adjusted **MP_accForwardFactor**

Setting the Traverse Range

Settings in the configuration editor:	
Axes ParameterSets [Key for parameter block] CfgPositionLimits **swLimitSwitchPos** **swLimitSwitchNeg**	

To define the software limit switches, proceed as follows:

- Traverse the reference points (e.g., by pressing the **PASS OVER REFERENCE** soft key.
- In the **Manual Operation** mode, press the **MOD** key to select the REF display.
- Position displays show the distance to the machine datum.
- With the axis direction buttons or handwheel, move all axes in positive and negative direction until they almost reach the EMERGENCY OFF limit switch. Write down the displayed positions.
- Enter the noted values in the machine parameters **MP_swLimitSwitchPos** or **MP_swLimitSwitchNeg**.
- Press the MOD key and select the ACTL display.

Note: You can enter different traverse ranges. You must define a separate parameter block per axis and traverse range. The individual traverse ranges are activated by switching the parameter blocks (e.g., by PLC).

Activating Monitoring Functions

The monitoring functions of the control must be activated now.

Note: To ensure that the monitoring functions of the control become effective at the right moment, you must enter meaningful values.

- Activate the position monitoring (refer to "**Section 5, Monitoring Functions, Position Monitoring**").
- You define two limits in the machine parameters for the position monitoring: one for operation with following error, and one for operation with velocity feedforward control.
- Configure the movement monitoring (refer to "**Section 5, Monitoring Functions, Movement Monitoring**").
- Configure the standstill monitoring (refer to "**Section 5, Monitoring Functions, Standstill Monitoring**").

Note: Adjust the input values to the machine dynamics.

Hysteresis / Static Friction

For configuring the reversal-spike compensation, see "**Section 5, Compensation of Reversal Peaks for Analog Axes**".

Diagnosis with the On-Line Monitor (OLM)

The following topics are described:

- **Introduction**
- **Operation of the OLM**
- **Screen Layout**
- **Selecting Axes and Channels**
- **Group of NC Axes**
- **Group of Spindle Commands**
- **Group of NC Channels**
- **Hardware Group**
- **Auxiliary Group**
- **PLC Group**
- **Queue Trace**
- **END Soft Key**
- **Frequent Causes of Error**

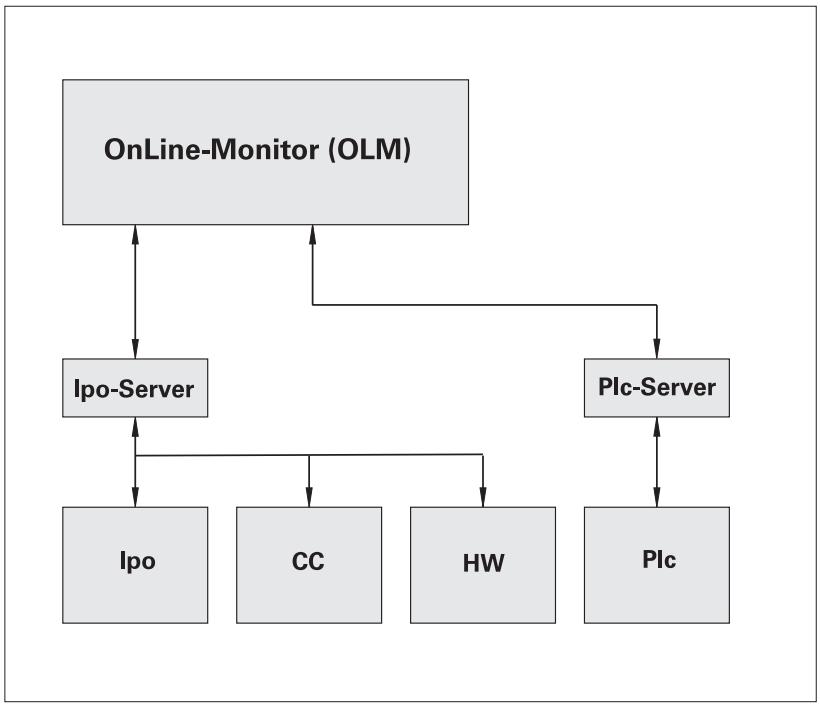
Introduction

The OLM (**O**n-**L**ine **M**onitor) assists you in commissioning and provides diagnostic functions to check control components:

- Display of IPO internal variables for axes and channels
- Display of CC internal variables (if a CC is present)
- Display of hardware signal states
- Different trace functions
- Activation of spindle commands
- Enabling IPO internal debug outputs

The OLM is included in the control software.

Software Structure



Operation of the OLM

The following topics are described:

- **Keyboard and Display**
- **Starting and Exiting the OLM**

Keyboard and Display

The OLM is operated using the soft keys of the control keyboard. The control screen is used for display.

The OLM distinguishes the following software and function areas:

- IPO
- Simulation IPO (SimIPO)
- PLC
- Trace

The software area or function area is selected by soft key on the “main level.”

For hardware reasons, only a subgroup of the IPO functions is available for the SimIpo. The available SimIpo functions are the same as the IPO functions.

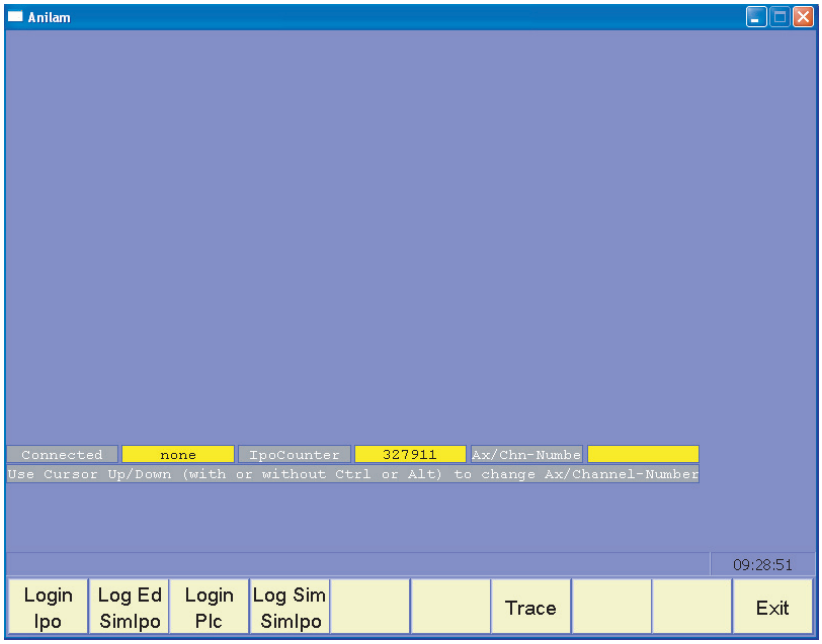
Starting and Exiting the OLM

The following topics are described:

- **To Start the OLM:**
- **To Start the Trace:**

To Start the OLM:

- On the Manual screen, press the **OLM** (**SHIFT** + **F8**) soft key, enter the password, and the control displays the OLM main screen.



Soft keys on OLM screen:

Login Ipo Opens the Login Ipo screen

Log Ed SimIpo Opens the Log Ed SimIpo screen

Login Plc Opens the Login Plc screen

Log Sim Ipo Opens the Log Sim Ipo screen

Trace Opens the Trace screen

Exit Exit the OLM screen, save changes, and return to the Manual screen

On the **Login Ipo** (**F1**), **Log Ed SimIpo** (**F2**), and **Log Sim SimIpo** (**F4**) screens, the following soft keys are displayed:

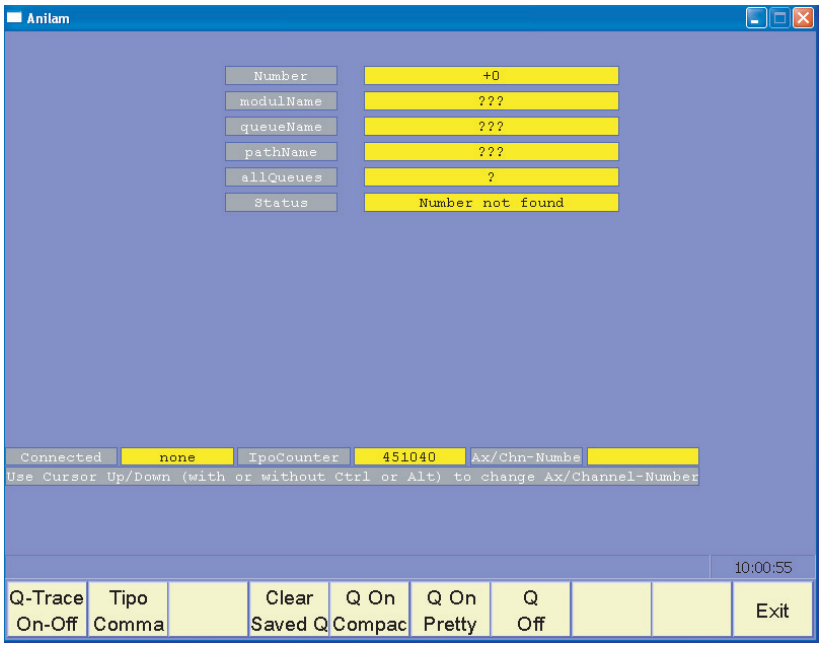
Axes (**F1**)	Opens the Axes screen
Chnls (**F2**)	Opens the Chnls screen
HW (**F3**)	Opens the HW screen
Drive Cmd (**F4**)	Opens the Drive Cmd screen
Auxil (**F5**)	Opens the Auxil screen
IpoTrace On (**F6**)	Opens the IpoTrace On screen
IpoTrace Off (**F7**)	Opens the IpoTrace Off screen
Exit (**F10**)	Exit the screen, save changes, and return to the OLM screen

On the **Login Plc** (**F3**) screens, the following soft keys are displayed:

AD-WERTE (**F1**)	Enables the AD-WERTE function
Plc Trc On-Off (**F3**)	Enables PLC Trace On-Off
Plc Trc Save (**F4**)	PLC Trace save
Exit (**F10**)	Exit the screen, save changes, and return

To Start the Trace:

- On the OLM screen, press (**F7**) **Trace** (see figure).



Soft keys on Trace screen:

Q-Trace On-Off (**F1**)	Displays the Trace On/Off dialog window
Tipo Comma (**F2**)	Displays the Tipo Commands dialog window
Clear Saved Q (**F4**)	Clears the saved Q
Q On Compac (**F5**)	Enables the Q on Compac
Q On Pretty (**F6**)	Enables the Q on Pretty
Q Off (**F7**)	Enables Q Off
Exit (**F10**)	Exit the Trace screen, save changes, and return to the OLM screen

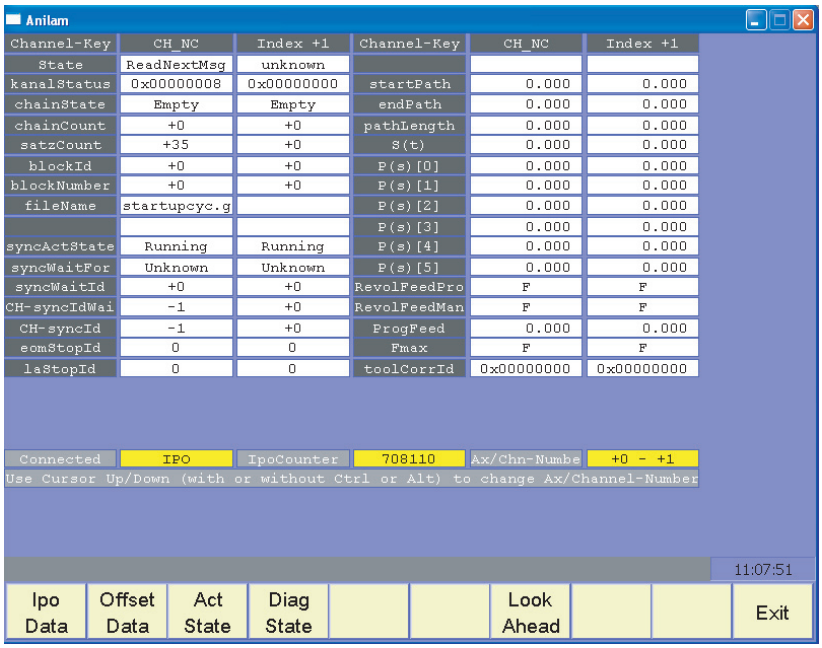
Screen Layout

The following topics are described:

- **Variable Display**
- **Units**
- **Status Display**

Variable Display

Example of screen layout when variables are displayed:



The OLM lists the variable designations, status designations, etc. in the **dark-highlighted boxes** of the “main window.”

The **white-highlighted boxes** contain the variable values. The OLM displays the values of two axes or channels.

In the **column heading,** the axis designation or channel designation defined in the parameters is shown.

- Parameter for axis designation:
  MP_Sytem/CfgAxes/axisList(axisId)
- Parameters for the IPO channel designation:
  MP_ChannelGroup/CfgChannelGroup/Machining/ChannelList
- Parameter for channel designation for SimIpo:
  MP_ChannelGroup/CfgChannelGroup/Simulation/ChannelList

The term **Index n** in the column heading means that no axis or no channel is defined for this index.

The following general data is displayed in the bottom display line:

- Connected: Indicates the software or function area to which the OLM is connected
  - Ipo
  - SimIpo
  - PLC
  - none: No connection
- IpoCounter: Counts the IPO clock pulses
  Note: The contents of the IpoCounter are also stored for trace information and error messages.
- Ax/Chn-Number: Logical axis number or channel number (depends on the selected function)
  - Number at left: Left column
  - Number at right: Right column

If the number of variables displayed exceeds the capacity of the main window, use PageDown/PageUp to scroll to the next/previous group of variables. One group consists of eight displayed variables.

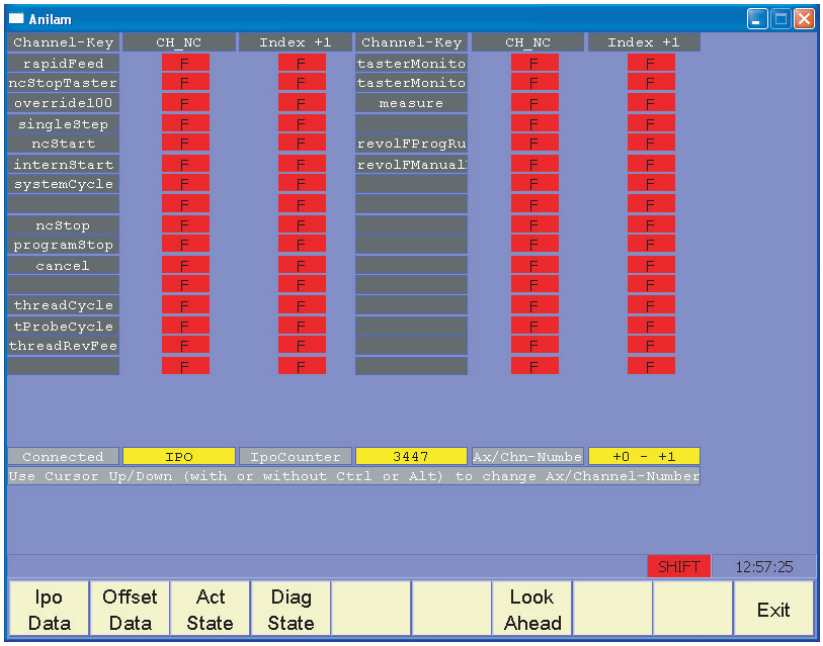
Units

The OLM displays data in the following units:

- For linear axes
  - For position, length, etc: in [mm]
  - For speed: in [mm/s]
  - For acceleration: in [mm/s^2]
- For rotary axes (spindles)
  - For position, etc: in [°]
  - For speed: in [°/s]
  - For acceleration: in [$°/s^2$]

Status Display

Example of screen layout for status display:



In the main window, the status of the binary variables is displayed. The status is identified by the background color and the code letter.

- Green or “T”: true
- Red or “F”: false
- Yellow: The status is not defined yet

The information given about the display of variables also applies to the column headings and the bottom display line.

Selecting Axes and Channels

To select axes or channels, proceed as follows:

- Press **CTRL** + UP ARROW / DOWN ARROW to influence the **left column.**
  - CTRL + UP ARROW: Display the next axis/channel.
  - CTRL + DOWN ARROW: Display the previous axis/channel.
- Press **ALT** + UP ARROW / DOWN ARROW to influence the **right column.**
  - ALT + UP ARROW: Display the next axis/channel.
  - ALT + DOWN ARROW: Display the previous axis/channel.
- Press UP ARROW / DOWN ARROW (without CTRL or ALT) to influence **both columns.**
  - UP ARROW: Display the next axes/channels.
  - DOWN ARROW: Display the previous axes/channels.

Group of NC Axes

The following topics are described:

- **Nominal Commands of the PLC (Plc Nom Data)**
- **IPO-Internal Variables (Ipo Act Data or Spindle)**
- **Internal Working Data of PLC-IPO (Plc Ipo Data)**
- **Data from the IpoOffset Module (Offset Data)**
- **Nominal Status of the Axes (Plc Nom State)**
- **Actual Status 1 of the Axes (Ipo Act State 1)**
- **Actual Status 2 of the Axes (Ipo Act State 2)**
- **Switching the Parameter Block of an Axis (Change ParSet)**
- **Deleting the Following Error (Clear PeakLag)**
- **Deleting the Reference Point (Clear RefOk)**

Nominal Commands of the PLC (Plc Nom Data)

To select the **PLC Nom Data** function:

- Press the **Login Ipo** (**F1**) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Plc Nom Data** (**F1**) soft key.

The function displays the nominal commands of the PLC for each axis in the following variables:

Variable	Description
PlcSollStatus	Axis status of the PLC as a bit line (listed in PLC-Nom_State)
MaxAchsVorschub	Maximum permissible axis feed rate in mm/s
AchsOverride	Override value for each axis (1 = 100%)
TempKorr	Temperature compensation in mm

IPO-Internal Variables (Ipo Act Data or Spindle)

The OLM displays the current IPO-internal variables of the selected axes (also spindle axes).

To select the **Ipo Act Data** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Ipo Act Data** (**F2**) soft key.

The display of the current IPO-internal variables is also activated when you select the **Spindle** function.

- Press the **Login Ipo** (**F1**) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Spindle** (**F4**) soft key.

The function displays the following variables:

Variable	Description
axisState1	Bit line (listed in IpoActState 1)
axisState2	Bit line (listed in IpoActState 2)
driveCmdWord	Command for universal controller
relNomPos	Relative nominal position
absNomPos	Absolute nominal position
absActPos	Absolute actual position
absNomFeed	Absolute nominal velocity
absActFeed	Absolute actual velocity
absNomAcc	Absolute nominal acceleration
absActAcc	Absolute actual acceleration
actParSet	Current parameter block index
ipoCtrlWord	Internal control bit line of nominal commands in the IPO chain
compensPos	Compensation value
masterId	Active master during synchronism
lag	Current following error
lagPeak	Peak of current following error. The maximum peak of the following error is determined and displayed here. Use CLEARPEAKLAG to delete it.
targetPos	Absolute target position of the axis
totalDistance	Total travel of the axis
motorTempera	Motor temperature in degrees Celsius
utilization	Utilization of axis in %
rpfOffset	Coordinate system offset between switch-on position and reference point
spindleTurns	Spindle revolutions – the value of the active spindle is calculated.
ModCounterAc	Only available for modulo axes. The counter is updated with each zero crossover of modulo axes. All positions leaving the IPO (display, PLC, etc.) are calculated from the current position (0–359.9999 degrees) + moduloCounter * 360. The counter can be set, cleared, stopped, and restarted from the geometry module.
absNomPosOfF	Absolute nominal position before the nominal position value filter (see configuration data **System/CfgFilter**)
absNomPosOfF	Absolute nominal position before the nominal position value filter (see configuration data **System/CfgFilter**)

Variable	Description
absNomPosBas	Nominal position without compensation of virtual axes
lastIpoPos	Last interpolated nominal value of interpolator
syncPosDiff	Position difference during spindle synchronism (for slave spindle)
absTouchPos	Absolute actual position provided by measuring system
virtStartPos	Starting position of virtual axis (reference position). Basis for determining the relative virtual offset.
requestedPos	Position commanded by HMI for returning to the contour

Internal Working Data of PLC-IPO (Plc Ipo Data)

To select the **Plc Ipo Data** function:

- Press the **Login Ipo** (**F1**) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Plc Ipo Data** (**F3**) soft key.

The function displays the following variables:

act-Cmd (currently active command)

last-Cmd (last assigned command)

Variable	Description
state	Possible states: • Finished: Command acknowledged. • Idle: Axis does not work and can be assigned a command. • MovingByHand: Manual direction key or PLC positioning is active. • StoppingByHand: Deceleration until standstill • WaitForPlcPosQuit: Waiting until the last nominal position value has been received by the axis (runtimes in the IPO chain) • RpfStart: Status during reference run • RpfFastToSwitchPreo: Status during reference run • RpfFastToSwitch: Status during reference run • RpfFastToSwitchPreo: Status during reference run • RpfFastFromSwitch: Status during reference run • RpfSlowToSwitchPrep: Status during reference run • RpfSlowToSwitch: Status during reference run • RpfAktivatePulse: Status during reference run • RpfWaitForPulse: Status during reference run • RpfWaitForStop: Status during reference run • RpfLatchPos: Status during reference run • RpfFinish: Status during reference run • WaitForSpindlemoveQuit: Waiting for acknowledgment of spindle (e.g., speed reached or synchronism switched on, etc.) • Spindlemove: Spindle rotates at programmed speed
typeOfMove	Possible states: • NONE = 0: Axis is in IDLE state • AXKEY: Manual direction key • PLCPOS: PLC positioning • LIFTOFF: Lift off at Cycle stop • RESTORE_POS: Return to contour (block scan) • SYNCHRON: Synchronism (only for spindles) • PLCMEASURE: Measuring with PLC axes • SG_POS: Positioning with safety-oriented (SG) package • REVOL_FEED: Feed per revolution • SPINDLE: Command to spindle (M3, M4, M19 etc.)
noLimitSw	During PLC positioning, software limit-switch monitoring can be switched off (T = switched off).

Variable	Description
error	The following errors can occur: • AxisAlreadyActive = 1: Axis is already working and cannot be given a new command. • PlcposAlreadyActive: Axis is already working and cannot be given a new command. • KeyposAlreadyActive: Axis is already working and cannot be given a new command. • OnlyOneAxWithKinem: When the kinematics model is switched on (M128), the PLC-IPO can give commands to only one axis at a time. • SweAlreadyActive: Axis is located at software limit switch. • MovementCanceled: Movement was canceled. • TsSwitched: Touch probe has triggered • ErrorPending: Probe error must be cleared first (in the error window). • NoToolAx: No tool axis. • MinFeed: Too small a feed rate is programmed. • M19_ACTIVE: M19 is active; axis cannot be moved. • NoChannel: Feed-per-revolution command was sent to an axis without NC channel. • M19WithoutRef: Not used. • ReconfigActive: Do not move any axis while changing parameters. • AlreadyMaster: This axis is a master and cannot be a slave at the same time (synchronism). • AlreadySlave: Only the SYCHRON_OFF command is allowed for a slave spindle. • NoSlave: The SYNCHRON_OFF command was sent to a non-slave spindle.
Vb-Prog	Programmed feed rate (mm/s)
Vb-Act	Active feed rate
Source	Type of handwheel (serial [e.g., HR410, etc., or encoder at handwheel input])
Dist/Revol	Traverse per handwheel revolution
DistMax	Maximum traverse range (+/–) of handwheel (e.g., M118)
Factor	Internal conversion factor (dist./rev. / incr./rev.)
Impulse	Handwheel pulses at current IPO clock
ImpulseLast	Handwheel pulses at previous IPO clock
Position	Handwheel position
PosRaster	Handwheel position (for handwheel with detent)
InputsToPLC	Handwheel keys are sent to the PLC as bit line
OutputsFromPLC	(e.g., LEDs on the RM 500)

Data from the IpoOffset Module (Offset Data)

The data of the interpolator and the PLC-IPO are collected in the module IpoOffset.

To select the **Offset Data** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F3**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Offset Data** (**F5**) soft key.

The function displays the following variables:

Example of coordinated movements of real and virtual axes:

- Absolute position of virtual axis = 5.0
- The virtual axis is moved to position 8.0.
- Resulting virtual offset (virtOffset) at the end of movement = 3.0.

Variable	Description
offsetIpoSteuer	Internal control bit line
IpoSteuer	Internal control bit line
IpoSteuerErlaubt	Internal control bit line
handValid	Nominal values from the PLC-IPO are available.
offsetPosition	SollPosition (relative) from PLC-IPO and/or kinematics.
kinematikOffset	Incremental offset of kinematics.
autoValid	Nominal values from the interpolator are available.
sollPosition	(Absolute) nominal position from the interpolator.
sollPosBase	Absolute position (sum of SollPosition and OffsetPosition)
lastPosition	Absolute position (sum of sollPosBase and virtual offset)
lastPosWithoutG	Position of the axis without grinding offset. The axis has reached this position by executing the movement in the standard interpolator.
grindingValid	Validity of the value in **grindingOffset.** True: The content of **grindingOffset** is added to the nominal axis value.
grindingOffset	Offset value generated by the grinding interpolator.
channelNr	Current channel number
The following variables coordinate the movements of the real and virtual axes. The nominal position value of the virtual axes is added to the nominal position value of the real axis (feedforward of nominal value). This applies only to the relative movements of the virtual axis.	
virtOffsValid	Validity of the value in **virtOffset.** true: **virtOffset** is effective.
virtOffset	Value of the relative movement of the virtual axis. (only important for real axes)
virtOffsActive = true	The feedforward of the nominal position value is configured and effective. (only important for virtual axes.)
virtStartPos	Starting position of the virtual axis before feedforward of nominal position value is activated. (Ist nur bei virtuellen Achsen von Bedeutung.)
realAxisNr	Number of the real axis to which the virtual axis was connected. (Ist nur bei virtuellen Achsen von Bedeutung.)
virtOffsetOn	Feedforward of position value of virtual axis is active. (Ist nur bei realen Achsen von Bedeutung.)

Nominal Status of the Axes (Plc Nom State)

The nominal status of the axes is requested by the PLC.

To select the **PLC Nom State** function:

- Press the **Login Ipo** (**F1**) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Next** (**F8**) soft key.
- Press the **Plc Nom State** (**F1**) soft key.

The function displays the status of the following binary variables (the descriptions refer to the status “true”):

Variable	Description
RpfNocken	Trip dog for reference end position
IstSollwUebern	Transfer the current values as nominal values.
ClampRequest	Request for clamping this axis.
PosCtrlRequest	Request for position feedback control for this axis
KeineUeberw	No monitoring of following error or standstill
VorschubFreigabe	Feed rate stop is not set
AntriebEin	Request to switch the drive on
IstSollUeberStrobe	Not used
CurrentOff	Switch off the current for wye/delta connection
SpiChangeDir	

Actual Status 1 of the Axes (Ipo Act State 1)

To select the **Ipo Act State 1** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Next** (**F8**) soft key.

Press the **Ipo Act State 1** (**F2**) soft key.

The function displays the status of the following binary variables (the descriptions refer to the status “true”):

Variable	Description
CMD_ACTIVE	Command is active for this axis
IM_FENSTER	Following error is within the positioning tolerance
SPEED_OK	Feed rate is OK
V_OK	No acceleration active
LGR_AKTIV	Position feedback control is active
reserve	
ANTRIEB_EIN	Drive is on
ANTRIEB_FREI	Drive ready for operation
MOVING	Axis is in motion (feed rate > 0)
DIRECTION	Direction of motion (true: negative direction or standstill)
SCHLEPP_OK	Not used
REF_OK	Axis was homed
VORSCHUB_FREI	Feed rate has been enabled (no feed stop)
NULLPULS	Reference pulse in one IPO cycle
LATCH_GUELTIG	Probe value is valid
ISTW_UEBERNOMMEN	The current value was transferred instead of the nominal value.
SCHLEPP_AUSF_REQ	If position feedback control is active, the “old” position is approached (no compensation of following error)
REQU_POS_REACHED	Requested position reached.
KEINE_UEBERW	Request from PLC: No monitoring of following error or standstill
MOVING_VNOM	Axis is in motion (nominal feed rate > 0)
SWE_POS	Positive software limit switch reached
SWE_NEG	Negative software limit switch reached
RELEASE_CONNECTOR	Status of axis-specific enabling (X150/X151)
BREAK_ON	Request for activating the brake within 100 ms to the PLC
POS_ERROR	Positioning error
I2T_WARN	Warning during I^2t monitoring
I2T_ERROR	Error during I^2t monitoring
TEMP_ERROR	Error during temperature monitoring
SG_REFERENCED	Axis was homed (SG: safety-oriented control)
SG_POS_TESTED	Axis was tested by the user (SG: safety-oriented control)
SG_PREPARED	Axis was homed and tested by the user (SG: safety-oriented control)
SG_SAVE	Safe axis (SG: safety-oriented control)

Actual status 2 of the axes (Ipo Act State 2)

To select the **Ipo Act State 2** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Next** (**F8**) soft key.
- Press the **Ipo Act State 2** (**F3**) soft key.

The function displays the status of the following binary variables (the descriptions refer to the status “true”):

Variable	Description
NO_CONTROL	Axis is not controlled (internal IPO status)
SPEED_CONTROL	Speed control is active (internal IPO status)
POS_CONTROL	Position control is active (internal IPO status)
INTERNAL_ERROR	Error has occurred (internal IPO status)
CHANNEL_AXIS	Axis belongs to a channel
CHANNEL_SPINDLE	Axis is a spindle of a channel
PLC_AXIS	Axis received a command from the PLC
PLC_SPINDLE	Axis was told to act as a spindle
IS_ACTIVE	Axis is physically available and can be given a command
IS_MANUAL	
IS_VIRTUAL	Virtual axis whose nominal position values can be added to those of other axes. (Axis does not have its own servo drive.)
IS_DISPLAY	Axis is only displayed. (Axis does not have its own servo drive.)
NORMAL_FEED	Feed rate in “travel/minute.”
REVOL_FEED_MANUAL	Feed rate in “travel per revolution” in the manual control mode.
REVOL_FEED_PROGR	Feed rate in “travel per revolution” in the automatic mode.
VCONST_FEED	Only for spindles – Feed rate at constant cutting speed
NC_CMD_ACTIVE	Command from the NC is active
PLC_CMD_ACTIVE	Command from the PLC is active
HR_ACTIVE	Handwheel is active
NC_STOP_ACTIVE	NC stop is active in the channel
SP_SPEED_REACHED	Only for spindles – last spindle speed reached
SP_MASTER	Axis is master spindle (for spindle synchronism)
SP_SLAVE	Axis is slave spindle (for spindle synchronism)
SP_SYNC_REACHED	Last spindle speed reached (for spindle synchronism)

Variable	Description
LGR_REQUEST	Only for spindles – IPO-internal request for position feedback control
TAKE_CYC_DATA	Use the nominal values from the cyclic message
IS_NOTACTIVE	Axis was configured but is not available physically
IS_ENDAT	

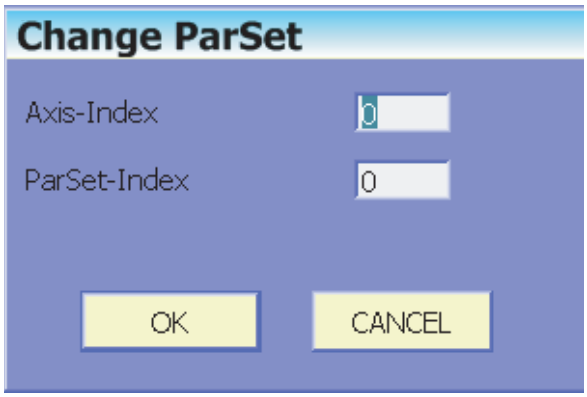
Switching the Parameter Block of an Axis (Change ParSet)

The active parameter block of an axis can be switched for test purposes.

To select the **Change ParSet** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Next** (**F8**) soft key.
- Press the **Change ParSet** (**F5**) soft key for the OLM to open the “Change ParSet” dialog box.
- Enter data in the dialog box (see below).
- Confirm with **OK** – the OLM switches to the defined parameter block.

“Change ParSet” dialog box:



Dialog box entries:

- **Axis-Index:** Logical axis number
- **ParSet-Index:** Index of the parameter block

Deleting the Following Error (Clear PeakLag)

The IPO saves the greatest following error (PeakLag) that occurred. The Clear PeakLag function deletes this variable for all axes.

To select the **Clear PeakLag** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Next** (**F8**) soft key.
- Press the **Clear PeakLag** (**F6**) soft key.

Deleting the Reference Point (Clear RefOk)

Deleting the reference point is necessary for being able to assign a new reference-run command. The function is effective for all axes.

To select the **Clear RefOK** function:

- Press the **Login Ipo** (**F1**) (or **Login SimIpo**) (**F1**)) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Next** (**F8**) soft key.
- Press the **Clear RefOK** (**F7**) soft key.

Group of Spindle Commands

The OLM transfers the spindle commands directly to the spindle. The current IPO internal variables are displayed in the display boxes.

- The standard spindle commands apply to the **spindle selected in the left column.**
- If you want to use commands for **spindle synchronism,** remember the following assignment:
  - Spindle in left column: Master spindle
  - Spindle in right column: Slave spindle

To select the **spindle commands:**

- Press the **Login Ipo** (**F1**) soft key.
- Press the **Axes** (**F1**) soft key.
- Press the **Plc Ipo Data** (**F3**) soft key.
- Press the **Spindle** (**F4**) soft key.
- Press the **M3** (**F1**) soft key or the soft key for another spindle command.

Specify the direction of rotation (**M3** [spindle forward] or **M4** [spindle reverse]) in the commands for spindle synchronism.

Spindle commands	
M3 (**F1**)	Spindle rotates (at 2345 rpm) in M3 direction
M4 (**F2**)	Spindle rotates (at 1234 rpm) in M4 direction
M5 (**F3**)	Spindle is stopped
V-Const (**F4**)	The spindle rotates at constant cutting speed (2000 m/sec in M3 direction)
M19 0 Grad (**F5**)	Spindle point stop at 0°
M19 179 Grad (**F6**)	Spindle point stop at 179°

Spindle synchronism commands	
Synchron +1* (**F1**)	Spindle synchronism • Same direction of rotation • Speed ratio master/slave: 1/1
Synchron +2* (**F2**)	Spindle synchronism • Same direction of rotation • Speed ratio master/slave: 1/2
Synchron -1* (**F3**)	Spindle synchronism • Reversed direction of rotation • Speed ratio master/slave: 1/1
Synchron +1* (+42°) (**F4**)	Spindle synchronism • Same direction of rotation • Speed ratio master/slave: 1/1 • Angle offset: 42°
Synchron -1/2 (-25°) (**F5**)	Spindle synchronism • Reversed direction of rotation • Speed ratio master/slave: 2/1 • Angle offset: –25°
Synchron OFF (**F6**)	Switch off spindle synchronism

Group of NC Channels

Data of the interpolator module (Ipo Data)

To select the **Ipo Data** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Chnls** (**F2**) soft key.
- Press the **Ipo Data** (**F1**) soft key.

The function displays the following variables:

Variable	Description
State	Possible IPO states: • Idle: IPO is idle (start-up) • RdNextMsg: IPO is waiting for job – reads from its input queue • Running: IPO is working (traverses the axes) • Waiting: Waiting during synchronism between IPO, PLC, and channel object • WaitingForLr: Waiting to ensure that the nominal value last generated was received by axes (IPO chain) • WaitingForCancel: Waiting for GmCanceled from the input queue • WaitingForAxes: Waiting until all axes are in the control window • WaitingForLiftOff: Waiting for lift off movement after NcStop • WaitingForLiftOffBack: Waiting for retraction movement after NcStart • WaitingForSpindle: Waiting for spindle command to be executed (internal M19 during drilling) • WaitingForClamping: Waiting for axis to be clamped. • WaitingForClampingOff: Waiting for axis to be unclamped. • WaitingForSync: Waiting for M97, G62, G63 (channel synchronism) • ShapeReset: • StartThreadCutting: Thread cutting
kanalStatus	Possible channel states: • rapidFeedActive: Rapid traverse is active for this channel • ncStopTasterActive: PLC request for ncStop at triggering of touch probe • override100Active: The override is frozen at 100% • singleStepActive: Single Block is active • ncStartActive: NC start is active • internStartActive: Internal NC start is active • systemCycleActive: A system cycle is active • ncStopActive: NC stop is active • programStopActive: Program stop (M00/M01) is active • cancelActive: Cancellation is active • threadCycleActive: A thread cycle is active • tProbeCycleActive: Touch probe cycle is active • tasterMonitorGeo: NC requests touch probe to be monitored • tasterMonitorPlc: PLC requests touch probe to be monitored • measure: The touch probe has triggered • revolFeedProgRun: Feed rate per revolution in automatic mode is programmed for this channel. • revolFeedManualMode: Feed rate per revolution in manual mode is programmed for this channel. • staticMask: Used internally.

Variable	Description
chainState	Status of the IPO chain: • IPO chain is “full.” • IPO chain is “empty.” • IPO chain is “almost empty” (waiting for the last acknowledgment message)
chainCount	Number of acknowledgment messages in the IPO chain
satzCount	Number of blocks in this NC program.
blockId	ID of the current block
blockNumber	Number of the current block (from NC program)
fileName	Current NC program
syncActState	
syncWaitFor	
syncWaitId	
CH-syncIdWait	
CH-syncId	
eomStopId	
laStopId	
startPath	Absolute starting position of current traverse on the path
endPath	Absolute end position of current traverse on the path
pathLength	Current path length
S(t)	(Absolute) position on the path
P(s)[0]	Position of the first axis to be interpolated
P(s)[1]	Position of the second axis to be interpolated
P(s)[2]	Position of the third axis to be interpolated
P(s)[3]	Position of the fourth axis to be interpolated
P(s)[4]	Position of the fifth axis to be interpolated
P(s)[5]	Position of the sixth axis to be interpolated
RevolFeedProg	Programmed feed rate per revolution in automatic mode
RevolFeedMan	Programmed feed rate per revolution in manual mode
ProgFeed	Programmed contour speed
Fmax	(if True:) Rapid traverse was programmed
toolCorrId	

The following topics are described:

- **Internal Data of the Offset Interface (Offset Data)**
- **Current Status of the Channel (Act State)**

Internal Data of the Offset Interface (Offset Data)

To select the **Offset Data** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Chnls** (**F2**) soft key.
- Press the **Offset Data** (**F2**) soft key.

The function displays the following variables from the IpoOffset:

Variable	Description
kindOfKinComp	Type of current kinematics
kindOfKinCompSave	Type of current kinematics
useFrozenAxVal	
v_bahn	Current contour speed
mySpindleNr	Spindle number belonging to this channel
achsAnz	Number of axes of this channel to be interpolated
logAchsNr[0] . . . logAchsNr[8]	First logical axis number of the channel . . . Ninth logical axis number of the channel

Current Status of the Channel (Act State)

To select the **Act State** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F1**)) soft key.
- Press the **Chnls** (**F2**) soft key.
- Press the **Act State** (**F3**) soft key.

The function displays the following binary variables:

Variable	Description
rapidFeed	Rapid traverse is active for this channel
ncStopTaster	PLC request for ncStop at triggering of touch probe
override100	The override is frozen at 100%
singleStep	Single Block is active
ncStart	NC start is active
internStart	Internal NC start is active
systemCycle	A system cycle is active
ncStop	NC stop is active
programStop	Program stop (M00/M01) is active
cancel	Cancellation is active
threadCycle	Thread cycle is active
tProbeCycle	Touch probe cycle is active
threadRevFee	
tasterMonitorGeo	NC request for monitoring of touch probe
tasterMonitorPlc	PLC request for monitoring of touch probe
measure	The touch probe has triggered. (If “tasterMonitorGeo” is set with the message “GmProbeMonitoring,” “tasterMonitorPlc” will also be set.) Use M141 to stop the monitoring of the touch probe temporarily (“GmPlcProbeMonitoring”). This resets “tasterMonitorPlc.”
revolFProgRun	Feed rate per revolution in automatic mode is programmed for this channel.
revolFManualMode	Feed rate per revolution in manual mode is programmed for this channel.

Hardware Group

The following topics are described:

- **Data of the Static RAM (S-RAM)**
- **Data of the Analog Outputs (Analog Output)**
- **Counter Function Blocks of the MC (GAL Data)**
- **Hardware Port States (HW ports)**

Data of the Static RAM (S-RAM)

This function displays the data stored in the static RAM of the IPO.

To select the **S-RAM** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **HW** (**F3**) soft key.
- Press the **S-RAM** (**F4**) soft key.

The function displays the following variables:

Variable	Description
kennung1	Internal code
kennung2	Internal code
absIstPos	Switch-off position of the individual axes
refPosition	Reference position of the individual axes
modCounterEndat	Overflow of multiturn EnDat encoder
checkSum	Checksum of current machine parameters
valid	Validity code

Data of the Analog Outputs (Analog Output)

Use **Analog Output** to display the nominal commands. (The values of the outputs are not returned.)

To select the **Analog Output** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **HW** (**F3**) soft key.
- Press the **Analog Output** (**F5**) soft key.

The function displays the values of the analog outputs in [V]:

- Output 0
- Output 1
- ...
- Output 15

Counter Function Blocks of the MC (GAL Data)

GAL Data allows you to display the internal registers of the counter function blocks of the MC.

To select the **GAL Data** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **HW** (**F3**) soft key.
- Press the **GAL Data** (**F6**) soft key.

The GAL Data variables have only an IPO-internal meaning:

Variable	Description
reg0_low	
reg0_mid	
reg0_hig	
reg1_low	
reg1_mid	
reg1_hig	
init_reg_1	
cntrl_reg_1	
RI_reg	
latch_reg	
irq_reg	
offset00_reg	
offset90_reg	
timer_reg	
cntrl_reg_2	
cntrl_reg_3	

Hardware Port States (HW ports)

The **HW-Ports** function displays the current status of some hardware ports. For the meaning of the displays, please refer to the description of the MC hardware. To make orientation easier, the relative addresses of the ports are indicated below. To obtain the actual port address, add the base address of the hardware to the relative address.

To select the **HW-Ports** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **HW** (**F3**) soft key.
- Press the **Next** (**F8**) soft key.
- Press the **HW Ports I** (**F1**) or **HW Ports II** (**F2**) or **HW Ports III** (**F3**) soft key.

The functions display the status of the following hardware signals:

Variable	Description
HW Ports I	• WD (IPO) • _SH2_p(CCU): Base address + 0x 330c • _NE1_p (I3): Base address + 0x330e • _NE2_p (I32): Base address + 0x3304 • _24V_plc2on: • _24V_plc3on: • _SH1AB_1_p: Base address + 0x3208 • _SHS1AB_1_p: Base address + 0x3204 • EN_SH2: Base address + 0x6000 • EN_NE1: Base address + 0x6004 • EN_PL: Base address + 0x6004 • EN_REG: Base address + 0x6006 • EN_MS: Base address + 0x6008 • EN_AT: Base address + 0x600a • EN_ACFAIL: Base address + 0x600c • IRQ_SH2: Base address + 0x6010 • IRQ_NE1: Base address + 0x6012 • IRQ_REG: Base address + 0x6016 • IRQ_MS: Base address + 0x6018 • IRQ_AT: Base address + 0x601a • IRQ_ACFAIL: Base address + 0x601c • IRQ_SYNCPWM: Base address + 0x601e

Variable	Description
HW Ports 2	• 3D-Signal • 3D-Bereit • 3d-Warng. • TT-Signal • TT-Bereit • X30-SpRef • WD-Reset • PLC-2+5V • oport1[0]: Base address + 0x3302 • oport1[1]: Base address + 0x3102 • oport1[2]: Base address + 0x3104 • oport1[3]: Base address + 0x3106 • oport1[4]: Base address + 0x3108 • oport1[5]: Base address + 0x310A • oport1[6]: Base address + 0x310C • oport1[7]: Base address + 0x310E • mg inst

Variable	Description
HW Ports 3	• IRQ SH2 • IRQ MNE1 • IRQ PLC • IRQ Reg/Spil • IRQ MS • IRQ Mitsu/Sp • IRQ AP • IRQ SyncPWM • IRQ busTimeo • IRQ UART1 • IRQ UART2 • IRQ FF • IRQ HWM • IRQ WD • IRQ • IRQ • MSK SH2 • MSK MNE1 • MSK PLC • msk Reg/Spil • MSR MS • MSK Mitsu/Sp • MSK AF • MSK SyncPWM • MSK busTimeo • MSK UART1 • MSK UART2 • MSK PF • MSK HWM • MSK WD • MSK • MSK enable a

Auxiliary Group

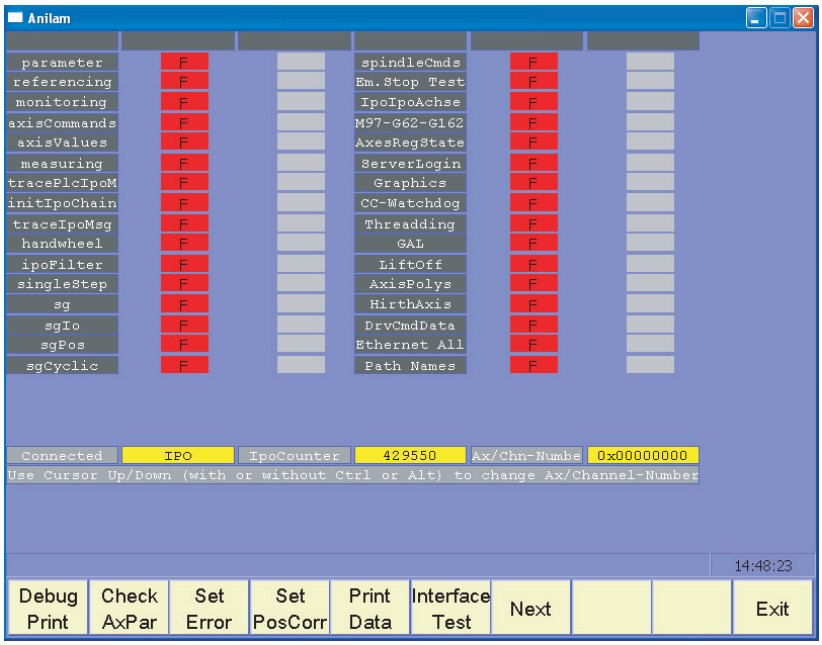
The following topics are described:

- **Enabling Debug Outputs (Get(Set DebugPrint)**
- **Saving Axis Parameters (Check AxPar)**
- **Generating Error Messages (Set Error)**
- **Generating an Asynchronous Position Compensation (Set PosCorr)**

Enabling Debug Outputs (Get(Set DebugPrint)

With **Get/Set DebugPrint,** you define the data to be logged and saved in the file r:\runtime_Xprint.txt. Data you identify by a “T” will be saved.

Example of a **selection:**



To select the **Get/Set DebugPrint** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Auxil** (**F5**) soft key.
- Press the **Debug Print** (**F1**) soft key for the OLM to open (Get...) the selection list (see figure below).
- Use the arrow keys to select the data whose identifiers you want to change.
- Use the **ENT** key to change the identifier (T or F).

To conclude the selection:

- Press the **Debug Print** (**F1**) soft key again for the OLM to save (Set ...) the selection.

The OLM uses the selected data to generate a bit line. The bit line is displayed in the bottom screen line at right. You can also use the bit line in the start batch of the IPO to start the IPO with the Debug Print function.

Meaning of the data:

Variable	Description
parameter	Output of information during parameter assignment. In addition, the parameters for every axis that are sent to the CC are written to the file r:\runtime_HelpTrace.txt.
referencing	Output of information during reference run
monitoring	Output of information during standstill monitoring and during monitoring of the absolute position (from zero pulse to zero pulse for distance-coded encoders)
axisCommands	Output of internal axis commands
axisValues	Output of information during actual-value transfer
measuring	Output of information on measuring process (probe on/off, monitoring, etc.)
tracePlcIpoMsg	All messages that are sent to the PLC-IPO are written to r:\runtime_HelpTrace.txt.
initIpoChain	Output during the initialization of the IPO chain
traceIpoMsg	All messages that are sent to the IpoInterpolator are written to r:\runtime_HelpTrace.txt.
handwheel	Output of information during the configuration and selection of the handwheel
ipoFilter	Output of information during the configuration and selection of two filters in the IPO chain
singleStep	Output of information during graphic simulation in the SingleStep mode of operation
sg	Output of additional information from the safety-oriented package (SG: safety-oriented control)
sgIo	Output of additional information from the safety-oriented package
sgPos	Output of additional information from the safety-oriented package
sgCyclic	Output of additional information from the safety-oriented package
spindleCmds	All spindle commands and their acknowledgments are recorded
Em.Stop Test	Outputs during the emergency stop test
IpoIpoAchse	Output during the configuration of the axes of a channel (exchanging axes in and removing axes from the interpolation context)
M97-G62-G162	Outputs during the synchronization of several NC channels
AxesRegState	Outputs at status change of axes in the controller
ServerLogin	Login to / logout of the IpoData server

Variable	Description
Graphics	Request of workpiece positions for the on-line graphics and graphic simulation
CC-Watchdog	Not used
Threadding	Outputs during thread cutting
GAL	Not used
LiftOff	Lift off of tool during Cycle stop
AxisPolys	Trace of the distance polynomials (result in file _HelpTrace.txt)
HirthAxis	Reserved
DrvCmdData	Reserved
Ethernet All	
Path Names	

Saving Axis Parameters (Check AxPar)

The **Check AxPar** function writes the collective axis parameters of all axes and parameter blocks to the file r:\runtime_ParamCheck.txt. The axis parameters are stored in such a way that they can be evaluated with a table calculation.

To select the **Check AxPar** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Auxil** (**F5**) soft key.
- Press the **Check AxPar** (**F2**) soft key.

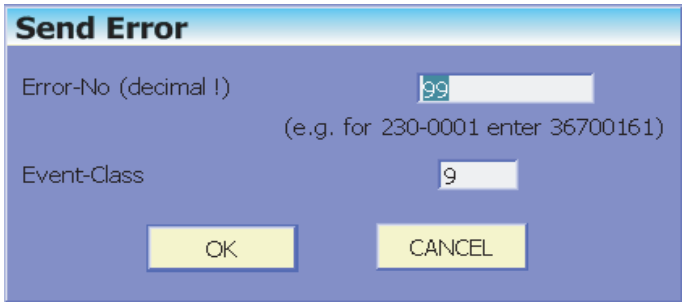
Generating Error Messages (Set Error)

The **Set Error** function generates error messages. You define the error number and the error class.

To select the **Set Error** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Auxil** (**F5**) soft key.
- Press the **Set Error** (**F3**) soft key for the OLM to open the “Send Error” dialog box (see figure below).
- Enter data in the dialog box (see below).
- Confirm with **OK** – the OLM generates an error message.

“Send Error” dialog box:



Dialog box entries:

- **Error-No:** Error number
- **Event-Class:** Error class

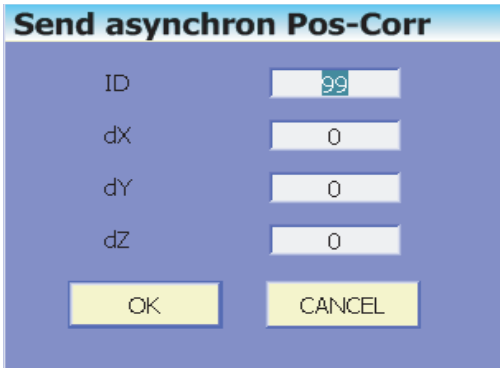
Generating an Asynchronous Position Compensation (Set PosCorr)

The **Set PosCorr** function generates an asynchronous position compensation (additive compensation). You define the number of the compensation and the compensation values.

To select the **Set PosCorr** function:

- Press the **Login Ipo** (**F1**) (or **Login Sim SimIpo** (**F4**)) soft key.
- Press the **Auxil** (**F5**) soft key.
- Press the **Set PosCorr** (**F4**) soft key for the OLM to open the “Send asynchron Pos-Corr” dialog box (see figure below).
- Enter data in the dialog box (see below).
- Confirm with **OK** – the OLM generates the compensation.

“Send asynchron Pos-Corr” dialog box:



Dialog box entries:

- **ID:** Compensation number
- **dX, dY, dZ:** Compensation values

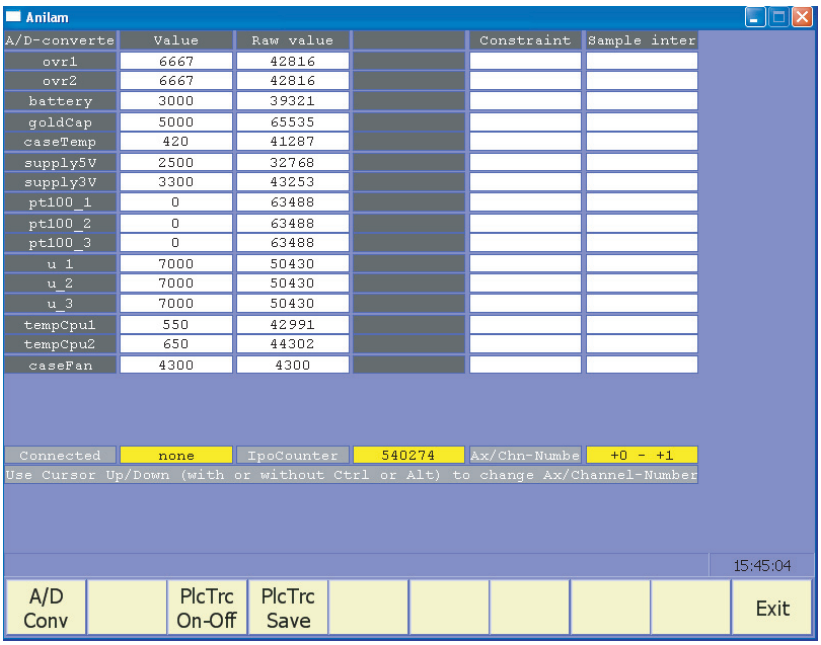
PLC Group

The following topics are described:

- **Displaying the Values of the Analog-to-Digital Converter**
- **PLC Trace**
- **PLC Trace On/Off**
- **PLC Trace Save**

Displaying the Values of the Analog-to-Digital Converter

Note: The displayed information is specific to the control and the machine. Refer to the technical documentation to find out which analog values are assigned on your control and the meaning they have.



To select the **A/D Conv** function:

- Press the **Login Plc** (**F3**) soft key.
- Press the **A/D Conv** (**F1**) soft key.

The function displays the values measured by the analog inputs as well as the permissible limits of some temperature and voltage values:

- **Value** column: Measured values
- **Raw value** column: Limit values

Displays:

Variable	Description
ovr1	Override (for example, values of the S and F potentiometers)
ovr2	
battery [V]	
goldCap [V]	
supply5V [V]	5 V supply voltage at the main board
supply3V [V]	3 V supply voltage at the main board
pt100_1 1–3 [°C]	Temperature inputs of the MC (X48
u_1 1–3 [V]]	Analog inputs of the MC (X48)
tempCpu1 [°C]	Temperature CPU 1
tempCpu2 [°C]	Temperature CPU 2
caseFan [rpm]	Fan speed

PLC Trace

The PLC Trace function saves the PLC modules called and errors that might have occurred during the module call. Depending on the setting of the PLC-TRACE ON-OFF soft key, the following modules are saved:

- **Plc Trace On:** The PLC Trace saves all module calls of the real-time thread and the submit/spawn thread.
- **Plc-Trace Off:** The PLC Trace saves only module calls that generate an error.

The PLC Trace saves the following information for each module call:

- IPO counter
- Module called
- Error number

Press the PLC-TRACE SAVE soft keys for the OLM to save the PLC-Trace data in the file r:\runtime\=APIModCall.txt.

PLC Trace On/Off

To define **Plc-Trace On-Off:**

- Press the **Login Plc** (**F3**) soft key.
- Press the **PlcTrc On-Off** (**F3**) soft key – the PLC-Trace status is changed.

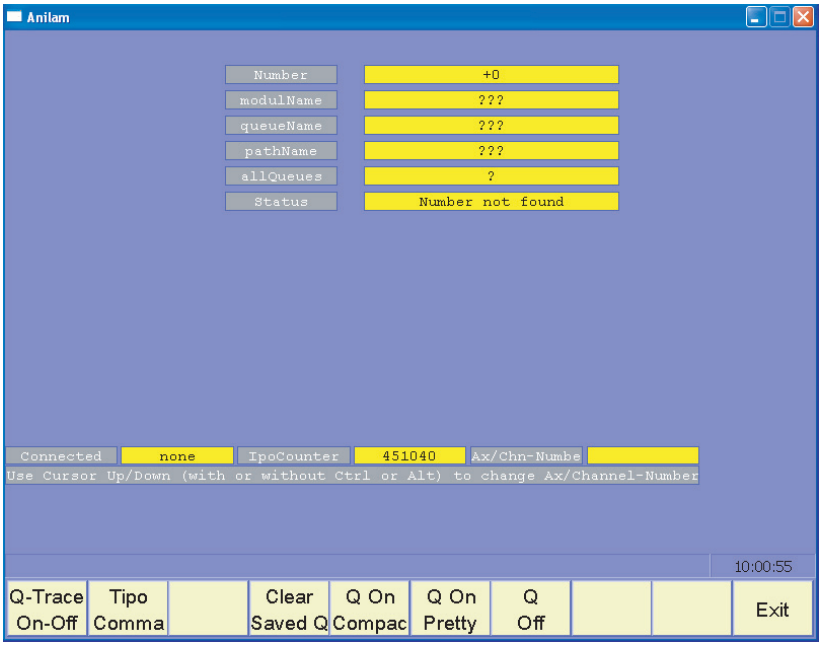
PLC Trace Save

To select the **Plc-Trace Save** function:

- Press the **Login Plc** (**F3**) soft key.
- Press the **PlcTrc Save** (**F4**) soft key – the PLC-Trace data is saved.

Queue Trace

The Q trace records the messages of the selected queues and saves them in a file.



Soft keys within the **Trace** function:

Soft key	Function
Q-Trace On-Off (**F1**)	Opens the “trace on/off” dialog box.
Tipo Comma (**F2**)	Tipo command is reserved for internal use only. Do not use.
Clear Saved Q (**F4**)	OLM deletes the existing file with trace information and starts a new file.
Q On Compac (**F5**)	The trace information is stored in a “compact” form (all information on one line).
Q On Pretty (**F6**)	The trace information is stored in a “structured” form (each information on a single line).
Q Off (**F7**)	The OLM stops the trace.
Exit (**F10**)	Returns to the previous screen.

After the Trace function has been selected, the OLM displays the data of the queue at the top of the screen (see figure).

- Press the **Trace** (**F7**) soft key for the OLM to display the data of a queue.
- **UP ARROW**: Displays the next queue.
- **DOWN ARROW**: Displays the previous queue.
- **CTRL + UP ARROW**: Scrolls forward in increments of 10.
- **ALT + UP ARROW**: Scrolls forward in increments of 100.
- **CTRL + DOWN ARROW**: Scrolls backward in increments of 10.
- **ALT + DOWN ARROW**: Scrolls backward in increments of 100.

The following topics are described:

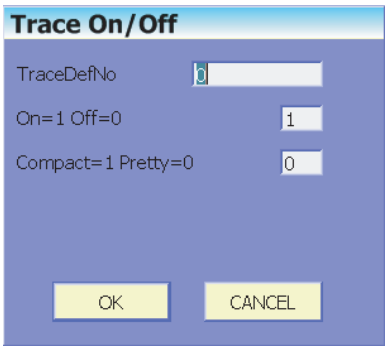
- **Activating a Q Trace**
- **Deleting Trace Information**
- **Saved Trace Information**
- **Stopping a Q-Trace**

Activating a Q Trace

The OLM saves the queues to be traced in a file. Define the entries in this file as follows:

- Press the **Q-Trace On-Off** (**F1**) soft key for the OLM to open the “trace on/off” dialog box.
- Enter data in the dialog box (see below).
- Confirm with **OK.**

“Trace On/Off” dialog box:



Dialog box entries:

- TraceDefNo: Enter the number of the queue (after selecting Trace, you can view details of the queue – see above).
- On/Off: Enter 0 or 1.

Deleting Trace Information

When you call the “trace onoff” dialog box for the first time **after starting the control,** the existing file will be deleted.

All entries made after that will be entered into a new file.

Saved Trace Information

To activate the “old” Q-Trace after starting the control:

- Press the Saved Q On soft key for the OLM to start the trace with the queues defined in the file.
  Prerequisite: The “trace onoff” dialog box has not been called yet or was not confirmed with OK.

Stopping a Q-Trace

- Press the **Saved Q Off** soft key for the OLM to stop the trace.

END Soft Key

When you press the END soft key on the main level, you exit the OLM. If soft-key rows on a sublevel are displayed, pressing the END soft key takes you back one level.

- Press the **END** soft key.

Frequent Causes of Error

The following topics are described:

- **Servo Drive Cannot Be Switched On**
- **Servo Drive Does Not Move**

Servo Drive Cannot Be Switched On

The servo drive cannot be switched on or does not move:

1 Check whether the drive was enabled by the CC.
Select: Login Ipo/Axes/Ipo Act State 1: **ANTRIEB_FREI** ()

ANTRIEB_FREI=false: Presumably an error on the CC or a hardware problem

2 Check whether “Drive on” was requested by the PLC.
Select: Login Ipo/Axes/Plc Nom State: **AntriebEin** ()

AntriebEin=false: Presumably an error in the PLC program

3 Check whether the drive was switched on.

Select: Login Ipo/Axes/Ipo Act State 1: **ANTRIEB_EIN** ()

ANTRIEB_EIN=false: Presumably an IPO-internal error

4 Check whether position feedback control was requested by the PLC.
Select: Login Ipo/Axes/Plc Nom State: **PosCtrlRequest** (does not apply for spindles)

PosCtrlRequest=false: Presumably an error in the PLC program

5 Check whether position feedback control is active.
Select: Login Ipo/Axes/Ipo Act State 1: **LGR_AKTIV**

6 Check whether “feed rate enable” was set by the PLC.
Select: Login Ipo/Axes/Plc Nom State: **VorschubFreigabe**

Servo Drive Does Not Move

The servo drive cannot be switched on or does not move although all enabling commands are available – check the following variables:

1 The maximum permissible axis feed rate must be > 0.
Select: Login Ipo/Axes/Plc Nom Data: **MaxAchsVorschub**

MaxAchsVorschub = 0: Presumably an error in the PLC program

2 The axis override must be greater than 0.
Select: Login Ipo/Axes/Plc Nom State: **AchsOverride**

AchsOverride = 0: Presumably an error in the PLC program

3 The IPO nominal speed must not be equal to 0.
Select: Login Ipo/Axes/Ipo Act Data: **absSollV**

absSollV = 0: Presumably an IPO-internal error

4 The IPO actual speed must not be equal to 0.
Select: Login Ipo/Axes/Ipo Act Data: **absIstV**

absIstV = 0: Presumably an IPO-internal error

Section 7 - PLC Programming

The following topics are described in this section:

- PLC Functions
- Operands
- Data Organization
- Tables
- Data Transfer NC -> PLC, PLC -> NC
- Program Creation
- PLC Commands
- INDEX Register (X Register)
- Commands for String Processing
- Submit Programs
- Cooperative Multitasking
- Constants Field (KF)
- Program Structures
- Linking Files
- PLC Modules

PLC Functions

The integrated PLC of the control contains its own text editor for creating the statement list for the PLC program. You enter PLC commands and comments using an optional USB keyboard. An even simpler way is to create your PLC program on a PC with the PLC development software **PLCdesignNT**. For more information on **PLCdesignNT**, contact ANILAM.

The control supports you with the COMPILE function, which compiles the PLC program and checks it for logical errors, and the API DATA, TABLE, TRACE, and WATCH LIST functions, with which you can check the status of the PLC operands.

Every 12 ms—the PLC cycle time—the control begins a new PLC scan (i.e., every 12 ms the inputs are reread and the outputs are reset). The PLC cycle time can be set in **MP_plcCount** and determined with Module 9196.

Module 9196 Find the PLC cycle time

The PLC cycle time is determined in µs.

Call:

CM	9196	
PL	D	<>PLC cycle time in [µs]>

MP_plcCount

PLC cycle time (Look Ahead cycle time)

Format: Numerical value

Input: 3 to 10 [**MP_lpoCycle**]

The PLC and the Look Ahead are running in a multiple of the IpoClock (interpolation clock). The Look Ahead is triggered exactly 2 IpoPulses after the PLC.

Default: 7 (this means the PLC cycle time is 21 ms)

Settings in the configuration editor:	
System CfgCycleTimes **ipoCycle** **plcCount**	

The following topics are described:

- **The Symbolic PLC-API (New Programming Interface)**
- **ANILAM PLC Basic Program**
- **Selecting the PLC Mode**
- **PLC Main Menu**
- **The Api Data Function**
- **The Watch List Function**
- **The Table Function**
- **The Compile Function**
- **The Edit Function**

The Symbolic PLC-API (New Programming Interface)

The 6000i is the first ANILAM control to be equipped with an expanded interface for communication between the PLC and NC. This interface is a new symbolic programming interface (**A**pplication **P**rogrammer **I**nterface. In this chapter it is referred to as **API**).

The previous, familiar interface between the PLC and the NC (the 6000i API with numerical addresses for markers, bytes, word, and double words) continues to exist in parallel and can be used as an option. However, ANILAM recommends working with the new expanded API. The **ApiMarker.def** file determines which API is used, see "**The Definition File ApiMarker.def**."

Note: ANILAM recommends using the new symbolic API (programming interface) for creating your PLC program.

Note: This 6000i Technical Manual only takes the symbolic interface PLC-API into account. All descriptions of modules and operands are based on the symbolic programming interface. The memory interface that is compatible with the 6000i (with default addresses for markers, bytes, words, and double words) is not described in this manual.

Danger: Ensure that you do not use any operands of the TNC-compatible API in your PLC program if you are using the symbolic API.
The use of the symbolic API prevents machine data from being saved to the addresses of the 6000i compatible API.
Using operands from the 6000i API in such a case can lead to a malfunction of the machine, possibly resulting in damages to persons or property.
If you are working with the symbolic API, ensure that you only use the PLC operands predefined by ANILAM for creating your PLC program, see "**Section 4, Overview of the PLC Operands**."

The following topics are described:

- **Benefits of the Symbolic API**
- **The Definition File ApiMarker.def**
- **Name Convention for Symbolic PLC Operands**
- **Programming with the Symbolic API**

Benefits of the Symbolic API

Overview of the most important improvements:

- Fundamental data of axes/spindle in symbolic operands
- The only basic types of data are markers (M) and double words (D)
- There is only one symbolic definition of the interface
- The data is organized in structures for associated ranges
- Structures can exist more than once (e.g., structures for axes)
- Redundant operands were eliminated, no multiple assignment
- Redundancy of operand/module no longer exists

The Definition File ApiMarker.def

ANILAM makes the ApiMarker.def file available to the PLC developer. As soon as the file is included in the PLC program via the **INCLUDE** command, the control uses the symbolic API. Collected in groups, the file contains all symbolic PLC operands.

Example:

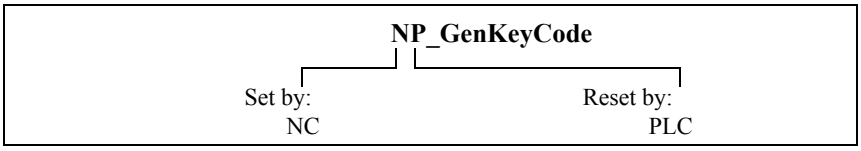
```
* ================================================
      #TYPEDEF PlcApiAxis
      * ================================================
        internalID
        NN_AxLogNumber  D * logical axis number
        NN_AxDriveReady M * drive is ready to work
        PP_AxDriveOnRequest M * drive on request
        NN_AxDriveOn M * drive is on
        .
        .
```

Note: The ApiMarker.def file is located on the PLC partition of the control.

