PA-CONTROL servoTEC

Version R4b
Edition Nov-06
1043974

IEF Werner GmbH
Wendelhofstr. 6
78120 Furtwangen
Phone:+49 (0)7723/925-0
Fax: +49 (0)7723/925-100
Email: INFO@IEF-WERNER.de
www.IEF-WERNER.de
The use of trademarks and trade names does not affect their legal status. These texts and examples were written with the utmost care. Nonetheless, errors cannot be ruled out. Special applications cases were not considered for the examples given. The use of the examples printed requires close checks on, for instance, the traverse distance or an acceleration value. IEF WERNER GmbH is neither legally responsible nor liable in any way for missing or erroneous information and its consequences.

IEF Werner reserves the right to change or improve software or hardware or parts thereof, including the printed matter supplied or parts thereof, without prior notification. All rights to copy (to photomechanical reproduction), even excerpts, are expressly reserved by IEF WERNER GmbH.

We are always grateful for suggestions for improvements and notices of errors.

© November 06 by IEF WERNER GmbH
Contents

1 Technical Short Information 13
  1.1 Design of the Device 13
  1.2 Declaration of Conformity 14
  1.3 Safety Instructions 15
    1.3.1 Connection Instructions 15
    1.3.2 Ambient Conditions 16
    1.3.3 Operating Instructions 16
  1.4 Technical Data 18
    1.4.1 Inputs and outputs of the PA-CONTROL servoTEC 18
  1.5 Installation of the CPU5 20

2 Operator interface 21
  2.1 The keyboard 21
  2.2 General operation 23
    2.2.1 Switch-on of PA-CONTROL 23
    2.2.2 The menu principle 24
    2.2.3 The input field 25
    2.2.4 Entry of a program name 26
    2.2.5 Selection of a name from the program list 27
  2.3 Menu structure of PA-CONTROL 27
  2.4 Automatic 29
  2.5 Manual 29
    2.5.1 Approaches to reference point 29
    2.5.2 Move axes manually via front plate 30
  2.6 Programming 32
    2.6.1 Display program directory 33
    2.6.2 Create new program 33
    2.6.3 Altering an existing program 35
    2.6.4 Copy program 35
    2.6.5 Rename program 35
    2.6.6 Delete program 36
    2.6.7 Program memory occupancy 36
  2.7 Diagnosis 37
    2.7.1 Limit Switch / Stand-by 37
    2.7.2 Inputs, Outputs, Flags 38
    2.7.3 Real number register (R), Integer register (N) 40
    2.7.4 CANopen 41
    2.7.5 D-A Converter (CANopen) 42
  2.8 Run definitions 43
    2.8.1 Start program 43
2.8.2 Program at stop
2.8.3 Program Start after Stop
2.8.4 Program at malfunction
2.8.5 Title
2.8.6 Activation of run definition
2.8.7 Deletion of assignments
2.9 Parameters
2.9.1 ASi parameters
2.9.2 Edit servoTEC parameters
2.10 Basic settings
2.10.1 Load system parameter default values
2.10.2 Load axis parameter default values
2.10.3 Delete program memory
2.10.4 Reinitialize PA-CONTROL
2.10.5 Reinitialization for devices without a display and keyboard
2.11 System Diagnosis
2.11.1 Display hardware configuration
2.11.2 Clock
2.11.3 Keyboard test
2.11.4 Stop key test
2.11.5 System diagnosis - start key test
2.11.6 Key switch test
2.11.7 Test of the COM Port
2.12 Communication via modem
2.12.1 Overview of the modem menu
2.12.2 Activate connection
2.12.3 Display connection status
2.12.4 Terminate connection
2.12.5 Edit Modem Settings
2.12.6 Select modem type
2.12.7 Examples of communication setup sequences

3 Commands of the PA-CONTROL Family
3.1 Overview of the command set
3.1.1 Program execution (automatic)
3.1.2 Axis
3.1.3 Feedback system (rotary encoder / encoder)
3.1.4 Input
3.1.5 Output
3.1.6 Dwell time, monitoring time, real-time clock
3.1.7 Flag
3.1.8 N/R register (variables)
<table>
<thead>
<tr>
<th>Section</th>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.1.9</td>
<td>String</td>
<td>67</td>
</tr>
<tr>
<td>3.1.10</td>
<td>Program branches</td>
<td>68</td>
</tr>
<tr>
<td>3.1.11</td>
<td>Call subroutines</td>
<td>68</td>
</tr>
<tr>
<td>3.1.12</td>
<td>Loops</td>
<td>68</td>
</tr>
<tr>
<td>3.1.13</td>
<td>Parallel runs</td>
<td>68</td>
</tr>
<tr>
<td>3.1.14</td>
<td>Serial port</td>
<td>69</td>
</tr>
<tr>
<td>3.1.15</td>
<td>Display</td>
<td>69</td>
</tr>
<tr>
<td>3.1.16</td>
<td>Keyboard</td>
<td>69</td>
</tr>
<tr>
<td>3.1.17</td>
<td>Mathematics</td>
<td>69</td>
</tr>
<tr>
<td>3.1.18</td>
<td>Comparisons</td>
<td>70</td>
</tr>
<tr>
<td>3.1.19</td>
<td>Logic operations</td>
<td>70</td>
</tr>
<tr>
<td>3.1.20</td>
<td>A-D converter</td>
<td>70</td>
</tr>
<tr>
<td>3.1.21</td>
<td>D-A converter</td>
<td>70</td>
</tr>
<tr>
<td>3.2</td>
<td>Programming tips</td>
<td>71</td>
</tr>
<tr>
<td>3.2.1</td>
<td>The run interpreter</td>
<td>71</td>
</tr>
<tr>
<td>3.2.2</td>
<td>System functions</td>
<td>72</td>
</tr>
<tr>
<td>3.2.3</td>
<td>Control of the automatic mode via external inputs</td>
<td>74</td>
</tr>
<tr>
<td></td>
<td>Program structure</td>
<td>78</td>
</tr>
<tr>
<td>3.3</td>
<td>Programming elements</td>
<td>79</td>
</tr>
<tr>
<td>3.4</td>
<td>Explanations</td>
<td>89</td>
</tr>
<tr>
<td>3.5</td>
<td>ln.m - Waiting for logical status of input</td>
<td>91</td>
</tr>
<tr>
<td>3.6</td>
<td>On.m - Waiting for logical status of output</td>
<td>92</td>
</tr>
<tr>
<td>3.7</td>
<td>Mn.m - Waiting for then logical status of a flag</td>
<td>93</td>
</tr>
<tr>
<td>3.8</td>
<td>Ni.n - Waiting for the logical status of an N register</td>
<td>94</td>
</tr>
<tr>
<td>3.9</td>
<td>On:=m - Set / reset output</td>
<td>95</td>
</tr>
<tr>
<td>3.10</td>
<td>Mn:=m - Set / reset flag</td>
<td>96</td>
</tr>
<tr>
<td>3.11</td>
<td>Tn - Hold-up time</td>
<td>97</td>
</tr>
<tr>
<td>3.12</td>
<td>JMP - Unconditional jump</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>3.12.1 General form of the unconditional jump</td>
<td>97</td>
</tr>
<tr>
<td></td>
<td>3.12.2 Start of an interpolation at a specified program line</td>
<td>99</td>
</tr>
<tr>
<td></td>
<td>3.12.3 Start of a program at a specified program line</td>
<td>100</td>
</tr>
<tr>
<td>3.13</td>
<td>SUB - Subroutine call</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>3.13.1 Standard subroutine call</td>
<td>101</td>
</tr>
<tr>
<td></td>
<td>3.13.2 Subroutine call, Program name is in a string</td>
<td>103</td>
</tr>
<tr>
<td></td>
<td>3.13.3 Subroutine call, program name is in an integer register</td>
<td>104</td>
</tr>
<tr>
<td>3.14</td>
<td>G11 – Switch display on / off</td>
<td>105</td>
</tr>
<tr>
<td>3.15</td>
<td>G21 - Conditional jump</td>
<td>106</td>
</tr>
<tr>
<td>3.16</td>
<td>G22 - Conditional Subroutine call</td>
<td>108</td>
</tr>
<tr>
<td>3.17</td>
<td>CASE.JMP - Jump distributor</td>
<td>110</td>
</tr>
<tr>
<td>3.18</td>
<td>CASE.SUB - Subroutine distributor</td>
<td>112</td>
</tr>
<tr>
<td>3.19</td>
<td>DEC - Loop with conditional jump</td>
<td>114</td>
</tr>
<tr>
<td>3.20</td>
<td>INC - Loop with conditional jump</td>
<td>116</td>
</tr>
</tbody>
</table>
3.21 BREAK - Cancel automatic mode
3.22 Communication with programs in parallel run
  3.22.1 RUN – Start of a parallel run
  3.22.2 SLEEP - Stopping a parallel run
  3.22.3 CANCEL - Ending a parallel run
  3.22.4 STOP - Stop automatic run
  3.22.5 START - Start automatic run
  3.22.6 CASE.RUN - Starting parallel runs with CASE
  3.22.7 CASE.SLEEP - Stopping of parallel runs with Case
  3.22.8 CASE.CANCEL - Ending parallel runs with Case
  3.22.9 PROGSTAT - Get program status
3.23 A1 - Positioning of the axes
3.24 Get axis position
3.25 Request for the set or current speed
  3.25.1 Set traversing speed of an axis
  3.25.2 Get current path speed
  3.25.3 Get the current speed of an axis during an interpolation
  3.25.4 Get the set path speed of an axis
3.26 Get following error
3.27 ENC - Transfer encoder position
3.28 Wait for position from the SSI interface
3.29 FAn - Traversing speed
  3.29.1 Particular features of the PA-CONTROL MP / LV-servoTEC:
3.30 G25 - Approach to reference point
3.31 G26 - Set position to zero / set to position value
3.32 G29 - Set position to dimension
3.33 G90 - Absolute dimension system
3.34 G91 - Incremental dimension system
3.35 G100 - Specifying the acceleration
3.36 G101 - Changing the motor current
3.37 G123 - Traverse as long as condition fulfilled
3.38 Write ServoTEC-Parameters
3.39 Reading ServoTEC parameters
3.40 Measuring mode for axis
  3.40.1 G140 – Activation of the measuring mode
  3.40.2 G141 - Deactivation of the measuring mode
3.41 Limit switch monitoring
  3.41.1 G142 Limit switch monitoring "OFF"
  3.41.2 G143 - Limit switch monitoring "ON"
3.42 G 150 - Traverse segment with Start-Stop
3.43 Linear interpolation
  3.43.1 G01 – Linear interpolation with 4 from 16 axes
3.43.2 Abort interpolation 165
3.44 G16x – Operation of outputs during interpolation 167
  3.44.1 G160 - Activate outputs before starting the interpolation 167
  3.44.2 G161 / G162 - Activate outputs during interpolation 168
3.45 G170 - Read character from PTX file 169
3.46 G171 - Write character to PTX file 170
3.47 G172 - Write line from PTX file to string (Sn) 171
3.48 G173 - Store string (Sn) in a line of a PTX file 172
3.49 STORE - Storage of values 173
3.50 CASE.STORE - Storage of values 175
3.51 Commands of the G2xx group 177
  3.51.1 G210 – Activate the start positioning mode 178
  3.51.2 G211 - Position-conditional jump 179
  3.51.3 G212 - Position-conditional subroutine call 181
  3.51.4 G213 - Activate the standard positioning mode 183
  3.51.5 G221 - Position-conditional jump (current position) 184
  3.51.6 G222 - Position-conditional subroutine call (current pos.) 186
  3.51.7 G230 - Wait until current position <> than value 188
3.52 G4xx - Time monitoring commands 190
  3.52.1 G421 - Time monitoring with a conditional jump 191
  3.52.2 G422 - Time monitoring with conditional subroutine call 192
  3.52.3 G423 - with conditional subroutine call (return in same line) 193
  3.52.4 G401 - Reset of the time condition 194
3.53 Real-time clock 195
  3.53.1 Read-out of the real-time clock 195
  3.53.2 Set the date from the user program 197
  3.53.3 Set the time from the user program 198
3.54 G5xx – Text and value output via the current data channel 199
  3.54.1 G500 - Selection of the data channel / Initialisation of the ports 200
  3.54.2 G501 - Clear the display 202
  3.54.3 G502 - Clear up to end of line 203
  3.54.4 G503/G504 - Position the cursor 204
  3.54.5 G510 - Text output 206
  3.54.6 G511 - Text output 207
  3.54.7 G512 - Output of control characters 208
  3.54.8 G515 - Text output 209
  3.54.9 G520 - Output of value 210
  3.54.10 G521 - Output of value 212
3.55 G5?? Text- / Value transfer from the current data channel 214
  3.55.1 G531 - Value transfer 214
  3.55.2 G532 - Character transfer to the local character buffer S0 216
  3.55.3 G533 - Character transfer in the background 217
3.55.4  G534 - Character transfer in the background 219
3.55.5  CHN - Check whether character transfer in the background has been completed221

3.56 Data transfer to and from the local character buffer 222
3.56.1  Sn - Copying character strings 222
3.56.2  S0 - Copying character strings 223
3.56.3  POS - Search for character position in local character buffer S0 226
3.56.4  Copy - Convert characters from the local character buffer 227
3.56.5  Merging strings 229
3.56.6  Get the length of a string 229
3.56.7  Write the content of a register (number) to a string 230
3.56.8  GET - Transfer the content of the local character buffer S0 to an integer register231
3.56.9  GETI - Transfer the content of the local character buffer S0 to an integer register in INTEL format 232
3.56.10 PUT – Transfer the content of an N register to the local character buffer 233
3.56.11 PUTI - Transfer the content of an N register in the INTEL format to the local character buffer S0 234

3.57 Character transfer from the keyboard 235
3.57.1  G540 - Check if a key has been actuated 235
3.57.2  G541 - Get a character from the keyboard 236
3.57.3  G542 - Entry of a value via the keyboard 237

3.58 G6?? - Image to register / Image from register 239
3.58.1  G600 - Binary notation to outputs 240
3.58.2  G601 - BCD notation to outputs 241
3.58.3  G602 - Binary notation to flags 242
3.58.4  G603 - Inputs in binary format to register 244
3.58.5  G604 - Flags in binary format to register 245

3.59 Arithmetic operations 246
3.59.1  Rn/Nn:= Load register 247
3.59.2  Rn:=Ran - Load register with axis parameters 248
3.59.3  Request for the current line number of a program 249
3.59.4  Addition 250
3.59.5  Subtraction 251
3.59.6  Multiplication 252
3.59.7  Division 253
3.59.8  SIN/ASIN - Sine functions 254
3.59.9  COS/ACOS - Cosine functions 255
3.59.10 TAN/ATAN - Tangent functions 256
3.59.11 SQRT - Root function 257
3.59.12 INT - Integer component 258
3.59.13 FRAC - Decimal component of a real number 259
3.59.14 ABS - Absolute value of a real number / integer 260
3.59.15 Bit-by-bit processing of N registers 261

3.60 Comparison operations 262
3.60.1 Comparisons
3.60.2 Complex examples

3.61 Logic operations
3.61.1 LD/AND/OUT/NOT - Logic AND operation
3.61.2 LD/OR/OUT/NOT - Logic OR operation
3.61.3 SET/RES - Supplemental instructions for logic operations
3.61.4 EOR logic operation (Exclusive-OR)
3.61.5 Multi-level logic AND operation
3.61.6 OR-LD - Multi-level logic OR operation
3.61.7 Complex logic operation

3.62 Analog-to-digital converter
3.62.1 Get A-D values
3.62.2 G18? – Acquisition of several A-D values
3.62.3 G180 - A-D values synchronous to the axis movement
3.62.4 G181 - Acquire A-D values in a defined time base

3.63 Digital-to-analog converter
3.63.1 Output of D-A values

3.64 Commands for devices on the CANopen bus
3.64.1 General
3.64.2 Network management commands
3.64.3 Functional monitoring of the devices
3.64.4 Commands for checking device errors
3.64.5 Commands for processing service data objects
3.64.6 Commands for the processing of process data objects
3.64.7 Typical applications

4 Startup
4.1 Important Information
4.2 Installation of a PA-CONTROL servoTEC
4.3 Wiring of the connections
4.3.1 Wiring structure of the PA-CONTROL servoTEC
4.3.2 Cables
4.3.3 Optional accessories
4.4 Connector Assignments
4.4.1 Inputs and outputs
4.4.2 Limit switches
4.4.3 Diagnosis port
4.4.4 Rotation monitoring
4.4.5 Brake connector
4.4.6 COM1 RS232
4.4.7 CAN BUS
4.4.8 Profibus Option
4.5 Function and status check
   4.5.1 Functional check via a connected IEF control console
   4.5.2 Functional check of the device with WINPAC

5 Parameters
   5.1 General information on parameters
   5.2 Parameter description
      5.2.1 System parameters
      5.2.2 Diagnosis
      5.2.3 CANopen-Bus
      5.2.4 RS232 port
      5.2.5 Profibus
      5.2.6 Axis and drive parameters

6 Options
   6.1 Options of the PA-CONTROL servoTEC
   6.2 Plug-in connections of the PA-CONTROL servoTEC
   6.3 CANopen interface
      6.3.1 CANopen devices with a permanent assignment
      6.3.2 CANopen devices without a permanent assignment
      6.3.3 Overview of the CAN IDs
   6.4 RS 232 port driver
   6.5 IEF module RS 232 port
   6.6 IEF module Profibus DP
   6.7 IEF module A-D converter
   6.8 IEF module SSI interface
      6.8.1 General
      6.8.2 Detection and assignment
      6.8.3 Parameters
      6.8.4 Application
      6.8.5 WINPAC and the absolute positioning System
   6.9 IEF control console
      6.9.1 General
      6.9.2 IEF control console - version for panel mounting
      6.9.3 IEF control console - standard version
   6.10 SÜTRON control console via CANopen Bus
      6.10.1 CAN addresses and monitoring
      6.10.2 TSwin Configuring software
      6.10.3 List of variables in the control console
      6.10.4 Serial signalling system with polling range
      6.10.5 Parallel signalling system
      6.10.6 Simulation of the PA-CONTROL front plate
7 Technical Appendix

7.1 Messages of the PA-CONTROL servoTEC
   7.1.1 Status messages of the LV servoTEC
   7.1.2 Status LED of the CPU5
   7.1.3 Status and operational status messages
   7.1.4 Error and fault signals

7.2 Basic board
   7.2.1 Voltage supply

7.3 Connector assignments
   7.3.1 Inputs and outputs

7.4 List of accessories and spare parts

7.5 Service
   7.5.1 Exchange of the battery

7.6 Service
   7.6.1 Exchange of the battery

7.7 PA-CONTROL key code (+ ASCII character set to ISO/IEC 8859-15)

INDEX Fehler! Verweisquelle konnte nicht gefunden werden. Fehler! Textmarke nicht definiert.
1 Technical Short Information

1.1 Design of the Device

The PA-CONTROL CPU5 is a controller which was especially developed for installation inside an LV-servoTEC servo amplifier. The PA-CONTROL servoTEC comes into being through the combination of the two devices.

Performance features:

- Comprehensive, high-performance command set
- Combines the advantages of NC-controls with those of an SPC
- 8 opto-decoupled inputs
- 5 opto-decoupled outputs
- Processing of I/O is possible during positioning
- 1 serial diagnosis port (RS232)
- 31 parallel programs with subroutine technology
- 16 axes can be controlled in an interconnected system by means of a CANopen bus

Options:

- IEF control console
- SÜTRON CANopen control console
- WINPAC software package
- The following IEF modules:

<table>
<thead>
<tr>
<th>IEF module</th>
<th>Slot 1 (J2)</th>
<th>Slot 2 (J3)</th>
</tr>
</thead>
<tbody>
<tr>
<td>RS232</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>IEF A-D converter</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>PROFIBUS</td>
<td></td>
<td>x</td>
</tr>
</tbody>
</table>

The PA-CONTROL servoTEC can either be programmed by means of the WINPAC software package or via the IEF control console. The IEF control console is composed of a complete alphanumeric keyboard and a 2-line illuminated LC display (2*40 characters). A key-operated switch on the IEF control console prevents unauthorised access.

The WINPAC software package enables comfortable online diagnosis, manual operation, and offline programming.

The following description refers to a PA-CONTROL servoTEC operated by the WINPAC software package. Please refer to Chapter 6.9 "Options", Section "IEF control console" of these operating instructions for a description of operation using the IEF control console.
1.2 Declaration of Conformity

IEF Werner GmbH
Wendelhofstraße 6
D-78120 Furtwangen
Telephone: +49 (0)7723 /925-0
Telefax: +49 (0)7723/925-100

We hereby declare that the products referred to in the following

<table>
<thead>
<tr>
<th>Designation of the products</th>
<th>Product nos.</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA-CONTROL-Single</td>
<td>1000244</td>
</tr>
<tr>
<td>PA-CONTROL-Compact</td>
<td>1000245</td>
</tr>
<tr>
<td>PA-CONTROL-Steuergerät</td>
<td>1000243</td>
</tr>
<tr>
<td>PA-CONTROL-MP</td>
<td>1000759</td>
</tr>
<tr>
<td>PA-CONTROL-servoTEC</td>
<td>1000598</td>
</tr>
<tr>
<td>PA-CONTROL smart</td>
<td>1000927</td>
</tr>
</tbody>
</table>

are intended for installation in a machine. The initial plant start-up is prohibited until it has been ascertained that the machine in which these products are to be installed complies with the provisions of EC Directive 91/368/EEC

The following standards were applied:

89/336/EEC                 Electromagnetic Compatibility
EN 50081-1                Generic Emission Standard
EN 50082-2                Generic Immunity Standard
pr EN 50082-1             Generic Immunity Standard
pr EN 50082-2

Test procedures:

IEC 801-2 / Level 3        Immunity against Electrostatic Discharges (ESD)
IEC 801-3 / Level 3        Immunity against Electrostatic Fields
IEC 801-4 / Level 3        Immunity against Fast Transients (Burst)
IEC 801-5 / Level 3        Immunity against Surge Voltages (Surge)
EN 55011 / Class B         Limits and Methods of Measurement of Radio Disturbance Characteristics

Furtwangen, 01.12.2002     (Manfred Bär, managing director)

This declaration does not include a guarantee of properties. The safety and protection information in the operating manual must always be followed.

IEF Werner GmbH has the following technical documentation available for inspection:

<table>
<thead>
<tr>
<th>User</th>
<th>EU authorities</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

• Operating manual
• Operator software
• Plans / software source code
• Description of the measures to guarantee conformity
• Test records
• Other technical documentation
1.3 Safety Instructions

Observe the instructions and warnings contained in the operating instructions. These are identified as follows:

**WARNING**

If these instructions and warnings are disregarded, hazards may arise from dangerous electrical voltages.

**CAUTION**

This sign indicates warnings and information about general hazards

1.3.1 Connection Instructions

**WARNUNG**

Observe the discharging time for the condensers

After disconnecting the control system, the condensers are charged with a high voltage for a short period of time. Do not handle the device until 5 minutes after switching it off.

Installation and start-up should only be carried out by qualified personnel.

Observe the general installation regulations for assembling and operating electrical operating systems (EN 60204).

Protective equipment for humans and machine should comply with local conditions and regulations.

Do not connect or disconnect power or control cables while the control system is still connected to the mains.

Plugs may only be connected and/or removed when voltage-free.

Control and power lines must be laid separately (ca 10 cm apart)

Protective system IP 20

**WARNING**

The controller must be deenergized when carrying out assembly, disassembly and repair work or replacing individual components or switching over the operating voltage.
1.3.2 Ambient Conditions

The limit values for the ambient temperature (min. 0°C to max. 40°C) must be adhered to.
No mist or water may be allowed to enter the controller.
Dust must be prevented from entering the controller.
The controller must be protected from aggressive gases and liquids.
Ensure that air can circulate freely (air inlet and outlet ports must be kept free).

1.3.3 Operating Instructions

CAUTION

It is imperative that the parameter values are checked for compliance with the actual conditions.

When using the program examples contained in the operating instructions, always check whether the program can be implemented on the machine in question.
1.4 Technical Data

<table>
<thead>
<tr>
<th>Specification</th>
<th>Details</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ambient temperature:</td>
<td>(0 to 40) °C</td>
</tr>
<tr>
<td>Connecting voltages:</td>
<td>24VDC, (18 to 28 VDC)</td>
</tr>
<tr>
<td>Power consumption:</td>
<td>max. 20 VA</td>
</tr>
<tr>
<td>8 inputs:</td>
<td>• Opto-decoupled</td>
</tr>
<tr>
<td></td>
<td>• 24 VDC</td>
</tr>
<tr>
<td></td>
<td>• Typ. current 5mA</td>
</tr>
<tr>
<td></td>
<td>• Low level (0 - 3) VDC</td>
</tr>
<tr>
<td></td>
<td>• High level (10 - 30) VDC</td>
</tr>
<tr>
<td>5 outputs:</td>
<td>• Opto-decoupled switching positive</td>
</tr>
<tr>
<td></td>
<td>• 4 VDC / 0.5A (ohmic load)</td>
</tr>
<tr>
<td>1 serial diagnosis port:</td>
<td>RS 232, (only for PC with WINPAC)</td>
</tr>
<tr>
<td>Data security:</td>
<td>Lithium battery (&gt; 5 years)</td>
</tr>
<tr>
<td>Protection type:</td>
<td>IP20</td>
</tr>
<tr>
<td>Fitting dimensions W x H x D:</td>
<td>(30 x 150 x 125) mm</td>
</tr>
<tr>
<td>Weight:</td>
<td>0.25 kg</td>
</tr>
</tbody>
</table>

1.4.1 Inputs and outputs of the PA-CONTROL servoTEC

In addition to the 8 inputs and 5 outputs available on the CPU5, further inputs and outputs extend the options of this controller. These options result from the various combinations of the I/O modules connected to the LV-servoTEC or the CANopen bus.

A maximum of 16 axes can be connected to a PA-CONTROL servoTEC. Any LV-servoTEC connected via CANopen bus extends the number of available inputs as per the table below. Only the inputs of existing axes are available.

<table>
<thead>
<tr>
<th>Axis number</th>
<th>LV servoTEC on CANopen</th>
<th>Axis number</th>
<th>LV servoTEC on CANopen</th>
</tr>
</thead>
<tbody>
<tr>
<td>used *1</td>
<td>used *1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>721 – 724</td>
<td>9</td>
<td>849 – 852</td>
</tr>
<tr>
<td>2</td>
<td>737 – 740</td>
<td>10</td>
<td>865 – 868</td>
</tr>
<tr>
<td>3</td>
<td>753 – 756</td>
<td>11</td>
<td>881 – 884</td>
</tr>
<tr>
<td>4</td>
<td>769 – 772</td>
<td>12</td>
<td>897 – 900</td>
</tr>
<tr>
<td>5</td>
<td>785 – 788</td>
<td>13</td>
<td>913 – 916</td>
</tr>
<tr>
<td>6</td>
<td>801 – 804</td>
<td>14</td>
<td>929 – 932</td>
</tr>
<tr>
<td>7</td>
<td>817 – 820</td>
<td>15</td>
<td>945 – 948</td>
</tr>
<tr>
<td>8</td>
<td>833 – 836</td>
<td>16</td>
<td>961 – 964</td>
</tr>
</tbody>
</table>

*1 The four inputs are at Digital-IN1, Digital_IN2, PSTOP, NSTOP and ENABLE
Inputs of the LV-servoTEC and their addressing in PA-CONTROL

<table>
<thead>
<tr>
<th>Meaning on the LVservoTEC</th>
<th>A1</th>
<th>A2</th>
<th>A3</th>
<th>…</th>
<th>A16</th>
</tr>
</thead>
<tbody>
<tr>
<td>Digital IN1</td>
<td>I 721</td>
<td>I 737</td>
<td>I 753</td>
<td>…</td>
<td>I 961</td>
</tr>
<tr>
<td>Digital IN2 (Reference)</td>
<td>I 722</td>
<td>I 738</td>
<td>I 754</td>
<td>…</td>
<td>I 962</td>
</tr>
<tr>
<td>IN3 (PSTOP)</td>
<td>I 723</td>
<td>I 739</td>
<td>I 755</td>
<td>…</td>
<td>I 963</td>
</tr>
<tr>
<td>IN4 (NSTOP)</td>
<td>I 724</td>
<td>I 740</td>
<td>I 756</td>
<td>…</td>
<td>I 964</td>
</tr>
<tr>
<td>ENABLE</td>
<td>I 725</td>
<td>I 741</td>
<td>I 757</td>
<td>…</td>
<td>I 965</td>
</tr>
<tr>
<td>Reserve</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Traversing job active</td>
<td>I 729</td>
<td>I 745</td>
<td>I 761</td>
<td>…</td>
<td>I 969</td>
</tr>
<tr>
<td>Reference position set</td>
<td>I 730</td>
<td>I 746</td>
<td>I 762</td>
<td>…</td>
<td>I 770</td>
</tr>
<tr>
<td>Current position = Home position</td>
<td>I 731</td>
<td>I 747</td>
<td>I 763</td>
<td>…</td>
<td>I 771</td>
</tr>
<tr>
<td>In position</td>
<td>I 732</td>
<td>I 746</td>
<td>I 764</td>
<td>…</td>
<td>I 772</td>
</tr>
<tr>
<td>Speed = 0</td>
<td>I 733</td>
<td>I 749</td>
<td>I 765</td>
<td>…</td>
<td>I 773</td>
</tr>
<tr>
<td>Safety relay has actuated</td>
<td>I 734</td>
<td>I 750</td>
<td>I 766</td>
<td>…</td>
<td>I 774</td>
</tr>
<tr>
<td>Output stage enabled</td>
<td>I 735</td>
<td>I 751</td>
<td>I 767</td>
<td>…</td>
<td>I 775</td>
</tr>
<tr>
<td>Error is active</td>
<td>I 736</td>
<td>I 752</td>
<td>I 768</td>
<td>…</td>
<td>I 776</td>
</tr>
</tbody>
</table>

From V 5.00

Inputs via I/O modules on the CANopen bus

<table>
<thead>
<tr>
<th>Input number</th>
<th>Use</th>
<th>Special application</th>
</tr>
</thead>
<tbody>
<tr>
<td>1025 - 1536</td>
<td>Inputs via CANopen I/O modules</td>
<td></td>
</tr>
<tr>
<td>1282 - 2048</td>
<td>Not used at present</td>
<td></td>
</tr>
</tbody>
</table>

Outputs via I/O modules on the CANopen bus

<table>
<thead>
<tr>
<th>Output number</th>
<th>Use</th>
<th>Special application</th>
</tr>
</thead>
<tbody>
<tr>
<td>1025 - 1536</td>
<td>Outputs via CANopen I/O modules</td>
<td></td>
</tr>
<tr>
<td>1282 - 2048</td>
<td>Not used at present</td>
<td></td>
</tr>
</tbody>
</table>
1.5 Installation of the CPU5

Fig. 1: Outside measurements, position and designation of the plug-in connections

Fig. 2: PA-CONTROL servoTEC
2 Operator interface

2.1 The keyboard

Note

In principle, each variant of PA-CONTROL can be operated with or without a keyboard.
While the ex works equipment of PA-CONTROL Single, Compact and Steuer usually includes a keyboard integrated in the front panel, this possibility is not provided for the PA-CONTROL servoTEC or PA-CONTROL MP versions. However, these controllers can also be provided with this convenient display and input output tool through connection of the optional IEF control console. See Chapter 6, page 345

As an alternative to connection of the IEF control, a control console can be connected via CANopen bus. The front plate of PA-CONTROL can be simulated by means of this console.

![Fig. 3: The keyboard of PA-CONTROL](image)

The key of the key-operated switch has been removed. This is only possible in the vertical position. As a result, PA-CONTROL is in the "Automatic" mode. Other modes are only accessible with the key inserted (i.e. in the horizontal position). The diagnosis port is located at the bottom left.
PA-CONTROL has a complete alphanumeric keyboard (0-9, A-Z). There are also several special keys, whose functions are described in the following:

<table>
<thead>
<tr>
<th>Special key</th>
<th>Name</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESC</td>
<td>ESC key</td>
<td>Exit sub-menu, cancel input field</td>
</tr>
<tr>
<td>ALT</td>
<td>ALT key</td>
<td>no function assigned</td>
</tr>
<tr>
<td>CTRL</td>
<td>CTRL key</td>
<td>no function assigned</td>
</tr>
<tr>
<td>SHIFT</td>
<td>SHIFT key</td>
<td>Double assignment of keys, auxiliary key for cursor keys</td>
</tr>
<tr>
<td>↓</td>
<td>Arrow down key</td>
<td>Scrolling in the menu to next line (with autorepeat)</td>
</tr>
<tr>
<td>↑</td>
<td>Arrow up key</td>
<td>Scrolling in the menu to previous line (with autorepeat)</td>
</tr>
<tr>
<td>→</td>
<td>Arrow right key</td>
<td>Cursor one character to the right in the input field (with autorepeat)</td>
</tr>
<tr>
<td>←</td>
<td>Arrow left key</td>
<td>Cursor one character to the left (with autorepeat)</td>
</tr>
<tr>
<td></td>
<td>Space bar</td>
<td>Inserts space in the editor</td>
</tr>
<tr>
<td>ENTER</td>
<td>Enter key</td>
<td>End input, end line in the editor, insert new line</td>
</tr>
<tr>
<td>INS</td>
<td>INS key</td>
<td>no function assigned</td>
</tr>
<tr>
<td>DEL</td>
<td>DEL key</td>
<td>Deletes the character on which the cursor is positioned</td>
</tr>
<tr>
<td>START</td>
<td>Start key</td>
<td>Starts automatic mode</td>
</tr>
<tr>
<td>STOP</td>
<td>Stop key</td>
<td>Stops automatic mode</td>
</tr>
<tr>
<td>Automatic</td>
<td>Key-operated switch</td>
<td>Access lock for front plate (the key is removable in the position shown, only &quot;Automatic&quot; is possible)</td>
</tr>
</tbody>
</table>
2.2 General operation

Note The menu structure differs depending on the equipment of the controller or the device type.