ANILAM releases a revised version of the ApiMarker.def file at irregular intervals. The current version of the file is a part of the most recent NC software and can also be found on the ANILAM file base on the Internet (filebase.anilam.de).

Name Convention for Symbolic PLC Operands

The first two letters at the beginning of the symbolic operands give information about the setting and resetting behavior:



Operand	Data direction	Setting or resetting behavior
NN_xxx	NC –> PLC	Set by NC, reset by NC
NP_xxx	NC –> PLC	Set by NC, reset by PLC
PP_xxx	PLC –> NC	Set by PLC, reset by PLC
PN_xxx	PLC –> NC	Set by PLC, reset by NC

For a list and description of all PLC operands see “**Section 4, Overview of the PLC Operands**.”

Programming with the Symbolic API

In the definition file **ApiMarker.def,** associated symbolic PLC operands are structured into five groups with the **#TYPEDEF** command:

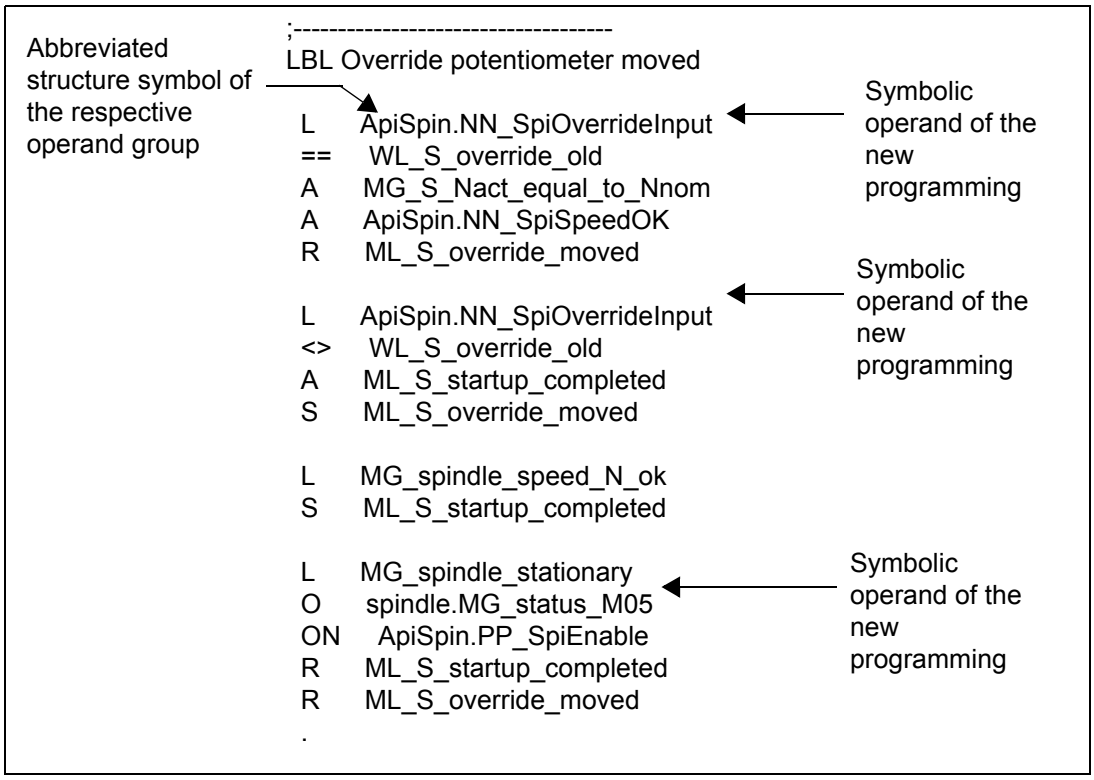
- PlcApiGeneral
- PlcApiOmg
- PlcApiChannel
- PlcApiAxis
- PlcApiSpindle

The **#TYPE** command assigns an abbreviated structure symbol to each of the five operand groups:

•PlcApiGeneral=**ApiGen**
•PlcApiOmg= **ApiOmg**
•PlcApiChannel=**ApiChn**
•PlcApiAxis= **ApiAxis**
•PlcApiSpindle=**ApiSpin**

You address individual elements of these five structures by entering the structure symbol, followed by a period as a separator, and then the name of the operand (e.g., **ApiSpin.NN_SpiOverrideInput**).

Example of program:



Note: For more information about the PLC program structures, refer to the on-line help of PLCdesignNT.

ANILAM PLC Basic Program

A PLC basic program for the control is available from ANILAM. This comprehensive PLC program serves as a basis for adapting the control to the requirements of the respective machine.

You need the software **PLCdesignNT** to adapt the PLC basic program.

The following functions are covered by the PLC basic program:

- Controlling all axes
- Positioning the axes after the reference run
- Clamped axes
- Homing the axes, reference end positions
- Temperature compensation of the axes
- Feed rate control
- Indexing fixture
- Controlling and orienting the spindle
- Activating tool-specific torque monitoring
- Manual or automatic tool change (pick-up device; single gripper or dual gripper). This definition is only a preselection; the PLC programmer must adapt the corresponding type to the exact requirements of the respective machine.
- Functions for setting up the tool changer
- Type of tool magazine (controlled by pulses or as an asynchronous axis)
- PLC soft keys
- Display and management of PLC error messages
- Display functions in the small PLC window
- Controlling the hydraulics
- Electronic handwheel
- Controlling the coolant system
- Handling of M functions
- Lubrication
- Chip conveyor
- Touch probe systems
- Controlling the doors

Selecting the PLC Mode

To select the PLC mode:

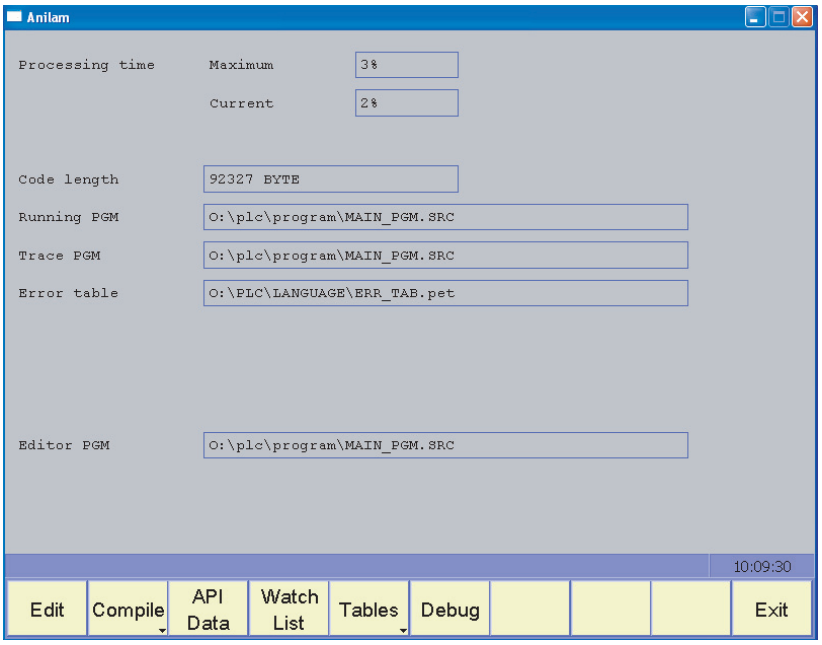
- Select the **Programming** mode of operation.
- Press the MOD key.
- Enter the code number 807667. Confirm with the **ENT** key or the **OK** soft key. If you had already entered the code number, press the **PLC EDIT** soft key.

To exit PLC mode:

- Press the **END** soft key or the **END** key.

PLC Main Menu

Press the **PLC** (**SHIFT** + **F5**) soft key, enter the password, and the control displays the PLC main screen:



Processing maximum:

Maximum run time of the PLC program.
The PLC processing time (time for a PLC scan) is given as a percentage of the maximum time: 1 ms is the equivalent of a run time of 100%.

The maximum cycle time of the sequential program must not exceed 2000 % (= 20 ms). If it is higher, the control outputs the blinking error message **PLC time out.**

Processing current:
The time taken for the latest PLC scan in %.

Code length:
Length of the compiled sequential program in bytes.

Running PGM:
Name of the last compiled PLC program (program in process memory).

During switch-on, the control automatically compiles the program that was selected in process memory before switch-off.
The files only become active after they have been compiled.

Editor PGM:
Name of the program or file in the editor's main memory.

The following topic is described:

- **PLC Functions of the Main Menu**

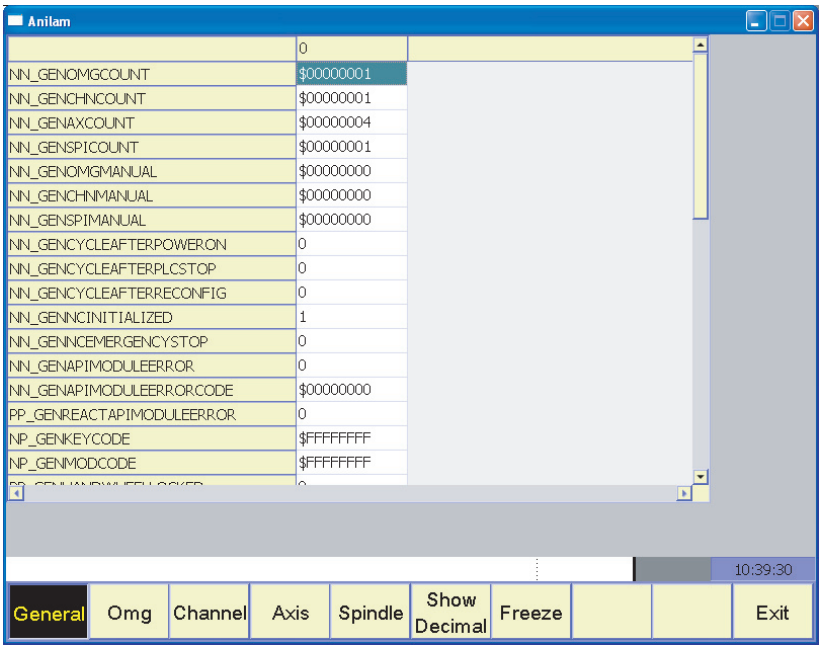
PLC Functions of the Main Menu

From the PLC main menu you can use soft keys to access the following PLC functions:

Soft key	Function
Edit (**F1**)	Edits the file located in RAM memory.
Compile (**F2**)	Compiles the PLC program.
API Data (**F3**)	Displays the states of the symbolic operands (new programming interface).
Watch List (**F4**)	Displays the states of the selected operands in a table.
Tables (**F5**)	Displays the logical states of the PLC operands (M/I/O/T/C/B/W/D).
Debug (**F6**)	Displays the logical states of the PLC operands (M/I/O/T/C/B/W/D).
Exit (**F10**)	Exits the **PLC programming** mode of operation.

The API DATA Function

The **API DATA** (**F3**) function enables you to display the states or contents of the symbolic API markers and API double words on the control. This function requires that your PLC program use the new PLC-API, see "**The Symbolic PLC-API (New Programming Interface)**."



Note: If you are using the programming interface (API) compatible with the TNC, the API DATA function is not active and does not provide useful display values.

Soft keys for the API Data function

Soft key	Function
General (**F1**)	Displays the contents of general API markers.
Omg (**F2**)	Displays the contents of the API markers for machining groups.
Channel (**F3**)	Displays channel-specific API data.
Axis (**F4**)	Displays axis-specific API data.
Spindle (**F5**)	Displays the API markers that apply to the spindle.
Show Decimal (**F6**)	Shows contents of operands as decimals or hexadecimals.
Freeze (**F7**)	Freezes the screen.
Exit (**F10**)	Returns to previous menu.

The Watch List Function

The **Watch List** (**F4**) function provides a dynamic overview of the states of the selected PLC operands.

Meaning of the columns in **Watch List**:

- MODULE: **<Global>** for global symbolic operands or path with the name of the *.SRC file in which the operand is defined
- ADDR: Absolute address of the operand
- VALUE: Content of the operand
- COMMENT: Comment for the operand

Soft keys within the **Watch List** function:

Soft key	Function
Insert Line (**F1**)	Inserts a new line above the current line.
Delete Line (**F1**)	Deletes the active line.
Symbol List (**F3**)	Displays a selection list of all symbolic operands used in the active PLC program.
Nav. Cmds (**F4**)	Displays the Navigation Commands softs listed below:
Arrow Left (**F1**)	Moves the screen cursor to the left.
Arrow Right (**F2**)	Moves the screen cursor to the right.
Page Up (**F3**)	Moves the screen cursor Page Up.
Page Down (**F4**)	Moves the screen cursor Page Down.
Freeze (**F7**)	Freezes the screen.
Exit (**F10**)	Returns to the previous menu.
Show Decimal (**F5**)	Shows contents of operands as decimals or hexadecimals.
Save Chgs (**F6**)	Saves the active WATCH LIST.
Load (**F7**)	Loads the saved WATCH LIST selection (*.WLT file).
Exit (**F10**)	Returns to the PLC main menu.

The following topics are described:

- **Display of Symbolic Operands in the Watch List**
- **Display of Operands in the Watch List**
- **Internal Process of the Watch List Function**

Display of Symbolic Operands in the Watch List

Press the **Watch List** (**F4**) soft key to open the WATCH LIST menu.

Press the **Symbol List** (**F3**) soft key to open a list box containing all global and local operands used in the PLC program.

Use the arrow keys to move within the SYMBOL LIST. Press the RIGHT ARROW key to open a tree structure. Press the LEFT ARROW key to close an open tree structure.

Use the arrow keys to select the desired operand, and press the **SELECT** soft key to transfer it.

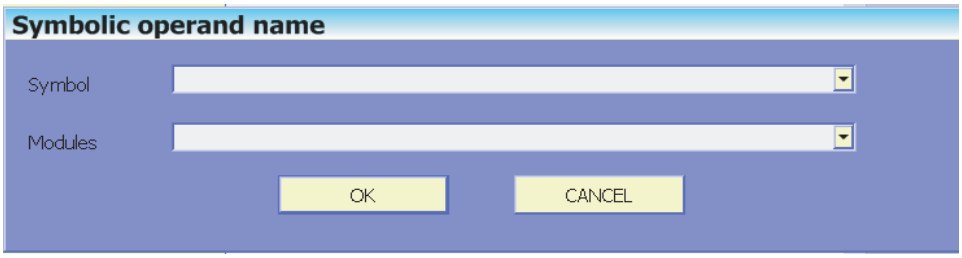
Press the **INSERT** soft key to insert the selected operand.

Note: Operands can only be selected with the **Symbol List** soft key if you are working with the *.SRC source files of the PLC program on the control. Otherwise, the error message **Selection list is empty** is displayed.

Display of Operands in the Watch List

- Press the **Watch List** (**F4**) soft key to open the WATCH LIST menu.
- Press the **Insert Line** (**F1**) soft key.

The following dialog box is displayed:



Enter the symbolic operand name under **Symbol**, or enter the module name under **Module.**

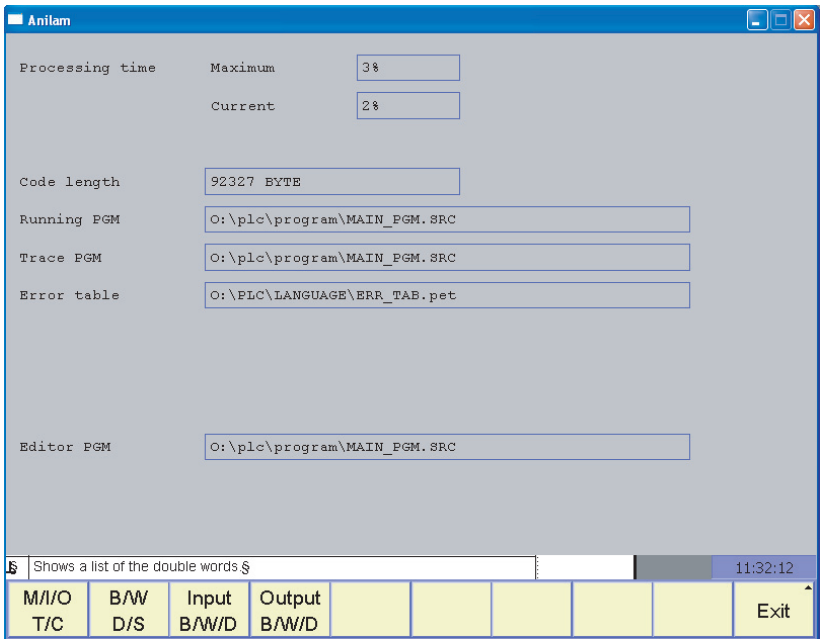
Confirm your entry with the **OK** soft key or button.

Internal Process of the Watch List Function

If you are working with the source files (*.SRC), the control internally creates a complete symbol list from the *.MAP file, identifying the structures and arrays. In another step, the control resolves the structure elements and array elements, removes the nesting levels, and internally creates a new list file. The generated information is displayed in a tree structure (SYMBOL LISTE function). When you select and then insert symbols the first time, the file PLC:\TABLE\TMP.WLT is automatically created. This file is automatically saved when you exit the WATCH LIST function and loaded when you call the WATCH LIST again. You can save the active WATCH LIST under any desired name and then reload it. The control remembers the last active WATCH LIST and automatically loads it when you call the function again.

The Table Function

From the PLC main menu, press the **Tables** (**F5**) soft key to select the table of markers, inputs, outputs, counters, timers, bytes, words, double words, and strings. The states are displayed dynamically on the screen.



Soft keys within the **Tables** (**F5**) function:

Soft key	Function
M/I/O/T/C (**F1**)	Displays the M/I/O/T/C screen (see figure)
B/W/D/S (**F2**)	Displays the B/W/D/S screen (see figure)
Input B/W/D (**F3**)	Displays the Input B/W/D screen (see figure)
Output B/W/D (**F4**)	Displays the Output B/W/D screen (see figure)
Exit (**F10**)	Returns to previous menu.

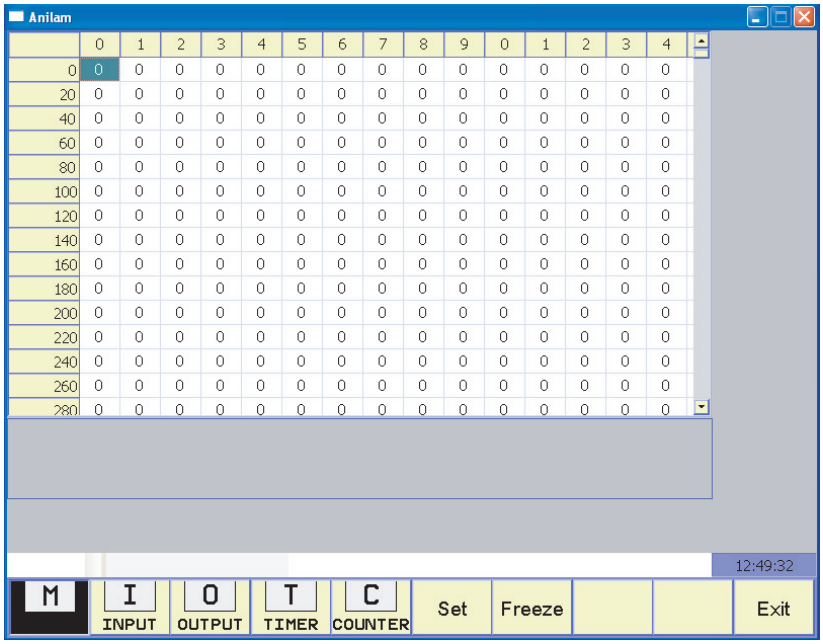
To select a certain operand:

- Press the GOTO key. A dialog box is displayed. Enter the number of the operand and confirm your entry with the **OK** soft key.

To set or change a certain operand:

- Press the **Set (F6)** soft key. A dialog box is displayed. Enter the number and the new value of the operand. Confirm your entry with the **OK** soft key.

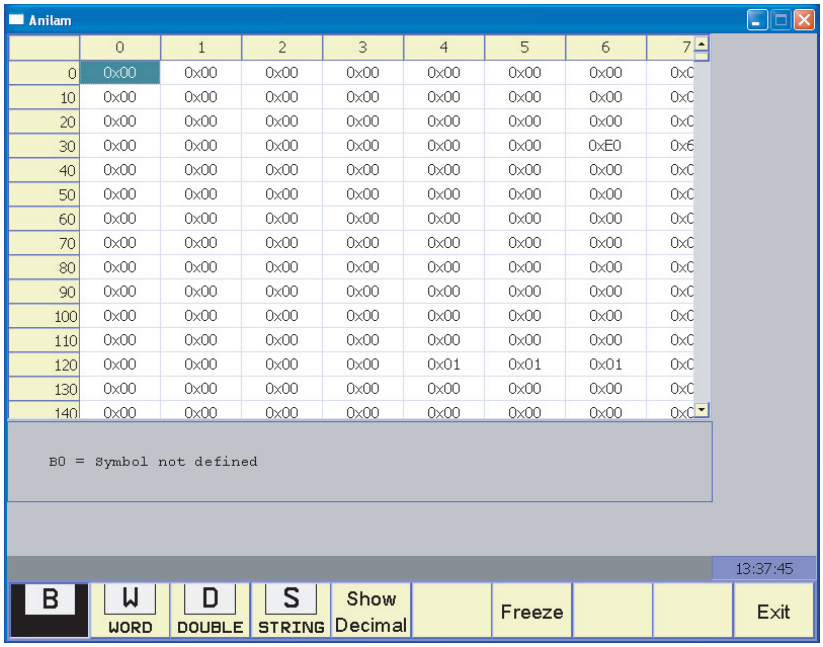
Select **M/I/O/T/C** (**F1**) on the Tables screen to display:



Soft keys within the **Tables** – **M/I/O/T/C** (**F1**) function:

Soft key	Function
MARKER (**F1**)	Shows a list of markers.
INPUT (**F2**)	Shows a list of the inputs.
OUTPUT (**F3**)	Shows a list of the outputs.
TIMER (**F4**)	Shows a list of the timers.
COUNTER (**F5**)	Shows a list of the counters.
Set (**F6**)	Opens the list box for selecting and setting an operand.
Freeze (**F7**)	Freezes the screen. Current changes to PLC operands are no longer shown.
Exit (**F10**)	Returns to previous menu.

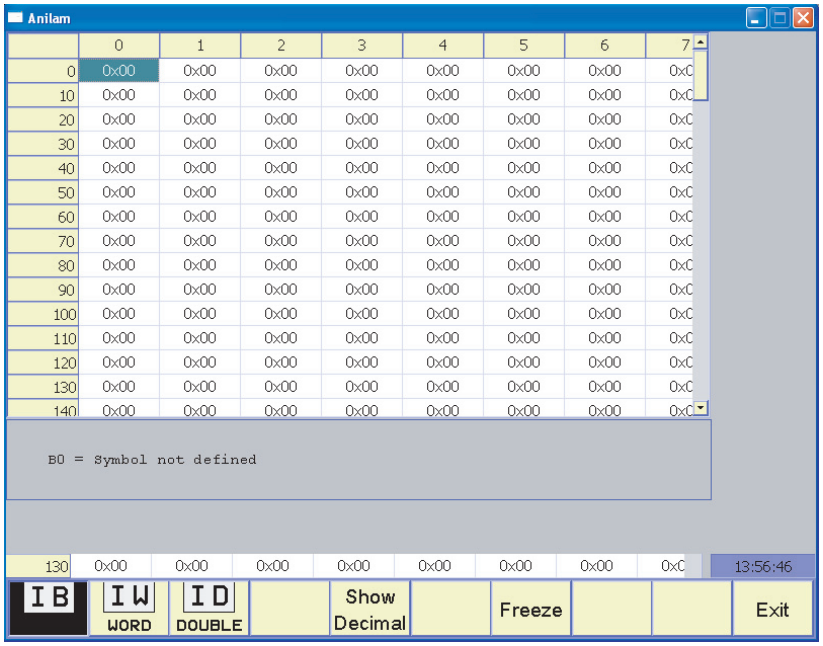
Select **B/W/D/S** (**F2**) on the Tables screen to display:



Soft keys within the **Tables** – **B/W/D/S** (**F2**) function:

Soft key	Function
BYTE (**F1**)	Shows a list of the bytes.
WORD (**F2**)	Shows a list of the words.
DOUBLE (**F3**)	Shows a list of the double words.
STRING (**F4**)	Shows a list of strings.
Show Decimal (**F6**)	Shows contents of operands as decimals or hexadecimals.
Freeze (**F7**)	Freezes the screen. Current changes to PLC operands are no longer shown.
Exit (**F10**)	Returns to previous menu.

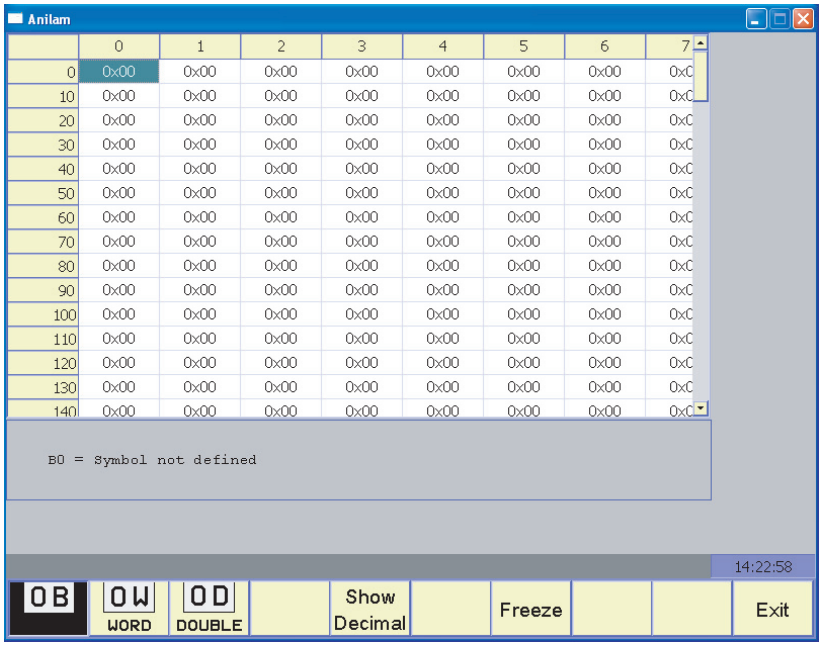
Select **Input B/W/D** (**F3**) on the Tables screen to display:



Soft keys within the **Tables** – **Input B/W/D** (**F3**) function:

Soft key	Function
Input BYTE (**F1**)	Shows a list of the input bytes.
Input WORD (**F2**)	Shows a list of the input words.
Input DOUBLE (**F3**)	Shows a list of the input double words
Show Decimal (**F5**)	Shows contents of operands as decimals or hexadecimals.
Freeze (**F7**)	Freezes the screen. Current changes to PLC operands are no longer shown.
Exit (**F10**)	Returns to previous menu.

Select **Output B/W/D** (**F4**) on the Tables screen to display:

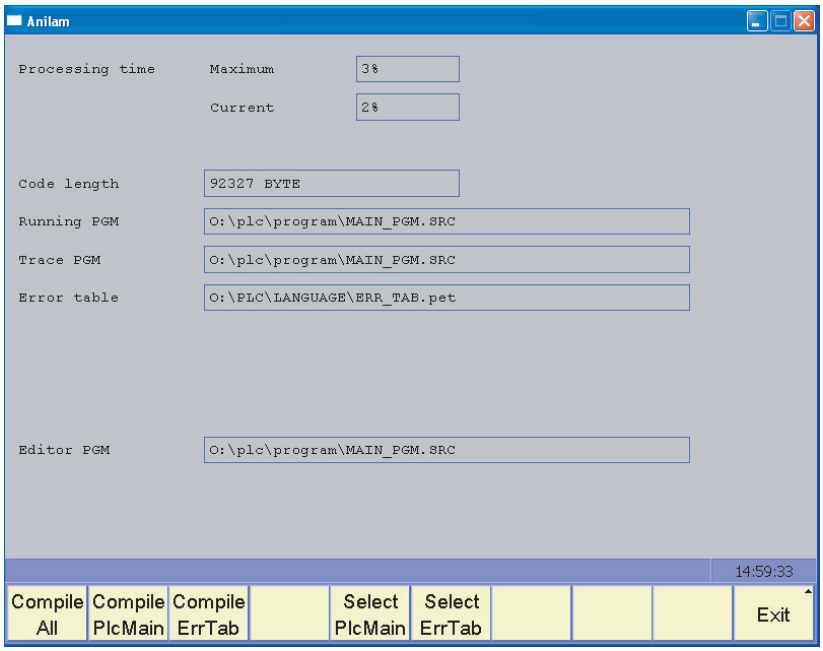


Soft keys within the **Tables** – **Output B/W/D** (**F4**) function:

Soft key	Function
Output BYTE (**F1**)	Shows a list of the output bytes.
Output WORD (**F2**)	Shows a list of the output words.
Output DOUBLE (**F3**)	Shows a list of the output double words.
Show Decimal (**F5**)	Shows contents of operands as decimals or hexadecimals.
Freeze (**F7**)	Freezes the screen. Current changes to PLC operands are no longer shown.
Exit (**F10**)	Returns to previous menu.

The Compile Function

Compiling a completed PLC program transfers it to the process memory where it can then become active. On the Manual screen, select (**SHIFT** + **F5**) **PLC**, then select **Compile** (**F2**). The name of the compiled program then is displayed in the main menu next to **Running PGM**.



To compile a PLC program:

Press the **Compile PlcMain** (**F2**) soft key: The control switches to the program manager.

Use the arrow keys to select the PLC program to be compiled.

Press the **SELECT** soft key.

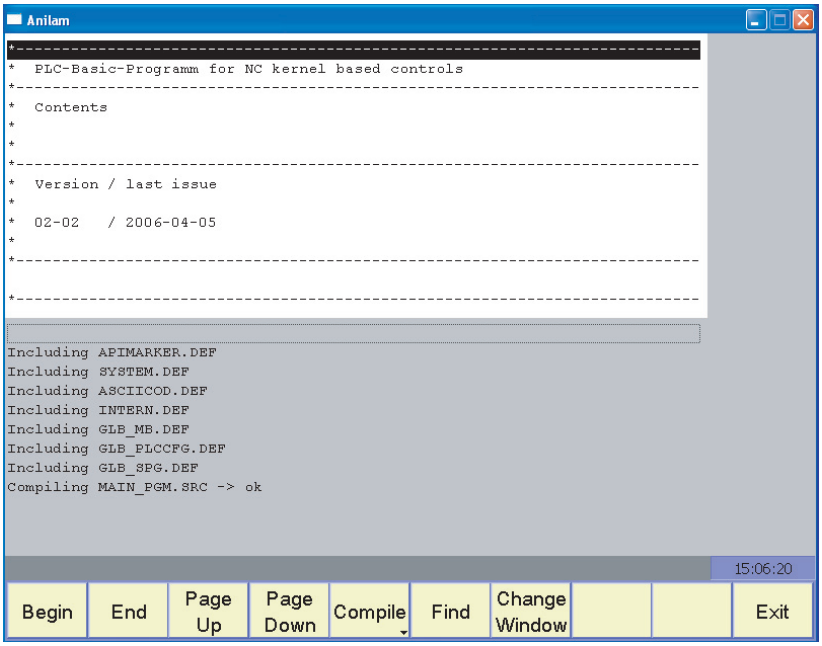
Note: The compilation of very extensive PLC programs may take some time. Compilation is completed when the PLC main menu is displayed on the screen again and values are displayed under **Processing time**. If errors occur during the compilation of the program, the control displays a corresponding message in the PLC main menu.

The Edit Function

Pressing the **Edit** (**F1**) soft key or selecting an editable file in the program manager opens the editor. PLC source texts (*.SRC, *.PLC), PLC definition files (*.DEF), and PLC error tables are editable files.

Tables are opened in a separate table editor.

The display window of the editor is divided into two parts: The upper part of the window is the workspace. In the lower part of the window, the PLC compiler displays status and error messages. If errors occurred during the compilation of the PLC program, these messages will assist you in finding the faults in the program.
The complete feature content of the editor only becomes available after you have connected an external USB keyboard.



Soft keys within the EDIT function:

Soft key	Function
Begin (**F1**)	The cursor jumps to the beginning of the file in the active window.
End (**F1**)	The cursor jumps to the end of the file in the active window.
Page Up (**F3**)	Moves the cursor page up in the active window.
Page Down (**F4**)	Moves the cursor page down in the active window.
Compile (**F5**)	Compiles the PLC program currently open in the editor (is only displayed if the cursor is in the upper part of the window).
Find (**F6**)	Opens the dialog box of the search function.
Change Window (**F7**)	Switches between the upper part (workspace) and the lower part of the window (status messages of the PLC compiler).
Exit (**F10**)	Returns to previous menu.

Operands

The following topics are described:

- **Operand Overview**
- **Timers**
- **Counter**
- **Fast PLC Inputs**

Operand Overview

Use the new expanded API (i.e., the symbolic PLC operands) in your PLC program. For a list of the symbolic PLC operands predefined by ANILAM, see "**Section 4, Overview of the PLC Operands**."

Operand	Short designation	Address range
Marker	M (marker)	**M0 to M9999** **M0 to M999** are free; they are deleted only after entering the code number 531210, but not during a reset (nonvolatile range); the range can be reduced in the *.CFG file of the PLC compiler. **M1000 to M3999** are free; they are deleted during reset. **M4000 to M5999** are reserved for the NC-PLC interface if you are using the old TNC-API. (M4800 to M4999 are deleted before the first run of the PLC program (e.g., after compilation or restarting)). They are freely usable if you are using the new symbolic API. **M6000 to M9999** are free; they are deleted during reset.
Input	I (input)	**I0 to I31 (X42)** **I128 to I152 (X46 machine operating panel)** **I64 to I127** (first PL input/output unit) **I192 to I255** (second PL) **I256 to I319** (third PL) **I320 to I383** (fourth PL)
Output	O (output)	**O0 to O30 (X41)** **O32 to O62** (first PL) **O64 to O94** (second PL) **O128 to O158** (third PL) **O160 to O190** (fourth PL)
Counter	C (counter)	Set counter: **C0 to C47** Counter content: **C48 to C95** Counter pulse release: **C96 to C143**
Timer	T (timer)	Timer start: **T0 to T47** Timer is running: **T48 to T95 and T96 to T999**
Byte Word Double word	B (byte) W (word) D (double word)	**B0 to B9999** (8 bits) **B0 to B255** are free; depending on the definition in the *.CFG file of the PLC compiler, the defined range is deleted only after entering the code number 531210, but not during a reset (nonvolatile range). If no range is defined in the *.CFG file, B0 to B127 is the nonvolatile range. **B256 to B2047** are reserved for the NC-PLC interface if you are using the old TNC-API. They are freely usable if you are using the new symbolic API. **B2048 to B9999** are free; they are deleted during reset.
Constant	K	**–2 147 483 647 to +2 147 483 647**
String	S	**S0 to S99**

The following topics are described:

- **Module 9405 Convert a symbolic operand into a numerical PLC operand**
- **Operand Addressing (Byte, Word, Double Word)**

Module 9405 Convert a symbolic operand into a numerical PLC operand

Module 9405 converts symbolic names of variables in character strings into the absolute addresses of the corresponding PLC operands. This module enables you to reduce the run time of your PLC program if you regularly read data from SQL tables.

The symbolic names must be contained within single quotes and follow a colon (e.g., **:'S_StringVariable'**).

If this conversion already occurred once during the initialization of the PLC program, then the processing time is reduced for the subsequent module calls that replace the embedded variables with the momentary values. This affects modules 9440 and 9450, for example.

Example:

Two symbolic operands are to be converted. The operand **MG_W_TOOLNR** has the absolute address W1234 and the numerical value 5.
The operand **MG_W_POCKET** has the absolute address W3456 and the numerical value 19.

Output string:
UPDATE TOOL_P SET T = :'MG_W_TOOLNR' WHERE P = :'MG_W_POCKET'

After execution of Module 9405:
UPDATE TOOL_P SET T = :'W1234' WHERE P = :'W3456'

After execution of Module 9450:
UPDATE TOOL_P SET T = 5 WHERE P = 19

Call:

PS	B/W/D/K/S	<>String with symbolic name>
PS	B/W/D/K/S	<>Target for string with resolved symbols>
CM	9405	
PL	B/W/D	<>Error number> 0: Module executed correctly 2: Parameter does not exist 3: Invalid address transferred 11: String could not be converted 12: String too long 15: Module was not called in a submit job

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Module executed successfully
	1	See above for errors

Operand Addressing (Byte, Word, Double Word)

The memory for operands B (8 bits), W (16 bits), and D (32 bits) is only 8 bits wide. Since the operands can be 8, 16, or 32 bits wide, an overlap of the memory areas will occur, which you must take into account when addressing the memory.

Double word	Word	Byte	Memory	Word address	Double-word address
D0	W0	B0	8 bits	High byte	Highest byte
		B1	8 bits	Low byte	
	W2	B2	8 bits	High byte	
		B3	8 bits	Low byte	Lowest byte
D4	W4	B4	8 bits	High byte	
		B5	8 bits	Low byte	
•	•	•	•	•	•
•	•	•	•	•	•
•	•	•	•	•	•
D1020	W1020	B1020	8 bits	High byte	Highest byte
		B1021	8 bits	Low byte	
	NN_GenApi ModuleError Code (W1022)	B1022	8 bits	High byte	
		B1023	8 bits	Low byte	Lowest byte

For byte addressing, every address is accessible; for word addressing, every second address; and for double word addressing, every fourth from 0 to 4092. The address parameter indicates the high byte of the word address (W) and the highest byte of the double-word address.

Markers, timers, and counters are addressed with the corresponding code letters M, T, or C followed by the operand number (e.g., M500, T7, C18).

Timers

Settings in the configuration editor:	
System PLC CfgPlcTimer **[timer]**	

The PLC has over 952 timers, which you control through special markers with the symbol T. You must define the run time of timers T0 to T47 in the machine parameter **[timer]>**. As the time unit, under **MP_unit** you can choose between seconds and PLC cycles. The name of the timer is freely selectable in the machine configuration.

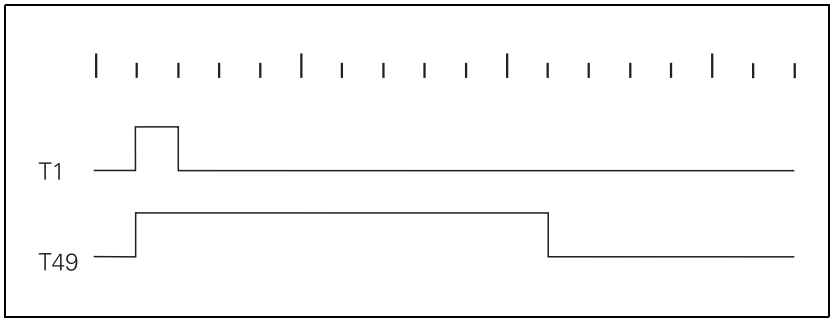
You start the first 48 timers by setting one of the timers T0 to T47 for at most one PLC scan (otherwise, the control restarts the timer with the negative edge for each additional scan). The control reserves the timer with the duration defined in **[timer]**, and sets the markers T48 to T95 (timer is running) until the defined duration has expired. A change of the preset value for a PLC timer only becomes effective after a PLC program restart.

You can also set and start the timers T0 to T47 with Module 9006.
The timers T96 to T999 can be started only via Module 9006.

Module 9197 can define and start cyclic timers (> T96). They are reset for a PLC cycle and are then automatically restarted.

Example:

Start of timer 1
Run time in **[timer]** = 10 (PLC cycles)



Timer starts	Timer is running	Machine parameters
T0	T48	**timer[0]**
T1	T49	**timer[1]**
T2	T50	**timer[2]**
T3	T51	**timer[3]**
T4	T52	**timer[4]**
T5	T53	**timer[5]**
T6	T54	**timer[6]**
T7	T55	**timer[7]**
T8	T56	**timer[8]**
T9	T57	**timer[9]**
T10	T58	**timer[10]**
T11	T59	**timer[11]**
T12	T60	**timer[12]**
T13	T61	**timer[13]**
T14	T62	**timer[14]**
T15	T63	**timer[15]**
T16	T64	**timer[16]**
T17	T65	**timer[17]**
T18	T66	**timer[18]**
T19	T67	**timer[19]**
T20	T68	**timer[20]**
T21	T69	**timer[21]**
T22	T70	**timer[22]**
T23	T71	**timer[23]**
T24	T72	**timer[24]**
T25	T73	**timer[25]**
T26	T74	**timer[26]**
T27	T75	**timer[27]**

Timer starts	Timer is running	Machine parameters
T28	T76	**timer[28]**
T29	T77	**timer[29]**
T30	T78	**timer[30]**
T31	T79	**timer[31]**
T32	T80	**timer[32]**
T33	T81	**timer[33]**
T34	T82	**timer[34]**
T35	T83	**timer[35]**
T36	T84	**timer[36]**
T37	T85	**timer[37]**
T38	T86	**timer[38]**
T39	T87	**timer[39]**
T40	T88	**timer[40]**
T41	T89	**timer[41]**
T42	T90	**timer[42]**
T43	T91	**timer[43]**
T44	T92	**timer[44]**
T45	T93	**timer[45]**
T46	T94	**timer[46]**
T47	T95	**timer[47]**

The following topics are described:

- **Module 9006 Set and start PLC timer**
- **Module 9197 Start cyclic timer**

Module 9006 Set and start PLC timer

With Module 9006, you define the run time for a PLC timer and start the timer.

Constraints:

- If during a PLC scan a timer from T0 to T47 is set in the PLC program, and the same timer is activated through Module 9006, then the direct activation through T0 to T47 has priority regardless of whether the module is called before or after setting T0 to T47.
- Immediately after the module call, one of the markers T48 to T96 is set. T0 to T47 are not set.
- The control rounds the actual run time up to integral PLC cycle times.
- Cancel run time: Reset timers T48 to T999.

Call:

PS	B/W/D/K	<>Timer number> Input value: 0 to 999
PS	B/W/D/K	<>Run time> 0 to 1 000 000 000 [ms] –1: Run time from machine parameter
CM	9006	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Timer started
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Illegal timer number or excessive run time
	2	Timer already allotted for cyclic timer
	3	Timer is started as cyclic timer (Module 9197)

Module 9197 Start cyclic timer

Module 9197 can define and start a timer > T96 as cyclic timer. After expiration of the defined time, the timer is reset for a PLC cycle, and afterwards is automatically restarted.

Stop timer: Transfer run time 0

The control rounds the actual run time up to integral PLC cycle times.

Call:

PS	B/W/D/K	<>Timer number> 96 to 999
PS	B/W/D/K	<>Run time> 0 to 1 000 000 000 [ms] –1: Run time from MP4111.x
CM	9197	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Timer started
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Excessive run time
	3	Illegal timer number

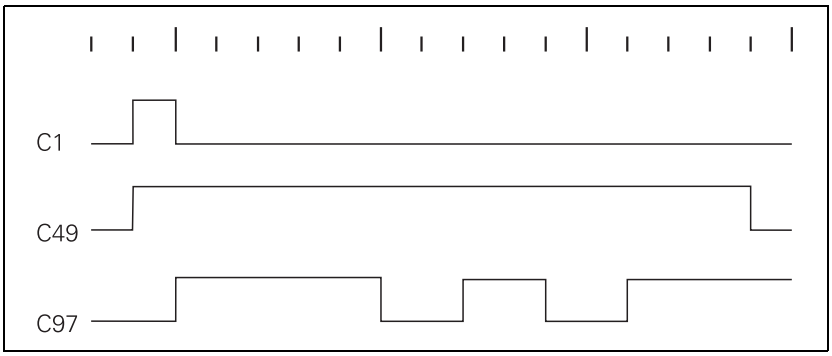
Counter

Settings in the configuration editor:	
System PLC CfgPlcTimer **[counter]**	

The PLC has 48 counters, which you control via special markers with the symbol **C**. After you have set a marker from the C0 to C47 range, the control loads the counter with the value that is saved in machine parameter **[counter]**. As the time unit, under **MP_unit** you can choose between seconds and PLC cycles. The marker range C48 to C79 indicates whether the counter has expired. With markers C96 to C127 you can start and stop the counter. The name of the counter is freely selectable in the machine configuration.

Example:

Logic diagram for counter C1
Counter default in **[counter]** = 10 (PLC cycles)



Counter is set	Counter is running	Counter is started	Machine parameters
C0	C48	C96	**counter[0]**
C1	C49	C97	**counter[1]**
C2	C50	C98	**counter[2]**
C3	C51	C99	**counter[3]**
C4	C52	C100	**counter[4]**
C5	C53	C101	**counter[5]**
C6	C54	C102	**counter[6]**
C7	C55	C103	**counter[7]**
C8	C56	C104	**counter[8]**
C9	C57	C105	**counter[9]**
C10	C58	C106	**counter[10]**
C11	C59	C107	**counter[11]**
C12	C60	C108	**counter[12]**
C13	C61	C109	**counter[13]**

Counter is set	Counter is running	Counter is started	Machine parameters
C14	C62	C110	**counter[14]**
C15	C63	C111	**counter[15]**
C16	C64	C112	**counter[16]**
C17	C65	C113	**counter[17]**
C18	C66	C114	**counter[18]**
C19	C67	C115	**counter[19]**
C20	C68	C116	**counter[20]**
C21	C69	C117	**counter[21]**
C22	C70	C118	**counter[22]**
C23	C71	C119	**counter[23]**
C24	C72	C120	**counter[24]**
C25	C73	C121	**counter[25]**
C26	C74	C122	**counter[26]**
C27	C75	C123	**counter[27]**
C28	C76	C124	**counter[28]**
C29	C77	C125	**counter[29]**
C30	C78	C126	**counter[30]**
C31	C79	C127	**counter[31]**

Fast PLC Inputs

Settings in:	
System PLC CfgPlcFastInput **number** **significance** **operand**	

With **CfgPlcFastInput,** you define the PLC inputs that are not interrogated within the defined PLC cycle time (see "**MP_plcCount**"); but rather, within the position controller cycle time (see "**Section 5, MP_ipoCycle**"). In MP_number, enter the PLC input of the control that is to be used as fast PLC input. You define the associated symbolic PLC operands set by the fast PLC input in **MP_operand.**

A maximum of five PLC inputs can be defined as fast PLC inputs. For this purpose, five input arrays [arrays 0–4] are available in the configuration editor of the control.

For the previous TNC-API, you activate the **fast PLC inputs** function in the PLC program, using W522 bit 2 to bit 5.

For the control to identify with certainty a signal change, the signal duration at the fast PLC input must last a minimum of 4 ms.

Warning: Only the PLC inputs of the control can be defined as fast PLC inputs, and not the inputs on a IEB 404.

MP_number

Numbers of fast PLC inputs

Format: Array [0–4]

Input: 0 to 31 [number of the PLC input on the LE]

Default: 0

MP_significance

Activation criterion for fast PLC inputs

Format: Array [0–4]

Selection:

[lowActive]
Activate at LOW level

[highActive]
Activate at HIGH level

[allEdges]
Activate at both levels

[disabled]
Not active

Default: disabled

MP_operand

PLC operand for fast PLC inputs

Format: Array [0–4]

Input: String of max. 24 characters
Example: NN_FastInput_01

Data Organization

The following topic is described:

- **PLC System Files**

PLC System Files

Paths and names for the PLC system files and data are specified in the file plc.cfg.

Note: Make sure that you only make entries to the plc.cfg file by using the configuration editor.

Path for the plc.cfg file

Settings in:	
CfgConfigDataFiles dataFiles **...** **...** **[10] PLC:\config\plc.cfg** **...** **...**	

Press to the **CONFIG FILE LISTS** soft key to enter the path for the **plc.cfg** file by using the configuration editor (code number 95148).
The paths of all *.cfg files necessary for the system are entered in the object CfgDataFiles under **dataFiles** (see "**Section 3, Machine Parameters**").
The default path for the plc.cfg file is PLC:\config\plc.cfg.
This means that the software looks for the necessary entries for PLC system data in this file in this path.

The following topics are described:

- **Paths and Names for PLC Programs and Text Files**
- **PLC Program Version**

Paths and Names for PLC Programs and Text Files

Settings in:	
System Paths CfgPlcpath **mainPgm** **pwmPgm** **errorTable** **errorText** **dialog** **softkeyProject** **compErrorTable** **compCfgFile** **events** **keymapFile**	

In the configuration editor, specify under **cfgPlcPath** which directories and names are used for storing PLC programs and files for PLC error messages.

The following path and file names must be specified:

- **MP_mainPgm:**
  The path and name of the PLC main program (*.src or *.plc). All necessary program modules (*.src) are called from this program as required. The program modules must be in the same directory as the PLC main program.
- **MP_pwmPgm:**
  Path and name for the PLC program for commissioning the machine. This PLC program is used as an alternative if the machine parameter **MP_currentControlAdjust** is set to “on.”
- **MP_errorTable:**
  Path and file name for the PET table (file name is plc_err.pet). PLC error table with reference to error message texts (plc_err.a) and attributes for the control’s behavior when an error occurs (created with the PLCtext software).
- **MP_errorText:**
  File name for error message texts (e.g., ErrorText.csv). Only the file name is entered – the language-sensitive text files must be saved in language-specific directories which cannot be changed.
  - German texts in PLC:**language\de**\plc_err.a
  - English texts in PLC:**language\en**\plc_err.a
  - Czech texts in PLC:**language\cs**\plc_err.a
  - French texts in PLC:**language\fr**\plc_err.a
  - Italian texts in PLC:**language\it**\plc_err.a
  - Spanish texts in PLC:**language\es**\plc_err.a
  - Portuguese texts in PLC:**language\pt**\plc_err.a
  - Swedish texts in PLC:**language\sv**\plc_err.a
  - Danish texts in PLC:**language\da**\plc_err.a
  - Finnish texts in PLC:**language\fi**\plc_err.a

- Dutch texts in PLC:**language\nl**\plc_err.a
- Polish texts in PLC:**language\pl**\plc_err.a
- Hungarian texts in PLC:**language\hu**\plc_err.a

- **MP_dialog:**
  Path and file name for PLC dialogs (e.g., Dialog.csv). Only the file name is entered – the language-sensitive text files must be saved in language-specific directories which cannot be changed. (see error message texts)
- **MP_softkeyProject:**
  Path and name of project file for PLC soft keys;
- **MP_compErrorTable:**
  Preset by ANILAM; do not change it.
  Path and file name of the error table of the PLC compiler;
- **MP_compCfgFile:**
  Path and name of the configuration file for the PLC compiler;
- **MP_events:** (Currently not evaluated.)
  Path and file name of the PLC event list (spawn processes)
- **MP_keymapFile:**
  Preset by ANILAM; do not change it.
  Path and name of the configuration file for the keyboard mapping.

MP_mainPgm
Path and file name of the PLC main program
Format: String max. 260 characters
Input: Path and file name of the PLC program
Example: %OEM%\BasisPgm\main.src

MP_pwmPgm
Path and file name of the PLC commissioning program
Format: String max. 260 characters
Input: Path and file name of the PLC program for commissioning the machine.
Example: %OEM%\PLC\SetUp.plc

MP_errorTable
Path and file name of the PET table
Format: String max. 260 characters
Input: Path and file name of the PLC error table
Example: %OEM%\PLC\LANGUAGE\ERR_TAB.PET

MP_errorText
Name of the text file for PLC error texts
Format: String max. 260 characters
Input: File name
Example: ErrorText.csv

MP_dialog
Name of the text file for PLC dialogs
Format: String max. 260 characters
Input: File name

Example: Dialog.csv

MP_softkeyProject

Path and name of the project file for PLC soft keys;

Format: String max. 260 characters

Input: Path and name of the file
Example: %OEM%\BasisPgm\Softkeys\softkeys.xrs

MP_compErrorTable

Path and file name of the error table of the PLC compiler
(Default set by ANILAM; do not change it.)

Format: String max. 260 characters

Input: Path and name of the file
Example: %SYS%\config\plccomp.ert

MP_compCfgFile

Path and name of the configuration file for the PLC compiler

Format: String max. 260 characters

Input: Path and name of the file
Example: %OEM%\BasisPgm\Programm\OEM.cfg

PLC Program Version

Settings in:	
System Versions CfgPlcVersion **plcVersion**	

The PLC program number, which is assigned by the machine tool builder and is displayed under MOD, is entered in **MP_plcVersion.**

MP_plcVersion

PLC software version; displayed version of the PLC program

Format: String

Input: PLC program version
Example: TNC320_BasisPgm_v1.5

Tables

Different types of tables are managed in the control, such as motor tables, datum tables, and tool tables.

Each table has a different setup, meaning a different number of columns, different column names, different dialogs for column entries, etc.

Each table type also has certain characteristics. A table type is identified by its file extension (e.g., ".T" for tool tables).

Therefore, tables with the same extension have the same characteristics.

Note: Tables of different types, meaning tables with different extensions, are not compatible with each other. This means that you cannot copy tables from one extension to another, or simply change extensions.

Table characteristics are set in the configuration editor.
The different table characteristics are set in **CfgTableProperties**, and the column characteristics in **CfgColumnDescription**.

The following topics are described:

- **Creating a New Table Type**
- **Creating a New Table with File Manager**
- **Inserting Additional Columns in an Existing Table**
- **Deleting Columns from an Existing Table**
- **Removing Column Names and Column Descriptions**
- **Symbolic Names for Tables**
- **Editing Tables Via the PLC**
- **Access to Tables Via SQL Commands**
- **Reference for Syntax Elements**
- **PLC Modules for the SQL Statements**

Creating a New Table Type

Settings in the configuration editor:	
System Paths CfgTablePath **path** System ProgramManager CfgFileType **unitOfMeasure** **standardEditor** **fileSize** **alternateEditor** TableSettings CfgTableProperties Key name for table **columnKeys** **primaryKey** Columns Key name for table column CfgColumnDescription **width** **unit** **initial** **minimum** **maximum** **charset** **unique** **readonly** **unitIsInch** CfgColumnText dialogText **dialogRes** **text** **info** **softkeyIcon** choice **value** dialog **dialogRes** **text** **info** lockValue **value** dialog **dialogRes** **text** **info**	

To create a new table type, proceed as follows:

- Specify a new file extension (System/ProgrammManager/CfgFileType)
- Create a table configuration (System/TableSettings/CfgTableProperties)
- Create a column configuration (System/TableSettings/columns)
- Create a new table using the file manager
- Insert rows into the table with the table editor

The following topics are described:

- **Specifying the File Extension**
- **Table Description**
- **Column Description**

Specifying the File Extension

The extension determines the type of table; keep the following in mind:

- Maximum length of three characters
- Only numbers or capital letters are permitted (this means no blank spaces or special characters)
- The extension may not already be used for other tables or types of files.
  In the configuration editor, under System/ProgramManager/CfgFileType and System/TableSettings/CfgTableProperties check whether the desired extension is displayed. The desired extension should not already display here.

The new file extension must be entered in the configuration editor so that a table with this extension can be opened by the table editor:

- Press the MOD key.
- Enter the code number 95148.
- Press the CONFIG DATA soft key.
- Select the object **System/ProgramManager/CfgFileType.**
- Press the INSERT soft key.
- Select **MP_unitOfMeasure;**
  If you want to be able to choose if the table or file is to contain values in mm or in inches (selectable via soft key when opening a new file), enter “UNIT_MMINCH.”
- Select **MP_standardEditor;**
  Select the input value TABLE EDITOR.

- Save the information with the END or SAVE soft key.

MP_unitOfMeasure

Alternative unit of measure for file/table

Format: Pull-down selection menu

Selection:

[UNIT_INDEPENDENT]

Input without unit of measure

[UNIT_MM]

Input in mm

[UNIT_INCH]

Input in inches

[UNIT_MMINCH]

Input in mm or inches

MP_standardEditor

Editor for file/table

Format: Pull-down selection menu

Selection:

[TEXT EDITOR]

Opens the text editor of the control when a file is selected.

[PROGRAM EDITOR]

Opens the NC program editor of the control when a file is selected.

[TABLE EDITOR]

Opens the table editor of the control when a file is selected.

Default: TEXT-EDITOR

Table Description

Specify in the configuration editor which columns are used in a table.

- Press the MOD key.
- Enter the code number 95148.
- Press the CONFIG DATA soft key.
- Select the object **System/TableSettings/CfgTableProperties.**
- Press the INSERT soft key.
- Enter the extension of the new table;
  Specify the file (memory file) in which the configuration data are to be saved (normally O:\config\oemtable.cfg)
- Confirm with **OK**.
- Select **MP_columnKeys.**
- Enter the first column name in the field [0].
  The name has the following structure: Table extension.Column name.
  The **column name** may not have any other periods, commas or special characters. Hyphens are permitted.
  For columns with the same meaning that are used in different tables, only the column name needs to be given. Examples of such columns are the columns predefined by ANILAM, such as “NAME or “NR”.
  The table extension must always be written in uppercase letters.
- Use the INSERT soft key to create more columns according to the above procedure.
- Confirm with **OK**.
- Select **MP_primaryKey** and enter the column name (name from MP_columnKeys).
- Confirm with **OK**.
- Save the information with the SAVE soft key.

MP_columnKeys

List of column names

Format: Array [0...] 0.. Counting of this value starts beginning at “0” and is incremented every cycle.

Input: xxx.xxx

Column designation with uppercase letters and the structure<Table extension>.<Columnn name>;
max. 20 characters

MP_primaryKey

Name of the column, based upon which the data is sorted in ascending order.

Format: String max. 18 characters

Input: <Column name>

The name of the column must be defined in MP_columnKeys.

Column Description

Description of the individual columns takes place in the configuration editor.

- Press the MOD key.
- Enter the code number 95148.
- Press the CONFIG DATA soft key.
- Select the object **System/TableSettings/Columns**.
- Press the INSERT soft key;
- The name of the new column is indicated (see column name in “Table description”).
- Select the object **System/TableSettings/columns/CfgColumnsDescription**.
- Press the INSERT soft key; confirm the suggestion with **OK**.
- Select **MP_width**; enter the max. column width (number of characters).
- Select **MP_unit**; enter the unit of measurement.
- Select **MP_initial** (optional parameter); enter the default value.
- Select **MP_minimum**; enter the minimum value for numerical values.
- Select **MP_maximum**; enter the maximum value for numerical values.

Note: You must enter minimum and maximum values!

- Select **MP_charset** (optional parameters); this defines the number of permissible characters. If the number is not defined, all characters are allowed.
- Select **MP_unique** (optional parameter), and insert this parameter if the column is to contain unambiguous values. If the attribute is not defined, the same values may display more than once in different rows.
- Select **MP_readonly;** protect data from access; The column representing the “primaryKey” of the table should be write-protected (set MP_readonly to TRUE).
- Select **MP_ unitIsInch**; specify the unit of measurement.
- Select **MP_choice** (optional).
  A list of value/text pairs can be defined here. Only these texts are then available in the table via a selection list (in the **Programming** mode of operation, the COLUMN NAME soft key opens the selection list). This stands for a value, which is then displayed. The text for the selection list can be entered directly (MP_text), or you can enter a link to a dialog table (MP_dialogRes).
- Select **MP_value** (optional), and enter the value for MP_dialog.
- Select **MP_dialog** (optional).
- **MP_dialogRes** (optional), leave the attribute empty if the text is not to be language-sensitive.
- Select **MP_text** (optional), and enter the text for MP_value.
  (keep MP_width in mind)
- Select **MP_lockValue** (optional).
  If the value entered in the column equals the value given here, the text in MP_text is displayed. It can no longer be edited. This way editing can be disabled depending on the value.
- Select **MP_value** (optional), and enter the value for MP_dialog.
- Select **MP_dialog** (optional).
- (MP_dialogRes in preparation).

- Select **MP_text** (optional); enter the text for MP_value. (keep MP_width in mind)
- Select **MP_CfgColumnText**; enter the dialog text for the columns here.
- Carry out the above procedure for all new columns, and save the information with the SAVE soft key.

The following machine parameters are used for defining the columns.

MP_width

Column width

Format: Numerical value

Input: 2 to 50 (column width of max. 50 characters)

Default: 2

MP_unit

Unit of measure for the column entries

Format: Pull-down selection menu

Selection:

[TEXT]
Text entry
[SIGN]
Algebraic sign + or –
[BIN]
Binary number
[DEC]
Decimal, positive, whole number (cardinal number)
[HEX]
Hexadecimal number
[INT]
Whole number
[LENGTH]
Length
[FEED]
Feed rate (mm/min or 0.1 ipm)
[IFEED]
Feed rate (mm/min or ipm)
[FLOAT]
Floating-point number
[BOOL]
Logical value
[INDEX]
Index with subindices

MP_initial

Value automatically entered in a column when a new table is created (optional).

Format: String

Input: max. 50 characters
NULL: No defined default value. This column can stay empty.
Value: Default value. During insertion of a new line, this value is assigned as default to the column.
If a default value other than NULL is given, then a valid value must always be entered in the column.

MP_minimum

Minimum permissible input value

Format: String

Input: max. 50 characters
(e.g., –99999.9999)
The minimum value is considered only for columns with numerical values or logical values. It defines the smallest permissible numerical input value or the text representing the logical value FALSE. For a value of the data types FLOAT, FEED, IFEED, or LENGTH, the given number of decimal places determines the number of decimal places saved for values in this column (e.g., 0.001 means 3 decimal places).

MP_maximum

Maximum permissible input value

Format: String

Input: max. 50 characters
(e.g., 99999.9999)
The maximum value is considered only for columns with numerical values or logical values. It defines the largest permissible numerical input value or the text representing the logical value TRUE. For a value of the data types FLOAT, FEED, IFEED, or LENGTH, the given number of decimal places determines the number of decimal places saved for values in this column, (e.g., 300.000 means 3 decimal places).

MP_charset

Number of permissible characters for text columns (optional)

Format: String

Input: max. 224 characters
The number of permissible characters is evaluated only for text columns. If this parameter is not defined, all characters are allowed; otherwise, only the characters listed here are allowed.

MP_unique

Defines whether only unambiguous values are allowed in the column (optional)

Format: Pull-down selection menu

Selection:

[TRUE]

Only unambiguous values allowed

[FALSE]

Values may occur more than once

MP_readonly

Write protection on column entry
If the attribute is set to TRUE, the value assigned when inserting the line cannot be changed. If the attribute is not set or set to FALSE, values may be overwritten.

Format: Pull-down selection menu

Selection:

[TRUE]

Values are write-protected

[FALSE]

Values may be overwritten

MP_unitIsInch

Values in inches (optional)
If lengths and feed rates are to be specified in the column in a definite unit of measure, enter TRUE here for values in inches and FALSE for values in mm. If the attribute is not set, the unit of measure is taken from the corresponding table.

Format: Pull-down selection menu

Selection:

[TRUE]

Column entry in inches

[FALSE]

Column entry in mm

If, in the table editor, you want to display language-sensitive texts for columns, you must insert a CfgColumnText data object. However, this object is not absolutely necessary. If the object is missing, the column name is shown in the dialog line in the table editor. When inserting, the same column name must be given as in CfgColumnDescripton.

Creating a New Table with File Manager

To create a new table of the desired type, proceed as follows:

- Change to the Programming operating mode.
- Press the PGM MGT key
- Press the **NEW** soft key.
- Enter the name of the new table. You can enter any desired name, but the name must not begin with a number. The file extension must correspond to the new type of table. The new table is opened in the table editor. The table does not yet have any cells.

Note: he table name must not begin with a number. **Always** begin the file name of a table with a letter (e.g., **M123.T**).
Reason:
The control internally manages the tables via an SQL server. The SQL name convention does not allow the use of file names beginning with a number.
If you enter a file name that begins with a number, the control issues an error message.

- Insert a line with the **INSERT LINE** soft key.
  The lines are sorted by the “primaryKey” column (see **MP_primaryKey**).
  No lines in this column may contain the same value. Therefore, you cannot insert a line whose value in the column “primary key” already is displayed in the table.
- Insert multiple lines with the **APPEND N LINES** soft key.

Inserting Additional Columns in an Existing Table

To insert additional columns in an existing table, proceed as follows:

- Select the configuration data under **System/TableSettings/CfgTableProperties.**
- Select the entry (key name for table) for the table to be expanded.
- In **MP_columnKeys,** enter the new column names.
  Insert the new columns at the desired places in the list so that they will display in the correct places in the table.
- Select the configuration data under **System/TableSettings/columns/ CfgColumnsDescription.**
- Enter the desired column description.
- Press the **SAVE** soft key to save the data.
- Select the table in which the columns are to be inserted by using the program manager in the **Programming** mode of operation.
- Press the MOD key, and enter the code number **555343;**
  The dialog box “Edit table characteristics” is displayed.
  All of the columns already assigned to this table type display after the  symbol.
  All of the newly defined columns not yet assigned to this table type display after the  symbol.
- Use the arrow keys to select the column to be inserted.
- Press the **INSERT COLUMN** soft key.
  This column is then marked with the symbol 
- Use the same procedure to mark other columns that are also to be inserted.
- Press the **INSERT ALL COLUMNS** soft key to mark all columns.
- Press the **SAVE** soft key to change the configuration of the selected table.
- Press END to open the changed table.
- The above steps must be followed for all tables of the same type.

Deleting Columns from an Existing Table

To delete columns from an existing table, proceed as follows:

- Select the table from which columns are to be deleted by using the program manager in the **Programming** mode of operation.
- Press the MOD key, and enter the code number 555343;
  The dialog box “Edit table characteristics” is displayed.
  All of the columns assigned to this table type display after the  symbol;
- Use the arrow keys to select the column to be deleted.
- Press the **DELETE COLUMN** soft key.
  This column is then marked with the symbol 
- Use the same procedure to mark other columns that are also to be deleted.
- Pressing the **SAVE** soft key changes the configuration of the selected table, meaning that the column has been deleted.
- The above steps must be followed for all tables of the same type.

Removing Column Names and Column Descriptions

Proceed as follows using the configuration editor to remove column names and column descriptions no longer required:

- Press the MOD key.
- Enter the code number 95148.
- Press the **CONFIG DATA** soft key.
- Select **System/TableSettings/CfgTableProperties**.
- Select the table type from which the columns are to be removed.
- Use the **DELETE** soft key to remove the desired columns from **MP_columnKeys**.
- Select **System/TableSettings/CfgColumnDescription.**
- Place the cursor on the column description to be removed.
- Press the **DELETE** soft key.
- Press the **SAVE** soft key to save the data.
- No more tables containing the “old” columns can be opened.

Symbolic Names for Tables

For access via SQL commands, tables are identified with a symbolic name and a file name including the path for the table characteristics.
Direct use of these paths, such as from cycles, has the disadvantage that if the drives or paths are changed, or if other tables are selected, the cycles must be changed.

In order to avoid this disadvantage, symbolic table names are used. These table names are place holders for the actual table name and path. When accessing a table, the control replaces the symbolic table name with the real table path and name.

Symbolic table names are saved in the control's configuration data in the **CfgTablePath** object (new key name). A symbolic table name should consist only of capital letters.

The logic names do not have to be in any certain format. Any name can refer to any table or table type.

The table being referenced does not even have to exist at the time that the logic table name is given. It can also be created afterwards.

Proceed as follows for specifying a symbolic table name:

- Press the MOD key.
- Enter the code number 95148.
- Press the **CONFIG DATA** soft key.
- Select the **System/Paths/CfgTablePath** object.
- Press the **INSERT** soft key.
- Enter the symbolic table name (key name) and confirm with **OK**.
- In **MP_path** enter the path for the file in which the table characteristics are to be saved. This is generally the file PLC:\config\oemtable.cfg.
- Save the information with the **SAVE** soft key.

MP_path

Path for tables

Format: String

Input: max. 84 characters
Path/name consisting of device name, up to 6 directories, file name and extension
Example:
%USR%\table\tool.t

Editing Tables Via the PLC

You can also read tables and overwrite individual fields via PLC modules.

Note: The following modules must be called in a submit or spawn job.
When entering the column names, pay attention to the case of the letters (whether they are small or capital).