2.2.1 Switch-on of PA-CONTROL

After PA-CONTROL has been switched on, it first of all checks the hardware configuration. The extent of this check, and therefore also the messages which may be displayed, obviously depends on the configuration of PA-CONTROL. If a change is discovered, the following is displayed:

E127: Axis 1: new axis detected
error reset? 1=yes

If 1 is entered, i.e. Error Reset, PA-CONTROL displays the following menu:

Error on System – Re-initialize
1 = reset error on system
2 = RE-initialize PA CONTROL

Note Menu item 2 can only be reached by means of the key-operated switch in the “Program” position.

If the hardware configuration is correct, the devices on the CANopen bus are checked in the next step. The following is displayed during this check:

Wait until all axis initialized

If an error occurs during the initialization, e.g. CAN axes not connected or a cable error, the following message is displayed after approx. 10s.

A 1: Time-Out by waiting on axis
1 = repeat
2 = Cancel, continue without this axis
3 = Cancel all, continue without axes
After successful initialization, a check of the parameters for all the axes in an LV-servoTEC is carried out. If this check finds any differences, the following error message appears on the display, and the flashing 7-segment display on the LV-servoTEC also provides information about the error.

<table>
<thead>
<tr>
<th>A1 : Parameters are different</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = send to all axes</td>
</tr>
</tbody>
</table>

2 = ignore all axes  
3 = apply all axes  
4 = Ignore  
5 = apply parameters from SLAVE  
6 = send PA-CONTROL parameters to SLAVE

As a rule, the main menu appears on the display at the end of the switch-on sequence:

<table>
<thead>
<tr>
<th>PA-CONTROL 4.xx</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = Automatic</td>
</tr>
</tbody>
</table>

The operator interface of PA-CONTROL is arranged in a menu structure. The structure and handling of the main menu and the various sub-menus are based on the same principle.

### 2.2.2 The menu principle

<table>
<thead>
<tr>
<th>Title of the menu</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = First sub-menu item</td>
</tr>
</tbody>
</table>

2 = Second sub-menu item  
3 = Third sub-menu item  
...  
...  
9 = Ninth sub-menu item

The number of menu sub-options in the individual menus was customized to the respective requirements and restricted accordingly. If an arrow is displayed on the far right of the second line, a secondary menu item is available in this menu.

Since only two lines are available on the display, only the title of the current menu and one of the menu sub-options is displayed in each case.
Navigating through the menu interface:

<table>
<thead>
<tr>
<th>Key:</th>
<th>Action/effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow down/Arrow right</td>
<td>next menu sub-option is displayed</td>
</tr>
<tr>
<td>Arrow up</td>
<td>previous menu sub-option is displayed</td>
</tr>
<tr>
<td>SHIFT + Arrow up</td>
<td>first menu sub-option is displayed</td>
</tr>
<tr>
<td>SHIFT + Arrow down</td>
<td>last menu sub-option is displayed</td>
</tr>
<tr>
<td>ENTER</td>
<td>The displayed menu sub-option is activated</td>
</tr>
<tr>
<td>ESC</td>
<td>Sub-menu is exited and the previous menu returned to</td>
</tr>
<tr>
<td></td>
<td>(The first menu line is displayed in the main menu)</td>
</tr>
</tbody>
</table>

The input of the number prefixed to the menu item is another way of activating a menu item.

2.2.3 The input field

If the user is requested to enter a numerical value (parameter value, register value, or similar), an input field appears on the display.

The input field is bounded by two square brackets [ _......]. The current value and cursor are located between these two brackets (blinking field in the display).

The user can update the entry by pressing the relevant keys.

Assignment of the keys in the input field:

<table>
<thead>
<tr>
<th>Key:</th>
<th>Reaction/effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow left</td>
<td>Cursor one character to the left</td>
</tr>
<tr>
<td>Arrow right</td>
<td>Cursor one character to the right</td>
</tr>
<tr>
<td>SHIFT + DEL</td>
<td>Deletes character on the left of the cursor</td>
</tr>
<tr>
<td>DEL</td>
<td>Deletes character on which the cursor is positioned</td>
</tr>
<tr>
<td>ENTER</td>
<td>Conclusion of an entry; the entered value is checked and accepted if it is within the permitted limits. If it is below or exceeds the limits, an error message is output and a new input expected.</td>
</tr>
<tr>
<td>ESC</td>
<td>Cancellation of the input, the old value is retained</td>
</tr>
</tbody>
</table>
2.2.4 Entry of a program name

If the controller expects the entry of a program name, the following appears in the display:

```
Please enter new program name!
[ _                  ]
```

The cursor (blinking field in the display) is positioned at the first position of the input field. The user now enters the letters or numbers of the desired program name. After all the characters of the program name have been entered, the input is concluded by the pressing the "ENTER" key.

The input program name must fulfil certain conditions (see below) and is checked for these. In the event of a violation, the following message is displayed:

```
ERROR : Program name not valid
Program : 3STAR%T
```

The error message is acknowledged by pressing a key, and the user can correct the error in the input field.

The program name must fulfil the following conditions:

- A character can be a number or a letter
- Spaces are not allowed in the name
- The length is restricted to 20 characters
- Only "_" and "-" are practical as special characters
- Uppercase letters

Moving the cursor in the input field:

<table>
<thead>
<tr>
<th>Key:</th>
<th>Reaction/effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow left</td>
<td>Cursor one character to the left</td>
</tr>
<tr>
<td>Arrow right</td>
<td>Cursor one character to the right</td>
</tr>
<tr>
<td>SHIFT + DEL</td>
<td>Deletes character on the left of the cursor</td>
</tr>
<tr>
<td>DEL</td>
<td>Deletes character on which the cursor is positioned</td>
</tr>
<tr>
<td>ENTER</td>
<td>Conclusion of an entry</td>
</tr>
</tbody>
</table>
2.2.5 Selection of a name from the program list

If the user wishes to access an existing program (e.g. change of the program, selection to define as the start-up program or similar), the user is provided with a list of the existing programs in alphabetical order with the possible program types.

The user has two possibilities to make a selection:

1. The user can scroll in the list by means of the "Arrow up" and "Arrow down" keys until the desired program is displayed. The selection is concluded by pressing the "ENTER key" and the program is transferred for the action.

2. An empty input field ( [ ] ) is shown at the end of the first display line. The user can input the name of the desired program in this input field via the keyboard. The characters from the keyboard are only applied if a program name exists with the same character string. The display (second display line) is corrected according to the input field after each input character.

The selection is concluded by pressing the "ENTER key" and the program is transferred for the action.

The contents of the input field are deleted by pressing the "Arrow down" or "Arrow up" keys.

2.3 Menu structure of PA-CONTROL

The menu items from item 2 are only accessible with the key switch in the horizontal position.

<table>
<thead>
<tr>
<th>Main menu:</th>
<th>Menu sub-options:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = Automatic</td>
<td>- Start</td>
</tr>
<tr>
<td></td>
<td>- Stop</td>
</tr>
<tr>
<td>2 = Manual</td>
<td>1 = Approach to reference point</td>
</tr>
<tr>
<td></td>
<td>2 = Move axes manually via front plate</td>
</tr>
<tr>
<td>3 = Programming</td>
<td>1 = Display program directory</td>
</tr>
<tr>
<td></td>
<td>2 = Create new program</td>
</tr>
<tr>
<td></td>
<td>3 = Alter existing program</td>
</tr>
<tr>
<td></td>
<td>4 = Copy program</td>
</tr>
<tr>
<td></td>
<td>5 = Rename program</td>
</tr>
<tr>
<td></td>
<td>6 = Delete program</td>
</tr>
<tr>
<td></td>
<td>7 = Display program memory occupancy</td>
</tr>
<tr>
<td>4 = Diagnosis</td>
<td>1 = Limit switch / stand-by</td>
</tr>
<tr>
<td></td>
<td>2 = Inputs (I)</td>
</tr>
<tr>
<td></td>
<td>3 = Outputs (O)</td>
</tr>
<tr>
<td></td>
<td>4 = Flag (M)</td>
</tr>
<tr>
<td>Number</td>
<td>Description</td>
</tr>
<tr>
<td>--------</td>
<td>-------------</td>
</tr>
<tr>
<td>5</td>
<td>Run definitions - Start program - Program at STOP - Program START after STOP - Program at MALFUNCTION - Title</td>
</tr>
<tr>
<td>6</td>
<td>Parameters 1 = Edit system parameters 2 = Edit axis parameters 3 = Edit ASi BUS * 4 = Edit servoTEC parameters</td>
</tr>
<tr>
<td>7</td>
<td>Basic settings 1 = Boot system parameters 2 = Boot axis parameters 3 = Delete program memory 4 = Re-initialize PA-CONTROL</td>
</tr>
<tr>
<td>8</td>
<td>System diagnosis 1 = Display hardware configuration 2 = Clock 3 = Keyboard test 4 = Stop key test 5 = Start key test 6 = Key switch test 7 = Test COM ports</td>
</tr>
<tr>
<td>9</td>
<td>Communication via modem 1 = Activate connection 2 = Display connection status 3 = Terminate connection 4 = Edit modem settings 5 = Select modem type</td>
</tr>
</tbody>
</table>

* only if ASi bus available
2.4 Automatic

The program defined as the start program is executed in automatic mode. Further programs can be invoked as subprograms or programs which are to be processed in parallel. A running program may be interrupted by "STOP". A program is started or an interrupted program is continued by means of "START". The required input is assigned at parameter level for the functions external START and external STOP (see Chapter 5 Parameters).

2.5 Manual

Overview of the manual menu

After selecting this menu item, the user is offered the following choice:

<table>
<thead>
<tr>
<th>Manual</th>
<th>1 = Approach reference point</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 = Axes manual via front plate</td>
</tr>
</tbody>
</table>

2.5.1 Approaches to reference point

After selecting this menu item, the user is informed:

- "Wait until initialized (ESC=Cancel)"
- "Axis : . . . . . . . . . . (START=Continue)"

If an error occurs during the initialization of the axis, the following message is output via the display:

```
E900: A1 Parameter "VBUSBAL" different
Error, continue <key>
```

If there is no error, the select menu for the axes is displayed:

<table>
<thead>
<tr>
<th>Approach to reference point</th>
<th>1 = Axis 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 = Axis 2</td>
<td></td>
</tr>
<tr>
<td>.</td>
<td></td>
</tr>
<tr>
<td>n = Axis n</td>
<td></td>
</tr>
</tbody>
</table>
The number of axes available for selection depends on the configuration of PA-CONTROL (1, 2, 4, 8, 12, 16). After selection of the axis, the following is displayed:

```
Approach reference point axes : 1
Start = <START>,  Aboard = <key>
```

The user can start the approach to reference point by means of the "Start key", or return to the manual menu using any key. The following is displayed during the approach to reference point:

```
Approach to reference point axis : 1
Running, cancel with <STOP>
```

The approach to reference point can be cancelled at any time using the "Stop key". If there is a malfunction, the following message may appear:

```
Approach to reference point axis :1 E563
Limit switch reached → <Key>
```

Possible causes:
- The drive was at the positive limit switch before the approach to reference point.
- The drive moved to the positive limit switch during the approach to reference point, because the assignment of the rotational direction of the motor and the assignment of the limit switch do not coincide.

The drive can be run down from this position in manual mode.

### 2.5.2 Move axes manually via front plate

After selecting this menu item, the user is informed:

```
"Wait until initialized (ESC=Cancel)"
"Axis : . . . . . . . (START=Continue)"
```

If an error occurs during the initialization of the axis, the following message is output via the display:

```
E900: A1 Parameter "VBUSBAL" different
Error, continue <key>
```
If there is no error, the select menu for the axes is displayed:

```
Axis selection
1 = Axis 1
```

Select the axis to be manually traversed by pressing the key with the relevant axis number (1..16). Depending on the configuration of the device, selection is limited to 1, 2, 4, or 16. The following will then be displayed (example for axis 1):

```
A1 = 0.000000  V=3200.00
1=320.00      2=3200.00       3=6400.00
```

The following assignment applies for the display:

**1st line:**
- A1 = 0.000000 → Current absolute position of axis, here A1
- V = 3200.00 → Currently selected traversing speed

**2nd line:**
- 1 = 320.00 → Traversing speed, taken from the creep speed parameter
- 2 = 3200.00 → Traversing speed, taken from the manual speed parameter
- 3 = 6400.00 → Traversing speed, taken from the reference speed parameter

The following key assignment is valid for this status:

<table>
<thead>
<tr>
<th>Key:</th>
<th>Reaction/effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow left</td>
<td>Traverse in negative direction up to range limit</td>
</tr>
<tr>
<td></td>
<td>(short actuation: single step; long actuation: continuous run)</td>
</tr>
<tr>
<td>Arrow right</td>
<td>Traverse in positive direction up to range limit</td>
</tr>
<tr>
<td></td>
<td>(short actuation: single step; long actuation: continuous run)</td>
</tr>
<tr>
<td>SHIFT arrow left</td>
<td>Traverse in negative direction beyond range limit</td>
</tr>
<tr>
<td></td>
<td>(short actuation: single step; long actuation: continuous run)</td>
</tr>
<tr>
<td>SHIFT arrow right</td>
<td>Traverse in positive direction beyond range limit</td>
</tr>
<tr>
<td></td>
<td>(short actuation: single step; long actuation: continuous run)</td>
</tr>
<tr>
<td>Arrow down</td>
<td>To next axis</td>
</tr>
<tr>
<td>Arrow up</td>
<td>To previous axis</td>
</tr>
<tr>
<td>1</td>
<td>Set current traverse speed on value after 1= ...</td>
</tr>
<tr>
<td>2</td>
<td>Set current traverse speed on value after 2= ...</td>
</tr>
<tr>
<td>3</td>
<td>Set current traverse speed on value after 3= ...</td>
</tr>
<tr>
<td>ESC</td>
<td>End (back to previous menu)</td>
</tr>
</tbody>
</table>
2.6 Programming

Overview of programming menu

After selecting this menu item, the user is offered the following choice:

<table>
<thead>
<tr>
<th>Edit program</th>
<th>1 = display program directory</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 = create new program</td>
<td></td>
</tr>
<tr>
<td>3 = alter existing program</td>
<td></td>
</tr>
<tr>
<td>4 = copy program</td>
<td></td>
</tr>
<tr>
<td>5 = rename program</td>
<td></td>
</tr>
<tr>
<td>6 = delete program</td>
<td></td>
</tr>
<tr>
<td>7 = display programme memory occupation</td>
<td></td>
</tr>
</tbody>
</table>

Note

The application memory of PA-CONTROL is limited owing to the size of the RAM module on the CPU4. The free application memory can be displayed under menu item 7.

After activation of a menu sub-option in the programming menu, PA-CONTROL checks if there is enough free application memory for this action. If not, the activated function will not be executed and is cancelled with the following message display:

not enough free memory

Press any key to return to the menu.

If the user still wishes to create new programs, the free application memory must be expanded.

As an option, you can also expand the RAM.

Possibilities:

- Delete programs which are not required any more.
- Delete unimportant comments in the programs or transfer programs to PA-CONTROL without comments.
2.6.1 Display program directory

This menu item allows the user to scroll through the current program directory (in alphabetical order). The following display appears:

```
- No -    -- name . type --
- Size -   S
  1   EXAMPLE . PNC
  213  0
```

The following assignments apply:

- No. : Consecutive numbering of the programs
- Name: Program name
- Type: Program type (PNC=NC program; PTX=Text program; PAB=Parallel run program)
- Size: Number of ASCII characters in the program
- S: Program status (0=write/read access possible; 1=write protected; 2=horizontal checksum error in ASCII area; 3=horizontal checksum error in the code area)

2.6.2 Create new program

When activating this menu item, the user is asked to enter a new program name (Refer to the description for Entry of program name).

The user must then select the program type:

```
Selection of program type
1 = PNC
2 = PTX
3 = PAB
4 = PNX
```

- PNC program: Executable NC program with positioning and I/O commands
- PTX program: Text program for labelling of inputs, outputs, etc. (for more information refer to the chapters "Program test and Diagnosis" on page 37 und "Run definitions" on page 43).
- PAB program: Executable NC program with positioning and I/O commands und PAB behaviour (see Chapter 3, "Commands of the PA family", page 65).
- PNX program: Executable NC program with positioning and I/O commands (for more information see Chapter 3, "Commands of the PA family", page 65).

After selection, the user finds himself in the 1st program line of the PA-CONTROL program editor. See Chapter 3, Section on Programming tips 71 for a description of the program editor.
Note

Once a program type has been selected, it cannot be subsequently altered in the PAC.
2.6.3 Altering an existing program

After activation of this menu item, a list of the available programs in alphabetical order is displayed. The user can scroll through the list and select the required program for programming by pressing the "ENTER key".

Please select ! [ 
] 
Program : 1 EXAMPLE.PNC

After selection, the user finds himself in the PA-CONTROL program editor. The first two lines of the selected program are displayed.

See Chapter 3, Section on Programming tips 71 for a description of the program editor.

2.6.4 Copy program

Using this menu item, the user can copy the content of an existing program to a new program.

After activation of this menu item, a list of the available programs is displayed in alphabetical order. The user can scroll through the list and select the required program:

Please select ! [H 
] 
Program : 21 MANUAL_RELEASE.PNC

After selection, the user is asked to enter a new program name. After correct entry, the program is copied and added to the directory as a further program.

Note

The program type cannot be changed during copying.

2.6.5 Rename program

The user can change the name of an existing program in this menu item.

After activation of this menu item, a list of the available programs is displayed in alphabetical order. The user can scroll through the list and select the required program:

Please select ! [S TOE 
] 
Program : 82 MALFUNCTION.PNC

After selection, the user is asked to enter a new program name. After entry, the program is added to the directory unaltered with a new name.
2.6.6  Delete program

The user can delete an existing program (from the program memory) in this menu item.

After activation of this menu item, a list of the available programs is displayed in alphabetical order. The user can scroll through the list and select the required program:

Please select ! [L    ]
Program :  44  L_LAMP.PAB

After selection, the user is again asked to confirm the selection and the deletion of the program:

Delete program  1=YES  /  Key=NO
L_LAMP.PAB

Pressing the "1" key will delete the program. Pressing any other key will cancel the selection and the program will not be deleted.

2.6.7  Program memory occupancy

The user can display the occupied and free bits of the application memory in this menu item.

ASCII free / occupied :  107949 / 507039
Code free / occupied :  107949 / 202608

Press any key to return to the menu.

The maximum application memory available is divided into the areas ASCII format and Code format. In the ASCII area, the programs are stored in the originally created form. For faster execution, the programs are converted from ASCII to an internal code format.

Note  The value for "free" varies depending on the development state and the equipment!
2.7 Diagnosis

Overview of the Program Test and Diagnosis menu

After selecting this menu item, the user is offered the following choice:

<table>
<thead>
<tr>
<th>Diagnosis</th>
<th>1 = Limit switch / Stand-by</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>→</td>
</tr>
</tbody>
</table>

2 = Inputs (I)  
3 = Outputs (O)  
4 = Flag (Marker)  
5 = Real number register (R)  
6 = Integer number register (N)  
7 = CANOpen  
8 = DA-Converter (CANOpen)

2.7.1 Limit Switch / Stand-by

This menu sub-option is used to check the limit switches of the individual axes as well as the stand-by of the motor power circuits. The displays vary in this menu item depending on the configuration of PA-CONTROL, because only the available motor axes are considered. After activation of this menu item, the following is displayed:

<table>
<thead>
<tr>
<th>Selection of axis</th>
<th>1 = Axis 1</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>→</td>
</tr>
</tbody>
</table>

2 = Axis 2  
..  
n = Axis n

Select the axis by pressing the key with the relevant axis number (1..16). Depending on the configuration of the device, selection is limited to 1, 2, 4,...,16.

The following key assignment applies for this status:

<table>
<thead>
<tr>
<th>Key:</th>
<th>Reaction / effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow down</td>
<td>To next axis</td>
</tr>
<tr>
<td>Arrow up</td>
<td>To previous axis</td>
</tr>
</tbody>
</table>

The following will then be displayed (example for axis 2):

<table>
<thead>
<tr>
<th>Axis 2</th>
<th>positiv limit switch:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td></td>
</tr>
<tr>
<td>Ready: 1</td>
<td>negative limit switch:</td>
</tr>
<tr>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

The following assignments apply:

0 : Power circuit not ready, limit switch (opener) actuated  
1 : Power circuit ready, limit switch (opener) not actuated
2.7.2 Inputs, Outputs, Flags

These menu sub-options are used to manually check or change the status of inputs, outputs and flags. After activation of these menu items, the following is displayed (example for inputs):

```
I  5 : -S01.04
I  1 → 0000000000000000
```

The following assignments apply:

1st Line: The number after the "I" input corresponds to the input on which the cursor field is positioned. "-S01.04" indicates the icon name for input 5 stored in the icon file. If the space bar is pressed, the assigned comment appears instead of the icon name, e.g. "Jog mode position 1"

2nd line: "I 1 →" means that, starting from input 1, the states of the next 16 inputs are displayed in this line. They are followed by the logic states of the inputs (0 or 1).

0: → Input cold, output or flag reset
1: → Input hot, output or flag set

**Moving and actions in the display:**

<table>
<thead>
<tr>
<th>Key:</th>
<th>Reaction / effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow right</td>
<td>Cursor field one digit to the right (to next element)</td>
</tr>
<tr>
<td>Arrow left</td>
<td>Cursor field one digit to the left (to previous element)</td>
</tr>
<tr>
<td>SHIFT + Arrow right</td>
<td>Cursor on last cursor field of the row (last element)</td>
</tr>
<tr>
<td>SHIFT + Arrow left</td>
<td>Cursor on first cursor field of the row (first element)</td>
</tr>
<tr>
<td>Arrow down</td>
<td>Display next group (e.g. input 17-32)</td>
</tr>
<tr>
<td>Arrow up</td>
<td>Display previous group (e.g. input 1-16)</td>
</tr>
<tr>
<td>SHIFT + Arrow down</td>
<td>Display last group (e.g. input 2027-2048)</td>
</tr>
<tr>
<td>SHIFT + Arrow up</td>
<td>Display first group (e.g. input 1-16)</td>
</tr>
<tr>
<td>ENTER</td>
<td>Select element directly, enter the required element in the input field and display with ENTER</td>
</tr>
<tr>
<td>1</td>
<td>Set the output or flag on which cursor field is positioned</td>
</tr>
<tr>
<td>0</td>
<td>Reset the output or flag on which cursor field is positioned</td>
</tr>
<tr>
<td>ESC</td>
<td>Quit sub-option</td>
</tr>
<tr>
<td>Space bar</td>
<td>Exchange icon name in the display for comment</td>
</tr>
</tbody>
</table>

If you are in the display for inputs I1...16, the following option for input can be reached by means of ENTER:

```
Enter new number
0 <= [ 25 ] <= 2048
```
After input and confirmation of the required element number with ENTER, it will be displayed in its 16 group. This guarantees a targeted and rapid diagnosis.

The following is an example of a display for inputs 17 - 32:

I 25 : +8-X01.18/4
I 17 -> 0000000000000000

The information for diagnosis of outputs or flags is displayed in the same way.

The following is an example of a display for outputs 33 - 48:

O 34 : +9-a13
O 33 -> 0000000000000000

The following is an example of a display for flags 497 - 512:

M 504 :
M 497 -> 0000000000000000

Note

The statuses of the outputs and flags are retained when the menu is exited!
The number of actual inputs and outputs can vary depending on the equipment of the controller! See also Chapter 1.4.4 of the Technical Short Documentation and the section on the CANopen bus in Chapter 6, Options, page 347.
2.7.3 **Real number register (R), Integer register (N)**

These menu sub-options are used to manually check or alter the contents of the real number and integer register. After activation of these sub-options, the following is displayed (example for real number register):

<table>
<thead>
<tr>
<th>R  1 :  Axis 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>R  1 = 2.50000</td>
</tr>
</tbody>
</table>

The following assignments apply:

1st Line: The number after "R" (real number register) is the number of the displayed register. "Axis 1" indicates the icon name for real number register 5 stored in the icon file. If the space bar is pressed, the assigned comment appears instead of the icon name, e.g. "Intermediate position in press DO"

2nd Line: \( R \ 1 = 2.50000 \) is the content of the displayed register.

**Moving and actions in the display:**

<table>
<thead>
<tr>
<th>Key:</th>
<th>Reaction / effect:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow down</td>
<td>Display next register</td>
</tr>
<tr>
<td>Arrow up</td>
<td>Display previous register</td>
</tr>
<tr>
<td>SHIFT + Arrow down</td>
<td>Display register number +10</td>
</tr>
<tr>
<td>SHIFT + Arrow up</td>
<td>Display register number -10</td>
</tr>
<tr>
<td>SHIFT + Arrow left</td>
<td>Last register</td>
</tr>
<tr>
<td>SHIFT + Arrow right</td>
<td>Display first register</td>
</tr>
<tr>
<td>ESC</td>
<td>Quit sub-option</td>
</tr>
<tr>
<td>Numeric key (1..9+)</td>
<td>The character is applied at the first position of the displayed value (automatic activation of the input field)</td>
</tr>
<tr>
<td>Arrow right</td>
<td>Activation of the input field</td>
</tr>
<tr>
<td>ENTER</td>
<td>Conclude the input after input of a value</td>
</tr>
<tr>
<td>ENTER</td>
<td>Select element directly, enter the required element in the input field and display with ENTER</td>
</tr>
<tr>
<td>ESC</td>
<td>Cancel the input, the old value is retained</td>
</tr>
<tr>
<td>Space bar</td>
<td>Exchange icon name in the display for comment</td>
</tr>
</tbody>
</table>

See the section on "General Operation" on page 25 in this chapter for the key assignment in the input field.

The input value is only transferred to the register if it is within the defined value limits (+/-8,000,000).
Example for the integer register:

N  465 : TEACH_hfkt. ST5
N  465 = 71

The following assignments apply:
1st Line: The figure after "N" (= integer register) is the number of the displayed register. "TEACH-hfkt. ST5" indicates the icon name for integer register 465 stored in the icon file. If the space bar is pressed, the assigned comment appears instead of the icon name, e.g. "n+4 characters of string"

2nd Line: N  4656 = 71 is the content of the displayed register.

2.7.4 CANopen

After selecting this menu item, the following messages are shown in the display:

CAN Diagnosis / Manual
1 = display CAN-Hardware configuration

2 = Reset all CAN-Axes
3 = Create list of current devices

Menu item "Display CANopen devices"

The user can view the current configuration of the axes as devices on the CANopen bus in this menu item.

Example:
The following devices were on the CANopen bus after booting PA-CONTROL:
- An LV-servoTEC with the address 1
- A PA-CONTROL-MP with the address 2 and
- An LV-servoTEC with the address 4

Note
Only the devices which directly follow each other are transferred during booting. The LV-servoTEC with the address 4 is therefore not transferred to the configuration!

The following is displayed:

List of CANopen devices
Address 1 : CANopen servoTEC

Address 2 : CANOpen EP
Address 3 : ----- 
Address 4 : -----
Menu item "Reset all CAN axes"
The user has the possibility of activating a reset/cold start for the connected LV-servoTEC axes in this menu item. This resets possible errors.

Menu item "Create list of current devices"
A list of the current devices on the CANopen bus is created in this menu item. Here PA-CONTROL attempts to read the "device type" of addresses 1 to 20. The result is displayed as follows:

Example:
The following devices were on the CANopen bus after booting PA-CONTROL:
- An LV-servoTEC with the address 1
- A PA-CONTROL-EP with the address 2 and
- An LV-servoTEC with the address 4
- An unknown device at address 6

After selecting the menu item, the following message appears for approx. 2 seconds:

```
actual CANopen-Nodes (devices 1-24) 
---------------------- (ESC=cancel)
```

The search for the devices is activated and the following appears on the display:

```
actual CANopen-Nodes (devices 1-24)
12.?............A.. (ESC=cancel)
```

- no device
- 1 LV-servoTEC
- 2 PA-CONTROL-EP (as Slave)
- A IO modules
- ? unknown device

2.7.5 D-A Converter (CANopen)
PA-Control can operate up to 4 analog outputs (DA1 to DA4) via the CANopen bus (see Chapter 6 "Options", CANopen interface, page Fehler! Textmarke nicht definiert.. The user can specify a new D-A value for the D-A.converter in this menu item.

Example:
The value 2345 is to be specified

```
DA 1 : 2345
```

The D-A value to be output is "2345". The menu is exited by means of the ESC key.
2.8 Run definitions

Overview of the run definition menu

Definitions for the automatic run of PA-CONTROL are set in this menu. After selecting this menu item, the user is offered the following choice:

```
PA-CONTROL  4.xx
5 = Run definitions
```

```
START program
Program : START.PNC
```

Program at stop
Program START after STOP
Program at malfunction
Title
Program in BASIC-Mode
Delete programm assignment ? 1=yes
Adopt modifications ? 1=yes
Assignment not avaible ! <key>

Except for the start program, which always has to be defined, the other assignments can be optional.

Note

The start program, program after stop, program start after stop, and program at malfunction must be different programs. In the case of double use of a program, the error message "Assignment not possible" is displayed at storage.

2.8.1 Start program

The PA-CONTROL starts in automatic mode with this program. In principle, this program is the main program.

2.8.2 Program at stop

Special actions (e.g. closing valves) can be carried out with this assignment when "STOP" is detected in automatic run. This program is run after the axes have been stopped.

CAUTION

In the program after stop, the set of commands has the following restriction:

1. This program may not invoke any further programs as subprograms or programs to be processed in parallel.
2. Positioning commands are not permissible in this program.
3. The commands "Wait for logical state of..." inputs, outputs and flags may not be used.
4. Any time monitoring which is still active is reset by the operating system without a message and is not reactivated at "Start".
The following commands may not be used in the program "after STOP":

- SUB, CASE.SUB
- RUN, CASE.RUN
- G22
- Ai:=n, Ai:=Rn
- G01
- G212, G222

PA-CONTROL checks in automatic mode and generates the errors E506, E507 or E508 if an error is found.

### 2.8.3 Program Start after Stop

If "STOP" has been activated during the automatic run of PA-CONTROL and the program is to be restarted with "START" (automatic mode has not been exited), the program "Start after Stop" will be executed before continuation of the interrupted program. This option can be used to reset actions which were triggered by a stop.

This program may not invoke any further programs as subprograms or programs to be processed in parallel. Positioning commands are not permissible in this program.

The PA-CONTROL checks in automatic mode and generates the errors E506, E507 or E508 if an error is found.

### 2.8.4 Program at malfunction

With this assignment, special actions (e.g. closing valves) can be implemented when a malfunction is detected during the program run (e.g. error messages such as value too large, power circuit not ready...).

This program may not invoke any further programs as subprograms or programs to be processed in parallel. Positioning commands are not permissible in this program.
2.8.5 Title

The user can replace the title in the main menu "PA-CONTROL  Vxx.xx" by a customised user-defined title. To do this, the user must create a text program (type: PTX) and assign this text program under the menu item "Title". The first program line of this text program is then displayed as a title in the main menu.

2.8.6 Activation of run definition

A list in alphabetical order of the possible programs of the type "PNC/PAB" is displayed after activation of these sub-options. The user can scroll through the list and select the required program with "Enter".

```
Please select !   [   ]
Program :   1   SOLDER_MAIN.PNC
```

After selection has been made, a second confirmation is required. If confirmation with 1=yes is not made, the "old" assignment remains in effect.

```
Adopt modifications ?  1=yes
```

Note
This assignment has to be made with the utmost of care, because the complete machine run is influenced.

2.8.7 Deletion of assignments

If the user wishes to delete an assignment (Start program,..., Text program), the assignment to be deleted must be displayed in the second line of the display. By pressing the "DEL" key, the user informs PA-CONTROL that this assignment is to be deleted. The user must confirm the action before the assignment is deleted.

```
Start program
Delete program assignment ?  1 = Yes
```
2.9 Parameters

Overview of the parameter menu

After selecting this menu item, the user is offered the following choice:

<table>
<thead>
<tr>
<th>Parameters</th>
<th>1 = system parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 = axis parameters</td>
<td></td>
</tr>
<tr>
<td>3 = edit Asi-BUS</td>
<td></td>
</tr>
<tr>
<td>4 = edit servoTEC-Parameters</td>
<td></td>
</tr>
</tbody>
</table>

There are two types of parameter in PA-CONTROL, the system parameters and the axis parameters.

The meaning of the individual parameters and their useful application is explained in Chapter 5, page 319.

After entering the "System parameters" sub-option, the list of parameters is displayed as follows:

<table>
<thead>
<tr>
<th>Parameters (definitions)</th>
</tr>
</thead>
</table>

The following is displayed for the first system parameter, language selection:

<table>
<thead>
<tr>
<th>Language (1=Ger, 2=Eng, 3=Fre)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 &lt;= 1 &lt;= 3</td>
</tr>
</tbody>
</table>

In this example, the set language is German.

The user can scroll through the parameter list and edit the required parameter by activating the input field by means of the corresponding numeric key or the cursor key "Arrow left". The input (altered) value must be within the min. and max. value limits, otherwise an error message is displayed after conclusion of the input. The input must be repeated or cancelled by "ESC".

Before entering the axis parameters, the user must first select the desired axis.

The submenu "Edit ASi BUS" can only be called when one or more ASi cards are connected. See Chapter 6, "Options", page Fehler! Textmarke nicht definiert, for further information on the configuration of the ASi.

The submenu "Edit servoTEC parameters" can only be invoked if one or more LV-servoTECs are connected to PA-Control via CANopen bus.
2.9.1 **ASi parameters**

After selecting this menu item, the following is displayed:

```
Edit Asi-BUS
1 = Apply actual configuration

2 = Display state
3 = Activate automatic programming
4 = Change operation mode
5 = display configuration
6 = program slave number
7 = transfer projection to Asi-Master
```

### 2.9.1.1 Apply actual configuration

```
sure, apply actual configuration?
1=Yes
```

Key = no

### 2.9.1.2 Display status

**Example:**

```
OFL APF NORM PROJ Aavai Aakti LDs.0 OK
0 0 1 1 0 1 0 1
```

- **OFL**: Offline mode
- **APF**: ASi power supply is down
- **NORM**: Normal mode
- **PROJ**: Configuration mode
- **Aavai**: Auto programming of the slave address is possible
- **Aakti**: Auto programming of the slave address is active
- **LDs.0**: Slave with address 0 exists
- **OK**: There are no configuration errors

### 2.9.1.3 Activate automatic programming

```
ASi.BUS automatic slave programming
Activated : yes (ENTER/ESC)
```

Activated : no (ENTER/ESC)

You can toggle between yes and no by means of the Enter and the ESC key.
2.9.1.4 Change mode

**Protected operation mode (automatic)**
activated : no (ENTER/ESC)

2.9.1.5 Display configuration

The configuration of the individual slaves on the ASi bus is displayed in this menu item. You can scroll by means of the arrow keys, ESC means end.

**Example:**
Slave 0

| ASiActual : no |
| Activated : yes (ENTER/ESC) |

**Example:**
Slave 1

(detected and configured, 4 inputs and 4 outputs)

| ASiActual : yes 4E/4A I : 1100 |
| 1-1Pro : yes 4E/4° O : 0000 |

2.9.1.6 Program slave

Only for changing a slave address

1. **Asi slave actual address [01]**

2. **Asi slave actual address [01]
  ASi slave new address [12]**

3. **Asi slave actual address [01]
  ASi slave new address [12] ENTER**

4. **OK, continue with key**
2.9.1.7 Transfer configuration to ASi Master

- Transfer projection to Asi-Master
  - Wait

- Transfer projection to Asi-Master
  - OK -> ESC
2.9.2 Edit servoTEC parameters

After selecting this menu item, the following is displayed:

<table>
<thead>
<tr>
<th>LV-servoTEC Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = Show (all)</td>
</tr>
</tbody>
</table>

- 2 = Edit
- 3 = Send + Save + Coldstart
- 4 = Load + initialize axis

2.9.2.1 Display all

<table>
<thead>
<tr>
<th>Display (all)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = Axis 1</td>
</tr>
</tbody>
</table>

2 = Axis 2

After selection of the desired axis, the user can display the current parameter values of the LV-servoTEC.

Example:

<table>
<thead>
<tr>
<th>A1 LV-servoTEC Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>O (P)-No. 3500 (hex) MAXSDO = 0</td>
</tr>
</tbody>
</table>

- O (P)-No. 3501 (hex) ACC = 10
- O (P)-No. 3502 (hex) ACCR = 200
- ...
- O (P)-No. 3672 (hex) DRVCFG = 0

Functions of the keys:

<table>
<thead>
<tr>
<th>Key</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Arrow down</td>
<td>Next parameter (increment 1)</td>
</tr>
<tr>
<td>Arrow up</td>
<td>Previous parameter (increment -1)</td>
</tr>
<tr>
<td>SHIFT + Arrow down</td>
<td>Next parameter (increment +16)</td>
</tr>
<tr>
<td>SHIFT + Arrow up</td>
<td>Previous parameter (increment –16)</td>
</tr>
<tr>
<td>SHIFT + Arrow left</td>
<td>First parameter</td>
</tr>
<tr>
<td>SHIFT + Arrow right</td>
<td>Last parameter</td>
</tr>
<tr>
<td>ESC</td>
<td>Exit menu</td>
</tr>
</tbody>
</table>
2.9.2.2  Editing

Some parameters of the LV-servoTEC can be edited by means of the program WINPAC and also via PA-CONTROL. Please refer to the LV-servoTEC manual or the "ASCII list" supplied as a pdf file on CD-ROM for the meaning of the parameters.

<table>
<thead>
<tr>
<th>Edit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = Axis 1</td>
</tr>
</tbody>
</table>

2 = Axis 2

<table>
<thead>
<tr>
<th>A1 : Rotary direction (1=positive/0=negative)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 &lt;= 1 &lt;= 1</td>
</tr>
</tbody>
</table>

A1 : In Position [counts]
A1 : Disable Ramp / Emergency Ramp [ms]
A1 : Max. tow error [counts]
A1 : Position controller -> KV
A1 : Position controller -> Tn
A1 : Position controller -> Kp
A1 : Speed controller -> KP
A1 : Speed controller -> Tn
A1 : Speed controller -> PID-T2
A1 : Speed controller -> PI-Plus
A1 : Speed controller -> T-Tacho
A1 : Standstill barrier [rpm]

On exiting the editing window, the parameters are compared with those in the LV-servoTEC. If the LV-servoTEC cannot be reached via CANopen bus, an error message is displayed.

| A1 Verify LV-servoTEC parameters |

If a parameter has been changed, the PA-CONTROL finds a difference and displays the following menu:

<table>
<thead>
<tr>
<th>A1 Parameter &quot;PEINPOS&quot; different</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 = Send + save + Coldstart</td>
</tr>
</tbody>
</table>

2 = Ignore
3 = Get + Initialize axis

The user must now decide:

- (1 = Send + store + cold start) The parameters are transferred from PA-CONTROL to the LV-servoTEC, stored there, and a "cold start" is subsequently carried out by means of the amplifier (All parameters are transferred).
- (2 = Ignore) The difference is ignored.
- (3 = Get + Initialize axis) The change is rejected and the old parameters are retrieved from the LV-servoTEC.
2.9.2.3  Send + store + cold start

Send + Save + Coldstart
1 = Axis 1
2 = Axis 2

PA-Control transfers the parameters of the selected axis to the LV-servoTEC, stores the parameters in the LV-servoTEC and then carries out a cold start with the LV-servoTEC.

The following successively appears on the display:

Send Parameters of A1 to LV-servoTEC
TRANSFER : ACC (Cancel with ESC)

Send Parameters of A1 to LV-servoTEC
SAVE : (Cancel with ESC)

Send Parameters of A1 to LV-servoTEC
COLDSTART : (Cancel with ESC)

Transfer finished

2.9.2.4  Get + Initialize axis

Load + initialize axis
1 = Axis 1
2 = Axis 2

PA-CONTROL retrieves the parameters of the selected axis from the LV-servoTEC. The following successively appears on the display:

Load parameters of A1 from LV-servoTEC
TRANSFER : ACC (Cancel with ESC)

Load parameters of A1 from LV-servoTEC
Transfer finished
2.10 Basic settings

Overview of the basic settings menu

After selecting this menu item, the user is offered the following choice:

<table>
<thead>
<tr>
<th>Basic settings</th>
<th>1 = load default system parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 = load default axis parameters</td>
</tr>
<tr>
<td></td>
<td>3 = delete all programs in memory</td>
</tr>
<tr>
<td></td>
<td>4 = reinitialize PA-CONTROL</td>
</tr>
<tr>
<td></td>
<td>5 = CANopen baud rate edit</td>
</tr>
</tbody>
</table>

The user can set PA-CONTROL in a defined initial state by means of these functions.

**Note**

On selection of these functions, all the current machine parameters will be overwritten and programs will be deleted.

These functions should only be selected by qualified and authorized personnel.

Always carry out a data backup beforehand using WINPAC or hardcopy!

Please refer to Chapter 5, "Parameters", page 319 for the values for setting the parameters.

2.10.1 Load system parameter default values

The system parameters (language, PA-CONTROL address, etc.) are set to the default values (external stop input no.= 0; external start input no.= 0).

2.10.2 Load axis parameter default values

The parameters can either be set to default values for a selected axis or for all axes.

2.10.3 Delete program memory

All the programs in the program memory are deleted.
2.10.4 Reinitialize PA-CONTROL

PA-CONTROL is reinitialized in the following order:

- Determine the hardware configuration of PA-CONTROL
  - Number of axes, axis types
  - Number and type of IEF modules
  - Number of COM ports
  - Number of ASis
  - Number of other hardware components
- Load default parameters (see Chapter 5, "Parameters", page 319) as per the determined hardware configuration
- Delete complete contents of the program memory

Note

- External stop input no. = 0
- External start input no. = 0
- etc.

During this action, all flags are reset, all registers are set to 0, all programs are deleted and the parameters are loaded with the initial values.

Reinitialization of PA-CONTROL is essential in the following cases:

- For the initial start-up of the CPU4 for PA-CONTROL (carried out at IEF Werner)
- After changing the battery on the CPU4
- Change of the hardware configuration (other cards and modules)

Example - Load System parameter default values:

The following appears on the display:

```
Load default system parameters
1 = yes / Key = no
```

The user can start the boot of the system parameters by pressing the numeric key "1" or cancel the action by pressing any other key.

The user can also start this function via WINPAC (see 2.10.5).
2.10.5 Reinitialization for devices without a display and keyboard

Reinitialisation of PA-CONTROL is also essential in the following cases:
- For the initial start-up of the CPU4 for PA-CONTROL (carried out at IEF Werner)
- After changing the battery on the CPU4
- Change of the hardware configuration (other cards and modules)

Reinitialization can only be carried out by means of "WINPAC" and a PC via the serial port for devices with a 7 segment display.

Proceed as follows:

Start WINPAC
Select
→ "PA-CONTROL" and
→ "Reinitialize PA-CONTROL" in the "Settings" menu.

**Note**
During this action, all flags are reset, all registers are set to 0, all programs are deleted and the parameters are loaded with the initial values.
2.11 System Diagnosis

A further menu list is displayed after activation of this sub-option:

```
System Hardware Diagnosis
1 = display hardware configuration →
```

2 = clock
3 = keyboard test
4 = STOP key test
5 = START key test
6 = Key switch test
7 = test COM 1 - 4

2.11.1 Display hardware configuration

The user can call up the configuration of his PA-CONTROL in this sub-option.

```
PA-Control hardware configuration
Serial number : 123456
```

Example:

```
CPU-Boot version
PA-CONTROL version : 4.50
CPU type : CPU 4
CPU version : 2.0
ROM size : 1024k*8
RAM size : 1024k*8
Number axes/AbsPoSys. : 4/0
Axis type (A1) : PLS7 V4.02.1
Profibus DP : 0
Number of Asi-Masters : 1
Number of COM : 2
Number of AD converter : 0
Interbus-S : 0
```

The current values are determined by the controller after reinitialization.

Should the determined values differ from the previous ones, the following message appears in the display after a power-on reset:

```
Exxx : Plain-text message
System error reset ? 1 = yes
```

This message is usually displayed when a component (card or module) is inserted or removed from the system.
A reset is not possible for some system errors. In this case, the controller must be informed of the altered configuration via "2 = Reinitialize PA-CONTROL".

<table>
<thead>
<tr>
<th>System error</th>
<th>1 : System error reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 = Reinitialize PA-CONTROL</td>
<td></td>
</tr>
</tbody>
</table>

Note
On selection of these functions, all the current machine parameters will be overwritten and programs will be deleted.
These functions should only be selected by qualified and authorized personnel.
Always carry out a data backup beforehand using WINPAC or hardcopy!

2.11.2 Clock
PA-CONTROL contains a battery-buffered clock module. The information for this module can be displayed and altered in this menu item.

<table>
<thead>
<tr>
<th>Date</th>
<th>&gt;DD&lt;MM YYYY WD</th>
<th>Time</th>
<th>HH MM SS</th>
</tr>
</thead>
<tbody>
<tr>
<td>02.05.2000 3-Wednesday</td>
<td>14:36:38</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The selected element is framed by the characters > <. The individual elements (DD=day etc.) are selected by means of the keys → and ←. Information about the selected element is obtained by means of the ENTER key. It can then be modified in the input field.

2.11.3 Keyboard test
After selection of this sub-option, the following is displayed:

keyboard test, ESC = End!

On actuation, the keys are shown on the display or executed according to their described function (See Chapter 2 "User interface" Heading Keyboard). Exceptions: Arrow keys, INS, DEL, ENTER, START, STOP, ALT, CTRL.
2.11.4 Stop key test

After selection of this sub-option, the following is displayed:

```
Stop key test, ESC = End!
activated : no_
```

On pressing the stop key, the display switches from "no" to "yes". This function refers to the stop key on the front plate. The possible external stop input can be checked via diagnosis inputs.

2.11.5 System diagnosis - start key test

After selection of this sub-option, the following is displayed:

```
Start key test, ESC = End!
Start key activated: no_
```

On pressing the start key, the display is toggled from "no" to "yes". This function refers to the start key on the front plate. The possible external start input can be checked via diagnosis inputs.

2.11.6 Key switch test

After selection of this sub-option, the following is displayed:

```
Key switch test, ESC = End!
Key switch is on : Program
```

On turning the key switch, the display is toggled from "program" to "automatic".

2.11.7 Test of the COM Port

The hardware connection (RS232 cable) as well as the setting of the serial port can be tested by means of this function.

After selection of this sub-option, the following is displayed:

```
Test COM1 - 4
1 = COM 1
```

The input window is switched to by means of the arrow keys or by entering a number between 1 and 4 of the port to be checked.

All characters received are output to the display. After pressing a key, they are immediately output via the port.

```
Test COM 1 (ESC=End) 22111122345566778 899aabbccddeeeAABBCCDDEE
```
2.12 Communication via modem

Communication between WINPAC and PA-CONTROL can be effected via modem. In this case, the operator can execute the following via the modem:

- Load and transfer programs
- Load and transfer parameters
- Load and transfer register values
- Diagnosis
- Load and transfer register values
- Start / Stop of the PA-CONTROL
- Direct modification of outputs, flags and registers in PA-CONTROL
- Manual traversing of the axes of PA-CONTROL

The modem can be connected to PA-CONTROL via the diagnosis port. A remote user can access PA-CONTROL by this means. Connection can be activated by means of the menu sub-option Activate connection (keyboard) or using the input defined in the system parameters.

If a teleservice input has been defined (system parameters), the modem can be activated using the defined input. Depending on the setting, the service PC is called or the modem switches to reception. The connection can be started either in the main menu or during automatic run.

A defined teleservice output indicates the connection status:
- Fast flashing: PA-CONTROL is calling WINPAC (approx. 200ms)
- Slow flashing: PA-CONTROL is waiting for call (approx. 1 sec)
- Continuous light: Connection between PA-CONTROL and WINPAC has been established

2.12.1 Overview of the modem menu

After selecting menu item 9, the user is offered the following choice:

<table>
<thead>
<tr>
<th>Modem</th>
<th>1 = activate connection</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>2 = display connection status</td>
</tr>
<tr>
<td></td>
<td>3 = terminate connection</td>
</tr>
<tr>
<td></td>
<td>4 = edit modem settings</td>
</tr>
<tr>
<td></td>
<td>5 = Select type of modem</td>
</tr>
</tbody>
</table>
2.12.2 Activate connection

Really activate modem connection 1
   = yes / key = no

Once the modem has been connected and switched on, press key 1. The connection is set up - the steps which are carried out are shown in the status display.

Status communication via modem
7: Init OK, waiting for call

Any problems are indicated by the corresponding status number and a plain-text message, e.g.:

Status Communication via modem
17 : Error: modem not ready

Normally, the next message to be displayed is:

Status Communication via modem
8 : Call arrived, waiting until connect

After the connection has been successfully set up, the following is displayed:

Status Communication via modem
9: Line established, OK

The user must now return to the main menu by means of the ESC key, so that the modem communication can ensue without restriction.
2.12.3 Display connection status

The current status is indicated by the corresponding status number and a plain-text message.

```
Status Communication via modem
9: Line established, OK
```

2.12.4 Terminate connection

If a connection is active, the following is displayed:

```
Terminate modem communication ?
1 = yes / Key = no
```

If there is no connection at this time, the following is immediately displayed:

```
Status Communication via modem
0: not activated
```
2.12.5 Edit Modem Settings

In this menu, the user can select the modem commands for the initialization of his modem using the ↑ ↓ keys.

The settings can be edited by means of the → key. You will need the manufacturer's specifications for the modem.

If modems supported by PA-CONTROL are used (refer to "Select modem type"), the initialization string is generated by PA-CONTROL.

- **Modem: Type**
  
  : extension, not restricted

The settings can be edited by means of the → key. You will need the manufacturer's specifications for the modem.

- **Modem: Initialise**
  
  : AT&FX3S0=1

The initialisation string is transmitted to the modem when it is called. Refer to the operating instructions of the modem for details of the initialization commands. The initialization string can be extended depending on the type of telephone installation.

- **Modem: PA-CONTROL calls service PC**
  
  (1=yes) : 0

If PA-CONTROL is to call a service PC, select "1". PA-CONTROL initializes the modem and automatically dials the specified number of the service PC. If PA-CONTROL is to a call from the service PC, "0" is set, the modem is initialized and waits for an incoming call.

- **Modem: Service PC number**
  
  : 07723-925180

If the PA-CONTROL setting is "PA-CONTROL will call service PC" = 1, the number specified here is called.

The command string is sent to the modem before the telephone number (refer to the manual of the modem manufacturer).

- **Modem: Dial prefix**
  
  : ATDT0W

The command string is sent to the modem after the telephone number (refer to the manual of the modem manufacturer).

- **Modem: Dial suffix**
  
  :
2.12.6 Select modem type

The basic settings for the recommended modems are stored for modem communication.

Really activate modem connection ?
1 = yes / key = no

A list of modems recommended by us is displayed by pressing key 1.

List Type
1 = user defined
2 = line, not restricted
3 = extension, not restricted
4 = internal, extension to extension

Only analog modems are supported.
2.12.7 Examples of communication setup sequences

Communication Example: Service calls customer (WINPAC → PA-CONTROL)

<table>
<thead>
<tr>
<th>Service person with PC and WINPAC</th>
<th>Customer with PA-CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set up connection from PC to the telephone network via modem and switch on the device.</td>
<td>Connect PA-CONTROL to the telephone network via modem</td>
</tr>
<tr>
<td>Select modem type</td>
<td>Execute modem settings</td>
</tr>
<tr>
<td>Edit connection data (name, telephone number)</td>
<td>&quot;1 = activate connection&quot; or make a remote connection using teleservice input.</td>
</tr>
<tr>
<td>Wait until customer is ready with PA-CONTROL</td>
<td></td>
</tr>
<tr>
<td>&quot;Select connection&quot; using WINPAC (red telephone) and wait until message &quot;Connected&quot; (green LED) is displayed</td>
<td>The following message appears on the display of PA-CONTROL: Status communication via modem</td>
</tr>
<tr>
<td></td>
<td>9: Connected, OK</td>
</tr>
<tr>
<td></td>
<td>The assigned output is permanently active with teleservice</td>
</tr>
<tr>
<td></td>
<td>If necessary, exit this menu item to allow all access options via WINPAC</td>
</tr>
<tr>
<td>PA-CONTROL can now be accessed via WINPAC.</td>
<td></td>
</tr>
</tbody>
</table>

Disconnection can be executed via WINPAC or PA-CONTROL (3=terminate connection or teleservice input).

Communication Example: Customer calls service (PA-CONTROL → WINPAC)

<table>
<thead>
<tr>
<th>Customer with PA-CONTROL</th>
<th>Service person with PC and WINPAC</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect PA-CONTROL to the telephone network via modem</td>
<td>Set up connection from PC to the telephone network via modem and switch on the device.</td>
</tr>
<tr>
<td>Execute modem settings (normally only needs to be carried out once)</td>
<td>Select modem type</td>
</tr>
<tr>
<td>Wait until programmer is ready with WINPAC</td>
<td>Activate WINPAC &quot;Wait for call&quot;</td>
</tr>
<tr>
<td>&quot;1 = activate connection&quot; or make a remote connection with teleservice input.</td>
<td>Wait for a connection</td>
</tr>
<tr>
<td>The following message appears on the display of PA-CONTROL: Status communication via modem</td>
<td></td>
</tr>
<tr>
<td>9: Connected, OK</td>
<td>The assigned output is permanently active with teleservice</td>
</tr>
<tr>
<td></td>
<td>If necessary, exit this menu item to allow all access options via WINPAC</td>
</tr>
<tr>
<td></td>
<td>PA-CONTROL can now be accessed via WINPAC.</td>
</tr>
</tbody>
</table>

Disconnection can be executed via WINPAC or PA-CONTROL (3=terminate connection or teleservice input).
3 Commands of the PA-CONTROL Family

3.1 Overview of the command set

Some commands of PA-CONTROL can only be used with specific PA-CONTROL types.

Command 1 \( \rightarrow \) 1 = PA-CONTROL MP
Command 2 \( \rightarrow \) 2 = PA-CONTROL STEUER with PLS7 / PLS9
Command 3 \( \rightarrow \) 3 = PA-CONTROL COMPACT with PLS7
Command 4 \( \rightarrow \) 4 = PA-CONTROL servoTEC

The superscripted number in the descriptions of the commands identifies the PA-CONTROL type in which the command can only be used.

3.1.1 Program execution (automatic)

Cancel ........................................................................................ BREAK .................................... 117
Stop ........................................................................................... STOP ...................................... 122
Continue ................................................................................... START .................................... 123

3.1.2 Axis

Approach to reference point ..................................................... G25.A1.................................... 143
Traversing .................................................................................. A1:=20 # A1:=R3 ................. 132
Absolute traversing ................................................................ G90.A0 # G90.A1 .................. 146
Relative traversing .................................................................. G91.A0 # G91.A1 .................. 149
Acceleration .............................................................................. G100.A1.15 ............................ 150
Traversing speed ...................................................................... FA1:=200 # FA2:=R4 .............. 133
Set position to 0 ........................................................................ G26.A1 ................................... 145
Set position to size (set to new value) ...................................... G29.A1.1.345 # G29.A2.R3 ...... 146
Traverse as long as condition is fulfilled ................................... G123.A1 I3.1 .......................... 151
Activate start positionning mode.............................................. G210.A3 ..................................178
Activate normal positioning mode.......................................... G213.A3 ..................................183
Activate measuring mode....................................................... G140 .........................................157
Deactivate measuring mode.................................................... G141 .........................................158
Switch off limit switch monitoring .......................................... G142 .........................................159
Switch on limit switch monitoring .......................................... G143 .........................................160
Determine path speed ............................................................. Rn:=FB .....................................134
Load axis position ................................................................... R1:=A1 # N1:=A1 ................... 133
Get axis parameter (load) ....................................................... R1:=RA1.1.............................. 248
Change motor current 1,4 ..................................................... G101.A1.50 ............................ 151

Linear interpolation 2,3 ........................................................... G01 A1:=40 A2:=60 ............... 163
Activate output during an interpolation ................................ G160.20.O123.1 ..........................167

3.1.3 Feedback system (rotary encoder / encoder)

Get following error from LV-servoTEC ............................... Ni:=PEAn # Ri:=PEAn .............137
Get encoder position (load) 1 ................................................ R1:=ENC1 # N1:=ENC1 .............138
3.1.4 Input

Wait for status of input .............................................................. I1.1 ............................................ 91
Jump in program when status of input ..................................... G21 I1.1 ...................................103
Call subroutine when status of input ........................................ G22 I1.1 .................................. 108
Logic AND operation with input ................................................ LD / AND ................................ 270
Logic OR operation with input .................................................. LD / OR................................... 270

Image of inputs to a register ..................................................... G603.R3.2.8 ........................... 244

3.1.5 Output

Set / Reset output ..................................................................... O1:=1 # O2:=0.......................... 95
Wait for output status ................................................................ O1.1 ......................................... 92
Jump in program when status of output ................................... G21 O1.1 ................................ 103
Call subroutine when status of output ...................................... G22 O1.1 ................................ 108
Logic AND operation with output .............................................. LD / AND / OUT...................... 270
Logic OR operation with output ................................................ LD / OR / OUT ........................ 270
Set, reset, negate output .......................................................... SET / RES / NOT.................... 271

Image of register to outputs (binary) ........................................ G600.R3.5.8 ........................... 240
Image of register to outputs (BCD) ........................................... G601.N2.2.4 ........................... 241

3.1.6 Dwell time, monitoring time, real-time clock

Delay program execution until time has elapsed ..................... T10, TN1................................... 97
Start monitoring time with jump in program .............................. G421.1.200 ERROR_1........... 191
Start monitoring time with call of subroutine ............................. G422.1.500 ERROR_2........... 192
Start monitoring time with repeat function ................................ G423.1.300 CRASH ............... 193
Delete (reset) monitoring time .................................................. G401.1.................................... 194
Read real time clock .................................................................. S0:=TIME.HOUR.MIN .............195
Set date ..................................................................................... Date:=N1.N2.N3.N4 .................197
Set time...................................................................................... TIME:=N1.N2.N3 ......................198

3.1.7 Flag

Direct or indirect addressing ..................................................... M1.1 # M!10.1  95
Set / Reset flag ......................................................................... M1:=1 # M2:=0  96
Wait for flag status ................................................................. M1.1  93
Jump in program when status of flag ...................................... G21 M1.1................................ 103
Call subroutine when status of flag ........................................... G22 M1.1 ................................ 108
Logic AND operation with flag .................................................. LD / AND / OUT...................... 270
Logic OR operation with flag .................................................... LD / OR / OUT........................ 270
Set, reset, negate flag .............................................................. SET / RES / NOT.................... 271

Image of register to flag ............................................................ G602.R3.4.8 242
Image of flags to a register ....................................................... G604.N4.2.8  245

Assign comparison result ......................................................... M36:=R1<56  262
3.1.8 \textbf{N/R register (variables)}

- Direct and indirect addressing ................................................................. N1, N2 .......................... 246
- Load register with value (constant) .......................................................... N1:=23, R1:=33,5 ................. 247
- Wait for status of an N register .............................................................. Ni.n, Ni.Nn, N!i.n, N!i.N!n .............
- Addition .................................................................................................. N1:=N1+N2 ..................... 249
- Subtraction ............................................................................................. N1:=N2-N3 .......................... 251
- Multiplication ......................................................................................... N1:=N2*N3 .......................... 252
- Division .................................................................................................. N1:=N1/N2 .......................... 253
- Trigonometrical functions ....................................................................... SIN / COS / TAN ..................... 254
- Root function .......................................................................................... R100:=SQRT.10 ..................... 257
- Integer component .................................................................................. R10:=INT.R70 ..................... 258
- Decimal component ............................................................................... R1:=FRAC.R90 ..................... 259
- Value ...................................................................................................... R2:=ABS.N23 ........................ 260
- Load axis position .................................................................................. R1:=A1 .................................. 247
- Load axis parameter ............................................................................... R11:=RA1.2.......................... 248
- Load encoder position ........................................................................... R111:=ENC1 .......................... 168
- Image to outputs (binary) ....................................................................... G600.R2.2.4 .......................... 240
- Image to outputs (BCD) ........................................................................ G601.N3.2.4 .......................... 241
- Image to flag .......................................................................................... G602.R3.17.32 ...................... 242
- Image from inputs ................................................................................... G603.R4.2.8 .......................... 244
- Image from flags ..................................................................................... G604.R4.2.8 .......................... 245
- Storage of current values in a program file ............................................ STORE TA2:=PE_001 ....... 173
- Storage if contents of register ............................................................... CASE.STORE.N3 ............... 175

3.1.9 \textbf{String}

- Copy ........................................................................................................ S0:=CHN # S1:=S0 .............. 222
- Input via keyboard ................................................................................ G542.10.2.3.N1 ...................... 237
- Value transfer from current data channel .............................................. G531.R1 ERROR ..................... 214
- Value transfer from current data channel .............................................. G532.13 FINE ......................... 214
- Search for character in the string .......................................................... N1:=POS1.35 .......................... 226
- Copy character to register ...................................................................... R1:=COPY.2.5 ERROR ......... 226
- Transfer characters to register ............................................................. N3:=GET.5.2 .......................... 231
- Transfer characters to register ............................................................. N4:=GETI.3.2 .......................... 232
- Transfer register to string ...................................................................... PUT.1.4 .................................. 233
- Transfer register to string ...................................................................... PUTI.4.2 .................................. 234
3.1.10 Program branches

Always jump ................................................................. JMP CYCLE 97
Jump if (logical) condition is met ................................. G21 I5.1 DRILL 103
Jump if contents of N register ................................. CASE.JMP.N8 110
Jump if monitoring time has elapsed .................. G421.1.500 SAVE_1 191
Jump if axis at standstill / moving ......................... G211.A1.0 WAIT 179
Jump if current position of the axis .................. G221.A2.1 READY 184

3.1.11 Call subroutines

End .................................................................................. END ...............................................
Always call ............................................................. SUB LAMBLI ........................ 100
Call if (logical) condition is met ................................. G22 I5.1 SAW ........................ 108
Call if contents of N register ................................. CASE.SUB.N8 ....................... 112
Call if monitoring time has elapsed .................. G422.1.500 ERROR .............. 192
Call if time monitoring has elapsed with repetition .................. G423.1.200 ERRORO_2........ 193
Call if axis at standstill / moving ................................. G211.A0.0 ELAPS ................. 179
Call if current position of the axis .................. G221.1.A2.1500 PASTE ........ 184

3.1.12 Loops

Increment counter ................................................... INC .N2.N3 START ................ 116
Decrement counter .................................................. DEC .N3 CYCLE .................... 114

3.1.13 Parallel runs

Start parallel run .................................................. RUN FLAG ............................. 118
Delete parallel run .................................................. CANCEL OFF1 ..................... 121
Stop parallel run .................................................. SLEEP BLINK ...................... 120
Start if contents of register ......................................... CASE.RUN.N8 ................. 125
Delete if contents of register ......................................... CASE.CANCEL.N10 .......... 129
Stop if contents of register ......................................... CASE.SLEEP.N5 ............... 127
Get program status .................................................. PROGSTAT ...................... 131
3.1.14 Serial port

Initialize ................................................................. G500.1.6.1.0 ........................................... 200
Output text ................................................................. G510.ERROR # G511.ERR ...................... 206
Output text line from a PTX file ................................. G515.5.ERRORTEXT ........................... 209
Output special character .............................................. G512.36 ............................................. 208
Output contents of register ........................................... G520.R2 # G521.N5 ............................ 210
Output status of flag, output or input ........................... G520.I7 # G521.M25 ........................... 210

Transfer content of register ........................................ G531.R7 ERROR_1 ......................... 214
Transfer status of flag or output .................................. G531.M34.DATA ............................ 214
Transfer string ........................................................... G531.R2 ERROR .............................. 214
Transfer string in background ...................................... G533.13 ....................................... 217
Transfer defined number of characters ......................... G534.4 ....................................... 219

Check transfer from port .......................................... N5:=CHN ..................................... 221

3.1.15 Display

Switch off run displays .............................................. G11.0 .............................................. 3-36
Initialize ................................................................. G500.0 ............................................. 200
Clear display .............................................................. G501 ............................................. 202
Delete line ................................................................. G502 ............................................. 203
Position cursor .......................................................... G503.4.1 # G504.N2.N6 ..................... 204

Output text ............................................................... G510.MACHINE ERROR ............. 206
Output special character ............................................ G512.31 ....................................... 3-98
Output text line from a PTX file ................................ G515.3.MESSAGE ....................... 209
Output contents of register ........................................ G520.R10 ..................................... 210
Output status of flag, output or input ........................... G520.M122 # G520.O6 .................. 210

3.1.16 Keyboard

Initialize ................................................................. G500.0 ........................................... 200
Check if key has been pressed .................................... G540.N6 ..................................... 232
Get character (key code) ............................................. G541.N2 ..................................... 236
Input register ........................................................... G542.30.1.3.N1 ............................ 237
Poll key switch ......................................................... LD SM3.0 ...................................... 79

3.1.17 Mathematics

Addition ................................................................. R2:=R2+57 # R1:=R1+R2 .................. 249
Subtraction ............................................................ R4:=R4-33 # R5:=R5-R6 .................. 251
Multiplication ......................................................... N2:=N1*12 # N3:=N2*N4 ............. 253
Division ................................................................. N5:=N5/12 # N6:=N6/N7 ............. 252
Trigonometrical functions ........................................ B2:=SIN.45 # R3:=SIN.R12 ........... 254
Root function ........................................................ R99:=SQRT.R43 ............................ 257
Integer component .................................................... R45:=INT.R70 ............................. 258
Decimal component ................................................... R43:=FRAC.R55 .......................... 259
Value ................................................................. N55:=ABS.N22 ............................... 260
3.1.18 Comparisons

Greater ................................................................. M11:=N22>12 ..................263
Equal to ............................................................... M12:=R21=R22 .................263
Smaller ............................................................... M13:=R23<R24 ..................263
Not equal to .......................................................... M113:=N100<>123 ..........263

3.1.19 Logic operations

LD ................................................................. LD I3.1 .........................269
AND ......................................................... AND I4.0 .........................269
OR ................................................................. OR I5.1 .........................269
SET .............................................................. SET O6 .........................269
NOT ............................................................... NOT O7 .........................269
RESET .......................................................... RESET M56 .................269
Multilevel AND (brackets) .................................. AND-LD ......................274
Multilevel OR (brackets) .................................... OR-LD ......................275

3.1.20 A-D converter

Get A-D values .................................................. N1:=AD1 # R1:=AD1 ..........278
A-D measuring series synchronous to axis movement .... G180.A2.R22.R23.1.100.25 ... 279
A-D measuring series with time base ...................... G181.5.1.100.50 ............ 281

3.1.21 D-A converter

Set D-A value (increments) .................................. DA1:=N2 ......................282
3.2 Programming tips

3.2.1 The run interpreter

PA-CONTROL is a sequence controller. It processes one command after another (e.g. traverse axis, set output, wait for input, etc.). Once a command has been processed, the next command in the program is executed.