The following topics are described:

- **Module 9240 Open a file**
- **Module 9241 Close a file**
- **Module 9242 Positioning in a file**
- **Module 9243 Reading from an ASCII file line by line**
- **Module 9244 Writing to an ASCII file line by line**
- **Module 9245 Read a field in a table**
- **Module 9255 Read a data field in a table**
- **Module 9246 Write to a field in a table**
- **Module 9256 Write to a datafield in a table**
- **Module 9247 Search for a condition in a table**
- **Module 9249 Read and reset ‘errno’**

Module 9240 Open a file

The module opens the file for access via the PLC. The “file handle” is created. This is a number which must be given for each subsequent access (such as in another PLC module).

Up to eight files may be open at once. However, the file can only be accessed by the process that opened it (SUBMIT job or SPAWN job). A file can also be opened more than once. If you want to prevent the file from being opened in more than one process, use the “lock file” mode.

To maintain a high processing speed, the file should be opened with the “BUFFERED” option for reading and writing ASCII texts. In this mode, a part of the file is buffered in the main memory. This mode is not permitted for tables.

Files should not be kept open unnecessarily, since they cannot be erased by the file manager during this time.
Ending a process (EM in a submit job) also closes all files opened by the process. The same applies if a process is canceled by a CAN instruction or by a renewed compiling of the PLC program.

The file handle must be saved in a double word.

To append data to an existing file, set bit 0=1 (reading and writing) **and** bit 2=0 (record oriented).

Call:

PS B/W/D/K<>Mode>
Bit 0=0: Read only
Bit 0=1: Read and write
Bit 1=0: Do not lock file
Bit 1=1: Lock file
Bit 2 = 0: Record oriented (for tables)
Bit 2 = 1: Buffered (for ASCII files)

PS B/W/D/K/S<>String with file name>
Complete path, file name and extension

CM 9240

PL D <>File handle>
Number for use in other modules
–1: Error code in NN_GenApiModuleErrorCode (W1022)

Error code:

Marker	Value	Meaning
NN_GenApiModuleErrorCode (W1022)	1	Impermissible mode
	3	Incorrect string number
	7	File could not be opened
	20	Module was not called in a submit job or spawn job

Module 9241 Close a file

With this module you close a file that has been opened with Module 9240. You must close the file in the process (submit job or spawn job) in which you opened it.

Call:

PS D <>File handle>
Number from Module 9240

CM 9241

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	File was closed
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Incorrect file handle
	20	Module was not called in a submit job or spawn job

Module 9242 Positioning in a file

With this module you change the position of the cursor in a file opened with Module 9240. The new position is provided as the result from Module 9242.
If the file was opened in the “record oriented” mode (tables), the cursor is positioned line by line.
If the file was opened in the “buffered” mode, the cursor is positioned character by character.
If you indicate a position before the beginning or after the end of the file, the cursor is positioned at the beginning or end of the file, respectively. The addressing of the new position is relative to the beginning or end of the file, or to the current position. You can interrogate the current position by transferring the position value zero relative to the current position.

Call only in a submit job or spawn job.

Call:

PS	D	<>File handle>
		Number from Module 9240
PS	B/W/D/K	<>Desired position>
PS	B/W/D/K	<>Mode>
		0: Position relative to the file beginning
		1: Position relative to the current position
		2: Position relative to the file end
CM	9242	
PL	B/W/D/K	<>New position>
		–1: Error code in NN_GenApiModuleErrorCode (W1022)

Error code:

Marker	Value	Meaning
NN_GenApiModuleErrorCode (W1022)	1	Impermissible mode
	2	Incorrect file handle
	7	File system error
	20	Module was not called in a spawn job or submit job

Module 9243 Reading from an ASCII file line by line

To read from a table, use Module 9245.

The module reads a line from the ASCII file opened with Module 9240, and writes it to a PLC string.
Depending on if you opened the file with the “buffered” option, different processing times will result (buffered is faster).

The module reads up to a line break (line feed, ‘\n’), with a maximum of 127 characters. The line break is not saved in the result string, but is counted for the number of characters read.

The result is undefined when reading non-ASCII-coded files.

A certain amount of binary data is saved in the target string, but cannot be used.

Call:

PS	D	<>File handle> Number from Module 9240
PS	B/W/D/K	<>String number with result> 0 to 7
CM	9243	
PL	B/W/D	<>Number of read bytes> >0: Line has been read 0: File end has been reached –1: Error code in NN_GenApiModuleErrorCode (W1022)

Error code:

Marker	Value	Meaning
NN_GenApiModuleErrorCode (W1022)	2	Incorrect file handle
	3	Incorrect string number
	7	File system error
	20	Module was not called in a spawn job or submit job

Module 9244 Writing to an ASCII file line by line

To write to a table, use Module 9246.

The module writes a line from a PLC string to an ASCII file already opened by Module 9240 in “buffered” mode.

If file is opened in “buffered” mode:

- Processing time is shorter.
- Files are saved to the hard disk only if more than 512 bytes are overwritten in several calls, or if the file is closed.
- The number of data specified in the transfer string is overwritten.

If file is opened in “record oriented” mode:

- Processing time is longer.
- The data is immediately saved to the hard disk.
- Exactly one line is overwritten. If there is a difference in length, the following data is displaced by the difference.

Call:

PS	D	<>File handle> Number from Module 9240
PS	B/W/D/K/S	<>String number, source data> 0 to 7
CM	9244	
PL	B/W/D	<>Number of written bytes (including LF)> –1: Error code in NN_GenApiModuleErrorCode (W1022)

Error code:

Marker	Value	Meaning
NN_GenApiModuleErrorCode (W1022)	2	Incorrect file handle
	3	Incorrect string number
	7	File system error
	20	Module was not called in a spawn job or submit job

Module 9245 Read a field in a table

The module reads a data field from a table opened by Module 9240 in “recorded” mode into a string. The data field is addressed by the field name and the line number.

To maintain a high processing speed, multiple lines should be read in ascending order.

Pay attention to the upper/lower case of field names.

If an error occurs, the content of the target string is undefined.

The module provides the contents as a string.

Call:

PS	D	<>File handle> From Module 9240
PS	B/W/D/K	<>Line> 0 to 65 535
PS	B/W/D/K/S	<>String number, column name> 0 to 15
PS	B/W/D/K/S	<>String number, result> 0 to 15
CM	9245	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Field was read
	1	Error code in W1022
NN_GenApiModuleErrorCode (W1022)	1	Line does not exist in table
	2	Incorrect “file handle” or table was opened in “buffered” mode
	3	Impermissible string numbers
	7	Module could not read from the table
	20	Module was not called in a spawn job or submit job
	29	The opened file is not a table (extension: .TAB, .P)
	30	Column name not found

Module 9255 Read a data field in a table

The module reads a data field from a table opened by Module 9240 in “recorded” mode as an integer value. The data field is addressed by the field name and the line number. To maintain a high processing speed, multiple lines should be read in ascending order.

Pay attention to the upper/lower case of field names.

If an error occurs, the number value of the result is undefined.

The function can only be used on fields containing numerical values.

If digits can be entered after the decimal point in the selected field, the numerical value is standardized to the last digit after the decimal point, meaning that for *n* digits after the decimal point, the value is multiplied by 10^*n*.

Call:

PS	D	<>File handle> From Module 9240
PS	B/W/D/K	<>Line> 0 to 65 535
PS	B/W/D/K/S	<>String number, column name> 0 to 15
CM	9255	
PL	B/W/D	<>Result>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Field was read
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)		See **Module 9245**

Module 9246 Write to a field in a table

The module writes a string to a data field in a table opened by Module 9240 in “recorded” mode. The data field is addressed by the field name and the line number.

To maintain a high processing speed, multiple lines should be written in ascending order.

Pay attention to the upper/lower case of field names.

The field defined by the column name and line number is overwritten.

The module transfers a string.

Call:

PS	D	<>File handle> From Module 9240
PS	B/W/D/K	<>Line> 0 to 65 535
PS	B/W/D/K/S	<>String number, column name> 0 to 15
PS	B/W/D/K/S	<>String number, contents to be written> 0 to 15
CM	9246	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Field was written to
	1	Error code in W1022
NN_GenApiModuleErrorCode (W1022)	1	Line does not exist in table
	2	Incorrect “file handle” or table was opened in “buffered” mode
	3	Impermissible string numbers
	6	Table is write-protected
	7	Not a numerical field (Module 9256)
	11	The transferred value cannot be saved to the addressed field. Incorrect format.
	30	Column name not found

Module 9256 Write to a datafield in a table

The module writes an integer value to a data field in a table opened by Module 9240 in “recorded” mode. The data field is addressed by the field name and the line number.

The field defined by the column name and line number is overwritten.

This module can be used only for an integer. Values with decimal places are written without the decimal point.

Call:

PS	D	<>File handle> From Module 9240
PS	B/W/D/K	<>Line> 0 to 65 535
PS	B/W/D/K/S	<>String number, column name> 0 to 15
PS	B/W/D/K	<>Numerical value to be written>
CM	9256	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Field was written to
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)		See **Module 9246**

Module 9247 Search for a condition in a table

In a table opened by Module 9240 in “recorded” mode, the function searches for a data record which fulfills one or more conditions. The conditions are formulated with a subgroup of the System Query Language (SQL) data bank language.

Pay attention to the case of the letters (whether they are small or capital) in the commands and column names.
If you indicate a starting line, the module can search for several suitable field entries.

Permissible SQL commands:

Command	Meaning
+,–,*,/	Arithmetical operators
NOT, AND, OR	Logical operators
<,>, <=, >=, ==, <>	Comparisons
LIKE ´abc´	Text comparison
LIKE ´_abc%´	Partial string
()	Parentheses
MIN(column name)	Minimal value from the column
MAX(column name)	Maximum value from the column

Example:

Search in a pallet table for the line with the NC program 1.H and the set datum X=–10. String contents:

WHERE (PAL/PGM LIKE´PGM´) AND (NAME LIKE´1.H´) AND (X==–10)

Call:

PS	D	<>File handle> From Module 9240
PS	B/W/D/K	<>Starting line> 0 to 65 535
PS	B/W/D/K/S	<>String number of condition or string with condition> 0 to 7
CM	9247	
PL	B/W/D	<>Line that fulfills the condition> –1: Error code in NN_GenApiModuleErrorCode (W1022)

Error code:

Marker	Value	Meaning
NN_GenApiModuleErrorCode (W1022)	1	Start line does not exist in table
	2	Incorrect “file handle” or table was opened in “buffered” mode
	3	Impermissible string numbers
	7	Module could not be read from the table
	20	Module was not called in a spawn job or submit job
	29	Incorrect file format
	30	Column name not found
	31	Syntax error in the transferred condition
	32	No data record found that fulfills the condition

Module 9249 Read and reset ‘errno’

This function reads the error status ‘errno’ of the operating and file system, and resets this status to 0. This status can be used for more accurate determination of the errors in certain modules (e.g., 9240.9242.9243.9244).

Constraints:

- ‘errno’ always contains the code of the most recent error. The variable is only cleared with Module 9249.
- ‘errno’ is only valid within a PLC process (submit job), and is separately present for each process.
- The definitions of the C programming language (UNIX compatible) as well as specific expansions by ANILAM are valid for the contents of ‘errno’. A separate documentation exists for this.
- In order to use ‘errno’ to see if an error has occurred, the module must be called before a program sequence in order to clear the contents of ‘errno’.

Call:

```
CM      9249
PL      W/D   <>errno>
```

Access to Tables Via SQL Commands

The control software access the tables via a server. Accesses from the NC program, the PLC program, and the editor are synchronized with each other, and locked against each other. This server is controlled with SQL commands.

The server is based on a transaction model which performs read or write operations for more than one row or column in such a way that they cannot be interrupted by other read or write operations.

A transaction consists of the following steps

- Selecting data
- Reading data
- Editing data (if required)
- Confirming or rejecting the changes

The following topics are described:

- **Selecting the Data**
- **Reading Data**
- **Editing Data**
- **Concluding the Transaction**

Selecting the Data

To select data, use the SQL statement **SELECT.** The SELECT statement finds the columns of the rows of a table, which satisfy the entered **WHERE** condition.

The table is usually designated by a synonym. These synonyms are configuration data which can also be created, assigned, and deleted via the SQL server. In addition, you can specify a table by using its path name, provided that you enclose the path name in single quotation marks.

The result of the query is a result set which is created and managed by the SQL server. The SQL server assigns a **handle** to the result set, which enables you to identify the result set for reading/editing data and committing the transaction. The handle is the result of the query, which is visible in the NC program. The value 0 indicates an invalid handle (i.e., it was not possible to create a result set for that query). If no rows that satisfy the specified condition are found, an empty result set is created and assigned a valid handle.

The ORDER BY function sorts the rows in the result set according to the values of a column in increasing or decreasing order. The row numbers in the result set are listed in increasing order and are not related to the row numbers in the table file.

You can create a dynamic query, for example, by replacing the comparison value in the **WHERE** condition with a reference to a Q parameter. To do this, program a colon (':') instead of the value and enclose the Q parameter in single quotation marks (e.g., 'Q2'). During interpretation this expression is replaced by the current content of the Q parameter.

The **FOR UPDATE** function locks the selected rows for the duration of the transaction. As a result, third parties can only read the data, but cannot edit them. However, this query cannot be executed if it is to access data that have already been locked.

Reading Data

To be able to read the selected data, use **SQL BIND** to bind the data of a column to a parameter to be accessed from the NC program.

The data can be bound to a Q parameter (e.g., SQL BIND Q5 "TOOL.L") or directly to a system datum of the interpreter (e.g., SQL SYSBIND ID350 NR52 IDX 1 "PROBE.L"). When a row is read, the content of the respective columns is stored in the bound parameter. To cancel the binding, it must be programmed again without specifying a column. The bindings are globally effective and must be canceled explicitly. The bindings must be unambiguous. The attempt to bind more than one parameter to a column will fail.

With SQL FETCH, the data of the result set are read row by row. The result set is identified by the indicated **handle.** It is therefore possible to read various result sets alternately. An index in the range from 0 to n can be defined for the result-set row to be read. If no index is specified, the first row is read. The result of the function is 0 if data was read and stored in the bound parameters. The result of the function does not equal 0 if no data was read.

Editing Data

If you want to edit data, you first have to edit the data in the bound parameters. The **SQL UPDATE** command, copies the data into the result set which is identified by the **handle.** An index in the range from 0 to n can be defined for the result-set row to be written. If no index is specified, the first row is edited. The result of the function is 0 if the data have correctly been transferred to the result set. The result of the function is not 0 if an error occurred during transfer.

Afterwards the changes are only visible within the edited result set. If you reread the edited row, the edited data are shown; if you enter a new query and reread the row, the original data will be shown. Third parties will still see the original data. The data is not transferred to the table until the transaction has been committed. Thus it is possible to edit more than one row in a consistent manner in one transaction.

Concluding the Transaction

Be sure to complete every transaction. This way the resources assigned in the SQL server are released for the result set.
After completing the transaction with **SQL COMMIT,** all changes are transferred from the result set into the table.
To cancel all changes, complete the transaction with **SQL ROLLBACK.**
Once transferred, these changes cannot be undone. After the transaction has been completed, the rows locked during the selection are unlocked.
The transaction to be completed is identified by the **handle.** After the transaction has been completed successfully, the **handle** becomes invalid and cannot be used for accessing data any longer.
The result of the **SQL COMMIT** function will not be 0 if the edited data could not be transferred into the table file. This happens when edited rows were not locked during the selection and are locked by third parties at the time of commitment. The SQL server first checks whether all edited rows may be rewritten before it transfers the first change. The isolation of accesses ensures that the effects of the transaction are invisible to others until the transaction is committed. It may, however, happen that errors occur while a file is being accessed.

The **SQL ROLLBACK** function enables you to remove all rows, except for the indicated row, from the result set by defining an index. As a result, the changes made to the removed rows are cancelled. This is especially useful if you first select and, as a consequence, lock more than one row, but then decide that you want to edit only one row. The other rows can then immediately be released. The handle remains effective until the transaction has been completed for all rows. If no index or an invalid index has been defined, the entire transaction is completed. The result of the function does not equal 0 if an error has occurred.

Reference for Syntax Elements

The following topics are described:

- BNF Notation
- Context Elements
- SQL Parameters
- SQL-HANDLE
- SQL Index
- SQL Column
- SQL Statement
- SQL
- SQL BIND
- SQL SYSBIND
- SQL FETCH
- SQL UPDATE
- SQL COMMIT
- SQL ROLLBACK
- Command Options for SELECT and UPDATE
- WHERE
- ORDER BY
- FOR UPDATE
- FOR NOTIFICATION
- SQL Commands
- SELECT
- UPDATE
- INSERT
- RENAME TABLE
- CREATE TABLE
- ALTER TABLE
- COPY TABLE
- DROP TABLE
- RENAME COLUMN
- CREATE SYNONYM
- ALTER SYNONYM
- DROP SYNONYM
- Application Example for SQL commands
- Read Data from Table
- Write Data to Table

BNF Notation

The following specifies the individual syntax elements in BNF notation. The individual NC blocks are listed in alphabetical order.

A specified syntax element is identified by a name in italics.

Individual characters are enclosed in single quotation marks; entire code words are plain text and are not enclosed in single quotation marks.

A single simple expression is enclosed in square brackets [ ], whereas a multiple expression is enclosed in braces { }. Two expressions separated by the "|" symbol indicate alternative options.

Expressions may contain letters, numbers, and symbols.

- uppercase := 'A'–'Z'
- lowercase := 'a'–'z'
- digit := '0'–'9'
- index := digit { digit }
- number := ['+' | '-'] digit { digit } ['.' digit { digit }
- literal := "" { uppercase | lowercase | digit | symbol } ""
- name := uppercase { uppercase | lowercase | digit | '$' | '#' | '_' }

Context Elements

The following elements can only occur within the context of an NC block.

SQL Parameters

As in the functions FN17 and FN18, system parameters can be bound to the column name of a table.

Definition

- system-group := ID index
- system-number := NR index
- system-index := IDX index
- system-parameter := system-group system-number system-index
- sql-parameter := q-parameter | system-parameter

SQL-HANDLE

The SQL handle identifies the result set of a previous SQL query. Only values assigned by the SQL server are valid handles.
The value 0 identifies an invalid handle.

Definition

- sql-handle := HANDLE q-number

SQL Index

The SQL index identifies the row from the result set. The indices start at 0 and are increasing in order.
If no index is specified, the first row from the result set is automatically transferred.

Definition

- sql-index := INDEX (index | q-number)

SQL Column

The name of the table and of the column to be bound is to be enclosed in double quotation marks.

Definition

- table-name := name
- column-name := name
- sql-column := "" table-name '.' column-name ""

SQL Statement

Enclose the statement to be executed in double quotation marks. A reference to a Q parameter can be used within a statement.
Enclose the Q parameter in single quotes after a colon.
The interpreter replaces this sequence with the value of the Q parameter.

Definition

- sql-replacement:= ':' "' q-parameter "'
- sql-statement:= literal

Example:
SQL Q5 " SELECT L, R FROM TOOL WHERE N = :'Q2' "

SQL

The NC block SQL defines an SQL statement to be executed. The SQL HANDLE, which will enable you to access the data at a later date, is stored in the specified parameter. It is valid until the transaction has been committed or cancelled for all rows of the result set.

Definition

- sql-execute := SQL q-parameter sql-statement

Example:
SQL Q5 " SELECT L, R FROM TOOL WHERE N = :'Q2' "

SQL BIND

The NC block SQL BIND binds a Q parameter to a column of a table. If you redefine the NC block without specifying a column, the binding will be cancelled.
Otherwise, the binding remains in effect until the current subprogram or cycle is completed.

Definition

- sql-bind:= SQL BIND q-parameter [sql-column]

Example:
SQL BIND Q63 "TCHPROBE.OFFS0"

SQL SYSBIND

The NC block SQL SYSBIND binds a system parameter and a column of a table. If you redefine the NC block without specifying a column, the binding will be cancelled. As with SQL BIND, if not otherwise specified, the binding remains in effect until the subprogram or cycle is completed.

Definition

- sql-bind := SQL SYSBIND system-parameter [sql-column]

Example:
SQL SYSBIND Q63 "ID50 NR1 IDX1"

SQL FETCH

The NC block SQL FETCH reads a row from the result set of an SQL query and assigns the data to the bound parameters. If the values in the table are expressed in inches, lengths and feed rates are converted into millimeters during the reading process. The values in the bound parameters are always assumed to be metric. As with FN18, this also applies if the current program is entered in inches. If no index has been specified, the first row of the result set is transferred. The specified Q parameter is assigned a return code. If the command has been completed successfully, the Q parameter is assigned a zero. If not, it is assigned a one.

Definition

- sql-fetch:= SQL FETCH q-parameter sql-handle [sql-index]

Example:
SQL FETCH Q80 HANDLE Q5 INDEX0

SQL UPDATE

The NC block SQL UPDATE assigns the data from the bound parameters to the corresponding rows or columns of the table. If the values in the table are expressed in inches, lengths, and feed rates are converted into millimeters before the assignment process. The values in the bound parameters are always assumed to be metric. As with FN17, this also applies if the current program is entered in inches.
The specified Q parameter is assigned a return code. If the command has been completed successfully, the Q parameter is assigned a zero. If not, it is assigned a one.

Definition

- sql-update := SQL UPDATE q-parameter sql-handle [sql-index]

Example:
SQL UPDATE Q80 HANDLE Q5 INDEX0

SQL COMMIT

The NC block SQL COMMIT cancels locks on table rows or table columns. Edited table data are permanently transferred through SQL COMMIT.
The specified Q parameter is assigned a return code. If the command has been completed successfully, the Q parameter is assigned a zero. If not, it is assigned a one.

Definition

- sql-commit := SQL COMMIT q-parameter sql-handle

Example:
SQL COMMIT Q80 HANDLE Q5

SQL ROLLBACK

The NC block SQL ROLLBACK undoes a transaction. In particular, the lock on rows in an SQL statement "SELECT ... FOR UPDATE" is canceled.
The specified Q parameter is assigned a return code. If the command has been completed successfully, the Q parameter is assigned a zero. If not, it is assigned a one.
If required, you can specify in the index the row for which the transaction is to take effect.

Definition

- sql-rollback := SQL ROLLBACK q-parameter sql-handle [sql-index]

Example:
SQL ROLLBACK Q80 HANDLE Q5

Command Options for SELECT and UPDATE

Command options allow you to define conditions, sorting sequences, and locks that modify the effect of a command.

The following topics are described:

- **WHERE**
- **ORDER BY**
- **FOR UPDATE**
- **FOR NOTIFICATION**

WHERE

The WHERE option limits the effect of a command to the rows of a table which satisfy the specified condition.

Definition

- where-option:= WHERE condition

ORDER BY

The ORDER BY option influences the sequence of rows in the result set. At present, it is only possible to sort by column (default ASC).

Definition

- order-option:= ORDER BY column [ASC | DESC]

FOR UPDATE

The FOR UPDATE option already locks the rows during selection (pessimistic locking). Without the FOR UPDATE option, the selected rows are not locked until the COMMIT command is executed (optimistic locking).

Definition

- update-option:= FOR UPDATE

FOR NOTIFICATION

The FOR NOTIFICATION option monitors the table for changes.
FOR NOTIFICATION provides the client with a result containing information on the change.

Definition

- lock-option := FOR UPDATE | FOR NOTIFICATION

SQL Commands

The following topics are described:

- **SELECT**
- **UPDATE**
- **INSERT**
- **RENAME TABLE**
- **CREATE TABLE**
- **ALTER TABLE**
- **COPY TABLE**
- **DROP TABLE**
- **RENAME COLUMN**
- **CREATE SYNONYM**
- **ALTER SYNONYM**
- **DROP SYNONYM**

SELECT

In a SELECT statement a list of the columns to be selected and the table preceded by the keyword FROM must be specified. In addition, it may contain a condition with the keyword WHERE, a sorting sequence with the keyword ORDER BY and a command for pessimistic locking with the keyword FOR UPDATE.

Definition

- select-list := '*' | colum-list
- select-option:= [where-option] [order-option] [lock-option]
- select-statement := SELECT select-list FROM table select-option

Examples:

SELECT * FROM TOOL WHERE RT == 5 AND LOCK <> 1 ORDER BY TIME

SELECT TIME FROM TOOL WHERE NR==7 FOR UPDATE

SELECT L,R,R2 FROM 'OEM:\TOOL.T' WHERE NAME LIKE 'T1999'

UPDATE

In an UPDATE statement, the table and the columns to be edited preceded by the keyword SET must be specified. Furthermore, it may contain a condition with the keyword WHERE. If the WHERE condition is not specified, all rows are edited.

Definition

- assignment := column '=' expression
- update-list := assignment { ',' assignment }
- update-option:= [where-option]
- update-statement := UPDATE table SET update-list update-option

Examples:

UPDATE TOOL SET LOCK = 1 WHERE RT == 5 AND LOCK <> 1

UPDATE TOOL SET TIME = 0, LOCK = 0

UPDATE 'OEM:\TOOL.T' SET TIME = MAXTIME WHERE NAME LIKE 'T1999'

INSERT

In an INSERT statement, the table and the values to be set which are to be enclosed within parentheses, separated by commas and preceded by the keyword VALUES must be specified. Be sure to assign all columns. The INSERT command appends a new row to the table. It is not possible to insert a new row between two rows.

Definition

- insert-list := '(' expression-list ')'
- insert-statement:= INSERT INTO table VALUES insert-list

Examples:

INSERT INTO TOOL VALUES (9.1,'T2000'.0.1000)

INSERT INTO 'OEM:\TOOL.T' VALUES (9.1,'T2000'.0.1000)

RENAME TABLE

The name of a table file is edited. If a logical table name is specified, the file identified by the name will be edited. Make sure that the name of the target file does not already exist. The name stored internally will be edited accordingly.

With this command, the previous table is copied into a new table. Then the previous table is deleted. This command allows you to move a table to another directory.

Definition

- rename-table-statement:= RENAME TABLE table TO table

Examples:

RENAME TABLE TOOL TO 'OEM:\TOOL.T'

RENAME TABLE 'OEM:\TOOL.T' TO 'OEM:\WERKZEUG.T'

CREATE TABLE

A CREATE TABLE statement creates a new table (new table file). Make sure that the specified table name does not already exist. The names of the columns to be inserted are given as a list. The properties of the columns are read from the configuration. A column configuration for each column name must be available for this. The column width is determined from the width configured for a column or from the length of the column name, depending on which width is larger.

If you enter an asterisk * instead of the list of column names, all the columns defined in the configuration for this type of table (table extension) will be used.

The ASINCH option is used to specify whether a table is to contain values in inches. As a result, inch-sensitive columns (defined in the column configuration) are created as inch columns in the table. If this option is not specified, the respective columns are created as millimeter columns.

With the LOCALCONFIG option, the properties of the table are stored locally in the table.

Definition

- create-list := '*' | column-list
- create-table-statement:= CREATE TABLE table '(' create-list ')' [ASINCH] [LOCALCONFIG]

Examples:

CREATE TABLE TOOL(L,R,R2,DL,DR,DR2,TL,RT,TIME1,TIME2, CUR_TIME,PLC) ASINCH

CREATE TABLE 'OEM:\TOOL.T' (L,R,R2,DL,DR,DR2,TL,RT,TIME1,TIME2, CUR_TIME,PLC)

CREATE TABLE *

ALTER TABLE

ALTER TABLE modifies the properties of a table. This option enables you to add or delete columns, and to modify the properties of columns. When you add columns or modify the properties of columns, the new properties are read from the configuration. When column properties are modified, the values are not modified. If the width of the new column is larger than the width of the previous one, the column is extended to the new width. If the width of the new column is smaller, however, the column is not changed, so as to avoid any loss of data.

A new column is inserted in the table at the position which is determined from the columns defined for the respective type of table (table extension) in the configuration. The sequence of columns in the table is derived from the configuration data.

Definition

- alter-table-options:= ADD | MODIFY | DROP
- alter-table-statement:= ALTER TABLE table alter-table-options '(’column-list ')'

Examples:

ALTER TABLE TOOL ADD (DOC)

ALTER TABLE 'OEM:\TOOL.T' MODIFY (L)

COPY TABLE

COPY TABLE copies the table into a new table. Make sure that the name of the target file does not already exist. The name stored internally will be modified accordingly. It is possible to specify logical table names.

Definition

- copy-table-statement:= COPY TABLE table TO table

Examples:

COPY TABLE TOOL TO 'OEM:\TOOL.T'

COPY TABLE 'OEM:\TOOL.T' TO 'OEM:\WERKZEUG.T'

DROP TABLE

DROP TABLE deletes an existing table file. If a logical table name is specified, the file identified by the name will be deleted.

Definition

- drop-table-statement:= DROP TABLE table

Examples:

DROP TABLE TOOL

DROP TABLE 'OEM:\TOOL.T'

RENAME COLUMN

RENAME COLUMN changes the name of an existing column. The properties of the column are not changed. If the configuration of the table is stored locally in the table, the name of the column is also changed in the respective configuration data.

Definition

- rename-column-statement:= RENAME COLUMN table '(' column-list ')' TO '(' column-list ')'

Example:

RENAME COLUMN TOOL (DR2) TO (DIR)

CREATE SYNONYM

CREATE SYNONYM creates a new logical table name. It is not necessary that the table file identified by the logical name already exists.

Definition

- create-synonym-statement:= CREATE SYNONYM table-name FOR table-literal

Example:

CREATE SYNONYM OUTIL FOR 'OEM:\TOOL.T'

ALTER SYNONYM

ALTER SYNONYM assigns another table file to the logical name.

Definition

- alter-synonym-statement:= ALTER SYNONYM table-name TO table-literal

Examples:

ALTER SYNONYM TOOL TO 'OEM:\WERKZEUG.T'

DROP SYNONYM

DROP SYNONYM removes a logical name. The table file identified by the logical name will not be removed.

Definition

- drop-synonym-statement:= DROP SYNONYM table-name

Examples:

DROP SYNONYM OUTIL

Application Example for SQL commands

The following topics are described:

- **Read Data from Table**
- **Write Data to Table**

Read Data from Table

To determine the positions in a measuring cycle, calibration data of a touch probe are to be transferred from the tchprobe.tp table.
The center offsets are indicated in the OFFS0 and OFFS1 columns.

- BIND links the Q parameters with the column names:
  SQL BIND Q63 "TCHPROBE.OFFS0"
  SQL BIND Q64 "TCHPROBE.OFFS1"
- SELECT chooses those columns from the table (TNC:\table\tchprobe.tp) that are to be assigned to the active touch probe (WHERE ACTNR==1):
  SQL Q5 "SELECT OFFS0,OFFS1 FROM 'TNC:\table\tchprobe.tp' WHERE ACTNR==1"
- FETCH reads a row from the result set (HANDLE Q5) and assigns the data to the bound parameters. With INDEX0, the first row of the selected data is read:
  SQL FETCH Q80 HANDLE Q5 INDEX0
- Safety check (Q80 is equal to zero?)
- The data are assigned to Q parameters.
  Q43 = Q63
  Q44 = Q64
- ROLLBACK releases the selected data.
  SQL ROLLBACK Q80 HANDLE Q5
- Safety check (Q80 is equal to zero?)
- Following that, the bindings are undone:
  SQL BIND Q63
  SQL BIND Q64

Write Data to Table

The triggering touch probe was newly calibrated. The Q parameters Q891, Q798, and Q799 contain the values determined for radius, center offset in the principal axis and center offset in the secondary axis.
The following SQL commands update the table entries with the calibration data.

- BIND links the Q parameters with the column names:
  SQL BIND Q891 "TCHPROBE.R0"
  SQL BIND Q798 "TCHPROBE.OFFS0"
  SQL BIND Q799 "TCHPROBE.OFFS1"
- SELECT chooses those columns from the table (TNC:\table\tchprobe.tp) that are to be assigned to the active touch probe (WHERE ACTNR==1):
  SQL Q5 "SELECT R0,OFFS0,OFFS1 FROM 'TNC:\table\tchprobe.tp' WHERE ACTNR==1"
- FETCH reads a row from the result set (HANDLE Q5) and assigns the data to the bound parameters. With INDEX0, the first row of the selected data is read:
  SQL FETCH Q80 HANDLE Q5 INDEX0
- Safety check (Q80 is equal to zero?)
- UPDATE writes the values from the bound Q parameters into the table:
  SQL UPDATE Q80 HANDLE Q5 INDEX0
- Safety check (Q80 is equal to zero?)
- COMMIT then makes the changes to the table permanent:
  SQL COMMIT Q80 HANDLE Q5
- Safety check (Q80 is equal to zero?)
- Following that, the bindings are undone:
  SQL BIND Q891
  SQL BIND Q798
  SQL BIND Q799

Note: The SQL commands FETCH, UPDATE, COMMIT, and ROLLBACK assign a return code to the Q parameters.
If the command has been completed successfully, the Q parameter is assigned a zero. If not, it is assigned a one.
It is advisable to perform a safety check after these commands.

PLC Modules for the SQL Statements

The following topics are described:

- Module 9440 Open a transaction
- Module 9441 Conclude and close a transaction
- Module 9442 Seek a record in the result set
- Module 9443 Fetch a record from the result set
- Module 9444 Change a record in the result set
- Module 9445 Read a single value from a table
- Module 9447 Delete record from result set
- Module 9448 Load a column description
- Module 9449 Extract a value from a comma separated list
- Module 9450 Execute an SQL statement
- Module 9451 Roll back and close a transaction
- Module 9452 Seek next record in the result set of a query
- Module 9453 Fetch binary data from the result set of a query
- Module 9454 Update binary data in the result set of a query
- Module 9455 Read a single numeric value from a table
- Module 9458 Unload a column description
- Module 9459 Change or insert a value in a comma separated list
- Return Codes of PLC Modules 9440-9459 (Error Stack)

Module 9440 Open a transaction

Module 9440 executes the SELECT statement that is given to the module. For a description of the supported queries, see "**SQL Commands**." If the statement was executed successfully, a transaction is opened and its handle is returned. This handle can be used to read data from a machine table (tool table, for example), or to change data in the table. The cursor is played on the first record of the query result. In order to save changes in the tables, the transaction must be concluded and closed with module 9441 after changing the records.

If the statement contains a string **:'Bnnn'** or **:'Wnnn'** or **:'Dnnn'** (with nnn reading as number from 0 to the maximal number of BYTEs, WORDs or DWORDs respectively); this string is replaced by the integer value found in the corresponding PLC data.

Constraints:

- The module can only be executed within a submit job.
- No more than 10 transactions may be open at the same time.

Possible errors:

- The module was not called in a submit job
- The statement is syntactically not correct
- The table given does not exist, is not accessible or is fully or partially locked
- The columns given do not exist within the table
- No records were selected

Call:

PL	B/W/D/K/S	<>Valid SQL statement>
CM	9440	
PL	D	<>Transaction handle>
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Transaction was successfully opened
	1	For error, see <Error number>

Module 9441 Conclude and close a transaction

Module 9441 concludes a transaction. The module tries to write all buffered changes to the selected machine table. If the action is successful, the transaction is concluded and the transaction handle is invalidated. Otherwise, the transaction remains open. In order to close the transaction despite this, correct the modifications so that no constraints are violated. If the changes are not successful, the buffered changes cannot be saved (conclusion with module 9451).

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle
- At least one modification made violates a uniqueness constraint defined for a column
- At least one modification made violates a foreign key constraint defined for a column

Call:

PS	B/W/D/K	<>Transaction handle>
CM	9441	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Transaction was successfully closed
	1	For error, see <Error number>

Module 9442 Seek a record in the result set

Module 9442 positions the cursor on the record, defined by the record number, in the result set. If the given record number does not identify the desired record unambiguously, the cursor is then placed on the first or last record (depending on the value of the record number given). The first record is addressed by the record number 0.

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9442 can be used

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle
- The record number exceeds the number of selected records
- The statement did not lock the selected records and the record was deleted by another statement

Call:

PS	B/W/D/K	<>Transaction handle>
PS	B/W/D/K	<>Record number>
CM	9442	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Record was found
	1	For error, see <Error number>

Module 9443 Fetch a record from the result set

Module 9443 reads a record (line) from a table and saves it in a string. There must already be a transaction open. Its transaction handle is given to the module. The values are returned as a comma separated list.

Empty fields are output as two successive commas (**...,,...**).

A decimal point is always used for data types **REAL**, **LENGTH**, and **FEED**. Values of the data types **SIGN**, **BOOL**, and **TEXT** are enclosed in single quotes (**'**).

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9443 can be used.

Possible errors:

- The module was not called in a submit job
- An invalid string address is given to the module
- No transaction was opened for the given handle
- No record was picked
- The length of the string exceeds the maximal string length

Call:

PS	B/W/D/K	<>Transaction handle>
PS	B/W/D/K	<>String address in which the TNC saves the record>
CM	9443	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Record was read and copied
	1	For error, see <Error number>

Module 9444 Change a record in the result set

Module 9444 reads a string and saves it in the current record. The current record is the one in which the cursor is located at present. If the cursor is at the end of the query result, a new entry is added. There must already be a transaction open. Its transaction handle is given to the module. The values must be given in the string as a comma separated list and in the appropriate table format.

Empty fields are output as two successive commas (**...,,...**) or via the keyword NULL (**...,NULL,...**).

A decimal point is always used for data types **REAL**, **LENGTH**, and **FEED**. Values of the data types **SIGN**, **BOOL**, and **TEXT** are enclosed in single quotes (**'**).

The modifications are buffered until the transaction is committed.

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9444 can be used.

Possible errors:

- The module was not called in a submit job
- An invalid string address is given to the module
- At least one transferred value is outside the valid range
- At least one transferred value is syntactically incorrect.

Call:

PS	B/W/D/K	<>Transaction handle>
PS	B/W/D/K	<>String address>
CM	9444	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Record was updated and inserted
	1	For error, see <Error number>

Module 9445 Read a single value from a table

Module 9445 reads a single value from a table cell and copies this value to a string. The cell content is selected via a SQL statement. This SQL stated is given to the module beforehand.

If the statement contains a string **:'Bnnn'** or **:'Wnnn'** or **:'Dnnn'** (with nnn reading as number from 0 to the maximal number of BYTEs, WORDs or DWORDs respectively); this string is replaced by the integer value found in the corresponding PLC data.

A decimal point is always used for data types **REAL**, **LENGTH**, and **FEED**. Values of the data types **SIGN**, **BOOL**, and **TEXT** are enclosed in single quotes (**'**).

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- An invalid string address is given to the module
- The statement is syntactically not correct
- The table given does not exist or is not accessible
- The columns given do not exist within the table
- More than one column was named in the statement
- No record or more than one record were selected by the statement

Call:

PS	B/W/D/K	<>Valid SQL statement>
PS	B/W/D/K	<>String address for the read value>
CM	9445	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Value was read
	1	For error, see <Error number>

Module 9447 Delete record from result set

Module 9447 deletes the current record (table line). The current record is the one in which the cursor is located at present. There must already be a transaction open. Its transaction handle is given to the module. The modification is buffered until the transaction is committed.

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9447 can be used.

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle

Call:

PS	B/W/D/K	<>Transaction handle>
CM	9447	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Description was successfully loaded
	1	For error, see <Error number>

Module 9448 Load a column description

Module 9448 loads the description of one or more columns into a cache. The column is specified by its qualified name in the form table.column. If a wildcard '*' is given as the column (i.e., a qualified name in the form table.*), the description of all columns of the given table is loaded. The description is used by various modules for the conversion to or from binary data. Required descriptions that were not found in the cache are automatically loaded during the execution of these modules. However, the execution time of these modules can be improved, if descriptions are cached.

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The referenced table does not exist.
- The referenced column does not exist.

Call:

PS	B/W/D/K	<>String address with the qualified column name>
CM	9448	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Description was successfully loaded
	1	For error, see <Error number>

Module 9449 Extract a value from a comma separated list

Module 9449 extracts a value from a comma separated list of values.

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- The index for the value exceeds the number of values in the string

Call:

PS	B/W/D/K	<>String address for the list of values>
PS	B/W/D/K	<>Index of the value to extract>
PS	B/W/D/K	<>String address for the extracted value>
CM	9449	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Value was successfully extracted
	1	For error, see <Error number>

Module 9450 Execute an SQL statement

Module 9450 executes the SQL statement that is given to the module. For a description of the supported SQL statements, see "**SQL Commands**." This module may not be used to open a transaction (such as via a SELECT instruction).

If the statement contains a string **:'Bnnn'** or **:'Wnnn'** or **:'Dnnn'** (with nnn reading as number from 0 to the maximal number of BYTEs, WORDs or DWORDs respectively), this string is replaced by the integer value found in the corresponding PLC data.

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- The statement is syntactically not correct
- The table given does not exist, is not accessible or is fully or partially locked
- The columns given do not exist within the table

Call:

PS	B/W/D/K/S	<>Valid SQL statement>
CM	9450	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Statement was successfully executed
	1	For error, see <Error number>

Module 9451 Roll back and close a transaction

Module 9451 does not save all buffered modifications of a table during a transaction to the table. The transaction is closed and the transaction handle is invalidated.

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle

Call:

PS	B/W/D/K	<>Transaction handle>
CM	9451	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Transaction was successfully closed
	1	For error, see <Error number>

Module 9452 Seek next record in the result set of a query

Module 9452 positions to the cursor to the next entry in the result set.

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9452 can be used.

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle
- The last record in the result set has been reached
- The statement did not lock the selected records and the record was deleted by another statement

Call:

PS	B/W/D/K	<>Transaction handle>
CM	9452	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Record was found
	1	For error, see <Error number>

Module 9453 Fetch binary data from the result set of a query

Module 9453 reads a record from a table and converts the data in the selected columns to binary values. There must already be a transaction open. Its transaction handle is given to the module. The column values are copied to a number of successive DWORDs. The index of the first DWORD and the number of DWORDS are given to the module.

Data in a column of the type

- SIGN are converted to 0 or –1.
- BOOL are converted to 0 (FALSE) or +1 (TRUE).
- INDEX are each converted to a DWORD.
- REAL are converted to a DWORD by shifting the decimal separator to the right according to the maximum number of decimal places.
  For example, if the value 10.5 is in the table, module 9453 supplies the value 105000.
- LENGTH and FEED are converted to a DWORD.
- Measurements in INCH are converted to metric units.

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9453 can be used.

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle
- No record was fetched
- The given number of DWORDs does not match the number of values
- The given range of DWORDs does not fit into the available memory

Call:

PS	B/W/D/K	<>Transaction handle>
PS	B/W/D/K	<>Index of the first DWORD>
PS	B/W/D/K	<>Number of the DWORDs to store>
CM	9453	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Data were read and converted
	1	For error, see <Error number>

Module 9454 Update binary data in the result set of a query

Module 9454 reads binary data from a number of successive DWORDs. These data are used to update the current record in the result set. If the cursor is at the end of the query result, a new entry is added. There must already be a transaction open. Its transaction handle is given to the module. The values are read from a number of successive DWORDs. The index of the first DWORD and the number of DWORDs are given to the module.

Data in a column of the type

- SIGN are converted to 0 or –1.
- BOOL are converted to 0 (FALSE) or +1 (TRUE).
- INDEX are each converted to a DWORD.
- REAL are converted to a DWORD by shifting the decimal separator to the right according to the maximum number of places.
  For example, if the value 10.5 is in the table, module 9453 supplies the value 105000.
- LENGTH and FEED are converted to a DWORD.
- Measurements in INCH are converted to metric units.

The modifications to the table are buffered until the transaction is committed.

Constraints:

- The module can only be executed within a submit job.
- There must already be a query result before module 9454 can be used.

Possible errors:

- The module was not called in a submit job
- No transaction was opened for the given handle
- The given number of DWORDS does not match the number of values
- The given range of does not fit into the available memory
- At least one transferred value is outside the valid range

Call:

PS	B/W/D/K	<>Transaction handle>
PS	B/W/D/K	<>Index of the first DWORD in which the control reads the values>
PS	B/W/D/K	<>Number of DWORDs with values>
CM	9453	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Data were updated or inserted
	1	For error, see <Error number>

Module 9455 Read a single numeric value from a table

Module 9455 reads a single value from a table and converts it to a binary value. The value is chosen via a given SELECT statement.

If the statement contains a string :'Bnnn' or :'Wnnn' or :'Dnnn' (with nnn reading as number from 0 to the maximal number of BYTEs, WORDs or DWORDs respectively), this string is replaced by the integer value found in the corresponding PLC data.

Data in a column of the type

- SIGN are converted to 0 or –1.
- BOOL are converted to 0 (FALSE) or +1 (TRUE).
- INDEX are each converted to a DWORD.
- REAL are converted to a DWORD by shifting the decimal separator to the right according to the maximum number of places.
  For example, if the value 10.5 is in the table, module 9453 supplies the value 105000.
- LENGTH and FEED are converted to a DWORD.
- Measurements in INCH are converted to metric units.

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- An invalid string address is given to the module
- The given table does not exist or is not accessible
- The column given does not exist in the table
- The column does not hold a numerical value
- More than one column was named in the statement
- No record or more than one record were selected by the statement

Call:

PS	B/W/D/K	<>Valid SQL statement>
CM	9455	
PS	B/W/D	<>Element value>
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Value was read
	1	For error, see <Error number>

Module 9458 Unload a column description

Module 9458 removes the description of one or more columns from the cache. The column is specified by its qualified name in the form table.column. If a wildcard '*' is given as the column (i.e., a qualified name in the form table.*), the description of all columns of the given table is unloaded. In order to save memory, descriptions should be unloaded if they are no longer used. They must be unloaded, if a different table file is used with the same table name.

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The column given to the module was not found in the cache

Call:

PS	B/W/D/K	<>String address with the qualified column name>
CM	9458	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Description was successfully unloaded
	1	For error, see <Error number>

Module 9459 Change or insert a value in a comma separated list

Module 9459 changes or inserts a value in a comma separated list of values. If the index is less than the number of values, the value in the list is replaced. If the index is equal to the number of values, the value is appended. .

Constraints:

- The module can only be executed within a submit job.

Possible errors:

- The module was not called in a submit job
- The index for the value exceeds the number of values in the string

Call:

PS	B/W/D/K	<>String address for the list of values>
PS	B/W/D/K	<>Index of the value to insert>
PS	B/W/D/K	<>String address for the extracted value>
CM	9459	
PL	B/W/D	<>Error number> For the meaning of the error number, see "**Return Codes of PLC Modules 9440-9459 (Error Stack)**."

Error code:

Marker	Value	Meaning
NN_GenApiModuleError	0	Value was successfully updated or inserted
	1	For error, see <Error number>

Return Codes of PLC Modules 9440-9459 (Error Stack)

The error messages returned by modules 9440 to 9459 have the following meanings:

Value (Error Stack)	Meaning
0	Module executed successfully
1	Parameter out of range
2	Parameter not defined
3	Invalid address programmed
4	Address too high or block too long
5	Specified address is not a double word address
11	String could not be converted
12	String too long
15	Module was not called in a submit job
16	No connection with SQL server established
17	Invalid transaction handle was programmed
20	Syntax of the statement is incorrect
25	Table file not found
26	Table file cannot be accessed
27	New file already exists
30	Table header is invalid
31	Configuration message is invalid
32	Table type has not been configured
33	Table contains no columns
34	Unexpected end of table
35	Table has already been opened
36	Table is write-protected
40	Column description is invalid
41	Column type was not configured
42	Column defined several times
43	Column already exists in table
44	Columns do not exist in table or record
45	No column given with the statement
50	Symbolic name already exists
51	Symbolic name does not exist
52	Symbolic name cannot be accessed
55	Index name already exists
56	Index name does not exist
57	Index must not be created
60	Record has already been locked
61	Data record already deleted
62	Invalid length of a record

Value (Error Stack)	Meaning
63	Index for query result too large
70	Invalid default value
71	Invalid value type
72	Invalid number of values
73	Given value is not unique
75	Assigned value must not be null
76	Assigned value is invalid
77	Assigned value is too long
78	Assigned value is out of range
79	Assigned value already exists
80	Primary key must not be dropped or renamed
81	Primary key must not be updated
82	Primary key must not be set null
85	Action violates the referential integrity
86	Referential action conflicts with statement
90	Function not yet implemented
91	Internal (software) error

Data Transfer NC –> PLC, PLC –> NC

Information is exchanged between PLC and NC by markers, bytes, words, and double words. The function of the individual markers, bytes, words, and double words is freely definable in the machine configuration.

The transfer of certain data to the PLC is controlled by strobes:

- M codes
- S codes
- T codes
- G codes
- Q codes

Example:

If an M function is output, the NC sets the strobe signal NP_MG_Strobe_M_Funktion (defined by the PLC programmer). After evaluating the M function, the PLC must set an acknowledgment marker (e.g., PN_MG_Quit_M_Function). The PLC must then reset the acknowledgment marker; otherwise, no further strobes can be sent by the NC.

The M functions are configured through machine parameters, see “**Section 6, M Functions (M Strobe)**.”

The following topics are described:

- **Data Transfer of NC Program –> PLC (“FN19: PLC =” or “FN29: PLC =)”**
- **Data Transfer of NC Program –> PLC (FN17: SYSWRITE)**
- **Data Transfer NC –> NC Program (FN18: SYSREAD)**
- **Data Transfer Machine Parameters –> PLC**
- **Interrogate PLC Operands in the NC Program (FN20: WAIT FOR)**

Data Transfer of NC Program –> PLC (“FN19: PLC =” or “FN29: PLC =)”

The Q-parameter functions **FN19: PLC =** and **FN29: PLC =** transfers numerical values from an NC program to the PLC. The function **FN19:** transfers two values, and the function **FN29:** eight values. The control stores the transferred values as integer values of the form
1/10 000 in double words.

The symbolic name and the address of the double words can be chosen as desired. Each of the two functions is mapped on an M function. You define the associated symbolic markers and double words in the machine configuration.

Settings in the configuration editor:	
System PLC CfgPlcMStrobe (see page 6 – 45) FN19 FN29 CfgPlcStrobeAlias FN19 **type** **mCode** **mOffset** FN29 **type** **mCode** **mOffset**	

The control treats the functions **FN19:** and **FN29:** as M functions. (See “**Section 6, Configuration of M Functions**.”) With **MP_CfgPlcStrobeAlias,** you assign an M function from **MP_CfgPlcMStrobe** to the control-specific FN function.

MP_type

Specifies the type of strobe

Format: Pull-down selection menu

Selection:

[FN19]
Data transfer NC Æ PLC, two values simultaneously
[FN29]
Data transfer NC Æ PLC, eight values simultaneously
[CYCLE13]
Define spindle position for M19
[TCHPROBE]
Call measuring cycles

MP_mCode

Number of the M function The function entered in MP_type is mapped on the given M function.

Format: String

Input: Numerical value from 0 to 9999

MP_mOffset

Offset for the functions **FN19:** and **FN29:**

Format: Pull-down selection menu

Selection:

[TRUE]

The first numerical value transferred is used as an offset, and entered in the attribute MP_min of the associated M function. The remaining numerical values are written to the double word entered in MP_data.

Example: FN19: is mapped on M190

FN19: PLC = +4 / +44

The control uses the numerical value +44 to describe the double word entered in M194.

[FALSE]

No offset is used.

Data Transfer of NC Program –> PLC (FN17: SYSWRITE)

The **FN17: SYSWRITE** function is particularly useful for OEM cycles if you wish to overwrite certain NC data with data from the part program. The system datum that you wish to overwrite is identified by a group number, a system data number, and an index.

FN17: SYSWRITE IDxxxx NRxxxx IDXxxxx = Qxxx or numerical value; comment.

In the NC program you cannot define function FN17 (soft keys: Q-parameter programming, special functions, second soft-key row), until you have entered the code number 555 343. After a control reset, the code number must be entered again if you wish to program **FN17.** The control provides the following functions:

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Branch addresses of the system				
	13	1	–	Label to which the control jumps if M2/M30 is programmed, instead of ending the current program (value =0: M2/M30 has the usual effect).
		2	–	Label to which the control jumps in the event of FN14:ERROR with the reaction NC-CANCEL, instead of aborting the program with an error. The error number programmed in FN14 can be read in ID992 NR14. Value = 0: FN14 has the usual effect.
		3	–	Label to which the control jumps if an internal server error (SQL, PLC, CFG) occurs, instead of aborting the program with an error message. Value = 0: The server error has the usual effect.
Cycle parameters				
	30	1	–	Set-up clearance
		2	–	Hole depth / milling depth
		3	–	Plunging depth
		4	–	Feed rate for plunging
		5	–	First side length of pocket
		6	–	Second side length of pocket
		7	–	First side length of slot
		8	–	Second side length of slot
		9	–	Radius of circular pocket
		10	–	Feed rate for milling
		11	–	Rotational direction of the milling path
		12	–	Dwell time
		13	–	Pitch
		14	–	Finishing allowance
		15	–	Roughing angle
		21	–	Probing angle
		22	–	Probing path
		23	–	Probing feed rate

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Data from the tool table				
	50	1	Tool number	Tool length L
		2	Tool number	Tool radius R
		3	Tool number	Tool radius R2
		4	Tool number	Oversize for tool length DL
		5	Tool number	Oversize for tool radius DR
		6	Tool number	Oversize for tool radius DR2
		7	Tool number	Tool locked TL 0 = not locked, 1 = locked
		8	Tool number	Number of the replacement tool RT
		9	Tool number	Maximum tool age TIME1
		10	Tool number	Maximum tool age TIME2
		11	Tool number	Current tool age CUR. TIME
		12	Tool number	PLC status
		13	Tool number	Maximum tooth length LCUTS
		14	Tool number	Maximum plunge angle ANGLE
		15	Tool number	TT: Number of tool teeth CUT
		16	Tool number	TT: Wear tolerance for length LTOL
		17	Tool number	TT: Wear tolerance for radius RTOL
		18	Tool number	TT: Direction of rotation DIRECT 0 = positive, –1 = negative
		19	Tool number	TT: Offset in plane R-OFFS R = 99 999.9999
		20	Tool number	TT: Offset in length L-OFFS
		21	Tool number	TT: Break tolerance for length LBREAK
		22	Tool number	TT: Break tolerance for radius RBREAK
		23	Tool number	PLC value
		24	Tool number	Probe center offset in reference axis CAL-OF1
		25	Tool number	Probe center offset in minor axis CAL-OF2
		26	Tool number	Spindle angle during calibration CAL-ANG
		27	Tool number	Tool type for pocket table
		28	Tool number	Maximum speed NMAX

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Freely definable for tool management				
	56	1	–	Freely avaliable memory range, reserved for tool management cycles
		2	–	
		3	–	
		4	–	
		5	–	
		6	–	
		7	–	
		8	–	
		9	–	
Freely available for OEM cycles				
	72	1	<Value 0 to 9>	Freely available memory range, reserved for OEM cycles
		2	<Value 0 to 9>	
		3	<Value 0 to 9>	
		4	<Value 0 to 9>	
		5	<Value 0 to 9>	
		6	<Value 0 to 9>	
		7	<Value 0 to 9>	
		8	<Value 0 to 9>	
		9	<Value 0 to 9>	

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Tool compensation				
	200	1	1 = without oversize 2 = with oversize	Active tool radius R
		2	1 = without oversize 2 = with oversize	Active tool length L
		3	1 = without oversize 2 = with oversize	Active tool radius R2
Coordinate transformation				
	210	1	–	Basic rotation (manual)
		2	–	Programmed rotation
		3	–	Active mirror axis Bits 0 to 2 and 6 to 8: Axes X, Y, Z and U, V, W
		4	1	Active scaling factor in X
			2	Active scaling factor in Y
			3	Active scaling factor in Z
			7	Active scaling factor in U
			8	Active scaling factor in V
			9	Active scaling factor in W
		5	1	3-D ROT A
			2	3-D ROT B
			3	3-D ROT C
		6	–	Tilt working plane in Program Run mode (0 = inactive, –1 = active)
	211	–	–	Current coordinate system 1 = input system (default) 2 = REF system 3 = tool changer
Change axis				
	212	–	–	0: Normal axis assignment 1: X Æ Y, Y Æ Z, Z Æ X 2: X Æ Z, Y Æ X, Z Æ Y 3: Tool axis from TOOL CALL

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Range of traverse				
	230	2	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Negative software limit switches
		3	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Positive software limit switches
		4	Number of axes whose software limit switches are to be overwritten	Number of the first of several consecutive Q parameters 1st Q: Neg. limit switch in 1st axis 2nd Q: Pos. limit switch in 1st axis 3rd Q: Neg. limit switch in 2nd axis etc.
		5	–	Limit-switch monitoring (1 = off, 0 = on)
Modification of geometric behavior				
	310	97	–	0 = insert transitional arcs at outside corners –1 = no transitional arcs at outside corners (= M97 until the end of the main program)
		114	–	0 = M114 off 1 = M114 on
		128	–	0 = M128 off 1 = M128 on
TS touch-trigger probe				
	350	50	1	Type of touch probe
			2	Line in the touch-probe table
		51	–	Effective length
		52	1	Radius of ring gauge
			2	Rounding radius
		53	1	Center offset (reference axis)
			2	Center offset (minor axis)
		54	–	Direction of the center offset with respect to spindle 0°
		55	1	Rapid traverse
			2	Measuring feed rate
		56	1	Maximum measuring range
			2	Safety clearance
		57	1	Oriented spindle stop possible? 0 = no, 1 = yes
			2	Spindle-orientation angle in degrees

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Coordinate transformation				
	420	–	–	0 = currently active cycle transformation is transferred to the main program
Write values into active datum table				
	500	Line	Column	
Look-ahead parameters				
	610	1	–	Minimum feed rate (**MP_minPathFeed**) in mm/min
		2	–	Minimum feed rate at corners (**MP_minCornerFeed**) in mm/min
		3	–	Feed-rate limit for high speeds (**MP_maxG1Feed**) in mm/min
		4	–	Max. jerk at low speeds (**MP_maxPathJerk**) in m/s^3
		5	–	Max. jerk at high speeds (**MP_maxPathJerkHi**) in m/s^3
		6	–	Tolerance at low speeds (**MP_pathTolerance**) in mm
		7	–	Tolerance at high speeds (**MP_pathToleranceHi**) in mm
		8	–	Max. derivative of jerk (**MP_maxPathYank**) in m/s^4
		9	–	Tolerance factor for curves (**MP_curveToleranceFactor**)
		10	–	Factor for max. permissible jerk at curvature changes (**MP_curveChangeJerkFactor**)
		11	–	Limit frequency for position filter (**MP_posFilterLimitFrequency**) in Hz – effective for all axes
		12	–	Angle tolerance at low speeds (**MP_angleTolerance**)
		13	–	Angle tolerance at high speeds (**MP_angleToleranceHi**)
		20	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. feed rate (**MP_maxFeed**) in mm/min

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
		21	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. acceleration (**MP_maxAcceleration**) in m/s^2
		22	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. deceleration (**MP_maxDeceleration**) in m/s^2
		23	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. jerk (**MP_maxJerk**) in m/s^3
		24	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Accleration feedforward control (**MP_compAcc**)
		99	–	0 = reset all values Resets all look-ahead parameters to the values in the machine configuration.
Enable axis keys				
	910	1	–	1= enable axis keys during program run 0 = disable axis keys during program run
Write current tool data				
	950	1	–	Tool length L
		2	–	Tool radius R
		3	–	Tool radius R2
		4	–	Oversize for tool length DL
		5	–	Oversize for tool radius DR
		6	–	Oversize for tool radius DR2
		7	–	Tool locked TL 0 = not locked, 1 = locked
		8	–	Number of the replacement tool RT
		9	–	Maximum tool age TIME1
		10	–	Maximum tool age TIME2
		11	–	Current tool age CUR. TIME
		12	–	PLC status
		13	–	Maximum tooth length LCUTS
		14	–	Maximum plunge angle ANGLE
		15	–	TT: Number of tool teeth CUT
		16	–	TT: Wear tolerance for length LTOL
		17	–	TT: Wear tolerance for radius RTOL
		18	–	TT: Direction of rotation DIRECT 0 = positive, –1 = negative

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
		19	–	TT: Offset in plane R-OFFS R = 99 999.9999
		20	–	TT: Offset in length L-OFFS
		21	–	TT: Break tolerance for length LBREAK
		22	–	TT: Break tolerance for radius RBREAK
		23	–	PLC value
		24	–	Tool type (TYPE) 0 = milling cutter, 21 = touch probe
Lift off at NC stop				
	980	1	–	Retract tool at NC stop 0 = Do not retract tool
		2	–	Suppress tool retraction temporarily 1: Suppress lift-off 0 = cancel suppression
Touch probe cycles				
	990	1	–	Approach behavior: 0 = Standard behavior 1 = Effective radius, safety clearance zero
		2	–	0 = probe monitoring off 1 = probe monitoring on
Transfer axis positions				
	991	100	–	Transfer interpolator position
PLC data				
	2000	10	Marker number	PLC markers

Data Transfer NC –> NC Program (FN18: SYSREAD)

System Data

The **FN18: SYSREAD** function is particularly useful for OEM cycles if you wish to have access from the part program to certain NC data, such as active tool compensation. The system datum that you wish to read is identified by a group number, a system data number, and an index.

FN18: SYSREAD Qxxx = IDxxxx NRxxxx IDXxxxx (xxxx: Q parameter or numerical value) comment

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Program information				
	10	3	–	Number of the active fixed cycle
		103	Q parameter number	Relevant within NC cycles; For inquiring whether the Q parameter indicated under IDX was specified in the associated CYCLE DEF.
Branch addresses of the system				
	13	1	–	Label the branch leads to if M2/M30 is programmed, instead of ending the current program (value =0: M2/M30 has the usual effect).
		2	–	Label to which the control jumps in the event of FN14:ERROR with the reaction NC-CANCEL, instead of aborting the program with an error. The error number programmed in FN14 can be read in ID992 NR14. Value = 0: FN14 has the usual effect.
		3	–	Label to which the control jumps if an internal server error (SQL, PLC, CFG) occurs, instead of aborting the program with an error message. Value = 0: Server error has the usual effect.
Machine status				
	20	1	–	Active tool number
		2	–	Prepared tool number
		3	–	Active tool axis 0 = X 6 = U 1 = Y 7 = V 2 = Z 8 = W
		4	–	Programmed spindle speed
		5	–	Active spindle status –1 = Spindle status undefined 0 = M3 active 1 = M4 active 2 = M5 active after M3 3 = M5 active after M4
		8	–	Active coolant status 0 = off, 1 = on
		9	–	Active feed rate

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
		10	–	Index of prepared tool
		11	–	Index of active tool
Channel data				
	25	1	–	Channel number
Cycle parameters				
	30	1	–	Safety clearance
		2	–	Hole depth / milling depth
		3	–	Plunging depth
		4	–	Feed rate for plunging
		5	–	First side length of pocket
		6	–	Second side length of pocket
		7	–	First side length of slot
		8	–	Second side length of slot
		9	–	Radius of circular pocket
		10	–	Feed rate for milling
		11	–	Rotational direction of the milling path
		12	–	Dwell time
		13	–	Pitch
		14	–	Finishing allowance
		15	–	Roughing angle
		21	–	Probing angle
		22	–	Probing path
		23	–	Probing feed rate
Modal status				
	35	1	–	Dimensions: 0 = absolute (G90) 1 = incremental (G91)
Data for SQL tables				
	40	1	–	Result code for last SQL command

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Data from the tool table				
	50	1	Tool number	Tool length L
		2	Tool number	Tool radius R
		3	Tool number	Tool radius R2
		4	Tool number	Oversize for tool length DL
		5	Tool number	Oversize for tool radius DR
		6	Tool number	Oversize for tool radius DR2
		7	Tool number	Tool locked TL 0 = not locked, 1 = locked
		8	Tool number	Number of the replacement tool RT
		9	Tool number	Maximum tool age TIME1
		10	Tool number	Maximum tool age TIME2
		11	Tool number	Current tool age CUR. TIME
		12	Tool number	PLC status
		13	Tool number	Maximum tooth length LCUTS
		14	Tool number	Maximum plunge angle ANGLE
		15	Tool number	TT: Number of tool teeth CUT
		16	Tool number	TT: Wear tolerance for length LTOL
		17	Tool number	TT: Wear tolerance for radius RTOL
		18	Tool number	TT: Direction of rotation DIRECT 0 = positive, –1 = negative
		19	Tool number	TT: Offset in plane R-OFFS R = 99 999.9999
		20	Tool number	TT: Offset in length L-OFFS
		21	Tool number	TT: Breakage tolerance in length LBREAK
		22	Tool number	TT: Breakage tolerance in radius RBREAK
		23	Tool number	PLC value
		24	Tool number	Probe center offset in reference axis CAL-OF1
		25	Tool number	Probe center offset in minor axis CAL-OF2
		26	Tool number	Spindle angle during calibration CAL-ANG
		27	Tool number	Tool type for pocket table
		28	Tool number	Maximum speed NMAX

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Data from the pocket table				
	51	1	Pocket number	Tool number
		2	Pocket number	0 = no special tool 1 = special tool
		3	Pocket number	0 = no fixed pocket 1 = fixed pocket
		4	Pocket number	0 = pocket not locked 1 = pocket locked
		5	Pocket number	PLC status
Tool pocket				
	52	1	Tool number	Pocket number P
		2	Tool number	Tool magazine number
Freely definable for tool management				
	56	1	–	Freely avaliable memory range, reserved for tool management cycles
		2	–	
		3	–	
		4	–	
		5	–	
		6	–	
		7	–	
		8	–	
		9	–	
Values programmed in TOOL CALL				
	60	1	–	Tool number T
		2	–	Active tool axis 0 = X 6 = U 1 = Y 7 = V 2 = Z 8 = W
		3	–	Spindle speed S
		4	–	Oversize for tool length DL
		5	–	Oversize for tool radius DR
		6	–	Automatic TOOL CALL 0 = yes, 1 = no
		7	–	Oversize for tool radius DR2
		8	–	Tool index
		9	–	Active feed rate

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Values programmed in TOOL DEF				
	61	1	–	Tool number T
		2	–	Length
		3	–	Radius
		4	–	Index
		5	–	Tool data programmed in TOOL DEF 1 = yes, 0 = no
Freely available memory range				
	72	1	<Value 0 to 9>	Freely available memory range for OEM cycles
		2	<Value 0 to 9>	
		3	<Value 0 to 9>	
		4	<Value 0 to 9>	
		5	<Value 0 to 9>	
		6	<Value 0 to 9>	
		7	<Value 0 to 9>	
		8	<Value 0 to 9>	
		9	<Value 0 to 9>	

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Tool compensation				
	200	1	1 = without oversize 2 = with oversize 3 = with oversize and oversize from TOOL CALL	Active radius
		2	1 = without oversize 2 = with oversize 3 = with oversize and oversize from TOOL CALL	Active length
		3	1 = without oversize 2 = with oversize 3 = with oversize and oversize from TOOL CALL	Rounding radius R2
Coordinate transformation				
	220	2	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Current datum shift in mm

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Range of traverse				
	230	2	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Negative software limit switches
		3	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Positive software limit switches
		5	–	Switch software limit switches on/off: 0 = on, 1 = off
Nominal position in the REF system				
	240	1	1 to 9 (X, Y, Z, A, B, C, U, V, W)	
Current position in the active coordinate system				
	270	1	1 to 9 (X, Y, Z, A, B, C, U, V, W)	
TS touch-trigger probe				
	350	50	1	Type of touch probe
			2	Line in the touch-probe table
		51	–	Effective length
		52	1	Radius of ring gauge
			2	Rounding radius
		53	1	Center offset (reference axis)
			2	Center offset (minor axis)
		54	–	Direction of the center offset with respect to spindle 0°
		55	1	Rapid traverse
			2	Measuring feed rate
		56	1	Maximum measuring range
			2	Safety clearance
		57	1	Oriented spindle stop possible? 0 = no, 1 = yes
			2	Spindle-orientation angle in degrees

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
Datum point from touch probe cycle (probing results)				
	360	1	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Last datum of a manual touch probe cycle, or last touch point from Cycle 0 without compensation of stylus length, but with compensation of stylus radius (workpiece coordinate system)
		2	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Last datum of a manual touch probe cycle or last touch point from Cycle 0 without probe length or probe radius compensation (machine coordinate system)
		3	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Measurement result of Touch Probe Cycles 0 and 1 without compensation of stylus radius or length
		4	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Last datum of a manual touch probe cycle or last touch point from Cycle 0 without probe length or probe radius compensation (workpiece coordinate system)
		10	–	Oriented spindle stop
Read values from active datum table				
	500	Line	Column	Read values
Look-ahead parameters				
	610	1	–	Minimum feed rate (**MP_minPathFeed**) in mm/min
		2	–	Minimum feed rate at corners (**MP_minCornerFeed**) in mm/min
		3	–	Feed-rate limit for high speeds (**MP_maxG1Feed**) in mm/min
		4	–	Max. jerk at low speeds (**MP_maxPathJerk**) in m/s^3
		5	–	Max. jerk at high speeds (**MP_maxPathJerkHi**) in m/s^3
		6	–	Tolerance at low speeds (**MP_pathTolerance**) in mm
		7	–	Tolerance at high speeds (**MP_pathToleranceHi**) in mm
		8	–	Max. derivative of jerk (**MP_maxPathYank**) in m/s^4
		9	–	Tolerance factor for curves (**MP_curveToleranceFactor**)

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
		10	–	Factor for max. permissible jerk at curvature changes (**MP_curveChangeJerkFactor**)
		11	–	Limit frequency for position filter (**MP_posFilterLimitFrequency**) in Hz – minimum frequency for all axes
		12	–	Angle tolerance at low speeds (**MP_angleTolerance**)
		13	–	Angle tolerance at high speeds (**MP_angleToleranceHi**)
		20	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. feed rate (**MP_maxFeed**) in mm/min
		21	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. acceleration (**MP_maxAcceleration**) in m/s^2
		22	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. deceleration (**MP_maxDeceleration**) in m/s^2
		23	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Max. jerk (**MP_maxJerk**) in m/s^3
		24	1 to 9 (X, Y, Z, A, B, C, U, V, W)	Accleration feedforward control (**MP_compAcc**)
Read data of current tool				
	950	1	–	Tool length L
		2	–	Tool radius R
		3	–	Tool radius R2
		4	–	Oversize for tool length DL
		5	–	Oversize for tool radius DR
		6	–	Oversize for tool radius DR2
		7	–	Tool locked TL 0 = not locked, 1 = locked
		8	–	Number of the replacement tool RT
		9	–	Maximum tool age TIME1
		10	–	Maximum tool age TIME2
		11	–	Current tool age CUR. TIME
		12	–	PLC status
		13	–	Maximum tooth length LCUTS
		14	–	Maximum plunge angle ANGLE

Group name	Group number ID....	System data number NR....	System data index IDX....	System data item
		15	–	TT: Number of tool teeth CUT
		16	–	TT: Wear tolerance for length LTOL
		17	–	TT: Wear tolerance for radius RTOL
		18	–	TT: Direction of rotation DIRECT 0 = positive, –1 = negative
		19	–	TT: Offset in plane R-OFFS R = 99 999.9999
		20	–	TT: Offset in length L-OFFS
		21	–	TT: Break tolerance for length LBREAK
		22	–	TT: Break tolerance for radius RBREAK
		23	–	PLC value
		24	–	Tool type (TYPE) 0 = milling cutter, 21 = touch probe
Touch probe cycles				
	990	1	–	Approach behavior: 0 = Standard behavior 1 = Effective radius, safety clearance zero
		2	–	0 = Probe monitoring off 1 = Probe monitoring on
Status of execution				
	992	10	–	Block scan active 1 = yes, 0 = no
		11	–	Search phase
		14	–	Number of the last FN14 error
		16	–	Real execution active 1 = execution, 2 = simulation
PLC data				
	2000	10	Marker number	PLC markers
		20	Input number	PLC input
		30	Output number	PLC output
		40	Counter number	PLC counter
		50	Timer number	PLC timers
		60	Byte number	PLC byte
		70	Word number	PLC word
		80	Double-word number	PLC double word

System Strings

For some system data groups, you can also interrogate system strings. The system strings can not directly been written to, but like with normal system variables, they can be linked to table columns (provided that these table columns contain texts).
The group number (ID) of the system string results from the group number of the system datum + 10000. It is not necessary to specify a system data index IDX for reading them.