The sequence controller has been designed for parallel sequence control, in order to be able to control parallel processes in a machine.

The run interpreter can process up to 31 parallel processes simultaneously. Each parallel process can call subroutines up to a nesting depth of 16 programs. This is referred to as multitasking.

PNC, PNX und PAB programs are available as executable programs. All commands can be used irrespective of the program type.

Text data can be saved in an available PTX programme (PTX files). In this programme presentable ASCII-signs should be used only.

NOTE Please do not use the ASCII-signs 01hex (SOH), 02hex (STX), 03hex (ETX) and 04hex (EOT), as they disturb the transfer protocol between the PA-CONTROL and WINPAC.

The difference between PNC, PNX und PAB programs can only be noticed when PA-CONTROL is stopped (STOP_MODE).

When activating a parallel process (RUN, ...), the program type is stored, so that the parallel process can be handled appropriately in the STOP MODE.

PNC processes: Axes are stopped, commands will not be further processed.
PNX processes: Axes are stopped, commands will not be further processed.
PAB processes: If a traverse command is currently active, the axis is stopped and the command is not processed any further. Otherwise all the other commands are processed as if STOP were not active.

This difference enables e.g. operator prompting or communication via serial ports to be continued in STOP MODE.

Each parallel process has an individual pointer to an actual data channel and an individual (local) S0 character buffer (max. 80 characters), which is deleted at the start of the automatic mode.

The system does not change any outputs on entering the automatic mode (START). The system resets all outputs when the automatic mode is exited (CANCEL).
3.2.2 System functions

The following definitions can be made for system functions:

- Program at START (Start program)
- Program at STOP (Stop program)
- Program at START after STOP (START after STOP program)
- Program at MALFUNCTION

The above system functions can be activated in different ways.

<table>
<thead>
<tr>
<th>Activated by</th>
<th>START</th>
<th>STOP</th>
<th>START after STOP</th>
<th>END</th>
<th>CANCEL</th>
<th>MALFUNCTION</th>
</tr>
</thead>
<tbody>
<tr>
<td>Keyboard</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Digital inputs (external)</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>&quot;Start program&quot;</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PAB process</td>
<td>-</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>PNC process/PNX process</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Diagnosis port</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RS232 Online</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Interbus-S command</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>Profibus-DP command</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>RS232 monitoring</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Interbus-S monitoring</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Profibus-DP monitoring</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>Axis monitoring</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>X</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: System functions

System function START:

The program defined as "program at START" (start program) is registered in the first parallel run and processed. Further parallel runs can be executed as required by means of the RUN commands.

When the start program reaches the END command, the automatic operation is ended.

System function STOP:

All active positioning operations are stopped and all parallel runs started as PNC/PNX programs are not processed any further (like SLEEP). After all the positioning operations have been stopped, the stop program is processed (if defined).

The parallel runs started as PAB programs are only affected by the stop operation in that the positioning operations are stopped. Otherwise they are further processed.
**System function START after STOP:**
The START after STOP program is processed after the end of the stop program (if defined). All stopped positioning operations are then restarted and all parallel runs further processed.

**System function END-CANCEL:**
All positioning operations are stopped. Parallel runs are not processed any further, and the automatic mode of the controller is exited (System function MALFUNCTION). A stop program (if defined) is not processed.

**System function MALFUNCTION:**
All positioning operations are stopped. Parallel runs are not processed any further. The "Program at MALFUNCTION" will be processed (if defined). The automatic mode is only exited after confirmation.

The following confirmations are possible:
- BREAK command
- ESC key on the keyboard
- "External START" via digital input without "External STOP"
- CANCEL command from port (Profibus-DP, RS232-Online, etc.)

After exiting the "AUTOMATIC mode", an attempt is made to reset the malfunction.

The system malfunctions are archived in the controller. Up to 50 malfunctions with the following information content can be stored:
- Time (from operating hours counter)
- Error number (with text)

If more than 50 system errors have occurred, the oldest messages will be overwritten.
3.2.3 Control of the automatic mode via external inputs

Definition:

<table>
<thead>
<tr>
<th>Input/Output</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I [START]</td>
<td>Start input from the system parameters (push button NO contact, output from a master controller)</td>
</tr>
<tr>
<td>I [STOP]</td>
<td>Stop input from the system parameters (push button NO contact, output from a master controller)</td>
</tr>
<tr>
<td>O [READY]</td>
<td>Ready output from the system parameters (signal lamp, input from a master controller)</td>
</tr>
<tr>
<td>O [AUTOMATIC ACTIVE]</td>
<td>Is set by a command (On: =) at the beginning of the program (signal lamp, input from a master controller)</td>
</tr>
<tr>
<td>O [AUTOMATIC STOPPED]</td>
<td>Is controlled by commands in the programs &quot;STOP&quot;, &quot;START AFTER STOP&quot; and &quot;MALFUNCTION&quot; (signal lamp, input from a master controller)</td>
</tr>
</tbody>
</table>

Table 2: Definition of external inputs

3.2.3.1 Start of automatic operation with Cancel:

Flow diagram 1: Start automatic mode

Description of the statuses:

1.0 → PA-CONTROL is in the initial position, no error
1.1 → Automatic mode is started
1.2 → PA-CONTROL is in automatic mode, no error
1.3 → PA-CONTROL is stopped
Waiting time of "START" t1 ≥ 10 ms
1.4 → PA-CONTROL aborts the automatic mode and switches to the initial position
1.5 → PA-CONTROL is in the initial position, no error
3.2.3.2 Start of automatic operation with STOP and START AFTER STOP:

Flows diagram 2: Start automatic mode with STOP and START after STOP

Description of the statuses:

2.0 → PA-CONTROL is in the initial position, no error
2.1 → Automatic mode is started
2.2 → PA-CONTROL is in automatic mode, no error
2.3 → PA-CONTROL is stopped
2.4 → STOP input is de-activated
   Waiting time t ≥ 2ms
2.5 → START input is activated, PA-CONTROL continues in automatic mode
2.6 → PA-CONTROL is in automatic mode, no error
3.2.3.3 PA-CONTROL is in automatic mode and a MALFUNCTION occurs:

Flow diagram 3: Sequence of malfunction in automatic mode

Description of the statuses:

3.0 → PA-CONTROL is in AUTOMATIC mode, no error

3.1 → Malfunction occurs

3.2 → Malfunction is acknowledged, AUTOMATIC mode is aborted, PA-CONTROL switches to the initial position

3.3 → PA-CONTROL is in the INITIAL POSITION and the malfunction is corrected, possible errors:
- Limit switch actuated
- Value too high
- Rotation monitoring
- Power section not ready
- Temperature error

3.4 → PA-CONTROL is in the initial position, the malfunction is still present, possible cause:
- Error in the supply voltage
- etc.

NOTE PA-CONTROL must be switched off and the hardware error corrected!
3.2.3.4 PA-CONTROL is in the initial position and a malfunction is present:

Flow diagram 4: Sequence of malfunction, PA-CONTROL in the initial position

Description of the statuses:

4.0 → PA-CONTROL is in the initial position, a malfunction is present
   Waiting time $t_1 \geq 80$ms
4.1 → The error is reset
4.2 → The error has been reset, PA-CONTROL signals READY
3.2.4 Program structure

A program has a program name and an extension. The extension identifies the program type. The program name consists of a maximum of 20 alphanumeric characters (0-9, A-Z). The only special characters which are legal in the name are the hyphen "-" and underscore "_". The program type consists of three letters.

A program consists of single commands which are processed one after the other and must be concluded with the END command.

A program can call a further program as a subroutine by means of the SUB command (up to 16 nesting levels are possible) or activate a further parallel run by means of the RUN command (up to 31 parallel run programs).

Conditional jumps and unconditional jumps to labels can be executed within a program. The label names are only known within a program. The same label names can therefore be used in different programs.

Jump commands from one program to another program are not possible.

A label name consists of a maximum of 20 alphanumeric characters (0-9, A-Z). The only special charaters which are legal in the name are the hyphen "-" and underscore "_". The label name is concluded by a blank character, semicolon or the line end. A label must be defined in the first column of a line headed by the character "$". A label can only be defined once within a program. The label can be jumped to from different positions in the program.
3.3 Programming elements

Various elements are used for programming in PA-CONTROL:

Inputs: An input can be polled for its logical status (see Chapter 3, page 91).

Outputs: An output can be assigned with the logical status 0 or 1 (Chapter 3, page 92). Its logical status can be polled for conditional jumps (Chapter 3, page 95).

Flags: A flag can be assigned with the logical status 0 or 1 (Chapter 3, page 96) and its logical status can be polled (max. 1-2048) (Chapter 93).

System flags: The intrasystem flags (logical status 0 or 1) can only be polled by the user. (See list of system flags).

Times: Times are used to create defined waiting times (Chapter 3, page 97).

Real number registers: Real number registers (1-4096) are special memory areas for real numbers (+/- 8,000,000,000). The contents of the registers can be used for positioning as well as for calculation and comparison operations. Up to 3 decimal places may be used (Chapter 3, page 67).

Integer registers: Integer registers (1-4096) are special memory areas for whole numbers (+/-2^31, i.e. 2,147,483,648). The contents of the registers can be used for times as well as for count, calculation and comparison operations. No decimals may be used (Chapter 3, page 67).

System registers: integer The intrasystem integer registers can only be polled by the user (see list of system integer registers).

System real number registers: The intrasystem real number registers can only be polled by the user (see list of system real number registers).

Strings: Strings are used to store up to 80 ASCII characters received e.g. via the serial port. Global strings S1 to S16 are available for access by each process. Each process also has its individual local S0. The contents of the global strings remain in the controller until they are overwritten. The local strings are reset at each start of the automatic mode.

Axes: Axes are stepping motor or servomotor axes and are used for positioning. They are addressed by the names A1, A2, A3, A4, A5, A6, A7, A8 (only unless they exist) (Chapter 3, page 132).

Label: Labels are orientation points and serve as targets for conditional jumps and for unconditional jumps. Example of a label: $REDBLUE

Comment: A comment is introduced by a semicolon (;) and is used for program documentation (see examples in the command explanations) and operator prompting in the automatic mode.

Direct addressing: In direct addressing, a new value is transferred directly to registers, respectively a new status is assigned to flags.

Indirect addressing: Indirect addressing is identified by an "!" after the designator (R!5, M!6). In the indirect addressing of registers, there is a pointer to an additional register which is addressed by the contents of the called register and the assigned value is transferred to this register. In the case of indirect flag addressing, the flag points to the contents of the integer register with the same index and sets the contents of the integer register as an index of the flag to be addressed. Only available in PNC programs!
List of system flags:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Power On: The flag is set by PA-CONTROL after switch-on and after polling (LD SM1, G21 SM1.n, ...)</td>
</tr>
<tr>
<td>2</td>
<td>A keyboard is connected to PA-Control and ready for use.</td>
</tr>
<tr>
<td>3</td>
<td>The key switch on the PA-CONTROL keyboard is in the &quot;manual&quot; position (horizontal, key cannot be removed)</td>
</tr>
<tr>
<td>4</td>
<td>Not assigned</td>
</tr>
<tr>
<td>5</td>
<td>The &quot;G542 command&quot;, input of a register value, has been concluded with ENTER</td>
</tr>
<tr>
<td>6</td>
<td>The &quot;G171 command&quot;, write characters to a PTX file, was not executed correctly</td>
</tr>
<tr>
<td>7</td>
<td>Battery OK</td>
</tr>
<tr>
<td>8</td>
<td>Axis distance is greater than interpolation path distance</td>
</tr>
<tr>
<td>9</td>
<td>Interpolation running (0 = interpolation not active, 1 = interpolation running)</td>
</tr>
<tr>
<td>10</td>
<td>Flag status always &quot;0&quot;</td>
</tr>
<tr>
<td>11</td>
<td>Axis 1 was stopped by the G123 function</td>
</tr>
<tr>
<td>12</td>
<td>Axis 2 was stopped by the G123 function</td>
</tr>
<tr>
<td>13</td>
<td>Axis 3 was stopped by the G123 function</td>
</tr>
<tr>
<td>14</td>
<td>Axis 4 was stopped by the G123 function</td>
</tr>
<tr>
<td>15</td>
<td>Axis 5 was stopped by the G123 function</td>
</tr>
<tr>
<td>16</td>
<td>Axis 6 was stopped by the G123 function</td>
</tr>
<tr>
<td>17</td>
<td>Axis 7 was stopped by the G123 function</td>
</tr>
<tr>
<td>18</td>
<td>Axis 8 was stopped by the G123 function</td>
</tr>
<tr>
<td>19</td>
<td>Axis 9 was stopped by the G123 function</td>
</tr>
<tr>
<td>20</td>
<td>Axis 10 was stopped by the G123 function</td>
</tr>
<tr>
<td>21</td>
<td>Axis 11 was stopped by the G123 function</td>
</tr>
<tr>
<td>22</td>
<td>Axis 12 was stopped by the G123 function</td>
</tr>
<tr>
<td>23</td>
<td>Axis 13 was stopped by the G123 function</td>
</tr>
<tr>
<td>24</td>
<td>Axis 14 was stopped by the G123 function</td>
</tr>
<tr>
<td>25</td>
<td>Axis 15 was stopped by the G123 function</td>
</tr>
<tr>
<td>26</td>
<td>Axis 16 was stopped by the G123 function</td>
</tr>
<tr>
<td>27</td>
<td>CANOpen control console &quot;Sütron&quot; ID 63 is active</td>
</tr>
<tr>
<td>28</td>
<td>CANOpen control console &quot;Sütron&quot; ID 62 is active</td>
</tr>
<tr>
<td>29</td>
<td>CANOpen control console &quot;Sütron&quot; ID 61 is active</td>
</tr>
<tr>
<td>30</td>
<td>CANOpen control console &quot;Sütron&quot; ID 60 is active</td>
</tr>
<tr>
<td>31</td>
<td>Axis 1 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>32</td>
<td>Axis 2 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>33</td>
<td>Axis 3 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>34</td>
<td>Axis 4 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>35</td>
<td>Axis 5 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>36</td>
<td>Axis 6 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>37</td>
<td>Axis 7 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>38</td>
<td>Axis 8 is ready / stand-by input is hot</td>
</tr>
</tbody>
</table>
### Part 2 List of system flags:

<table>
<thead>
<tr>
<th>No.:</th>
<th>Function:</th>
</tr>
</thead>
<tbody>
<tr>
<td>39</td>
<td>Achse 9 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>40</td>
<td>Achse 10 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>41</td>
<td>Achse 11 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>42</td>
<td>Achse 12 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>43</td>
<td>Achse 13 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>44</td>
<td>Achse 14 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>45</td>
<td>Achse 15 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>46</td>
<td>Achse 16 is ready / stand-by input is hot</td>
</tr>
<tr>
<td>47</td>
<td>Flashing flag 0, 160ms cycle (pulse plus pause)</td>
</tr>
<tr>
<td>48</td>
<td>Flashing flag 1, 320ms cycle (pulse plus pause)</td>
</tr>
<tr>
<td>49</td>
<td>Flashing flag 2, 640ms cycle (pulse plus pause)</td>
</tr>
<tr>
<td>50</td>
<td>Flashing flag 3, 1280ms cycle (pulse plus pause)</td>
</tr>
<tr>
<td>51</td>
<td>Achse 1 has been referenced</td>
</tr>
<tr>
<td>52</td>
<td>Achse 2 has been referenced</td>
</tr>
<tr>
<td>53</td>
<td>Achse 3 has been referenced</td>
</tr>
<tr>
<td>54</td>
<td>Achse 4 has been referenced</td>
</tr>
<tr>
<td>55</td>
<td>Achse 5 has been referenced</td>
</tr>
<tr>
<td>56</td>
<td>Achse 6 has been referenced</td>
</tr>
<tr>
<td>57</td>
<td>Achse 7 has been referenced</td>
</tr>
<tr>
<td>58</td>
<td>Achse 8 has been referenced</td>
</tr>
<tr>
<td>59</td>
<td>Achse 9 has been referenced</td>
</tr>
<tr>
<td>60</td>
<td>Achse 10 has been referenced</td>
</tr>
<tr>
<td>61</td>
<td>Achse 11 has been referenced</td>
</tr>
<tr>
<td>62</td>
<td>Achse 12 has been referenced</td>
</tr>
<tr>
<td>63</td>
<td>Achse 13 has been referenced</td>
</tr>
<tr>
<td>64</td>
<td>Achse 14 has been referenced</td>
</tr>
<tr>
<td>65</td>
<td>Achse 15 has been referenced</td>
</tr>
<tr>
<td>66</td>
<td>Base programme processed an END-command</td>
</tr>
<tr>
<td>67</td>
<td>Reserve</td>
</tr>
<tr>
<td>68</td>
<td>Reserve</td>
</tr>
<tr>
<td>69</td>
<td>Reserve</td>
</tr>
<tr>
<td>70</td>
<td>Reserve</td>
</tr>
</tbody>
</table>
Part 3 List of system flags:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function:</th>
</tr>
</thead>
<tbody>
<tr>
<td>71</td>
<td>Axis 1 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>72</td>
<td>Axis 2 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>73</td>
<td>Axis 3 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>74</td>
<td>Axis 4 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>75</td>
<td>Axis 5 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>76</td>
<td>Axis 6 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>77</td>
<td>Axis 7 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>78</td>
<td>Axis 8 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>79</td>
<td>Axis 9 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>80</td>
<td>Axis 10 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>81</td>
<td>Axis 11 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>82</td>
<td>Axis 12 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>83</td>
<td>Axis 13 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>84</td>
<td>Axis 14 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>85</td>
<td>Axis 15 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>86</td>
<td>Axis 16 is released for traverse (G25.A0)</td>
</tr>
<tr>
<td>87</td>
<td>Reserve</td>
</tr>
<tr>
<td>88</td>
<td>Reserve</td>
</tr>
<tr>
<td>89</td>
<td>Reserve</td>
</tr>
<tr>
<td>90</td>
<td>Reserve</td>
</tr>
<tr>
<td>91</td>
<td>Axis 1 is running</td>
</tr>
<tr>
<td>92</td>
<td>Axis 2 is running</td>
</tr>
<tr>
<td>93</td>
<td>Axis 3 is running</td>
</tr>
<tr>
<td>94</td>
<td>Axis 4 is running</td>
</tr>
<tr>
<td>95</td>
<td>Axis 5 is running</td>
</tr>
<tr>
<td>96</td>
<td>Axis 6 is running</td>
</tr>
<tr>
<td>97</td>
<td>Axis 7 is running</td>
</tr>
<tr>
<td>98</td>
<td>Axis 8 is running</td>
</tr>
<tr>
<td>99</td>
<td>Axis 9 is running</td>
</tr>
<tr>
<td>100</td>
<td>Axis 10 is running</td>
</tr>
<tr>
<td>101</td>
<td>Axis 11 is running</td>
</tr>
<tr>
<td>102</td>
<td>Axis 12 is running</td>
</tr>
</tbody>
</table>
### Part 4 List of system flags:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function:</th>
</tr>
</thead>
<tbody>
<tr>
<td>103</td>
<td>Axis 13 is running</td>
</tr>
<tr>
<td>104</td>
<td>Axis 14 is running</td>
</tr>
<tr>
<td>105</td>
<td>Axis 15 is running</td>
</tr>
<tr>
<td>106</td>
<td>Axis 16 is running</td>
</tr>
<tr>
<td>107</td>
<td>Reserve</td>
</tr>
<tr>
<td>108</td>
<td>Reserve</td>
</tr>
<tr>
<td>109</td>
<td>Reserve</td>
</tr>
<tr>
<td>110</td>
<td>Reserve</td>
</tr>
<tr>
<td>111</td>
<td>Axis 1 has stopped</td>
</tr>
<tr>
<td>112</td>
<td>Axis 2 has stopped</td>
</tr>
<tr>
<td>113</td>
<td>Axis 3 has stopped</td>
</tr>
<tr>
<td>114</td>
<td>Axis 4 has stopped</td>
</tr>
<tr>
<td>115</td>
<td>Axis 5 has stopped</td>
</tr>
<tr>
<td>116</td>
<td>Axis 6 has stopped</td>
</tr>
<tr>
<td>117</td>
<td>Axis 7 has stopped</td>
</tr>
<tr>
<td>118</td>
<td>Axis 8 has stopped</td>
</tr>
<tr>
<td>119</td>
<td>Axis 9 has stopped</td>
</tr>
<tr>
<td>120</td>
<td>Axis 10 has stopped</td>
</tr>
<tr>
<td>121</td>
<td>Axis 11 has stopped</td>
</tr>
<tr>
<td>122</td>
<td>Axis 12 has stopped</td>
</tr>
<tr>
<td>123</td>
<td>Axis 13 has stopped</td>
</tr>
<tr>
<td>124</td>
<td>Axis 14 has stopped</td>
</tr>
<tr>
<td>125</td>
<td>Axis 16 has stopped</td>
</tr>
<tr>
<td>126</td>
<td>Axis 16 has stopped</td>
</tr>
<tr>
<td>127</td>
<td>Reserve</td>
</tr>
<tr>
<td>128</td>
<td>Reserve</td>
</tr>
<tr>
<td>129</td>
<td>Reserve</td>
</tr>
<tr>
<td>130</td>
<td>Reserve</td>
</tr>
<tr>
<td>131</td>
<td>Reserve</td>
</tr>
<tr>
<td>132</td>
<td>Reserve</td>
</tr>
<tr>
<td>...</td>
<td>Reserve</td>
</tr>
<tr>
<td>210</td>
<td>Reserve</td>
</tr>
</tbody>
</table>
### Part 5 List of system flags:

<table>
<thead>
<tr>
<th>No.:</th>
<th>Function:</th>
<th>0 = Axis is not traversed by an interpolation</th>
<th>1 = Axis is traversed by an interpolation</th>
</tr>
</thead>
<tbody>
<tr>
<td>211</td>
<td>Axis 1 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>212</td>
<td>Axis 2 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>213</td>
<td>Axis 3 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>214</td>
<td>Axis 4 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>215</td>
<td>Axis 5 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>216</td>
<td>Axis 6 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>217</td>
<td>Axis 7 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>218</td>
<td>Axis 8 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>219</td>
<td>Axis 9 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>220</td>
<td>Axis 10 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>221</td>
<td>Axis 11 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>222</td>
<td>Axis 12 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>223</td>
<td>Axis 13 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>Axis 14 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>225</td>
<td>Axis 15 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>226</td>
<td>Axis 16 is busy with an interpolation</td>
<td></td>
<td></td>
</tr>
<tr>
<td>227</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>228</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>229</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>230</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>231</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>232</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>233</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>234</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>235</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>236</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>237</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>238</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>239</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>240</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>241</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
<tr>
<td>242</td>
<td>Reserve</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
List of system N registers:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
</tr>
</thead>
</table>
| 1   | System run time of PA-CONTROL in seconds  
( the run time is only reset after a new operating system has been loaded ) |
| 2   | System run time of PA-CONTROL in tenths of a second  
( the run time is only reset after a new operating system has been loaded ) |
| 3   | Measurement of run times with high resolution from V4.74G register SN3 is reset to 0 in the event of polling and each millisecond is incremented by 1 by PA-CONTROL. As a result, the time in milliseconds between two polls can be determined by means of register SN3, e.g. for measurement of the run time. |
| 4   | Reserve |
| 5   | PA-CONTROL version |
| 6   | PA-CONTROL serial number |
| 7   | Reserve |
| 8   | Reserve |
| 9   | Reserve |
| 10  | System error number |
| 11  | Run error number |
| 12  | Number of the axis or of one axis from an axis group (concerned in an error) |
| 13  | Number of the parallel sequence with the error |
| 20  | PA-CONTROL was stopped by  
<SN20> = 1 → STOP by control console (IEF or CANOpen)  
<SN20> = 2 → STOP by external STOP input  
<SN20> = 3 → STOP by diagnosis port (WINPAC)  
<SN20> = 4 → STOP by Profibus DP  
<SN20> = 5 → STOP by RS232-ONLINE CMD port  
<SN20> = 6 → STOP by Interbus-S  
<SN20> = 7 → STOP by PAB program |
| 21  | PA-CONTROL is in stop mode  
If PA-CONTROL is stopped in automatic mode, various steps, which can be polled in SN21, are completed in the STOPMODE.  
<SN21> = 0 → No STOP  
<SN21> = 1 → STOP detected  
<SN21> = 2 → Wait until all axes are at standstill  
<SN21> = 3 → Start program after STOP  
<SN21> = 4 → Process program after STOP  
<SN21> = 5 → Stopped  
<SN21> = 6 → Stopped and wait for Continue  
<SN21> = 7 → START after STOP detected, start stopped axes  
<SN21> = 8 → Process program START after STOP  
<SN21> = 9 → Activate Process all programs |
Part 2 List of system N registers:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>31</td>
<td>Axis 1 / Counter absolute positioning system</td>
</tr>
<tr>
<td>32</td>
<td>Axis 2 / Counter absolute positioning system</td>
</tr>
<tr>
<td>33</td>
<td>Axis 3 / Counter absolute positioning system</td>
</tr>
<tr>
<td>34</td>
<td>Axis 4 / Counter absolute positioning system</td>
</tr>
<tr>
<td>35</td>
<td>Axis 5 / Counter absolute positioning system</td>
</tr>
<tr>
<td>36</td>
<td>Axis 6 / Counter absolute positioning system</td>
</tr>
<tr>
<td>37</td>
<td>Axis 7 / Counter absolute positioning system</td>
</tr>
<tr>
<td>38</td>
<td>Axis 8 / Counter absolute positioning system</td>
</tr>
<tr>
<td>39</td>
<td>Axis 9 / Counter absolute positioning system</td>
</tr>
<tr>
<td>40</td>
<td>Axis 10 / Counter absolute positioning system</td>
</tr>
<tr>
<td>41</td>
<td>Axis 11 / Counter absolute positioning system</td>
</tr>
<tr>
<td>42</td>
<td>Axis 12 / Counter absolute positioning system</td>
</tr>
<tr>
<td>43</td>
<td>Axis 13 / Counter absolute positioning system</td>
</tr>
<tr>
<td>44</td>
<td>Axis 14 / Counter absolute positioning system</td>
</tr>
<tr>
<td>45</td>
<td>Axis 15 / Counter absolute positioning system</td>
</tr>
<tr>
<td>46</td>
<td>Axis 16 / Counter absolute positioning system</td>
</tr>
<tr>
<td>47</td>
<td>Reserve</td>
</tr>
<tr>
<td>48</td>
<td>Reserve</td>
</tr>
<tr>
<td>49</td>
<td>Reserve</td>
</tr>
<tr>
<td>50</td>
<td>Reserve</td>
</tr>
</tbody>
</table>
Part 3 List of system N registers:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>51</td>
<td>Axis 1 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>52</td>
<td>Axis 2 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>53</td>
<td>Axis 3 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>54</td>
<td>Axis 4 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>55</td>
<td>Axis 5 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>56</td>
<td>Axis 6 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>57</td>
<td>Axis 7 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>58</td>
<td>Axis 8 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>59</td>
<td>Axis 9 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>60</td>
<td>Axis 10 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>61</td>
<td>Axis 11 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>62</td>
<td>Axis 12 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>63</td>
<td>Axis 13 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>64</td>
<td>Axis 14 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>65</td>
<td>Axis 15 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>66</td>
<td>Axis 16 / Number of A-D values actually got by the G180 command</td>
</tr>
<tr>
<td>67</td>
<td>Reserve</td>
</tr>
<tr>
<td>68</td>
<td>Reserve</td>
</tr>
<tr>
<td>69</td>
<td>Reserve</td>
</tr>
<tr>
<td>70</td>
<td>Reserve</td>
</tr>
<tr>
<td>71</td>
<td>Reserve</td>
</tr>
<tr>
<td>72</td>
<td>Reserve</td>
</tr>
<tr>
<td>73</td>
<td>Reserve</td>
</tr>
<tr>
<td>74</td>
<td>Reserve</td>
</tr>
<tr>
<td>75</td>
<td>Reserve</td>
</tr>
<tr>
<td>76</td>
<td>Reserve</td>
</tr>
<tr>
<td>101</td>
<td>COM 1</td>
</tr>
<tr>
<td>102</td>
<td>COM 2</td>
</tr>
<tr>
<td>103</td>
<td>COM 3</td>
</tr>
<tr>
<td>104</td>
<td>COM 4</td>
</tr>
<tr>
<td>105</td>
<td>COM 5</td>
</tr>
<tr>
<td>106</td>
<td>COM 6</td>
</tr>
<tr>
<td>107</td>
<td>COM 7</td>
</tr>
<tr>
<td>108</td>
<td>COM 8</td>
</tr>
</tbody>
</table>
List of system R registers:

<table>
<thead>
<tr>
<th>No.</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>1...9</td>
<td>Reserve</td>
</tr>
<tr>
<td>10</td>
<td>Reference path speed for interpolation</td>
</tr>
<tr>
<td>11...30</td>
<td>Reserve</td>
</tr>
<tr>
<td>31</td>
<td>Axis A1 / Position in the absolute system</td>
</tr>
<tr>
<td>32</td>
<td>Axis A2 / Position in the absolute system</td>
</tr>
<tr>
<td>33</td>
<td>Axis A3 / Position in the absolute system</td>
</tr>
<tr>
<td>34</td>
<td>Axis A4 / Position in the absolute system</td>
</tr>
<tr>
<td>35</td>
<td>Axis A5 / Position in the absolute system</td>
</tr>
<tr>
<td>36</td>
<td>Axis A6 / Position in the absolute system</td>
</tr>
<tr>
<td>37</td>
<td>Axis A7 / Position in the absolute system</td>
</tr>
<tr>
<td>38</td>
<td>Axis A8 / Position in the absolute system</td>
</tr>
<tr>
<td>39</td>
<td>Axis A9 / Position in the absolute system</td>
</tr>
<tr>
<td>40</td>
<td>Axis A10 / Position in the absolute system</td>
</tr>
<tr>
<td>41</td>
<td>Axis A11 / Position in the absolute system</td>
</tr>
<tr>
<td>42</td>
<td>Axis A12 / Position in the absolute system</td>
</tr>
<tr>
<td>43</td>
<td>Axis A13 / Position in the absolute system</td>
</tr>
<tr>
<td>44</td>
<td>Axis A14 / Position in the absolute system</td>
</tr>
<tr>
<td>45</td>
<td>Axis A15 / Position in the absolute system</td>
</tr>
<tr>
<td>46</td>
<td>Axis A16 / Position in the absolute system</td>
</tr>
</tbody>
</table>
3.4 Explanations

Command code

Description:
The command is described in general terms in this section.

NOTE Here we will point out particular features and provide you with help by means of cross references.

Command format:

mmmm     nnnnn

The commands are divided into an operand and an operator, in other words, a command code and a number.

NOTE The spaces between the operand and the operator have been included in the command format section for the sake of clarity. They are illegal for input into PA-CONTROL!

Several commands can be written in one program line. Commands in one line must be separated by at least one space.

Application:
This section explains the practical reference of the command.

CAUTION
The examples have been prepared without knowledge of the actual mechanical system and must always be checked to ensure their application is appropriate.

Example

1   I8.1   The controller waits until input 8 is logically 1
2   I4.0   The controller waits until input 4 is logically 0
3   END

The numbers at the beginning of the lines are the line numbers generated by the editor in PA-CONTROL and are only used for orientation in the program editor.

The examples can be transferred directly. As in PA-CONTROL, comments are separated by a semicolon and can also be transferred to the program provided they do not exceed the maximal line length of 80 characters. As a general principle, all programs are concluded with an END command.
NOTE

When transferring a program from the PC to PA-CONTROL, you can specify in the selection window for the program which is to be transferred whether the program should be transferred to the controller with or without comments.

If two semicolons are written in front of the comment during programming, the comment is always transferred to the controller!
3.5  **In.m - Waiting for logical status of input**

<table>
<thead>
<tr>
<th><strong>ln.m</strong></th>
</tr>
</thead>
</table>

*I (Number).(Status)*

**Description:**
The controller checks by means of the command I (input) whether the input n defined in the operand has the status m. If the poll result is not true, the controller waits until the condition is provided in operator m.

**Command format:**

I12.1

I12.0

**Application:**
Polling of proximity switches, push buttons, switches, etc. …

**Example**

1  I8.1       Controller waits until input 8 is logical 1
2  I4.0       Controller waits until input 4 is logical 0
3  END
3.6 On.m - Waiting for logical status of output

<table>
<thead>
<tr>
<th>On.m</th>
</tr>
</thead>
</table>

O (Number).(Status)

Description:
The controller checks by means of the command O (output) whether the output n defined in
the operand has the status m. If the polling result is not true, the controller waits until the
condition is provided in operator m.

NOTE
This command is only appropriate in case of multiple program runs in parallel,
because only another parallel process can change the output while this process
is waiting for the output.

Command format:
On:=1
On:=0
On:=Mm Status of flag m is transferred for output n
On:=Im Status of input m is transferred for output n
On:=Om Status of output m is transferred for output n

Application:
Polling of outputs

Example
1 i O8:=1 ;The controller sets output 8
(e.g. for the accurately timed start of a parallel run)
2 O7.1 ;Controller waits until output 7 is logical 1
(is set by a program operating in parallel)
3 END
### 3.7 Mn.m - Waiting for then logical status of a flag

**Mn.m**

**M (Number).(Status)**

**Description:**

The controller checks by means of the command M (flag) whether the flag n defined in the operand has the status m. If the polling result is not true, the controller waits until the condition is provided in operator m.

The flags can be addressed indirectly. The flag for indirect addressing is identified by an exclamation mark after the designator M. The number n after the exclamation mark points to the integer register with the same address. The content of the register provides the actual flag address. See the following example.

<table>
<thead>
<tr>
<th>N10:=5</th>
<th>→</th>
<th>The contents of integer register N10 is 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>M!10.1</td>
<td>→</td>
<td>The number 10 after the exclamation mark points to the contents of integer register N10 whose content =5. The controller now waits until the status of the flag M5 is equal to &quot;1&quot;</td>
</tr>
<tr>
<td>M5.1</td>
<td>→</td>
<td>The command M5.1 is identical to the previous example with indirect addressing</td>
</tr>
</tbody>
</table>

**Command format:**

<table>
<thead>
<tr>
<th>M1.1</th>
<th>Set flag 1 to status logical 1</th>
</tr>
</thead>
<tbody>
<tr>
<td>M1.0</td>
<td>Reset flag 1</td>
</tr>
<tr>
<td>M!3.1</td>
<td>Indirect addressing for flag polling</td>
</tr>
</tbody>
</table>

**Application:**

Synchronization of parallel runs.

**Example**

1. O8:=1 ; The controller sets output 8 (e.g. triggering for parallel run control)
2. M7.1 ; Controller waits until input 7 is logical 1 (was set by parallel run control)
3. N10:=5 ; Load N register for indirect addressing
4. M!10.1 ; Wait until M5 has the logical status 1
5. END
### 3.8 Ni.n - Waiting for the logical status of an N register

<table>
<thead>
<tr>
<th>Ni.Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>N (Number).(Value)</strong></td>
</tr>
</tbody>
</table>

**Description:**
Delays the processing of the following program section until the status of the N register is equal to the constant. If the polling result is not true, the controller waits until the condition is provided in operator n.

The register value is changed e.g. in a program operating in parallel.

**Command format:**
- Ni.n
- Ni.Nm
- NiI.n
- NiI.NI.m

**Application:**
Polling of program sections running in a cyclic time frame.

**Example**

1. $A
2. N2.5
   
   Controller waits until the content of register N2 is equal to 5
3. O1:=1 T10 O1:=0 T10
   
   Flashing of a lamp
4. JMP A
   
   Jump to A
5. END
3.9 On:=m - Set / reset output

On:=m

O (Number):=(Status)

Description:
The available outputs can be manipulated by means of the command O (output). If output n is set to the status \( m = 1 \), it completes the circuit, i.e. the transistor is conductive and remains switched on until it is reset by \( m = 0 \) and the transistor is off.

Command format:
On:=1
On:=0
On:=Mm Status of flag m is transferred for output n
On:=Im Status of input m is transferred for output n
On:=Om Status of output m is transferred for output n

Application:
Switching of relays or pneumatic valves, signal transmission to other electrical devices

Example

1  O8:=1 ; Output 8 is set to 1
2  O7.0 ; Controller waits until output 7 is logical 0
3  END
3.10 Mn:=m - Set / reset flag

<table>
<thead>
<tr>
<th>Mn:=m</th>
</tr>
</thead>
</table>

\[ M (\text{Number}):=(\text{Status}) \]

Description:
The available flags can be controlled by means of the command \( M \). If flag \( n \) is set to the logical status \( m=1 \), it remains in this status until it is reset with \( m=0 \) (see also description of commands for comparison operations).

Command format:
- \( M1:=0 \) Reset flag 1
- \( M1:=1 \) Set flag 1
- \( Mn:=Mm \) Status of flag \( m \) is transferred for flag \( n \)
- \( Mn:=Im \) Status of input \( m \) is transferred for flag \( n \)
- \( Mn:=Om \) Status of output \( m \) is transferred for flag \( n \)
- \( M!n:=M!m \) Indirect addressing

Application:
- Storage of statuses, processing situations
- Synchronization of parallel runs

Flags can be addressed indirectly from version 3.72 and higher. For indirect addressing, the flag is identified by an exclamation mark after the designator \( M \). The flag points to the contents of the integer register with the same index.

Command format:
- \( M!2:=0 \) → Indirect addressing when setting flags

The indirect addressing must be declared before application!

<table>
<thead>
<tr>
<th>N15:=10</th>
<th>The content of integer register N15 is 10</th>
</tr>
</thead>
<tbody>
<tr>
<td>M!15:=1</td>
<td>The number 15 after the exclamation mark points to the contents of integer register N10 whose content =5. Flag M10 is set to &quot;1&quot;.</td>
</tr>
</tbody>
</table>

Application:
- Parameter setting for programs

Example

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>I8.1 ;Controller waits until input 8 is logical 1</td>
</tr>
<tr>
<td>2</td>
<td>M12:=1 ;Flag 12 is set to logical 1</td>
</tr>
<tr>
<td>3</td>
<td>M10:=0 ;Flag 10 is set to logical 0</td>
</tr>
<tr>
<td>4</td>
<td>N15:=10 ;Load register N15 for indirect addressing</td>
</tr>
<tr>
<td>5</td>
<td>M!15:=1 ;M10 is set to logical 1 by the indirect addressing of M15</td>
</tr>
<tr>
<td>6</td>
<td>T100 ;Wait 1 second</td>
</tr>
<tr>
<td>7</td>
<td>N16:=10 ;Load register N16 for indirect addressing</td>
</tr>
<tr>
<td>8</td>
<td>M!16:=0 ;M10 is set to logical 0 by the indirect addressing of M16</td>
</tr>
<tr>
<td>9</td>
<td>END</td>
</tr>
</tbody>
</table>
3.11 Tn - Hold-up time

\[ T \ (Time \ in \ 10/milliseconds) \]

Description:
The command T causes the controller to hold for the time defined in the operand.
The following applies here:

<table>
<thead>
<tr>
<th>Time</th>
<th>n</th>
<th>1</th>
<th>2^{31} = 2.147.483.648</th>
<th>21.474.836.480 ms</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>n</td>
<td>1</td>
<td>2^{31} = 2.147.483.648</td>
<td>21.474.836.480 ms</td>
</tr>
<tr>
<td></td>
<td>10 ms</td>
<td>21.474.836.480 ms</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Command format:
T1000
TN12

Application:
Waiting times, keeping outputs set for defined times.

Example:
1 T1 ;Hold-up time 10ms
2 T300 ;Hold-up time 3s
3 TN2 ;Hold-up time according to the contents of N2
3 END

3.12 JMP - Unconditional jump

3.12.1 General form of the unconditional jump

\[ JMP \]

Description:
An "unconditional jump" to the defined label is executed in the program by means of the function JMP.

Command format:
JMP (Label)

Application:
For program loops
Example

1  $ENDLESS
2  I1.1 ;Wait until input 1 is hot
3  O1:=1
4  T100
5  O1:=0
6  JMP ENDLESS ;Jump to the label "ENDLESS"
7  END

NOTE
A jump target (label) or a program name can be denoted both in uppercase and lowercase characters. Numeric characters and letters can be used. PA-CONTROL works internally only with uppercase letters. The jump targets "BEGINNING" and "beginning" are identical!
3.12.2 Start of an interpolation at a specified program line

**JMP-LINE-IPO.Ni**

**Description:**
A specified line in the program can be jumped to and the interpolation started from there.

**Prerequisite:**
- An interpolation may not be active.
- A G01, FB or G160 command must be located in the programme line which is stated as a jump target. If this condition is not complied with, error "E577=incorrect jump target" is output.

**NOTE**
The path acceleration (G100.B...) and the path speed (FB...) must be set to appropriate values before execution of the command "JMP-LINE-IPO", since otherwise an error Exxx is output.

**NOTE**
An FB command is processed with JMP-IPO-LINE, and the next IPO command is then changed to.

**NOTE**
G16x commands are not processed, and the next IPO command is then changed to. The probability that a line number of a G16x command is recognised is very low, since the G16x commands are implemented in the IPO interval during the execution of the IPO commands. A G16x command is only recognised via a line number if the interpolation is aborted during the start phase (loading phase of the G01 intervals).
3.12.3 Start of a program at a specified program line

**JMP-LINE.Ni**

*Description:*
A specified line in the program can be jumped to by means of this command.

**NOTE**
If the jump target is in an interpolation (second, third G01 command) or if the command jumped to is a G161, G162 or FB command, error "E577=incorrect jump target" is output.

**NOTE**
The line number of an FB command cannot be determined via the functions "Ni:=LINE.Name" or "Ni:=LINE-Sn". As a result, it is not actually possible to jump to this command using the JMP-LINE (or JMP-LINE-IPO) command.
3.13 SUB - Subroutine call

Description:

The subroutine call SUB enables the user to create a well structured program. Thus, functions which are required several times can be stored in a subroutine. The programs thereby remain manageable and the subroutines can be tested individually.

The subroutine call SUB executes a jump to a program stored in PA-CONTROL. The name can be explicitly stated (see Chapter 3.13.1, page 101) or taken from a string (see Chapter 3.13.2, page 103), or an integer register (see Chapter 3.13.3, page 104).

NOTE A subroutine may not call itself. Subroutines must be concluded with the END command. Up to 15 subroutine calls may be nested.

NOTE A jump target (label) or a program name can be denoted both in uppercase and lowercase characters. Numeric characters and letters can be used. PA-CONTROL works internally only with uppercase letters. The jump targets "BEGINNING" and "beginning" are identical!

Command formats:

SUB.name (see Chapter 3.13.1, Page 101)
SUB.S0 (see Chapter 3.13.2, Page 103)
SUB.S1 (see Chapter 3.13.2, Page 103)
SUB.N1 (see Chapter 3.13.3, Page 104)
SUB.N2.PNX (see Chapter 3.13.3, Page 104)

3.13.1 Standard subroutine call

A jump to a stored program with the stated name is executed by the subroutine call SUB.name.

Command format:

SUB.name

Example on next page
Example

Example 1:

I2.1

SUB LAMBLI

;//Subroutine "LAMBLI" is called and processed
;//After the END command in the subroutine has
//been detected, the next command in the calling
//program is processed (in this case I3.1)

I3.1

I4.1

SUB LAMBLI

;//Subroutine "LAMBLI" is called and processed,
;//After the END command in the subroutine has
//been detected, the next command in the calling
//program is processed (in this case I5.1)

I5.1

END

Program: LAMBLI

I1  O1:=1

T100

O1:=0

END

Examples of nesting:

Program: Example

Program: LAMBLI

I2.1

SUB LAMBLI

→→→

1  O1:=1

2  T100

3  O1:=0

4  END

Program: Example

Program: TEST

Program: LAMBLI

I2.1

SUB TEST

→→→

1  T10

2  SUB LAMBLI

→→→

1  O1:=1

2  T100

3  O1:=0

4  END

Program: LAMBLI

I5.1

←←←

1  I6.1

3  I6.1

←←←

4  END

Program: Example

Program: LAMBLI

I5.1

←←←

1  I6.1

3  I6.1

←←←

4  END

Program: Example
3.13.2 Subroutine call, Program name is in a string

Command format:

SUB.S0.[Program type]
SUB.Sn.[Program type]

**SUB.S0 / SUB.Sn / SUB.S0.<Program type> / SUB.Sn.<Program type>**

The program name of the program to be called is in a string. If the program does not exist, error message "E513 – Program does not exist" is output.

The statement of the program type (PNC, PAB or PNX) is a command option. If a program type is not specified, the subroutine is called without a check of the program type.

If the program type, e.g. SUB.S0.PNX, is stated, a check is made as to whether the program exists and corresponds to the requested type. If there is no match, error message "E513 – Program does not exist" is output.

**Example**

Content of an S0 string in ASCII format: **BLINK**

```
1 $A
2 N2:=PROGSTAT.S0 Status check, program name from string S0
3 T1
4 M1:=N2>0 Program exists
5 G21 M1.0 A
6 SUB.S0 Call of the program whose name is in the S0 string
7 JMP A
8 END
```

**Program: BLINK**

```
1 $BEGINNING
2 O10:=1 Set output 10
3 T10 Wait 100ms
4 O10:=0 Reset output 10
5 T10 Wait 100ms
6 END
```
3.13.3 Subroutine call, program name is in an integer register

Command format:

SUB.Nn[.Program type]
SUB.Nn.i[.Program type]

SUB.Nn / SUB.Nn.i / SUB.Nn.<Program type> / SUB.Nn.i<Program type>

The content of the stated integer register is converted to a string consisting of numerical characters. This string provides the program name of the desired subroutine. At the command SUB.Nn.i, the N register is converted to a string with "i" characters. If the number in the N register is smaller than the specified number of places, leading zeros are added (see following example).

The statement of the program type (PNC, PAB or PNX) is a command option. If a program type is not specified, the subroutine is called without a check of the program type.

If the program type, e.g. SUB.N24.PNC, is stated, a check is made as to whether the program exists and corresponds to the requested type. If there is no match, error message "E513 – Program does not exist" is output.

**Example**

```
1  $A
2  N5:=4567  Load the number 4567 to integer register N5
... 
10  SUB.N5  Call of the program 4567 without check of the program type
...
20  SUB.N5.PNC  Call of the program 4567 and check of the program type PNC
...
30  SUB.N5.7  Call of the program 0004567 without check of the program type. The content of the register is converted with three leading zeros, i.e. 7 characters.
...
40  SUB.N5.10.PNX  Call of the program 0000004567 and check of the program type. The content of the register is converted with six leading zeros, i.e. 10 characters.

41  END
```

Program: 4567

```
1  $BEGINNING
2  O10:=1  Set output 10
3  T10  Wait 100ms
4  O10:=0  Reset output 10
5  T10  Wait 100ms
6  END
```
3.14  G11 – Switch display on / off

G11

G11.n

Command format:
G11.0
G11.1

Application:
For processes, in which the display is to be used for operator prompting and messages by means of the commands of the G500 command group.

Description:
The display of the commands during the program flow is controlled by means of the function G11. The display is switched off by means of the command G11.0 and switched on by means of G11.1.

NOTE
The last executed command is displayed irrespective of the program version.
The following commands are never displayed: JMP, M1:=, etc.

Only by this means is it possible to display a previous program line when programming loops.

Example:

1  G11.0  ;Switch off display
2  G500.0 ;Selects the LC display of PA-CONTROL
3  G501  ;Clears the display and replaces the cursor in the 1st column of the 1st line
4  G510.Machine error! ;Outputs the text "Machine error" to the current display medium, the LC display
5  G11.1 ;Switch on display
6  END
3.15 **G21 - Conditional jump**

<table>
<thead>
<tr>
<th>G21</th>
</tr>
</thead>
</table>

**G21 (Condition) (Label)**

**Description:**
A "conditional jump" is executed by means of the function G21. The condition defined in the operand can be the logical status of an input, output, flag or system flag. If the condition is fulfilled, a jump to the label defined in the operator is executed, and the program is continued there. If the condition is not fulfilled, the next line is processed.

**Command format:**
- G21 I1.0 Label
- G21 O2.1 Label
- G21 M4.1 Label
- G21 M!3.1 Label
- G21 SM51.1 Label

**Application:**
Program branch depending on the logical status of an input, output, flag or system flag.
Example

1  G21 I1.1 ON_5  ;If input 1 is hot, the processing of the program is continued at the label "ON_5", otherwise with the next command
2  O1:=1 T100 O1:=0  ;Output 1 set for 1 second
3  $EIN_5
4  T10
5  G21 O5.1 FLAG_31  ;If output 5 is set to 1, the program is continued at the label "FLAG_31", otherwise with the next command
6  O1:=1 T100 O1:=0  ;Output 1 set for 1 second
7  $FLAG_31
8  T10
9  G21 M31.0 END  ;If flag 31 is set to logical 0, the program is continued at the label "END", otherwise with the next command
10  O1:=1 T100 O1:=0  ;Output 1 set for 1 second
11  $END
12  T10
13  END
3.16 G22 - Conditional Subroutine call

G22 (Condition) (Name)

Description:
A "Conditional subroutine call" is executed by means of the function G22. The condition defined in the operand can be the logical status of an input, output, flag or system flag. If the condition is fulfilled, the subroutine call defined in the operand is executed, and the program is continued there. As soon as the subroutine has been processed or if the condition is not fulfilled, the next command in the calling program is processed.

Command format:
G22 I1.1 Name
G22 O2.1 Name
G22 M2.0 Name
G22 M!3.1 Name
G22 SM6.0 Name

Application:
A subroutine is executed depending on the logical status of an input, output, flag or system flag.
Example

2. G22 I1.1 OUTP_2 ;If input 1 is hot, the subroutine "OUTP_2" is executed and processing continued with the next command in the calling program
3. G22 M21.1 OUTP_2 ;If flag 21 is set, the subroutine "OUTP_2" is executed and processing continued with the next command in the calling program
4. G22 O7.1 OUTP_2 ;If output 7 is set, the subroutine "OUTP_2" is executed and processing continued with the next command in the calling program
5. END

Program: OUTP_2

1. O2:=1 T100 O2:=0 END ;Set output 2 for 1s
3.17 CASE.JMP - Jump distributor

CASE.JMP.N

CASE.JMP.(Variable)

Description:
The CASE.JMP command is a program branch depending on the contents of an integer register. Jump distributors can be easily created using this command.

PA-CONTROL checks the contents of the integer register and jumps to the labels in accordance with the value. If the content of the integer register is less than 1 or greater than the number of elements in the label table (in this example: (RED), (BLUE); (YELLOW), (PINK)), a jump to the label after ELSE is executed. Otherwise a jump to the corresponding label is executed.

Command format:

CASE.JMP.Nn
(Label1)
(Label2)
...
(Label_i)
ELSE Label_e

Or with indirect addressing of the register

CASE.JMP.N!n
(Label1)
(Label2)
...
(Label_i)
ELSE Label_e

Application:
Program branches in the case of more than two possibilities, e.g. various manufacturing types, branch by reason of a counter content, etc.
Example

1  $GREY
2  N8:=N13 ;Read value via register
3  CASE.JMP.N8 ;Check contents of N8 and branch if:
4   (RED) ;N8=1 : Jump to label "RED"
5   (BLUE) ;N8=2 : Jump to label "BLUE"
6   (YELLOW) ;N8=3 : Jump to label "YELLOW"
7   (PINK) ;N8=4 : Jump to label "PINK"
8   ELSE GREY ;Jump to label "GREY" if N8<1 or N8>4 (value of N8 is not within the specified range of 1 to 4)
9  $RED
10  O1:=1 T100 O1:=0 ;Set output 1 for 1s
11  JMP END
12  $BLUE
13  O2:=1 T100 O2:=0 ;Set output 2 for 1s
14  JMP END
15  $YELLOW
16  O3:=1 T100 O3:=0 ;Set output 3 for 1s
17  JMP END
18  $PINK
19  O4:=1 T100 O4:=0 ;Set output 4 for 1s
20  $END
21  END
3.18 CASE.SUB - Subroutine distributor

CASE.SUB.N

**CASE.SUB.(Variable)**

**Description:**
The CASE.SUB command is a program branch depending on an integer register. Program distributors can be easily created using this command. PA-CONTROL checks the contents of the integer register and executes a subroutine call in accordance with the value. If the content of the integer register is less than 1 or greater than the number of elements in the names table (in this example: SMALL, MEDIUM, LARGE), a jump to the label defined after ELSE is executed. Otherwise the respective subroutine is called up and processed. Processing is continued with the program line after the ELSE branch (in this example: Line 8 with O12:= 0).

**The following assignments apply:**

<table>
<thead>
<tr>
<th>Value in register</th>
<th>Name in name table</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1st Name</td>
</tr>
<tr>
<td>2</td>
<td>2nd Name</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>nth Name</td>
</tr>
</tbody>
</table>

**NOTE**
The subroutines must exist, otherwise the error message "Subroutine not found" is displayed.

**Subroutines may not call themselves.**

**Command format:**
CASE.SUB.Nn
(Name1)
(Name2)
...
ELSE Label_e

Or with indirect addressing of the register
CASE.SUB.N!n
(Name1)
(Name2)
...
ELSE Label_e
Application:
Program branch in the case of more than two possibilities, e.g. various manufacturing types, variants of traversing frequencies.

Example

1 $ERROR
2 N8:=N13 ;Read value via register
3 CASE.SUB.N8 ;Check contents of N8 and call subroutine if:
4 (SMALL) ;N8=1; process subroutine "SMALL"
5 (LARGE) ;N8=2; process subroutine"LARGE"
6 (MEDIUM) ;N8=3; process subroutine "MEDIUM"
7 ELSE ERROR ;Jump to label "ERROR", if the value of N8 is outside the specified limits (here if N8<1 or N8>3)
8 O12:=0
9 I1.1
10 END

Program: SMALL
1 O1:=1 T100 O1:=0 ;Set output 1 for 1s
2 END

Program: LARGE
1 O2:=1 T100 O2:=0 ;Set output 2 for 1s
2 END

Program: MEDIUM
1 O3:=1 T100 O3:=0 ;Set output 3 for 1s
2 END
3.19 DEC - Loop with conditional jump

**DEC**

**DEC.(Counter) (Label)**

**Description:**

The DEC command is used to create program loops. It can be compared with the Pascal command "Repeat...until" or the C command "Do... While".

The DEC command checks the contents of the integer register. If the contents of the integer register are greater than 0, it is decremented by 1 and a jump to the label executed. Otherwise the next command (next line) is processed.

**NOTE** The DEC command is only applicable for integer registers.

**Command format:**

DEC.Nn Label

**Application:**

For the creation of program loops whose number of runs is stored in an integer register, e.g. process "n" parts, traverse "n" times, etc.
Examples of loops using the DEC command:

**Example 1**

1. \(N2:=8\)  ; Number of loops
2. $BEGINNING
3. O1:=1
4. T100
5. O1:=0
6. T100
7. DEC.N2 BEGINNING  ; All reruns executed? no, then jump to
8. "BEGINNING"
9. END

**Example 2**

THE loop program section is passed through at least once (Repeat...Until, Do...While), i.e. the program section is always processed. Afterwards a check is made (DEC.) as to whether the program section has to be processed once again.

1. \(N3:=10\)
2. $DO_STH
3. O1:=1
4. T100
5. O1:=0
6. T100
7. DEC.N3 DO_STH
8. END

**Example 3**

The loop program section is not run through depending on the value (While), i.e. the register is checked first (DEC.N3), and the program section is then processed depending on the contents of the register.

1. \(N3:=15\)
2. $LOOP
3. DEC.N3 DO_STH
4. JMP DONE
5. $DO_STH
6. O1:=1
7. T100
8. O1:=0
9. T100
10. JMP LOOP
11. $DONE
12. END
3.20 INC - Loop with conditional jump

INC

Description:
The INC command is used to create program loops. It can be compared with the Pascal command Repeat...Until.
The INC command checks the contents of the integer register Nn (example N2). If the contents of the integer register are less than the default value of the integer register or number, the content (N2) is incremented by 1 and a jump to the label is executed. Otherwise processing is continued with the next command (next line).

NOTE The INC command is only applicable for integer registers.

Command format:
INC.Nn.Nn Label
INC.Nn.n Label

Application:
For the creation of program loops whose number of runs is stored as a number or in an integer register, e.g. process "n" parts, traverse "n" times, etc.

Example

1 N3:=5 ;Number of loops equals 5
2 N2:=1 ;Loop counter
3 BEGINNING ;Beginning of loop
4 O1:=1
5 T100
6 O1:=0
7 T100
8 INC.N2.N3 BEGINNING ;Repeat or jump to "BEGINNING", provided N2<N3.
9 END

Or:
8 INC.N2.5 BEGINNING ;Repeat or jump to "BEGINNING", provided N2<5.

3.21 BREAK - Cancel automatic mode

**BREAK**

**Description:**
The BREAK command cancels the automatic mode of PA-CONTROL. PA-CONTROL returns to the main menu.

**Command format:**
BREAK

**Application:**
The BREAK command is used to discontinue the automatic cycle of PAC when e.g. a machine error has occurred in a subroutine. Using the BREAK command avoids returning to the program structure, in order to return to the main program from several subroutine levels and from there to branch to the end of the program.

**NOTE**
If the BREAK command is used in "Program at malfunction", the user will not receive a message via error number "EXXX". PA-CONTROL goes to the initial position. Error analysis cannot be carried out. If a "Program at STOP" has been specified, it will not be processed!

**Example**

```
1 $LOOP
2 I10.1 ;Start
3 SUB GET ;Subroutine for get part
4 SUB PROCESS ;Subroutine for process part
5 SUB DROP ;Subroutine for drop part
6 G21 I11.1 LOOP ;Continue production
7 END
```

**Program:** GET.PNC

```
1 G421.1.100 ERROR ;Error, if "get part" operation not completed in 1 s
2 O1:=1 ;Open gripper
3 I1.0 I2.1 ;gripper open
4 O2:=1 ;Lower gripper
5 I3.0 I4.1 ;Gripper down
6 O1:=0 ;Close gripper
7 I2.0 I1.1 ;Gripper closed
8 O2:=0 ;Lift gripper
9 I4.0 I3.1 ;Gripper up
10 G401.1
11 END
12 $ERROR
13 I12.1 ;ACKNOWLEDGE ERROR FOR GET PART
14 BREAK
```
3.22 Communication with programs in parallel run

3.22.1 RUN – Start of a parallel run

**RUN**

**Description:**
Further tasks (processes) can be started in parallel by means of the command RUN.

During processing of the RUN command, the program type (*.PNC/ *.PNX / *.PAB) is checked and the parallel run started accordingly.

The difference between the run of a PNC/PNX program and that of a PAB program can only be seen if a stop is executed in AUTOMATIC mode.

The following applies for STOP in automatic mode:

- **For the PNC/ PNX run (task):**
  - Axes are stopped
  - Programs are stopped (like SLEEP) and not processed any further

- **For the PAB run (task):**
  - Programs are further processed
  - If a traversing command is being processed, or a traversing command is to be activated, the run stops at this command

The different operating principles of these program runs offer an advantage. If PA-CONTROL is stopped, output to the display, communication via the serial port and/or the flashing of signal lights can be implemented by means of PAB runs.

**NOTE** The program is processed until it is ended by an END command or CANCEL.

**NOTE** A program can only be registered once in the run interpreter. Up to 31 programs (tasks) can be processed in parallel in the process interpreter.

**Command format:**

```
RUN NAME
```

**Application:**
Start of parallel sequencing programs.
### Example 1

**PNC program**

1. M1:=0 ; Reset flag 1
2. M2:=0 ; Reset flag 2
3. RUN FLASHING ; Start PAB program "FLASHING"
4. RUN OPEN_DOOR ; Start programm "OPEN_DOOR"
5. M1.1 ; Wait until flag 1 has been set to logical "1"
6. M2.1 ; Wait until flag 2 has been set to logical "1"
7. O3:=1 ; Set output 1
8. END

**PAB program "OPEN_DOOR.PAB"**

1. O30:=1 ; Set output 31, open door
2. I30.1 ; Door is open
3. O30:=0 ; Reset output 30
4. M2:=1 ; Set flag 2, door open
5. END

**PAB program "FLASHING.PAB"**

1. $B_CYCLE
2. O2:=1 ; Set output 2
3. T20 ; Waiting time 200ms
4. O2:=0 ; Reset output 2
5. M1:=1 ; Set flag 1
6. JMP $B_CYCLE ; Jump to label "$B_CYCLE"
7. END

While the PAB program "OPEN_DOOR" is terminated after the opening and therefore deleted from the parallel run interpreter, the PAB program "FLASHING" runs until it is stopped through another process by the command "CANCEL".
3.22.2 SLEEP - Stopping a parallel run

### SLEEP

**Description:**
Operational programs can be stopped by means of the SLEEP command. The program is placed in a sleep mode by the SLEEP command of PA-CONTROL, i.e. it is no longer processed and remains at the current command. This program can be continued from the current command by means of a RUN command.

**NOTE**
The SLEEP command can only be applied to programs which are in RUN mode.

**Command format:**
```
SLEEP Name
```

**Application:**
Stopping of parallel run programs, e.g. if a safety door is opened.

**Example**

### PNC program

1. `M1:=0` ;Reset flag 1
2. `M2:=0` ;Reset flag 2
3. `RUN OFF1` ;Start program "OFF1"
4. `M1.1` ;Wait until flag 1 has been set to logical "1"
5. `SLEEP OFF1` ;Stop program "OFF1"
6. `M1:=0` ;Reset flag 1
7. `I1.1` ;Wait until input 1 has been set to logical "1"
8. `RUN OFF1` ;Continue program "OFF1"
9. `M2.1` ;Wait until flag 2 has been set to logical "1"
10. `O3:=1` ;Set output 1
11. `END`

### PAB program: "OFF1.PAB"

1. `O1:=1` ;Set output 1
2. `M1:=1` ;Set flag 1
3. `M1.0` ;Wait until flag 0 has been set to logical "1"
4. `T10` ;Waiting time 100ms
5. `O1:=0` ;Reset output 1
6. `M2:=1` ;Set flag 2
7. `END` ;End program "OFF1"
### 3.22.3 CANCEL - Ending a parallel run

**CANCEL**

**Description:**
The CANCEL command terminates operational programs.

**NOTE**
The CANCEL command can be also used for programs that are not in the RUN or SLEEP mode.

The program will be aborted at the command which is currently being processed, i.e. elements which the command is controlling may have to be maintained (flag, outputs, registers).

**NOTE**
If the program addressed in the CANCEL command does not exist in the controller, the controller stops with error E513, Program does not exist.

**Command format:**
CANCEL Name

**Application:**
Immediate termination of operational programs. This creates space in the process interpreter for a new program, e.g. when changing the operating mode of a manufacturing facility.

**Example**

```
1  M1:=0 ;Reset flag 1
2  M2:=0 ;Reset flag 2
3  RUN OFF1 ;Start program "OFF1"
4  I7.1 ;Wait until input 7 is hot
5  CANCEL OFF1 ;End program "OFF1"
6  END
```

**Program: AUS1.PAB**

```
1  $BEGINNING ;Set output 1
2  O1:=1
3  T10 ;Waiting time 100ms
4  O1:=0 ;Reset output 1
5  T10 ;Waiting time 100ms
6  JMP BEGINNING ;Jump to label "BEGINNING"
7  END
```
3.22.4 STOP - Stop automatic run

STOP

Description:
The STOP command stops all axes, irrespective of whether a PNC or a PAB program was started. All processes behave as specified in RUN. This means that PAB programs continue to run.

Command format:
STOP

Application:
The STOP command is used to stop all positioning commands, axes and PNC processes started by RUN. For example, if an application error occurs, the program can be stopped in an elegant way without being aborted.

NOTE The STOP command can only be used in a PAB program.

Example

PAB program: GRIPPER.PAB

1 M10.1 ;Wait for finished flag
2 G421.1.200 Error ;Error if "get part" operation not completed in 2 s
3 O1:=0 O2:=1 ;Open gripper
4 I1.0 I2.1 ;Gripper open
5 O3:=0 O4:=1 ;Lower gripper
6 I3.0 I4.1 ;Gripper down
7 O1:=1 O2:=0 ;Close gripper
8 I1.1 I2.0 ;Gripper closed
9 O3:=1 O4:=0 ;Raise gripper
10 I4.0 I3.1 ;Gripper up
11 G401.1
12 M11:=1
13 END
14 $ERROR
15 STOP ;Activate system function STOP
16 END
3.22.5 START - Start automatic run

START

Description:
The **START** command starts programs and positioning commands which have been stopped and cancels the Stop command. This means that traversing commands which have been stopped are restarted and ended at their specified position.

Command format:
START

Application:
The **START** command restarts axes which have been stopped and continues the processing of PNC/PNX processes which have been stopped (started by RUN).

**NOTE**
The **START** command can only be used in a PAB program.
Example

1 $LOOP
2 I10.1 ;Start
3 RUN LOOK ;Start of a PAB program
4 RUN CONTINUE ;Start of a PAB program
5 SUB PROCESS ;Subroutine for processing the part
6 SUB DROP ;Subroutine for dropping the part
7 G21 I11.1 LOOP ;Continue production
8 END

Program: LOOK.PAB

1 G421.1.100 Error ;Error if "get part" operation not completed in 1 s
2 I1.0 I2.1 ;Gripper open
3 I3.0 I4.1 ;Gripper down
4 I2.0 I1.1 ;Gripper closed
5 I4.0 I3.1 ;Gripper up
6 G401.1
7 END
8 $ERROR ;Activate system function STOP
9 STOP ;Activate system function STOP
10 END

Program: CONTINUE.PAB

1 $LOOP ;Beginning of loop
2 G21 I3.1 Start ;To start if input 3 is hot
3 JMP LOOP ;Go to beginning
4 $Start ;Jump label START
5 Start ;Activate system function START after STOP
6 END
### 3.22.6 CASE.RUN - Starting parallel runs with CASE

**CASE.RUN**

**Description:**

The CASE.RUN command starts parallel runs depending on an integer register. Program distributors can be easily set up using this command.