QSxx = SYSSTR(IDxxxx NRxxxx) (xxxx: Group number and system string number)

Group name	Group number ID....	System string number NR....	System string
Program information			
	10010	3	Path of the cycle selected with SEL CYCLE or CYCLE DEF 12 PGM CALL, or path of the currently active cycle
		10	Path of the NC program selected with SEL PGM „...“
Channel data			
	10025	1	Channel name
Data for SQL tables			
	10040	1	Symbolic name of the preset table
		2	Symbolic name of the datum table
		10	Symbolic name of the tool table
		11	Symbolic name of the pocket table
Current tool data			
	10950	1	Current tool name

Data Transfer Machine Parameters –> PLC

Settings in the configuration editor:	
System PLC CfgOemBool **[Key of freely selectable parameter]** value **[0]** **ignorePlc** CfgOemInt **[Key of freely selectable parameter]** CfgOemPosition **[Key of freely selectable parameter]**	

Freely definable machine parameters are available for data transmission to the PLC. The control saves the contents of the machine parameters in PLC words.

In the machine parameters you can enter, for example, values for PLC positioning movements and datum shifts, feed rates for PLC positioning movements or codes for the enabling of certain PLC functions. You must evaluate the transferred numerical values in your PLC program.

The freely definable user parameters are divided into three areas:

- **MP_CfgOemBool:**
  User parameters with logical values (True, False)
- **MP_CfgOemInt:**
  User parameters with integer values (whole numbers)
- **MP_CfgOemPosition:**
  User parameters with fixed decimal values (position values)

Each user parameter is in a subfolder (key). The name of the key specifies at the same time the name of the user parameter. The specify the value of the parameter in the **value** subfolder.

If you do not want to copy a user parameter to the PLC run-time system, the optional machine parameter **MP_ignorePlc** must be inserted and set to TRUE.

Interrogate PLC Operands in the NC Program (FN20: WAIT FOR)

With **FN20: WAIT FOR** you can interrupt the NC program until the condition programmed in the FN20 block is fulfilled. These conditions can be comparisons of a PLC operand with a constant. Permitted PLC operands: M, B, W, D, T, C, I, O

Operator	Function
==	Equal
!= or <>	Not equal
<	Less than
>	Greater than
<=	Less than or equal
>=	Greater than or equal

If you enter no condition, the interruption will continue until the operand = 0.

Examples:

FN20: WAIT FOR I10==1
Continue the NC program if PLC input I10 is set.

FN20: WAIT FOR I10
Continue the NC program if PLC input I10 equals zero.

FN20: WAIT FOR B3000>255
Continue the NC program if the content of B3000 is greater than 255.

Program Creation

The following topics are described:

- **ASCII Editor**
- **Program Structure**

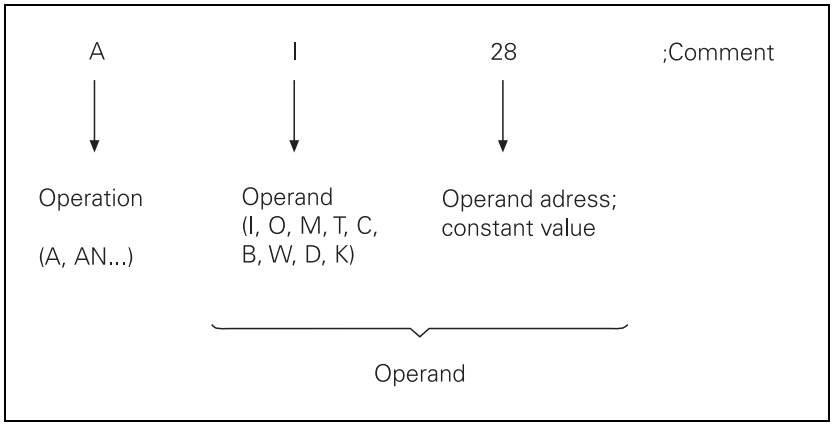
ASCII Editor

The integrated editor enables you to create and edit the PLC program and all other necessary files right at the control through the ASCII keyboard. You will find a complete description of the editor, including the associated soft keys, in the User's Manual for the control.

Program Format

Command

A command is the smallest unit of a PLC program. It consists of the operation part and the operand part.



The operation describes the function to be executed. It says how the operand is to be processed by the TNC. The operand shows what is to be operated with. It consists of the operand abbreviation and a parameter (address). With the PLC commands you can combine (gate), delete and load register and memory contents, both with bit and word processing. For word processing, you can address memory contents with a length of 8 bits (byte), 16 bits (word) or 32 bits (double word).

Program Structure

To make it easier to maintain and expand your PLC program, you should give it a modular structure. Modular means that you write a separate program module for each function. You can then call the individual modules from the main program. You should interrogate improper functioning of the machine in the PLC program and indicate such malfunctions on the screen with plain-language error messages.

Module 9019 Size of the processing stack

To debug functions you can use Module 9019 to interrogate the contents of the processing stack. The function answers with the number of the bytes that lie on the processing stack of the PLC at the moment. If the processing stack is empty, the PLC run-time system returns the value zero. A byte, word or double word occupies four bytes on the stack; a marker, input, output, timer or counter occupies two bytes.

Call:

CM	9019
PL	B/W/D <>Number of bytes on processing stack>

PLC Commands

The following topics are described:

- Overview
- LOAD (L)
- LOAD NOT (LN)
- LOAD TWO'S COMPLEMENT (L–)
- LOAD BYTE (LB)
- LOAD WORD (LW)
- LOAD DOUBLE WORD (LD)
- ASSIGN (=)
- ASSIGN BYTE (B=)
- ASSIGN WORD (W=)
- ASSIGN DOUBLE WORD (D=)
- ASSIGN NOT (=N)
- ASSIGN TWO'S COMPLEMENT (=–)
- SET (S)
- RESET (R)
- SET NOT (SN)
- RESET NOT (RN)
- AND (A)
- AND NOT (AN)
- OR (O)
- OR NOT (ON)
- EXCLUSIVE OR (XO)
- EXCLUSIVE OR NOT (XON)
- ADDITION (+)
- SUBTRACTION (–)
- MULTIPLICATION (X)
- DIVISION (/)
- REMAINDER (MOD)
- INCREMENT (INC)
- DECREMENT (DEC)
- EQUAL TO (==)
- LESS THAN (<)
- GREATER THAN (>)
- LESS THAN OR EQUAL TO (<=)
- GREATER THAN OR EQUAL TO (>=)
- NOT EQUAL (<>)
- AND [ ] (A[ ])
- AND NOT [ ] (AN[ ])
- OR [ ] (O[ ])
- OR NOT [ ] (ON[ ])
- EXCLUSIVE OR [ ] (XO[ ])

- EXCLUSIVE OR NOT [ ] (XON[ ])
- ADDITION [ ] (+[ ])
- SUBTRACTION [ ] (–[ ])
- MULTIPLICATION [ ] (x[ ])
- DIVISION [ ] (/[ ])
- REMAINDER [ ] (MOD[ ])
- EQUAL TO [ ] (==[ ])
- LESS THAN [ ] (<[ ])
- GREATER THAN [ ] (>[ ])
- LESS THAN OR EQUAL TO [ ] (<=[ ])
- GREATER THAN OR EQUAL TO [ ] (>=[ ])
- NOT EQUAL [ ] (<>[ ])
- SHIFT LEFT (<<)
- SHIFT RIGHT (>>)
- BIT SET (BS)
- BIT CLEAR (BC)
- BIT TEST (BT)
- Push Data onto the Data Stack (PS)
- Pull Data from the Data Stack (PL)
- Push LOGIC ACCUMULATOR onto the Data Stack (PSL)
- Push WORD ACCUMULATOR onto the Data Stack (PSW)
- Pull LOGIC ACCUMULATOR from the Data Stack (PLL)
- Pull WORD ACCUMULATOR from the Data Stack (PLW)
- UNCONDITIONAL JUMP (JP)
- JUMP IF LOGIC ACCUMULATOR = 1 (JPT)
- JUMP IF LOGIC ACCUMULATOR = 0 (JPF)
- CALL MODULE (CM)
- CALL MODULE IF LOGIC ACCUMULATOR = 1 (CMT)
- CALL MODULE IF LOGIC ACCUMULATOR = 0 (CMF)
- END OF MODULE, END OF PROGRAM (EM)
- END OF MODULE IF LOGIC ACCUMULATOR = 1 (EMT)
- END OF MODULE IF LOGIC ACCUMULATOR = 0 (EMF)
- LABEL (LBL)

Overview

The following table provides an overview of all commands explained in this chapter:

Group of functions	Syntax	Function
Loading and saving instructions		
	L	Load
	LN	Load NOT
	L–	Load two's complement
	LB	Load BYTE
	LW	Load WORD
	LD	Load double word
	=	Assign
	B=	Assign BYTE
	W=	Assign WORD
	D=	Assign DOUBLE WORD
	=N	Assign NOT
	=–	Assign two's complement
Setting commands		
	S	Set
	R	Reset
	SN	Set NOT
	RN	Reset NOT
Logical operations		
	A	and
	AN	And NOT
	O	or
	ON	Or NOT
	XO	Exclusive OR
	XON	Exclusive OR NOT
Arithmetical instructions		
	+	Addition
	–	Subtraction
	x	Multiplication
	/	Division
	MOD	Remainder

Group of functions	Syntax	Function
Increment		
	INC	Increment operand
	INCW	Increment word accumulator
	INCX	Increment index register
Decrement		
	DEC	Decrement operand
	DECW	Decrement word accumulator
	DECX	Decrement index register
Comparisons		
	==	Equal
	<	Less than
	>	Greater than
	<=	Less than or equal
	>=	Greater than or equal
	<>	Not equal
Parenthetical expression in logical operations		
	A[ ]	And [ ]
	AN[ ]	And NOT [ ]
	O[ ]	Or [ ]
	ON[ ]	Or NOT [ ]
	XO[ ]	Exclusive OR [ ]
	XON[ ]	Exclusive OR NOT [ ]
Parenthetical expressions with arithmetical instructions		
	+[ ]	Addition [ ]
	–[ ]	Subtraction [ ]
	x[ ]	Multiplication [ ]
	/[ ]	Division [ ]
	MOD[ ]	Remainder [ ]
Parenthetical expressions in comparisons		
	==[ ]	Equal [ ]
	<[ ]	Less than [ ]
	>[ ]	Greater than [ ]
	<=[ ]	Less than or equal [ ]
	>=[ ]	Greater than or equal [ ]
	<>[ ]	Not equal [ ]
Shifting instructions		
	<<	Shift left
	>>	Shift right

Group of functions	Syntax	Function
Bit commands		
	BS	Set bit
	BC	Clear bit
	BT	Test bit
Stack operations		
	PS	Push data onto the data stack
	PL	Pull data from the data stack
	PSL	Push logic accumulator onto the data stack
	PSW	Push word accumulator onto the data stack
	PLL	Pull logic accumulator from the data stack
	PLW	Pull word accumulator from the data stack
Jump commands		
	JP	Unconditional jump
	JPT	Jump if logic accumulator = 1
	JPF	Jump if logic accumulator = 0
	CM	Call module
	CMT	Call module if logic accumulator = 1
	CMF	Call module if logic accumulator = 0
	EM	End of module, program end
	EMT	End of module if logic accumulator = 1
	EMF	End of module if logic accumulator = 0
	LBL	Label

LOAD (L)

Logic Processing with the LOAD Command

Syntax: L (LOAD)

Operands: M, I, O, T, C

Action:

Load the value of the addressed operand into the logic accumulator. Always use the L command at the beginning of a logic chain in order to be able to gate the operand in the following program sequence.

Example:

Gate the inputs I4 and I5 with AND, and assign the result to output O2.
Initial state:

Input	I4	= 1
Input	I5	= 0
Output	O2	= ?

Function	STL	Logic accu.	Operand content
Load the operand content into the logic accu.	L I4	Logic accumulator = 1	
Gate the content of the logic accumulator and input I5 with AND.	A I5		0
Assign the gating result to output O2.	= O2		0

Word processing with the LOAD command

Syntax: L (LOAD)

Operands: B, W, D, K

Action:

Load the value of the addressed operand, or of a constant, into the word accumulator. If necessary, the accumulator is supplemented with the correct algebraic sign. In contrast to logical operations, you must always begin a sequence of word gating operations with an L command. You cannot replace the L command with a logical gating instruction.

Example:

Gate a constant and byte B5 with AND, and assign the result to byte B8.
Initial state:

Constant	54	= 36 (hex)
Byte	B5	= 2A (hex)
Output	B8	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K+54	36	
Gate the contents of word accumulator and byte B5 with AND.	A B5		2A
Assign the gating result to byte B8.	= B8		22

LOAD NOT (LN)

Logic processing with the LOAD NOT command

Syntax: LN (LOAD NOT)

Operands: M, I, O, T, C

Action:

Load the complement of the addressed operand into the logic accumulator. Always use the L command at the beginning of a logic chain in order to be able to gate the operand in the following program sequence.

Example:

Gate the inverted logical state of inputs I4 and I5 with AND, and assign the result to output O2.
Initial state:

Input	I4	= 0
Input	I5	= 1
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the inverted operand content into the logic accumulator.	LN I4	0	
Gate the content of the logic accumulator and input I5 with AND.	A I5		1
Assign the gating result to output O2.	= O2		1

Word Processing with the LOAD NOT Command

Syntax: LN (LOAD NOT)

Operands: B, W, D, K

Action:

Load the complement of the addressed operand, or of a constant, into the word accumulator. If necessary, the accumulator is supplemented with the correct algebraic sign. In contrast to logical operations, you must always begin a sequence of word gating operations with an L command. You cannot replace the L command with a logical gating instruction.

Example:

Gate the complement of byte B6 and byte B5 with AND, and assign the result to byte B8.
Initial state:

Byte	B5	= 2A (hex)
Byte	B6	= B6 (hex)
Byte	B8	= ?

Function	STL	Accu. content	Operand content
Invert byte B6, and load into the word accu.	LN B6	2A	
Gate the contents of word accumulator and byte B5 with AND.	A B5		B6
Assign the gating result to byte B8.	= B8		22

LOAD TWO'S COMPLEMENT (L–)

Syntax: L– (LOAD MINUS)

Operands: B, W, D, K

Action:

Load the two's complement of the addressed operand, or of a constant, into the word accumulator. If necessary, the control fills the accumulator with the correct algebraic sign. The two's complement allows negative numbers to be stored (i.e., a number loaded with the L– command is displayed in the accumulator with an inverted sign). This command can be used only with word processing.

Example:

Negate the content of byte B5 and then add it to the content of byte B6. Assign the result to byte B8.
Initial state:
Byte B5 = 15 (dec)
Byte B6 = 20 (dec)
Byte B8 = ?

Function	STL	Accu. content	Operand content
Load byte B5 into the word accumulator, invert the algebraic sign.	L– B5	–15	+15
Add the contents of the word accumulator and byte B6.	+ B6	+5	+20
Assign the gating result to byte B8.	= B8	+5	+5

LOAD BYTE (LB)

Syntax: LB (LOAD BYTE)

Operands: M, I, O, T, C

Action:

Copy 8 markers, inputs, outputs, timer or counters with ascending numbering into the word accumulator. Each operand occupies one bit in the accumulator. The control saves the entered operand address in the accumulator as LSB, the entered address +1 as LSB +1 etc. The last (8th) operand becomes the MSB! If necessary, the control fills the accumulator with the correct algebraic sign.

Example:

A pure-binary coded value is read through inputs I3 to I10 and saved in byte B8 in order to process it later.

Initial state:

Input	I3	= 1	Input	I7	= 0
Input	I4	= 1	Input	I8	= 1
Input	I5	= 1	Input	I9	= 1
Input	I6	= 0	Input	I10	= 0

Function	STL	Accu. content	Operand content							
		7 6 5 4 3 2 1 0	I10	I9	I8	I7	I6	I5	I4	I3
Load inputs I3 to I10 into the accumulator (bit 0 to bit 7).	LB I3	1 1 1 0 0 1 1 0	0	1	1	0	0	1	1	1
			7	6	5	4	3	2	1	0
Assign accumulator contents to byte 8.	= B8	1 1 1 0 0 1 1 0	1	1	1	0	0	1	1	0

LOAD WORD (LW)

Syntax: LW (LOAD WORD)

Operands: M, I, O, T, C

Action:

Copy 16 markers, inputs, outputs, timer or counters with ascending numbering into the word accumulator. Each operand occupies one bit in the accumulator. The control saves the entered operand address in the accumulator as LSB, the entered address +1 as LSB +1 etc. The last (16th) operand becomes the MSB! If necessary, the control fills the accumulator with the correct algebraic sign.

Example:

See example command LB. Use command LW in the same way as LB. However, the control processes 16 operands.

LOAD DOUBLE WORD (LD)

Syntax: LD (LOAD DOUBLE WORD)

Operands: M, I, O, T, C

Action:

Copy 32 markers, inputs, outputs, timer or counters with ascending numbering into the word accumulator. Each operand occupies one bit in the accumulator. The control saves the entered operand address in the accumulator as LSB, the entered address +1 as LSB +1 etc. The last (32nd) operand becomes the MSB! If necessary, the control fills the accumulator with the correct algebraic sign.

Example:

See example command LB. Use command LD in the same way as LB. However, the control processes 32 operands.

ASSIGN (=)

Logic processing with the LOAD command

Syntax: = (STORE)

Operands: M, I, O, T, C

Action:

Assign the content of the logic accumulator to the addressed operand. Use the = command only at the end of a sequence of logical gating operations in order to transfer a gating result to a logic operand. This command can be used several times in succession (see example).

Example:

Gate the inputs I4 and I5 with AND, and assign the result to outputs O2 and O5.
Initial state:

Input	I4	= 1
Input	I5	= 0
Output	O2	= ?
Output	O5	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	1	1
Gate the content of the logic accumulator and input I5 with AND.	A I5	0	0
Assign the gating result to output O2.	= O2	0	0
Assign the gating result to output O5.	= O5	0	0

Word Processing with the ASSIGN Command

Syntax: = (STORE)

Operands: B, W, D

Action:

Assign the content of the word accumulator to the addressed operand. Unlike bit processing, in word processing you can also use the = command within a sequence of word-gating operations. This command can be used several times in succession.

Example:

Gate a constant and byte B5 with AND, and assign the result to byte B8 and byte B10.
Initial state:

Constant	54	= 36 (hex)
Byte	B5	= 2A (hex)
Byte	B8	= ?
Byte	B10	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K+54	36	
Assign the contents of the word accumulator to byte B8.	= B8	36	36
Gate the contents of word accumulator and byte B5 with AND.	A B5	22	2A
Assign the gating result to byte B8.	= B8	22	22
Assign the gating result to byte B10.	= B10	22	22

ASSIGN BYTE (B=)

Syntax: B= (STORE BYTE)

Operands: M, I, O, T, C

Action:

Assign 8 markers from the word accumulator to inputs, outputs, timers or counters with ascending numbering. Every bit occupies an operand. The control assigns the LSB in the accumulator to the operand address specified in the command, the specified address +1 as LSB +1 etc. The last (8th) operand is assigned the MSB.

Example:

See example command W=. Use command B= in the same way as W=. However, the control processes 8 operands.

ASSIGN WORD (W=)

Syntax: W= (STORE WORD)

Operands: M, I, O, T, C

Action:

Assign 16 markers from the word accumulator to inputs, outputs, timers or counters with ascending numbering. Every bit occupies an operand. The control assigns the LSB in the accumulator to the operand address specified in the command, the specified address +1 as LSB +1 etc. The last (16th) operand is assigned the MSB.

Example:

Transfer a certain bit pattern, located in word W8, to the output addresses O1 to O16.
Initial state:
Word W8 = 36FF (hex)

Function	STL	Accu. content	Operand content
Load content of word W8 into the word accumulator.	L W8	36FF	
			O16 ... O1
Assign accumulator content to outputs O5 to O20.	W= O1	36FF	0 1 1 0 1 1 0 1 1 1 1 1 1 1 1 1

ASSIGN DOUBLE WORD (D=)

Syntax: D= (STORE DOUBLE WORD)

Operands: M, I, O, T, C

Action:

Assign 32 markers from the word accumulator to inputs, outputs, timers or counters with ascending numbering. Every bit occupies an operand. The control assigns the LSB in the accumulator to the operand address specified in the command, the specified address +1 as LSB +1 etc. The last (32nd) operand is assigned the MSB.

Example:

See example command W=. Use command D= in the same way as W=. However, the control processes 32 operands.

ASSIGN NOT (=N)

Logic processing

Syntax: =N (STORE NOT)

Operands: M, I, O, T, C

Action:

Assign the complement of the logic accumulator to the addressed operand. For procedure, see the example for the command ASSIGN (=).

Word processing

Syntax: =N (STORE NOT)

Operands: B, W, D

Action:

Assign the complement of the word accumulator to the addressed operand. For procedure, see the example for the command ASSIGN (=).

ASSIGN TWO'S COMPLEMENT (=–)

Syntax: =– (STORE MINUS)

Operands: B, W, D

Action:

Assign the TWO'S COMPLEMENT of the word accumulator to the addressed operand. For procedure, see the example for the command ASSIGN (=).

SET (S)

Syntax: S (SET)

Operands: M, I, O, T, C

Action:

If the logic accumulator = 1, then set the addressed operand to 1; otherwise, do not change it. Use the S command at the end of a sequence of logical gating operations in order to influence an operand, depending on the result of gating. This command can be used several times in succession (see example).

Example:

Gate input I4 and I5 with OR. If the gating result is 1, then set output O2 and marker M500.
Initial state:

Input	I4	= 1
Input	I5	= 0
Output	O2	= ?
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	1	1
Gate the content of the logic accumulator and input I5 with OR.	O I5	1	0
Since the result of the operation is 1, set output O2.	S O2	1	1
Since the result of the operation is 1, set marker M500.	S M500	1	1

RESET (R)

Syntax: R (RESET)

Operands: M, I, O, T, C

Action:

If the logic accumulator = 1, then set the addressed operand to 0; otherwise, do not change it. Use the R command at the end of a sequence of logical gating operations in order to influence an operand, depending on the result of gating. This command can be used several times in succession (see example).

Example:

Gate input I4 and I5 with OR. If the gating result is 1, then reset output O2 and marker M500.
Initial state:

Input	I4	= 1
Input	I5	= 0
Output	O2	= ?
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	1	1
Gate the content of the logic accumulator and input I5 with OR.	O I5	1	0
Since the result of the operation is 1, reset output O2.	R O2	1	0
Since the result of the operation is 1, reset marker M500.	R M500	1	0

SET NOT (SN)

Syntax: SN (SET NOT)

Operands: M, I, O, T, C

Action:

If the logic accumulator = 0, then set the addressed operand to 1; otherwise, do not change it. Use the SN command at the end of a sequence of logical gating operations in order to influence an operand, depending on the result of gating. This command can be used several times in succession (see example).

Example:

Gate input I4 and I5 with OR. If the gating result is 0, then set output O2 and marker M500.
Initial state:

Input	I4	= 0
Input	I5	= 0
Output	O2	= ?
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	0	0
Gate the content of the logic accumulator and input I5 with OR.	O I5	0	0
Since the result of the operation is 0, set output O2.	SN O2	0	1
Since the result of the operation is 0, set marker M500.	SN M500	0	1

RESET NOT (RN)

Syntax: RN (RESET NOT)

Operands: M, I, O, T, C

Action:

If the logic accumulator = 0, then set the addressed operand to 0; otherwise, do not change it. Use the RN command at the end of a sequence of logical gating operations in order to influence an operand, depending on the result of gating. This command can be used several times in succession (see example).

Example:

Gate input I4 and I5 with OR. If the gating result is 0, then reset output O2 and marker M500.
Initial state:

Input	I4	= 0
Input	I5	= 0
Output	O2	= ?
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	0	0
Gate the content of the logic accumulator and input I5 with OR.	O I5	0	0
Since the result of the operation is 0, reset output O2.	RN O2	0	0
Since the result of the operation is 0, reset marker M500.	RN M500	0	0

AND (A)

Logic processing with the AND command

Syntax: A (AND)

Operands: M, I, O, T, C

Action:

At the beginning of a logic sequence, this command functions like an L command (i.e., the logical state of the operand is loaded into the logic accumulator). This is to ensure compatibility with the TNC 355, which does not have the special L command. In PLC programs, a sequence of logical gating operations should always begin with a load command (see L, LN, L–).

Within a logic sequence, gate the content of the logic accumulator and the logical state of the operand with AND. The control saves the result of the operation in the logic accumulator.

Example:

Gate the inputs I4 and I5 with AND, and assign the result to output O2.
Initial state:

Input	I4	= 1
Input	I5	= 0
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	1	1
Gate the content of the logic accumulator and input I5 with AND.	A I5	0	1
Assign the gating result to output O2.	= O2	0	0

Word Processing with the AND Command

Syntax: A (AND)

Operands: B, W, D, K

Action:

Gate the contents of the word accumulator and the operand with AND. In accordance with the different data widths of the operands (B = 8 bits; W = 16 bits; D = K = 32 bits), 8, 16, or 32 bits, respectively, are influenced in the accumulator. Thus, bit 0 of the accumulator is gated with bit 0 of the operand, bit 1 of the accumulator with bit 1 of the operand, etc. The control saves the result of the operation in the word accumulator.

Example:

Gate the content of byte B5 and byte B6 with AND, and assign the result to byte B8.
Initial state:

Byte	B5	= 2A (hex)
Byte	B6	= 36 (hex)
Byte	B8	= ?

Function	STL	Accu. content	Operand content
Load byte B6 into the word accumulator.	L B6	2A	2A
Gate the contents of word accumulator and byte B5 with AND.	A B5	22	36
Assign the gating result to byte B8.	= B8	22	22

AND NOT (AN)

Logic Processing with the AND NOT Command

Syntax: AN (AND NOT)

Operands: M, I, O, T, C

Action:

At the beginning of a logic sequence, this command functions like an LN command (i.e., the logical state of the operand is loaded into the logic accumulator). However, you should always begin a sequence of logical gating operations with a load command (see L, LN, L–).

Within a logic sequence, gate the content of the logic accumulator and the logical state of the operand with AND NOT. The control saves the result of the operation in the logic accumulator.

Example:

Gate the inputs I4 and I5 with AND NOT, and assign the result to output O2.
Initial state:

Input	I4	= 1
Input	I5	= 1
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	1	1
Gate the content of logic accumulator and input I5 with AND NOT.	AN I5	1	1
Assign the gating result to output O2.	= O2	1	1

Word Processing with the AND NOT Command

Syntax: AN (AND NOT)

Operands: B, W, D, K

Action:

Gate the contents of the word accumulator and the operand with AND NOT. In accordance with the different data widths of the operands (B = 8 bits; W = 16 bits; D = K = 32 bits), 8, 16, or 32 bits, respectively, are influenced in the accumulator. Thus, bit 0 of the accumulator is gated with bit 0 of the operand, bit 1 of the accumulator with bit 1 of the operand, etc. The control saves the result of the operation in the word accumulator.

Example:

Gate the content of words W4 and W6 with AND NOT, and assign the result to word W8.
Initial state:

Word	W4	= 36 AA (hex)
Word	W6	= 3C 36 (hex)
Word	W8	= ?

Function	STL	Accu. content	Operand content
Load W6 into the word accumulator.	L W6	3C36	3C36
Gate the contents of word accumulator and word W4 with AND NOT.	AN W4	814	36AA
Assign the gating result to word W8.	= W8	814	814

OR (O)

Logic Processing with the OR Command

Syntax: O (OR)

Operands: M, I, O, T, C

Action:

At the beginning of a logic sequence, this command functions like an L command (i.e., the logical state of the operand is loaded into the logic accumulator). However, you should always begin a sequence of logical gating operations with a load command (see L, LN, L–).

Within a logic sequence, gate the content of the logic accumulator and the logical state of the operand with OR. The control saves the result of the operation in the logic accumulator.

Example:

Gate the inputs I4 and I5 with OR, and assign the result to output O2.
Initial state:

Input	I4	= 0
Input	I5	= 1
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	0	0
Gate the content of the logic accumulator and input I5 with OR.	O I5	1	1
Assign the gating result to output O2.	= O2	1	1

Word Processing with the OR Command

Syntax: O (OR)

Operands: B, W, D, K

Action:

Gate the contents of the word accumulator and the operand with OR. In accordance with the different data widths of the operands (B = 8 bits; W = 16 bits; D = K = 32 bits), 8, 16, or 32 bits, respectively, are influenced in the accumulator. Thus, bit 0 of the accumulator is gated with bit 0 of the operand, bit 1 of the accumulator with bit 1 of the operand, etc. The control saves the result of the operation in the word accumulator.

Example:

Gate the content of byte B5 and byte B6 with OR, and assign the result to word W8.
Initial state:

Byte	B5	= 2A (hex)
Byte	B6	= 36 (hex)
Word	W8	= ?

Function	STL	Accu. content	Operand content
Load byte B6 into the word accumulator.	L B6	36	36
Gate the contents of the word accumulator and byte B5 with OR.	O B5	3E	2A
Assign the gating result to word W8.	= W8	3E	3E

OR NOT (ON)

Logic Processing with the OR NOT Command

Syntax: ON (OR NOT)

Operands: M, I, O, T, C

Action:

At the beginning of a logic sequence, this command functions like an LN command (i.e., the complement of the operand is loaded into the logic accumulator). However, you should always begin a sequence of logical gating operations with a load command (see L, LN, L–).

Within a logic sequence, gate the content of the logic accumulator and the logical state of the operand with OR NOT. The control saves the result of the operation in the logic accumulator.

Example:

Gate the inputs I4 and I5 with OR NOT, and assign the result to output O2.
Initial state:

Input	I4	= 0
Input	I5	= 0
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	0	0
Gate the content of logic accumulator and input I5 with OR NOT.	ON I5	1	0
Assign the gating result to output O2.	= O2	1	1

Word Processing with the OR NOT Command

Syntax: ON (OR NOT)

Operands: B, W, D, K

Action:

Gate the contents of the word accumulator and the operand with OR NOT. In accordance with the different data widths of the operands (B = 8 bits; W = 16 bits; D = K = 32 bits), 8, 16, or 32 bits, respectively, are influenced in the accumulator. Thus, bit 0 of the accumulator is gated with bit 0 of the operand, bit 1 of the accumulator with bit 1 of the operand, etc. The control saves the result of the operation in the word accumulator.

Example:

Gate the content of words W4 and W6 with OR NOT, and assign the result to word W8.
Initial state:

Word	W4	= 36 AA (hex)
Word	W6	= 3C 36 (hex)
Word	W8	= ?

Function	STL	Accu. content	Operand content
Load W6 into the word accumulator.	L W6	3C36	3C36
Gate the content of word accumulator and word W4 with OR NOT.	ON W4	814	36AA
Assign the gating result to word W8.	= W8	814	814

EXCLUSIVE OR (XO)

Logic Processing with the EXCLUSIVE OR Command

Syntax: XO (EXCLUSIVE OR)

Operands: M, I, O, T, C

Action:

At the beginning of a logic sequence, this command functions like an L command (i.e., the logical state of the operand is loaded into the logic accumulator). However, you should always begin a sequence of logical gating operations with a load command (see L, LN, L–).

Within a logic sequence, gate the content of the logic accumulator and the logical state of the operand with EXCLUSIVE OR. The control saves the result of the operation in the logic accumulator.

Example:

Gate the inputs I4 and I5 with EXCLUSIVE OR, and assign the result to output O2.
Initial state:

Input	I4	= 1
Input	I5	= 1
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L I4	1	1
Gate the content of logic accumulator and input I5 with EXCLUSIVE OR.	XO I5	0	1
Assign the gating result to output O2.	= O2	0	0

Word Processing with the EXCLUSIVE OR Command

Syntax: XO (EXCLUSIVE OR)

Operands: B, W, D, K

Action:

Gate the contents of the word accumulator and the operand with EXCLUSIVE OR. In accordance with the different data widths of the operands (B = 8 bits; W = 16 bits; D = K = 32 bits), 8, 16, or 32 bits, respectively, are influenced in the accumulator. Thus, bit 0 of the accumulator is gated with bit 0 of the operand, bit 1 of the accumulator with bit 1 of the operand, etc. The control saves the result of the operation in the word accumulator.

Example:

Gate the content of byte B5 and byte B6 with EXCLUSIVE OR, and assign the result to word W8.
Initial state:

Byte	B5	= 2A (hex)
Byte	B6	= 36 (hex)
Word	W8	= ?

Function	STL	Accu. content	Operand content
Load byte B6 into the word accumulator.	L B6	36	36
Gate the contents of the word accumulator and byte B5 with EXCLUSIVE OR.	XO B5	1C	2A
Assign the gating result to word W8.	= W8	1C	1C

EXCLUSIVE OR NOT (XON)

Logic processing with the EXCLUSIVE OR NOT command

Syntax: XON (EXCLUSIVE OR NOT)

Operands: M, I, O, T, C

Action:

At the beginning of a logic sequence, this command functions like an LN command (i.e., the logical state of the operand is loaded into the logic accumulator). However, you should always begin a sequence of logical gating operations with a load command (see L, LN, L–).

Within a logic sequence, gate the content of the logic accumulator and the logical state of the operand with EXCLUSIVE OR NOT. The control saves the result of the operation in the logic accumulator.

Example:

Gate the inputs I4 and marker M500 with EXCLUSIVE OR NOT, and assign the result to output O2.
Initial state:

Input	I4	= 0
Marker	M500	= 0
Output	O2	= ?

Function	STL	Accu. content	Operand content
Load the operand content into the logic accu.	L M500	0	0
Gate the content of logic accumulator and input I4 with EXCLUSIVE OR NOT.	XON I4	1	0
Assign the gating result to output O2.	= O2	1	1

Word Processing with the EXCLUSIVE OR NOT Command

Syntax: XON (EXCLUSIVE OR NOT)

Operands: B, W, D, K

Action:

Gate the contents of the word accumulator and the operand with EXCLUSIVE OR NOT. In accordance with the different data widths of the operands (B = 8 bits; W = 16 bits; D = K = 32 bits), 8, 16, or 32 bits, respectively, are influenced in the accumulator. Thus, bit 0 of the accumulator is gated with bit 0 of the operand, bit 1 of the accumulator with bit 1 of the operand, etc. The control saves the result of the operation in the word accumulator.

Example:

Gate the content of words W4 and W6 with EXCLUSIVE OR NOT, and assign the result to word W8.
Initial state:

Word	W4	= 36 AA (hex)

Word W6 = 3C 36 (hex)
Word W8 = ?

Function	STL	Accu. content	Operand content
Load W6 into the word accumulator.	L W6	3C36	3C36
Gate the contents of word accumulator and word W4 with EXCLUSIVE OR NOT.	XON W4	FFFFF563	36AA
Assign the gating result to word W8.	= W8	FFFFF563	FFFFF563

ADDITION (+)

Syntax: + (PLUS)

Operands: B, W, D, K

Action:

The control extends the operand to the width of the accumulator (32 bits) and then adds the content of the operand to the content of the word accumulator. The result of the operation is stored in the word accumulator where you can process it further.

Example:

Add the constant and the number saved in word W6, then assign the result to double word D8.
Initial state:

Constant		= 10 000 (dec)
Word	W6	= 200 (dec)
Double word	D8	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K10000	10000	
Add the content of the word accumulator and word W6.	+ W6	10200	200
Assign the result to double word D8.	= D8	10200	10200

SUBTRACTION (–)

Syntax: – (MINUS)

Operands: B, W, D, K

Action:

The control extends the operand to the width of the accumulator (32 bits) and then subtracts the content of the operand from the content of the word accumulator. The result of the operation is stored in the word accumulator where you can process it further.

Example:

Subtract the number saved in word W6 from the constant, and then assign the result to double word D8.
Initial state:

Constant		= 10 000 (dec)
Word	W6	= 200 (dec)
Double word	D8	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K10000	10000	
Subtract word W6 from the content of the word accumulator.	– W6	9800	9800
Assign the result to double word D8.	= D8	9800	9800

MULTIPLICATION (X)

Syntax: x (MULTIPLY)

Operands: B, W, D, K

Action:

The control extends the operand to the width of the accumulator (32 bits) and then multiplies the content of the operand with the content of the word accumulator. The result of the operation is stored in the word accumulator where you can process it further. If the control cannot execute the multiplication correctly, it then sets marker M4200; otherwise, it resets it.

Example:

Multiply the constant and the number saved in word W6, then assign the result to double word D8.
Initial state:

Constant		= 100 (dec)
Word	W6	= 20 (dec)
Double word	D8	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K100	100	
Multiply the content of the word accumulator with word W6.	x W6	2000	20
Assign the result to double word D8.	= D8	2000	2000

		Type
4200	Overflow during multiplication	M

DIVISION (/)

Syntax: / (DIVIDE)

Operands: B, W, D, K

Action:

The control extends the operand to the width of the accumulator (32 bits) and then divides the content of the word accumulator by the content of the operand. The result of the operation is stored in the word accumulator where you can process it further. If the control cannot execute the division correctly, it then sets marker M4201; otherwise, it resets it.

Example:

Divide the constant by the number saved in word W6, then assign the result to double word D8.
Initial state:

Constant		= 100 (dec)
Word	W6	= 20 (dec)
Double word	D8	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K100	100	
Divide the content of the word accumulator by word W6.	/ W6	5	20
Assign the result to double word D8.	= D8	5	5

		Type
4201	Division by 0	M

REMAINDER (MOD)

Syntax: MOD (MODULO)

Operands: B, W, D, K

Action:

The control extends the operand to the width of the accumulator (32 bits) and then calculates the remainder resulting from the division of the content of the word accumulator by the content of the operand. The remainder is stored in the word accumulator where you can process it further. If the control cannot execute the MOD command correctly, it sets marker M4202; otherwise, it resets it.

Example:

Divide the number saved in word W6 by the constant, then calculate the REMAINDER and assign the result to double word D8.

Initial state:

Word	W6	= 50 (dec)
Constant		= 15 (dec)
Double word	D8	= ?

Function	STL	Accu. content	Operand content
Load W6 into the word accumulator.	L W6	50	50
Divide the content of the word accumulator by a constant, then save the integral REMAINDER in the word accumulator.	MOD K15	11	15
Assign the REMAINDER to double word D8.	= D8	11	11

		Type
4202	Incorrectly executed modulo	M

INCREMENT (INC)

INCREMENT operand

Syntax: INC (INCREMENT)

Operands: B, W, D

Action:

Increase the content of the addressed operand by one.

INCREMENT word accumulator

Syntax: INCW (INCREMENT WORD)

Operands: None

Action:

Increase the content of the word accumulator by one.

INCREMENT index register

Syntax: INCX (INCREMENT INDEX)

Operands: None

Action:

Increase the content of the index register by one.

DECREMENT (DEC)

DECREMENT operand

Syntax: DEC (DECREMENT)

Operands: B, W, D

Action:

Decrease the content of the addressed operand by one.

DECREMENT word accumulator

Syntax: DECW (DECREMENT WORD)

Operands: None

Action:

Decrease the content of the word accumulator by one.

DECREMENT index register

Syntax: DECX (DECREMENT INDEX)

Operands: None

Action:

Decrease the content of the index register by one.

EQUAL TO (==)

Syntax: == (EQUAL)

Operands: B, W, D, K

Action:

This command sets off a direct transition from word to logical processing. Gate the content of the word accumulator with the content of the addressed operand. If the word accumulator and the operand are equal, the condition is true and the control sets the logic accumulator to 1. If they are not equal, the logic accumulator is set to 0. The comparison takes place over the number of bits corresponding to the operand:
where B = 8 bits, W = 16 bits, and D = K = 32 bits.

Example:

Compare a constant with the content of double word D8, and assign the result to marker M500.
Initial state:

Constant		= 16 000 (dec)
Double word	D8	= 15 000 (dec)
Marker	M300	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K16000	16000	
Gate the content of the word accumulator with the operand content D8; if not equal, set the logic accumulator to 0.	== D8	0	15000
Assign the result to marker M500.	= M500	0	0

LESS THAN (<)

Syntax: < (LESS THAN)

Operands: B, W, D, K

Action:

This command sets off a direct transition from word to logical processing. Gate the content of the word accumulator with the content of the addressed operand. If the word accumulator is less than the operand, the condition is true and the control sets the logic accumulator to 1. If the word accumulator is greater than or equal to the operand, it sets the logic accumulator to 0. The comparison takes place over the number of bits in the operand:
where B = 8 bits, W = 16 bits, and D = K = 32 bits.

Example:

Compare a constant with the content of double word D8, and assign the result to marker M500.
Initial state:

Constant		= 16 000 (dec)
Double word	D8	= 15 000 (dec)
Marker	M500	= ? i

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K16000	16000	
Check whether word accumulator < operand; if not, set logic accumulator to 0.	< D8	0	15000
Assign the result to marker M500.	= M500	0	0

GREATER THAN (>)

Syntax: > (GREATER THAN)

Operands: B, W, D, K

Action:

This command sets off a direct transition from word to logical processing. Gate the content of the word accumulator with the content of the addressed operand. If the word accumulator is greater than the operand, the condition is true and the control sets the logic accumulator to 1. If the word accumulator is less than or equal to the operand, it sets the logic accumulator to 0. The comparison takes place over the number of bits in the operand:
where B = 8 bits, W = 16 bits, and D = K = 32 bits.

Example:

Compare a constant with the content of double word D8, and assign the result to marker M500.
Initial state:

Constant		= 16 000 (dec)
Double word	D8	= 15 000 (dec)
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K16000	16000	
Check whether word accumulator > operand; if so, set logic accu to 1.	> D8	1	15000
Assign the result to marker M500.	= M500	1	1

LESS THAN OR EQUAL TO (<=)

Syntax: <= (LESS EQUAL)

Operands: B, W, D, K

Action:

This command sets off a direct transition from word to logical processing. Gate the content of the word accumulator with the content of the addressed operand. If the word accumulator is less than or equal to the operand, the condition is true and the control sets the logic accumulator to 1. If the word accumulator is greater than the operand, it sets the logic accumulator to 0. The comparison takes place over the number of bits in the operand:
where B = 8 bits, W = 16 bits, and D = K = 32 bits.

Example:

Compare a constant with the content of double word D8, and assign the result to marker M500.
Initial state:

Constant		= 16 000 (dec)
Double word	D8	= 15 000 (dec)
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K16000	16000	
Check whether word accumulator <= operand; if not, set logic accu to 0.	<= D8	0	15000
Assign the result to marker M500.	= M500	0	0

GREATER THAN OR EQUAL TO (>=)

Syntax: >= (GREATER EQUAL)

Operands: B, W, D, K

Action:

This command sets off a direct transition from word to logical processing. Gate the content of the word accumulator with the content of the addressed operand. If the word accumulator is greater than or equal to the operand, the condition is true and the control sets the logic accumulator to 1. If the word accumulator is less than the operand, it sets the logic accumulator to 0. The comparison takes place over the number of bits in the operand:
where B = 8 bits, W = 16 bits, and D = K = 32 bits.

Example:

Compare a constant with the content of double word D8, and assign the result to marker M500.
Initial state:

Constant		= 16 000 (dec)
Double word	D8	= 15 000 (dec)
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K16000	16000	
Check whether word accumulator >= operand; if so, set logic accu to 1.	>= D8	1	15000
Assign the result to marker M500.	= M500	1	1

NOT EQUAL (<>)

Syntax: <> (NOT EQUAL)

Operands: B, W, D, K

Action:

This command sets off a direct transition from word to logical processing. Gate the content of the word accumulator with the content of the addressed operand. If the word accumulator and the operand are not equal, the condition is true and the control sets the logic accumulator to 1. If they are equal, the logic accumulator is set to 0. The comparison takes place over the number of bits corresponding to the operand:
where B = 8 bits, W = 16 bits, and D = K = 32 bits.

Example:

Compare a constant with the content of double word D8, and assign the result to marker M500.
Initial state:

Constant		= 16 000 (dec)
Double word	D8	= 15 000 (dec)
Marker	M500	= ?

Function	STL	Accu. content	Operand content
Load the constant into the word accumulator.	L K16000	16000	
Check whether word accumulator <> operand; if so, set logic accu to 1.	<> D8	1	15000
Assign the result to marker M500.	= M500	1	1

AND [ ] (A[ ])

Syntax: A[ ] (AND [ ])

Operands: None

Action:

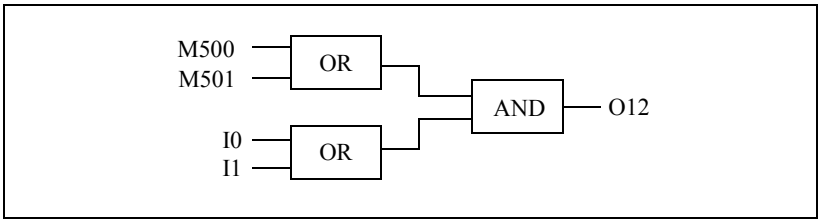
The use of parentheses enables you to alter the sequence of processing logical commands in a statement list. The opening-parenthesis command puts the content of the accumulator onto the program stack. If you address the logic accumulator in the last command before an opening-parenthesis statement, the control loads the content of the logic accumulator onto the program stack. If you address the word accumulator, the control loads the contents of the word accumulator. The “close-parenthesis” statement gates the buffered value from the program stack with the content of the logic accumulator or the word accumulator, depending on which accumulator was addressed prior to the “open-parenthesis” instruction. The control assigns the result of the gating operation to the corresponding accumulator. Maximum nesting depth: 16 parentheses.

Note: The sequence of function is the same for word processing; however, the control writes the complete word accumulator onto the program stack.

Example:

Example for the commands AND [ ], AND NOT [ ], OR [ ], OR NOT [ ], EXCLUSIVE OR [ ], EXCLUSIVE OR NOT [ ]:

Use parentheses to develop a statement list in accordance with the following logic circuit diagram:



Initial state:

Marker	M500	= 0
Input	I0	= 0
Marker	M501	= 1
Input	I1	= 1
Output	O12	= ?

Function	STL	Accu. content	Operand content
Load marker M500 into the logic accumulator.	L M500	0	0
Gate logic accumulator with marker M501.	O M501	1	1
Opening parenthesis: Buffer the accu content onto the program stack.	A[		
Load the state of input I0 into the logic accu.	L I0	0	0
Gate the logic accu with the state of input I1.	O I1	1	1
Closing parenthesis: Gate the accumulator content with the program stack (A[, O[...).	]		
Assign the result of the total operation to output O12.	= O12	1	1

AND NOT [ ] (AN[ ])

Syntax: AN[ ] (AND NOT [ ])

Operands: None

Action:

See example A[ ] (AND [ ])

OR [ ] (O[ ])

Syntax: O[ ] (OR [ ])

Operands: None

Action:

See example A[ ] (AND [ ])

OR NOT [ ] (ON[ ])

Syntax: ON[ ] (OR NOT [ ])

Operands: None

Action:

See example A[ ] (AND [ ])

EXCLUSIVE OR [ ] (XO[ ])

Syntax: XO[ ] (EXCL: OR [ ])

Operands: None

Action:

See example A[ ] (AND [ ])

EXCLUSIVE OR NOT [ ] (XON[ ])

Syntax: XON[ ] (EXCL: OR NOT [ ])

Operands: None

Action:

See example A[ ] (AND [ ])

ADDITION [ ] (+[ ])

Syntax: +[ ] (PLUS[ ])

Operands: None

Action:

Use parentheses together with arithmetical commands **only** for word processing. By using parentheses you can change the sequence of processing in a statement list. The opening-parenthesis command puts the content of the word accumulator onto the program stack. This clears the accumulator for calculation of intermediate results. The closing-parenthesis command gates the buffered value from the program stack with the content of the word accumulator. The control saves the result in the accumulator again. Maximum nesting depth: 16 parentheses. If an error occurs during calculation, the control sets the marker M4201.

Example:

Example for the commands ADD [ ], SUBTRACT [ ], MULTIPLY [ ], DIVIDE [ ], REMAINDER [ ].

Divide a constant by double word D36, add the result to double word D12, and assign the result to double word D100.
Initial state:

Constant		= 1000 (dec)
Double word	D12	= 15000 (dec)
Double word	D36	= 100 (dec)
Double word	D100	= ?

Function	STL	Accu. content	Operand content
Load the double word D12 into the word accu.	L D12	15000	15000
Opening parenthesis: Buffer the accu content onto the program stack.	+[		
Load the constant K 1000 into the word accu.	L K1000	1000	
Divide the word accu by the content of the double word D36.	/ D36	10	100
Closing parenthesis: Gate the accumulator content with the program stack (+[, –[.....).	]		
Assign the result of the total operation to double word D100.	= D100	15010	15010

SUBTRACTION [ ] (–[ ])

Syntax: –[ ] (MINUS –[ ])

Operands: None

Action:

See example for ADDITION [ ]

MULTIPLICATION [ ] (x[ ])

Syntax: x[ ] (MULTIPLY [ ])

Operands: None

Action:

See example for ADDITION [ ]

DIVISION [ ] (/[ ])

Syntax: /[ ] (DIVIDE [ ])

Operands: None

Action:

See example for ADDITION [ ]

REMAINDER [ ] (MOD[ ])

Syntax: MOD[ ] (MODULO [ ])

Operands: None

Action:

See example for ADDITION [ ]

EQUAL TO [ ] (==[ ])

Syntax: ==[ ] (EQUAL[ ])

Operands: None

Action:

By using parentheses you can change the sequence of processing comparative commands in a statement list. The opening-parenthesis command puts the content of the word accumulator onto the program stack. This clears the accumulator for calculation of intermediate results.

The closing-parenthesis command gates the buffered value from the program stack with the content of the word accumulator. The control saves the result in the accumulator again. Maximum nesting depth: 16 parentheses.

Comparative commands cause a direct transition from word to logical processing. If the specified comparative condition is true, the control sets the logic accumulator to 1; if the condition is not fulfilled, it sets it to 0.

Example:

Multiply a constant with double word D36, compare the result with double word D12, and assign the result to output O15.
Initial state:

Constant		= 1000 (dec)
Double word	D12	= 15 000 (dec)
Double word	D36	= 10 (dec)
Output	O15	= ?

Function	STL	Accu. content	Operand content
Load the double word D12 into the word accu.	L D12	15000	15000
Opening parenthesis: Buffer the accu content onto the program stack.	== [		
Load the constant into the word accumulator.	L K1000	1000	
Multiply the content of the word accumulator with double word D36.	x D36	10000	10
Closing parenthesis: Gate the accumulator content with the program stack (==[, >=[...), if condition not fulfilled, set logic accumulator to 0.	]		
Assign the result to output O15.	= O15	0	0

LESS THAN [ ] (<[ ])

Syntax: <[ ] (LESS THAN [ ])

Operands: None

Action:

See example for EQUAL TO [ ]

GREATER THAN [ ] (>[ ])

Syntax: >[ ] (GREATER THAN [ ])

Operands: None

Action:

See example for EQUAL TO [ ]

LESS THAN OR EQUAL TO [ ] (<=[ ])

Syntax: <=[ ] (LESS EQUAL [ ])

Operands: None

Action:

See example for EQUAL TO [ ]

GREATER THAN OR EQUAL TO [ ] (>=[ ])

Syntax: >=[ ] (GREATER EQUAL [ ])

Operands: None

Action:

See example for EQUAL TO [ ]

NOT EQUAL [ ] (<>[ ])

Syntax: <>[ ] (NOT EQUAL [ ])

Operands: None

Action:

See example for EQUAL TO [ ]

SHIFT LEFT (<<)

Syntax: << (SHIFT LEFT)

Operands: B, W, D, K

Action:

A SHIFT LEFT statement multiplies the content of the word accumulator by two. This is done by simply shifting the bits in the accumulator by one place to the left. The result must lie in the range of -2 147 483 648 to +2 147 483 647; otherwise, the accumulator contains an undefined value. You define the number of shifts through the operand. The control fills the right end of the accumulator with zeros.

This statement is one of the arithmetic commands because it includes the sign bit. For this reason, and to save time, you should not use this command to isolate bits.

Example:

Shift the content of double word D8 four times to the left, then assign it to double word D12.
Initial state:
Double word D8 = 3E 80 (hex)
Double word D12 = ?

Function	STL	Accu. content	Operand content
Load the double word D8 into the word accu.	L D8	3E80	3E80
Shift the content of the word accumulator to the left by the number of bits that are specified in the operand.	<< K+1	7D00	
	<< K+1	FA00	
	<< K+1	1F400	
	<< K+1	3E800	
Assign the result to double word D12.	= D12	3E800	3E800

Instead of using the << K+1 command four times, simply use the << K+4 command.

SHIFT RIGHT (>>)

Syntax: >> (SHIFT RIGHT)

Operands: B, W, D, K

Action:

A SHIFT RIGHT statement divides the content of the word accumulator by two. This is done by simply shifting the bits by one place to the right. You define the number of shifts through the operand. The bits that the control shifts to the right out of the accumulator are then lost. The control extends the left side of the accumulator with the correct sign.

This statement is one of the arithmetic commands because it includes the sign bit. For this reason, and to save time, you should not use this command to isolate bits.

Example:

Shift the content of double word D8 four times to the right, then assign it to double word 12.
Initial state:
Double word D8 = 3E 80 (hex)
Double word D12 = ?

Function	STL	Accu. content	Operand content
Load the double word D8 into the word accu.	L D8	3E80	3E80
Shift the content of the word accumulator to the right by the number of bits that are specified in the operand.	>> K+1	1F40	
	>> K+1	FA0	
	>> K+1	7D0	
	>> K+1	3E8	
Assign the result to double word D12.	= D12	3E8	3E8

Instead of using the >> K+1 command four times, simply use the >> K+4 command.

BIT SET (BS)

Syntax: BS (BIT SET)

Operands: B, W, D, K, X

Action:

With the BIT SET command you can set each bit in the accumulator to 1. The corresponding bits are selected (addressed) by the content of the specified operand or by a constant. As to the bit numbering, bit 0 = LSB and bit 31 = MSB. For operand contents greater than 32, the control uses the operand value modulo 32 (i.e., the integral remainder of the result of the operand value divided by 32).

Example:

Load double word D8 into the accumulator, set bit 0 of the accumulator to 1, and save the result in double word D12.
Initial state:
Double word D8 = 3E 80 (hex)
Double word D12 = ?

Function	STL	Accu. content	Operand content
Load the double word D8 into the word accu.	L D8	3E80	3E80
Set the bit specified in the operand to 1.	BS K+0	3E81	
Assign the result to double word D12.	= D12	3E81	3E81

BIT CLEAR (BC)

Syntax: BC (BIT CLEAR)

Operands: B, W, D, K, X

Action:

With the BIT CLEAR command you can set each bit in the accumulator to 0. The corresponding bits are selected (addressed) by the content of the specified operand or by a constant. As to the bit numbering, bit 0 = LSB and bit 31 = MSB. For operand contents greater than 32, the control uses the operand value modulo 32 (i.e., the integral remainder of the result of the operand value divided by 32).

Example:

Load double word D8 into the accumulator, set bit 0 of the accumulator to 0, and save the result in double word D12.
Initial state:
Double word D8 = 3E 81 (hex)
Double word D12 = ?

Function	STL	Accu. content	Operand content
Load the double word D8 into the word accu.	L D8	3E81	3E81
Set the bit specified in the operand to 0.	BC K+0	3E80	
Assign the result to double word D12.	= D12	3E80	3E80

BIT TEST (BT)

Syntax: BT (BIT TEST)

Operands: B, W, D, K, X

Action:

-With the BIT TEST command, you can interrogate the status of each bit in the accumulator. With the BT command there is a direct transition from word to logic processing (i.e., the control checks the state of a bit in the word accumulator and then sets the logic accumulator). If the interrogated bit = 1, the control sets the logic accumulator to 1; otherwise, it sets it to 0. The corresponding bits are selected (addressed) by the content of the specified operand or by a constant. As to the bit numbering, bit 0 = LSB and bit 31 = MSB. For operand contents greater than 32, the control uses the operand value modulo 32 (i.e., the integral remainder of the result of the operand value divided by 32).

Example:

Load the double word D8 into the accumulator and assign the logical state of bit 0 to output O12.
Initial state:
Double word D8 = 3E 81 (hex)
Output O12 = ?

Function	STL	Accu. content	Operand content
Load the double word D8 into the word accu.	L D8	3E81	3E81
Check the state of the bit specified in the operand.	BT K+0	1	
Assign the result to output O12.	= O12	1	1

Push Data onto the Data Stack (PS)

Logic Processing with the PS Command

Syntax: PS (PUSH)

Operands: M, I, O, T, C

Action:

The PS command enables you to buffer data. To do this, the control loads the addressed operand onto the data stack. Because the data stack has a width of 32 bits, you must write to it with a minimum width of one word. The control copies the operand value into bit 7 of the data stack's current address. The vacant bits of the occupied memory remain undefined or unused. In the event of a stack overflow, the control outputs an error message.

Memory assignment in the data stack [bit]														
31 ... 14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x ...	x	x	x	x	x	x	L	x	x	x	x	x	x	x

Example:

See PSW command.

Word Processing with the PS Command

Syntax: PS (PUSH)

Operands: B, W, D, K

Action:

The PS command enables you to buffer data. The control copies the addressed operand value into the current address of the data stack. During the word processing, the control copies two words per PS command onto the data stack and extends the operand—in accordance with the MSB—with the correct algebraic sign. In the event of a stack overflow, the control displays an error message.

Data stack for byte, word, double word and constant [bit]			
31 24	23 16	15 8	7 0
x x x x x x x x	x x x x x x x x	x x x x x x x x	B B B B B B B B
x x x x x x x x	x x x x x x x x	W W W W W W W W	W W W W W W W W
D D D D D D D D	D D D D D D D D	D D D D D D D D	D D D D D D D D
K K K K K K K K	K K K K K K K K	K K K K K K K K	K K K K K K K K

Example:

See PSW command.

Pull Data from the Data Stack (PL)

Logic Processing with the PL Command

Syntax: PL (PULL)

Operands: M, I, O, T, C

Action:

The PL command is the counterpart to the PS command. Data that has been buffered with the PUSH command can be taken from the data stack by using the PULL command. The control copies bit 7 of the data stack's current address into the addressed operand. If the stack is empty, the control displays an error message.

Example:

See PSW command.

Word Processing with the PL Command

Syntax: PL (PULL)

Operands: B, W, D, K

Action:

The PL command is the counterpart to the PS command. Data that has been buffered with the PUSH command can be taken from the data stack by using the PULL command. During the word processing, the control copies with the PL command two words of the current data stack address into the addressed memory area. If the stack is empty, the control displays an error message.

Example:

See PSW command.

Push LOGIC ACCUMULATOR onto the Data Stack (PSL)

Syntax: PSL (PUSH LOGICACCU)

Operands: None

Action:

The PSL command enables you to buffer the logic accumulator. With the PSL command, the control copies the logic accumulator onto the data stack. Because the data stack has a width of 32 bits, you must write to it with a minimum width of one word. The control copies the operand value into bit 7 of the data stack's current address. The vacant bits of the occupied memory remain undefined or unused. In the event of a stack overflow, the control outputs an error message.

Memory assignment in the data stack [bit]															
31 ... 14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
x ...	x	x	x	x	x	x	L	x	x	x	x	x	x	x

Example:

See PSW command.

Push WORD ACCUMULATOR onto the Data Stack (PSW)

Syntax: PSW (PUSH WORDACCU)

Operands: None

Action:

The PSW command enables you to buffer the word accumulator. With the PSW command, the control copies the word accumulator onto the data stack. The content of the word accumulator (32 bits) occupies two words on the data stack. In the event of a stack overflow, the control displays an error message.

Example:

Since the sequence is the same for all stack operations, this example also applies to the commands PS, PL, PSW, PLL, PLW. The difference between the individual operations lies merely in the transferred data width.

Call Module 15 at a certain place in the program. After returning to the main program, restore the original accumulator content. Accumulator contents prior to the module call: 1A 44 3E 18

Function	**STL**	**Accu. content**	**Data stack**
Buffer the word accu in the data stack.	PSW	1A443E18	1A443E18
Call subroutine 15.	CM 15		
Restore data stack into word accumulator.	PLW	1A443E18	1A443E18

Pull LOGIC ACCUMULATOR from the Data Stack (PLL)

Syntax: PLL (PULL LOGICACCU)

Operands: None

Action:

The PLL command is the counterpart to the PSL command. Data that has been buffered with the PUSH command can be restored from the data stack by using the PULL command. The control copies bit 7 of the data stack's current address into the logic accumulator. If the stack is empty, the control displays an error message.

Example:

See PSW command.

Pull WORD ACCUMULATOR from the Data Stack (PLW)

Syntax: PLW (PULL WORDACCU)

Operands: None

Action:

The PLW command is the counterpart to the PSW command. Data that has been buffered with the PUSH command can be restored from the data stack by using the PULL command. During the word processing, the control copies with the PLW command two words of the current data stack address into the word accumulator. If the stack is empty, the control displays an error message.

Example:

See PSW command.

UNCONDITIONAL JUMP (JP)

Syntax: JP (JUMP)

Operands: Label (LBL)

Action:

After a JP command, the control jumps to the label that you have entered and resumes the program from there. The JP command interrupts a logic sequence.

Example:

See JPT command.

JUMP IF LOGIC ACCUMULATOR = 1 (JPT)

Syntax: JPT (JUMP IF TRUE)

Operands: Label (LBL)

Action:

The JPT command is a conditional jump command. If the logic accumulator = 1, the control resumes the program at the label that you have entered. If the logic accumulator = 0, the control does not jump. The JPT command interrupts a logic sequence.

Example:

This example also applies to the JP and JPF commands.

Depending on the state of the input I5, skip a certain program section.
Initial state:
Input I5 = 1

Function	STL	Accu. content	Operand content
Load the operand content into the logic accumulator.	L I5	1	1
If logic accumulator =1, jump to LBL 10.	JPT 10	1	
Skip the function.	L I3		
Skip the function.	O M500		
Skip the function.	= 020		
Label	LBL 10		
Resume the program run.	L M100	0	0

JUMP IF LOGIC ACCUMULATOR = 0 (JPF)

Syntax: JPF (JUMP IF FALSE)

Operands: Label (LBL)

Action:

The JPF command is a conditional jump command. If the logic accumulator = 0, the control resumes the program at the label that you have entered. If the logic accumulator = 1, the control does not jump. The JPF command interrupts a logic sequence.

Example:

See JPT command.

CALL MODULE (CM)

Syntax: CM (CALL MODULE)

Operands: Label (LBL)

Action:

After a CM command, the control calls the module that begins at the label that you have entered. Modules are independent subroutines that must be ended with the command EM. You can call modules as often as you wish from different places in your program. The CM command interrupts a logic sequence.

Example:

See CMF command.

CALL MODULE IF LOGIC ACCUMULATOR = 1 (CMT)

Syntax: CMT (CALL MODULE IF TRUE)

Operands: Label (LBL)

Action:

The CMT command is a conditional module call. If the logic accumulator = 1, the control calls the module that begins at the label you have entered. If the logic accumulator = 0, the control does not call the module. The CMT command interrupts a logic sequence.

Example:

See CMF command.

CALL MODULE IF LOGIC ACCUMULATOR = 0 (CMF)

Syntax: CMF (CALL MODULE IF FALSE)

Operands: Label (LBL)

Action:

The CMF command is a conditional module call. If the logic accumulator = 0, the control calls the module that begins at the label you have entered. If the logic accumulator = 1, the control does not call the module. The CMF command interrupts a logic sequence.

Example:

This example also applies to the CM and CMT commands.

Depending on the state of the input I5, call the Module 10. Initial state:
Input I5 = 0 -

Function	STL	Accu. content	Operand content
Load the operand content into the logic accumulator.	L I5	0	0
If logic accumulator =0, jump to LBL 10.	CMF 10	0	
Resume main program after module execution.	L M100	1	1
	⋮		
End of the main program.	EM		
Label: Beginning of module.	LBL 10		
Statement in the module.	L I3	0	0
Statement in the module.	O M500	1	1
Statement in the module.	= O20	1	1
End of module, resume the main program with the command L M100.	EM		

END OF MODULE, END OF PROGRAM (EM)

Syntax: EM (END OF MODULE)

Operands: None

Action:

You must end each program or subroutine (module) with the command EM. An EM command at the end or within a module causes a return jump to the module call (CM, CMT, CMF). The control then resumes the program with the statement that follows the module call. The control interprets the EM command as program end. The control can reach the subsequent program instructions only through a jump command.

END OF MODULE IF LOGIC ACCUMULATOR = 1 (EMT)

Syntax: EMT (END OF MODULE IF TRUE)

Operands: None

Action:

An EMT command causes a return jump to the module call (CM, CMT, CMF) only if the logic accumulator = 1.

END OF MODULE IF LOGIC ACCUMULATOR = 0 (EMF)

Syntax: EMF (END OF MODULE IF FALSE)

Operands: None

Action:

An EMF command causes a return jump to the module call (CM, CMT, CMF) only if the logic accumulator = 0.

LABEL (LBL)

Syntax: LBL (LABEL)

Operands: ASCII name; maximum length: 32 characters

Action:

The label defines a program location as an entry point for the JP and CM commands. You can define up to 1000 jump labels per file. The ASCII name of the label may be up to 32 characters long. However, the control evaluates only the first 16 characters.

For importing global labels, see “**EXTERN Statement (EXTERN)**.”

INDEX Register (X Register)

You can use the index register for:

Data transfer

Buffering results

Indexed addressing of operands

The index register is 32 bits wide.

You can use the X register anywhere in the program. The control does not check whether the current content is valid. Exception: For indexed write-accesses, the control checks whether the permissible address range is exceeded.

Example: = B100[X]

When the permissible address range is exceeded, the control issues the flashing error message **PLC: index range incorrect.** Acknowledge the error message by pressing the END key. After restarting the control, you must not acknowledge the **POWER INTERRUPTION** message. Switch to the PLC editor, where you will be shown the error line.

Note: At the beginning of the PLC cycle, the control sets the index register to 0. Assign the index register a defined value before using it in your program.

The following addresses are valid:

- Mn[X]
- In[X]
- On[X]
- Cn[X]
- Tn[X] Operand number = n+X
- Bn[X] Operand number = n+X
- Wn[X] Operand number = n+2*X
- Dn[X] Operand number = n+4*X
- BTX Content of index register = operand
- BCX Content of index register = operand
- BSX Content of index register = operand
- Sn[X] String number = n+X
- S#Dn[X] Dialog text number = n+X
- S#En[X] Error text number = n+X
- S#An[X] ASCII code +X
- Sn^X Substring from X-th character of the n-th string

The types “S”, K, and K$ cannot be indexed.

Note: If you address S#Dn[X] or S#En[X], the control loads the sequence <SUB>Dnnn or <SUB>Ennn in the string accumulator, where nnn is the modified string number.

The following topic is described:

- **Commands for Operating the Index Register**

Commands for Operating the Index Register

The following commands are available for exchanging data between the word accumulator and index register, or between the stack and index register:

- LX (Load index to accu)Index register – word accumulator
- =X (Store accumulator to index)Word accumulator – index register
- PSX (Push index register)Index register – stack
- PLX (Pull index register)Stack – index register
- INCX (Increment index register)
- DECX (Decrement index register)

Commands for String Processing

String processing enables you use the PLC program to generate and manipulate any texts. With Module 9082 you can display these texts in the PLC window of the screen and delete them again with Module 9080. A string accumulator as well as 100 string memories (S0 to S99), in each of which you can save a maximum of 128 characters, are provided in the control for string processing:

String accumulator (characters)	
1	128
x x x x x x x x x x x x x x ..... x x x x x x x x x x x x x x	

String memory (characters)		
	1	128
S0	x x x x x x x x x x x x x x	x x x x x x x x x x x x x x
...	x x x x x x x x x x x x x x	x x x x x x x x x x x x x x

Example

String accumulator (characters)	
1	128
K U E H L M I T T E L E I N	

String accumulator and string memory are volatile, which means that they are erased by the control when power is switched off. The operand “S” is available for string processing. You can use the operand “S” with different arguments.

Operand Declaration

The “S” operands are to be used only with string processing. You can target the following addresses with the various arguments:

- Address the string memory: Enter the number of the desired memory (S0 to S99) after the operand designation.
- Address part of a string: Use the address Sn^X (see “**INDEX Register (X Register)**”. The control addresses the substring beginning with the X-th character of the specified string.
- Immediate string: You can also enter a string directly in the PLC program. The text string, which may contain a maximum of 128 characters, must be indicated by quotation marks.
  Example: “Coolant 1 on”
- Texts from the PLC error message file or from the PLC dialog file: By specifying the line number you can read texts from the active error message file or dialog file: **PLC-ERROR: S#Exx xx:** Line number from the PLC error message file (0 to 999)
  PLC-DIALOG: S#Dxx xx: Line number from the PLC dialog file (0 to 999). Enter the string #Exx or #Dxx in the argument <arg> of the string command. The control then saves a 5-byte-long string <SUB> E0xx or <SUB> D0xx (<SUB> = ASCII <SUB>) in the accumulator. Instead of this string, the control reads the line xx of the active error message or dialog file on the screen.
- Enter an ASCII character in the string. Define the ASCII character through its code: S#Axxx

Logical Comparisons During String Processing

Use the following procedure to compare two strings, depending on the argument:

- If you compare string memories or immediate strings, the control checks both strings character by character. After the first character that does not fulfill the condition of comparison, the control resets the logic accumulator. The control does not check the remaining characters. During a comparison, the control always uses the significance of the characters from the ASCII table. This results, for example, in:
  A < B
  AA > A
- If you have entered PLC error messages or PLC dialog texts in the argument, the control compares the position in the error-message file or dialog file (0 to 999), but not the actual text as in an immediate string.

The processing times depend on the length of the strings.

The following topics are described:

- **LOAD String (L)**
- **ADD String (+)**
- **STORE a string (=)**
- **OVERWRITE a string (OVWR)**
- **EQUAL TO Command for String Processing (==)**
- **LESS THAN Command for String Processing (<)**
- **GREATER THAN Command for String Processing (>)**
- **LESS THAN OR EQUAL TO Command for String Processing (<=)**
- **GREATER THAN OR EQUAL TO Command for String Processing (>=)**
- **NOT EQUAL Command for String Processing (<>)**
- **Modules for String Processing**

LOAD String (L)

Syntax: L (LOAD)

Operands: S <arg>

Action:

Load the string accumulator. The string that the control is to load is selected through the argument <arg> after the operand designation. See also “**Operand Declaration**.”

Example:

See OVWR command.

ADD String (+)

Syntax: + (PLUS)

Operands: S <arg>

Action:

Append another string to a string in the string accumulator. The string that the control is to load is selected through the argument <arg> after the operand designation. See also "**Operand Declaration**." The resulting string must not be longer than 128 characters.

Example:

See OVWR command.

STORE a string (=)

Syntax: = (STORE)

Operands: S <arg>

Action:

Assign the content of the string accumulator to the string memory. The memory into which the control is to copy the string is selected through the argument <arg> after the operand designation. Permissible arguments:
0 to 15 (string memory S0 to S99). See also "**Operand Declaration**."

Example:

See OVWR command.

OVERWRITE a string (OVWR)

Syntax: OVWR (OVERWRITE)

Operands: S <arg>

Action:

Save the string from the string accumulator in a string memory. This command differs from the = command in that the control does not transfer the “string end” character along with it. In this way you can overwrite the beginning of a string that is already in the string memory. The memory into which the control is to copy the string is selected through the argument <arg> after the operand designation. Permissible arguments: 0 to 99 (string memory S0 to S99). See also “**Operand Declaration**.”

Example:

This example also applies to the string commands L, + and =.

Add a string from the string memory S0 to an immediate string. The result is to overwrite the contents of string memory S1. Initial state:

Immediate string = **HYDRAULICS**
String memory S0 = **OIL**
String memory S1 = **COOLANT MISSING**

String memory (characters)		
	1	128
S0	O E L	
S1	K U E H L M I T T E L F E H L T	
...	...	

Function	**STL**	**String accumulator (characters)**
		1 128
Load the immediate string into the string accumulator.	L S “HYDRAULICS”	H Y D R A U L I K
Add content of string memory S0 to string accumulator.	+ S0	H Y D R A U L I K O E L
Overwrite content of string memory S1 with content of string accumulator.	OVWR S1	H Y D R A U L I K O E L

Final status:

String memory (characters)		
	1	128
S0	O E L	
S1	H Y D R A U L I K O E L F E H L T	
...	...	

EQUAL TO Command for String Processing (==)

Syntax: == (EQUAL)

Operands: S <arg>

Action:

This command sets off a direct transition from string to logical processing. Compare the content of the string accumulator with the string in the argument. If the string accumulator and the operand are equal, the condition is true and the control sets the logic accumulator to 1. If they are not equal, the control sets the logic accumulator is set to 0.

Example:

See command <>.

LESS THAN Command for String Processing (<)

Syntax: < (LESS THAN)

Operands: S <arg>

Action:

This command sets off a direct transition from string to logical processing. Compare the content of the string accumulator with the string in the argument. If the string accumulator is less than the operand, the condition is true and the control sets the logic accumulator to 1. If the string accumulator is greater than or equal to the operand, it sets the logic accumulator to 0.

Example:

See command <>.

GREATER THAN Command for String Processing (>)

Syntax: > (GREATER THAN)

Operands: S <arg>

Action:

This command sets off a direct transition from string to logical processing. Compare the content of the string accumulator with the string in the argument. If the string accumulator is greater than the operand, the condition is true and the control sets the logic accumulator to 1. If the string accumulator is less than or equal to the operand, it sets the logic accumulator to 0.

Example:

See command <>.

LESS THAN OR EQUAL TO Command for String Processing (<=)

Syntax: <= (LESS EQUAL)

Operands: S <arg>

Action:

This command sets off a direct transition from string to logical processing. Compare the content of the string accumulator with the string in the argument. If the string accumulator is less than or equal to the operand, the condition is true and the control sets the logic accumulator to 1. If the string accumulator is greater than the operand, it sets the logic accumulator to 0.

Example:

See command <>.

GREATER THAN OR EQUAL TO Command for String Processing (>=)

Syntax: >= (GREATER EQUAL)

Operands: S <arg>

Action:

This command sets off a direct transition from string to logical processing. Compare the content of the string accumulator with the string in the argument. If the string accumulator is greater than or equal to the operand, the condition is true and the control sets the logic accumulator to 1. If the string accumulator is less than the operand, it sets the logic accumulator to 0.

Example:

See command <>.

NOT EQUAL Command for String Processing (<>)

Syntax: <> (NOT EQUAL)

Operands: S <arg>

Action:

This command sets off a direct transition from string to logical processing. Compare the content of the string accumulator with the string in the argument. If the string accumulator is not equal to the operand, the condition is true and the control sets the logic accumulator to 1. If the string accumulator is equal to the operand, it sets the logic accumulator to 0.

Example:

This example of string processing also applies to the commands =, <, >, <=, >=, <>.

Compare the immediate string with the content of the string memory S0. Depending on the result, call Module 50.
Initial state:
String memory S0 = SPINDLE 2
Immediate string = SPINDLE 1

String memory (characters)		
	1	128
S0	S P I N D E L 2	
...	...	

Function	STL	String accu. (characters), or logic accu.
		1 128
Load the string into the string accu.	L S “SPINDLE 1”	S P I N D E L 1
Gate the content of string memory S0 with content of string accumulator (=, <, >, >=, ...)	<>S0	S P I N D E L 2
If the condition is fulfilled, set logic accumulator to 1 and call the module.	CMT 50	Logic accumulator = 1

Modules for String Processing

The following topics are described:

- **Module 9070 Copy a number from a string**
- **Module 9071 Find the string length**
- **Module 9072 Copy a byte block into a string**
- **Module 9073 Copy a string into a byte block**

Module 9070 Copy a number from a string

The control searches a selectable string memory (S0 to S99) for a numerical value. When the numerical value is first found, the control copies it as a string into another selectable string memory. The control does not check whether a conflict arises between the source and target string. It may overwrite the source string (even then, however, the function of the module is ensured). The control recognizes unsigned and signed numbers, with and without decimal places. Both a period or comma are permitted as decimal point. The control returns the position (in characters) of the first character after the found number in the string memory to be searched.

Call:

PS	K/B/W/D	<>Address of the string memory to be searched>
PS	K/B/W/D	<>Address of the string memory for the found number>
CM	9070	
PL	B/W/D	<>Offset end of numerical string in the searched string memory>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number was copied
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Incorrect address of the source or target string
	11	No number, no string end, or number string has a length of more than 79 characters

Example

```
L S"X-POS.:123"
= S0
PS K+0
PS K+1
CM 9070
PL W520
```

String memory (characters)						Data stack [bits]
	1	...	10	...	128	
S0	X - P O S . : 1 2 3					
S1	1 2 3					10
...	...					

Module 9071 Find the string length

The control finds the length of the string in a selectable string memory (S0 to S99).

Call:

PS	K/B/W/D/S	<>String number or string>
CM	9071	
PL	B/W/D	<>Length of the string>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	String length found
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid immediate strings, address of the source or target string is outside the valid range (S0 to S99), string memory was searched without finding a string end

Module 9072 Copy a byte block into a string

The control copies a byte block from a PLC word memory into a PLC string (S0 to S99). The control does not check whether the byte block consists of valid ASCII characters. The module always copies the entire programmed length of the byte block, regardless of any string-end code (0x00) in the byte block. The control automatically sets a string end code (0x00) after the last copied byte. If there are any ASCII special characters in the copied byte block, the contents of the string may not be displayed in the PLC diagnosis correctly.

Call:

PS	K/B/W/D	<>Start address of byte block>
PS	K/B/W/D	<>Length of byte block>
PS	K/B/W/D	<>PLC string>
CM	9072	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Byte block copied into PLC string (S0 to S99)
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Invalid start address of the programmed byte block
	2	Invalid length of the programmed byte block (max. 127 characters)
	4	Invalid sum of the start address and the programmed block length
	11	Invalid target string

Module 9073 Copy a string into a byte block

The control copies a PLC string into the word range of the PLC. The control does not check whether the string consists of valid ASCII characters. The programmed length of the string is always copied, regardless of any end-of-string identifiers (0x00). If there are any ASCII special characters or an end-of-string identifier (0x00) in the copied string, the contents of the string will not be displayed in the PLC diagnostics correctly.

Call:

PS	K/B/W/D	<>Target address of byte block>
PS	K/B/W/D	<>Length of byte block>
PS	K/B/W/D	<>PLC string>
CM	9073	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	PLC string copied into byte block
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Invalid target address of the programmed byte block
	2	Invalid length of the programmed byte block (up to 127 characters permitted)
	4	Invalid sum of the target address and the programmed block length
	11	Invalid source string

Submit Programs

Submit programs are subroutines that the PLC submits to the NC for processing. This allows you to solve problems that are very processor-intensive, require program loops, or must wait for external results. It is a prerequisite that these programs are not restricted to a definite time frame. Depending on the processor load, the control allocates a certain processing power to a submit program. You start submit programs from the PLC program. They can access all data memory areas (M/B/W/D) as the main program can. To avoid problems, ensure that data processed by the PLC program is clearly separated from data processed by the submit program. You can place up to eight submit programs in a queue (submit queue). Each submit program receives an “identifier” (a number between 1 and 255, assigned by the NC), which the control enters in the word accumulator. With this identifier and the REPLY function you can then interrogate whether the program is in the queue, is being processed, or has already been processed.
The control processes the submit programs in the sequence in which they were entered in the queue. If errors occur during execution of the submit program, the NC sets the following markers:

		Type
4200	**Overflow during multiplication**	M
4201	**Division by 0**	M
4202	**Incorrectly executed modulo**	M
4203	**Error status for PLC module**	M
4204	**Reserved for errors that the PLC programmer would like to catch**	M

The control lists these markers separately in the submit job. This means that the same markers can be edited simultaneously in the PLC run program without changing the original markers. No exact times can be stated for the commands for managing the submit queue.

The following topics are described:

- **Calling the Submit Program (SUBM)**
- **Interrogating the Status of a Submit Program (RPLY)**
- **Canceling a Submit Program (CAN)**

Calling the Submit Program (SUBM)

Syntax: SUBM (SUBMIT)

Operands: Label (LBL)

Action:

Assign an identifier (1 to 255) to a labeled subroutine and put it in the queue. At the same time, the control writes the assigned number in the word accumulator. If programs are already entered in the submit queue, the control does not run the addressed program until the programs before it are finished. A submission to the queue may only take place from a PLC program. A SUBM command in a submit program is not possible.

If there is no room in the queue, or if you program the SUBM command in a submit program (nesting), the control assigns the value “0” to the word accumulator.

Example:

See CAN command.

Interrogating the Status of a Submit Program (RPLY)

Syntax: RPLY (REPLY)

Operands: B/W

Action:

Interrogate the status of the submit program with the specified identifier. You must have already stored the identifier in a byte or word when you call the submit program. With the RPLY command and the defined memory address (byte or word containing the identifier) the control transfers one of the following processing states to the word accumulator:

- Word accumulator = 0: Program complete/not in the queue
- Word accumulator = 1: Program running
- Word accumulator = 2: Program in the queue

Example:

See CAN command.

Canceling a Submit Program (CAN)

Syntax: CAN (CANCEL)

Operands: B/W

Action:

Cancel a submit program with the specified identifier during processing, or remove it from the queue. You must have already stored the identifier in a byte or word when you call the submit program. After you have canceled the program, the control immediately starts the next submit program from the queue. The following PLC modules cannot be canceled at just any location with CANCEL:

- PLC module for access to screen (908X)
- PLC module for reading NC files (909X)

For these modules, you must check with the RPLY command whether the CAN command may be executed.

Example:

This example also applies to the SUBM and RPLY commands.

Depending on input I10, submit the subroutine with the label LBL 300 to the NC for processing. In addition, check the processing status of the subroutine in the main program with the RPLY command, and cancel it with the CAN command, depending on input I11.

Function	STL
Load the state of input I10 into the logic accu.	L I10
If logic accumulator =0, jump to LBL 100.	JPF 100
Interrogate the status of the submit program and load it into the word accumulator.	RPLY B128
If the word accumulator is not equal to 0 (i.e., the submit program has already been transferred to the NC for processing, set the logic accumulator to 1).	<> K+0
If logic accumulator =1, jump to LBL 100.	JPT 100
Call submit program 300.	SUBM 300
Save the identifier of the submit program in byte 128.	= B128
Label	LBL 100
Load the state of input I11 into the logic accu.	L I11
If logic accumulator =0, jump to LBL 110 (skip the program cancellation).	JPF 110
Cancel the submit program.	CAN B128
Label	LBL 110
	⋮
End of the main program.	EM
Label: Beginning of the submit program	LBL 300
	⋮
End of the submit program	EM

Always insert submit programs, like any module, at the end of the main program. In this case, the content of the submit program could be a display in the PLC window that is realizable through permanently assigned PLC modules.

Cooperative Multitasking

You can run several processes in the PLC with cooperative multitasking. Unlike genuine multitasking, with cooperative multitasking information and tasks are exchanged only at places that you define. Cooperative multitasking permits up to eight parallel PLC processes and the submit queue. In a program that you have started with SUBM, you can use commands for changing tasks and controlling events (Module 926x). You should additionally insert a task change between the individual jobs in the submit queue, so that the control can execute parallel processes by the end of a job at the latest. The cyclic PLC main program does not participate in cooperative multitasking, but interrupts a submit job and the parallel processes at whatever their current stage is.

The following topics are described:

- **Starting a Parallel Process (SPAWN)**
- **Control of Events**

Starting a Parallel Process (SPAWN)

Syntax: SPAWN <label>

Operands: D

Action:

In the specified double word, the control returns the identifier, see “**Submit Programs**.” The control returns –1 if no process could be started. You can call the spawn command only in a submit job or in another spawn process (maximum of eight parallel processes are permitted). If such a process ends with EM, the control removes it from the memory, and the memory space is again available.

Control of Events

The parallel processes can make events available to one another. This saves processing time otherwise spent in the constant interrogating of operating states by the individual processes. A special feature of event control is the waiting period, during which the process can “sleep” for a programmed time. With this function you can repeat program sections in a slow time grid, for example for display or monitoring functions.

The following topics are described:

- **Module 9260 Receiving events and waiting for events**
- **Module 9261 Sending events**
- **Module 9262 Context change between spawn processes**
- **Module 9263 Interrupting a spawn process for a defined time**
- **Module 9264 Wait for a condition**

Module 9260 Receiving events and waiting for events

Call the module only in a submit job or spawn job. The module enables a spawn job or submit job to interrogate or wait for the occurrence of one or more events. At the same time, the module triggers a change in context.

If you transfer the value zero for the event mask, the control returns all set events without deleting them. Otherwise, in a call with a waiting period, the control returns all the requested events and deletes them. For a call without a waiting period, the control returns and deletes the events only if the condition is met.

If the events are OR-gated, the control returns and deletes only the set events. You can specify the events to be deleted by calling without a waiting period and with an OR gate.

Event bits 16 to 31 are reserved for the operating system:

- Bit 16: BREAK, cancels a function. Setting and reading is permitted. If you transmit this event, the control cancels access to interfaces and the network!
- Bit 17: Reserved, do not use
- Bit 18: Reserved, do not use
- Bit 19: QUIT, acknowledgment of a request. Use this bit only in the immediate context of a request.
- Bit 20 to bit 31: Reserved, do not use

Call:

PS	B/W/D/K	<>Wait> 0 = Do not wait –1 = Wait
PS	B/W/D/K	<>AND/OR> 0 = OR-gated; otherwise, AND-gated
PS	B/W/D/K	<>Event mask> 0 = Available events
CM	9260	
PS	B/W/D/K	<>Events> Read events

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Event has been read
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Incorrect transfer value for <Wait> parameter
	20	Module was not called in a spawn job or submit job

Module 9261 Sending events

With this module you can send events to a spawn or submit job and then interrogate them with Module 9260. You can call the module in the cyclic program section, in submit jobs and in spawn processes. The control addresses the receiver through the identifier that the spawn command has returned. The submit queue is addressed through the identifier $80000000 (not through the identifier returned by the SUBM command!). The control always assigns the events that you send to the submit queue to the job that is running at the time of arrival. If they are not read by this job, they remain for the next one. If you wish the receiver process to start immediately, after Module 9261 you must also call Module 9262 to enable a change of context.

Event bits 16 to 31 are reserved for the operating system (see **Module 9260**).

Call:

PS	D/K	<>Identifier> Identifier from the spawn command of the receiver K$80000000 = submit queue
PS	B/W/D/K	<>Events> Events to be triggered, bit encoded
CM	9261	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Event has been sent
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	30	Incorrect identifier

Module 9262 Context change between spawn processes

You can call Module 9262 only in a submit job or spawn job. The module switches the context to another PLC process or submit queue if such a process exists and is not waiting for an event or for the expiration of a dwell time.

Call:

CM 9262

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Context was changed
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	30	Module was not called in a spawn job or submit job

Module 9263 Interrupting a spawn process for a defined time

You can call Module 9263 only in a submit job or spawn job. The module interrupts the calling process for at least the specified time. If other processes or the submit queue are ready to run, the control changes the context to one of these processes. The waiting period is interpreted as an unsigned number, so that negative values result in very long waiting periods.

Call:

PS B/W/D/K <>Waiting period in ms>

CM 9263

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Delay is active
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	30	Module was not called in a spawn job or submit job

Module 9264 Wait for a condition

You can call Module 9264 only in a submit job or spawn job. The module enables a spawn process or submit job to wait for specific condition.

If at least one bit in the event mask is set, the process is continued immediately, and the event signalizes that the condition has been fulfilled. Module 9260 must wait for this. If the event mask equals zero, the process is paused until the condition is fulfilled. Only then is the module call ended.

Call:

PS	B/W/D/K/S	<>Condition> (e.g., “ML_TestMemory[0] = 1”) Syntax corresponds to the NC syntax from **FN20: WAIT FOR:** see the *6000i CNC User’s Manual*, P/N 627785-2X. Following conditions are permissible: =: Equal < : Less than > : Greater than <= : Less than or equal to >= : Greater than or equal
PS	B/W/D/K	<>Event mask> 0 : Process is paused until condition is fulfilled
CM	9264	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Condition is waited for
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	30	Module was not called in a spawn job or submit job

Constants Field (KF)

You can use the constants-field data type to access one of several constants, defined in tabular form, depending on the value of the index register X. You address it with KF <Name>[X], where <Name> is a label indicating the beginning of the constants field. Constants fields must be introduced with the label KFIELD <Name>. This is followed by any quantity (other than zero) of constants and the end label ENDK. Constants fields can only be programmed where the program has previously been concluded with an EM or JP statement. The name of constants fields corresponds to the rules for naming labels.

The following topic is described:

- **Addressing**

Addressing

Types of addresses:

- L KF <Name> [X], with X Š 0:
  The control transfers the value of the constant defined by X in the constants field <Name>.
- L KF <Name> [X], with X = –1:
  The control transfers the length of the constants field <Name>.
- L KF <Name>:
  The control transfers the absolute address of the constants field <Name>. This is only worthwhile in conjunction with modules (e.g., Module 9200). You can also use this addressing in a constants field.

Example:

Function	STL
Access value field with X = [0 to 3].	L KF VAL_FIELD [X]
Assign one of the constants to word W0.	= W0
End of the main program.	EM
Define the constants field. Constant to be loaded with X = 0 Constant to be loaded with X = 3 End of the constants field.	KFIELD VAL_FIELD K+10 K+1 K$ABC K–100000 ENDK

The control checks the access to constants fields in the same way as the write access for indexed operands. X can assume only positive values from 0 to <Length of constants field –1>.

Program Structures

To design an easily understandable program, divide it into program sequences. Use labels (LBL) as well as conditional and unconditional jumps. If you use structured statements, the compiler creates the labels and jump commands. Remember that using these labels and jump commands reduces the number of available labels accordingly. You can nest structured instructions in up to 16 levels. It is not possible to share levels.

Example:

Correct program structure	Incorrect program structure
IFT	**IFT**
⋮	⋮
WHILEF	**WHILEF**
⋮	⋮
ENDW	**ENDI**
⋮	⋮
ENDI	**ENDW**

The instructions IFT, IFF, WHILET, WHILEF, ENDW, UNTILT, and UNTILF require a valid gating result in the logic accumulator. You conclude the sequence of gating operations. The instructions ELSE, ENDI, and REPEAT require that all previous operations sequences have been concluded.

The following topics are described:

- **IF ... ELSE ... ENDI Structure**
- **REPEAT ... UNTIL Structure**
- **WHILE ... ENDW Structure**
- **Case Branch**

IF ... ELSE ... ENDI Structure

The IF ... ELSE ... ENDI structure permits the alternative processing of two program branches depending on the value in the logic accumulator. The ELSE branch is not mandatory. The following commands are available:

- IFT (IF LOGIC ACCU TRUE):
  Following code only if logic accumulator = 1
- IFF (IF LOGIC ACCU FALSE):
  Following code only if logic accumulator = 0
- ELSE (ELSE):
  Following code only if IF is not fulfilled
- ENDI (END OF IF STRUCTURE):
  End of the IF structure

Function	STL
Load input I0 into the logic accumulator	L I0
Run the following code if logic accumulator = 1	IFT
Program code for I0 = 1	⋮
Run the following code if logic accumulator = 0; command can be omitted	ELSE
Program code for I0 = 0, can be omitted	⋮
End of the conditional processing	ENDI

REPEAT ... UNTIL Structure

The REPEAT ... UNTIL structure repeats a program sequence until a condition is fulfilled. Under no circumstances can you wait with this structure in the cyclic PLC program for the occurrence of an external event! The following commands are available:

- REPEAT (REPEAT):
  Repeat the program sequence from here.
- UNTILT (UNTIL TRUE):
  Repeat the sequence until the logic accumulator = 1.
- UNTILF (UNTIL FALSE):
  Repeat the sequence until the logic accumulator = 0.

The control runs a REPEAT ... UNTIL loop at least once!

Function	STL
Assign the content of the logic accumulator to marker 100; conclusion of the previous commands	= M100
Repeat the following program code	REPEAT
Program code to be run	⋮
Load the index register	L X
Check the index register	>= K100
Repeat until X >= 100	UNTILT

WHILE ... ENDW Structure

The WHILE ... ENDW structure repeats a program sequence if a condition is fulfilled. Under no circumstances can you wait with this structure in the cyclic PLC program for the occurrence of an external event! The following commands are available:

- WHILET (WHILE TRUE):
  Run the sequence if logic accumulator = 1.
- WHILEF (WHILE FALSE):
  Run the sequence if logic accumulator = 0.
- ENDW (END WHILE):
  End of the program sequence, return to the beginning

The control runs a WHILE ... ENDW loop only if at the beginning the WHILE condition is fulfilled. Before the ENDW instruction you must reproduce the condition for execution. For the WHILE ... ENDW structure the control generates two internal labels. The condition can also be produced in a way different from before the WHILE instruction!

Function	STL
	⋮
Load marker 100 into the logic accumulator; create condition for 1st WHILE scan.	L M100
Run the following code if logic accumulator = 1	WHILET
Program code for logic accumulator = 1	⋮
Produce the condition of repeated execution: Load marker 101 in the logic accumulator and gate the content of marker M102 with AND.	L M101 A M102
Jump back to the WHILE request.	ENDW

Case Branch

The following topics are described:

- **Indexed Module Call (CASE)**
- **End of indexed module call (ENDC)**

Indexed Module Call (CASE)

Syntax: CASE (CASE OF)

Operands: B/W

Action:

Selects a certain subroutine from a list of module calls (CM). These CM commands must follow the CASE statement immediately and are numbered internally in ascending order from 0 to a maximum of 127. The content of the operand (B, W) addresses the desired module. Subsequent entries in the jump table (CM) must have addresses at least four bytes higher than the previous entry.

Example:

See ENDC command.

End of indexed module call (ENDC)

Syntax: ENDC (ENDCASE)

Operands: None

Action:

You use the ENDC command in connection with the CASE command. It must come immediately after the list of CM commands.

Example:

Function	STL
Case command and operand; the internal address of the desired module must be saved in the operand	CASE B150
Call module if operand content = 0 Internal addressing from 0 to max. 127	CM 100
Call module if operand content = 1	CM 200
Call module if operand content = 2	CM 201
Call module if operand content = 3	CM 202
Call module if operand content = 4	CM 203
Call module if operand content = 5	CM 204
Call module if operand content = 6	CM 300
End of the CASE statement	ENDC

Linking Files

You can store the source code of the PLC program in several files. To manage these files, use the following commands:

USES

GLOBAL

EXTERN

These instructions must be located at the beginning of your PLC program (i.e., before the first PLC command). With the USES instruction you link another file into the program. The GLOBAL instruction supplies a label from its own file for an entry that can be used by all other files. The EXTERN instruction provides a label that is defined in another file and is identified there with GLOBAL. You can then call this label from the active file. You can dramatically improve the transparency of your program by dividing your source code by function into individual groups and then save these groups in individual files. The number of labels is not limited. You can link up to 256 files to one program. The total size is only limited by the available memory. If the memory is exceeded the error message **System memory overflow** is displayed. If you use more than one file, the main program must have the status flag “M” in the directory. This is done in the RAM by using the PLC program function “COMPILE” once and selecting the main program in the file window.

The following topics are described:

- **USES Statement (USES)**
- **GLOBAL Statement (GLOBAL)**
- **EXTERN Statement (EXTERN)**

USES Statement (USES)

Syntax: USES <file name>

Operands: None

Action:

You can use the USES statement in the main program to link other files. Files that are linked with USES can themselves also use the statement to link further files. It is also permissible to use the USES statement to link a single file to several other files. The code for this file is generated only once. The USES statement requires a file name as an argument. The USES statement only links a file; it does not run the file's program code. The USES statement cannot be compared with a CM statement. The linked files must therefore contain individual modules that you can then call with the CM statement.

Example:

```
USES PLCMOD1
USES EPRUPG
USES RAMPLC
```

Linking of files:

Function	STL
Main program	PLCMAIN.PLC
Link the file for spindle control.	USES SPINDLE.PLC
Link the file for tool change.	USES TCHANGE.PLC
Program code	⋮

Function	STL
File for spindle control	SPINDLE.PLC
Integrate file with general subroutines.	USES PLCUPG.PLC
Program code	⋮

Function	STL
File for tool change	TCHANGE.PLC
Integrate file with general subroutines.	USES PLCUPG.PLC
Program code	⋮

Function	STL
File with general subroutines	PLCUPG.PLC
Program code	⋮

GLOBAL Statement (GLOBAL)

Syntax: GLOBAL <Label, declaration beyond the file boundary>

Operands: None

Action:

There is no limit to the number of labels in each file linked with USES. To enable a module that was defined in a file to be called from another file, you must declare the module to be global. This is done by entering the GLOBAL statement at the beginning of the file. You can set labels globally only if they are defined with LBL (and not with KFIELD!) later on in the program.

The main program must not contain any GLOBAL definitions. A single label cannot be declared global by more than one module. However, a name that is declared global in file A can be used again locally in file B. The number of labels is not limited.

EXTERN Statement (EXTERN)

Syntax: EXTERN <Label, a module from another file can now be called with a CM command>

Operands: None

Action:

To enable a label in one file to access modules that other files have declared as GLOBAL, you must declare the label with EXTERN. You must write the EXTERN statement at the beginning of the file. In the program code, you can then jump to this label with the commands CM, CMT, and CMF.

The following functions are not permitted with external labels:

- JP, JPF, JPT
- Access to a constants field
- Linking a CM statement in a CASE branch

The name of the external label cannot be used again as a local label in the same file.

PLC Modules

A number of PLC modules are available for PLC functions that are very difficult or even impossible to perform with PLC commands alone. You will find descriptions of these modules under the corresponding functions. (See "**Section 4, Overview of Modules**.")

If the control processes a module incorrectly, it sets the marker NN_GenApiModuleError (M4203). You can evaluate this marker for displaying an error message.

The following topics are described:

- **Markers, Bytes, Words, and Double Words**
- **Number Conversion**

Markers, Bytes, Words, and Double Words

- **Module 9000/9001 Copy in the marker or word range**
- **Module 9010/9011/9012 Read in the word range**
- **Module 9020/9021/9022 Write in the word range**
- **Module 9025 Writing a value as BCD code to eight successive markers**
- **Module 9040 Reading of axis coordinates by the PLC in the format 1/1000 (0.001) mm**
- **Module 9041 Reading of axis coordinates by the PLC in the format 1/10000 (0.0001) mm**
- **Module 9042 Reading of spindle coordinates by the PLC in the format 1/1000 (0.001) degrees**
- **Module 9044 Reading of spindle coordinates by the PLC in the format 1/10000 (0.0001) degrees**
- **Module 9111 Receive a message via LSV2**
- **Module 9140 Set axis-specific feed-rate limit**
- **Module 9141 Read axis-specific feed-rate (status)**
- **Module 9147 Assigning a reference value to an axis**
- **Module 9171 Start of a spindle orientation with adjustable parameters**
- **Module 9248 Copy, rename, or delete file**
- **Module 9270 Read OEM-define string value**
- **Module 9271 Write OEM-define string value**
- **Module 9277 Writing data into the OEM log**
- **Module 9291 Starting an NC macro**
- **Modules 9300 to 9302, 9304 to 9306 for Managing Multiple Tool Magazines**
- **Module 9300 Locking and releasing the pocket table**
- **Module 9301 Find the number of an entry in the pocket table**
- **Module 9302 Search for a vacant pocket in the tool magazine**
- **Module 9304 Copying OEM values from the pocket table**
- **Module 9305 Moving tools in the pocket table**
- **Module 9306 Moving tools between magazines**
- **Module 9322 Information of the current NC program**
- **Module 9340 Searching for a pocket depending on magazine rules**
- **Module 9341 Editing a pocket table depending on magazine rules**
- **Module 9342 Find magazine and pocket number**

- **Module 9343 Compilation and activation of magazine rules**
- **Module 9350 Read data from the tool table**
- **Module 9351 Write data to tool table**
- **Module 9407 Give default tool number for an NC channel**
- **Module 9411 Read the actual spindle values (speed, coordinates)**
- **Module 9416 Select gear range and assigned settings for spindle**
- **Module 9417 Set default shaft speed for spindle**
- **Module 9418 Set status for spindle**

Module 9000/9001 Copy in the marker or word range

Modules 9000 (markers) and 9001 (byte/word/double) copy a block with a certain number of markers or bytes, beginning with the start address, to the specified target address. For module 9001 the length should always be defined in bytes.

The control copies sequentially, beginning with the first memory cell. Therefore, the function is not ensured if the source block and the target block overlap and the source block begins at a lower address than the target block. In this case the control overwrites the overlapping part of the source block before the copying process.

Call:

PS	B/W/D/K	<>Number of the 1st marker in source block>
PS	B/W/D/K	<>Number of the 1st marker in target block>
PS	B/W/D/K	<>Length of block in markers>
CM	9000	

PS	B/W/D/K	<>Number of the 1st word in source block>
PS	B/W/D/K	<>Number of the 1st word in target block>
PS	B/W/D/K	<>Length of block in markers>
CM	9001	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Markers, bytes, words, or double words were copied
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Operand address invalid
	2	Address too high or block too long
	4	Programmed source or target block too long

Module 9010/9011/9012 Read in the word range

From the specified location in the word memory the control reads a byte, word, or double word and returns it as an output quantity to the stack. Indexed reading is possible by specifying a variable as designation of the memory location.

Call:

```
PS      B/W/D/K   <>Address of the byte to be read>
CM      9010      ; READ BYTE
PL      B         <>Target address for byte that was read>

PS      B/W/D/K   <>Address of the word to be read>
CM      9011      ; READ WORD
PL      B         <>Target address for word that was read>

PS      B/W/D/K   <>Address of the double word to be read>
CM      9012      ; READ DOUBLE WORD
PL      B         <>Target address for double word that was read>
```

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Byte was read
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	3	Invalid address was programmed
	5	Module 9011: Specified address is not a word address Module 9012: Specified address is not a double word address

Example of Module 9010

Initial state:

```
Byte    B10   = 35 (address)
Byte    B35   = 80 (byte to be read)
Byte    B100  = ?
```

Function	STL	Accumulator content (dec)	Data stack (dec)
Save the address (B10) of the byte to be read from the word accumulator in the data stack.	PS B10	35	35
Read byte B35 and save in the data stack.	CM 9010		80
Save data stack in byte B100.	PL B100	80	80

Module 9020/9021/9022 Write in the word range

The control writes the specified byte, word or double word to the defined location in the word memory. Indexed writing is possible by specifying a variable as designation of the memory location.

Call:

```
PS      B/W/D/K     <>Address of the byte to be written>
PS      B/W/D/K     <>Byte to be written>
CM      9020        ; WRITE BYTE TO ADDRESS

PS      B/W/D/K     <>Address of the word to be written>
PS      B/W/D/K     <>Word to be written>
CM      9021        ; WRITE WORD TO ADDRESS

PS      B/W/D/K     <>Address of the double word to be written>
PS      B/W/D/K     <>Double word to be written>
CM      9022        ; WRITE DOUBLE WORD TO ADDRESS
```

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Byte was written
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	3	Invalid address was programmed
	5	Module 9021: Specified address is not a word address Module 9022: Specified address is not a double word address

Example of Module 9020

Initial state:

```
Byte    B10     = 35 (address)
Byte    B100    = 120 (byte to be written)
Byte    B35     = ?
```

Function	STL	Accumulator content (dec)	Data stack (dec)
Save the address (B10) of the byte to be written from the word accumulator in the data stack.	PS B10	35	35
Save byte B100 from the word accu in the data stack.	PS B100	120	120
Write data stack to byte B35.	CM 9020	120	

Module 9025 Writing a value as BCD code to eight successive markers

The transferred value (between 0 and 99) is shown as BCD code with 2 nibbles of four bits each. These bits are copied to the eight successive logical markers indicated. This module makes it possible to output BCD codes for M, S, or T words in a simple manner.

The task can be influenced with the transferred, bit-coded mode:

- Bit 0 = 0: The transferred address is a logical marker M
- Bit 0 = 1: The transferred address is a logical output marker O
- Bit 1 = 0: The highest-value bit is output to the first address
- Bit 1 = 1: The lowest-value bit is output to the first address

Call:

PS	B/W/D/K	<Number of the first marker to be written>
PS	B/W/D/K	<Value to be written>
PS	B/W/D/K	<Bit-coded mode for output>
CM	9025	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error. Marker was written.
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	1	The given value is outside the valid range (0–99).
	3	The given address is outside the valid range.
	4	The sum of the given address and 8 is outside the valid range.

Module 9040 Reading of axis coordinates by the PLC in the format 1/1000 (0.001) mm

Module 9040 loads the axis coordinates from the control loop for all NC axes. The actual values in the reference system, the servo lag, the distance-to-go, and the deflection of a triggering touch probe can be loaded.

The values are saved in 10 double words in the format 1/1000 mm, beginning at the given target address.

The module is only supported if you use the 6000i compatible programming interface (API 1.0).

Note: This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls. This module is not supported if the symbolic programming interface is used. Use Module 9041 instead.

Possible errors:

- The argument for the type of coordinate is outside the permitted range (2).
- The specified target address is not a double word address (4).
- The double word block cannot be written to the specified target address (4)
- You are using the symbolic programming interface.

Call:

PS	K/B/W/D	<Target address Dxxxx>
PS	K/B/W/D	<Type of coordinate> 2: Actual values in the reference system 3: Following error 4: Distance-to-go 5: Deflection (measuring touch probe) 6: Actual values in the datum system
CM	9040	

Error recognition:

Marker	Value	Meaning
M4203	0	Data was read
	1	Faulty call data

Module 9041 Reading of axis coordinates by the PLC in the format 1/10000 (0.0001) mm

Module 9041 loads the axis coordinates from the control loop for all NC axes. The actual values in the reference system, the servo lag, the distance-to-go, and the deflection of a triggering touch probe can be loaded.

The values are saved in 10 double words in the format 1/10000 mm, beginning at the given target address.

Possible errors:

- The argument for the type of coordinate is outside the permitted range (2).
- The specified target address is not a double word address (4).
- The double word block cannot be written to the specified target address (4)

Call:

PS	K/B/W/D	<Target address Dxxxx>
PS	K/B/W/D	<Type of coordinate> 2: Actual values in the reference system 3: Following error 4: Distance-to-go 5: Deflection (measuring touch probe) 6: Actual values in the datum system
CM	9041	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data was read
	1	Faulty call data

Module 9042 Reading of spindle coordinates by the PLC in the format 1/1000 (0.001) degrees

The coordinate values for actual value, nominal value, actual value in the reference system, servo lag, and spindle distance-to-go are stored in five consecutive double words beginning from the given target address. The actual value, nominal value, and reference value are standardized to 0–+360.000 degrees. The values for servo lag and distance-to-go are shown between –2879.912 degrees and +2879.912 degrees. They are shown at a resolution of 1/1000 (0.001) degrees.

The module is only supported if you use the 6000i compatible programming interface (API 1.0).

Note: This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls. This module is **not** supported if the symbolic programming interface is used. Use Module 9411 instead.

During operation as an analog spindle (M3/M4 active or M5 and spindle not in closed-loop control), the nominal value is considered to be the actual value. The servo lag and distance-to-go are considered to be zero.

Possible errors:

- The specified target address is not a double word address (not divisible by 4).
- Five double words cannot be written to the specified target address (target address is too high).

Call:

PS	B/W/D/K	<Target address Cxxxx>
CM	9042	

Error recognition:

Marker	Value	Meaning
M4203	0	Spindle data was read
	1	Faulty call data

Module 9044 Reading of spindle coordinates by the PLC in the format 1/10000 (0.0001) degrees

The coordinate values for actual value, nominal value, actual value in the reference system, servo lag, and spindle distance-to-go are stored in five consecutive double words beginning from the given target address. The actual value, nominal value, and reference value are standardized to 0–+360.000 degrees. The values for servo lag and distance-to-go are shown between –2879.912 degrees and +2879.912 degrees. They are shown at a resolution of 1/1000 (0.0001) degrees.

The module is only supported if you use the 6000i compatible programming interface (API 1.0).

Note: This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls. This module is **not** supported if the symbolic programming interface is used. Use Module 9411 instead.

During operation as an analog spindle (M3/M4 active or M5 and spindle not in closed-loop control), the nominal value is considered to be the actual value. The servo lag and distance-to-go are considered to be zero.

Possible errors:

- The specified target address is not a double word address (not divisible by 4).
- Five double words cannot be written to the specified target address (target address is too high).

Call:

PS	B/W/D/K	<Target address Cxxxx>
CM	9044	

Error recognition:

Marker	Value	Meaning
M4203	0	Spindle data was read
	1	Faulty call data

Module 9111 Receive a message via LSV2

Module 9111 reads a message (double word or string) that has been received from a host computer connected by LSV2 protocol.

The message must be transmitted from the host by the LSV2 command: “M_PC<msg.l>”.

Call:

PS	B/W/D/K	<Data type> 0: Binary data double word 1: String
PS	B/W/D/K	<Target address> With binary: Number of the double word With string: Number of the string
CM	9111	
PL	B/W/D	<Error code> 0: Message was read 1: No connection to host 2: No message of this type in receiving buffer 3: Incorrect data type (not 0 or 1) 4: Incorrect target address

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Message was received
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Incorrect data type
	4	No double word address, or incorrect string number
	11	String too long
	13	No connection
	15	Transmit buffer not empty
	16	Receiving buffer empty

Module 9140 Set axis-specific feed-rate limit

Feed rates for multiple axes can be defined with this module. The feed rates for a certain number of axes must be stored consecutively in the double word memory.

Note: This module is **not** supported if the symbolic programming interface is used. This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls.

Constraints:

- If individual feed-rate values are invalid for a block of axes, the feed rate for these axes is set to 0 and M4203 is set. All other axes are given the prescribed value.
- The PLC module limits the feed rate to the maximum possible feed rate in the PLC.
- The resulting feed rate can also depend on other limitations (e.g., FMAX, programmed feed rate).
- The module is only supported if you use API version 1.0.

Call:

PS	B/W/D/K	<Start address of double word array> >= 0: Feed rate from corresponding double word –1: Maximum feed rate –2: Rapid-traverse feed rate for this axis –3: Manual feed rate for this axis
PS	B/W/D/K	<Number of axes>
CM	9140	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError M4203	0	Function was performed
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode W1022	1	Invalid feed-rate value (< –3)
	2	Invalid number of axes
	3	Invalid block length as of starting address
	5	Not a double word address (not divisible by 4)
	24	Module was called from a spawn/submit job

Module 9141 Read axis-specific feed-rate (status)

Module 9141 is used to interrogate the feed-rate status (limited by Module 9140) of multiple axes. The status for a certain number of axes is stored consecutively in the double word memory. .

Note: This module is **not** supported if the symbolic programming interface is used. This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls.

Call:

PS	B/W/D/K	<Start address of double word array> >= 0: Feed rate from corresponding double word –1: Maximum feed rate –2: Rapid-traverse feed rate for this axis –3: Manual feed rate for this axis
PS	B/W/D/K	<Number of axes>
CM	9141	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError M4203	0	Function was performed
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode W1022	2	Invalid number of axes
	3	Invalid block length as of starting address
	5	Not a double word address (not divisible by 4)
	24	Module was called from a spawn/submit job

Module 9147 Assigning a reference value to an axis

In some cases it may be necessary to assign a new reference value to an axis (e.g., if an axis is mechanically fixed and the encoder is moved). Since due to the mechanical fixing the position of the axis cannot be changed, you can assign it a new reference value.

- Enter the new reference value in Module 9147.

If a new reference value is assigned to an axis, the corresponding bit in NN_AxReferenceAvailable (W1032) is reset. .

Note: When calling the module for an NC axis during a strobe, the synchronization with the advance calculation (strobe with **MP_sync** = SYNC_CALC) must be configured for this strobe.

Call:

PS	B/W/D/K	<Axis number> 0 to 8: Axes 1 to 9
PS	B/W/D/K	<New reference value in 0.1 µm>
CM	9147	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	2	Invalid axis number
	21	Missing strobe in M4176 = 1
	24	Module was called in a spawn job or submit job

Module 9171 Start of a spindle orientation with adjustable parameters

Module 9171 can be used to start an orientation of the spindle. The orientation speed, orientation angle, and the direction of rotation can be set. The module sets the marker M4130, which displays that the positioning is running and for how long. .

Note: This PLC module was introduced in order to remain compatible with older PLC programs (with API version 1.0) of older ANILAM contouring controls. This module is **not** supported if the symbolic programming interface is used. Use Module 9414 instead.

Constraints:

- If no speed output has occurred for the spindle, the call will have no effect.
- If the marker M4130 is set in the same PLC scan and Module 9171 is called, the spindle is oriented with the parameters from the module call.
- If the module is called several times in the same scan, the spindle will be oriented with the parameters of the last call.
- If the module is called although an orientation from an earlier PLC scan is not yet finished, the call will have no effect.
- The module functions only in the cyclic PLC program.
- If the module is called while the spindle is turning, the direction of orientation that was transferred will be ignored. The spindle is always oriented in the direction of spindle rotation.
- If the values +2 to +4 are transferred as direction of rotation, the spindle can be oriented to the angle last defined in CYCL DEF 13. The transferred angle of orientation is added to the value from CYCL DEF 13. Therefore the PLC can transfer an additional spindle preset.
- The module is only supported for PLC programs that use API version 1.0.

Call:

PS	B/W/D/K	<Orientation angle [1/10000 degrees]> or additional preset if there is a value from CYCLE DEF 13
PS	B/W/D/K	<Speed [1/1000 rpm]>
PS	B/W/D/K	<Direction of rotation> –1: Negative direction (M04) 0: Direction of the shorter path 1: Positive direction (M03) 2: Same as –1, but angle from CYCLE DEF 13 3: Same as 0, but angle from CYCLE DEF 13 4: Same as +1, but angle from CYCLE DEF 13
CM	9171	

Error recognition:

Marker	Value	Meaning
M4203	0	Spindle is oriented, M2712/M4130=1
	1	Error code in NN_GenApiModuleErrorCode
W1022	1	The value for direction of rotation or rotational angle is invalid
	2	Spindle number/rotational speed incorrect, or no speed output yet
	19	Spindle is not a closed-loop spindle
	24	Module was called from a spawn/submit job
	27	A spindle orientation is already running

Module 9248 Copy, rename, or delete file

Module 9248 is used to copy, rename, and delete files.

Constraints:

- The file names must contain drive information (e.g., PLC:) and file types.
- The file types are used to determine whether renaming is permissible.
- Dependencies with other files are not considered when deleting files.

Call:

PS	B/W/D/K/S	<Name of the source file>
PS	B/W/D/K/S	<Name of the target file> Only used with mode 0 or 1
PS	B/W/D/K	<Mode> 0: Copy 1: Rename 2: Delete
CM	9248	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	2	Module was called in an invalid mode setting
	7	Error during file conversion, invalid source or target string or error during copying without file conversion
	20	Module was not called in a submit job or spawn job
	36	Not identical file types, conversion not possible

Module 9270 Read OEM-define string value

Module 9270 is used to read the values from the CfgOemString configuration entities. The first value in the list under “value” whose attribute “key” matches the transferred token is read.

The value can also be an empty string, which is then transferred into the target string. The target string is then only changed if a matching entity was found.

Trailing blank spaces of the token are removed before comparison with the keys of the CfgOemString entities.

Call:

PS	B/W/D/K/S	<String number or string with token>
PS	B/D/K/S	<String number>
CM	9270	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data was read
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	3	Parameter invalid/invalid number of the target string
	12	Token string too long
	13	Internal error
	20	Module was not started from a spawn job or submit job
	30	Parameter does not exist

Module 9271 Write OEM-define string value

With Module 9271 you change or create in the configuration data a CfgOemString entity whose attribute “key” matches the transferred token. If the entity already exists, the first value in the list “value” is replaced by the transferred value. Otherwise, a new entity is created in the SYS.CFG file, and the first value in the list “value” is set to the transferred value. Trailing blank spaces of the token are removed before comparison with the keys of the CfgOemString entities. Also, trailing blank spaces of the value are removed before entering it in the CfgOemString entity.

Call:

PS	B/W/D/K/S	<String number or string with token>
PS	B/W/D/K	<String number or string with value>
CM	9271	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data was changed
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	3	Parameter invalid/invalid number of the target string
	12	Token string or string value too long
	13	Internal error
	20	Module was not started from a spawn job or submit job

Module 9277 Writing data into the OEM log

With Module 9277 the PLC can write data into a specific OEM log. Up to eight OEM logs can be used at the same time. The logs created are UTF8-encoded, as is the control log.

The log can be written to the TNC: or to the PLC: partition.

The module can be called from a cyclic PLC program or from a spawn job or submit job. The string for the log entry may contain two place holders (data1 and data2). Only specified place holders will be replaced. The output format is controlled through the entry %d for integers or the entry %f for floating point numbers with three decimal places. Alternately, you can define the number of decimal places with %.1f to %.6f. Example of a string for the log entry:

S“data1: %.2f data2: %d”

If the maximum log size of 1 MB is exceeded, the log is copied to <name>.LOG.OLD, and a new log with the same name is created. Once the logs have been called, they remain open until the control is shut down.

Call:

PS	B/W/D/K/S	<Path with file name (without extension *.LOG)>
PS	B/W/D/K/S	<String with place holder for log entry>
PS	B/W/D/K	<Value for data1>
PS	B/W/D/K	<Value for data2>
PS	B/W/D/K	<Switch for additional entries> Bit 0 = 0/1: Entry without/with time stamp Bit 1 = 0/1: Entry without/with PLC cycle counter Bit 2 = 0: Changes are buffered, and written to the file in intervals of one minute Bit 2 = 1: Entry is written immediately, and file is closed again. **Important:** If you call the module cyclically, and Bit 2 =1 is set, the create a very high system load on the control!
CM	9277	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data written into OEM log
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	2	Invalid string number or invalid string
	22	Message cannot be transmitted
	36	Invalid path specified

Module 9291 Starting an NC macro

With Module 9291, you can call an NC macro in any operating mode. They are executed like cycles, without block display. The control-in-operation symbol is displayed while the macro is being executed. No macros can be activated if there is currently an **External emergency stop** error message.

The path name for the NC macro to be started is ascertained via the configuration object Path => CfgSystemCycle. The key (=name of the folder) defined under CfgSystemCycle also specifies the keyword with which the macro is called from the NC program.
You also specify with MP_path where the NC macro is stored.

Call:

PS	B/W/D/K/S	<Keyword>
CM	9291	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	NC macro was executed
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Key for the NC macro does not exist, or invalid string
	7	Macro cannot be executed.
	8	External emergency stop is active
	20	Module was not called in a spawn job or submit job
	28	NC program or other macro is already running
	29	The file given under the key is not an NC program (*.H or *.I)
	36	The file given under the key does not exist

Modules 9300 to 9302, 9304 to 9306 for Managing Multiple Tool Magazines

Any number of magazines and tools can be managed in the pocket table. The structure of the pockets in the magazine is specified via the table prototypes for the pocket table. The prototype for the pocket table consists of an empty pocket table, and is saved in the file **PLC:\proto\table\prototype.tch**.

The current tool magazine number is saved in a double word. You can configure the name for the operand any way you want in the T strobe via the **MP_pocketNumber** parameter.

Module 9302 searches for an open pocket in a tool magazine, and Module 9306 switches tools between the tool magazines.

Module 9301 determines the number of the entry in the pocket table. The number of the entry depends on the tool magazine and pocket numbers.

- Enter this number in the modules which cannot accept tool magazine numbers (e.g., Modules 9092, 9093, 9094).

The following topics are described:

- **Module 9300 Locking and releasing the pocket table**
- **Module 9301 Find the number of an entry in the pocket table**
- **Module 9302 Search for a vacant pocket in the tool magazine**
- **Module 9304 Copying OEM values from the pocket table**
- **Module 9305 Moving tools in the pocket table**
- **Module 9306 Moving tools between magazines**

Module 9300 Locking and releasing the pocket table

You must lock the pocket table with Module 9300 if you want to edit it with Modules 9305 or 9306.

After you are finished editing, you must release the pocket table with this module.

The module can only be called at standstill or during a strobe output.

The module may only be called in a spawn or submit job.

Call:

PS	B/W/D/K	<Code for locking/releasing> 0: Unlock 1: Lock
CM	9300	
PL	B/W/D	<Error code> 0: Locked / unlocked 1: Could not be locked 2: Could not be unlocked 3: Parameters for locking are invalid 4: Call was not in a submit or spawn job

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Pocket table locked/released
	1	See <error code> or NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	2	Invalid parameter for locking
	6	Table is already locked/is not reserved by the PLC
	20	Call was not in a submit or spawn job

Module 9301 Find the number of an entry in the pocket table

Module 9301 determines the number of an entry in the pocket table. This number is necessary for the modules in which no tool magazine numbers can be entered.

Call:

PS	B/W/D/K	<Tool magazine number>
PS	B/W/D/K	<Pocket number>
CM	9301	
PL	B/W/D	<Number of the entry in the pocket table> –1: M4203 = 1

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number of the entry was found
	1	Error code in W1022
NN_GenApiModuleErrorCode (W1022)	1	Invalid pocket number
	2	Invalid tool magazine number
	20	Module was not called in a spawn job or submit job

Module 9302 Search for a vacant pocket in the tool magazine

Module 9302 searches for a vacant pocket in a tool magazine.

Call:

PS	B/W/D/K	<Tool magazine number>
PS	B/W/D/K	<Pocket at which the search is to be started>
CM	9302	
PL	B/W/D	<Number of the vacant pocket> –1: No vacant pocket available

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Vacant pocket was found
	1	Error code in W1022
NN_GenApiModuleErrorCode (W1022)	1	Invalid pocket number
	2	Invalid tool magazine number
	20	Module was not called in a spawn job or submit job
	36	Error in file handling

Module 9304 Copying OEM values from the pocket table

Module 9304 copies the contents of columns p1 to p5 of the active pocket table a memory area of the PLC.

The module can only be called in a spawn or submit job.

Call:

PS	B/W/D/K	<Magazine number>
PS	B/W/D/K	<Pocket number>
PS	B/W/D/K	<Double-word address>
CM	9304	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	OEM value was copied
	1	See NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Pocket number is invalid
	2	Magazine number is invalid
	4	Double-word address is invalid
	20	Call was not in a submit or spawn job
	36	File error in pocket table
	41	Read of columns p1 to p5 failed because of invalid number format

Module 9305 Moving tools in the pocket table

With Module 9305 you can shift a tool in the pocket table to another pocket.

Constraints:

The module can only be called at standstill or during a strobe output.

The module can only be called in a spawn or submit job.

Before you call this module, the pocket table must be locked with Module 9300, and must later be released with Module 9300 again.

Only the column for the tool number is changed in the original and new entries. All other (pocket-specific) data is maintained.

Call:

PS	B/W/D/K	<Original pocket>
PS	B/W/D/K	<New pocket>
CM	9305	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Pocket exchange successful
	1	Pocket exchange not possible, see NN_GenApiModuleErrorCode for error
NN_GenApiModuleErrorCode (W1022)	2	Invalid parameter for pocket number
	6	Magazine management is active
	20	Call was not in a submit or spawn job
	21	Call was during program run without locking of the pocket table via Module 9300
	30	No valid tool entered in the original pocket
	36	Error during file access
	55	Pocket table cannot be locked

Module 9306 Moving tools between magazines

Module 9306 is used to exchange a tool between the defined magazines, or to move it beyond the magazine borders.

Constraints:

The module can only be called at standstill or during a strobe output.

Before you call this module, the pocket table must be locked with Module 9300, and must later be released with Module 9300 again.

The module can only be called in a spawn or submit job.

Only the columns for the tool number are changed in the original and new entries. All other (pocket-specific) data is maintained.

Call:

PS	B/W/D/K	<Original magazine>
PS	B/W/D/K	<Original pocket>
PS	B/W/D/K	<Target magazine>
PS	B/W/D/K	<Target pocket>
CM	9306	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Pocket has been exchanged
	1	Pocket exchange not possible, see NN_GenApiModuleErrorCode for error
NN_GenApiModuleErrorCode (W1022)	1	Pocket number is invalid
	2	Magazine number is invalid
	6	Magazine management is active
	20	Call was not in a submit or spawn job
	21	Call was during program run without locking of the pocket table via Module 9300
	30	Programmed pocket is not occupied
	36	Error during file access
	55	Pocket table cannot be locked

Module 9322 Information of the current NC program

Module 9322 can interrogate information about the NC program currently selected or being machined.

The currently selected NC program can be interrogated in the cyclic program part as well as from a spawn or submit task. The path and file name are returned. If no NC program is selected, the module returns an empty string. The interrogation of the selected NC programs always returns 0 as the block number.

The interrogation of the NC program currently being machined is only possible from a spawn or submit task. The path name, file name, and current block number of the program being machined are returned. If no NC program is currently being machined, the module returns an empty string and 0 as the current block number.

Call:

PS	B/W/D/K	<Mode> 0: Block number of the current NC (sub-)program, no information about cycle calls 2: Only name of the NC main program
PS	B/W/D/K	<String number>
PL	B/W/D	<Block number>
CM	9322	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Marker was written
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	1	Invalid mode setting
	2	Invalid PLC string number
	12	Path name is longer than PLC string
	13	Internal error
	20	Module was not started from a spawn job or submit job

Module 9340 Searching for a pocket depending on magazine rules

Module 9340 searches through a tool magazine for vacant, reserved or occupied pockets.

Constraints:

The module influences the **RSV** column and all **LOCKED** columns in the pocket table. These columns may not be changed by the PLC program or with the table editor.

The search for a vacant pocket depends on the magazine rules, which must be specified in the pocket table.

If the pocket table has not yet been locked with Module 9300, the module tries to lock it itself.

Call:

PS	B/W/D/K	<Magazine number>
PS	B/W/D/K	<Pocket number for starting the search>
PS	B/W/D/K	<Tool number or tool type>
PS	B/W/D/K	<Mode> Bit 0 = 0: Programmed number is tool type Bit 0 = 1: Programmed number is tool number Bit 1 = 1: Search for a vacant pocket (depending on magazine rules) Bit 2 = 1: Search for a reserved pocket (independent of the pocket for starting the search) Bit 3 = 1: Search for an occupied pocket (independent of the pocket for starting the search)
CM	9340	
PL	B/W/D/K	<Tool number in the programmed magazine> –1: Error code NN_GenApiModuleErrorCode –2: No free pocket or tool not found

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Pocket search performed successfully
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	1	Transferred pocket number is invalid
	2	Transferred magazine number is invalid
	3	Transferred mode is invalid
	4	Tool number or tool type is invalid
	20	The module was not called from a spawn or submit job
	36	File error in the tool or pocket table
	45	Module canceled, failure evaluation based on return value
	55	Pocket table could not be locked

Module 9341 Editing a pocket table depending on magazine rules

Module 9341 reserves, releases, or makes pockets unavailable in the pocket table, in accordance with the magazine rules.

The module affects the columns **RSV, LOCKED_ABOVE, LOCKED_BELOW, LOCKED_LEFT,** and **LOCKED_RIGHT.** Therefore, these columns may not be changed manually nor by the PLC program.

Call:

PS	B/W/D/K	<Magazine number>
PS	B/W/D/K	<Pocket number>
PS	B/W/D/K	<Tool number>
PS	B/W/D/K	<Mode> 0: Release pocket (depending on magazine and tool number) 1: Release pocket (depending on magazine and pocket number) 2: Reserve pocket (depending on magazine, pocket and tool number) 3: Make pocket unavailable (depending on magazine and pocket number) 4: Reserve pocket if previously unavailable (depending on magazine and pocket number)
CM	9341	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Pocket table edited
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Invalid pocket number
	2	Invalid magazine number
	3	Invalid mode
	4	Invalid tool number
	6	Reservation not possible
	7	Magazine rules not compiled or not present
	20	Module was not called in a spawn job or submit job
	36	File error in pocket table

Module 9342 Find magazine and pocket number

Module 9342 determines the magazine and pocket number from the tool number. The module takes the **RSV** column of the pocket table into account if magazine rules are in effect. If the module is used to find reserved pockets, it returns the first reserved pocket with ascending magazine number. However, further pockets can be reserved. In this case the search must be repeated with another “start magazine for the search.” .

Call:

PS	B/W/D/K	<Tool number>
PS	B/W/D/K	<Mode>
		0: Look for occupied pocket
		1: Look for reserved pocket
PS	B/W/D/K	<Start magazine for the search>
CM	9342	
PL	B/W/D/K	<Magazine number>
		–1: Magazine could not be found
PL	B/W/D/K	<Pocket number>
		–1: Pocket could not be found

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Magazine and pocket number found
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	1	Invalid mode
	2	Invalid start magazine for the search
	20	Module was not called in a submit job or spawn job
	30	Tool not found
	36	File error in pocket table

Module 9343 Compilation and activation of magazine rules

Module 9343 is used to compile and activate magazine rules (*.TCR) If an error occurs during compilation, the PLC program is stopped. The magazine rules must be activated during the first run of the PLC program or before the first call of Modules 934x. .

Call:

PS	B/W/D/K/S	<Path and file name of the magazine rules>
CM	9343	

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Magazine rules have been compiled and activated
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	11	Invalid string programmed
	20	Module was not called in a spawn or submit job
	38	Error during compilation

Module 9350 Read data from the tool table

Module 9350 reads the contents of a cell in the active tool table. The value is read as an integer value. .

Call:

PS	B/W/D/K	<Tool number>
PS	B/W/D/K	<Tool index> <= 0: Main entry
PS	B/W/D/K	<Element number> 0: Tool length L 1: Tool radius R 2: Not used 3: Replacement tool (–1 if not defined) 4: Not used 5: Maximum tool age TIME1 6: Maximum tool age for **TOOL CALL** TIME2 7: Current tool age CUR.TIME 8: Tool radius 2 R2 9: Oversize for tool length DL 10: Oversize for tool radius DR 11: Oversize for tool radius 2 DR2 12: Tool locked TL (0=No, 1=Yes)

13: Number of tool teeth CUT.
14: Wear tolerance in length LTOL
15: Wear tolerance in radius RTOL
16: Cutting direction DIRECT. (0=+; 1=–)
17: PLC status PLC
18: Tool offset for length TT: L-OFFS
19: Tool offset for radius TT: R-OFFS
20: Break tolerance for length LBREAK
21: Break tolerance for radius RBREAK
22: Tooth length LCUTS
23: Maximum plunge angle ANGLE
24: Tool number
25: Tool index
26: PLC value PLC-VAL
27: Probe center offset in reference axis CAL-OF1
28: Probe center offset in minor axis CAL-OF2
29: Spindle angle during calibration CAL-ANG
30: Tool type PTYP
31: Maximum speed NMAX
32: Retract tool LIFTOFF

CM	9350	
PL	B/W/D	<Element value>
PL	B/W/D	<Error number>

0: No error, element value was read
1: Module was not called in a spawn or submit job
2: File type does not exist
3: No tool table with status **M**
4: Line number does not exist
5: Incorrect element number

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Element value read
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	2	Incorrect element number
	7	Line number does not exist
	20	Module was not called in a spawn or submit job
	36	No tool table with status **M**

Module 9351 Write data to tool table

Module 9351 writes the contents of a cell to the active tool table. The value must be given as an integer value.

Call:

PS	B/W/D/K	<Tool number>
PS	B/W/D/K	<Tool index> –1: Write all indexes of a tool
PS	B/W/D/K	<Element number> See **Module 9350**
PS	B/W/D/K	<Element value>
CM	9351	
PL	B/W/D	<Error number> 0: No error, element value was written 1: Module was not called in a spawn or submit job 2: File type does not exist 3: No tool table with status **M** 4: Line number does not exist 5: Incorrect element number 6: Element value is out of range 7: Error while writing to the file

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Element value written
	1	Error code in NN_GenApiModuleErrorCode
NN_GenApiModuleErrorCode (W1022)	2	Incorrect element number
	7	Line number does not exist
	20	Module was not called in a spawn or submit job
	36	No tool table with status **M**

Module 9407 Give default tool number for an NC channel

You use Module 9407 to give a default tool number and index for an NC channel.