PA-CONTROL checks the contents of the integer register and starts a parallel run in accordance with the value. If the contents of the integer register are less than 1 or greater than the number of elements in the name table (in this example: PROG-1, PROG-2, PROG-3), a jump to the label defined after ELSE is executed. Otherwise the respective PAB program is called and continued with the program line after the ELSE branch (in this example: Line 8 with O12:= 0).

The following assignments apply:

<table>
<thead>
<tr>
<th>Value in register</th>
<th>Name in the name table</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1st Name</td>
</tr>
<tr>
<td>2</td>
<td>2nd Name</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>n</td>
<td>nth Name</td>
</tr>
</tbody>
</table>

**NOTE**

The programs must exist, otherwise the error message "Program does not exist" is output.

Once a parallel run has been started, it may not be started again. Otherwise the error message "Program already running" is output.

**Command format:**

CASE.RUN.Ni

(Name1)

(Name2)

...

ELSE Label_e

**Application:**

Depending on e.g. different production types, various processing operations can take place in parallel to other operations. These operations can be variably initiated by means of CASE.RUN.
Example

1. G11.0 ; Switch off run display
2. G500.0 ; LC display is current data channel
3. G501 ; Clear display
4. $ERROR ;
5. G503.1.1 ; Position cursor
6. G510.INPUT PROG. NUMBER ; Output of user request
7. G542.24.1.2.N8 ; Input value via the keyboard
8. CASE.RUN.N8 ; Check contents of N8 and start PAB program at:
9. (PROG-1) ; N8=1 : start program PROG-1
10. (PROG-2) ; N8=2: start program PROG-2
11. (PROG-3) ; N8=3: start program PROG-3
12. ELSE ERROR ; Jump to label "ERROR if value of N8 is out of the specified range (in this case if N8<1 or N8>3)
13. O12:=0
14. I1.1
15. END

Program: PROG-1.PAB

1. O1:=1 T100 O1:=0 ; Set output 1 for 1s
2. END

Program: PROG-2.PAB

1. O2:=1 ; Set output 2
2. I1.1 ; Wait for input 1
3. O2:=0 ; Reset output 2
4. END

Program: PROG-3.PAB

1. O3:=1 T100 O3:=0 ; Set output 3 for 1 s
2. END
3.22.7 CASE.SLEEP - Stopping of parallel runs with Case

**CASE.SLEEP**

**Description:**
Parallel sequences can be stopped depending on an integer register by means of the CASE.SLEEP command. Program distributors can be easily set up using this command.

PA-CONTROL checks the contents of the integer register and starts a parallel run in accordance with the value. If the contents of the integer register are less than 1 or greater than the number of elements in the name table (in this example: PROG-1, PROG-2,) a jump to the label defined after ELSE is executed. Otherwise the respective program is called and continued with the program line after the ELSE branch.

**The following assignments apply:**

<table>
<thead>
<tr>
<th>Value in register</th>
<th>Name in the name table</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1st Name</td>
</tr>
<tr>
<td>2</td>
<td>2nd Name</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>N</td>
<td>nth Name</td>
</tr>
</tbody>
</table>

**NOTE**
The programs must exist, otherwise the error message "Program does not exist" is output.

Once a parallel run has been stopped, it may not be stopped again. Otherwise the error message "Program already in sleep mode" is output.

**Command format:**
CASE.SLEEP.Ni
(Name1)
(Name2)
...
ELSE Label_e

**Application:**
Depending on e.g. different production types, various processing operations can take place in parallel to other operations. These operations can be variably stopped by means of CASE.SLEEP.
Example

1. RUN PROG-1 ;Start of the program PROG-1
2. RUN PROG-2 ;Start of the program PROG-2
3. $BEGINNING
   ...
19. G11.0 ;Switch off run display
20. G500.0 ;LC display is current data channel
21. G501 ;Clear display
22. $ERROR ;
23. G503.1.1 ;Position cursor
24. G510.INPUT PROG. NUMBER ;Output of user request
25. G542.24.1.2.N10 ;Input value via the keyboard
26. CASE.SLEEP.N10 ;Checks the contents of N10 and stops the corresponding PAB program at N10:
27. (PROG-1) ;N10=1: stops program PROG-1.PAB
28. (PROG-2) ;N10=2: stops program PROG-2.PAB
29. ELSE ERROR ;Jump to label "ERROR if value of N10 is out of the specified range (in this case if N10<1 and N10>2).
   ...
100. END

Program: PROG-1.PNC

1. $BEGINNING ;Jump label
2. O1:=1 ;Set output 1
3. I1.1 ;Wait until input 1 is hot
4. O1:=0 ;Reset output 1
5. T100 ;Wait 1s
6. JMP BEGINNING ;Go to label BEGINNING
7. END

Program: PROG-2.PAB

1. $BEGINNING ;Jump label
2. O2:=1 ;Set output 2
3. I2.1 ;Wait until input 2 is hot
4. O2:=0 ;Reset output 2
5. T50 ;Wait 0.5s
6. JMP BEGINNING ;Go to label BEGINNING
7. END
3.22.8 CASE.CANCEL - Ending parallel runs with Case

**CASE.CANCEL**

**Description:**

Parallel sequences can be ended depending on an integer register by means of the CASE.CANCEL command.

PA-CONTROL checks the contents of the integer register and ends a parallel run on the basis of the value stored in the integer register.

If the contents of the integer register are less than 1 or greater than the number of elements in the name table (in this example: PROG-1, PROG-2), a jump to the label defined after ELSE is executed. After the CANCEL command has been executed without error, processing is continued in the line which follows the ELSE Label_e.

**NOTE**

The CASE.CANCEL command can also be executed for programs which are not running (see also Section 3.22.3 on page 121).

**NOTE**

If the program addressed in the CASE.CANCEL command does not exist in the controller, the controller stops with error E513, Program does not exist.

The following assignments apply:

<table>
<thead>
<tr>
<th>Value in register</th>
<th>Name in the name table</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1st Name</td>
</tr>
<tr>
<td>2</td>
<td>2nd Name</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>n</td>
<td>nth Name</td>
</tr>
</tbody>
</table>

**Command format:**

CASE.CANCEL.Ni
(Name1)
(Name2)
... ELSE Label_e

**Application:**

Depending on e.g. different production types, various processing operations can arise which run in parallel or are dormant. The undesired parallel processes can be ended by means of CASE.CANCEL.
Example:

1. RUN PROG-1  ;Start of the program PROG-1
2. RUN PROG-2  ;Start of the program PROG-2
3. $BEGINNING
   ...
19. G11.0  ;Switch off run display
20. G500.0  ;LC display is current data channel
21. G501  ;Clear display
22. $ERROR  ;
23. G503.1.1  ;Position cursor
24. G510.INPUT PROG. NUMBER  ;Output of user request
25. G542.24.1.2.N20  ;Input value via the keyboard
26. CASE.CANCEL.N20  ;Checks the contents of N20 and stops the corresponding PAB program at:
   27. (PROG-1)  ;N20=1: stops program PROG-1.PAB
   28. (PROG-2)  ;N20=2: stops program PROG-2.PAB
   29. ELSE ERROR  ;Jump to label “ERROR if value of N10 is out of the specified range (in this case if N10<1 and N10>2).
   ...
100. END

Program: PROG-1.PNC

1. $BEGINNING  ;Jump label
2. O1:=1  ;Set output 1
3. I1.1  ;Wait until input 1 is hot
4. O1:=0  ;Reset output 1
5. I2.1  ;Wait until input 2 is hot
6. JMP BEGINNING  ;Go to label BEGINNING
7. END

Program: PROG-2.PAB

1. $BEGINNING  ;Jump label
2. O2:=1  ;Set output 2
3. T50  ;Wait 0.5s
4. O2:=0  ;Reset output 2
5. T50  ;Wait 0.5s
6. JMP BEGINNING  ;Go to label BEGINNING
7. END
3.22.9 PROGSTAT - Get program status

Ni:=PROGSTAT

Ni:=PROGSTAT.xxx

Description:

The command checks whether the program has been activated as a task (by means of RUN or CASE RUN). If the program cannot be found as a current task in the run interpreter, a check is made to ascertain that the program exists in PA-CONTROL. After the check, the following information is stored in register Ni:

<table>
<thead>
<tr>
<th>Contents of Ni</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Program does not exist in PA-CONTROL</td>
</tr>
<tr>
<td>1</td>
<td>Program exists in PA-CONTROL, but is not an active TASK</td>
</tr>
<tr>
<td>2</td>
<td>Exists as a task, is being processed</td>
</tr>
<tr>
<td>3</td>
<td>Exists as a task, is not being processed (SLEEP)</td>
</tr>
</tbody>
</table>

Command format:

Ni:= PROGSTAT.Name
N!i:=PROGSTAT.Name
Ni:= PROGSTAT.S0
Ni:= PROGSTAT.Sn
Ni:= PROGSTAT.Nn
Ni:= PROGSTAT.Nn.i

Example:

1   N10:=PROGSTAT.DOOR_OPE
   N  Is the program "DOOR_OPEN" active or does it exist in PA-CONTROL? Storage of the result in integer register N10.
2   $A
3   N5:=128  Load the number 128 to integer register N5
... 10  PROGSTAT.S0  Inquire about the status of the program whose name is in string S0
... 20  PROGSTAT.S1  Inquire about the status of the program whose name is in string S1
... 30  PROGSTAT.N10  Inquire about the status of the program whose name is in integer register N10
... 40  PROGSTAT.N5.7  Inquire about the status of program 0000128. The command PROGSTAT adds four leading zeros to the three-figure number from integer register N5
41   END
3.23 A1 - Positioning of the axes

**A1:**=(Operator)

**Description:**
The positioning command allows the axes to be traversed in the positive and negative direction, regarded from the zero point. The position is the absolute value in the absolute dimension system (G90) and the relative traverse path in the incremental dimension system (G91).

**NOTE** The end position of the movement is dependent upon the programmed system of units, either the absolute dimension (G90) or the incremental dimension (G91).

**Command format:**
A1:=nnnn
A1:=Rn

**Application:**
Positioning of axes

**Example**

1. G25.A1 ; Carry out approach to reference point with A1 axis
2. G90.A1 ; Absolute dimension system
3. A1:=200 ; Axis A1 200 in positive direction
4. A1:=198 ; Axis A1 by 2 in negative direction to absolute position 198
5. R3:=678 ; Axis A1 in positive direction to absolute position 678
6. A1:=R3 ; Incremental dimension system
7. G91.A1
8. R3:=-200 ; Traverse axis A1 by -200 in negative direction
9. A1:=-10 ; Traverse axis A1 by 10 in negative direction
10. A1:=R3
11. R15:=3
12. A1:=R!15 ; Traverse axis A1 by -200 in negative direction
13. END
3.24 Get axis position

\[ Rn:=An \]
\[ Nn:=An \]

**Description:**
The current axis position is transferred to an N or R register.
The position is available in the R register as per the set unit of measurement. The step counter is transferred to an N register.

**Command format:**
R1:=An
N1:=An

**Application:**
Storage of axis positions e.g. in the teach program.
Display of the axis position and step monitoring in the program.

**Example**

1. \[ N226:=0.01 \]; Maximum deviation of the step counter to the encoder position
2. \$LOOP ; Approach position 100 at reduced torque
3. \[ R222:=A1 \]; Get axis position
4. \[ R223:=ENC1 \]; Get encoder position
5. \[ R25:=R222-R223 \]; Calculate difference
6. \[ M221:=R225>R226 \]; Compare
7. \[ G21 M221.0 LOOP \]
8. \[ M220:=0 \]; STOP e.g. of an axis started with G123
9. END
3.25 Request for the set or current speed

Command format:
\[ R_n:=FAn \]
\[ R_n:=FB \]
\[ R_n:=FBA_n \]
\[ R_n:=SR \]

3.25.1 Set traversing speed of an axis

\[ R_n:=FAn \]

Command format:

This command supplies the last set speed of the axis. It does not provide information as to whether the axis is currently traversing at this speed (axis at standstill, axis accelerating, axis traversing or axis braking).

3.25.2 Get current path speed

\[ R_n:=FB \]

Command format:
\[ R_n:=FB \]

Description:
The current path speed can be retrieved and stored in a real number register in parallel to an interpolation in progress from another task (parallel run) by means of this command.

This information enables the application to control components of the plant on the basis of the current path speed.

<table>
<thead>
<tr>
<th>Action</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interpolation activated and traversing is underway</td>
<td>Current path speed</td>
</tr>
<tr>
<td>Acceleration or braking during interpolation</td>
<td>Current path speed in the ramp</td>
</tr>
<tr>
<td>Interpolation was stopped (STOP PA-CONTROL)</td>
<td>0</td>
</tr>
<tr>
<td>No interpolation active</td>
<td>-1</td>
</tr>
</tbody>
</table>
Example

...
11    RUN GET_CURRENT_FB       Start PAB program GET_CURRENT_FB
12    R1:=1000
13    R2:=5000
14    R3:=3000
15    R4:=1000
16    $A
17    I8.1
18    G100.B.10000
19    FB:=R1
20    G01 A1:=1000 A2:=2000
21    FB:=R2
22    G01 A1:=2000 A2:=4000
23    FB:=R3
24    G01 A1:=3000 A2:=6000
25    FB:=R4
26    G01 A1:=4000 A2:=8000
27    T100
28    A1:=0  A2:=0
29    T100
30    JMP A
31    END

Program GET_CURRENT_FB

1     $A
2     R5:=FB                   Get the current path speed and store the value in
3     JMP A
4     END

real number register 5.

3.25.3 Get the current speed of an axis during an interpolation

Rn:=FBAn

Command format:
Rn:=FBAn

Description:
The command supplies the current speed of the selected axis during an interpolation. The
speed is returned according to the path speed and with the proportional ratio of the axis.
3.25.4 Get the set path speed of an axis

\textbf{Rn:=Fan}

\textbf{Command format:}

\texttt{Rn:=Fan}

This command supplies the last set speed of the axis. It does not provide information as to whether the axis is currently traversing at this speed (axis at standstill, axis accelerating, axis traversing or axis braking).
### 3.26 Get following error

**Ni:= PEAn**

**Ri:= PEAn**

**Description:**

The current following error of axes which are driven via an LV-servoTEC or a PA-CONTROL MP can be transferred to an N or R register. Following errors of a servo-axis are transferred from the LV-servoTEC to an N or R register. Following errors of the PA-CONTROL MP are transferred from the internal counter to an N or R register.

In this connection, there are two fundamentally different forms:

- Following errors of the axis n in SI units, i.e. in increments or steps, are transferred to an integer register.
- Following errors of the axis n in display devices are transferred to real number registers.

**Command format:**

**Ni:= PEAn**

**Ri:= PEAn**

**Application:**

Recording of the “following error” in the case of screwing or force limitation

**NOTE**

Approximately 8-10ms is required for execution of the command. This means that a following error can only be acquired at a data rate of approx. 10ms.

**Example**

```
1  N26:=150 ; Max. following error
2  M22:=0 ; Delete cancel flag
3  $LOOP
4  N25:=PEA1 ; Get following error
5  N25:=ABS.N25 ; Generate absolute value unsigned
6  M21:=N25>N26 ; Difference too large
7  G21 M21.0 LOOP
8  M22:=1 ; Cancel
9  END
```
3.27 ENC - Transfer encoder position

\[
\begin{align*}
R_n &= \text{ENC}_n \\
N_i &= \text{ENC}_n
\end{align*}
\]

(\text{ENCAxisNo})

Description:
The encoder position can be transferred to an N or R register.

**NOTE**
- If the encoder value is stored in an R register, the rotary encoder position, which is comparable with the axis position, is stored.
- If the encoder value is stored in an N register, the rotary encoder increments are stored adjusted to the steps of the stepping motor.

**Command format:**
\[
\begin{align*}
R_1 &= \text{ENC}_n \\
N_1 &= \text{ENC}_n
\end{align*}
\]

Application:
Step deviations can be detected in the program, e.g. for screwing, for which a specified current can be assigned to a specified torque. When the torque is reached, the motor stops, because the previously set current has been exceeded. The difference between the encoder position and the step counter verifies that the screw has been correctly tightened.

\(^1\) = PA-CONTROL MP
Example 1

1  R226:=0.01 ;Maximum deviation of the axis position to the encoder position
2  $LOOP
3  R222:=A1 ;Get axis position
4  R223:=ENC1 ;Get encoder position
5  R224:=R222-R223 ;Calculate difference
6  R225:=ABS.R224 ;Calculate sum
7  M221:=R226>R225 ;Compare
8  G21 M221.0 LOOP
9  M220:=1 ; For triggering a stop e.g.
10  END

Example 2

1  N226:=32 ;Maximum deviation of the step counter to the encoder position counter. 32 corresponds to the maximum following error of a 2 phase stepping motor of 2 full steps (2 full steps with 16x resolution → 32).
2  $LOOP
3  N222:=A1 ;Get step counter
4  N223:=ENC1 ;Get encoder position counter
5  N224:=N222-N223 ;Calculate difference
6  N225:=ABS.N224 ;Calculate sum
7  M221:=N225>N226 ;Compare
8  G21 N221.0 LOOP
9  M220:=0 ; For triggering a stop e.g.
10  END
3.28 Wait for position from the SSI interface

SSIni.Rm
SSIni.i.Nm

(SSIni.i.<Register>)

Description:
Waits until the position or the count value of the SSI module is larger or smaller than the value of the register (real number or integer register).
- Wait until larger : i=1
- Wait until smaller : i=0

Command format:
SSIni.i.Rm
SSIni.i.Nm

Application:
Is used for setting or resetting outputs in the case of high-speed servoTEC axes, in which the repeat accuracy must be less than 10ms.

Example

1   R12:=123.45
2   N33:=446
...

n   SSIni.1.R12 ; Waits until the position of the SSI module is higher than the contents of R2, i.e. higher than 123.45
...

m   SSIni.0.N33 ; Waits until the counter value of the SSI module is less than the contents of N33s, i.e. less than 446
3.29 FAn - Traversing speed

\( F(Axis) := (Speed) \)

**Description:**

The command \( F \) prompts the controller to traverse at the traversing speed specified in the operator in the further program run. All axes which are actuated by operand \( F \) can be traversed at various speeds.

The axis parameters specified in the parameter list can also be loaded via registers.

FB is used to set the path speed during interpolation.

**NOTE**

If a speed \( F \) has not been programmed in the program run, all traversing commands are executed at the traversing speed defined in the parameter field.

A speed default entry which is higher than the one specified in the parameter field results in the error message "Value too high" (E518).

An error message is not output if the speed value is too low. The lowest possible speed is set instead.

**Command format for changing the speed of an axis:**

FAn:=200
FAn:=R4

**Command format for changing the speed of a path:**

FBAAn:=200;
FBAAn:=R4

**Application:**

Changing the speeds for traversing and moving axes during the program run.
3.29.1 Particular features of the PA-CONTROL MP / LV-servoTEC:

In contrast to the other variants of the controller, the speed can be changed in the PA-CONTROL MP and LV-servoTEC while positioning is in progress.

![Speed profile](image)

**Example: only for PA-CONTROL MP and LV-servoTEC**

1. G90.A0 ;Positioning ensues in the absolute dimension system
2. G25.A1 ;Approach to reference point
3. G100.A1.100 ;Specify acceleration of the A1 axis
4. G210.A0 ;Process commands beyond line limit
5. FA1:=1000 ;Traversing speed A1 axis $v_1$: 1000AE/s
6. A1:=1500 ;Positioning of the A1 axis
7. G230.1.A1.200 ;Wait until the current absolute position of the A1 axis is higher than 200
8. FA1:=500 ;Traversing speed A1 axis $v_2$: 500AE/s
10. FA1:=150 ;Traversing speed A1 axis $v_3$: 150AE/s
12. FA1:=1200 ;Traversing speed A1 axis $v_4$: 1200AE/s
13. G213.A0
14. END
3.30 G25 - Approach to reference point

G25.(Axis)

Description:
After the command G25.An has been started, PA-CONTROL carries out an approach to reference point with the respective axis (The variable "n" stands for the relevant axis here). The position reached after the approach to reference point is effective as a zero point for the axis (see also Section 5.2.6 in the chapter on "Parameters").

NOTE
The prerequisite for the positioning of an axis is the enabling by setting a system flag for traversing this axis. There are two possibilities in PA-CONTROL: the approach to reference point of an axis or setting the enabling bit for all axes without an approach to reference point (G25.A0).

The speed is transferred from the parameters for the approach to reference point (reference speed).

The parameter settings for the reference speed and the acceleration must be adjusted to the excursion (distance between the reference switch and the mechanical end stop).

Command format 1:
G25.An

The addressed axis, e.g G25.A7 for axis 7, moves towards the negative limit switch until the latter opens, changes direction and clears the switch at the start/stop frequency. If the enabling bit has been set, the position counter is also set to zero.

Command format 2:
G25.A0

If this command format is used, the position of the connected axes does not change. The axes remain in their position, and the enable flag is set.

Application:
After every restart, e.g. after a power failure or the switch-off of a PA-CONTROL, an approach to reference point is normally required. This is the only way to guarantee that the controller executes its the program in the defined manner. Command format 1 is provided for this task.

It may be appropriate to dispense with the approach to reference point in certain situations, e.g. the manual traversing of an axis to check and measure a path. Command format 2 is provided for this task.
Example 1

1  G25.A1  ;Approach to reference point is carried out for the A1 axis
2  G25.A2  ;Approach to reference point is carried out for the A2 axis
3  G25.A0  ;Setting of the enabling bit for all other axes
4  END

Example 2

1  G90
2  G21 I5.0 REFEREN  ;Reference condition for A2 axis provided?
3  G25.A0 G91  ;Set enabling bit and specify relative traversing
4  A1:=100  ;Create reference condition for A2 axis!
5  G90  ;Absolute dimension system
6  $REFEREN
7  G25.A2  ;Approach to reference point axis 2
8  G25.A1  ;Approach to reference point axis 1
9  END
3.31 G26 - Set position to zero / set to position value

**G26**

\[ G26.(\text{Axis}) \]
\[ G26.(\text{Axis}).\text{konst} \]
\[ G26.(\text{Axis}).\text{Rn} \]

**Description:**

The absolute position counter of each axis can be set to zero by means of the function `G26.An` and to the position value by means of the functions `G26.An.konst` and `G26.An.Rn`. The variable `An` of the operand specifies the selected axis. No range check takes place. The axis can therefore be set to a position outside the traversing range limits.

**NOTE**

The software limit switches which were specified by the range in the parameter level are transferred unchanged. If this command is repeated, positioning can be carried out practically infinitely in one direction, since no counter overflow and no overtravel of the range limits is possible.

**Important:** **Counter overflow is possible in servo-axes which are driven via the LV servoTEC!**

The remainders are deleted in the case of gear factors with decimal places not equal to 0, i.e. they cannot be considered during subsequent positioning.

**Command format:**

G26.An

**Application:**

An appropriate application case for this PAC command is for continuous movements, e.g. use of the DT140 turntable in only one direction of rotation.

**Example 1**

1. `G26.A1` ;Set absolute position of the A1 axis to zero
2. `G26.A2` ;Set absolute position of the A2 axis to zero
3. `END`

**Command format:**

G26.An.konst

G26.An.Rn

**Application:**

An appropriate application case for this PAC command is the specification of the axis position e.g. on the basis of stored or transferred values.

**Example 2**

1. `R113:=342.5`
4. `END`
3.32 G29 - Set position to dimension

G29

G29.(Axis)

Description:
The position counter of each axis can be set to a new value by means of the function G29. The operator specifies the selected axis. The software switches are also changed accordingly.

Command format:
- G29.Ai.constant  Sets the current axis position to the value of the constant
- G29.Ai.Rn        Sets the current axis position to the value of the R register
- G29.Ai Rn.Rm     Sets the axis position so that the axis is positioned on the value of the R register Rn for the position Rm.

NOTE
The G29 commands of the versions prior to version 4.71R are rejected completely. The axis position after the approach to reference point G25, G26) is used as a reference.

Example
Machining of three parts

Fig. 5: Setting the dimension
Program: START.PNC

2  R1:=100  ; A1 position of part 1
3  R2:=200  ; A1 position of part 2
4  R3:=300  ; A1 position of part 3
5  R4:=0
6  ;
7  $LOOP
8  G29.A1.R4.R1 ; Set axis position for machining part 1
9  SUB MACHINE_PART
11 SUB MACHINE_PART
13 SUB MACHINE_PART
14 G29.A1.R4.R4 ; Set axis position for traverse to initial position
15 A1:=0 A2:=0
16 ...
17 JMP LOOP
18 END
19

Program: MACHINE_PART

1  A1:=0 A2:=100 ; Approach start position for machining part
2  ...
3  ...
4  END
3.33 G90 - Absolute dimension system

G90.(Axis)

Description:
After the G90.An command has been invoked, all subsequent positioning movements ensue in the absolute dimension system until the incremental dimension system is switched over to by means of the command G91.An. The absolute dimension method is the standard dimension system.

Command format:
G90.An
G90.A0

Application:
For dimensions which are all measured from a zero point.

Example

1  G25.A1 ;Approach to reference point
2  G90.A1 ;Subsequent positioning commands are executed for axis 1 in the absolute dimension system
3  A1:=10 ;Position 10 is approached with the A1 axis, path = 10 mm [Sketch: 1.]
4  A1:=450 ;Position 450 is approached with the A1 axis, path = 440 mm [Sketch: 2.]
5  A1:=30 ;Position 30 is approached with the A1 axis, path = -420 mm [Sketch: 3.]
6  G90.A0 ;Subsequent positioning commands are executed for all axes in the absolute dimension system
7  END

Fig. 6: Absolute dimension system
3.34 G91 - Incremental dimension system

**G91**(Axis)

**Description:**
After the **G91.An** command has been invoked, all subsequent positioning movements ensue in the incremental dimension system until the absolute dimension system is switched over to by means of the command **G90.An**.

**Command format:**
G91.An
G91.A0

**Application:**
For dimensions which are not referenced to a zero point.
For distances which successively occur several times and can be repeated by means of a program loop (DEC.Ni).

**Example**

1. G25.A1 ;Approach to reference point
2. G91.A1 ;Subsequent positioning commands are executed for the A1 axis in the incremental dimension system
3. A1:=10 ;Position 10 is approached with the A1 axis, path = 10 mm [Sketch: 1.]
4. A1:=450 ;Position 460 is approached with the A1 axis, path = 450 mm [Sketch: 2.]
5. A1:=30 ;Position 490 is approached with the A1 axis, path = 30 mm [Sketch: 3.]
6. G90.A0 ;Subsequent positioning commands are executed for all axes in the incremental dimension system
7. END

![Diagram: Incremental dimension system](image)

**Fig. 7: Incremental dimension system**
3.35 G100 - Specifying the acceleration

G100

**G100.(Axis).(Value)**

**Description:**

"An" stands for the designation of the axis and can assume the name of all existing axes in multi-axis modules (number of axes varies depending on the equipment).

G100.B is used to set the path acceleration of the interpolation. The path speed **FB** must be set beforehand.

**NOTE**

If the value is higher or lower than the limits, the error message "Value too high" or "Value too low" is output.

**Command format:**

G100.An.15

G100.An.R7

**Application:**

Deviating from the values for acceleration stored in the parameter values, a mode of operation tailored to the situation can be achieved by means of the command G100. The specification is effective until the next time the command G100 is used or until automatic mode is exited. If the command G100 is not used, the stored parameter values are effective.

**Example**

2. R7:=20
3. G91.A0 ; Incremental dimension system
4. FA1:=250
5. G100.A1.15 ; Specification of the acceleration for the A1 axis
6. A1:=250 ; Travel command for the A1 axis
7. G100.A2.R7 ; Specification of the acceleration for the A2 axis in accordance with the contents of real number register 7
8. A2:=500 ; Travel command for the A1 axis
9. END
3.36 G101 - Changing the motor current

**G101**

*(G100.Axis.Value)*

**Description:**

The motor current can be changed in percentage terms in relation to the nominal current set in the parameters. The maximum torque of the motor is almost linear in the relation to the motor current.

<table>
<thead>
<tr>
<th>Device</th>
<th>Percental change compared to the parameter value [%]</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA-CONTROL EP</td>
<td>10 to 100</td>
</tr>
<tr>
<td>PA-CONTROL MP</td>
<td>10 to 300</td>
</tr>
<tr>
<td>LV-servoTEC</td>
<td>1 to 300</td>
</tr>
</tbody>
</table>

**Command format:**

G101.An.nnn

G101.An.Nn

A compliance check for the specified limits, which are located directly in the command or can be taken from an N register, is carried out before execution of the command.

**Application:**

- Screwing with a stepping motor or servomotor with a defined torque.
- Force limitation of spindle axes or motor-driven presses.

**NOTE**

The setting for electric current is only active during movement. At standstill, the motor on the PA-CONTROL EP or PA-CONTROL MP is under the "Phase current at standstill" set in the motor parameters.

**Example 1**

1. G101.A1.50 ;50% of the nominal current
2. A1:=100 ;Approach position 100 at reduced torque
3. END

**Example 2**

1. N1:=50 ;50% of the nominal current
2. G210.A1 ;Does not activate the start positioning mode for axis 1 until the axis has arrived at the stop position.
3. A1:=100 ;Approach position 100
4. G230.1.A1.50 ;Wait until the position >=50 has been reached
5. G101.A1.N1 ;Set the motor current according to N1
6. G213.A1 ;Wait until axes are in position and activate the standard positioning mode
7. END

1 only for PA-CONTROL MP, LV-servoTEC
3.37 G123 - Traverse as long as condition fulfilled

**G123**

*G123.(Axis) (Condition)*

**Description:**

In the function **G123**, the next traversing command of the respective axis is executed until the condition \( n.m \) of the operand is effective. If the condition is not fulfilled, the traversing command is aborted and the next command executed. **G123** only acts on the axis defined in the command. If the condition is not fulfilled on starting the axis, the positioning is not started. The program is immediately continued with the next command.

**NOTE**

The current absolute position is considered for subsequent positioning in the case of aborted traversing commands. The system flag (SM) is then set for the relevant axis.

Only one G123 can be active for each axis. Logic operations of several conditions are possible via the logic operation commands.

The following assignment applies:

SM11: Axis 1 (SM = system flag)

SM12: Axis 2

......

SM26: Axis 16

**Command format:**

G123.An I3.1
G123.An M3.1
G123.An M!5.1
G123.An O4.1

**Application:**

A workpiece is traversed until it is detected. On detection, a sensor or similar transmits a signal back to PA-CONTROL, so that the program can be continued.
Example

1. G25.A1 G90.A1 ;Carry out approach to reference point with A1 axis and switch over to absolute dimension system
2. R2:=1000 ;Load speed to register R2
3. FA1:=R2 ;Select slow speed
4. G123.A1 I3.0 ;The following traversing command is effective as long as the sensor of input 3 is cold
5. A1:=100 ;Traversing command is executed
6. G21SM11.0 NO_PART ;No workpiece present → the condition (system flag of axis 1, SM11, not set) is fulfilled → a conditional jump to the label NO_PART follows.

If a workpiece is present → condition not fulfilled → program is continued in the next line.

7. R1:=A1 ;Real number register is loaded (value 100 corresponds to e.g. the edge of the workpiece)
8. R1:= R1+10 ;Real number register is increased by 10 (value 110 corresponds to the point of the workpiece at which the drill hole is to be set – e.g. centre of the workpiece)
9. A1:=R1 ;Traverse e.g. to the centre of the workpiece
10. SUB DRILLHOLE ;Jump to subroutine/execute
11. $;NO_PART ;Label NO_PART
12. END

Programm: DRILLHOLE

1. O10:=1 T10 ;Switch on drill machine
2. O16:=1 ;Switch on pneumatic feed
3. I15.0 I16.1 ;Stop position reached
4. T50 ;Wait 0.5 seconds
5. O16:=0 O15:=1 ;Pneumatic feed off, backward movement
6. I16.0 I15.1 Idle position reached again
7. O15:=0 Switch off backward movement
8. O16:=0 Switch off the drill machine
9. END

Explanation of the program example:

In the example, the workpiece is traversed until a sensor detects that it is present. If the workpiece is present, the sensor switches on and the program is continued in the next line. If no workpiece is detected, a jump to the label $NO_PART takes place.

The system flag (SM) is only set if traversing commands are aborted.
3.38 Write ServoTEC-Parameters

**PARAMETER.ST.Ai.**

*Parameter. <Axis type>.<Axis>.<Parameter>:=<Value>*

**Description:**
Specific parameters can be sent to a specified LV-servoTEC by means of this command.

**NOTE**
Please refer to the document "DE_EN_servoTEC_ascii20.chm" or the help function of the program "DRIVE.EXE" for a list of the possible parameters including their meaning.