Constraints:

- The module must be called in the cyclic PLC program. It may not be executed in a submit job or spawn process.
- The module may only be executed as long as no NC program is being machined in this machining channel, or a strobe is currently synchronizing the calculation of the NC program (strobe with SYNC_CALC).
- This module is only supported by the new symbolic memory interface. If you are using the 6000i compatible interface, the module returns an error. The appropriate markers are to be used for the 6000i compatible interface.

Call:

PS	B/W/D/K	<Channel number>
PS	B/W/D/K	<Number of the tool>
PS	B/W/D/K	<Step index of the tool>
PS	B/W/D/K	<Mode bit-coded> Bit 0: Do not update pocket table
CM	9407	
PL	B/W/D	<Error number> 0: No error, default tool number assigned successfully 1: Invalid channel number 2: An NC program is running without a strobe, or a strobe without synchronization of the look-ahead calculation (SYNC_CALC) 3: Negative tool number 4: Negative tool index 5: Module was called in a spawn job or submit job

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Default tool number assigned successfully
	1	Error (error code in NN_GenApiModuleErrorCode)
NN_GenApiModuleErrorCode (W1022)	1	Invalid channel number given
	3	Negative tool number given
	4	Negative tool index given
	21	The calculation of the NC program is not synchronized with the execution of the module (no strobe with SYNC_CALC)
	24	The module was called in a submit job or spawn process

Module 9411 Read the actual spindle values (speed, coordinates)

Module 9411 reads the position and shaft-speed values of the spindle.

Constraints:

- This module is only supported by the new symbolic memory interface. If the 6000i compatible interface is used, the module returns an error.

Call:

PS	B/W/D/K	<Logical spindle number> 0: Spindle 1 1: Spindle 2 etc.
PS	B/W/D/K	<Desired spindle information> 1: Actual position 2: Nominal position 3: Actual position in the reference system 4: Following error (servo lag) 10: Actual shaft speed 11 Nominal shaft speed
CM	9411	
PL	D	<Spindle information> For 1 to 4: Value in 0.0001° For 10 to 11: Value in 0.0001 rpm For 20: 0: Spindle in wye operation

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error, spindle information was read
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Requested spindle number or spindle information is invalid
	99	Module is not supported (control does not operate with symbolic memory interface)

Module 9416 Select gear range and assigned settings for spindle

The module selects the gear range and the assigned parameter set, as well as other settings, for a spindle.

Constraints:

This module is only supported by the new symbolic memory interface. If the 6000i compatible interface is used, the module returns an error.

If no gear ranges are configured, but other settings are to be changed, then the gear range 0 must be entered.

Call:

PS	B/W/D/K	<Logical spindle number>
PS	B/W/D/K	<Mode> Bit 0: Direction of spindle rotation 0 = Direction of rotation not inverted 1 = Direction of rotation inverted
PS	B/W/D/K	<Gear range>
CM	9416	
PL	B/W/D	<Error number> 0: Module successfully executed 1: Faulty module call (invalid spindle number) 2: Faulty module call (negative gear range) 3: Invalid gear range

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	No error, spindle is being rotated
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid task data transferred (see error number in returned value)
	99	Module is not supported (control does not operate with symbolic memory interface)

Module 9417 Set default shaft speed for spindle

You use Module 9417 to enter a default shaft speed for the spindle for the display.

Constraints:

- The default shaft speed for the display is not offset by the spindle override (i.e., any value set with the override potentiometer must be included via the PLC program).
- The maximum actual shaft speed is shown, as well as the shaft speed entered via the module.
- The shaft speed for calculations in the NC program may only be set if no NC program is being executed or if there is currently a strobe for synchronization of the block scan. The value SYNC_CALC must be set for **MP_sync** when configuring the strobe.
- The shaft speed must not be negative.
- Modes 2 to 4 of this module are only supported by the new symbolic memory interface. If you are using the 6000i compatible memory interface (API 1.0), the module returns an error.

Call:

PS	B/W/D/K	<Logical spindle number>
PS	B/W/D/K	<Mode>
		1: Shaft speed for display
		2: Shaft speed for NC program
PS	B/W/D/K	<Speed in 1/1000 rpm>
CM	9417	
PL	B/W/D	<Error number>
		0: Module successfully executed
		1: Invalid spindle number given
		2: Invalid mode given
		3: Negative shaft speed given
		4: Shaft speed for NC program was changed during program run

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Module executed successfully
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid task data transferred (see error number in returned value)
	21	Shaft speed for NC program was changed during program run
	99	Module is not supported (control does not operate with symbolic memory interface)

Module 9418 Set status for spindle

You use Module 9418 to enter a new status for the spindle.

This module is only supported by the symbolic memory interface. If you are using the 6000i compatible memory interface (API 1.0), the module returns an error.

Call:

PS	B/W/D/K	<Logical spindle number>
PS	B/W/D/K	<Mode> 1: Evaluate the spindle reference mark again
PS	B/W/D/K	<Value> For mode =1: Without meaning
CM	9418	
PL	B/W/D	<Error number> 0: Module successfully executed 1: Invalid spindle number given 2: Invalid mode given

Error recognition:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Module executed successfully
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid task data transferred (see error number in returned value)
	99	Module is not supported (control does not operate with symbolic memory interface)

Number Conversion

The following topics are described:

- **Module 9050 Conversion of binary numbers Æ ASCII**
- **Module 9051 Conversion of binary numbers Æ ASCII**
- **Module 9052 Conversion of ASCII numbers Æ binary**
- **Module 9053 Conversion from binary Æ ASCII/hexadecimal**
- **Module 9054 Conversion from ASCII/hexadecimal Æ binary**

Module 9050 Conversion of binary numbers Æ ASCII

Module 9050 converts a binary numerical value consisting of a mantissa and exponent to base 10 into an ASCII-coded decimal number and saves it as a string in the specified address. The exponent refers to the least significant place of the number. The control detects a negative number when the mantissa corresponds to a negative number in the notation as a two's complement. The control sets an algebraic sign only before negative numbers. The control does not convert trailing zeros after the decimal point or leading zeros before the decimal point. The control writes the string left-aligned in the string address that you specify.

Constraints:

The decimal character is defined by machine parameter MP7280 as a comma (MP7280 = 0) or a period (MP7280 = 1).

Call:

PS	B/W/D/K	<>Mantissa of the number to be converted>
PS	B/W/D/K	<>Exponent to base 10 of the value>
PS	B/W/D/K	<>String address in which the control saves the ASCII-coded decimal number>
CM	9050	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number was converted
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid string address or invalid exponent

Module 9051 Conversion of binary numbers Æ ASCII

Module 9051 converts a binary numerical value into an ASCII-coded decimal number in the specified format and saves it as a string in the specified address. The number is interpreted as a two's complement. For algebraically unsigned notation, the control converts the absolute amount of the number without putting a sign before the string. For algebraically signed notation, the control sets an algebraic sign ("+" or "–") in front of the string in any event. For notation in inches, the number is divided by 25.4 before conversion. If the number has more decimal places than the total that you have specified for the number of places before and after the decimal point, then the control omits the most highly significant decimal places. In right-aligned notation leading zeros before the decimal point are replaced by blanks; in left-aligned notation they are suppressed. Trailing zeroes after the decimal point are always converted.

Constraints:

The decimal character is defined by machine parameter MP7280 as a comma (MP7280 = 0) or a period (MP7280 = 1).

Call:

PS	B/W/D/K	<>Numerical value to be converted>
PS	B/W/D/K	<>Display modes, bit-encoded> Bit 1/0: Format 00: Sign and number left-aligned 01: Sign left-aligned, number right-aligned 10: Sign and number right-aligned 11: Not permissible Bit 2: Display converted to INCH Bit 3: Display with sign
PS	B/W/D/K	<>Number of places after the decimal point>
PS	B/W/D/K	<>Number of places before the decimal point>
PS	B/W/D/K	<>String address in which the control saves the ASCII-coded decimal number>
CM	9051	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number was converted
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid string address, invalid display mode or invalid number of places before or after the decimal point

Module 9052 Conversion of ASCII numbers Æ binary

Module 9052 converts an ASCII-coded decimal number (possibly with decimal places) into a signed number and an exponent to the base of 10. You must assign the ASCII-coded decimal number to one of the string memories. If the number has no algebraic sign, the control interprets it as a positive number and accepts both a point and a comma as decimal character. If the full extent of the mantissa cannot be represented in a double word, then the last places are omitted and the exponent is corrected accordingly. If possible, the control adjusts the exponent so that it corresponds to the ASCII notation.

Call:

PS	B/W/D/K	<>String address in which the ASCII-coded decimal number is saved>
CM	9052	
PL	B/W/D	<>Numerical value>
PL	B/W/D	<>Exponent to the base of 10 of a value>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number was converted
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid string address or string contains none or too many characters

Module 9053 Conversion from binary Æ ASCII/hexadecimal

Module 9053 converts blocks of binary values from the word-marker range into a string of ASCII-coded hexadecimal numbers. The control reads the specified number of bytes from the word address that you have specified and converts it to a hexadecimally coded ASCII string. Each byte produces two characters in the string memory.

Call:

PS	B/W/D/K	<>Word address from which control saves the binary values>
PS	B/W/D/K	<>String address in which the control saves the hexadecimal numbers>
PS	B/W/D/K	<>Number of data bytes>
CM	9053	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number was converted
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Too many data bytes
	2	Invalid string address
	4	Invalid word address

Module 9054 Conversion from ASCII/hexadecimal Æ binary

Module 9054 converts strings of ASCII-coded hexadecimal values into a block of binary values in the word-marker range. The string in the specified string memory is interpreted as a sequence of ASCII-coded hexadecimal numbers and converted into a block of corresponding binary bytes. Two ASCII characters produce one binary byte. The control saves the binary block beginning at the specified address in the word-marker range.

Call:

PS	B/W/D/K	<>String address in which the hexadecimal value is saved>
PS	B/W/D/K	<>Word address from which the control saves the binary values>
CM	9054	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Number was converted
	1	Error, see NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	2	Invalid string address
	11	Invalid word address
		Odd number of characters in the string or a character that cannot be interpreted as hexadecimal

Example

Initial state:

SO = ”63”
BO = 99

Function	STL	String accu. (characters), data stack [bits]
Push string address S0 onto the data stack.	PS K+0	63
Push word address B0 onto the data stack.	PS B0	99
Conversion of the two ASCII characters 6 and 3 in the binary number 99.	CM 9054	01100011

Section 8 - Data Interfaces

The following topics are described in this section:

- **Introduction**
- **The Ethernet Interface**
- **The USB Interface of the Control**
- **The Serial Interface of the Control**
- **Configuring the Serial Interface**
- **Data Transfer by PLC**

Introduction

In addition to their Central Processing Unit (CPU), computer systems usually include various peripheral devices.

A CPU is, for example:

- PC
- Control

Peripheral devices include:

- Printers
- External storage devices, such as hard disks, floppy-disk drives, or USB memory sticks.
- Other computer systems

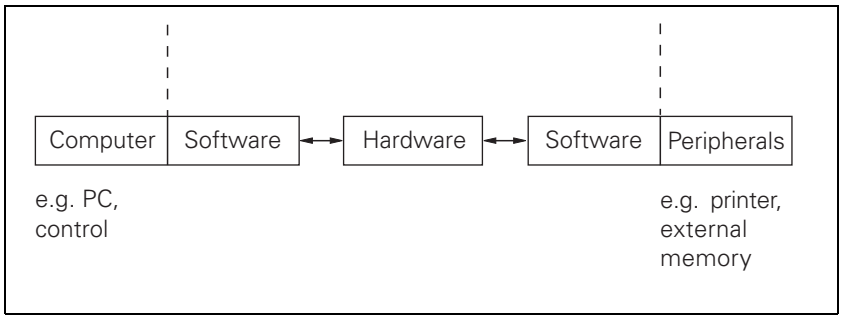
A data interface makes it possible for the CPU and its peripheral devices to communicate.

The interfaces, which consist of physical links between the computer system and the peripherals, need a transmission line, and appropriate software in order to transfer data between the individual units.

Standard interfaces include:

- Ethernet
- USB 1.1
- RS-232-C/V.24
- RS-422/V.11

The relationship between hardware and software, which fully defines an interface, is illustrated by the following diagram:



The hardware in the diagram covers all the physical components, such as

- Circuit construction
- Pin layout
- Electrical characteristics

The software is the operating software, which includes, for example, the drivers for the output modules.

The Ethernet Interface

You can connect the control to your plant's intranet or use a transposed cable to connect directly with a PC. The data transfer rate is dependent on the amount of traffic at the time on the net. For information on the pin layout: see “**Section 2, X26: Ethernet interface RJ45 Port**”.

X26: Ethernet interface RJ45 connection (10BaseT)

Maximum cable length:
Unshielded: 100 m
Shielded: 400 m

Network topology: Star configuration

This means a hub serves as a central node that establishes the connection to the other participants.

The control requires an NFS server (Network File System) or a Windows®**1 PC (SMB = Server Message Block) as the remote station. It must work according to the TCP/IP protocol principle.

OSI 7-layer model		Control
7	Application layer	NFS, SMB
6	Presentation layer	
5	Communications layer	
4	Transport layer	TCP protocol
3	Network layer	IP protocol
2	Data link layer	Ethernet card
1	Physical layer	

Before networking, the TNC must be properly configured. Please discuss the required settings with your network supervisor.

**1 Windows® is a registered trademark of Microsoft Corporation.

The USB Interface of the Control (USB 1.1)

The **U**niversal **S**erial **B**us (USB) interface is a standard serial interface.

USB 1.1 provides a maximum data transfer rate of 12 Mbps.

Various USB block devices, such as keyboard, mouse, external hard disks, and USB memory sticks, can be connected to the control via the USB interface (X141, X142).

Note: If USB components require more than 0.5 A, a separate power supply becomes necessary for these components. One possibility is the USB hub from ANILAM (368 735-01).

The USB interface features the “hot-plug capability.” This means that you can connect USB devices to the USB interface and remove them, without having to shut down and then restart the control.

Transmission distance without hub: Up to 6 meters

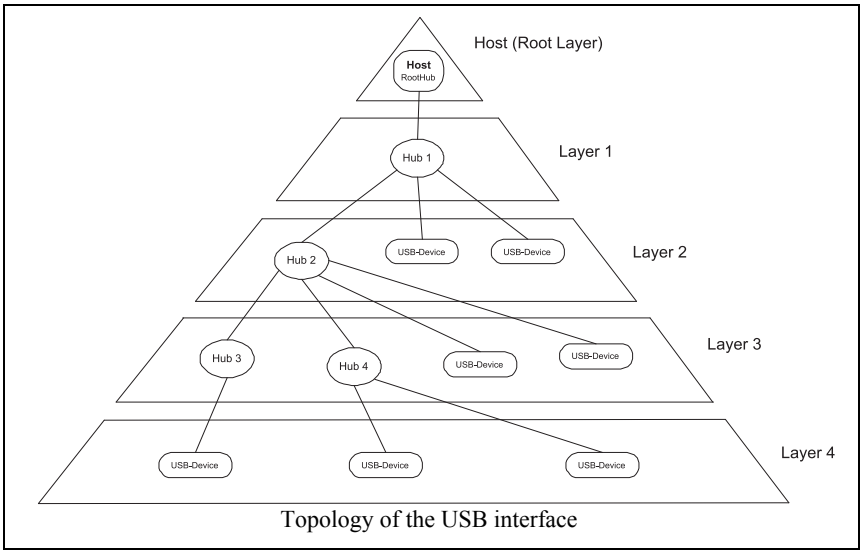
Note: For greater transmission distances, you must use a USB hub after every six meters in order to amplify the signal. You can use more than one hub for one transmission distance. USB cables with a length of up to 36 meters (with 6 integrated USB hubs) are available from ANILAM.

The following topics are described:

- **Structure**
- **Functionality and Signal Designations**
- **USB Devices on the Control**
- **USB Devices Tested by ANILAM**

Structure

The USB interface connects the USB peripheral devices with the USB host. The topology of a USB connection may consist of several levels arranged in a star configuration. Every level consists of a USB hub to which other USB devices or hubs are connected in a star configuration. A maximum of 127 USB devices can be connected to a USB host in this way.



Topology of the USB interface

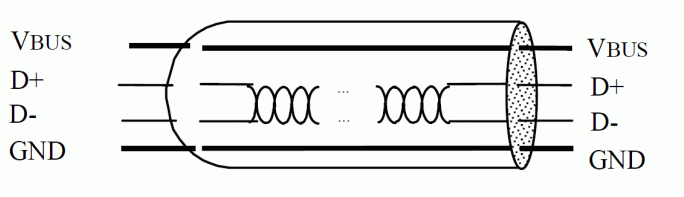
Functionality and Signal Designations

USB uses packet-based communication over two differential data lines. This reduces radiation and increases transmission reliability. USB provides significantly higher data transfer rates than the previous external interfaces (parallel / Centronics, serial / RS-232, RS-422):

- USB full speed of up to 12 Mbps
- USB low speed of up to 1.5 Mbps

Conventional interfaces, such as the RS-232, are more suitable for time-critical applications because they are not based on packets which reduce the transfer rate (in case of packets with only a few bytes) or delay transmission (when collecting bytes for filling a packet).

Only four wires of a USB cable are needed. Two for a power supply of 5 V (with max. 500 mA / 2.5 W) and two for data transmission.



Structure of the USB cable

USB Devices on the Control

The USB interface of the control allows for convenient and fast exchange of data. You can connect USB block devices, such as memory sticks, hard disks, CD-ROM drives, to your control via the USB interface without having to reboot the system. The data media can be accessed immediately after connection.

The control supports the following USB block devices:

- Floppy disk drives with FAT/VFAT file system
- Memory sticks with FAT/VFAT file system
- Hard disks with FAT/VFAT file system
- CD-ROM drives with FAT/VFAT file system

The control does not support USB devices with other file systems (e.g., NTFS). If you try to connect such devices, the control will issue an error message.

Note: It should basically be possible to connect all USB block devices with the above-mentioned file system to the control. If you nevertheless encounter problems, please contact ANILAM.

USB Devices Tested by ANILAM

A variety of USB storage media from different manufacturers is available on the market. It may happen that a USB device is not identified correctly by the control. The USB devices listed in the table below were tested by ANILAM for proper functioning in conjunction with the control (numerous other USB devices are supported by the control, but you should test them for proper functioning on the control before using them):

USB device	Manufacturer	Model designation	VendorID	ProductID	Revision
Floppy disk drive	TEAC	TEAC FD-05PUW	0644	0000	0.00
Floppy disk drive	TEAC	TEAC FD-05PUB	0644	0000	0.00
CD-ROM drive	TEAC	USB CD-ROM 210 PU	0644	1000	1.33
CD-ROM drive	FREECOM	USB2-IDE Controller	07ab	fc02	11.10
Hard disk	UNKNOWN	USB to IDE Converter	05e3	0702	0.02
Memory stick	TrekStor	USB MiniStick	0c76	0007	1.00
Memory stick	QDI	UNKNOWN	0c76	0005	1.00
Memory stick	Transcend	TS512MJFLASH	058f	9380	1.00
Memory stick	Transcend	Flash Disk	0ea0	2168	2.00
Memory stick	Generic	Mass Storage Device	058f	9384	1.05

The Serial Interface of the Control

The following topic is described:

- **RS-232-C/V.24 Interface**

RS-232-C/V.24 Interface

With RS-232C/V.24, data transfer is executed asynchronously, with a start bit before each character and one or two stop bits after each character.

Transmission distance: up to 20 m

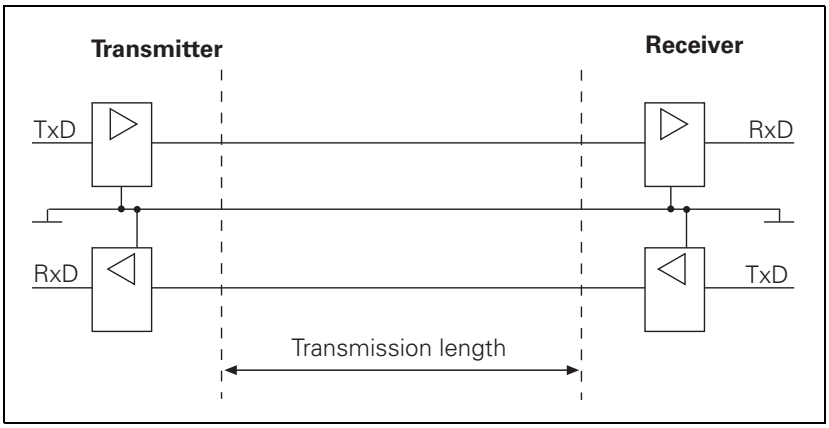
The following topics are described:

- **Hardware**
- **Signal Levels**
- **Signal Designations**
- **Pin Layout**

Hardware

The physical connection between two RS-232-C/V.24 interfaces is an asymmetrical line (i.e., the common ground connection between transmitter and receiver is used as a return wire).

Physical connections:

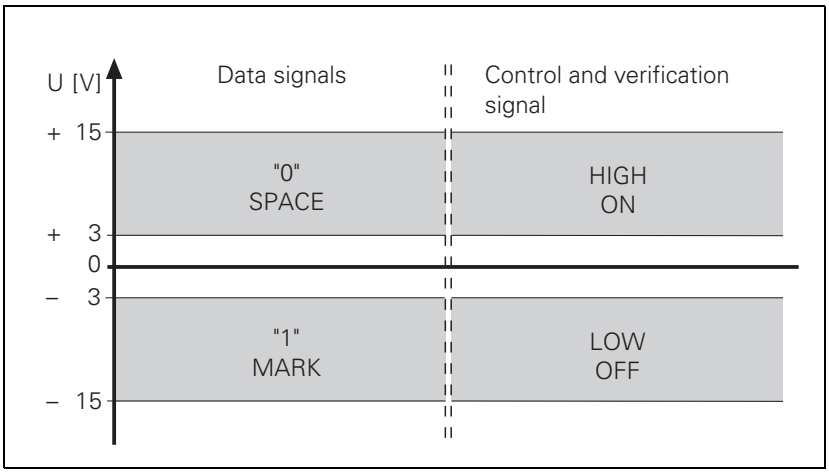


Signal Levels

The levels of the individual signal lines differ:

Data lines: The data signals are defined as being logical zero (SPACE) over the range +3 V to +15 V and logical one (MARK) over the range –3 V to –15 V.

Control and verification lines: These signals are defined as being ON (High) over the range +3 V to +15 V and as OFF (Low) over the range –3 V to –15 V.



Note: For all signals: The voltage range from –3 V to +3 V cannot be evaluated.

Signal Designations

One must differentiate between the following types of lines and their signals:

- Data lines:
  - TxDTransmitted data
  - RxDReceived data
- Control and signal lines:
  - DCD (Data Carrier Detect):
    Received signal level. The receiver signals that the information it has received lies within the defined level. The DCD signal is not used by the control. The control delivers no signal from this pin.
  - DTR (Data Terminal Ready):
    Control is ready / not ready for operation (e.g., the receiving buffer is full, the signal DTR indicates “LOW”).
  - DSR (Data Set Ready):
    Peripheral ready / not ready for service.
  - RTS (Request to Send):
    Switch transmission unit on. The control wants to transmit data.
  - CTS (Clear to Send):
    Readiness for transmission. The peripheral wishes to transmit data.
- Ground conductors (lines for power supply):
  - Chassis GND:
    Housing connection
  - Signal GND:
    0-V lines for all signals

Pin Layout

Keep in mind that there might be a difference between the pin layout of the control and the adapter block.

Configuring the Serial Interface

The following topics are described:

- **Control Characters**
- **Configuration of Interfaces**

Control Characters

Overview of control characters specific to ANILAM

Character	Designation	Description
SOH	Start of Header	Identifies the beginning of the data transfer header. The character string contains the program number and information about the type of program and the transfer mode.
STX	Start of Text	Identifies the beginning of a program block.
ETB	End of Text Block	Terminates a data transfer block. The character that follows (BCC) is used for data checking.
DC1	XON	Starts the transfer of data.
DC3	XOFF	Stops the transfer of data.
ETX	End of Text	Transmitted at the end of a program.
EOT	End of Transmission	Terminates the data transfer and establishes the idle state. This character is transmitted by the control at the end of a program input and to the external device in the event of an error.
ACK	Acknowledgment	Transmitted by the receiver when a data block has transferred without error.
NAK	Negative Acknowledgment	Transmitted by the receiver when a data block has transferred with an error. The transmitter must re-transmit the data block.

Configuration of Interfaces

Settings in the configuration editor:	
System Network Serial CfgSerialPorts **interfaceRs232** **interfaceRs422** interfacePlc **Key Interface PLC 0** **Key Interface PLC 1** **Key Interface PLC 2** baudRateLsv2 CfgSerialInterface Key Interface Parameter **baudRate** **protocol** **dataBits** **parity** **stopBits** **flowControl** **fileSystem** **bccAvoidCtrlChar** **rtsLow** **noEotAfterEtx**	

The following topics are described:

- **Selecting a Parameter Block**
- **Creating Parameter Blocks, Configuring Interface Ports**
- **Defining the LSV2 Baud Rate**
- **Data Transfer Rate**
- **Communications Protocol**
- **Word Length**
- **Transmission Reliability**
- **Synchronization**
- **Data Transfer Check: Handshaking**

Selecting a Parameter Block

You have the possibility of managing multiple parameter blocks for the serial interface at the same time by using the configuration editor. Use different keys in **MP_CfgSerialPorts** to activate the parameter blocks. This enables you to change quickly between different settings, for example if you frequently connect peripheral devices with different interface parameters. You configure the interface parameters in **MP_CfgSerialInterface.** Under each keyname, the properties of a serial port are defined.

Different parameter blocks are also available for accessing the interface by PLC. They are defined by keynames in **MP_interfacePlc.** If a key was not defined, the default parameter block from **MP_CfgSerialInterface** will automatically be used.

Creating Parameter Blocks, Configuring Interface Ports

MP_CfgSerialInterface allows you to manage the parameter blocks for the serial interface. Every parameter block contains the properties of a serial port. In **MP_CfgSerialPorts,** you define which of the parameter blocks is active, see "**Selecting a Parameter Block**". The interface settings to be defined are described below.

MP_interfaceRs232

Keyname of the data blocks for the RS-232 interface

Format: String max. 18 characters

Input: Define the default parameter block for the serial RS-232 interface here. The "Default" data block is selected by default. But you can use any desired designation. The specified data block must be contained in **MP_CfgSerialInterface.** The data block is not effective if another data block was activated by the PLC.

MP_interfaceRs422

Keyname of the data blocks for the RS-422 interface

Format: String max. 18 characters

Input: Define the default parameter block for the serial RS-422 interface here. The "Default" data block is selected by default. But you can use any desired designation. The specified data block must be contained in **MP_CfgSerialInterface.** The data block is not effective if another data block was activated by the PLC.

MP_interfacePlc

Keynames of the data blocks for interface access by the PLC

Format: Array [0–2]

Input: A string of max. 18 characters
Here you can enter a maximum of three different keynames for interface accesses by the PLC. If no parameter block is specified, the control automatically uses the default parameter block defined in **MP_CfgSerialInterface.**

Defining the LSV2 Baud Rate

MP_baudRateLsv2

Data transfer rate for LSV2 communication in baud-{}-

Format: Pull-down selection menu

Input: Use the drop-down menu to define the transfer rate for the LSV2 communication. Minimum value is 110 baud, maximum value 115200 baud.

Default: BAUD_57600

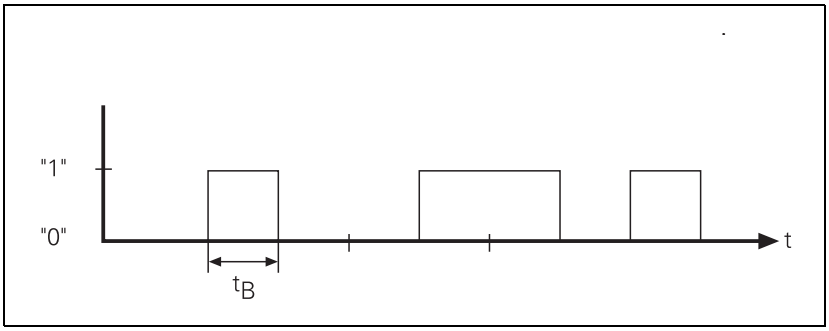
Data Transfer Rate

Baud Rate

The data transfer rate is given in baud (bits per second).
Common transfer rates are:

110, 150, 300, 600, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200 baud

The time taken to transmit one bit (t_B) can be calculated from the transfer rate:



$$t_B = \frac{1}{\text{transfer rate (bits/sec)}}$$

For example, a transfer rate of 19 200 bps will have a bit duration of t_B = 52.083 µs.

$$t_B = \frac{1}{19200\ (\text{bits/sec})} = 52{,}083\ \mu s$$

The number of characters transmitted per second can be calculated from the transfer rate and the transmission format:

$$\text{characters per second} = \frac{\text{transfer rate (bits/sec)}}{\text{number of bits per characters}}$$

Example:

With a transmission format of one start bit, 7 data bits, two stop bits, and a data transfer rate of 300 bps, exactly 30 characters per second will be transmitted.

$$\text{characters per second} = \frac{300\ (\text{bits/sec})}{1 + 7 + 2} = 30$$

MP_baudRate

Data transfer rate in baud

Format: Pull-down selection menu

Input: Use the drop-down menu to define the transfer rate for the interface. Minimum value is 110 baud, maximum value 115200 baud.

Communications Protocol

The protocol of a serial connection means the controlling of the data flow by feeding reserved ASCII characters into the data stream. Define the communications protocol of the interface in **MP_protocol**.

MP_protocol

Communications protocol

Format: Pull-down selection menu

Selection:

[STANDARD]

Standard data transfer
Transferring data line-by-line

[BLOCKWISE]

Blockwise data transfer
“ACK/NAK” protocol. Blockwise data transfer is controlled by the control characters ACK (Acknowledge) and NAK (Not Acknowledge).

[RAW_DATA]

Transfer without protocol
Transfer of characters without control characters
Protocol intended for transfer of “raw” data of the PLC.

Word Length

Data Bits

Define whether a character is transmitted with 7 or 8 data bits.

MP_dataBits

Data bits in each transferred character

Format: Pull-down selection menu

Selection:

[7 bits]

[8 bits]

Default: 8 bits

Transmission Reliability

Parity Bit

The parity bit is used to detect transmission errors.
The parity bit can take three different forms:

- No parity check (NONE): Error detection is dispensed with.
- Even parity (EVEN): The transmitter counts bits with a value of one. If the number is odd, the parity bit is set to one, otherwise it is cleared to zero. The sum of set data bits and the parity bit is therefore always even. Upon receiving a word, the receiver counts all of the set bits, including the parity bit. If the count is odd, there is a transmission error.
- Odd parity (ODD): The parity bit is so chosen by the transmitter that the total number of all the set bits is odd. An error will thus be detected if the receiver observes an even number of set bits in its evaluation.

Example:
The letter “z” corresponds to the bit sequence: 1 1 1 1 0 1 0

Parity bit

- With even parity = 1
- With odd parity = 0

MP_parity

Specifies the type of parity checking

Format: Pull-down selection menu

Selection:

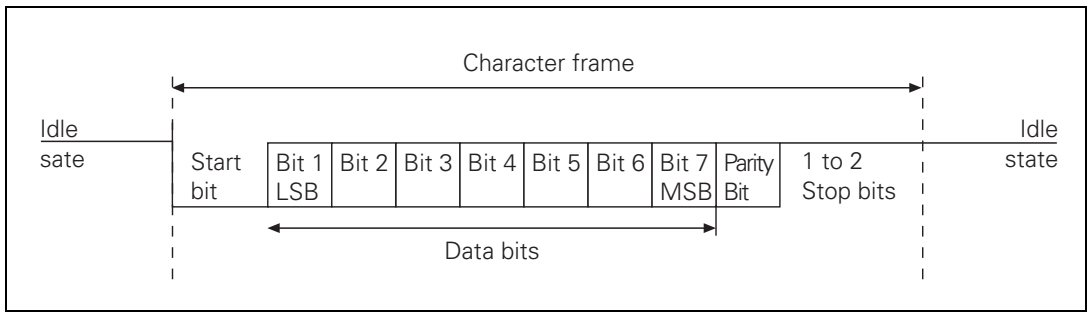
[NONE]
No parity check
[EVEN]
Even parity
[ODD]
Odd parity

Default: NONE

Synchronization

Stop Bits

The start bit and one or two stop bits enable the receiver to recognize each transmitted character during serial data transmission.



One start bit is sent before each character. In **MP_stopBits,** you determine the number of stop bits sent at the end of a character:

MP_stopBits

Number of stop bits

Format: Pull-down selection menu

Selection:

[1 stop bits]

[2 stop bits]

Default: 1 stop bit

Data Transfer Check: Handshaking

By handshaking, two devices control data transfer between them. A distinction is drawn between “software” and “hardware” handshaking.

The following topics are described:

- **Hardware Handshaking**
- **Software Handshaking**

Hardware Handshaking

Data transfer is controlled by electrical signals. Information, such as Clear to Send (CTS), Request to Send (RTS), “Start transmission” and “Stop transmission” is passed on by the hardware.

Example:

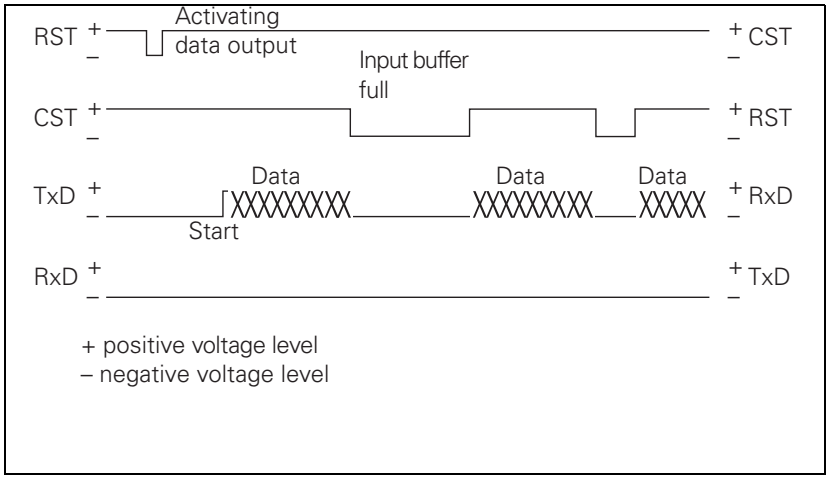
When a computer is to transmit a character, it checks the CTS signal line to see whether it is active (ON). If it is, the character is transmitted.

Hardware handshaking requires

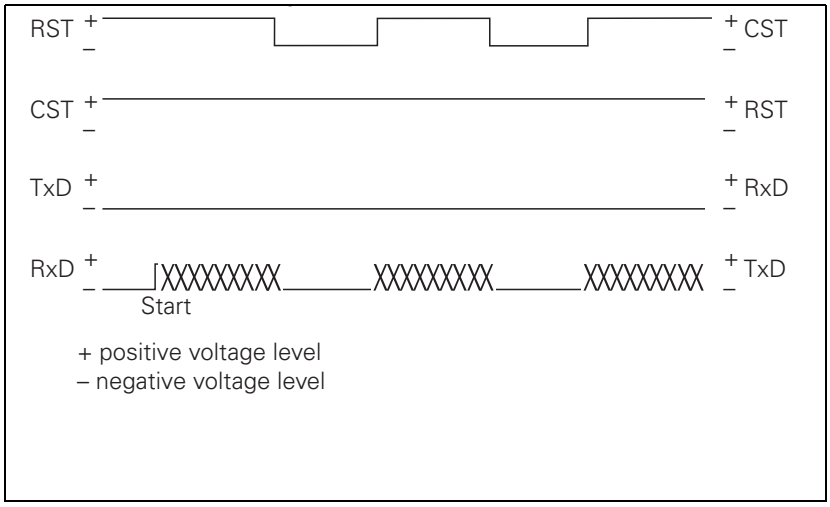
- The data lines TXD and RXD (transmitted and received data)
- The RTS control line (switching on transmitting unit)
- The CTS signal line (Clear to Send)
- A ground connection

The DTR and DSR signals indicate the operational status of the LE and peripheral device:

- DTR: Interrogated by peripheral; it is logic one if LE is ready for operation.
- DSR: Interrogated by LE.
- LOW level means: external data input/output not ready.
- HIGH level means: external data input/output ready.
- Data output from the control to EXT
  When the receiving buffer is full, the external device resets the RTS signal. The control detects that the peripheral unit receiving buffer is full at its CTS input:



- Data input from EXT to the control
  When the receiving buffer is full, the control removes the RTS signal. This is detected by the peripheral device at its CTS input:



Software Handshaking

Control of data transfer is achieved by control characters transmitted via the data line.

In **MP_flowControl,** you define whether the control stops transfer from an external device with control character <DC3>. Transfer is then resumed with character <DC1>. (XON/XOFF method)

If transfer is stopped with the control character <DC3>, up to three more characters can be stored; any further incoming characters are lost. Software handshake is normally recommended when interfaces are connected to an external device.

MP_MP_flowControl

Handshaking: Type of data-flow checking

Format: Pull-down selection menu

Selection:

[NONE]

No data-flow checking; handshaking not active

[RTS_CTS]

Hardware handshaking; transfer is stopped with RTS active

[XON_XOFF]

Software handshaking; transfer is stopped with DC3 (XOFF) active

Default: RTS_CTS

Data Transfer by PLC

The following topic is described:

- **PLC Modules**

PLC Modules

With the following PLC modules you can control the data interfaces from the PLC:

- Modules 9100 and 9101: Assign/release the data interfaces
- Module 9102: Interrogate the status of the interface
- Modules 9103 and 9104: Transmit and receive a string from the string memory. The transmit and receive buffers for the PLC are 128 characters long. Since every STRING ends with an END character, a STRING can only be up to 127 characters long.
- Modules 9105 and 9106: Transfer a block of binary values (bytes) from the word memory
- Module 9107: Read bytes from the receiving buffer without erasing the buffer
- Modules 9112 and 9113: Send or receive ASCII characters via the data interface

Strings and binary data are transmitted using ASCII characters. Example: Transfer of a block of binary data

Address	Value	ASCII character
.	.	.
B126	11111010	$FA
.	10000001	$81
.	.	
.	.	
.	.	

When transferring binary data starting from the address B126, the ASCII characters <F> <A> <8> <1> etc. are transmitted in sequence from the word memory through the interface. Each byte contains two ASCII characters. The transmitting and receiving buffers each hold 63 bytes.

The following topics are described:

- **Module 9100 Assign data interface**
- **Module 9101 Release data interface**
- **Module 9102 Status of data interface**
- **Module 9103 Transmit string through data interface**
- **Module 9104 Receive string through data interface**
- **Module 9105 Transmit binary data through data interface**
- **Module 9106 Receive binary data through data interface**
- **Module 9107 Read from receiving buffer**
- **Module 9112 Transmit ASCII characters via data interface**
- **Module 9113 Receive ASCII characters via data interface**

Module 9100 Assign data interface

With Module 9100 you assign an interface to the PLC and specify the transfer parameters. You initialize the interface. Any errors that occurred will be cleared. The interface is ready to receive.

Once assigned to the PLC, the interface is disabled for use by the input/output program of the user interface. The assignment is canceled when the PLC program is recompiled.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface>
		0: RS232
		1: RS422
PS	B/W/D/K	<>Parameter setting from MP_interfacePlc>
		0: Entry from interfacePlc[0] is used
		1: Entry from interfacePlc[1] is used
		2: Entry from interfacePlc[2] is used
CM	9100	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Interface was assigned
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface or transfer parameter
	13	No connection
	14	Interface busy or input/output not ready
	17	Incorrect data transfer rate
	20	Module was not called in a spawn or submit job

Module 9101 Release data interface

Module 9101 cancels the assignment of an interface to the PLC. The receive mode of the interface is canceled.

Can only be called in a submit job or spawn job!

Call:

PS	B/D/W/K	<>Interface> 0: RS232 1: RS422
CM	9101	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Interface enabled
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface
	14	Interface not assigned
	20	Module was not called in a spawn or submit job

Module 9102 Status of data interface

Module 9102 reads the status information about an interface in bit-coded form.

The information “interface ready” is updated when the interface is assigned to the PLC or NC. If the interface is not assigned, the module reads the last valid status.

Call:

PS	B/W/D/K	<>Interface>
		0: RS232
		1: RS422
CM	9102	
PL	B/W/D	<>Interface status>
		–1: Error code in NN_GenApiModuleErrorCode (W1022)
		Bit 0: Interface is assigned
		Bit 1: Interface is assigned to PLC
		Bit 2: Interface is ready
		Bit 3: Transmit buffer is empty
		Bit 4: Error during transmission
		Bit 5: Receive buffer is full
		Bit 6: Error in reception
		Bit 7: ETX was received (not ready to receive)
		Bit 8: Internal buffer from Module 9113 still contains characters

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Status read
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface

Module 9103 Transmit string through data interface

You must first assign the interface to the PLC and initialize it with Module 9100.

Module 9103 transmits a string from a string memory through one of the two interfaces. Links to the PLC error file and PLC dialog file are deleted.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface>
		0: RS232
		1: RS422
PS	K/B/W/D	<>Number of source string in the string buffer>
CM	9103	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	String sent
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface or incorrect string number
	12	No string end found
	13	Interface not ready
	14	Interface not assigned
	15	Transmit buffer not empty
	20	Module was not called in a spawn or submit job

Module 9104 Receive string through data interface

You must first assign the interface to the PLC and initialize it with Module 9100.

Module 9104 reads a string from the receive buffer of a serial interface in a string memory and resets the receive buffer.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface> 0: RS232 1: RS422
PS	K/B/W/D	<>Number of the string in the string buffer>
CM	9104	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Receive string
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface or incorrect string number
	12	String too long
	14	Interface not assigned
	16	Receiving buffer empty
	18	Transmission error or input/output not ready
	20	Module was not called in a spawn or submit job

Module 9105 Transmit binary data through data interface

You must first assign the interface to the PLC and initialize it with Module 9100.

Module 9105 transmits a block of binary values from the word memory of the PLC to one of the two interfaces. The transfer is in the form of ASCII-coded hexadecimal values. Every byte in the source block makes two ASCII characters at the interface.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface> 0: RS232 1: RS422
PS	K/B/W/D	<>Number of the first byte in the binary block>
PS	K/B/W/D	<>Length of the binary block (0 to 63)>
CM	9105	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data was transmitted
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface or incorrect byte number or block too long
	4	Block outside value range
	13	The interface is not ready or there is no connection
	14	Interface not assigned
	15	Transmit buffer not empty
	20	Module was not called in a spawn or submit job

Module 9106 Receive binary data through data interface

You must first assign the interface to the PLC and initialize it with Module 9100.

Module 9106 reads a block of binary values from one of the two interfaces into the word memory of the PLC. The transfer is in the form of ASCII-coded hexadecimal values. Every two ASCII characters from the serial interface make one byte in the binary block.

The length of the read binary block is returned as the initial variable.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface>
		0: RS232
		1: RS422
PS	K/B/W/D	<>Number of the first byte in the binary block>
CM	9106	
PL	B/W/D	<>Length of binary block in bytes>
		–1: Incorrect module call

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Data was received
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface or incorrect byte number or block too long
	4	Block outside value range
	11	Odd number of characters or illegal character
	12	String too long
	14	Interface not assigned
	16	Receiving buffer empty
	18	Transmission error or input/output not ready
	20	Module was not called in a spawn or submit job

Module 9107 Read from receiving buffer

You must first assign the interface to the PLC and initialize it with Module 9100.

Module 9107 reads two ASCII characters from the receive buffer to one of the two interfaces and codes them to a binary value.

You can specify an offset that corresponds to the position of the byte to be read in a binary block read by Module 9106. The contents of the receiving buffer are retained and can be read by Modules 9104 and 9106.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface>
		0: RS232
		1: RS422
PS	B/W/D/K	<>Offset of byte to be read in binary block>
CM	9107	
PL	B/W/D	<>Read binary value>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Receiving buffer was read
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface or byte number
	11	Illegal character
	12	String too long or offset too large
	14	Interface not assigned
	16	Receiving buffer empty
	18	Transmission error or input/output not ready
	20	Module was not called in a spawn or submit job

Module 9112 Transmit ASCII characters via data interface

You must first assign the interface to the PLC and initialize it with Module 9100. Module 9112 transmits a single ASCII character.

Note: Set MP5030.2 = 2 so that the transmitted characters do not disturb the set protocol procedure.

Define the characters in at least one word so that the values to 255 can be recognized.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface>
		0: RS232
		1: RS422
PS	W/D/K	<>ASCII code [0 to 255]>
CM	9112	

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Character was transmitted
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface
	13	The interface is not ready or there is no connection
	14	Interface not assigned
	15	Transmit buffer not empty
	20	Module was not called in a spawn or submit job

Module 9113 Receive ASCII characters via data interface

You must first assign the interface to the PLC and initialize it with Module 9100.

Module 9113 reads a single ASCII character from the receiving buffer of a serial interface and resets the receiving buffer.

If there is more than one character in the receiving buffer, the first is returned and the others are stored in a special buffer.

You can interrogate the current state with Module 9102, bit 8.

As long as data remains in the buffer, no further characters are collected from the interface.

If MP5030.2 < 2, the characters cannot be read from the interface until the line with the character requested in the protocol has been executed.

Note: Store the result in a word at least so that the values to 255 will be recognized.

Can only be called in a submit job or spawn job!

Call:

PS	B/W/D/K	<>Interface> 0: RS232 1: RS422
CM	9113	
PL	W/D	<>Read ASCII character [0 to 255 = ASCII characters; –1 = error>

Error code:

Marker	Value	Meaning
NN_GenApiModuleError (M4203)	0	Character was received
	1	Error code in NN_GenApiModuleErrorCode (W1022)
NN_GenApiModuleErrorCode (W1022)	1	Incorrect interface
	12	String too long
	13	The interface is not ready or there is no connection
	14	Interface not assigned
	16	Receiving buffer empty
	18	Transmission error or input/output not ready
	20	Module was not called in a spawn or submit job
	37	Receiving queue is full

Section 9 - Drawings

Drawings Listed

The following drawings are illustrated in this section:

- Figure 9-1, Console
- Figure 9-2, MC, CC, and Inverter
- Figure 9-3, MP 6000M Manual Panel
- Figure 9-4, MP 6001M Manual Panel
- Figure 9-5, CC 600 and MC 400
- Figure 9-6, CC 600 and MC 400 Dimensions
- Figure 9-7, I/O EXP. BASE: 4-SLOTS (P/N 624498-01, IEB 404), 6-SLOTS (P/N 624500-01, IEB 406), 8-SLOTS (P/N 624501-01, IEB 408)
- Figure 9-8, I/O EXP. BASE 4-SLOTS (P/N 624498-01, IEB 404) Connector Description
- Figure 9-9, Expansion Base Grounding
- Figure 9-10, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) Dimensions
- Figure 9-11, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) LEDs and Connectors
- Figure 9-12, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Dimensions
- Figure 9-13, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Connectors
- Figure 9-14, Hard Disk Drawer (P/N 574746-51, HDR) Dimensions
- Figure 9-15, Hard Disk Drawer (P/N 574746-51, HDR) Minimum Clearances
- Figure 9-16, Hard Disk Drawer (P/N 574746-51, HDR) Locking/Unlocking the Drive
- Figure 9-17, System ID Key (P/N 574744-51, SIK) Installation
- Figure 9-18, USB Hub (P/N 624508-01) Dimensions
- Figure 9-19, Basic Servo Turn On Circuit
- Figure 9-20, RM 500 Remote Handwheel, P/N 34000850
- Figure 9-21, PM 300 Panel-Mounted Handwheel, P/N 34000855
- Figure 9-22, Cable Overview
- Figure 9-23, Cable Overview, Modular
- Figure 9-24, Basic System Diagram

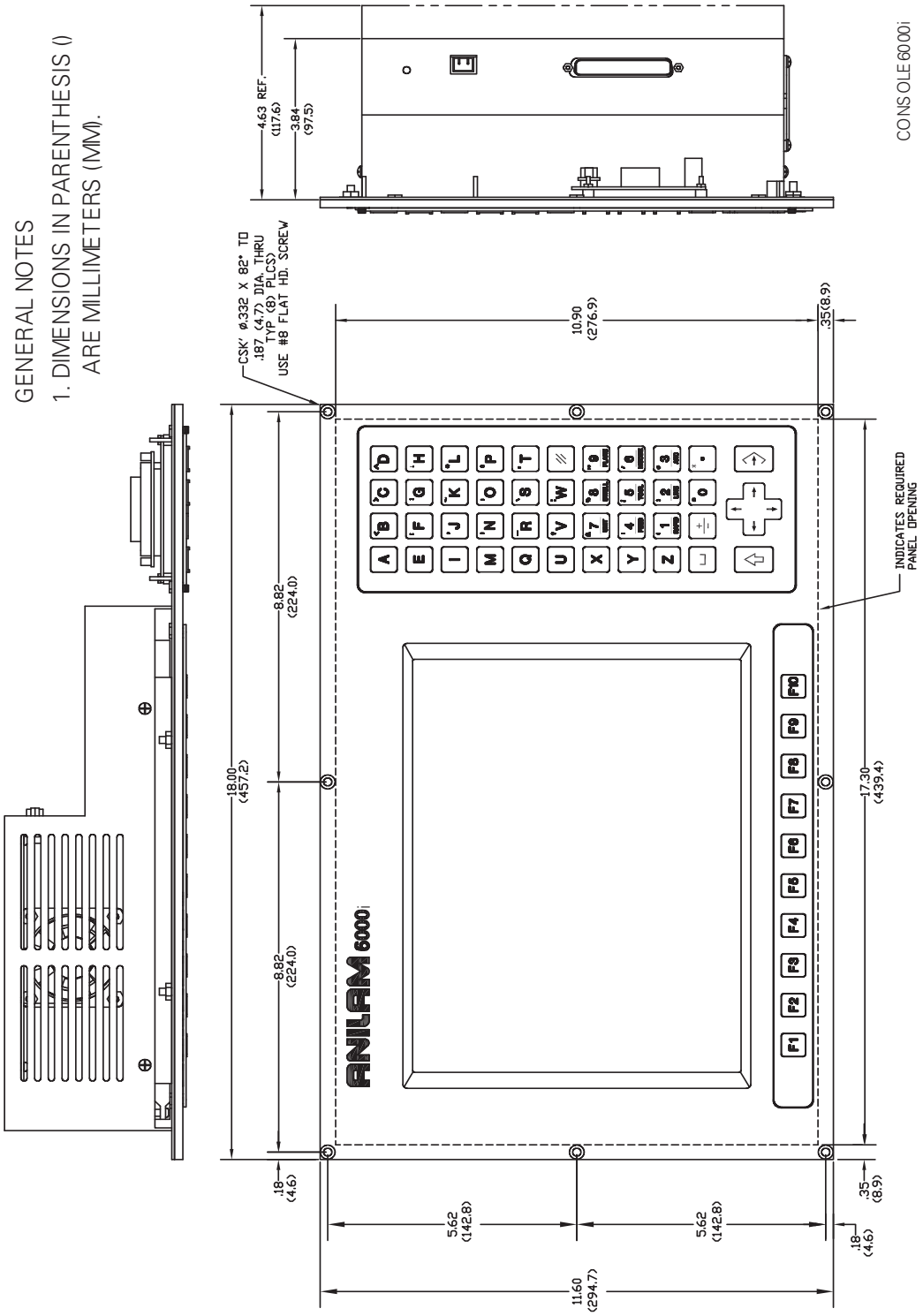


Figure 9-1, Console

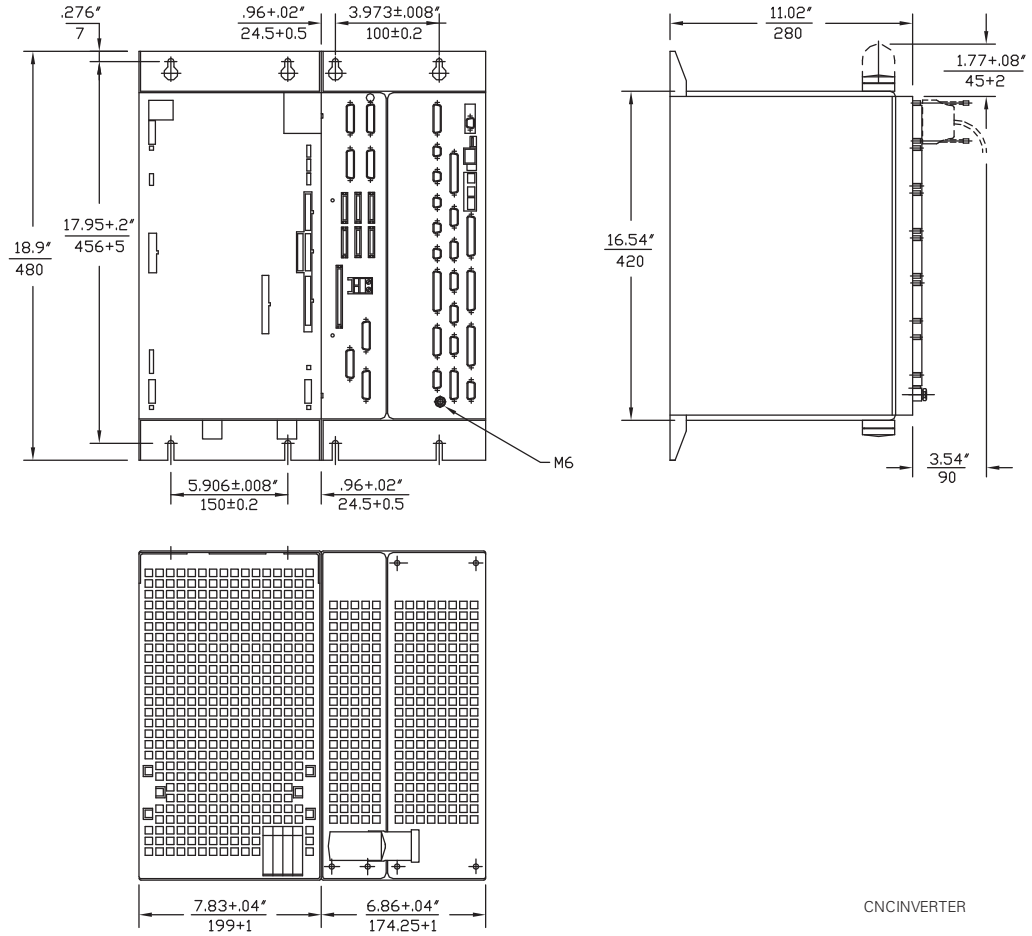


Figure 9-2, MC, CC, and Inverter

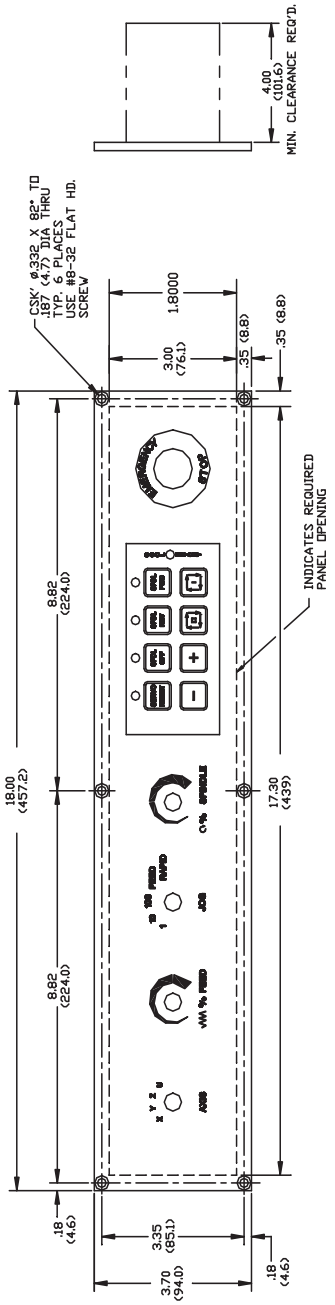


Figure 9-3, MP 6000M Manual Panel

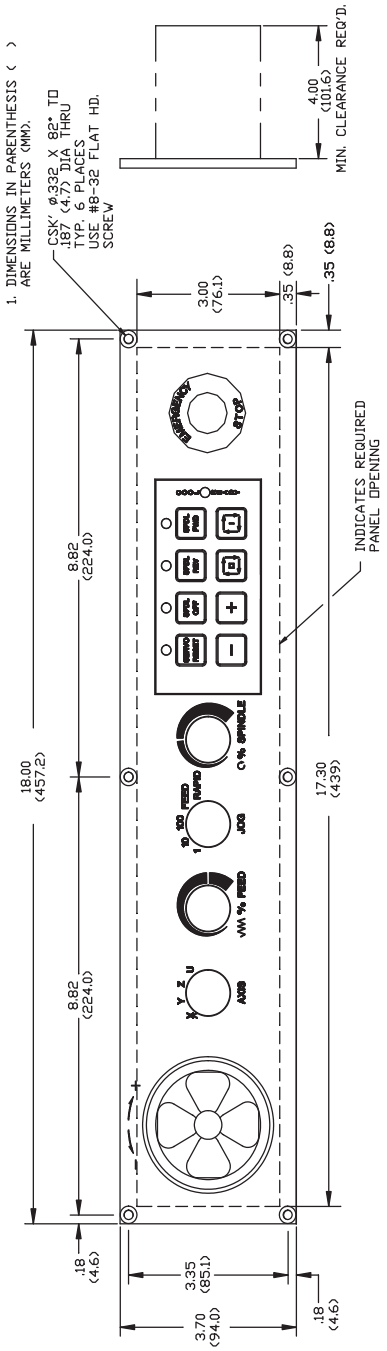


Figure 9-4, MP 6001M Manual Panel

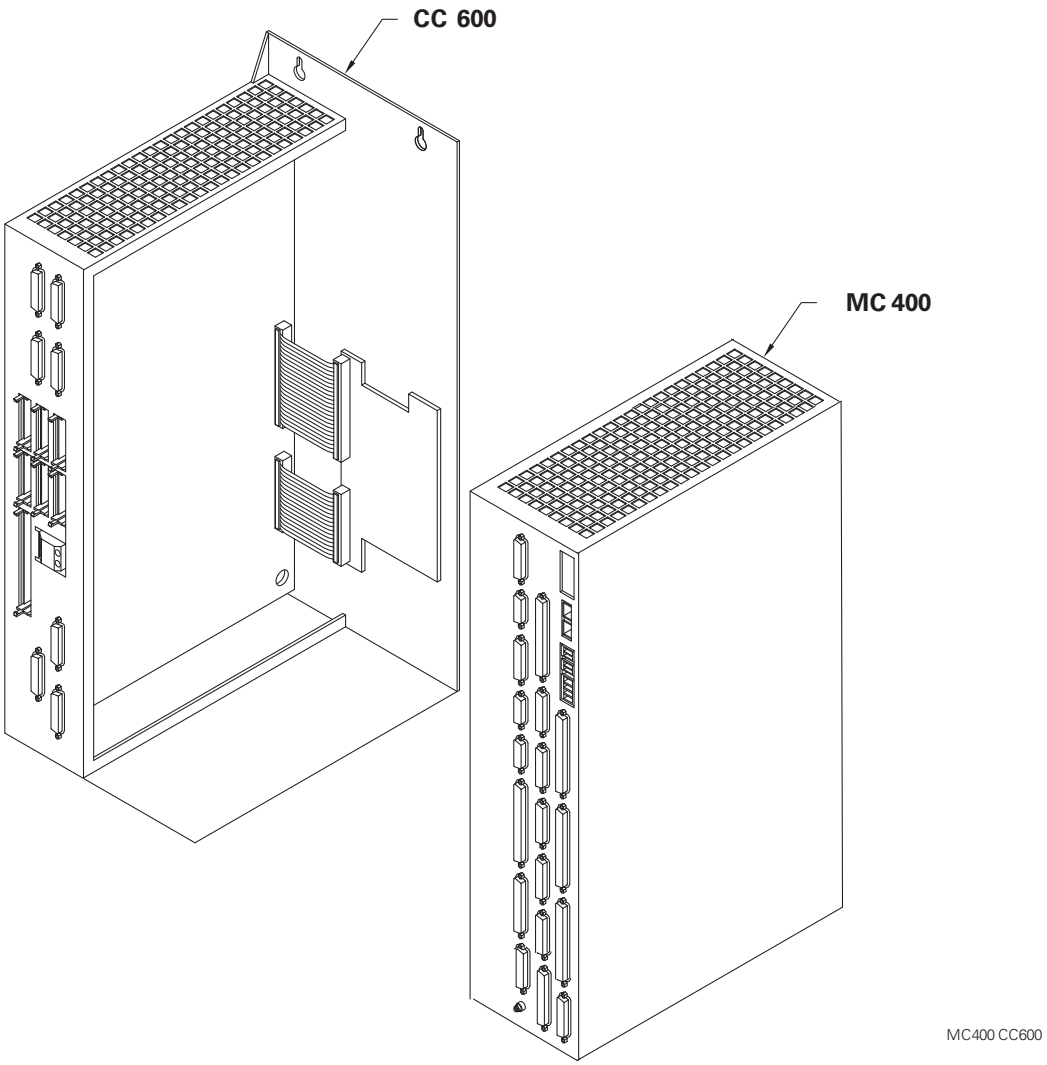


Figure 9-5, CC 600 and MC 400

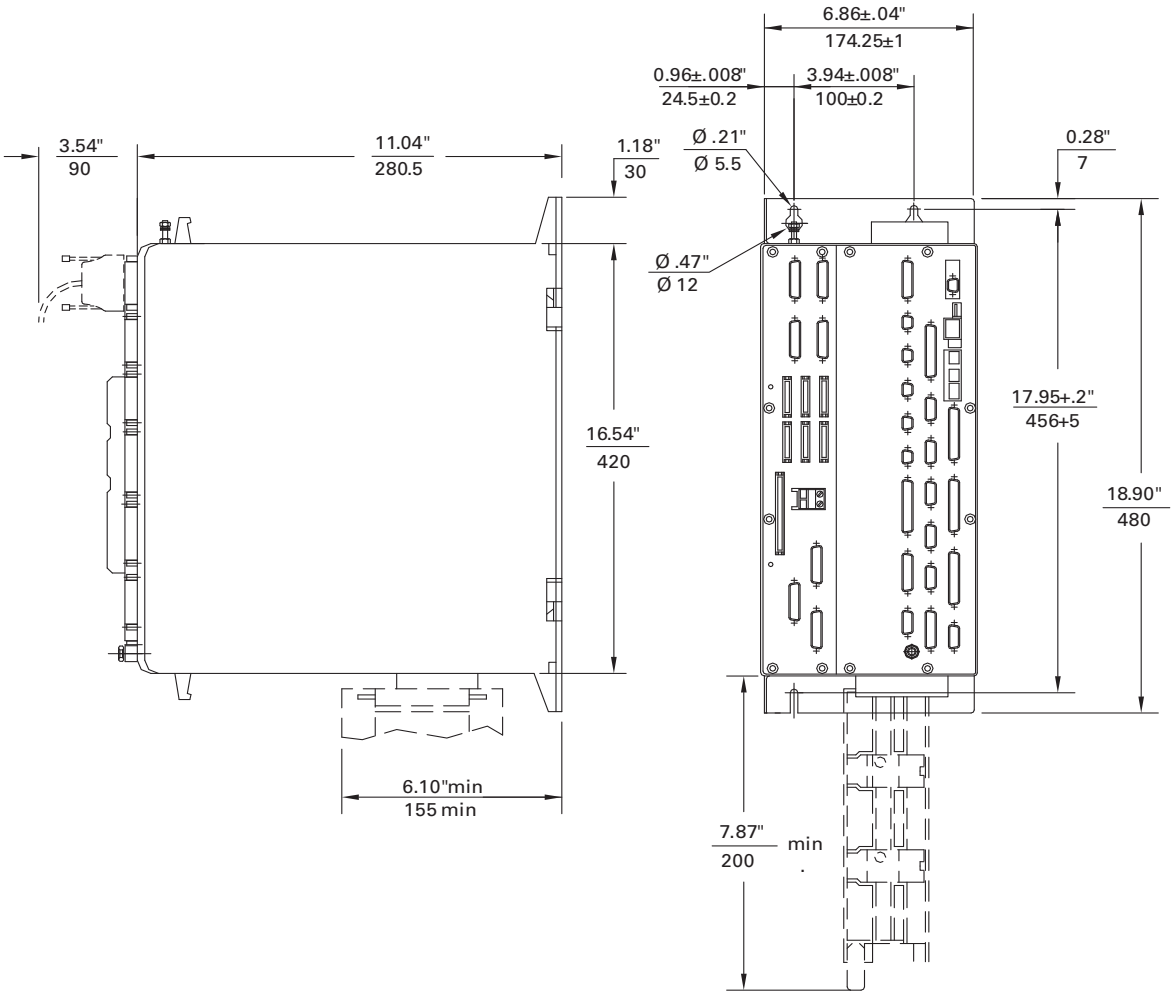


Figure 9-6, CC 600 and MC 400 Dimensions

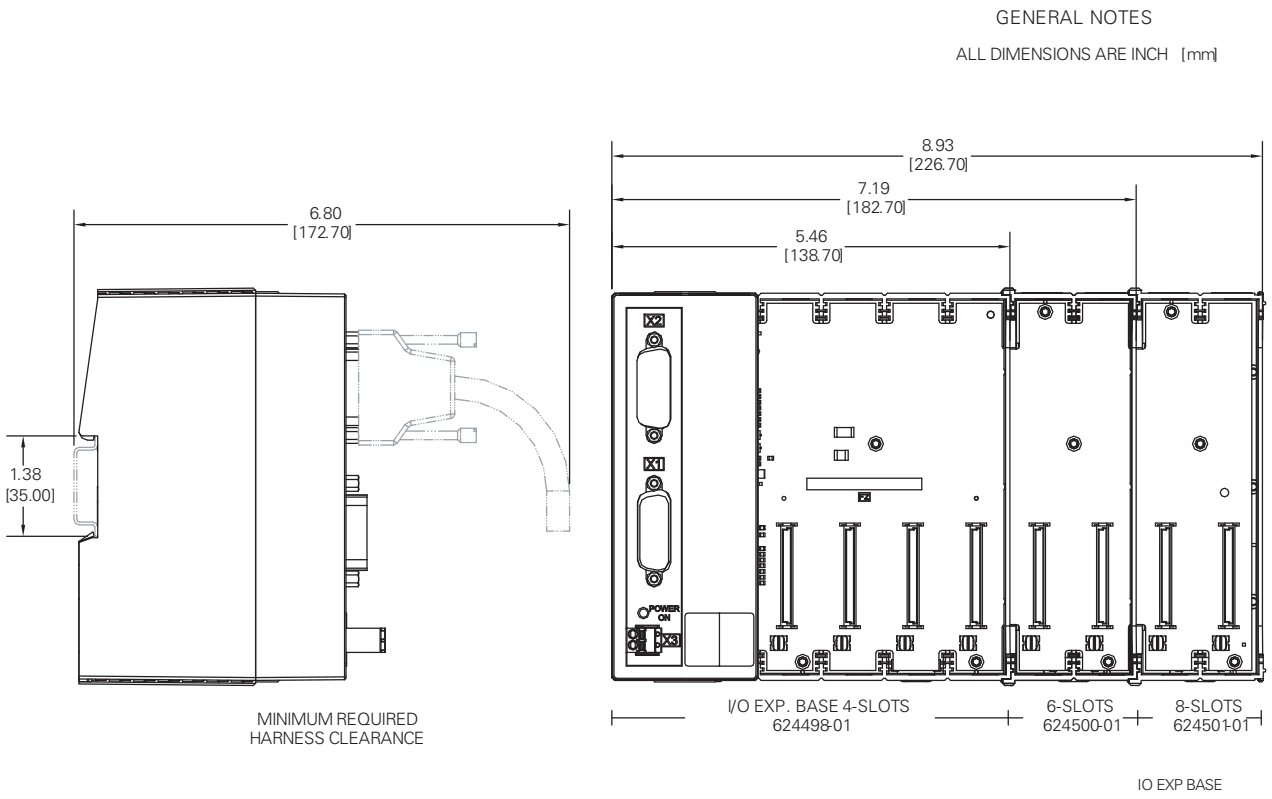
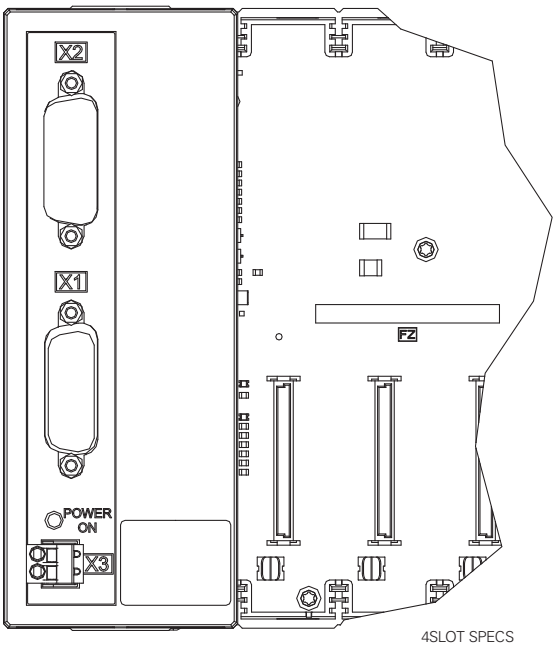


Figure 9-7, I/O EXP BASE MODULE: 4-SLOTS (P/N 624498-01, IEM 404), 6-SLOTS (P/N 624500-01, IEM 406), 8-SLOTS (P/N 624501-01, IEM 408)



Connector	Pin	Description
X1/X2	1	0V
	2	0V
	3	0V
	4	SERIAL IN 2
	5	ADDRESS 6
	6	INTERRUPT
	7	RESET
	8	WRITE EXTERN
	9	WRITE EXTERN
	10	ADDRESS 5
	11	ADDRESS 3
	12	ADDRESS 1
	13	
	14	
	15	
	16	PCB ID 2
	17	PCB ID 1
	18	ADDRESS 7
	19	SERIAL IN 1
	20	EM. STOP
	21	SERIAL OUT
	22	SERIAL OUT
	23	ADDRESS 4
	24	ADDRESS 2
	25	ADDRESS 0

Connector	Pin	Description
X3	1	+24 VDC
	2	0V

Figure 9-8, I/O EXP BASE MODULE 4-SLOTS (P/N 624498-01, IEB 404) Connector Description

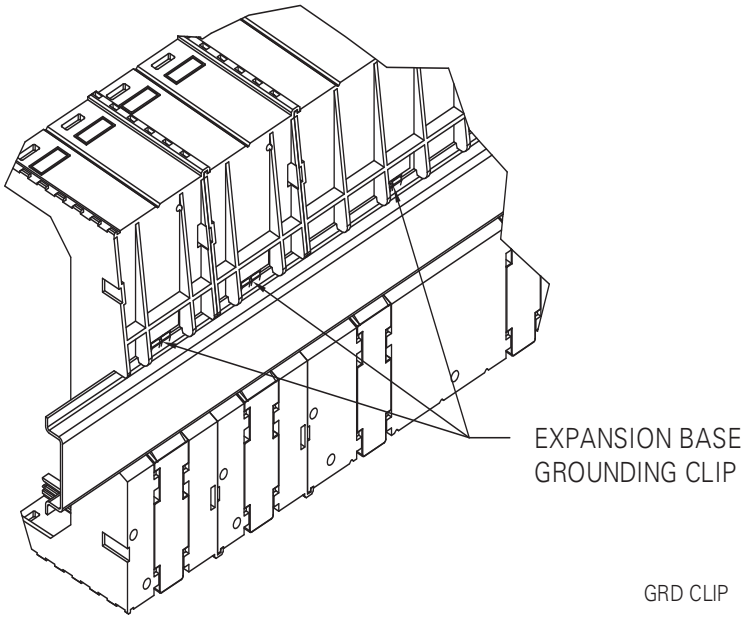


Figure 9-9, Expansion Base Grounding Clip

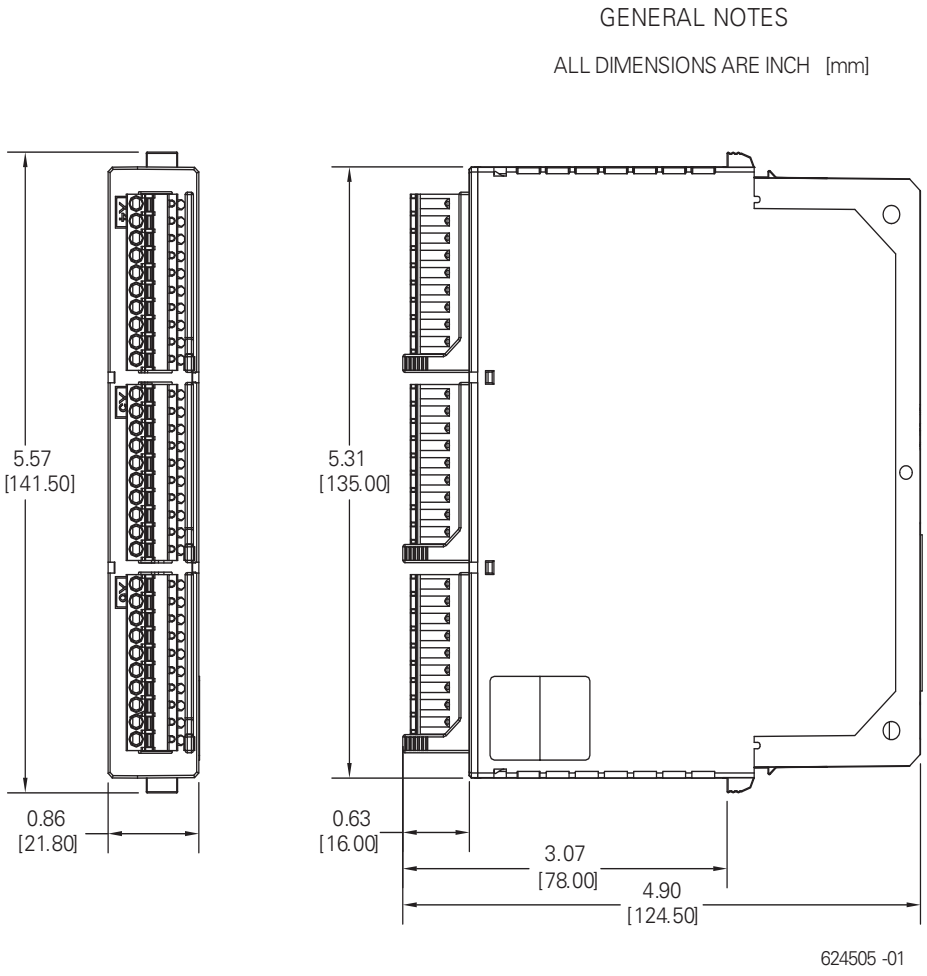
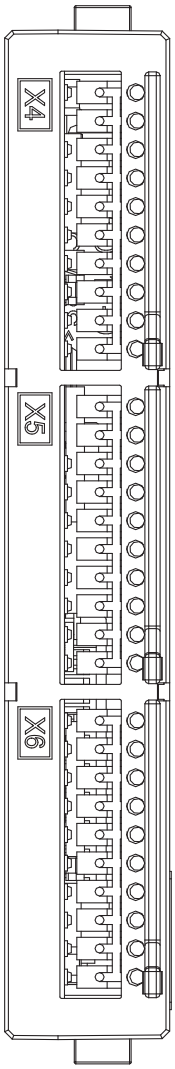


Figure 9-10, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) Dimensions



LED	MEANING
RED (X4-1)	ERROR (I/O SHORT CKT)
YEL	I/O STATUS
GRN (X6-9)	+24V (0 - 3)
GRN (X6-10)	+24V (4 - 7)

CONNECTOR	PIN	DESCRIPTION
X4	1	ERROR
	2	0V
	3	IN 0
	4	IN 1
	5	IN 2
	6	IN 3
	7	IN 4
	8	IN 5
	9	IN 6
	10	IN 7
X5	1	0V
	2	0V
	3	IN 8
	4	IN 9
	5	IN 10
	6	IN 11
	7	IN 12
	8	IN 13
	9	IN 14
	10	IN 15
X6	1	OUT 0
	2	OUT 1
	3	OUT 2
	4	OUT 3
	5	OUT 4
	6	OUT 5
	7	OUT 6
	8	OUT 7
	9	+24V (0 - 3)
	10	+24V (4 - 7)

IEM 16-8D

Figure 9-11, I/O MODULE, DIGITAL 16/8 (P/N 624505-01, IEM 16-8D) LEDs and Connectors

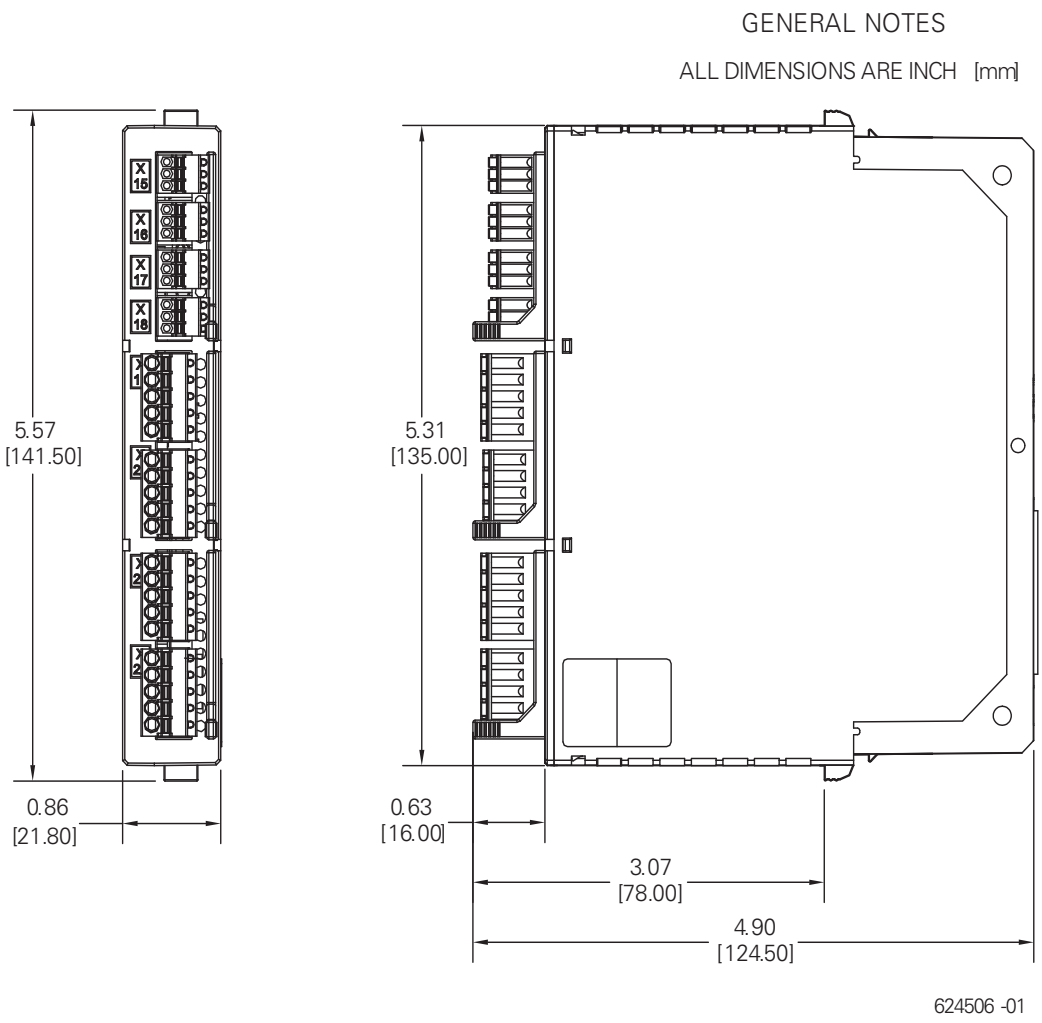
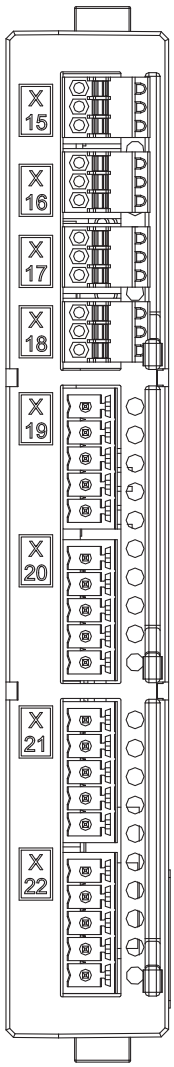


Figure 9-12, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Dimensions



CONNECTOR	PIN	DESCRIPTION
X15	1	± 10V INPUT
	2	
	3	
X16	1	± 10V INPUT
	2	
	3	
X17	1	± 10V INPUT
	2	
	3	
X18	1	± 10V INPUT
	2	
	3	
X19	1	PT 100 INPUT
	2	
	3	
	4	
	5	
X20	1	PT 100 INPUT
	2	
	3	
	4	
	5	
X21	1	PT 100 INPUT
	2	
	3	
	4	
	5	
X22	1	PT 100 INPUT
	2	
	3	
	4	
	5	

± 10V INPUT	
PIN	DESCRIPTION
1	±10V
2	0V
3	SHLD
PT100 INPUT	
PIN	DESCRIPTION
1	MEASURING CURRENT OUTPUT (I+)
2	MEASURING INPUT +
3	MEASURING INPUT -
4	MEASURING CURRENT INPUT (I-)
5	SHLD

IEM 4-4A

Figure 9-13, I/O MODULE, ANALOG 4/4 (P/N 624506-01, IEM 4-4A) Connectors

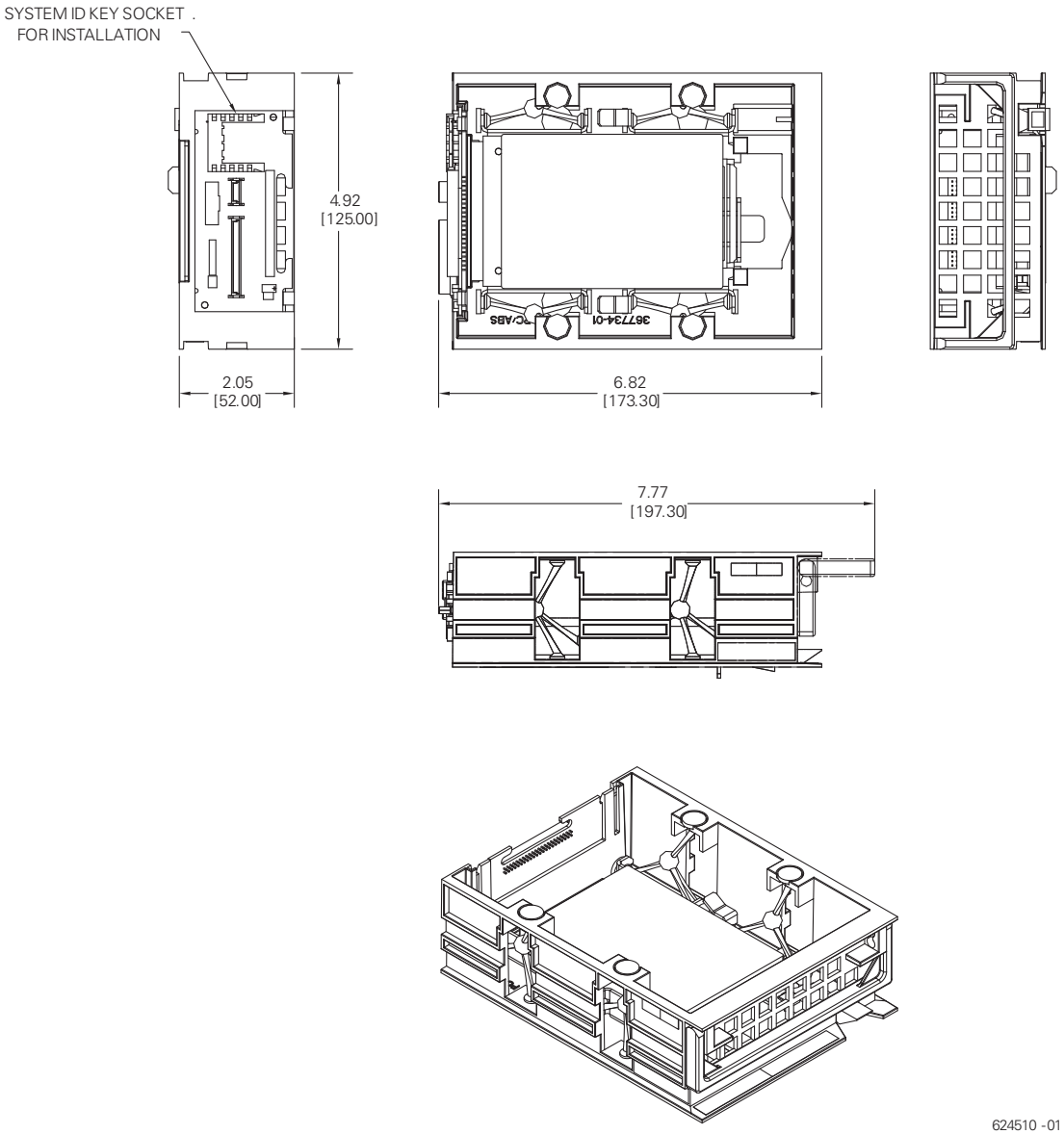


Figure 9-14, Hard Disk Drawer (P/N 574746-51, HDR) Dimensions

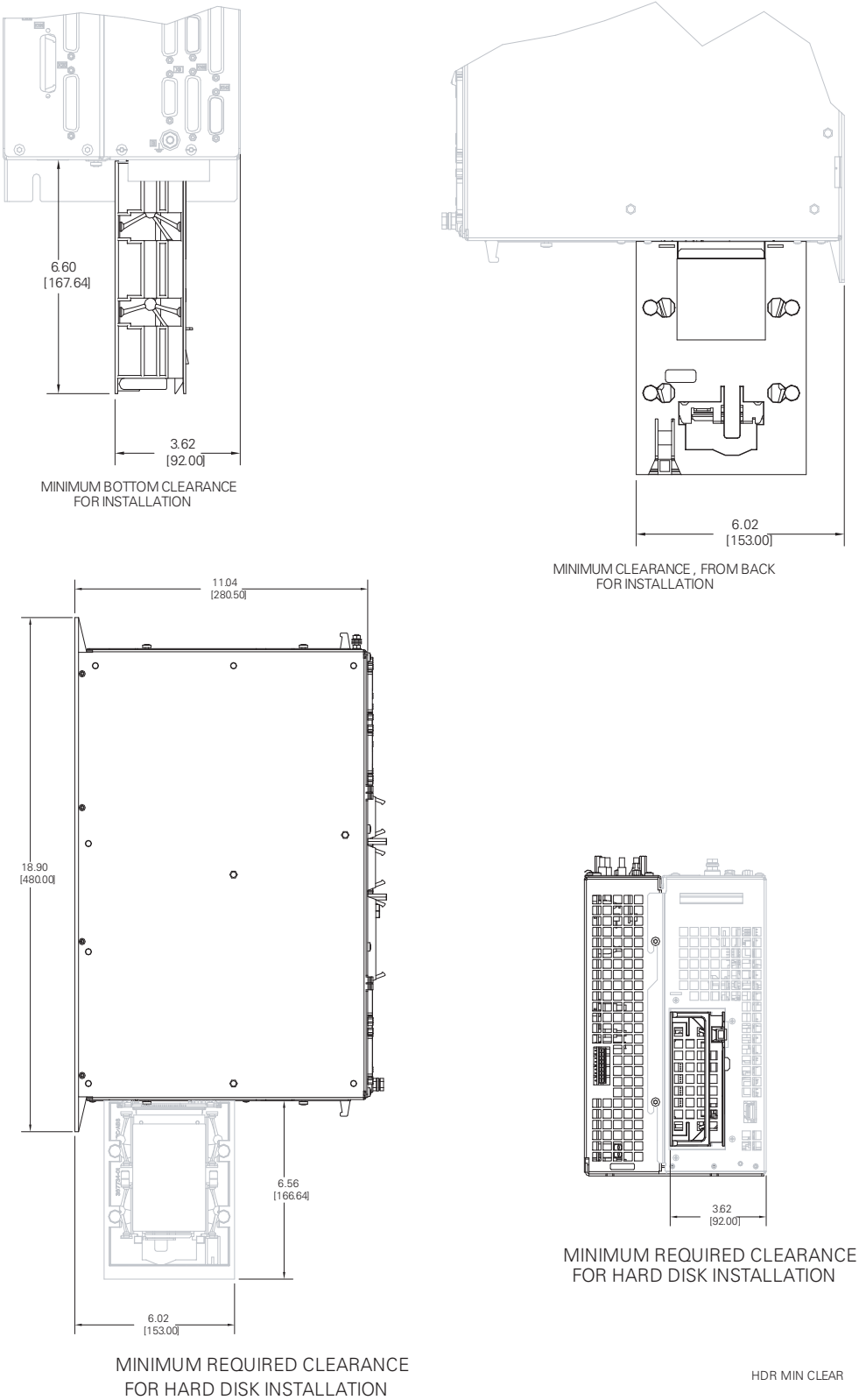
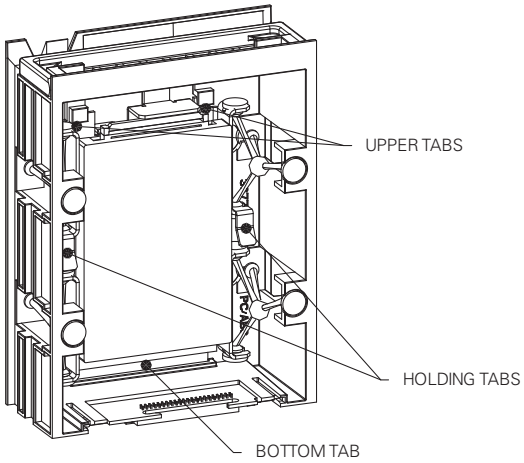
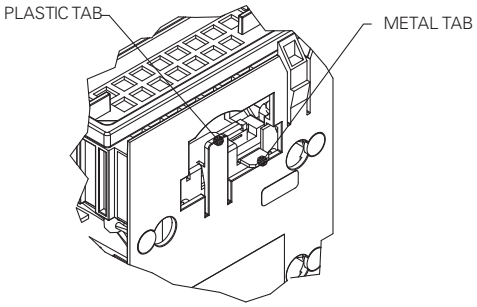


Figure 9-15, Hard Disk Drawer (P/N 574746-51, HDR) Minimum Clearances



LOCKING THE DRIVE

1. PUSH DRIVE IN , USE HOLDING TABS ; AND ALIGN METAL TAB ON BOTTOM SLOT.
2. INSERT TAB ON SLOT , BY PUSHING DOWN ON THE HOLDING TABS .
3. ALIGN METAL TABS ON TOP AND RELEASE THE HOLDING TABS , UNTIL IT LOCKS (CLICK) IN PLACE



UNLOCKING THE DRIVE

1. PULL ON PLASTIC TAB
2. PUSH DOWN ON METAL TAB AND RELASE

HDR LOCK

Figure 9-16, Hard Disk Drawer (P/N 574746-51, HDR) Locking/Unlocking the Drive

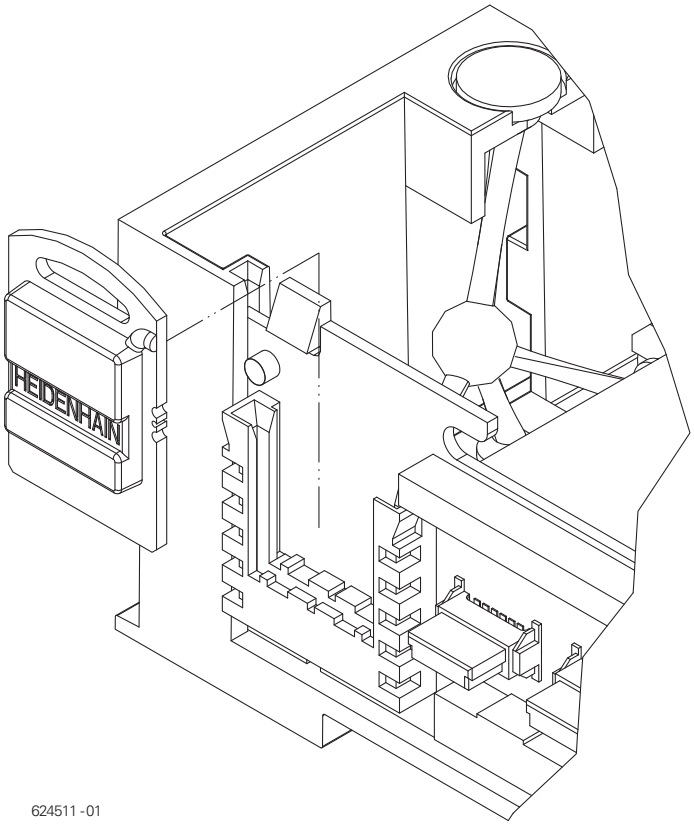
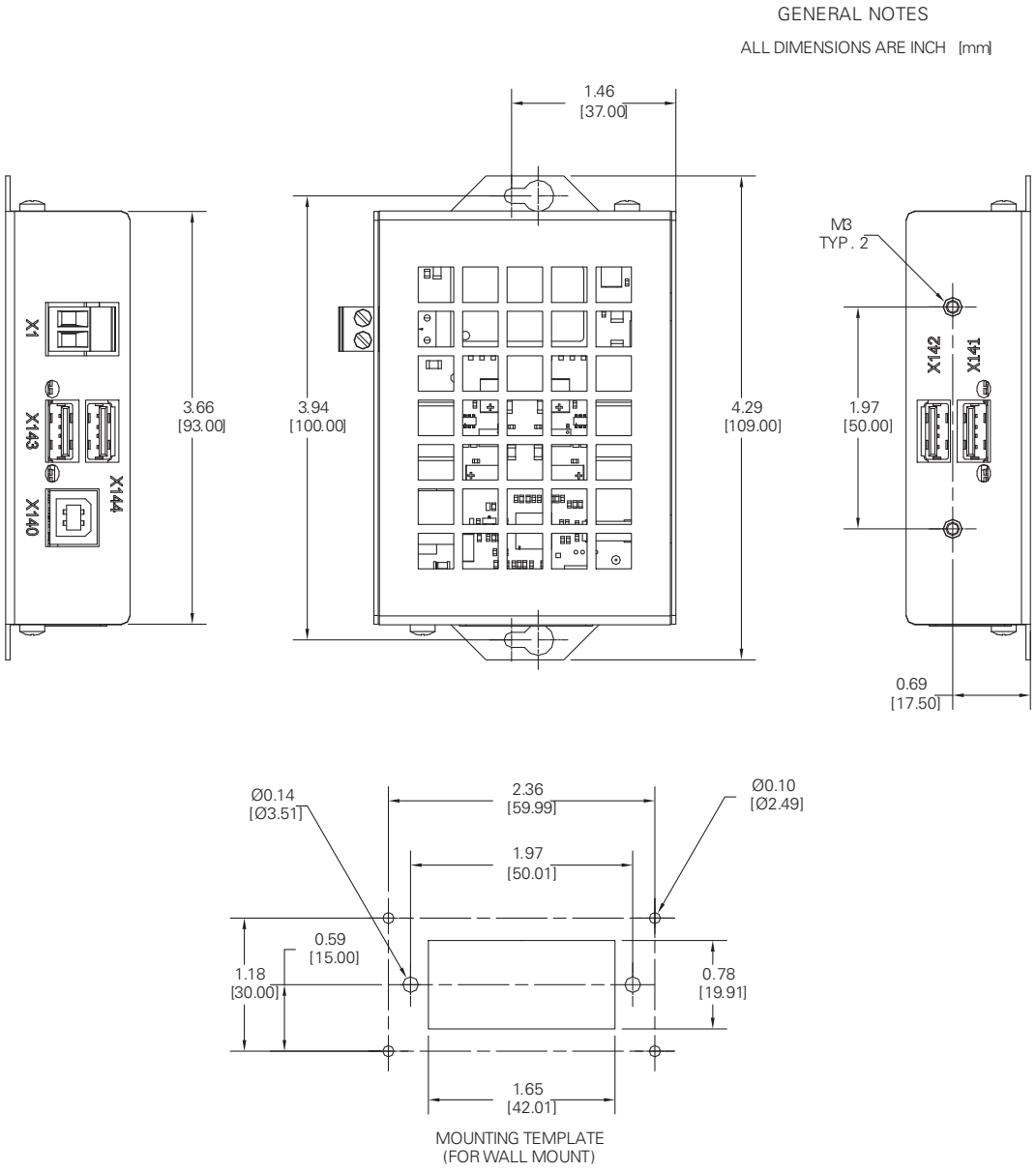


Figure 9-17, System ID Key (P/N 574744-51, SIK) Installation

See **Figure 9-14, Hard Disk Drawer (P/N 574746-51, HDR) Dimensions** for reference.



CONNECTOR	DESCRIPTION
X1	PWR SUPPLY +24V
X140	USB INPUT
X141	USB OUTPUT 1
X142	USB OUTPUT 2
X143	USB OUTPUT 3
X144	USB OUTPUT 4

USB HUB

Figure 9-18, USB Hub (P/N 624508-01) Dimensions

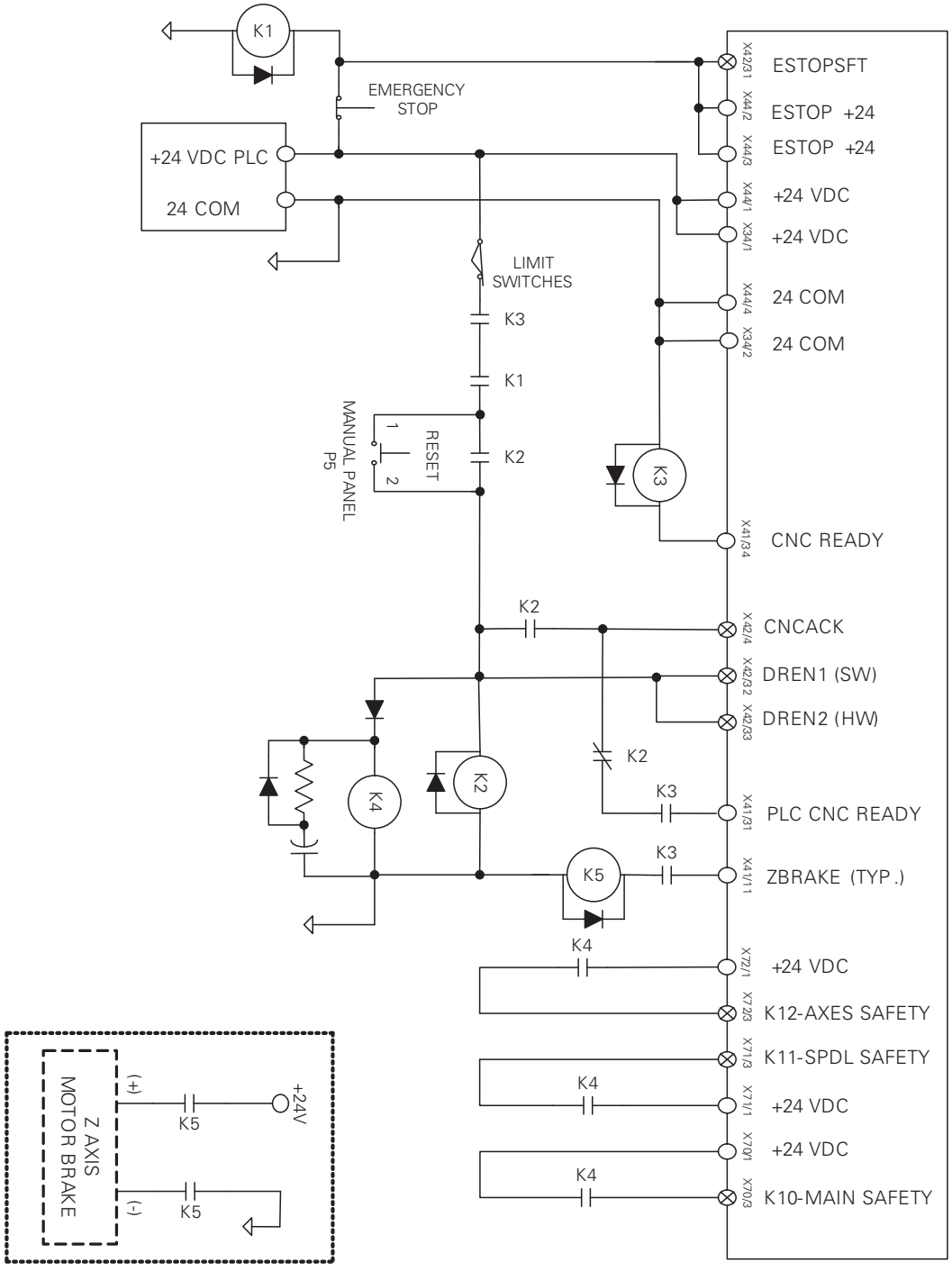


Figure 9-19, Basic Servo Turn On Circuit

NOTES:

Values of K4 relay off-delay circuit components: Capacitor - 2200 uF, Resistor- 1000 Ohm 1W

Reset signal is provided by relay on Manual Panel Interface board.

Z brake output as shown is typical. Functionality of Brake signal is determined by the Programmable Logic Controller (PLC) code.

+24 VDC provided by E-Stop at X41/1 services outputs O:00 through O:30.

K3 relay is energized when the drive enable signals DREN1, DREN2, and CNCACK are high. K5 (ZBRAKE) relay coil is to be energized typically from X41 pin 11 through a normally open contact on K3 relay to provide 24 volt power to the Z-axis brake. If the main safety loop is open [(K3) or ZBRAKE output (K5) is removed], Z-axis brake should engage (power is removed).

Basic Servo Turn On

+24 VDC and 24 Common are applied to Connector X44 and the Emergency Stop switch. The supply side +24 VDC of output drivers O:00 to O:30 is provided by the Emergency Stop switch at X44/1; so that, regardless of the state of the programmed output, devices connected to those outputs will be de-energized upon Emergency Stop. An Emergency Stop input required by the CNC software is also applied to X42/31.

The Emergency Stop switch provides +24 VDC to the “dead stop” limit switches of the system. If all limits are closed, +24 VDC is provided to normally open K1 and K3 relay contacts. The K1 relay is energized by 24 volts through the Emergency Stop button. If the CNC sees the Emergency Stop button input is present and there is no other fault, the CNC software will energize the CNC READY output which then energizes relay K3. CNC READY is energized when the motion control unit of the CNC is ready for operation and stays energized as long as the motion control unit is ready.

A normally open K1 and K3 relay contacts provide +24 VDC to a parallel circuit of a normally open K2 “holding” contact and the manual panel interface P5. P5 is shown as a switch for clarity, but is actually a relay.

When the RESET switch of the Manual Panel is pressed, it closes the relay contact on the interface board. This provides the +24 VDC signal to CNCACK X42/4 (CNC acknowledge), DREN1 X42/32, and DREN2 X42/33 (Drive Enable Software and Drive Enable Hardware), which is in turn seen by the PLC.

24 volts from the Emergency Stop button passes through the hard stop limit switches, the normally open contacts of K3 and K1, and the manual panel RESET switch to energize K2, which then closes the K2 “holding“ contact of step 3 and latches the CNCACK signal to X42/4.

Drive enable signals DREN1 and DREN2 are also latched at the same time as CNCACK. DREN1 (drive enable 1) is required by the CNCC software at X42/32. DREN2 (drive enable 2) is required by the hardware at X42/33.

Relay K4 is energized by the same 24 volts as K2. When K2 is de-energized, K3 de-energizes immediately, K4 de-energizes after a time delay determined by the RC network. K4 relay contacts are used to hold axis and spindle drives on long enough to allow dynamic braking after Emergency Stop.

The PLC code will develop the ZBRAKE signal based upon machine requirements. The K5 Z brake relay is energized via a normally open K3 contact. Thus, if the drives are de-energized for any reason, K5 will de-energize regardless of logic state of the ZBRAKE signal.

Spindle and Axis inverter drive outputs are enabled by applying +24 VDC to X71/3 and X72/3. These signals are applied via a normally open K4 contact. The K4 contact is delayed off to permit dynamic braking of the axis and spindle drives.

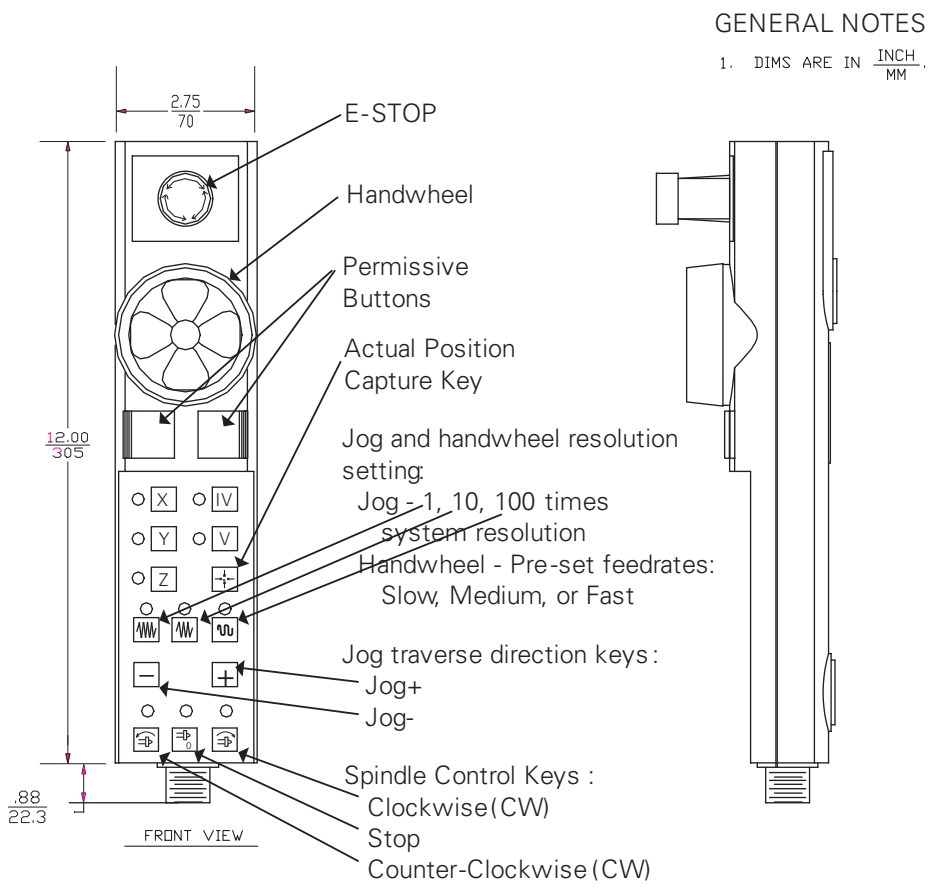


Figure 9-20, RM 500 Remote Handwheel, P/N 34000850

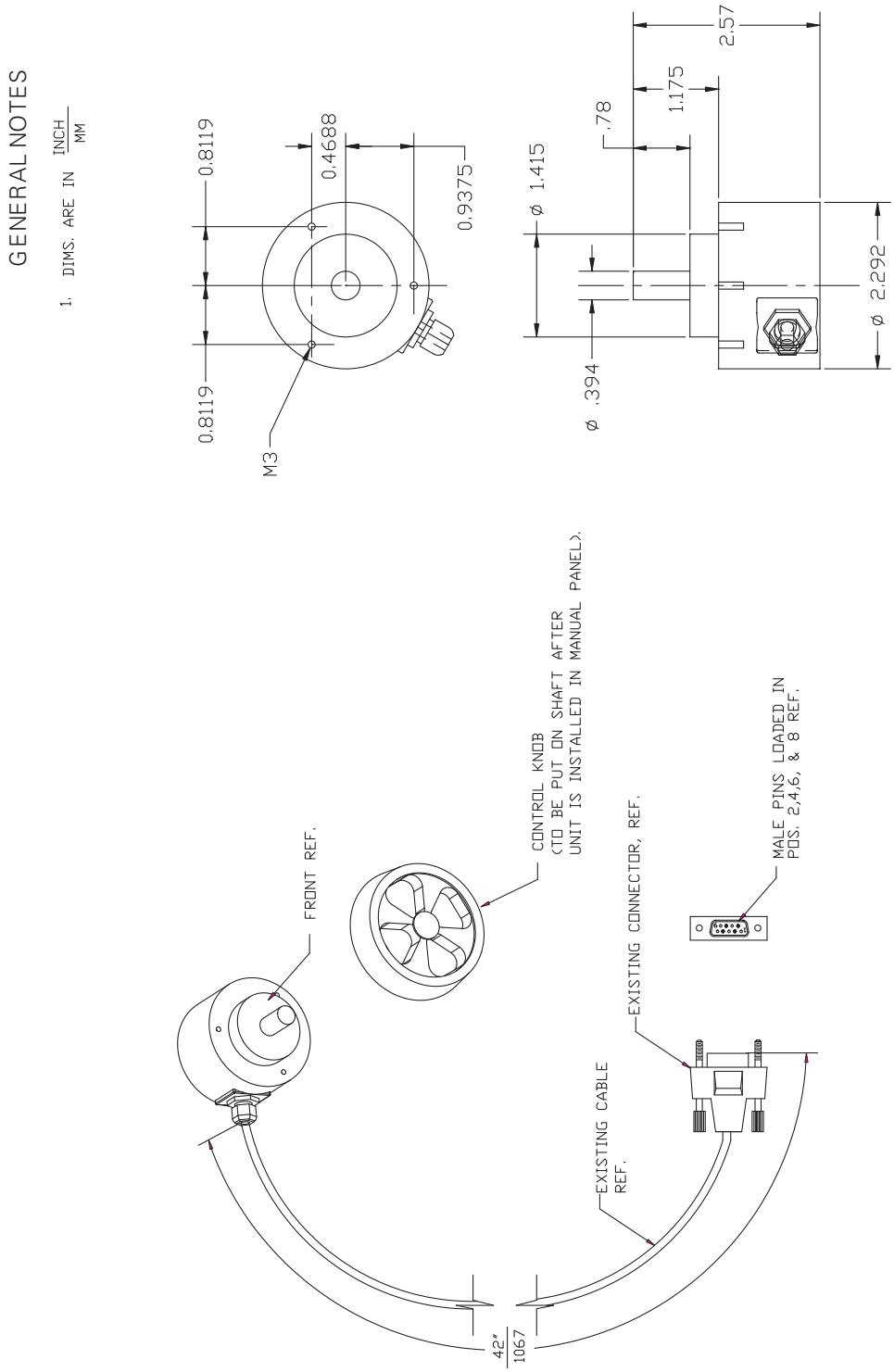
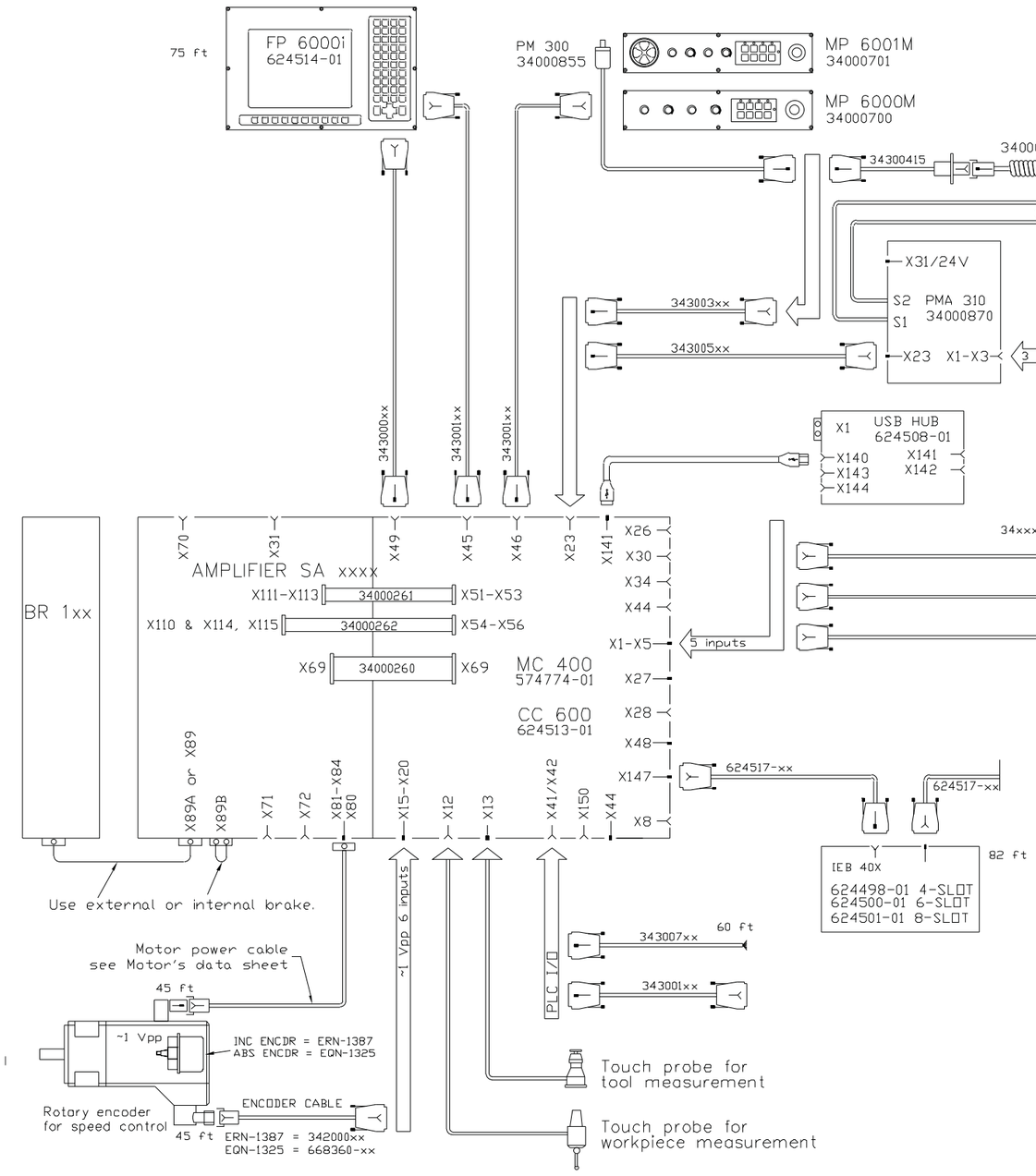


Figure 9-21, PM 300 Panel-Mounted Handwheel, P/N 34000855



GENERAL NOTES

REFER TO 6000i MANUALS FOR INSTALLATION AND SETUP DETAILS.

MC400 and CC600

Conn.	Description
X1-X5	Encoders for position
X8	Nominal value output, analog
X12	TS touch triger probe
X13	TT touch triger probe
X15-X20	Encoders for speed
X23	Handwheel input
X26	Ethernet data interface
X27	RS-232-C/V.24 data interface
X28	RS-422/V.11 data interface
X30	Reference signal for spindle
X34	24 V input for control-is-ready signal
X41	PLC output
X42	PLC input
X44	24 V input for PLC power supply
X45	Keyboard
X46	Manual panel
X48	PLC analog input
X49	Flat panel display
X51-X56	PWM output
X69	CNC Chassis power supply
X141	USB Interface
X147	IEB 40x Expansion for PLC
X150	Axis-specific drive release

Amplifier Connectors

Conn.	Description
X31	400VAC Input
X69	CNC Chassis power supply
X70	Power enable input
X71	Spindle power enable input
X72	Axes power enable input
X80	Spindle motor power
X81-X84	Axes motor power
X89A,X89	External braking resistor
X89B	Internal braking resistor
X110	Spindle PWM
X111-X115	Axes PWM

Console Connectors

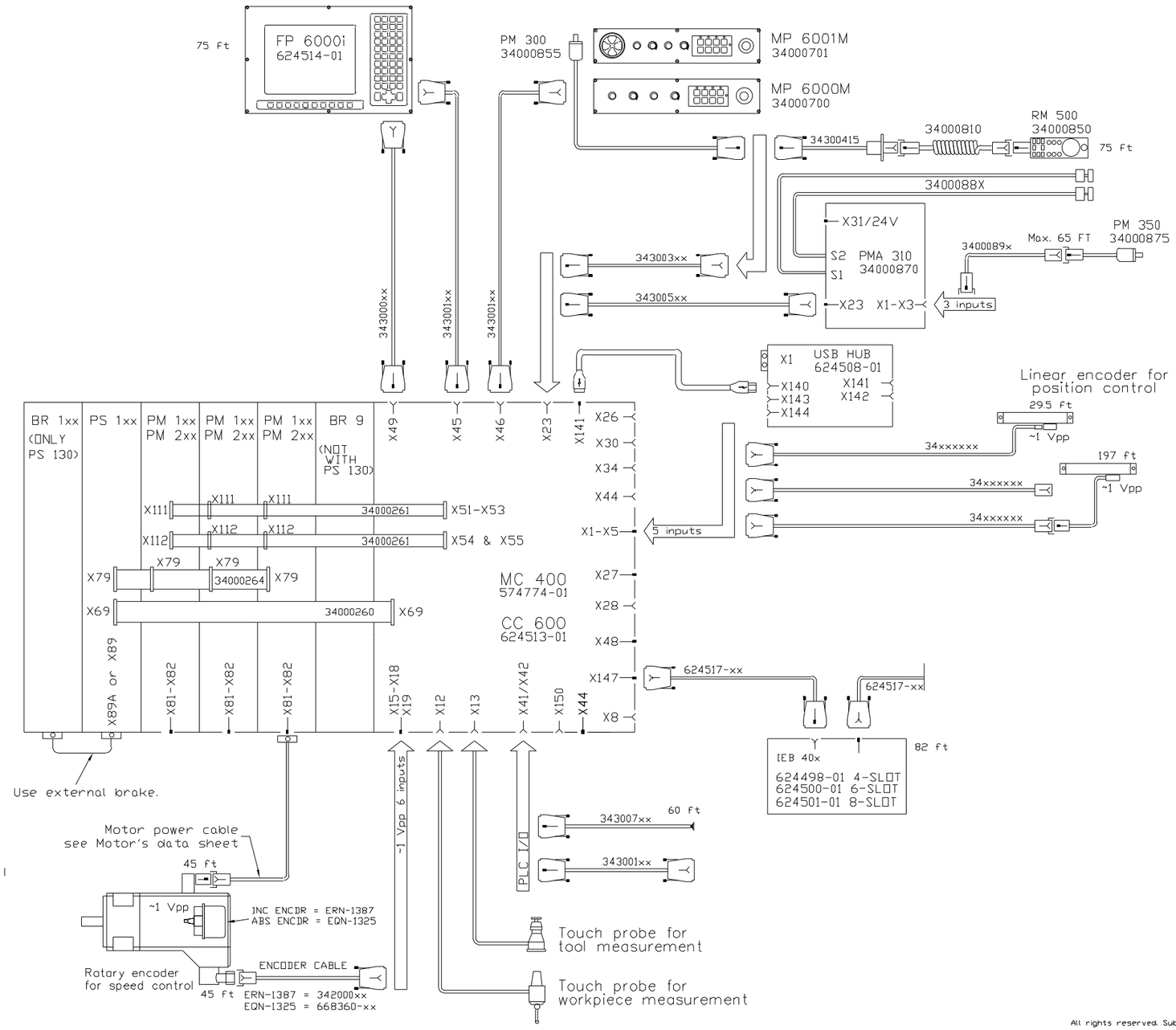
Conn.	Description
P1	Keyboard data (from X45)
X1	Flat panel display (from X49)
X2	+24 V, COM for flat panel

Manual Panel Connectors

Conn.	Description
P1	Manual panel data (from X46)
P5	Servo reset

D0009531B

Figure 9-22, Cable Overview



GENERAL NOTES

REFER TO 6000i MANUALS FOR INSTALLATION AND SETUP DETAILS.

MC400 and CC600

Conn.	Description
X1-X5	Encoders for position
X8	Nominal value output, analog
X12	TS touch triger probe
X13	TT touch triger probe
X15-X20	Encoders for speed
X23	Handwheel input
X26	Ethernet data interface
X27	RS-232-C/V.24 data interface
X28	RS-422/V.11 data interface
X30	Reference signal for spindle
X34	24 V input for control-is-ready signal
X41	PLC output
X42	PLC input
X44	24 V input for PLC power supply
X45	Keyboard
X46	Manual panel
X48	PLC analog input
X49	Flat panel display
X51-X56	PWM output
X69	CNC Chassis power supply
X141	USB Interface
X147	IEB 40x Epansion for PLC
X150	Axis-specific drive release

Power Supply

Conn.	Description
L1, L2, L3	400VAC Input
X69	CNC Chassis power supply
X70	Power enable input
X71	Spindle power enable input
X72	Axes power enable input
X79	Module power supply
X89A,X89	External braking resistor

Module Amplifier Connectors

Conn.	Description
X79	Module power
X81, X82	Axis motor power
X111	Axis PWM
X112	Axis PWM

Console Connectors

Conn.	Description
P1	Keyboard data (from X45)
X1	Flat panel display (from X49)
X2	+24 V, COM for flat panel

Manual Panel Connectors

Conn.	Description
P1	Manual panel data (from X46)
P5	Servo reset

D0009532B

Figure 9-23, Cable Overview, Modular

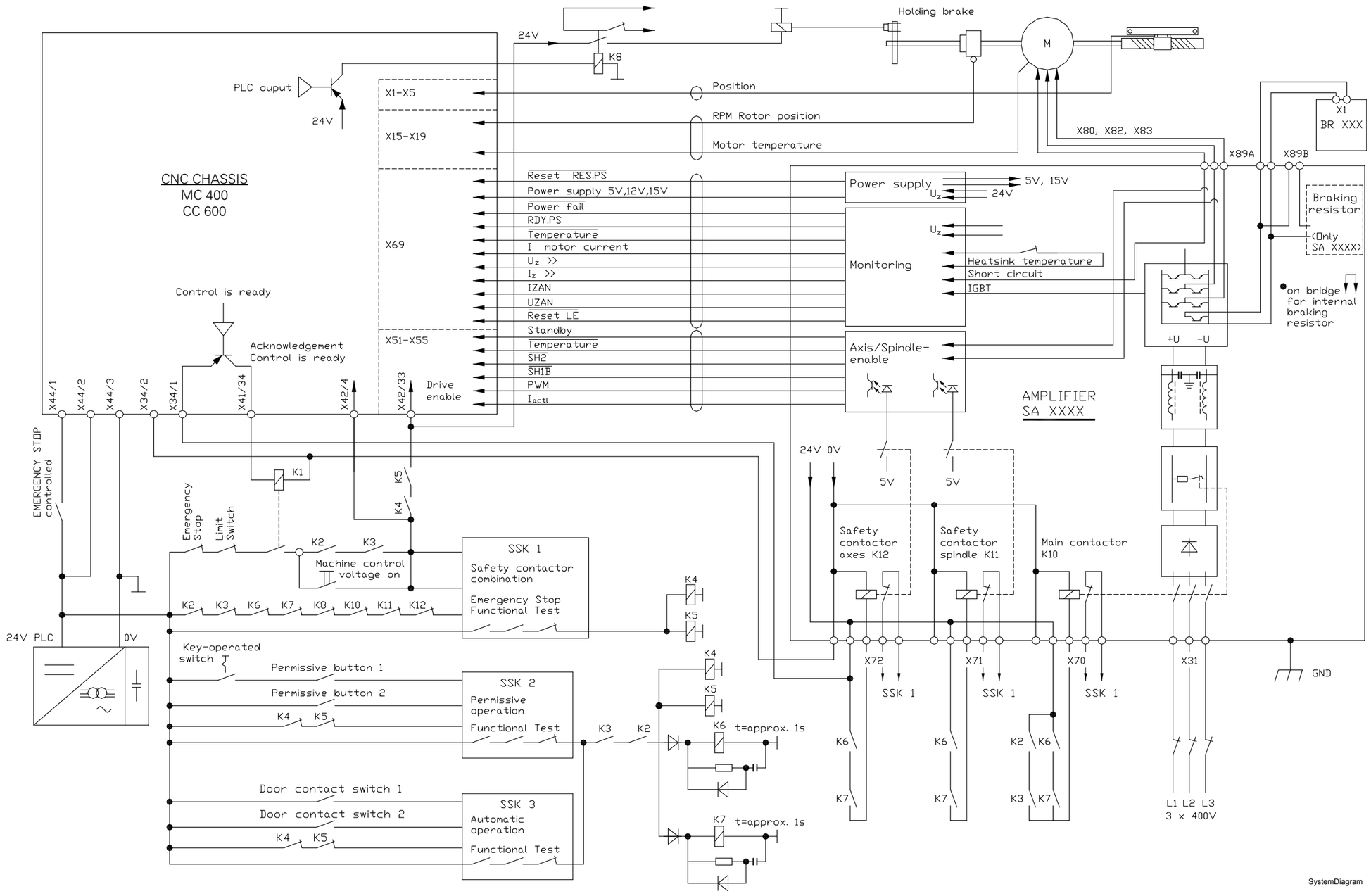


Figure 9-24, Basic System Diagram

Symbols

(SHIFT + F1) Msgs, description, 6-48
(SHIFT + F7) OSC, oscilloscope main screen, 5-170
(SHIFT + F5) PLC, PLC main menu, 7-7
.cfg extension, 3-1
.SCO extension (oscilloscope trace file), 5-179

Numerics

24 VDC switching input/outputs, 6-99
3-axis systems, configuration, 3-20
6000i
    overview, 1-3
    product designation, 1-2
6000i CNC User's Manual, P/N 627785-2X, referenced, 2-18, 2-55, 6-1, 6-115, 7-206

A

A/D Conv (F1), display analog-to-digital values, 6-183
acceleration
    description, 5-96
    determining, 6-135
    feedforward control, determining, 6-141
    feedforward control, for analog axes, 5-127
    values, filter, spindle, 5-153
acceleration and jerk, control loop, 5-96
access rights, description, 3-14
Act State (F1), current status of channel, 6-172
activate, axis, 5-19
activating, monitoring functions, 6-144
actual-to-nominal, value transfer, 5-122
adapting, control to machine structure, 5-2
ADD [ ] (+[ ]), 7-167
ADD STRING (+), 7-188
ADDITION (+), 7-153
adjusting, the servo amplifier, 6-128
alias function
    assigning to machining channel, 6-45
    configuration of, 6-46
    description, 6-45
alias strobe, description, 6-45
ALTER SYNONYM, SQL command, 7-72
ALTER TABLE, SQL command, 7-70
AM XXX, product designation, 1-2
ambient, temperature, 2-5
amplifier power module, positioning, illustration, 2-7
analog
    cable, voltage characteristics, 2-28
    compensation, reversal peaks, 5-128
    input, description, 2-28
    inputs for Pt 100 therminstors, description, 2-28
    inputs, description, 6-101
    nominal value output, 2-27
    offset, description, 5-132
    output, description, 5-35
    outputs, 6-103
    signal, analyze an individual, 5-177
    signals, integrated oscilloscope, 5-168
    signals, setup, oscilloscope, 5-171
    X8, output, pinout, 2-27
analog axes
    acceleration feedforward control, 5-127
    characteristic curve kink point, 5-126
    configuring, 5-35
    controller parameters, 5-125
    feedback control, formula, 5-126
    position loop, resolution, 5-133
    rapid traverse, 5-36
Analog Output (F5), display nominal commands, 6-173
analog-to-digital converter, displaying values, 6-183
analyze recording, oscilloscope, 5-176
analyze, individual analog signal, 5-177
AND (A), 7-141
AND [ ] (A[ ]), 7-165
AND NOT (AN), 7-143
AND NOT [ ] (AN[ ]), 7-166
API
    symbolic
        benefits, listed, 7-3
        description, 7-3
        programming, 7-5
API DATA (F3), display contents of symbolic markers, 7-9
ApiAxis, 7-5
ApiChn, 7-5
ApiGen, 7-5
ApiMarker.def, definition file, 7-4
ApiOmg, 7-5
ApiSpin, 7-5
APM 100A, product designation, 1-2
APM, defined, 2-7

application example, SQL commands, 7-72
arc, end-point tolerance, 6-16
ASCII editor, description, 7-119
ASSIGN (=), 7-133
ASSIGN BYTE (B=), 7-135
ASSIGN DOUBLE WORD (D=), 7-136
ASSIGN NOT (=N), 7-136
ASSIGN TWO'S COMPLEMENT (=–), 7-136
ASSIGN WORD (W=), 7-135
assigning
  alias functions, to machining channels, 6-45
  M functions, to machining channels, 6-27
  parameter blocks, 5-20
  S functions, to machining channels, 6-37
  T functions, to machining channels, 6-41
assignments, manual modes of operation, 6-26
asynchronous position compensation, OLM, 6-182
automatic, NC program start, 6-19
AUTOSTART, description, 6-19
Auxil (F5), define data to be logged and saved, 6-178
auxiliary group, OLM, 6-178
axes
  algebraic signs of the, 5-12
  analog, controller parameters, 5-125
  clamping, 5-121
  commissioning, 6-129
  configuration of, 5-11
  configuring, 5-2
  definition, machine structure, 5-3
  definition, spindles, 5-4
  in motion, monitoring functions, 5-145
  in position, monitoring functions, 5-144
  kinematics, special, 5-4
  linear V, V, W, properties of, 5-13
  logical, defining, 5-3
  machine parameters, listed, 3-104
  machining channels, 5-6
  moving, during program interruption, 6-22
  physical, description, 5-17
  programmable, description, 5-15
axis
  analog, feedback control, formula, 5-126
  designation & coordinates, 5-11
  group, PLC operands, listed, table, 4-12
  information, reading, 5-36
  motors, PWM connection, 2-22
  specific limit values, description, 5-103
  specific drive enable, description, 2-50
  without a separate drive motor, 5-16
axis error compensation
  backlash compensation, 5-58
  compensation of thermal expansion, 5-70
  description, 5-56
  linear axis, error compensation, description, 5-60
  nonlinear axis error compensation, 5-62
  nonlinear, rotary axis, 5-65, 6-85
axlist.cfg, example, 3-22

B

B/W/D/S (F2), show lists, 7-14
backlash compensation, description, 5-58
backup, configuration parameters, 3-18
basic
  components, listed, 1-1
  data, supported at system start-up, 3-27
  system diagram, referenced, 5-149
  system diagram, illustration, 9-29
baud rate, 8-13
BC (BIT CLEAR), 7-174
BIT CLEAR (BC), 7-174
BIT SET (BS), 7-173
block scan, start block search, 6-23
Blum laser touch probe
  description, 2-15
  illustration, 2-15
  M550, Laser Probe On, 2-15
  M551, Laser Probe Off, 2-15
  M552, Enable 1 On, 2-15
  M553, Enable 1 Off, 2-15
  M554, Enable 2 On, 2-15
  M555, Enable 2 Off, 2-15
  M556, Blow Nozzle On, 2-15
  M557, Blow Nozzle Off, 2-15
  PLC additions, 2-16
  plc.cfg, example, 2-17
  plc_oem.cfg, example, 2-17
  touchpro.scr, example, 2-17
BNF notation, syntax, 7-64
BS (BIT SET), 7-173

C

cable overview
  descriptions, 2-10

illustration, 9-25
modular, illustration, 9-27
cables
343007XX, description, 2-31
P/N 624517-XX, lengths, available, 2-44
rotary encoder, lengths, 2-12
calibAndToolMeasurementRPM, 2-19
descriptionCALL MODULE (CM), 7-181
CALL MODULE IF FALSE (CMF), 7-182
CALL MODULE IF TRUE (CMT), 7-181
call submit (SUBM), 7-198
CAN, canceling a submit program, 7-199
canceling a submit program (CAN), 7-199
cardan angles, orientation, 5-81
cascade control, description, 5-95
case branch, description, 7-211
CASE, indexed module call, 7-211
CC 600
connections, illustration, 2-9
description, 1-3
dimensions, drawing, 9-7
drawing, 9-6
product designation, 1-2
cfg extension, 3-1
change
keyname, 3-10
machine parameters, reaction, 3-15
Change Key (F4), description, 3-10
Change ParSet (F5), switching the parameter block of an axis, 6-165
channel specific settings, 6-16
channels, machine parameters, listed, 3-87
characteristic curve, kink point, analog axes, 5-126
Check AxPar (F3), writes axis parameters, 6-180
check, the counting direction, 6-130
Chnls (F2), channels, 6-168
circuit overview, basic, description, 2-10
clamped axes, monitoring, 5-140
clamping, the axes, 5-121
Clear PeakLag (F6), delete the following error, 6-165
Clear RefOK (F7), delete the reference point, 6-166
CNC
Computer Numerical Control, defined, 1-1
console, hardware, connections, 2-60
defined, 1-1
manual panel, connections, 2-61
power supply & control signals, 2-23
X2, I/O module, pinout, 2-45
X41, PLC output, 2-35
X42, PLC input, 2-31
X45, keyboard, pinout, 2-42
CNC and inverter, space requirements, referenced, 2-56
column description, tables, 7-41
COM1, 2-46
COM2, 2-47
command options
description, 7-67
FOR NOTIFICATION, 7-67
FOR UPDATE, description, 7-67
ORDER BY, 7-67
WHERE, 7-67
comments, user, description, 3-12
commissioning
adjusting the servo amplifier, 6-128
axes, 6-129
description, 6-126
preparation, 6-126
temporary input values, 6-127
communications protocol, serial interface, 8-14
compensation
of reversal peaks, 5-130
reversal peaks, analog axes, 5-128
thermal expansion, 5-70
value tables
assigning to axes, 5-65
description, 5-64
values, entering, 5-64
Compile (F2), program to become active, 7-17
Compile PlcMain (F2), compile PLC program, 7-17
component, designations, 1-2
components, basic, 1-1
Computer Numerical Control, see CNC
Config Data (F6)
displays Machine Parameter screen, 3-5
screen illustration, 3-7
Config Struct (F5), screen illustration, 3-13
configfiles.cfg
5-axis system, description, 3-19
description, 3-19
multiple variants, 3-20

configuration
data, allocation of, 3-19
data, parameters, screen capture, 2-18
data, parameters, screen capture, additional parameters, 2-20
machine kinematics, 5-72
of, alias function, 6-46
of, filters, before position control loop, 5-108
of, interfaces, 8-10
of, rotary axes A, B, C, properties, 5-13
of, S function, 6-38
of, T function, 6-42
setup screen, illustration, 3-3
configuration editor
calling, 3-3
data records, displaying/editing, 3-26
description, 3-1
entering & editing machine parameters, 3-7
files, managing, 3-13
files, sort content, 3-13
M function, settings, 6-28
machine parameter screen, 3-5
MP change list, 3-28
parameters, backup, 3-18
syntax error, remove, 3-17
table description, 7-40
tree view, 3-14
update rules, 3-16
version, reset update, 3-17
configuration of axes
algebraic signs, of the axes, 5-12
axis designation & coordinates, 5-11
coordinates, standard, 5-14
description, 5-11
linear axes U, V, W, properties of, 5-13
physical axes, 5-17
programmable, 5-15
configure, colors, oscilloscope display, 5-181
connection, overview, 2-9
console
description, 2-60
dimensions, drawing, 9-2
FP 6000i, description, 2-60
hardware, connections to CNC, 2-60
space requirements, referenced, 2-8, 2-60
constants field (KF), description, 7-207
context elements, syntax, 7-64
contour smoothing, description, 5-100
control characters, serial interface, 8-9
control in operation, 6-23
control loop
acceleration and jerk, 5-96
description, 5-95
distance, 5-97
filter before position, 5-108
function of filters, before position control loop, 5-109
geometry filter, 5-98
interpolation, 5-107
look-ahead, 5-100
position controller, 5-113
switching parameter blocks, 5-133
control of events, 7-202
control operation
in machining channel, 6-16
in operating mode group, 6-14
control signals, CNC & power supply, 2-23
control to machine structure, 5-2
control-is-ready signal, 24V power supply, 2-24
controller output, limiting, position controller, 5-116
controller parameters, for analog axes, 5-125
controlling, axes, by PLC, 5-44
conversational language, description, 6-11
cooling
description, 2-5
provide adequate, 2-5
cooperative multitasking
control of events, 7-202
description, 7-201
starting a parallel process (SPAWN), 7-201
coordinates, standard, configuration of axes, 5-14
Copy (F3), machine parameters, 3-9
COPY TABLE, SQL command, 7-70
counter, description, 7-28
counting direction, check, 6-130
CREATE SYNONYM, SQL command, 7-71
CREATE TABLE, SQL command, 7-69
current, per output, nominal operating, 2-25
cursor, second, activate/deactivate, 5-178
cursor, shifting the, 5-177