**Command format:**
Parameter.St.Ai.Parameter name:=Constant
Parameter.St.Ai.Parameter name:=Nn
Parameter.St.Ai.Parameter name:=Rn

**NOTE**
The use of an integer or real number register depends on the type of parameter.

**Programm: start_send.pnc**

1   $A
2   I9.1
3   N2:=1
4   R3:=0.5
5   PARAMETER.ST.A1.DIS:=N2 ; Send parameter DIS to servoTEC axis 1 from integer register N2
6   PARAMETER.ST.A1.IPEAK:=R3 ; Send parameter IPEAK to servoTEC axis 1 from real number register R3
7   PARAMETER.ST.A1.O1:=1 ; Send parameter O1 to servoTEC axis 1, the definition ensues with the constant "1"
8   T200
9   I10.1
10  N2:=0
11  R3:=1.5
12  PARAMETER.ST.A1.EN:=N2 ; Send parameter EN to servoTEC axis 1 from integer register N2
13  PARAMETER.ST.A1.IPEAK:=R3 ; Send parameter IPEAK to servoTEC axis 1 from real number register R3
14  PARAMETER.ST.A1.O1:=0 ; Send parameter O1 to servoTEC axis 1, the definition ensues with the constant "0"
15  T200
16  JMP A
17  END
3.39 Reading ServoTEC parameters

Ri:=PARAMETER.ST.Ai

<Register>:=Parameter <Axis type>.<Axis>.<Parameter>

Description:
Specific parameters can be retrieved from a specified LV-servoTEC by means of this command.

NOTE Please refer to the document "DE_EN_servoTEC_ascii20.chm" or the help function of the program "DRIVE.EXE" for a list of the possible parameters including their meaning.

Command format:
Ni:=Parameter.St.An.Parameter name
Ri:=Parameter.St.An.Parameter name

NOTE The use of an integer or real number register depends on the type of parameter.

Program: start_get.pnc

```
1  $A
2  R1:=PARAMETER.ST.A1.PFB ; Get parameter PFB from servoTEC axis 1 and enter it in real number register R1
3  N2:=PARAMETER.ST.A1.IN1 ; Get parameter IN1 from servoTEC axis 1 and enter it in integer register N2
4  N3:=PARAMETER.ST.A1.ACC ; Get parameter ACC from servoTEC axis 1 and enter it in integer register N3
5  R2:=PARAMETER.ST.A1.IPEAK ; Get parameter IPEAK from servoTEC axis 1 and enter it in real number register R2
6  T10
7  JMP A
8  END
```
3.40 Measuring mode for axis

Description:

The axis of a PA-CONTROL can be switched to the measuring mode. In this connection, the brake is activated (option), the motor current set to 0 (only PA-CONTROL MP) and the monitoring of the axis switched off.

The evaluation of the rotary encoder remains active, however, so that the position of the axis can be transferred from the rotary encoder position on exit of the measuring mode.

An approach to reference point is therefore not absolutely necessary.

---

CAUTION

The rotary encoder position can only be transferred in the PA-CONTROL MP. A vertical axis could slip down if it is switched over to the measuring mode. There is then a danger of collision.

---

Application:

Opening of a zone of protection for set-up operations or correction of faults.

NOTE

Only for PA-CONTROL MP, LV-servoTEC
### 3.40.1 G140 – Activation of the measuring mode

**Description:**
The G140 command activates the measuring mode.

**Actions during activation:**
- Brake is activated (option)
- Current reset in the output stage is activated (only possible with PA-CONTROL-MP)
- Monitoring of rotary encoder and standby of the power section is deactivated

**Result:**
- Output stage can be switched off (work safety)
- Axis can be moved manually
- The axis can no longer be traversed by the controller (positioning commands are inhibited and generate an error message)

**Command format:**

```
G140.An
```

**Example**

see next command

**NOTE** only for PA-CONTROL MP, LV-servoTEC
3.40.2 G141 - Deactivation of the measuring mode

Description:
The command G141 deactivates the measuring mode.

Actions for PA-CONTROL MP:
- Motor is under voltage again
- Brake is deactivated (option)
- Synchronization to rotary encoder is executed (option)
- Axis position is transferred from the rotary encoder position (option)
- Monitoring of rotary encoder and standby of the power section is activated

Actions for LV-servoTEC:
- Enable control unit
- Transfer values for rotary encoder position

Result:
The axis can be traversed by the controller again.

NOTE After exiting the measuring mode and synchronization to the rotary encoder, the axis will traverse by a maximum of 4 motor full steps in the positive direction.

Command format:
G141.An

Example

1  G25.A1 ;Carry out approach to reference point with A1 axis
2  A1:=100 ;Go to position 100
3  I5.1 ;Wait until set-up mode active, protective circuit bridged
4  G140.A1 ;Switch over A1 axis to measuring mode, axis can be moved manually
5  I5.0 ;Protective circuit active again, setup mode is disabled
6  G141.A1 ;Axis A1 executes actions as described above
7  A1:=1000 ;Go to position 1000 with the A1 axis
8  END

NOTE Only for-CONTROL MP, LV-servoTEC
3.41 Limit switch monitoring

3.41.1 G142 Limit switch monitoring "OFF" 

G142

G142.(Axis)

Description:
After the G142.A0 or G142.An command has been invoked, the limit switch monitoring is
switched off for one axis (all axes), until the limit switch monitoring for one axis (all axes) is
switched on again by the G143.A0 or G143.An command.

Command format:
G142.A0
G142.An

Application:
The command is of particular importance for round axes, which are often positioned beyond
the 360° limit.

Example

1  G25.A1 ;Approach to reference point
2  G91.A1 ;The following positioning commands are executed for axis 1 in the incremental dimension system
3  G142.A1 ;Limit switch monitoring for axis 1 OFF
4  A1:=10 ;Position 10 is approached with the A1 axis
5  A1:=450 ;Position 460 is approached with the A1 axis
6  A1:=30 ;Position 490 is approached with the A1 axis
7  G90.A1 ;The following positioning commands are executed for axis 1 in the absolute dimension system
8  G143.A1 ;Limit switch monitoring for axis 1 ON
9  END

¹ not for LV-servoTEC
3.41.2 G143 - Limit switch monitoring "ON" \(^1\)

**G143 (Axis)**

**Description:**
After the G142.A0 or G142.An command has been invoked, the limit switch monitoring is switched off for one axis (all axes), until the limit switch monitoring for one axis (all axes) is switched on again by the G143.A0 or G143.An command.

**Command format:**
- G143.A0
- G143.An

**Application:**
The command is of particular importance for round axes, which are often positioned beyond the 360° limit.

### Example

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>G25.A1 ;Approach to reference point</td>
</tr>
<tr>
<td>2</td>
<td>G91.A1 ;The following positioning commands are executed for axis 1 in the incremental dimension system</td>
</tr>
<tr>
<td>3</td>
<td>G142.A1 ;Limit switch monitoring for axis 1 OFF</td>
</tr>
<tr>
<td>4</td>
<td>A1:=10 ;Position 10 is approached with the A1 axis</td>
</tr>
<tr>
<td>5</td>
<td>A1:=450 ;Position 460 is approached with the A1 axis</td>
</tr>
<tr>
<td>6</td>
<td>A1:=30 ;Position 490 is approached with the A1 axis</td>
</tr>
<tr>
<td>7</td>
<td>G90.A1 ;The following positioning commands are executed for axis 1 in the absolute dimension system</td>
</tr>
<tr>
<td>8</td>
<td>G143.A1 ;Limit switch monitoring for axis 1 ON</td>
</tr>
<tr>
<td>9</td>
<td>END</td>
</tr>
</tbody>
</table>

\(^1\) not for LV-servoTEC
3.42 G 150 - Traverse segment with Start-Stop

G150

Description:

The G150 command provides the programmer with the possibility of initiating the braking operation earlier in the next positioning of the relevant axis and to position the segment defined by G150 at the Start-Stop speed. The transition from the braking ramp to the Start-Stop speed (segment) is continuous.

NOTE The G150 command cannot be used for a servoTEC axis. The syntax check checks the axis type and outputs an appropriate message.

NOTE The number after the axis name in the G150 command is a segment and is interpreted unsigned and non-directionally. The set gear factor is considered. The command is only considered for the next positioning of this axis. If the segment to be traversed for the next positioning is less than the segment defined in the G150 command, the G150 command is ignored and the positioning proceeds as usual.

Command format:

G150.An.200
G150.An.Rn

Application:

For positioning operations, in which the last segment is positioned in the start stop mode.

Example: Jointing operation: faster reaction to G123 commands.

---

¹ Not for PAC-servoTEC
### Example

1. G25.A1 ; Carry out approach to reference point with A1 axis
2. G91.A1 ; Switch over to incremental dimension system
3. G150.A1.200 ; Traverse the last 200 increments with the A1 axis at the Start-Stop speed in the next positioning
4. A1:=1000 ; Position 1000 increments with the A1 axis. The braking operation is initiated in such a way that the start-stop speed is reached 200 increments before the target position. The last 200 increments are then positioned at the start-stop speed

#### Fig. 8: Speed progression when using G150

![Speed Progression Diagram](image-url)
3.43 Linear interpolation

3.43.1 G01 – Linear interpolation with 4 from 16 axes

G01

_G01 A1:=(POS) A2:=(POS)........

Description:
In linear interpolation, traversing to a defined target position is executed on a straight line at path speed. Axially parallel movements and movements traversing at any angle can be executed. The path speed is set before G01 using the FBn command. Several interpolation commands defined in succession will be processed without stopping. The path speed can be altered during a complex run without stopping.

The transition to the new path speed is carried out at the preset path acceleration. It is set by means of the command G100.B.n and cannot be altered within a contour.

NOTE
This command is only possible in combination with a PA-CONTROL equipped with PLS7 / PLS9.

If interpolation commands follow each other in succession (without other commands in-between), they are processed in one piece. Path speed and interpolation commands as well as commands of the G16x group (operation of outputs during interpolation) can be combined.

NOTE
The path acceleration and the path speed must always be specified before execution of the interpolation (see example 1 on page 164).

Command format:
G01 → AiAn
G01 → Ari ARn

Application:
Simultaneous traversing of 2 to 4 axes moving along a straight line between the start and target position.

2 = PA-CONTROL STEUER with PLS7 / 9
3 = PA-CONTROL COMPACT with PLS7 / 9
Example 1

2. A1:=10 ;Traverse A1 axis to position 10
3. A2:=20 ;Traverse A2 axis to position 20
4. G100.B.100 ;Specify path acceleration for complete contour
5. FB:=1000 ;Set path speed to 1000AE/s
6. G01 A1:=40 A2:=30 ;Linear interpolation with axes A1 and A2 from current position to position A1=40 and A2=30
7. FB:=800 ;Change of the path speed to 800AE/s for next interpolation sections with the ramp of the pre-set acceleration
8. G01 A1:=100 A2:=100 ;Interpolation is continued at the changed path speed without stopping
10. END

Example 2

Complex example of multiple linear interpolations in succession to trace a contour at a constant path speed. Due to the fact that the interpolation commands are defined in succession (no other command in-between), the distances "a" to "f" are traced without stopping.

1. O6:=0 I5.1 ;Tool up
3. A1:=30 A2:=20 ;Approach edge of workpiece
4. O6:=1 ;Lower tool
5. I6.1 ;Wait until tool is down
6. FB:=500 ;Set path speed to 500AE/s
7. G100.B.30 ;Set path speed to 30AE/ss
8. G01 A1:=40 A2:=40 ;Trace lateral edge "a"
9. G01 A1:=60 A2:=40 ;Trace lateral edge "b"
10. FB:=300 ;Set path speed to 300AE/s ;no stop is executed here, the transition to the new path speed ensues with the ramp of the pre-set acceleration
11. G01 A1:=70 A2:=30 ;Trace lateral edge "c"
12. G01 A1:=70 A2:=25 ;Trace lateral edge "d"
13. G01 A1:=60 A2:=20 ;Trace lateral edge "e"
14. G01 A1:=30 A2:=20 ;Trace lateral edge "f"
15. O6:=0 ;Tool up
16. A1:=5 A2:=5 ;Approach initial position
17. END
3.43.2 Abort interpolation

**IPOEND**

*IPOEND.Counter.Limit*
IPOEND.Ni.Nn
IPOEND.Ni.Nn

**Description:**
The IPOEND command enables the user to abort an interpolation.

**Function:**
A check is carried out for each IPO command (G01) as to whether the function IPOEND has been activated. If this is not the case, the programme is continued as described. If this is the case, a check is carried out as to whether the counter is less than the limit (Ni<Nn). In this case, the IPO command is executed.

If the counter is the same as or higher than the limit, processing of the IPO command is discontinued and the program is continued after the last IPO command.

**Please note:**
- IPOEND is normally inactive and is deactivated at the end of interpolation.
- The counter and the limit cannot be changed during interpolation.
- IPOEND sets the counter to 0 (at the beginning of processing).
- At least one IPO command (G01) is executed.
Example

1. N7:=3
2. SUB TRAVERSE_IPO
   ...
6. N7:=5
7. SUB TRAVERSE_IPO
   ...
15. $ TRAVERSE_IPO
16. IPOEND.N2.N7  
   Activate IPOEND, i.e. three or five IPO commands are executed
17. G100.B.100  
   ;Specify the path acceleration for the complete contour
18. FB:=100  
   ;Set path speed to 100AE/s
19. G01 A1:=R1 A2:=R2  
   ;Linear interpolation with the A1 and A2 axis from the current position to position A1=<R1> and A2=>R2>
20. G01 A1:=R3 A2:=R4  
   ;Linear interpolation with the A1 and A2 axis from the current position to position A1=<R3> und A2=>R4>
   ;Linear interpolation with the A1 and A2 axis from the current position to position A1=<R5> and A2=>R6>
22. G01 A1:=R7 A2:=R8
25. ;
26. END
3.44 G16x – Operation of outputs during interpolation

**Description:**
The G160 commands are used to control outputs or flags synchronous to the interpolation command.

**General:**

\[G16x.\text{[Time]} .\text{[Element]}.\text{[Status]}\]

- **[Time]** : Time specification for the action in \(n \times 10\text{ms}\)
- **[Element]** : \(O \rightarrow \text{output}, M \rightarrow \text{flag}\)
- **[Status]** : \(0 \rightarrow \text{is reset}, 1 \rightarrow \text{is set}\)

**NOTE**
- Only one storage location is provided for a G160, G161 or G162 command in an interpolation interval.
- Only logical commands and parallel runs can be used to control multiple outputs.
- If the duration of an interpolation interval is shorter than specified in the G161 or G162 commands, the element is controlled by the G161 command at the end of the interval and by the G162 command at the beginning of the interval.

### Example 1

- **n**

  \[G160.20.O123.1\]
  - Set output O123 to 1 and wait 200ms before starting the subsequent interpolation

- **n+1**

  \[G01 A1:=10 A2:=20\]
  - Linear interpolation with axes A1 and A2 from the current position to position A1=10 and A2=20

- **...**

  \[G01 A1:=30 A2:=30\]
  - Linear interpolation with axes A1 and A2 from the current position to position A1=30 and A2=30. The second interpolation directly follows the first without any interruption!
3.44.2 G161 / G162 - Activate outputs during interpolation

**Example 2**

<table>
<thead>
<tr>
<th>n</th>
<th>G01 A1:=10 A2:=20</th>
</tr>
</thead>
<tbody>
<tr>
<td>n+1</td>
<td>G161.1.O333.1</td>
</tr>
<tr>
<td></td>
<td>Set output O333 to &quot;1&quot; 10ms after start of the subsequent interpolation interval.</td>
</tr>
<tr>
<td>n+2</td>
<td>G01 A1:=30 A2:=30</td>
</tr>
<tr>
<td></td>
<td>;Linear interpolation with axes A1 and A2 from the current position to position A1=30 and A2=30</td>
</tr>
<tr>
<td>...</td>
<td>G01 A1:=60 A2:=40</td>
</tr>
<tr>
<td></td>
<td>;Linear interpolation with axes A1 and A2 from the current position to position A1=60 and A2=40</td>
</tr>
<tr>
<td>...</td>
<td>G162.5.O333.0</td>
</tr>
<tr>
<td></td>
<td>Set output O333 to &quot;0&quot; 50ms before the next interpolation interval is completed.</td>
</tr>
<tr>
<td>...</td>
<td>G01 A1:=90 A2:=50</td>
</tr>
</tbody>
</table>

**Example 3**

<table>
<thead>
<tr>
<th>n</th>
<th>G01 A1:=10 A2:=20</th>
</tr>
</thead>
<tbody>
<tr>
<td>n+1</td>
<td>G161.1.M34.1</td>
</tr>
<tr>
<td></td>
<td>Set flag M34 to &quot;1&quot; 10ms after the subsequent interpolation interval was started.</td>
</tr>
<tr>
<td></td>
<td>;Linear interpolation with axes A1 and A2 from the current position to position A1=30 und A2=30</td>
</tr>
<tr>
<td>...</td>
<td>G162.5.M34.0</td>
</tr>
<tr>
<td></td>
<td>Set flag M34 to &quot;0&quot; 50ms before the next interpolation interval is completed.</td>
</tr>
<tr>
<td>...</td>
<td>G01 A1:=300 A2:=250</td>
</tr>
</tbody>
</table>

**Example 4**

<table>
<thead>
<tr>
<th>n</th>
<th>G01 A1:=10 A2:=20</th>
</tr>
</thead>
<tbody>
<tr>
<td>n+1</td>
<td>G161.1.O333.1</td>
</tr>
<tr>
<td></td>
<td>Set output O333 to &quot;1&quot; 10ms after the subsequent interpolation interval was started.</td>
</tr>
<tr>
<td>n+2</td>
<td>G01 A1:=30 A2:=30</td>
</tr>
<tr>
<td></td>
<td>;Linear interpolation with axes A1 and A2 from the current position to position A1=30 und A2=30</td>
</tr>
<tr>
<td>...</td>
<td>G162.5.O334.1</td>
</tr>
<tr>
<td></td>
<td>Set output O334 to &quot;1&quot; 50ms before the next interpolation interval is completed.</td>
</tr>
<tr>
<td>...</td>
<td>G01 A1:=60 A2:=40</td>
</tr>
<tr>
<td></td>
<td>;Linear interpolation with axes A1 and A2 from the current position to position A1=60 and A2=40</td>
</tr>
<tr>
<td>...</td>
<td>G162.5.M3.1</td>
</tr>
<tr>
<td></td>
<td>Set flag M3 334 to &quot;1&quot; 50ms before the next interpolation interval is completed.</td>
</tr>
<tr>
<td>...</td>
<td>G01 A1:=90 A2:=50</td>
</tr>
</tbody>
</table>
3.45  G170 - Read character from PTX file

**G170.(Line).(Column).(Character).(File)**

**Description:**

The G170 command reads a character from a PTX file and stores the determined ASCII value of the character in the integer register Nm (see Table "PAC key code + ASCII character set". 414 in Chapter 7).

The exact position of the character in the file is determined via the integer registers Ni and Nn. Ni specifies the line number and Nn the place in the selected line.

**NOTE**

The value of the integer registers Ni and Nn must be greater than 0.

If the program file does not exist, the message "Program does not exist" is output.

If line Ni does not exist in the PTX file or the selected line is shorter than Nn, register Nm is set to 0 and the system flag SM6 is set.

A check can be carried out as to whether the command has been executed without error by means of the system flag. SM6=0 \(\Rightarrow\) command has been correctly executed, SM6=1 command execution with error.

**Command format:**

G170.Ni.Nn.Nm.TEXTS  
G170.Ni.Nn.Nm.NAMES

**Example**

1. \(N1:=5\); Line number in file TEXTS.PTX
2. \(N2:=3\); 3rd character in the 5th line
3. ; N10

5. **END**

**Programm: TEXTS.PTX**

1. MONDAY
2. TUESDAY
3. WEDNESDAY
4. THURSDAY
5. FRIDAY
6. SATURDAY
7. SUNDAY

End of the assignment
### 3.46 G171 - Write character to PTX file

**G171**

\[ G171.(Line).(Column).(Character).(File) \]

**Description:**

The G171 command writes a character to a PTX file. The exact position of the character in the file is determined via the integer registers Ni and Nn. Ni specifies the line number and Nn the place in the selected line.

The numerical value from the integer register Nm is converted to ASCII code (see Table "PAC key code + ASCII character set; see Chapter 7.13). Illegal characters are: 0x00=NUL, 0x0A=LF and ox0D=CR.

**NOTE**

The value of the integer registers Ni and Nn must be greater than 0.

If the program file does not exist, the message "Program does not exist " is output. If line Ni does not exist in the PTX file or the selected line is shorter than Nn, the system flag SM6 is set.

A check can be carried out as to whether the command has been executed without error by means of the system flag. SM6=0 → command has been correctly executed, SM6=1 command execution with error.

**Command format:**

\[ G171.Ni.Nn.Nm.TEXTS \]
\[ G171.N!i.N!n.N!m.NAMES \]

**Example**

```
1  N1:=5 ;Line number in file TEXTS.PTX
2  N2:=3 ,3rd character in 5th line
3    ; N10 ;Register N10 contains the read ASCII value
4  G171.N1.N2.N10.TEXTE ;Overwrites the 3rd character in 5th line of file TEXTS.PTX with the ASCII value (value from N10)
5    END
```

**Program: TEXTS.PTX**

```
1  MONDAY
2  TUESDAY
3  WEDNESDAY
4  THURSDAY
5  FRIDAY
6  SATURDAY
7  SUNDAY
```

End of the assignment
3.47 G172 - Write line from PTX file to string (Sn)

G172

\[ G172.(\text{Line}).(\text{String}).(\text{File}) \]

**Description:**
The G172 command copies the line specified in the command from a PTX file to a character string. In this connection, the line is specified directly in the command or addressed via a register. The target address can be either a global string or the local string.

**NOTE**
- If the file does not exist, the error message "Program does not exist" will be output.
- If the addressed line does not exist or the line is longer than the maximum string length, the system flag "SM6" is set.
- A check can be carried out as to whether the command has been executed without error by means of the system flag. SM6=0 → command has been correctly executed, SM6=1 command execution with error.

**Command format:**
- G172.4.S2.ERRORTEXTS

**Example 1**

1. G172.5.S2.MENUTEXTS
   ;Copies the 5th line of the file "MENUTEXTS.PTX" to the string "S2"
2. END

**Example 2**

1. N8:=12
   ;Line number in file ERRORTEXTS.PTX
2. G172.N8.S0.ERRORTEXTS
   ;Copies the 12th line from the file "ERRORTEXTS.PTX" to the local character buffer S0
3. END
3.48  G173 - Store string (Sn) in a line of a PTX file

G173

G173.(Line).(String).(File)

Description:
The G173 command stores the content of the string "Sn" in a selected line of a PTX file. In this connection, the line is specified directly in the command or addressed via a register. The target address can be either a global string or the local string.

NOTE
If the file does not exist, the error message "Program does not exist" is output. If the addressed line does not exist or the string is longer than the line, the error message "STORE command could not be implemented correctly" is output.
If the line of the PTX file is longer, the rest of the line is overwritten with blank characters. A line should not be longer than 80 characters, otherwise the message "Line too long" is output when the command G172 is applied.

Command format:
G173.6.S0. MESSAGES
G173.N2.S1.ERROR_1

Example 1
1  G173.6.S0.MESSAGES ;Stores the contents of the local string "S0" in the 6th line of the file "MESSAGES"
2  END

Example 2
1  N2:=21 ;LINE number of the file ERROR_1.PTX
2  G173.N2.S1.ERROR_1 ;Stores the contents of the global string "S1" in the 21st line of the file "ERROR_1.PTX".
3  END
3.49 STORE - Storage of values

Store.(Name)

Description:
All the current values of N registers, R registers, strings and flags are stored in a PNC, PNX or PAB program by means of the command STORE.

When executing the STORE command, the invoked PNC program is checked for assignments. If a register, flag or string assignment is found, the current value is entered in the assignment. Sufficient blank characters must be available in the PNX, PNC oder PAB programs for this purpose.

- Maximum required number of placeholders for registers: 9 places
- Maximum required number of placeholders for flags: 1 place
- Maximum required number of placeholders for strings: current string length

Before executing the STORE command

1  R10:=0 → ; A1 pick-up position
2  R11:=0 → ; A1 intermediate position
3  N10:=0 → ; Number of products to be processed
4  M101:=0 → ; Product flag for product 1
5  S1:="--------" → ; Greeting
6  END

After executing the STORE command

1  R10:=145.345 → ; A1 pick-up position
2  R11:=34565.789 → ; A1 intermediate position
3  N10:=645 → ; Number of products to be processed
4  M101:=1 → ; Product flag for product 2
5  S1:="HELLO" → ; Greeting
6  END

NOTE

The invoked PNC/PNC/PAB program must exist. A program line may only contain one assignment.

Sufficient zeros or blank characters must be available for overwriting after the assignment up to ";," (comment) or "CR" (end of line).

Calculation assignments are not allowed in the program which is to be overwritten.

If the STORE instruction is not successful because of a lack of free space, error E569 is set.
Command format:
STORE.(Name)

Application:
Storage of current values in various program files, e.g. traversing positions (R register) or status flags can be stored for different product types and called up again as required.

Example

1  \texttt{R10:=R13} \quad ;\text{Read value via register e.g. 4678.123}
2  \texttt{N1:=N13} \quad ;\text{Read value via register e.g. 967}
3  \texttt{M25:=N1>11} \quad ;\text{M25 is set if content of N1 is greater than 11.}
4  \texttt{S1:=\text{GRIPPER}} \quad ;\text{GRIPPER is loaded to string S1}
5  \texttt{STORE.TA2\_PE\_001} \quad ;\text{Overwrite assignments in the PNC program TA2:=PE\_001 with current values of the registers and flags}
6  \texttt{END}

Program TA2\_PE\_001 before execution of the STORE command:
1  \texttt{R10:=0\_\_i} \quad ;\text{A1 pick-up position for product type 1}
2  \texttt{N1:=0\_\_i} \quad ;\text{Number of products to be processed}
3  \texttt{M25:=0} \quad ;\text{Product flag for product 1}
4  \texttt{S1:=\text{\_LAMP\_025\_}} \quad ;\text{Labelling for subsequent output}
5  \texttt{END}

Program TA2\_PE\_001 after execution of the STORE command:
1  \texttt{R10:=4678.123} \quad ;\text{A1 pick-up position}
2  \texttt{N1:=967} \quad ;\text{Number of products to be processed}
3  \texttt{M25:=1} \quad ;\text{Product flag for product 1}
4  \texttt{S1:=\text{\_LAMP\_016\_}} \quad ;\text{Labelling for subsequent output}
5  \texttt{END}

NOTE
In the example, blank characters are represented by an underscore " \_ ".


3.50 CASE.STORE - Storage of values

CASE.STORE.(Variable)

Description:
The CASE.STORE command is a branch which is dependent on the content of the integer register Ni. PA-CONTROL checks the content of the integer register and carries out a subroutine call in accordance with the value. If the content of the integer register is less than 1 or greater than the number of elements in the name table (in this example: TA2:=PE_001, TA2:=PE_002), a jump to the label defined after ELSE is carried out. Otherwise the respective subroutine is called and processed and the program run is continued with the program line after the ELSE branch.

The current values of N registers, R registers and flags are stored in a PNC/PAB program by means of the CASE.STORE command (see STORE command).

Command format:
CASE.STORE.Ni
(Name1)
(Name2)
...
ELSE Label_e

Application:
Branch for storing current register values in various PNC/PAB files, e.g. traversing positions (R register) or status flags can be stored for different product types and called up again as required.
Example

1. $ERROR
2. N8:=N13 ;Read value via register
3. CASE.STORE.N8 ;Check contents of N8 and call subroutine if:
4. (TA2_PE_001) ;N8=1: check subroutine "TA2:=PE_001" for assignments and update
5. (TA2_PE_002) N8=2: check subroutine "TA2:=PE_002" for assignments and update
6. ELSE ERROR ;Jump to label "ERROR" if value of N8 is out of the specified range (in this case if N8<1 or N8>2)
7. END

Program: TA2_PE_001
1. R10:=0 ;A1 pick-up position for product type 1
2. R11:=0 ;A1 intermediate position for product type 1
3. N10:=0 ;Number of products to be processed
4. M101:=0 ;Product flag for product 1
5. END

Program: TA2_PE_002
1. R10:=0 ;A1 pick-up position for product type 2
2. R11:=0 ;A1 intermediate position for product type 2
3. N10:=0 ;Number of products to be processed
4. M102:=0 ;Product flag for product 2
5. END
### 3.51 Commands of the G2xx group

<table>
<thead>
<tr>
<th>G2xx</th>
</tr>
</thead>
</table>

**Introduction:**
PA-CONTROL has two modes for processing positioning commands.

- **Standard positioning mode:** The positioning procedure is started and the process waits until the axis has reached its target position. The next command in the program is not processed until then. If several positioning commands are defined consecutively in one line, they are processed in parallel.

- **Start positioning mode:** The positioning procedure is started in order to continue with the next command in the program. The positioning procedure is controlled in the background.

The standard positioning mode is the default mode.

The switch-over and behaviour are selected according to the axis.

If an axis is in the start positioning mode, it can be accessed through various commands of the G200 group. Before re-starting the axis, a check should be made via one of the commands (G213, G211, G221, etc.) to ensure that it is not traversing.
3.51.1  G210 – Activate the start positioning mode

G210

G210.(Axis)

Description:
PA-CONTROL has two operating modes. In the standard positioning mode, see page 177, the controller waits for the movement to end after an axis start, before starting the next command. In the start positioning mode, the traverse of a second and further axes can be initiated immediately after the start of an axis. The axes traverse in the background.

The G210 command activates the start positioning mode, see page 177, and deactivates the standard positioning mode.

When programming, always remember that the movement of one axis must be terminated before a further start. An error results if this pre-requisite is disregarded. When the start positioning mode is used, a check should be made to determine whether the axis can be started at this time.

The G213 command is used to change over to the standard positioning mode.

Command format:
G210.A0
G210.A1

Application:
The positioning of an axis is started. The controller continues to process the programme without waiting for the axis to arrive at the stop position.

Example
See the following commands
3.51.2 G211 - Position-conditional jump

G211.(Axis).(Condition.) (Mask)

Please note:
The command only evaluates whether the axis/axes are in position or not in position.

Command format:
G211.A0.j Label  
G211.An.j Label

Application:
Execution of a jump, depending on whether the selected axis or all axes are traversing or are at standstill. A stands for the axis number, j for the condition.

The following applies:
• A0  →  All axes
• A1  →  Axis 1
• A2  →  Axis 2
• An  →  Axis n

• j = 0  →  jump if the position of the selected axes has not yet been reached
• j = 1  →  jump if the position of the selected axes has been reached

Example 1

2  G90.A0  ;Absolute dimension system
3  G210.A0  ;Activate the start positioning mode for all axes
4  A1:=10000 A2:=6000  ;Start axes
5  $SET
6  O1:=1 T50 O1:=0  ;Set output 1 for 500ms
7  T50  ;Wait 500ms
8  G211.A0.0 SET  ;Jump to the label "SET" if all axes have not yet reached their position
9  G213.A0  ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
10  O2:=1
11  END
Example 2

2. G90.A0 ; Absolute dimension system
3. G210.A0 ; Activate the start positioning mode for all axes
4. A1:=10000 A2:=6000 ; Start axes
5. $LOOP
6. O1:=1 T50 O1:=0 ; Set output 1 for 500ms
7. T50 ; Wait 500ms
8. G211.A2.1 READY ; Jump to "READY" if the A2 axis has reached its position
9. JMP LOOP ; Jump to "LOOP"
10. $READY
11. G213.A0 ; Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
12. O2:=1
13. END

Example 3

2. G90.A0 ; Absolute dimension system
3. G210.A0 ; Activate the start positioning mode for all axes
4. A1:=10000 A2:=6000 ; Start axes
5. $LOOP
6. O1:=1 T50 O1:=0 ; Set output 1 for 500ms
7. T50 ; Wait 500ms
8. G211.A1.0 LOOP ; Jump to the label "LOOP" if the A1 axis has not yet reached its position
9. G213.A0 ; Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
10. O2:=1
11. END
3.51.3 G212 - Position-conditional subroutine call

G212

G212.(Axis).(Condition) (Program)

**NOTE**
The command only evaluates whether the axis/axes are in position or not in position.

**Command format:**
- G212.A0.j Name
- G212.An.j Name

**Application:**
Execution of a subroutine call, depending on whether the selected axis or all axes are traversing or are at standstill. A stands for the axis name, j for the condition.

**The following applies:**
- A0 → All axes
- A1 → Axis 1
- A2 → Axis 2
- An → Axis n

- j = 0
  - been → jump to subroutine if the position of the selected axes has not yet reached.
- j = 1
  - reached. → jump to subroutine if the position of the selected axes has been reached.
Example

2. G210.A0 ;Activate the start positioning mode for all axes
3. A1:=100 A2:=500 ;Start axes
4. $LOOP ;Jump label
5. G212.A0.0 BLINKING ;Jump to subroutine BLINKING if all axes have not yet reached their position
6. G211.A1.0 LOOP ;Jump to the label "LOOP" if the A1 axis has not yet reached its position
7. G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
8. END

Program: BLINKING.PNC

1. O9:=1 ;Set output 1
2. T50 ;Wait 500ms
3. O9:=0 ;Reset output 1
4. T20 ;Wait 200ms
5. END
3.51.4 G213 - Activate the standard positioning mode

G213

G213.(Axis)

Description:
The G213 command activates the standard positioning mode and deactivates the start positioning mode. The controller waits until all current positioning processes for all axes (G213.A0) have been concluded or until the current positioning process for a defined axis (G213.Ai) has been concluded before starting the next command.