D

Data Backup (F2), screen illustration, 3-18

data bits, serial interface, 8-14
data interfaces
  Ethernet, 2-48, 8-2
  introduction, 8-1
  listed, 2-46
  RS-232, 2-46
  RS-232-C/V.24, 8-6
  RS-422, 2-47
  USB, 8-3
data organization, description, 7-32
data records, displaying/editing, 3-26
data transfer
  baud rate, 8-13
  by PLC, 8-19
  check, handshaking, 8-16
  machine parameters–>PLC, 7-117
  NC program–>PLC, 7-95
  NC programs–>PLC, 7-97
  NC–>NC program, 7-105
  NC–>PLC, 7-94
  PLC–>NC, 7-94
data, configuration, allocation of, 3-19
Debug Print (F1), OLM open selection list, 6-178
decimal
  character, 6-4
  separator, 6-4
DECREMENT (DEC), 7-158
defining, logical axes, 5-3
DefPoint systems, 5-74
degrees of protection, description, 2-2
delete, machine parameters, 3-9
delete columns, from existing table, 7-47
determining
  acceleration, 6-135
  acceleration feedforward control, 6-141
  kv factor, 6-137
  the jerk, 6-139
diagnosis, of the PL, 6-95
diameterOfSpindleProbeGauge, description, 2-21
diameterOfToolProbeGauge, description, 2-19
differences between positions, switch-on and shutdown, 5-140
digital signals, setup, oscilloscope, 5-173
direction, traverse reference marks, 5-86
Discard Chgs (F7), MP change list, 3-28
disclaimer, iii
display
  change, oscilloscope, 5-177
  oscilloscope, configure colors, 5-181
  picture distortion, 2-8
  position and status, 6-1
  signal, influence, 5-178
display and operation
  description, 6-1
  unit of measurement, 6-3
distance
  coded, reference marks, encoders, 5-27
  control loop, 5-97
  description, 5-96
DIVISION (/), 7-156
DIVISION [ ] (/[ ]), 7-168
drawings
  basic system diagram, illustration, 9-29
  cable overview, illustration, 9-25
  cable overview, modular, illustration, 9-27
  CC 600, 9-6
  CC 600, dimensions, 9-7
  console, dimensions, 9-2
  expansion base grounding clip, 9-10
  I/O exp. base module, 4-, 6-, 8-slots, 9-8
  I/O exp. base module, 4-slots, 9-9
  listed, 9-1
  MC 400, 9-6
  MC 400, dimensions, 9-7
  MC, CC, and inverter, dimensions, 9-3
  MP 6000M manual panel, dimensions, 9-4
  MP 6001M manual panel, dimensions, 9-5
  PM 300 panel-mounted handwheel, illustration, 9-24
  RM 500 panel-mounted handwheel, illustration, 9-23
  servo turn on circuit, basic, 9-20
drive controller enable
  description, 2-50
  X150, pinout, 2-50
drive monitor, utilization, read, 5-147
drives, monitoring, 5-136
DROP SYNONYM, SQL command, 7-72
DROP TABLE, SQL command, 7-70

E

Edit (F1), edit file, 7-18
editing, finish, 3-12

editor
changes, rules for entries, 3-21
configuration, description, 3-1
effective data, description, 3-28
electromagnetic compatibility, description, 2-2
electronic handwheel, description, 6-71
elements, of pocket table, 6-117, 6-118
elements, of tool table, 6-117
EMERGENCY STOP monitoring, description, 5-149
emergency stop, test, flowcharts, 5-150
encoders
connections
description, 5-28
position encoder, input, 5-28
position encoder, signal, 5-28
description, 5-23
EnDat interface, 5-83
monitoring, 5-32
monitoring, with EnDat interface, 5-33
position control, 2-11
reference marks, distance-coded, 5-27
signal period, description, 5-25
speed, control, 2-12
traverse direction, defining, 5-31
type of, 5-23
end of indexed module call (ENDC), 7-211
END OF MODULE (EM), 7-183
END OF MODULE IF FALSE (EMF), 7-183
END OF MODULE IF TRUE (EMT), 7-183
EnDat interface
description, 5-83
encoders, reference marks, 5-83
monitoring encoders, 5-33
ENDC, end of indexed module call, 7-211
entering, compensation values, 5-64
environmental conditions
heating and cooling, 2-5
humidity, 2-6
mechanical vibration, 2-6
EQUAL TO (==), 7-159
EQUAL TO [ ] (==[ ]), 7-169
EQUAL TO STRING (==), 7-190
error
behavior, machining channel, 5-7
compensation
activate, 5-66
linear axis, 5-60
nonlinear axis, 5-62
deleting, 6-49
handling, listed, 6-50
log file, current & previous file, 6-52
message
description, 6-47
information provided, 6-49
OLM, generating, 6-181
PLC, 6-56
OLM, frequent causes, 6-187
stack, 7-92
status, description, 6-25
table, PLC, 6-56
text file
PET table, 6-57
structure, 6-58
window, description, 6-48
Error Log (F1), description, 6-50
errors, reaction to, 6-15
Ethernet
data interface, 2-48
description, 8-2
X26, interface connection, 8-2
eulerian angles, orientation, 5-81
EXCLUSIVE OR (XO), 7-149
EXCLUSIVE OR [ ] (X0[ ]), 7-166
EXCLUSIVE OR NOT (XON), 7-151
EXCLUSIVE OR NOT [ ] (X0N[ ]), 7-166
Exit (F10), finish editing, 3-12
expansion base grounding clip, illustration, 9-10
eXtensible Markup Language, see XML
EXTERN statement, 7-214

F

F1 (A/D Conv), display analog-to-digital values, 6-183
F1 (Act State), current status of channel, 6-172
F1 (Debug Print), OLM open selection list, 6-178
F1 (Edit), edit file, 7-18
F1 (Error Log), description, 6-50
F1 (HW-Ports I), display current status of hardware ports, 6-175
F1 (Ipo Data), interpolator module, 6-168
F1 (Login Ipo), soft keys listed, 6-149
F1 (M/I/O/T/C), shows lists, 7-13
F1 (M3), select spindle commands, 6-166
F1 (Plc Nom Data), nominal commands of the PLC, 6-154

F1 (Plc Nom State), nominal status of the axes requested by PLC, 6-161
F1 (Q-Trace On-Off), define queues to be traced, 6-186
F1 (Sort Content), description, 3-13
F10 (Exit), finish editing, 3-12
F2 (B/W/D/S), show lists, 7-14
F2 (Chnls), channels, 6-168
F2 (Compile PlcMain), compile PLC program, 7-17
F2 (Compile), program to become active, 7-17
F2 (Data Backup), screen illustration, 3-18
F2 (Insert), machine parameters, 3-9
F2 (Ipo Act Data), description, 6-154
F2 (Ipo Act State 1), actual status 1 of the axes, 6-161
F2 (Key Log), screen illustration, 6-54
F2 (Log Ed SimIpo), soft keys listed, 6-149
F2 (Offset Data), offset interface, 6-171
F2 (Syntax Errors), description, 3-17
F3 (API DATA), display contents of symbolic markers, 7-9
F3 (CheckAxPar), writes axis parameters, 6-180
F3 (Copy), machine parameters, 3-9
F3 (Find), description, 3-11
F3 (HW), OLM hardware, 6-173
F3 (Input B/W/D), show lists, 7-15
F3 (Ipo Act State 2), actual status 2 of the axes, 6-163
F3 (Load Subfile), activating MP subfiles, 3-23
F3 (Log File), description, 6-49
F3 (Login Plc), soft keys listed, 6-149
F3 (Plc Ipo Data), description, 6-156
F3 (PlcTrc On-Off), traces PLC modules called, 6-184
F3 (Symbol List), list box of PLC operands, 7-11
F3 (Update Rules), screen illustration, 3-16
F4 (Change Key), description, 3-10
F4 (Help), screen illustration, 3-8
F4 (Log Sim SimIpo), soft keys listed, 6-149
F4 (Output B/W/D, shows lists, 7-16
F4 (PlcTrc Save), saves PLC modules called, 6-184
F4 (Reset Version), screen illustration, 3-17
F4 (Select Record), screen illustration, 3-26
F4 (Set Error), define OLM error number & error class, 6-181
F4 (Set PosCorr), generate asynchronous position compensation, 6-182
F4 (Spindle), description, 6-154
F4 (S-RAM), static RAM of the IPO, 6-173
F4 (Watch List), displays states of PLC operands, 7-10
F5 (Analog Output), display nominal commands, 6-173
F5 (Auxil), define data to be logged and saved, 6-178
F5 (Change ParSet), switching the parameter block of an axis, 6-165
F5 (Config Struct), screen illustration, 3-13
F5 (Offset Data), interpolator & PLC-IPO data, 6-159
F5 (Tables), display tables, 7-12
F5 (Tree View), screen illustration, 3-14
F6 (Clear PeakLag), delete the following error, 6-165
F6 (Config Data), displays Machine Parameter screen, 3-5
F6 (Config Data), screen illustration, 3-7
F6 (GAL Data), display internal registers of the counter function, 6-174
F6 (Info View), description, 3-12
F6 (Save Chgs), description, 3-10
F6 (Secure Area), MP change list, 3-28
F7 (Clear RefOK), delete the reference point, 6-166
F7 (Discard Chgs), MP change list, 3-28
F7 (Filter), screen illustration, 6-53
F7 (More Cmds)
  configuration editor, 3-4
  from Config Data (F6), 3-6
F7 (Trace), screen illustration, 6-150
factory default, preconfigured kinematics, 5-74
fast PLC inputs, description, 7-30
feed rate limitation, description, 5-118
feed rate values in PLC operands, 5-118
feedback control
  description, 5-113
  velocity feedforward control, 5-116
  with following error, 5-114
feed-rate enable, 5-123
feedrate override, description, 6-91

file
extension, table specify, 7-38
manager, create new table, 7-45
files
managing, configuration, 3-13
parameter, setup, 3-21
Filter (F7), screen illustration, 6-53
filter
acceleration values, spindle, 5-153
before position control loop, 5-108, 5-124
function, before position control loop, 5-109
find
block number, 6-23
NC program, 6-23
Find (F3)
description, 3-11
key log file entry, 6-55
screen illustration, 6-52
flat panel
CNC connections, 2-60
power connections, 2-60
X49, display, pinout, 2-38
FN17, SYSWRITE, 7-97
FN18, SYSREAD, 7-105
FN19, PLC =, 7-95
FN20, WAIT FOR, 7-118
FN29, PLC =, 7-95
following error, delete, Clear PeakLag (F6), 6-165
following error, feedback control, 5-114
FOR NOTIFICATION, command option, description, 7-67
FOR UPDATE, command option, description, 7-67
formula, analog axis feedback control, 5-126
FP 6000i
console, description, 2-60
product designation, 1-2
fundamentals, integrated oscilloscope, 5-168

G

GAL Data (F6), display internal registers of the counter function blocks, 6-174
gear shifting, spindles, 5-167
general data group, PLC operands, listed, table, 4-6
general information
connection components, 1-1
mounting & electrical installation, 2-1
geometry filter, description, 5-98
GLOBAL statement, 7-214
GREATER THAN (>), 7-161
GREATER THAN [ ] (>[ ]), 7-170
GREATER THAN OR EQUAL TO (>=), 7-163
GREATER THAN OR EQUAL TO [ ] (>=[ ]), 7-170
GREATER THAN OR EQUAL TO STRING (>=), 7-191
GREATER THAN STRING (>), 7-190
gridlines, hide/show, 5-175

H

handling, HDR hard disk and SIK, 2-3
handshaking
data transfer check, 8-16
hardware, description, 8-17
software, description, 8-18
handwheel
assigning to an axis, 6-81
connection, 6-77
description, 6-71
initialization, 6-73
input, 2-52
locking, 6-72
manual axis-direction keys, 6-75
parameters, 6-71
PM 300, illustration, 9-24
PM 300, panel-mounted, 2-56
PM 350, panel-mounted, 2-57
position encoder input, 6-77
pulses, evaluation, 6-74, 6-79
RM 500, illustration, 9-23
RM 500, remote, 2-53
serial, description, 6-73
threshold sensitivity, 6-72
traverse revolution, 6-80
X23, 2-52
hanging axes, monitoring, 5-140
hard disk drawer, P/N 574746-51
dimensions, 9-15
lock/unlock the drive, 9-17
minimum clearances, 9-16
Hard Disk dRrawer, see HDR
hard drive
installing and removing, 2-8
shipping, 2-8
shipping brace, 2-8
to lock & unlock, 2-8

unlocking/locking, 2-3
hardware group
analog output, data, 6-173
description, 6-173
hardware handshaking, description, 8-17
HDR
description, 1-3
dimensions, 9-15
handling, 2-3
Hard Disk Drawer. defined, 1-3
installng/removing, 2-4
lock/unlock the drive, 9-17
minimum clearances, 9-16
shipping brace, 2-3
heating, description, 2-5
Help (F4), screen illustration, 3-8
humidity, description, 2-6
HW (F3), OLM hardware, 6-173
HW-Ports I (F1), display current status of hardware ports, 6-175
hysteresis, static friction, 6-144

I

I/O base module, P/N summary, 2-51
I/O expansion base module, P/N summary, 2-51
I/O module
analog 4/4, illustration, 9-13
analog 4/4, LEDs and connectors, illustration, 9-14
digital 16-8, illustration, 9-11
digital 16-8, LEDs and connectors, illustration, 9-12
P/N summary, 2-51
space requirements, referenced, 2-44
X147, connection to MC 400, pinout, 2-43
X2, connection to CNC, pinout, 2-45
icons, object tree, defined, 3-7
IEB 404
24V power supply, 2-24
connector description, 9-9
I/O exp. base module, 4-slots, illustration, 9-8
power supply, 2-25
product designation, 1-2
X3, power supply for logic circuit, pinout, 2-25
X4–X5, PLC inputs, 2-33
X6, PLC outputs, pinout, 2-37
IEB 406
I/O exp. base module, 6-slots, illustration, 9-8
product designation, 1-2
IEB 408
I/O exp. base module, 8-slots, illustration, 9-8
product designation, 1-2
IEM 16-8D
dimensions, 9-11, 9-12
input/output module, 2-25
LEDs, description, 2-51
output, addresses, 2-34
output, signals, 2-34
product designation, 1-2
switching, inputs 24 VDC (PLC), 2-30
X4, I/O module, pinout, 2-33
X5, I/O module, pinout, 2-34
X6, PLC outputs on IEB 404, pinout, 2-37
IEM 4-4A
connectors, 9-14
dimensions, 9-13
product designation, 1-2
IIF...ELSE...ENDI, 7-209
INCREMENT (INC), 7-158
incremental, jog positioning, 6-104
INDEX register (X register), 7-184
INDEX register, commands, 7-185
indexed module call (CASE), 7-211
indexed tools, description, 6-125
individual, analog signal, analyze, 5-177
Info View (F6), description, 3-12
input
position and speed, 2-11
X1, X2, X3 for PM 350 handwheel, 2-58
X23, PM 300 handwheel, 2-52
X23, RM 500 handwheel, 2-52
input B/W/D (F3), show lists, 7-15
Insert (F2), machine parameters, 3-9
insert additional columns, in existing table, 7-46
INSERT, SQL command, 7-69
installing
hard drive, 2-8
HDR, 2-4
SIK, 2-4
integrated oscilloscope
analyze recording, 5-176
description, 5-168

recording, prepare, 5-170
signals, record, 5-174
interface ports, configuring, 8-11
interference, likely sources, 2-2
internal ADC, interrogate, 5-146
interpolation, control loop, 5-107
interrogate PLC operands in the NC program, 7-118
interrupting, NC program, 6-21
introduction, 1-1
inverter, positioning, illustration, 2-7
Ipo Act Data (F2), description, 6-154
Ipo Act State 1 (F2), actual status 1 of the axes, 6-161
Ipo Act State 2 (F3), actual status 2 of the axes, 6-163
Ipo Data (F1), interpolator module, 6-168

J

jerk, description, 5-96
jerk, determining, 6-139
jog, incremental positioning, 6-104
JUMP (JP), 7-179
JUMP IF FALSE (JPF), 7-180
JUMP IF TRUE (JPT), 7-180

K

Key Log (F2), screen illustration, 6-54
keyboard and display, 6-147
keyboard, X45, pinout, 2-42
keyname, changing, 3-10
keypad, CNC connections, 2-60
keystroke simulation
control keyboard, 6-65
description, 6-65
machine operating panel, 6-70
keysynonym, machine parameters, description, 3-131
kinematics
configuration, 5-72
machine, description, 5-72
machining channel, 5-7
properties of axes, 5-21
special axes, definition, 5-4
kink point, characteristic curve, analog axes, 5-126
kv factor, determining, 6-137
kv factor, feed rate, and following error, interrelationship, 5-115

L

LABEL (LBL), 7-183
language, define, 6-12
LED
defined, 2-48
IEM 16-8D, description, 2-51
X26, Ethernet interface, 2-48
LEDs
I/O module, analog 4/4, 9-14
I/O module, digital 16-8, 9-12
left stop, defined, 2-57
LESS THAN (<), 7-160
LESS THAN [ ] (<[ ]), 7-170
LESS THAN OR EQUAL TO (<=), 7-162
LESS THAN OR EQUAL TO [ ] (<=[ ]), 7-170
LESS THAN OR EQUAL TO STRING (<=), 7-191
LESS THAN STRING (<), 7-190
light emitting diode, See LED
limit values, machine parameters, 3-9
linear axes U, V, W, properties of, 5-13
linear axis, error compensation, 5-60
linear encoder, pinouts, 2-11
linking files
description, 7-212
EXTERN statement, 7-214
GLOBAL statement, 7-214
USES statement, 7-213
LOAD (L), 7-126
LOAD BYTE (LB), 7-131
LOAD DOUBLE WORD (LD), 7-132
LOAD DOUBLE WORK (LD), 7-132
LOAD MINUS (L–), 7-130
LOAD NOT (LN), 7-128
LOAD NOT command, word processing, 7-128
LOAD STRING (L), 7-187
Load Subfile (F3), activating MP subfiles, 3-23
LOAD TWO'S COMPLEMENT (L–), 7-130
LOAD WORD (LW), 7-132
loaded subfile, attributes, edited, 3-27
Log Ed SimIpo (F2), soft keys listed, 6-149
log file
description, 6-47
entering data in, 6-63
saving, 6-55
Log File (F3), description, 6-49
Log File, soft keys description, 6-49
Log Sim SimIpo (F4), soft keys listed, 6-149

Login Ipo (F1), soft keys listed, 6-149
Login Plc (F3), soft keys listed, 6-149
look-ahead
  axis-specific limit values, 5-103
  contour smoothing, 5-100
  description, 5-100
  path-specific limit values, 5-101
  tolerance for corners and arcs, 5-105
  tolerance for rotary axes, 5-106
lubrication pulse, description, 5-43
LSV2, baud rate, defining, 8-12

M

M550, Laser Probe On, 2-15
M551, Laser Probe Off, 2-15
M552, Enable 1 On, 2-15
M553, Enable 1 Off, 2-15
M554, Enable 2 On, 2-15
M555, Enable 2 Off, 2-15
M556, Blow Nozzle On, 2-15
M557, Blow Nozzle Off, 2-15
M function
  calling, NC macro, 6-31
  configuration of, 6-28
  execution, 6-30
  influencing the execution, 6-33
  listed, 6-34
  overview of the 6000i, 6-32
  status of, 6-34
  transfer & acknowledgment, 6-30
  transfer of, 6-28
M functions, description, 6-27
M strobe, description, 6-27
M/I/O/T/C (F1), shows lists, 7-13
M06, program stop, *see* M6
M19, trip dog position, 5-160
M3 (F1), select spindle commands, 6-166
M6, program stop, 6-34
machine datum
  defined, 5-83
  description, 5-93
machine integration, description, 6-1
machine kinematics
  configuration, 5-72
  description, 5-72
  example, illustrations, 5-76
  machine parameters, overview, 5-72
  preconfigured, factory default, 5-74
  transformations
    tool side, 5-77
    using direction vectors, defining, 5-80
    workpiece side, 5-78
    with angles, definition, 5-81
    with vectors, definition, 5-81
machine operating panel, description, 6-70
Machine Parameter, from Config Data (F6), screen illustration, 3-5, 3-6
machine parameters
  axes, listed, 3-104
  channels, listed, 3-87
  coping, 3-9
  entering & editing, 3-9
  inserting, 3-9
  keysynonym, description, 3-131
  limit values, 3-9
  Module 9430, change numeric value, 3-31
  Module 9431, read numeric value, 3-32
  Module 9432, change string value, 3-33
  Module 9433, read string value, 3-34
  overview, 3-35, 3-131
  reaction to change, description, 3-15
  read or change, 3-31
  system, listed, 3-35
  to delete, 3-9
machine structure
  adapting, 5-2
  axes, definition, 5-3
  description, 5-2
machining channels
  assigning
    alias functions to, 6-45
    M functions, 6-27
    S functions to, 6-37
    T functions to, 6-41
  axes, of, 5-6
  configuring, 5-5
  error behavior, 5-7
  group, PLC operands, listed, table, 4-9
  kinematics, of, 5-7
  moving to restore position, 5-10
  NC channel, type of, 5-5
  reference marks, traversing, 5-9
  saving Q/QS parameters, 5-8
  start/stop, 6-15
manual modes of operation, assignments, 6-26
manual panel
  MP 6000M
    description, 2-61

dimensions, drawing, 9-4
MP 6001M
description, 2-61
dimensions, drawing, 9-5
P4 interface, pinout, 2-61
P5 interface, pinout, 2-61
X46, pinout, 2-40
manual pulse generator, defined, 2-61
manual pulse generator, See MPG
manual transmission, controller parameters, 5-124
manual, axis-direction keys, handwheel, 6-75
master-slave axes, special cases, 5-69, 6-85
maximum, PLC program run time, 7-7
maxStrokeFromHome_FirstPick, description, 2-19
MC 400
connections, illustration, 2-9
description, 1-3
dimensions drawing, 9-7
drawing, 9-6
positioning, illustration, 2-7
product designation, 1-2
switching, inputs 24 VDC (PLC), 2-30
X147, I/O module, pinout, 2-43
X28, RS-422 data interface, 2-47
X42, PLC input, 2-31
X49, to flat panel display, pinout, 2-38
MC 400 pinouts
analog input, 2-28
analog nominal value output, 2-27
CNC keyboard, 2-42
CNC power supply and control signals, 2-23
control-is-ready signal, 2-24
data interfaces, 2-46
drive controller enable, 2-50
encoders for speed control, 2-12
flat panel display, 2-38
I/O module connection, 2-43
listed, 2-10
manual panel, 2-40
PLC analog input, 2-29
PLC input/output units, 2-51
position control for encoders, 2-11
PWM connection, axis/spindle motors, 2-22
switching, inputs 24 VDC (PLC), 2-30
switching, outputs 24 VDC (PLC), 2-34
touch probe, 2-13
USB hub, 2-49
USB interface, 2-49
MC 400 pinouts, power supply for PLC outputs, 2-24
MC 400, X27, RS-232 data interface, 2-46
MC, CC, and inverter, dimensions, drawing, 9-3
mechanical vibration, description, 2-6
message, for power interruption, 6-5
mode of operation, pass over reference point, 5-92
model numbers, listed, 1-2
modes of operation, description, 6-14
Module 9000/9001, copy in the marker or word range, 7-216
Module 9002, read inputs of PLC input/output unit, 6-99
Module 9004, read the edges of PLC inputs, 6-100
Module 9005, set the outputs of the PLC input/output unit, 6-100
Module 9006, set and start PLC timer, 7-26
Module 9007, read the diagnostic information of a PLC input/output unit, 6-95
Module 9010/9011/9012, read in the word range, 7-217
Module 9019, size of the processing stack, 7-120
Module 9020/9021/9022, write in the word range, 7-218
Module 9025, writing a value as BCD code to eight successive markers, 7-219
Module 9034, load a machine parameter subfile, 3-25
Module 9035, read NC status information, 5-67, 6-83, 6-106
Module 9036, write NC status information, 6-81, 6-105
Module 9038, read status information of axes, 5-37
Module 9040, reading of axis coordinates by the PLC in the format 1/1000 (0.001) mm, 5-39, 7-220
Module 9041, reading of axis coordinates by the PLC in the format 1/10000 (0.0001) mm, 5-40, 7-221
Module 9042, reading of spindle coordinates by the PLC in the format 1/1000 (0.001)

degrees, 7-222
Module 9044, reading of spindle coordinates by the PLC in the format 1/10000 (0.0001) degrees, 7-223
Module 9050, conversion of binary numbers–>ASCII, 7-253
Module 9051, conversion of binary numbers–>ASCII, 7-254
Module 9052, conversion of ASCII numbers –>binary, 7-255
Module 9053, conversion from binary –>ASCII/hexadecimal, 7-256
Module 9054, conversion from ASCII/ hexadecimal–>binary, 7-257
Module 9055, convert time (binary) to formatted string, 6-114
Module 9060, M function, status, 6-35
Module 9061, status of non-modal M functions, 6-35
Module 9070, copy a number from a string, 7-192
Module 9071, find the string length, 7-194
Module 9072, copy a byte block into a string, 7-195
Module 9073, copy a string into a byte block, 7-194
Module 9084, display PLC error messages with additional data, 6-59
Module 9085, display PLC error messages, 6-60
Module 9086, delete PLC error messages, 6-61
Module 9087, status of PLC error messages, 6-62
Module 9088, status display of M functions, 6-36
Module 9091, find the line number of a tool in the tool table, 6-125
Module 9092, search for an entry in the tables selected for execution (.T/.D/.TCH), 6-119
Module 9093, read data from tables selected for program (.T/.D/.TCH), 6-121
Module 9094, write data into a tool and datum table, 6-122
Module 9095, activate axis error compensation, 5-66
Module 9096, delete line in tool table, 6-123
Module 9100, assign data interface, 8-20
Module 9101, release data interface, 8-21
Module 9102, status of data interface, 8-22
Module 9103, transmit string through data interface, 8-23
Module 9104, receive string through data interface, 8-24
Module 9105, transmit binary data through data interface, 8-25
Module 9106, receive binary data through data interface, 8-26
Module 9107, read from receiving buffer, 8-27
Module 9111, receive a message via LSV2, 7-224
Module 9112, transmit ASCII characters via data interface, 8-28
Module 9113, receive ASCII characters via data interface, 8-29
Module 9120, position PLC axes, 5-44
Module 9121, stop PLC axis, 5-46
Module 9122, status of PLC axis, 5-47
Module 9123, traverse reference marks of PLC axes, 5-48
Module 9124, feed rate override for PLC axis, 5-49
Module 9125, stop PLC axis at next Hirth grid position, 5-50
Module 9130, output analog voltage, 6-103
Module 9133, interrogate the values, internal ADCs, 5-146
Module 9137, read the diagnostic information of the IEB 404, 6-96
Module 9138, read analog input of IEB 404, 6-101
Module 9139, monitoring functions for the IEB 404 PLC input/output units, 6-98
Module 9140, set axis-specific feed-rate limit, 7-225
Module 9141, read axis-specific feed-rate (status), 7-226
Module 9145, actual-to-nominal value transfer, 5-120
Module 9147, assigning a reference value to an axis, 7-227
Module 9166, read current utilization of drive motor, 5-148
Module 9171, start of a spindle orientation with adjustable parameters, 7-228
Module 9180, keystroke simulation, 6-66
Module 9181, disable NC key by PLC, 6-66

Module 9182, re-enable NC key by PLC, 6-67
Module 9183, disable NC key group by PLC, 6-67
Module 9184, enable locked NC key groups by PLC, 6-68
Module 9186, call a soft key function, 6-69
Module 9187, status query of a soft key call, 6-69
Module 9189, shut down the control, 6-9
Module 9190, start the PLC operating hours counter, 6-109
Module 9191, stop the PLC operating hours counter, 6-110
Module 9192, transfer the operating hours counter, 6-110
Module 9193, set the operating hours counter, 6-111
Module 9194, alarm when operating time exceeded, 6-112
Module 9195, transfer the real-time clock, 6-114
Module 9196, find the PLC cycle time, 7-2
Module 9197, start cyclic timer, 7-27
Module 9220, traverse reference marks, 5-83
Module 9221, start PLC positioning movement, 5-54
Module 9222, interrogate PLC positioning status, 5-55
Module 9223, free rotation, 6-2
Module 9224, stop PLC positioning movements, 5-55
Module 9231, compensation of thermal expansion, 5-71
Module 9240, open a file, 7-49
Module 9241, close a file, 7-50
Module 9242, positioning in a file, 7-51
Module 9243, reading from a ASCII file line by line, 7-52
Module 9244, writing to an ASCII file line by line, 7-53
Module 9245, read a field in a table, 7-54
Module 9246, write to a field in a table, 7-56
Module 9247, search for a condition in a table, 7-58
Module 9248, copy, rename, or delete file, 7-230
Module 9249, read and reset ’errno’, 7-59
Module 9255, read a data field in a table, 7-55
Module 9256, write to a data field in a table, 7-57
Module 9260, receiving events and waiting for events, 7-202
Module 9261, sending events, 7-204
Module 9262, context change between spawn processes, 7-205
Module 9263, interrupting a spawn process for a defined time, 7-205
Module 9264, wait for a condition, 7-206
Module 9270, read OEM-defined string value, 7-231
Module 9271, write OEM-defined string value, 7-232
Module 9275, write ASCII data into the log file, 6-63
Module 9276, write operand contents into error log file, 6-64
Module 9277, writing data into the OEM log, 7-233
Module 9279, shut down control (configurable), 6-10
Module 9291, starting an NC macro, 7-234
Module 9300, locking and releasing the pocket table, 7-236
Module 9301, find the number of an entry in the pocket table, 7-237
Module 9302, search for a vacant pocket in the tool magazine, 7-237
Module 9304, copying OEM values from the pocket table, 7-238
Module 9305, moving tools in the pocket table, 7-239
Module 9306, moving tools between magazines, 7-240
Module 9321, block number, find current, 6-23
Module 9322, information of the current NC program, 7-241
Module 9340, searching for a pocket depending on magazine rules, 7-242
Module 9341, editing a pocket table depending on magazine rules, 7-243
Module 9342, find magazine and pocket number, 7-244
Module 9343, compilation and activation of magazine rules, 7-245
Module 9350, read data from the tool table, 7-246
Module 9351, write data to tool table, 7-247

Module 9404, start movement when an NC stroke is present, 6-24
Module 9405, convert a symbolic operand into a numerical PLC operand, 7-21
Module 9407, give default tool number for an NC channel, 7-248
Module 9410, read spindle status, 5-156
Module 9411, read the actual spindle values (speed, coordinates), 5-41, 7-249
Module 9412, stop the spindle, 5-157
Module 9413, move the spindle, 5-158
Module 9414, position the spindle, 5-160
Module 9416, select gear range and assigned settings for spindle, 7-250
Module 9417, set default shaft speed for spindle, 7-251
Module 9418, set status for spindle, 7-252
Module 9430, change numeric value of machine parameter, 3-31
Module 9431, read numeric value of machine parameter, 3-32
Module 9432, change string value of machine parameter, 3-33
Module 9433, read string value of machine parameter, 3-34
Module 9434, select parameter block, 5-134
Module 9435, status of parameter block of an axis, 5-135
Module 9440, SQL: open a transaction, 7-75
Module 9441, SQL: conclude and close a transaction, 7-76
Module 9442, SQL: seek a record in the result set, 7-77
Module 9443, SQL: fetch a record from the result set, 7-78
Module 9444, SQL: change a record in the result set, 7-79
Module 9445, SQL: read a single value from a table, 7-80
Module 9447, SQL: delete record from result set, 7-81
Module 9448, SQL: load a column description, 7-82
Module 9449, SQL: extract a value from a comma separated list, 7-83
Module 9450, SQL: execute an SQL statement, 7-84
Module 9451, SQL: roll back and close a transaction, 7-85
Module 9452, SQL: seek next record in the result set of a query, 7-86
Module 9453, SQL: fetch binary data from the result set of a query, 7-87
Module 9454, SQL: update binary data in the result set of a query, 7-88
Module 9455, SQL: read a single numeric value from a table, 7-89
Module 9458, SQL: unload a column description, 7-90
Module 9459, SQL: change or insert a value in a comma separated list, 7-91
modules, listed, table, 4-1
Modules 9440–9459, return codes, 7-92
modules for string processing, 7-193
monitoring functions
  activating, 6-144
  axes in motion, 5-145
  axes in position, 5-144
  clamped axes, 5-140
  description, 5-136
  differences between positions, switch-on and shutdown, 5-140
  drive monitor, utilization, 5-147
  drives, 5-136
  EMERGENCY STOP monitoring, 5-149
  hanging axes, 5-140
  movement monitoring, 5-141
  position monitoring, 5-138
  positioning window, 5-143
  standstill monitoring, 5-142
  switching off, for individual axes, 5-137
  switching off, globally, 5-136
  temperature monitoring, 5-146
monitoring, encoders, 5-32
monitoring, encoders, with EnDat interface, 5-33
More Cmds (F7)
  configuration editor, 3-4
  from Config Data (F6), 3-6
mounting
  chassis, inverter, amplifier power module, 2-7
  considerations, 2-7
mounting & electrical installation
  general information, 2-1
  safety precautions, 2-1
movement monitoring, description, 3-29. 5-141

moving to restore position, machining channels, 5-10
MP 6000M
  manual panel, dimensions, drawing, 9-4
  product designation, 1-2
MP 6001M
  manual panel, dimensions, drawing, 9-5
  product designation, 1-2
MP change list, configuration editor, 3-28
MP movement monitoring, description, 3-29
MP programming station mode, description, 3-29
MP subfiles
  activating, 3-23
  description, 3-23
  PLC activation, 3-25
  syntax, 3-23
MP11 3-D, corded touch probe, illustration, 2-14
MPG, defined, 2-61
Msgs (SHIFT + F1), illustration, 6-48
multi-axis systems, configfiles.cfg, 3-19
multiple tool magazines, managing, 7-235
MULTIPLICATION (X), 7-155
MULTIPLICATION [ ] (x[ ]), 7-168

N

NC axes, group, OLM, 6-154
NC channels, group of, 6-168
NC macro
  calling, with M function, 6-31
  executing, 6-32
NC program
  automatic start, 6-19
  cancellation, 6-22
  interrupting, 6-21
  run, 6-17
  run, NC stops, 6-21
  run, PLC stops run, 6-21
  starting, 6-18
  terminating, 6-20
NC stop, retract tool, 6-16
NC stops NC program run, 6-21
 NC stops NC program run, 6-21
nominal value output
  analog, 2-27
  description, 2-27
nominalProbeStylusBallRadius, description, 2-21
nominalProbeStylusDiameter, description, 2-21
nonlinear axis, error compensation, description, 5-62
NOT EQUAL (<>), 7-164
NOT EQUAL [ ] (<>[ ]), 7-170
NOT EQUAL TO STRING (<>), 7-192
number conversion, PLC modules, 7-253

O

object tree
  icons, defined, 3-7
  illustration, 3-7
object, overview, illustration, 3-27
Offset Data (F2), offset interface, 6-171
Offset Data (F5), interpolator & PLC-IPO data, 6-159
OLM
  auxiliary group, 6-178
  diagnosis with, 6-145
  errors, frequent causes, 6-187
  generating asynchronous position compensation, 6-182
  generating error messages, 6-181
  group of NC axes, 6-154
  hardware group, 6-173
  operating, starting & exiting, 6-148
  operation, 6-147
  operation, description, 6-147
  PLC group, 6-183
  PLC trace, 6-184
  queue trace, 6-185
  screen display, units, 6-152
  screen layout, 6-151
  software structure, 6-146
  status display, illustration, 6-153
  status display, selecting axes and channels, 6-153
  variable display, illustration, 6-151
OLM (SHIFT + F8), screen illustration, 6-148
OMP-40, cordless touch probe, illustration, 2-18
on-line monitor, see OLM
operand
  addressing (byte, word, double word), 7-22
  timers, 7-23
operands
  counter, description, 7-28
  description, 7-20

overview, 7-20
operating mode group, PLC operands, listed, table, 4-8
operating times
description, 6-107
measuring, 6-107
OR (O), 7-147
OR [ ] (O[ ]), 7-168
OR NOT [ ] (0N[ ]), 7-167
ORDER BY, command option, description, 7-67
orientation settings, tool probe, table, 2-19
oriented, spindle stop, description, 5-162
OSC (SHIFT + F7), oscilloscope main screen, 5-170
oscilloscope
display, configure colors, 5-181
sampling rate, 5-169
OSC extension (oscilloscope trace file), 5-179
oscilloscope, integrated
fundaments, 5-168
recording, saving/loading, 5-179
signals, overview, 5-168
output, X41, table, 2-36
Output B/W/D (F4), shows lists, 7-16
override
compensation for potentiometers, 6-88
description, 6-86
devices, 6-86
feedrate, description, 6-91
functions, description, 6-89
rapid traverse, 6-93
speed, description, 6-89
overview, 6000i, 1-3
OVERWRITE STRING (OVWR), 7-189

P

P/N 34000850, RM 500 panel-mounted handwheel, illustration, 9-23
P/N 34000855, PM 300 panel-mounted handwheel, illustration, 9-24
P/N 624517-XX, cable lengths, available, 2-44
P/N 574744-51, system ID key (SIK), illustration, 9-18
P/N 574774-01, MC 400, product designation, 1-3
P/N 574746-51, hard disk drawer, HDR, 9-15
P/N 624498-01, IEB 404, connector description, 9-9
P/N 624505, IEM 16-8D, I/O module, digital 16/8, 9-11
P/N 624505-01, I/O module, digital 16/8, 9-12
P/N 624506-01, I/O module, analog 4/4, 9-13
P/N 624506-01, I/O module, analog 4/4, connectors, 9-14
P/N 624508-01, USB hub, dimensions, 9-19
P/N 627785-2X, *6000i CNC User's Manual*, referenced, 2-18, 2-55, 6-1, 6-115, 7-206
P/N summary, I/O modules and EXP base modules, 2-51
P4, manual panel interface, pinout, 2-61
P5, manual panel interface, pinout, 2-61
panel-mounted handwheel
PM 300, 2-56
PM 350, 2-57
parameter block
assigning, 5-20
creating, 8-11
switching, control loop, 5-133
parameter files
rules for entries, 3-21
setup, 3-21
parameter setup, spindle probe, description, 2-20
parameter setup, tool probe, description, 2-18
parameters, configuration
backup, 3-18
screen capture, 2-18
screen capture, additional parameters, 2-18
parity bit
serial interface, 8-15
transmission reliability, 8-15
pass over reference point, mode of operation, 5-92
password, for setup configuration, 3-3
paths and names, PLC programs and text files, 7-33
path-specific limit values, description, 5-101
per-revolution feed, spindle, 5-166
PET table
description, 6-56
structure, 6-57
PET, PLC error message table, 6-57

physical axes
activate axis, 5-19
description, 5-17
parameter blocks, assigning, 5-20
virtual axis, 5-21
PLC
analog inputs, 6-101
analog outputs, 6-103
axes, starting/stopping, 5-44
basic program, description, 7-6
Blum laser touch probe, input labels, 2-16
Blum laser touch probe, output labels, 2-16
commands, description, 7-121
controlling axes, 5-44
data transfer, 8-19
editing tables, 7-49
error message table, PET, 6-57
error messages, 6-56
error table, 6-56
fast inputs, description, 7-30
functions, description, 7-2
group, OLM, 6-183
IEB 404 inputs, 2-33
input/output units, description, 2-51
inputs, description, 6-94
main menu, description, 7-7
main menu, soft key functions, 7-8
mode, selecting, 7-6
module, machine parameters, read or change, 3-31
MP subfile, activation, 3-25
nominal operating current per output, 2-25
operands
axis group, listed, table, 4-12
feed rate values, in, 5-118
general data group, listed, table, 4-6
machining channels group, listed, table, 4-9
operating mode group, listed, table, 4-8
spindle group, listed, table, 4-14
outputs, description, 6-94
outputs, power supply, 2-24
overview, 7-121
positioning of axes, 5-49
program structure, 7-120
program version, 7-35
program, maximum run time, 7-7
programming, description, 7-2
programs and text files, paths and names, 7-33
power consumption, 2-24
spindle control, 5-156
stops NC program run, 6-21
switching, outputs 24 VDC, 2-34
symbolic, description, 7-3
symbolic, operands, name convention, 7-4
system files, 7-32
trace, OLM, 6-184
X41, output, on the CNC, 2-35
X41, output, table, 2-36
X42, input on the MC 400, 2-31
X44, power supply pinout, 2-25
X48, analog input, pinout, 2-29
X6, outputs on IEB 404, pinout, 2-37
PLC (SHIFT + F5), PLC main menu, 7-7
PLC =, FN19, 7-95
PLC =, FN29, 7-95
PLC commands
ADD [ ] (+[ ]), 7-167
ADDITION (+), 7-153
AND (A), 7-141
AND [ ] (A[ ]), 7-165
AND NOT (AN), 7-143
AND NOT [ ] (AN[ ]), 7-166
ASSIGN (=), 7-133
ASSIGN BYTE (B=), 7-135
ASSIGN DOUBLE WORD (D=), 7-136
ASSIGN NOT (=N), 7-136
ASSIGN TWO'S COMPLEMENT (=–), 7-136
ASSIGN WORD (W=), 7-135
BIT CLEAR (BC), 7-174
BIT SET (BS), 7-173
CALL MODULE (CM), 7-181
CALL MODULE IF FALSE (CMF), 7-182
CALL MODULE IF TRUE (CMT), 7-181
DECREMENT (DEC), 7-158
DIVISION (/), 7-156
DIVISION [ ] (/[ ]), 7-168
END OF MODULE (EM), 7-183
END OF MODULE IF FALSE (EMF), 7-183
END OF MODULE IF TRUE (EMT), 7-183

EQUAL TO (==), 7-159
EQUAL TO [ ] (==[ ]), 7-169
EXCLUSIVE OR (XO), 7-149
EXCLUSIVE OR [ ] (X0[ ]), 7-166
EXCLUSIVE OR NOT (XON), 7-151
EXCLUSIVE OR NOT [ ] (X0N[ ]), 7-166
GREATER THAN (>), 7-161
GREATER THAN [ ] (>[ ]), 7-170
GREATER THAN OR EQUAL TO (>=), 7-163
GREATER THAN OR EQUAL TO [ ] (>=[ ]), 7-170
INCREMENT (INC), 7-158
JUMP (JP), 7-179
JUMP IF FALSE (JPF), 7-180
JUMP IF TRUE (JPT), 7-180
LABEL (LBL), 7-183
LESS THAN (<), 7-160
LESS THAN [ ] (<[ ]), 7-170
LESS THAN OR EQUAL TO (<=), 7-162
LESS THAN OR EQUAL TO [ ] (<=[ ]), 7-170
LOAD (L), 7-126
LOAD BYTE (LB), 7-131
LOAD DOUBLE WORD (LD), 7-132
LOAD MINUS (L-), 7-130
LOAD NOT (LN), 7-128
LOAD WORD (LW), 7-132
MULTIPLICATION (X), 7-155
MULTIPLICATION [ ] (x[ ]), 7-168
NOT EQUAL (<>), 7-164
NOT EQUAL [ ] (<>[ ]), 7-170
OR (O), 7-145
OR [ ] (0[ ]), 7-166
OR NOT [ ] (0N[ ]), 7-166
PULL (PL), 7-177
PULL LOGICACCU (PLL), 7-179
PULL WORDACCU (PLW), 7-179
push data onto the data stack (PS), 7-176
PUSH LOGICACCU (PSL), 7-178
PUSH WORDACCU (PSW), 7-178
REMAINDER (MOD), 7-157
REMAINDER [ ] (MOD[ ]), 7-168
RESET (R), 7-138
RESET NOT (RN), 7-140
SET (S), 7-137
SET NOT (SN), 7-139
SHIFT LEFT (<<), 7-171
SHIFT RIGHT (>>), 7-172
SUBTRACTION (–), 7-154
SUBTRACTION [ ] (–[ ]), 7-168
Plc Ipo Data (F3), description, 6-154
PLC modules
description, 7-215
for SQL statements, 7-75
markers, bytes, words, and double works, 7-216
number conversion, 7-253
Plc Nom Data (F1), nominal commands of the PLC, 6-154
Plc Nom State (F1), nominal status of the axes requested by PLC, 6-161
PlcTrc On-Off (F3), traces PLC modules called, 6-184
PlcTrc Save (F4), saves PLC modules called, 6-184
plc.cfg, Blum laser probe, example, 2-17
plc_oem.cfg, Blum laser probe, example, 2-17
PM 300
description, 2-60
handwheel, illustration, 9-24
handwheel, input, 2-52
hardware, table, 2-56
PM 350
description, 2-57
internal wiring to PMA 310, illustration, 2-58
interpolation, assignments, 2-57
third wheel assignments, table, 2-57
X1, X2, X3 inputs, pinout, 2-57
X23, connection to CNC chassis, pinout, 2-58
X31, supply voltage, pinout, 2-59
PMA 310, PM 350 handwheel adapter cable, 2-57
pocket table, description, 6-115
pocket table, elements of, 6-117, 6-118
position control loop
activating & deactivating, 5-120
filter before, 5-124
opening, 5-120
position control, speed encoder, 2-12
position controller
controller parameters for manual transmission, 5-124
description, 5-112
feed rate values in PLC operands, 5-118

feedback control with following error, 5-114
feed-rate enable, 5-123
limiting, controller output, 5-116
rapid traverse, 5-118
position encoder
input, 5-28
input, handwheel, 6-77
machine datum, defined, 5-83
monitoring, 5-32
reference mark, one, 5-89
reference marks, position-coded, 5-88
signal, 5-28
position loop, resolution, analog axes, 5-133
position monitoring, description, 5-138
position-coded, position encoder, reference marks, 5-88
position, inputs, 2-11
positioning
after reference mark traverse, 5-93
axes by PLC, 5-49
spindle, 5-160
window, description, 5-143
positioningFeedrate_Normally, description, 2-21
potentiometers, compensation for, 6-88
power consumption, PLC outputs, 2-24
power interruption, message, 6-5
power supply
24V control-is-ready signal, 2-24
CNC & control signals, 2-23
PLC, X44 pinout, 2-25
powering up, the control, 6-5
prepare recording, oscilloscope, 5-170
probe
orientation, settings, table, 2-19
spindle
activateMStrobe, description, 2-21
actiivateRetries, description, 2-21
actiivateTimeout, description, 2-21
actiivationType, description, 2-21
autoDeactivateTimet, description, 2-21
deactivateMStrobe, description, 2-21
deactivateRetries, description, 2-21
deactivateMStrobe, description, 2-21
deactivateType, description, 2-21
diameterOfSpindleProbeGauge, description, 2-21
dwellTimeAfterProbeActive, description, 2-21
nominalProbeSylusBallRadius, description, 2-21
orientProperty, description, 2-21
orientSpeed, description, 2-21
parameter setup, description, 2-20
positionFeedrate_Normally, description, 2-21
updateTloOrWorkOffsetZAxis, description, 2-21
tool
calibAndToolMeasurementRPM, description, 2-19
diameterOfToolProbeGauge, description, 2-19
maxStrokeFromHome, description, 2-19
nominalProbeStylusDiameter, description, 2-19
orientation settings, table, 2-19
parameter setup, description, 2-18
probeOrientation, description, 2-19
settings found in OEM, illustration, 2-21
toolProbeType, description, 2-19, 2-20
useAnilamLaserCycles, description, 2-19
ZFirstPickFeedRate_Fast, description, 2-19
ZFirstPickFeedRate_Medium, description, 2-20
ZFirstPickFeedRate_Slow, description, 2-20
ZRapidToStartPositionFromHome, description, 2-20
probeOrientation, description, 2-19
product, designations, 1-2
program creation
ASCII editor, 7-119
description, 7-119
program structure, 7-120
program interruption, moving the axes, 6-22
program stop, M6, 6-34
program structures
case branch, 7-211
description, 7-208
REPEAT...UNTIL, 7-209
WHILE...ENDW, 7-210
program version, PLC, 7-35

programmable axes
description, 5-15
SysRead, SysWrite, index, 5-16
without, separate drive motor, 5-16
programming station mode, description, 3-29
programs, submit, description, 7-197
properties of, rotary axes A, B, C, 5-13
protective measures, listed, 2-3
PS (PUSH), 7-176
Pt 100 thermistor, inputs, description, 2-28
PULL (PL), 7-177
PULL LOGICACCU (PLL), 7-179
PULL WORDACCU (PLW), 7-179
Pulse Width Modulation, see PWM
PUSH (PS), 7-176
PUSH LOGICACCU (PSL), 7-178
PUSH WORDACCU (PSW), 7-178
PWM
axis/spindle motors, connection, 2-22
defined, 1-1, 2-22
Pulse Width Modulation, defined, 1-1
X51–X56, to motor, pinout, 2-22

Q

Q/QS parameters, saving, 5-8
Q-Trace On-Off (F1), define queues to be traced, 6-186
queue trace, OLM, 6-185

R

rapid traverse
analog axes, 5-36
description, 5-118
override, 6-93
reaction, to errors, 6-15
read data from table, example, 7-72
reading
actual spindle values, 5-41
axis coordinates, 5-38
axis information, 5-36
record
hide/show gridlines, 5-175
signals, oscilloscope, 5-174
start/stop, oscilloscope, 5-174
trigger conditions, oscilloscope, 5-175
recording
completed, oscilloscope, 5-176
oscilloscope, saving & loading, 5-179
prepare, oscilloscope, 5-170
reference end point, description, 5-92
reference for syntax elements
BNF notation, 7-64
command options
description, 7-67
FOR NOTIFICATION, 7-67
FOR UPDATE, 7-67
ORDER BY, 7-67
WHERE, 7-67
context elements, 7-64
description, 7-64
SQL, 7-65
SQL column, 7-65
SQL INDEX, 7-65
SQL parameters, 7-64
SQL statement, 7-65
SQL-HANDLE, 7-66
reference marks
definition, 5-82
description, 5-82
encoders, distance-coded, 5-27
encoders, with EnDat interface, 5-83
linear measurement through rotary encoder, 5-90
machine datum, 5-93
pass over reference point, 5-92
position encoder, distance-coded, 5-88
position encoder, with one, 5-89
positioning after traverse, 5-93
process of traversing, 5-85
reference end point, 5-92
renewed traversing, 5-83
traversing, 5-9, 5-82
traversing, direction and velocity, 5-86
reference point, delete, Clear RefOK (F7), 6-166
release, hard drive, 2-8
REMAINDER (MOD), 7-157
REMAINDER [ ] (MOD[ ]), 7-168
remove column names and column descriptions, 7-47
removing
hard drive, 2-8
HDR, 2-4
SIK, 2-4
RENAME COLUMN, SQL command, 7-71
RENAME TABLE, SQL command, 7-69
REPEAT...UNTIL structure, 7-209
replacement tool, 6-124
RESET (R), 7-138
RESET NOT (RN), 7-140

Reset Version (F4), screen illustration, 3-17
reset, update version, 3-17
resolution, position loop, analog axes, 5-133
retract tool, at NC stop, 6-16
return codes of PLC Modules 9440–9459, 7-92
reversal peaks
  compensation of, 5-130
  compensation, analog axes, 5-128
reversal, of traverse direction, 6-132
RJ45 port, 2-48
RM 500
  connection guidelines, 2-54
  handwheel, input, 2-52
  handwheel, illustration, 9-23
  handwheel, remote, pinout, 2-55
  internal wiring, to control buttons, 2-54
  operation, guidelines, 2-55
  remote handwheel, description, 2-53
  remote handwheel, pinout, 2-55
  replacement terminals, listed, 2-54
rotary axes
  look-ahead tolerance, 5-106
  properties of A, B, C, 5-13
rotary axis, nonlinear, axis error compensation, 6-85
rotary encoder
  cable lengths, available, 2-12
  connection to motor, pinout, 2-12
  linear measurement, reference marks, 5-90
  pinouts, 2-12
rotating, spindle, oriented, spindle stop, 5-162
rotational speed reached, tolerances, 5-169
RPLY, interrogating the status of a submit program, 7-198
RS-232, data interface, 2-46
RS-232-C/V.24 interface, 8-6
RS-422, data interface, 2-47
rules for entries, editor changes, 3-21

S

S function
  assigning to machining channels, 6-37
  configuration of, 6-38
  description, 6-37
  transfer, 6-38
  transfer & acknowledgement, 6-40
S strobe, description, 6-37
SA XXX, product designation, 1-2
safety precautions, mounting & electrical installation, 2-1
sampling rate, oscilloscope, 5-169
Save Chgs (F6), description, 3-10
SCO extension (oscilloscope trace file), 5-179
screens
  Config Struct (F5), illustration, 3-13
  Data Backup (F2), illustration, 3-18
  Filter (F7), illustration, 6-53
  Find (F3), dialog box, illustration, 3-11
  Find (F3), illustration, 6-52, 6-55
  Help (F4), illustration, 3-8
  Info View (F6), illustration, 3-12
  Key Log (F2), illustration, 6-54
  Machine Parameter, from Config Data (F6), illustration, 3-5
  More Cmds (F7), configuration editor, illustration, 3-4
  More Cmds (F7), from Config Data (F6), illustration, 3-6
  Msgs (SHIFT + F1), illustration, 6-48
  OLM, illustration, 6-148
  Reset Version (F4), illustration, 3-17
  Select Record (F4), illustration, 3-26
  setup configuration, illustration, 3-3
  Trace (F7), illustration, 6-150
  Tree View (F5), illustration, 3-14
  Update Rules (F3), illustration, 3-16
second cursor, activate/deactivate, 5-178
Secure Area (F6)
  description, 3-14
  MP change list, 3-28
Select Record (F4), screen illustration, 3-26
SELECT, description, 7-67
SELECT, SQL command, 7-68
selection from list, screen illustration, 3-9
serial handwheel, description, 6-73
serial interface
  communications protocol, 8-14
  configuration, 8-9, 8-10
  data bits, 8-14
  description, 8-6
  parity bit, 8-15
  selecting, parameter block, 8-11
  signal designations, 8-8
servo amplifier, adjusting, 6-128
servo drive
  cannot be switched off, correction, 6-187

does not move, correction, 6-188
servo, turn on circuit, basic
drawing, 9-20
notes, description, 9-21
SET (S), 7-137
Set Error (F4), define OLM error number & error class, 6-181
SET NOT (SN), 7-139
Set PosCorr (F4), generate asynchronous position compensation, 6-182
setting, the traverse range, 6-143
setup
analog signals, oscilloscope, 5-171
configuration, screen illustration, 3-3
digital signals, oscilloscope, 5-173
SHIFT LEFT (<<), 7-171
SHIFT RIGHT (>>), 7-172
shipping
brace
hard drive, 2-8
HDR, 2-3
shock, permissible, 2-6
secure hard drive, 2-8
shock, permissible, 2-6
shock, with shipping brace, permissible, 2-6
shutting down, the control, 6-6
signal display, influence, 5-178
signal period, description, 5-25
signals
integrated oscilloscope, 5-168
record, oscilloscope, 5-174
SIK
defined, 2-8
description, 1-3
handling, 2-3
installation, illustration, 9-18
installing/removing, 2-4
System Identification Key, defined, 1-3
SM XXX, product designation, 1-2
software
handshaking, description, 8-18
update, procedure, 1-4
update, reset version, 3-17
Sort Content (F1), description, 3-13
sort file content, description, 3-13
SPAWN, starting a parallel process, 7-201
special kinematics, axes, 5-4
special tools, 6-123
specifications, overview, 1-1
speed
adjustment, description, 6-132
encoder, description, 2-12
override, description, 6-89
spindle
commands, group of, 6-166
configuring, 5-2
configuring, description, 5-151
control by PLC, 5-156
controlling, 5-154
definition of axes, 5-4
drive, switching on/off, 5-155
filter, acceleration values, 5-153
gear shifting, 5-167
group, PLC operands, listed, table, 4-14
per-revolution feed, 5-166
position encoder, 5-152
positioning, 5-160
read, actual values, 5-41
rotating, stop, oriented, 5-162
stationary, stop, oriented, 5-164
stop, at trip dog position, 5-165
stop, oriented, 5-162
tapping, 5-167
tapping, description, 5-161
X19, speed control, 2-12
X56, to motor, pinout, 2-22
Spindle (F4), description, 6-154
SQL
column, syntax, 7-65
defined, 7-58
parameters, syntax, 7-64
statement, syntax, 7-65
syntax, 7-65
System Query Language, defined, 7-58
SQL commands
access to tables, 7-60
application example, 7-72
concluding transaction, tables, 7-62
described, 7-68
reading data, tables, 7-61
selecting the data, tables, 7-61
SQL-HANDLE, syntax, 7-64
SQLINDEX, syntax, 7-65
S-RAM (F4), static RAM of the IPO, 6-171
standard, coordinates, configuration of axes, 5-14
standstill monitoring, description, 5-142
start block search, block scan, 6-23
start, machining channels, 6-15

start/stop record, oscilloscope, 5-174
starting, NC program, 6-18
starting/stopping axes, by PLC, 5-44
static friction, hysteresis, 6-144
Static RAM (S-RAM), 6-173
stationary, spindle, oriented, spindle stop, 5-164
status submit (RPLY), 7-198
stiction compensation, 5-56
stop
  bits, synchronization, 8-16
  machining channels, 6-15
  spindle, at trip dog position, 5-165
stopping/starting axes, by PLC, 5-44
STORE STRING (=), 7-188
string processing
  ADD STRING (+), 7-188
  commands for, 7-186
  EQUAL TO STRING (==), 7-190
  GREATER THAN OR EQUAL TO STRING (>=), 7-191
  GREATER THAN STRING (>), 7-190
  LESS THAN OR EQUAL TO STRING (<=), 7-191
  LESS THAN STRING (<), 7-190
  LOAD STRING (L), 7-187
  logical comparisons, 7-187
  modules for, 7-193
  NOT EQUAL TO STRING (<>), 7-192
  operand declaration, 7-187
  OVERWRITE STRING (OVWR), 7-189
  STORE STRING (=), 7-189
structure, PET table, 6-57
subfile, unload, description, 3-28
SUBM, calling the submit program, 7-198
submit programs
  call submit (SUBM), 7-198
  canceling a submit program (CAN), 7-199
  description, 7-197
  status submit (RPLY), 7-198
SUBTRACTION (–), 7-154
SUBTRACTION [ ] (–[ ]), 7-168
switching
  inputs, 6-99
  inputs, 24 VDC (PLC), 2-30
  inputs, addresses, 2-30
  off, monitoring functions, globally, 5-136
  outputs, 6-99
  parameter blocks, control loop, 5-133
Symbol List (F3), list box of PLC operands, 7-11
symbolic
  names, for tables, 7-48
  operands, display, 7-11
  PLC-API, description, 7-3
symbols used in this manual, description, 1-2
Syntax Errors (F2), description, 3-17
syntax errors, remove, 3-17
syntax, MP subfiles, 3-23
SYSREAD, FN18, 7-105
SysRead, index, description, 5-16
system
  diagram, basic, illustration, 9-29
  diagram, basic, referenced, 5-149
  files, PLC, 7-32
  machine parameters, listed, 3-35
  overview, 1-1
  strings, 7-116
  time, description, 6-113
  times, description, 6-107
System Identification Key, see SIK
System Query Language, see SQL
SYSWRITE, FN17, 7-97
SysWrite, index, description, 5-16

T

T function
  assigning to machining channels, 6-41
  configuration of, 6-42
  description, 6-41
  transfer & acknowledgment of, 6-44
T strobe, description, 6-41
tables
  access via SQL commands, 7-60
  column description, 7-41
  concluding transaction, using SQL, 7-62
  delete columns from an existing, 7-47
  description, 7-36
  description, configuration editor, 7-40
  editing data, using SQL, 7-61
  editing via the PLC, 7-49
  file extension, specify, 7-38
  insert additional columns in existing, 7-46
  new type, creating, 7-37
  new, create with file manager, 7-45
  reading data, using SQL, 7-61
  remove column names and descriptions, 7-47
  selecting the data, using SQL, 7-60

symbolic names, 7-48
Tables (F5), display tables, 7-12
tapping
description, 5-161
spindles, 5-167
temperature
monitoring, description, 5-146
ambient, 2-5
temporary
data, description, 3-27
input values, commissioning, 6-127
terminating, NC program, 6-20
thermal expansion, compensation, 5-70
thermistor, inputs, 2-28
threshold sensitivity, handwheel, 6-72
timer, description, 7-23
tolerance, for corners and arcs, 5-105
tolerances, rotational speed reached, 5-159
tool
changer, description, 6-115
life, description, 6-124
table, description, 6-115
table, elements of, 6-117
tools, indexed, description, 6-125
touch probe
description, 2-19
MP11 3-D, corded, illustration, 2-15
OMP-40, cordless, illustration, 2-18
TS27R, corded, illustration, 2-14
X12, input for workpiece measurement, 2-13
X13, input for tool measurement, 2-13
touchpro.scr, Blum laser probe, 2-17
Trace (F7), screen illustration, 6-150
Trace, description, 6-150
transfer of, S function, 6-38
transmission reliability, parity bits, 8-15
traverse
direction, defining, 5-31
direction, reversal of, 6-132
motion, in parallel with M, S, or T function, 6-24
range, setting, 6-143
ranges, description, 5-42
traversing, reference marks, 5-9, 5-82
Tree View (F5), configuration editor, 3-14
tree view, description, 3-14
trigger conditions, record, oscilloscope, 5-175
trip dog position, M19, 5-160
trip dog position, stop spindle, 5-165
TS27R, corded touch probe, illustration, 2-14

U

unit of measurement, display and operation, 6-3
Universal Serial Bus, see USB
unload subfile, description, 3-28
Update Rules (F3), screen illustration, 3-16
UPDATE, description, 7-67
UPDATE, SQL command, 7-69
USB
defined, 8-3
devices supported, 8-5
functionality, 8-4
hub, 2-49
hub, dimensions, 9-19
interface, of control, 8-3
signal designations, 8-4
Universal Serial Bus, defined, 8-3
user commenting, description, 3-12
user parameters, configuration of, 6-8
USES statement, 7-213
utilization display, read actual, 5-147

V

value transfer, actual to nominal, 5-122
velocity
description, 5-96
feedforward control, feedback control, 5-116
traverse reference marks, 5-86
version, reset update, 3-17
vibration, permissible, 2-6
virtual axis, description, 5-18

W

WAIT FOR, FN20, 7-118
warranty, iii
Watch List (F4), displays states of PLC operands, 7-10
watch list, internal process, 7-11
WHERE, command option, description, 7-67
WHILE...ENDW structure, 7-210
work length, serial interface, 8-14
write data to table, example, 7-73

X

X1, X2, X3, inputs for PM 350 handwheel, 2-58

X12, touch probe, input for workpiece measurement, 2-13
X13, touch probe, input for tool measurement, 2-13
X141, X142, USB interface, pinout, 2-49
X147, I/O module, pinout, 2-43
X150, drive controller enable for axis groups, pinout, 2-50
X15–X20, rotary encoders, 2-12
X19, spindle, speed encoder, 2-12
X1–X4, linear encoder, position control, 2-11
X1–X5, position encoder, 2-11
X2, I/O module, pinout, 2-45
X23
  PM 300 handwheel, 2-52
  PM 350 handwheel, connection to CNC chassis, 2-59
  RM 500 handwheel, input, 2-52
X26, Ethernet interface
  LEDs, 2-48
  RJ45 pinout, 2-48
  RJ45 connection, 8-2
X27, RS-232,data interface, 2-46
X28, RS-422, data interface, 2-47
X3, IEB 404, power supply for logic circuit, pinout, 2-25
X31, PM 350 handwheel, supply voltage, pinout, 2-59
X34, 24V control-is-ready signal, 2-24
X4, IEM 16-8D I/O module, pinout, 2-33
X41, PLC output on the CNC, 2-35
X42, PLC input, pinout, 2-31
X44, PLC, power supply, 24V input, 2-25
X45, CNC keyboard, pinout, 2-42
X46, manual panel, pinout, 2-40
X48, PLC analog input, pinout, 2-29
X49, flat panel display to MC 400, pinout, 2-38
X4–X5, PLC inputs, 2-33
X5, IEM 16-8D, I/O module, pinout, 2-34
X5, spindle encoder, position control, 2-11
X51–X55, PWM connection, 2-22
X51–X56, PWM outputs, X150 drive enable, 2-50
X6
  PLC outputs on IEB 404, pinout, 2-37
  power supply for PLC outputs, pinout, 2-25
X69, MC 400 power supply, 2-23
X8, analog output, pinout, 2-27
XML commands, creating layout files, 6-11
XML, defined, 6-8

Z

ZFirstPickFeedRate_Fast, description, 2-19
ZFirstPickFeedRate_Medium, description, 2-20
ZFirstPickFeedRate_Slow, description, 2-20
ZRapidToStartPositionFromHome, description, 2-20

ANILAM

U.S.A.
ANILAM
One Precision Way
Jamestown, NY 14701
☎ (716) 661-1899
FAX (716) 661-1884
✉ anilaminc@anilam.com

ANILAM, CA
16312 Garfield Ave., Unit B
Paramount, CA 90723
☎ (562) 408-3334
FAX (562) 634-5459
✉ anilamla@anilam.com

Dial "011" before each number when calling from the U.S.A.

Germany
ANILAM GmbH
Fraunhoferstrasse 1
D-83301 Traunreut
Germany
☎ +49 8669 856110
FAX +49 8669 850930
✉ info@anilam.de

Italy
ANILAM Elettronica s.r.l.
10043 Orbassano
Strada Borgaretto 38
Torino, Italy
☎ +39 011 900 2606
FAX +39 011 900 2466
✉ info@anilam.it

Taiwan
ANILAM, TW
No. 246 Chau-Fu Road
Taichung City 407
Taiwan, R.O.C.
☎ +886-4 225 87222
FAX +886-4 225 87260
✉ anilamtw@anilam.com

United Kingdom
ACI (UK) Limited
16 Plover Close, Interchange Park
Newport Pagnell
Buckinghamshire, MK16 9PS
England
☎ +44 (0) 1908 514 500
FAX +44 (0) 1908 610 111
✉ sales@aciuk.co.uk

China
Acu-Rite Companies Inc.(Shanghai Representative Office)
Room 1986, Tower B
City Center of Shanghai
No. 100 Zunyi Lu Road
Chang Ning District
200051 Shanghai P.R.C.
☎ +86 21 62370398
FAX +86 21 62372320
✉ china@anilam.com

September 2008
Ve 04
627787-21 · 1/2008 · KS · Printed in USA · Subject to change without notice