Command format:
G213.A0
G213.An

Example
see other G200 commands
3.51.5 G221 - Position-conditional jump (current position)

G221

\[ G221.(\text{Condition}).(\text{Axis}).(\text{Position}) \ (\text{Label}) \]

**Command format:**

G221.j.An.Pos Label
G221.j.ARn.Pos Label

**Application:**

A conditional jump depending on a current axis position can be executed by means of the command G221. j stands for the jump condition, An for the axis and the position and label for the target to be jumped to.

**The following applies:**

| j = 0 | → jump if the current axis position is less than the position (An) |
| j = 1 | → jump if the current axis position is greater than the position (An) |

**NOTE**

This command evaluates the absolute position of the stated axis irrespective of the selected measuring system (G90.An / G91.An).

**Example 1**

1. \( G25.A1 \ G25.A2 \) ;Approach to reference points of the A1 and A2 axis
2. \( G90.A0 \) ;Absolute dimension system
3. \( G210.A0 \) ;Activate the start positioning mode for all axes
4. \( A1:=10000 \ A2:=6000 \) ;Start axes
5. \( \$\text{LOOP} \)
6. \( \text{O1:=1} \ \text{T50 O1:=0} \ \text{T50} \)
7. \( \text{G221.1.A1.8000} \ \text{RED} \) ;Jump to "RED" if the A1 position is greater than 8000
8. \( \text{O2:=1} \ \text{T10 O2:=0} \)
9. \( \$\text{ROT} \)
10. \( \text{G211.A0.0} \ \text{LOOP} \) ;Jump to "LOOP" if all axes are not yet in position
11. \( \text{G213.A0} \) ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
12. \( \text{END} \)
Example 2

2. G90.A0 ;Absolute dimension system
3. G210.A0 ;Activate the start positioning mode for all axes
4. A1:=10000 A2:=6000 ;Start axes
5. $LOOP
6. O1:=1 T50 O1:=0 T50
7. G221.1.A2.2000 RED ;Jump to "RED" if the A2 position is greater than 2000
8. O2:=1 T10 O2:=0
9. $RED
10. G211.A0.0 LOOP ;Jump to "LOOP" if all axes are not yet in position
11. G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
12. END

Example 3

2. G90.A0 ;Absolute dimension system
3. G210.A0 ;Activate the start positioning mode for all axes
4. A1:=10000 A2:=6000 ;Start axes
5. $LOOP
6. O1:=1 T50 O1:=0 T50
7. G221.0.A1.8000 RED ;Jump to "RED" if the A1 position is greater than 8000
8. O2:=1 T10 O2:=0
9. $RED
10. G211.A0.0 LOOP ;Jump to "LOOP" if all axes are not yet in position
11. G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
12. END
3.51.6 G222 - Position-conditional subroutine call (current pos.)

G222

G222.(Condition) (Axis).(Position) (Program)

NOTE
This command evaluates the absolute position of the stated axis irrespective of
the selected measuring system (G90.An / G91.An).

Command format:
G222 j.An.Pos Name

Application:
A conditional subroutine call depending on a current axis position can be executed by means
of the command G222.
j stands for the jump condition, An for the position of the axis and Name for the subroutine
which is to be called.
The following applies:

| j = 0  | → jump if the current axis position is less than the position (An) |
| j = 1  | → jump if the current axis position is greater than the position (An) |

Example 1

2  G90.A0 ;Absolute dimension system
3  G210.A0 ;Activates the start positioning mode for all axes
4  A1:=10000 A2:=6000 ;Start axes
5  $LOOP
6  O1:=1 T50 O1:=0 T50
7  G222.1.A1.8000 RED ;Call of the subroutine "RED" if the A1 position is greater than 8000
8  G211.A0.0 LOOP ;Jump to "LOOP" if all axes are not yet in position
9  G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
10 END

Program: RED
1  O2:=1 T10 O2:=0
2  END
Example 2

2   G90.A0 ;Absolute dimension system
3   G210.A0 ;Activates the start positioning mode for all axes
4   A1:=10000 A2:=6000 ;Start axes
5   $LOOP
6   O1:=1 T50 O1:=0 T50
7   G222.1.A2.1000 RED ;Call of the subroutine "RED" if the A2:= position is greater than 1000
8   G211.A0.0 $LOOP ;Jump to "LOOP" if all axes are not yet in position
9   G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
10  END

Program: RED
1  O2:=1 T10 O2:=0
2   END

Example 3

2   G90.A0 ;Absolute dimension system
3   G210.A0 ;Activates the start positioning mode for all axes
4   A1:=10000 A2:=6000 ;Start axes
5   $LOOP
6   O1:=1 T50 O1:=0 T50
7   G222.0.A1.4000 RED ;Call of the subroutine "RED" if the A1 position is less than 4000
8   G211.A0.0 $LOOP ;Jump to "LOOP" if all axes are not yet in position
9   G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
10  END

Program: RED
1  O2:=1 T10 O2:=0
2   END
3.51.7 G230 - Wait until current position </> than value

G230

\[ G230.(\text{Condition}).(\text{Axis}).(\text{Position}) \]

**NOTE**

This command evaluates the absolute position of the stated axis irrespective of the selected measuring system (G90.An / G91.An).

**Command format:**

\[ G230.j.An.Pos \]
\[ G230.j.An.R!n \]

**Application:**

The further processing of the program can be interrupted until a certain position has been reached by means of the G230 command.

j stands for the condition, An for the axis and the position.

The following applies:

<table>
<thead>
<tr>
<th>j</th>
<th>Condition</th>
<th>Result</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>wait until the position is less than the current position</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>wait until the position is greater than the current position</td>
<td></td>
</tr>
</tbody>
</table>
Example

1  G25.A1 ;Approach to reference point of the A1 axis
2  G90.A0 ;Absolute dimension system
3  R10:=3000
4  G210.A0 ;Activate the start positioning mode for all axes
5  A1:=10000 ;Start the axis
6  O1:=1 T10 O1:=0
7  G230.1.A1.3000 O2:=1 ;Output 2 is set from A1:=3000 to A1:=5000
8  G230.1.A1.5000 O2:=0
9  O1:=1 T10 O1:=0
10 G213.A0
11 G210.A0 ;Process commands beyond line limit; beginning
12 A1:=0
13 O1:=1
14 R5:=10
15 G230.0.A1.R!5 O1:=0 ;Output 1 is set from A1:=10000 to A1:=3000 (value 3000 is in R10)
16 G213.A0 ;Wait until all axes are in position and activate the standard positioning mode (deactivate the start positioning mode)
17 END
3.52 G4xx - Time monitoring commands

Introduction

Time monitoring commands of the PA-CONTROL Family

The commands described below are used if the defined end of processes have to be monitored and utility programs for troubleshooting or error registration have to be started in the case of irregularities. Such situations can occur when processes which run in parallel to and independently of PA-CONTROL programs, e.g. the activation of pneumatic cylinders, are started by the controller.

After the time has elapsed, a conditional jump or subroutine call can be executed depending on the specified command. The set time conditions can be reset or cancelled. The time condition is checked and also reacted to before PA-CONTROL executes a command.
3.52.1 G421 - Time monitoring with a conditional jump

G421.1.

G421.1.(Time) (Label)

Description
The monitoring time is activated by the command G421.1.200 label. The monitoring is reset if the reset command G401.1 is detected before the monitoring time has elapsed. Otherwise the program is continued at the specified label.

A G421 command running in the program is cancelled by another G421 or G422, and a new time monitoring command is started at the same time. The remaining time of the first monitoring job is not processed any further.

NOTE After a monitoring time has been started, a subroutine call should not be executed, since the branch target for the G421 command is no longer available.

Command format
G421.1.nnn Label
G421.1.N1 Label

NOTE The maximum transfer time is:
with specification via constant 65535 x 10ms
with specification via N register $2^{31}$ x 10ms

Example

1 $BEGINNING
2 O1:=1 ;Set output 1
3 G421.1.200 ERROR_1 ;Jump to label "ERROR_1" if 2 seconds have elapsed and the time condition has not been reset
4 I1.1 O1:=0 ;Wait for input 1 = "1" and reset output 1
5 G401.1 ;Reset the time condition
6 END
7 $ERROR_1
8 O2:=1 ;Set malfunction signal
9 I2.1 ;Wait for error acknowledgement signal
10 O1:=0 O2:=0 ;Reset output 1 and malfunction signal
11 JMP BEGINNING
3.52.2 G422 - Time monitoring with conditional subroutine call

G422.1.

**G422.1.(Time) (File)**

**Description**

The monitoring time is started by the command G422.1.200 Name. The monitoring is reset if the reset command G401.1 is detected before the monitoring time of 2 seconds (200*10ms) has elapsed. Otherwise a jump to the specified subroutine is executed.

**NOTE** The return from the time monitoring subroutine (in the example ERROR) follows the PAC command which was interrupted by the time monitoring.

**Command format**

G422.1.nnn Name

G422.1.Ui Label

**NOTE** The maximum transfer time is:

- with specification via constant \( 65535 \times 10\text{ms} \)
- with specification via N register \( 2^{31} \times 10\text{ms} \)

**Example**

1. \( O1:=1 \); Set output 1
2. \( G422.1.200 \text{ ERROR} \); Subroutine call if 2 seconds have elapsed and the time condition has not been reset
3. \( I1.1 \text{ } O1:=0 \); Wait for input 1 = "1" and reset output 1
4. \( G401.1 \)
5. \( \text{END} \)

**Program: ERROR**

1. \( O3:=1 \); Malfunction at ...
2. \( I5.1 \); Acknowledgement by the operator
3. \( I5.0 \)
4. \( \text{END} \)
3.52.3 G423 - with conditional subroutine call (return in same line)

**G423.1.**

**G423.1.(Time) (File)**

**Description**

The monitoring time is started by the command G423.1.200 Name. The monitoring is reset if the reset command G401.1 is detected before the monitoring time of 2 seconds (200*10ms) has elapsed. Otherwise a jump to the specified subroutine is executed.

**NOTE**

After returning from the subroutine, the program is continued in the same line in which the G423 command is located. The program in which the G423 command was activated must still be active, i.e. termination may not be executed at another point by the CANCEL command.

**Command format**

G423.1.nnn Name

G423.1.Ni Label

**NOTE**

The maximum transfer time is:

- with specification via constant $65535 \times 10\text{ms}$
- with specification via N register $2^{31} \times 10\text{ms}$

**Example**

1. O1:=1 ;Set output 1
2. G423.1.200 ERROR ;Subroutine call if 2 seconds have elapsed and the time condition has not been reset. After returning from the subroutine, the program is continued in the same line in which the G423 command is located. The time monitoring is run through once again, thereby ensuring that input 1 really has the logical state 1. The program in which the G423 command was activated must still be active
3. I1.1 O1:=0 ;Wait for input 1 = "1" and reset output 1
4. G401.1
5. END

**Program: ERROR**

1. O3:=1 ;Malfunction at ...
2. I5.1 ;Acknowledgement by the operator
3. END
3.52.4 G401 - Reset of the time condition

G401.1

Description
The time monitoring which was started by the commands G421, G422 or G423 is stopped by means of this command. Time monitoring is therefore no longer active.

Command format
G401.1

Example
See examples of the commands G421 or G422 or G423
3.53 Real-time clock

NOTE All the commands in this group are only implemented in the PA-CONTROL CPU4

The commands of this command group enable access i.e. the reading of and writing to the real-time clock implemented on the CPU4 board.

The following table applies for the commands of this command group:

<table>
<thead>
<tr>
<th>Syntax</th>
<th>Name</th>
<th>Number of digits</th>
<th>Value range (with leading zeros)</th>
</tr>
</thead>
<tbody>
<tr>
<td>YEAR</td>
<td>Year</td>
<td>4</td>
<td>1997 – 2040</td>
</tr>
<tr>
<td>MONTH</td>
<td>Month</td>
<td>2</td>
<td>01 – 12</td>
</tr>
<tr>
<td>DAY</td>
<td>Day</td>
<td>2</td>
<td>01 – 31</td>
</tr>
<tr>
<td>HOUR</td>
<td>Hours</td>
<td>2</td>
<td>00 – 23</td>
</tr>
<tr>
<td>MIN</td>
<td>Minutes</td>
<td>2</td>
<td>00 – 59</td>
</tr>
<tr>
<td>SEC</td>
<td>Seconds</td>
<td>2</td>
<td>00 – 59</td>
</tr>
<tr>
<td>WD</td>
<td>Weekday</td>
<td>1</td>
<td>0=Sunday - 6=Saturday</td>
</tr>
<tr>
<td>YD</td>
<td>Year day</td>
<td>3</td>
<td>001 – 365</td>
</tr>
<tr>
<td>W</td>
<td>Calendar week</td>
<td>2</td>
<td>01 - 53</td>
</tr>
</tbody>
</table>

Only the maximum number of elements (max. 9) are verified in the syntax check.

3.53.1 Read-out of the real-time clock

\[ S0:=\text{TIME} \]

\[ S0:=\text{TIME.YEAR\[.MONTH\[.DAY\].HOUR\[.MIN\].SEC\].WD\[.YD\].W} \]

**Description**

This command enables access to the real-time clock of the controller. Information is always output to the global S0 string. The elements in the [ ] are optional and may be mixed. The elements can be separated by means of a dot, hyphen or colon. The separator used is transferred to the same position in the S0 string.

The following input examples refer to the 4th December 2002, 08:41:53

<table>
<thead>
<tr>
<th>Command format</th>
<th>Output</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0:=TIME.DAY.MONTH.YEAR</td>
<td>04.12.2002</td>
</tr>
<tr>
<td>S0:=TIME.DAY.MONTH.YEAR-HOUR:MIN:SEC</td>
<td>04.12.02-08:41:53</td>
</tr>
<tr>
<td>S0:=TIME.HOUR:MIN:SEC</td>
<td>08:41:53</td>
</tr>
<tr>
<td>S0:=TIME.W</td>
<td>49</td>
</tr>
<tr>
<td>S0:=TIME.W-WWD</td>
<td>49-3</td>
</tr>
</tbody>
</table>
Example

1  \texttt{S0:=TIME.DAY.MONTH.YEAR} ; Get day, month and year from the real-time clock and write this information to the global string S0
2  \texttt{G11.0} ; Switch off run displays
3  \texttt{G500.0} ; Switches the G500 commands over to the current display medium (LC display or the serial port with "Simulation of the front plate")
4  \texttt{G501} ; Clears the display and places the cursor in the upper left corner (1st. column, 1st line)
5  \texttt{G510.CURRENT DATE} ; Output of text "CURRENT DATE"
6  \texttt{G503.17.1} ; Position the cursor in the 17th column, 1st line
7  \texttt{G520.S0} ; Display the contents of the global string
8  \texttt{T500} ; Wait 5 seconds
9  \texttt{END}
3.53.2 Set the date from the user program

\[ \text{DATE}:= \]

\[ \text{DATE}=(\text{Day}).(\text{Month}).(\text{Year}).(\text{Weekday}) \]

**Description**

The date and the week day of the real-time clock on the CPU4 can be set from the user program by means of this command.

**Command format:**

Date:=Ni.Nj.Nn.Nm

**NOTE**

On execution of the command, a check is carried out to ensure that the values for the date are legal. If this is not the case, error E532 "Value outside range" is set.

**Example**

1. \( N1:=15 \)
2. \( N2:=8 \)
3. \( N3:=2003 \)
4. \( N4:=5 \)
5. \( \text{DATE}:=N1.N2.N3.N4 \quad \text{Sets the date to Friday 15th August 2003} \)
3.53.3 Set the time from the user program

\[
\text{TIME:=}[\text{Hours}].[\text{Minutes}].[\text{Seconds}]
\]

**Description**

The time of the real-time clock on the CPU4 can be set from the user program by means of this command.

**Command format:**

\[
\text{TIME:=}Ni.Nj.Nn
\]

**NOTE**

On execution of the command, a check is carried out to ensure that the values for the time are legal. If this is not the case, error E532 "Value outside range" is set.

**Example**

1. \[ N1:=14 \]
2. \[ N2:=50 \]
3. \[ N3:=23 \]
4. \[ \text{TIME:=}N1.N2.N3 \]

Sets the time to 14:50:23
3.54 **G5xx – Text and value output via the current data channel**

<table>
<thead>
<tr>
<th>Data channel 0</th>
<th>LC display on the IEF console of PA-CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data channel 1</td>
<td>1st serial port of PA-CONTROL (COM 1)</td>
</tr>
<tr>
<td>Data channel 2</td>
<td>2nd serial port of PA-CONTROL (COM 2)</td>
</tr>
<tr>
<td>Data channel 3</td>
<td>3rd serial port of PA-CONTROL (COM 3)</td>
</tr>
<tr>
<td>Data channel 4</td>
<td>4th serial port of PA-CONTROL (COM 4)</td>
</tr>
</tbody>
</table>

**Introduction:**
Texts, register contents, statuses of inputs, outputs and flags can be output via the current data channel during the program run (automatic mode) by means of this command group.

The following data channels can be used:

- Data channel 0: LC display on the IEF console of PA-CONTROL
- Data channel 1: 1st serial port of PA-CONTROL (COM 1)
- Data channel 2: 2nd serial port of PA-CONTROL (COM 2)
- Data channel 3: 3rd serial port of PA-CONTROL (COM 3)
- Data channel 4: 4th serial port of PA-CONTROL (COM 4)

**NOTE**
The information about the current data channel is assigned to each task (to each parallel run) and must be initialised before using the commands.

If a serial port has already been initialised and communication is active, switch-over is only possible on specification of the data channel number (e.g. G500.1). The receive and transmit buffers remain unchanged.
3.54.1 G500 - Selection of the data channel / Initialisation of the ports

G500.

**G500.(Channel number).(Baud rate).(Data format).(Handshake)**

**Description:**
The current data channel can be selected by means of the "G500." command. Serial ports can also be initialised (baud rate, number of data bits, parity,...).

If the serial port is selected with initialisation (e.g. G500.1.6.1.0), the receive buffer of the serial port is also deleted. If, on the other hand, the serial port is selected without initialisation (e.g. G500.1), the receive buffer of the serial port remains unchanged.

See the list for the meaning of the transfer parameters.

**NOTE**
Outputs to the LCD display of PA-CONTROL are only practical without CR & LF.

If the G500 command is executed after a G533.n command, the end criterion is deleted!

**Command format:**
G500.No.Baud rate.Data format.Handshake
G500.No
G500.0

**Example**
See examples of the following commands.
Meaning of the transfer parameters:

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>LC display on the PAC front plate is active</td>
</tr>
<tr>
<td>1</td>
<td>Port 1 is activated (COM 1)</td>
</tr>
<tr>
<td>2</td>
<td>Port 2 is activated (COM 2)</td>
</tr>
<tr>
<td>3</td>
<td>...</td>
</tr>
</tbody>
</table>

Baud rate:

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Baud Rate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>110 baud</td>
</tr>
<tr>
<td>2</td>
<td>300 baud</td>
</tr>
<tr>
<td>3</td>
<td>1200 baud</td>
</tr>
<tr>
<td>4</td>
<td>2400 baud</td>
</tr>
<tr>
<td>5</td>
<td>4800 baud</td>
</tr>
<tr>
<td>6</td>
<td>9600 baud</td>
</tr>
<tr>
<td>7</td>
<td>19200 baud</td>
</tr>
<tr>
<td>8</td>
<td>38400 baud (currently not available)</td>
</tr>
</tbody>
</table>

Data format:

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Data Format</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>8 data bits, 1 stop bit, no parity</td>
</tr>
<tr>
<td>2</td>
<td>7 data bits, 1 stop bit, no parity</td>
</tr>
<tr>
<td>3</td>
<td>7 data bits, 1 stop bit, even parity</td>
</tr>
<tr>
<td>4</td>
<td>7 data bits, 1 stop bit, odd parity</td>
</tr>
<tr>
<td>5</td>
<td>8 data bits, 1 stop bit, even parity (from V4.49)</td>
</tr>
<tr>
<td>6</td>
<td>8 data bits, 1 stop bit, odd parity (from V4.49)</td>
</tr>
<tr>
<td>7</td>
<td>8 data bits, 2 stop bits, no parity (from V4.55 K)</td>
</tr>
<tr>
<td>8</td>
<td>7 data bits, 2 stop bits, no parity (from V4.55 K)</td>
</tr>
<tr>
<td>9</td>
<td>7 data bits, 2 stop bits, even parity (from V4.55 K)</td>
</tr>
<tr>
<td>10</td>
<td>7 data bits, 2 stop bits, odd parity (from V4.55 K)</td>
</tr>
<tr>
<td>11</td>
<td>8 data bits, 2 stop bits, even parity (from V4.55 K)</td>
</tr>
<tr>
<td>12</td>
<td>8 data bits, 2 stop bits, odd parity (from V4.55 K)</td>
</tr>
</tbody>
</table>

Handshake:

<table>
<thead>
<tr>
<th>Nr.</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Hardware handshake (CTS) inactive</td>
</tr>
<tr>
<td>1</td>
<td>Hardware handshake (CTS) active</td>
</tr>
</tbody>
</table>
### 3.54.2 G501 - Clear the display

**G501**

**Description:**
The G501 command clears the display of the current display medium and places the cursor in the top left corner (1st column, 1st line).

**NOTE**
Before using the G501 command, switch over to the current display medium with the "G500." command.
This command is only meaningful for output to the display!

**Command format:**
G501

**Example**

1. G11.0 ;Switch off run displays
2. G500.0 ;Switches the G500 commands over to the current display medium (LC display or serial port in the case of "Simulation of the front plate")
3. G501 ;Clears the display and places the cursor in the top left corner (1st column, 1st line)
4. G510.Hello ;Outputs the text "Hello" to the current display medium
5. END
3.54.3 G502 - Clear up to end of line

Description:
The G502 command clears the display on the current display medium from the current cursor position to the end of the line. The cursor position remains unchanged.
The command can be used to delete the value previously displayed during the output of register values, in order to output the new register value.

NOTE
Before using the G502 command, switch over to the current display medium with the "G500." command.
This command is only appropriate for output to the display!

Command format:
G502

Example

1  G11.0                     ;Switch off run displays
2  G500.0                   ;Switches the G500 commands over to the current display medium (LC display or serial port in the case of "Simulation of the front plate")
3  G501                     ;Clears the display and places the cursor in the top left corner (1st column, 1st line)
4  N1:=234                  
5  $LOOP                    
6  G503.1.2                 ;Place cursor in the 1st column of the 2nd line
7  G510.VALUE N1            ;Place cursor in the 11th column of the 2nd line
8  G503.11.2                ;Clear from the cursor position to the end of the line
9  G502                     ;Output the value of N1 to the display (left-justified)
10 G520.N1                  
11 T100                     
12 DEC.N1 LOOP              
13 END
3.54.4 G503/G504 - Position the cursor

**G503/504**

\[ G50X.(\text{Column}).(\text{Line}) \]

**Description:**
The G503 command is used to position the cursor in the display, so that characters are output to the correct position in the display.

The difference from the G503 command is that the blinking cursor is fixed blinking at the specified position in the display for a keyboard input. The cursor is deactivated on input of the first character or by the G504 command with the parameters 0 (G504.0.0).

The LC display is divided into 40 columns and two lines.

**NOTE**
Before using the G503 command, switch over to the current display medium with the "G500." command.

This command is only appropriate for output to the display!

**Command format:**
- G503.n.m
- G503.Nn.m
- G503.n.Nm
- G503.Nn.Nm

- G504.n.m
- G504.Nn.m
- G504.n.Nm
- G504.Nn.Nm
Example

1. G11.0 ;Switch off run displays
2. G500.0 ;Switches the G500 commands over to the current display medium (LC display or serial port in the case of "Simulation of the front plate")
3. N23:=5
4. N35:=2
5. G501 ;Cleans the display and places the cursor in the top left corner (1st column, 1st line)
6. G503.4.1 ;Places the cursor in the 4th column of the 1st line
7. G510.HELLO ;Outputs the text "HELLO" to the current cursor position on the current display medium
8. G503.N23.N35 ;Places the cursor in the 5th column of the 2nd line, see values of N23 and N35
10. G510.OPERATOR ;Outputs the text "OPERATOR" to the current cursor position on the current display medium
11. END
3.54.5 G510 - Text output

**G510.**

**G510.(Text)**

**Description:**
G510. outputs the text via the selected serial port or to the display.

**NOTE**
No CR or LF are sent at the end of the text. No further command may be in this line after G510.

The text begins after "G510" and ends with the end of line (Conclude entry with ENTER in the program editor).

**Command format:**
G510.Text

**Example**

1  G500.0 ;Selects the LC display of PA-CONTROL
2  G501 ;Clears the display and places the cursor in the 1st column, 1st line
3  G510.Machine Error! ;Outputs the text "Machine Error" to the current display medium, the LC display
3   END
3.54.6 G511 - Text output

G511.(Text)

Description:
G511. outputs texts via the selected serial port and sends a CR and LF after the text and is therefore not appropriate for output to the PA-CONTROL display.

NOTE
The transfer is ended with CR and LF. No further command may be in this line after G511.

The text begins after "G511" and ends with the end of line (Conclude entry with ENTER in the program editor).

Command format:
G511.Text

Example
1  G500.1.6.1.0 ;Initialisation of the current port (COM1)
2  G511.Machine Error! ;Outputs the text "Machine Error!" via the serial port COM1 and concludes the transfer with CR and LF
3  END
3.54.7 G512 - Output of control characters

G512.

G512.(0-255)

Description:
G512. interprets n as the nth character (ordinal number) in the ASCII table and outputs the corresponding character via the serial port. The corresponding character can also be selected by the content of an integer register or by the indirect addressing of an integer register.
All numbers from 0 to 255 are permissible.

NOTE No CR or LF are sent after the character.

Command format:
G512.n
G512.Ni
G512.N!i

Application:
Output of control characters to a printer. Please refer to the ASCII table of your printer for the meaning of the individual characters.

Example

<table>
<thead>
<tr>
<th></th>
<th>G500.2.6.1.0</th>
<th>;Initialisation of the current port (COM2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>G512.7F</td>
<td>;Outputs the command „DEL“ via the serial port</td>
</tr>
<tr>
<td>3</td>
<td>G512.36</td>
<td>;Outputs the symbol &quot;$&quot; via the serial port</td>
</tr>
<tr>
<td>4</td>
<td>G512.4</td>
<td>;Outputs 04Hex (EOT) via the serial port</td>
</tr>
<tr>
<td>5</td>
<td>END</td>
<td></td>
</tr>
</tbody>
</table>
3.54.8 G515 - Text output

**G515.(Line).(PTX File)**

**Description:**
G515 outputs a text, the contents of the line i of a program file (*.PTX) to the current data channel. This command is suitable for using stored messages or user tips in different program sections.

**NOTE**
The file must be a *.PTX type file and available during the runtime of the program. If the respective text line cannot be found (number is greater than the number of lines in the file), the last program line is specified.

**Command format:**
G515.i.Name
G515.Ni.Name

**Example**

**Program: MESSAGE.PTX**

1 Machine ready
2 Malfunction part feed
3 Part magazine empty
4 Text not defined
5 END

**Program: EXAMPLE.PNC**

1 G500.0 ;Initialization, LC display is current data channel
2 G501 ;Clears the display and places the cursor in the 1st position of the 1st line
3 G515.2.MESSAGE ;Outputs the 2nd line of the program "MESSAGE.PTX" to the current data channel
4 N5:=3
5 G503.1.2 ;Places the cursor in the 1st position of the 2nd line
6 G515.N5.MESSAGE ;Outputs the 3rd line of the program "MESSAGE.PTX" to the current data channel
6 END
3.54.9 G520 - Output of value

**G520.** *(Operand).*(Field width).*(Decimal places)*

**Description:**
G520. determines the status of the operand (register, input, output, flag) and outputs the corresponding value via the current serial port or to the display.

**NOTE**
The output can be formatted or unformatted.

**Command format unformatted:**
G520.Rn
G520.Nn
G520.In
G520.On
G520.Mn
G520.Sn

**Command format formatted** (with leading blank characters):
G520.Nn.m
G520.Rn.m.i
G520.Rn.m.0

**Limits for the formatting**

**Formatted :** *(G520.Nn.m)*

- **Field width**
  - max. = 20

**Formatted: (G520.Rn.m.i)**

- **Decimal places**
  - min. = 0
  - max. = 6

- **Field width**
  - max. = 20

**NOTE**
No CR or LF are sent in the output.
Output values for inputs, outputs and flags

<table>
<thead>
<tr>
<th>Operand</th>
<th>Status</th>
<th>Output (ASCII characters)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>hot</td>
<td>1</td>
</tr>
<tr>
<td>Input</td>
<td>cold</td>
<td>0</td>
</tr>
<tr>
<td>Output</td>
<td>set</td>
<td>1</td>
</tr>
<tr>
<td>Output</td>
<td>reset</td>
<td>0</td>
</tr>
<tr>
<td>Flag</td>
<td>set</td>
<td>1</td>
</tr>
<tr>
<td>Flag</td>
<td>reset</td>
<td>0</td>
</tr>
</tbody>
</table>

Example

1.  $O10:=1$  
   Set output 10
2.  $O20:=0$  
   Reset output 20
3.  $M25:=1$  
   Set flag 25
4.  $M35:=0$  
   Reset flag 35
5.  $R2:=546.067$  
   Load the number 546.06 to real number register R2
6.  $N3:=34$  
   Load the number 34 to integer register N3
7.  $G500.2.6.1.0$  
   ;Initialisation of the current port (COM2)
8.  $G520.I7$  
   ;Outputs the status of input 7 via the current data channel (0 or 1, depending on the status)
9.  $G520.O10$  
   ;Outputs the status of output 10 via the current data channel (output 1)
10. $G520.O20$  
    ;Outputs the status of output 20 via the current data channel (output 0)
11. $G520.M25$  
    ;Outputs the status of flag 25 via the current data channel (output 1)
12. $G520.M35$  
    ;Outputs the status of flag 35 via the current data channel (output 0)
    ;Outputs the content of N3, formatted, right-justified, (10 places with eight leading characters) via the current data channel
14. $G520.R2$  
    ;Outputs the content of R2, left-justified unformatted, via the current data channel
15. $G520.R2.10.2$  
    ;Outputs the content of R2, right-justified and formatted, via the current data channel (field 10 characters wide with two decimal places)
16. $G520.R2.10.0$  
    ;Outputs the content of R2 right-justified and formatted via the current data channel (field 10 characters wide with no decimal places)
17. $END$
3.54.10 G521 - Output of value

G521.

G521.(Operand).(Field width).(Decimal places)

Description:
G521. determines the status of the operand (register, input, output, flag) and outputs the corresponding value via the current serial port or to the display.

NOTE
The output can be formatted or unformatted.

Command format unformatted:
G521.Rn
G521.Nn
G521.In
G521.On
G521.Mn
G521.Sn

Command format formatted (with leading blank characters):
G521.Nn.m
G521.Rn.m.i
G521.Rn.m.0

Limits for the formatting
Formatted : (G521.Nn.m)

<table>
<thead>
<tr>
<th>Field width</th>
<th>max. = 20</th>
</tr>
</thead>
</table>

Formatted: (G521.Rn.m.i)

<table>
<thead>
<tr>
<th>Decimal places</th>
<th>min. = 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Field width</td>
<td>max. = 20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Field width</th>
<th>max. = 20</th>
</tr>
</thead>
</table>

NOTE
CR and LF are sent in addition to the actual value.
Output values for inputs, outputs and flags

<table>
<thead>
<tr>
<th>Operand</th>
<th>Status</th>
<th>Output (ASCII characters)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Input</td>
<td>hot</td>
<td>1</td>
</tr>
<tr>
<td>Input</td>
<td>cold</td>
<td>0</td>
</tr>
<tr>
<td>Output</td>
<td>set</td>
<td>1</td>
</tr>
<tr>
<td>Output</td>
<td>reset</td>
<td>0</td>
</tr>
<tr>
<td>Flag</td>
<td>set</td>
<td>1</td>
</tr>
<tr>
<td>Flag</td>
<td>reset</td>
<td>0</td>
</tr>
</tbody>
</table>

**Example**

1. `O10:=1`  Set output 10
2. `O20:=0`  Reset output 20
3. `M25:=1`  Set flag 25
4. `M35:=0`  Reset flag 35
5. `R2:=546.067`  Load the number 546.06 to real number register R2
6. `N3:=34`  Load the number 34 to integer register N3
7. `G500.2.6.1.0`  ;Initialisation of the current port (COM2)
8. `G521.I7`  ;Outputs the status of input 7 via the current data channel (0 or 1, depending on the status as well as CR and LF)
9. `G521.O10`  ;Outputs the status of output 10 via the current data channel (output 1 as well as CR and LF)
10. `G521.O20`  ;Outputs the status of output 20 via the current data channel (output 0 as well as CR and LF)
11. `G521.M25`  ;Outputs the status of flag 25 via the current data channel (output 1 as well as CR and LF)
12. `G521.M35`  ;Outputs the status of flag 35 via the current data channel (output as well as CR and LF)
13. `G521.N3.10`  ;Outputs the content of N3 right-justified and formatted, (10 places with eight leading characters sowie CR und LF) via the current data channel
14. `G521.R2`  ;Outputs the content of R2 as well as CR and LF left-justified and unformatted via the current data channel
15. `G521.R2.10.2`  ;Outputs the content of R2 right-justified and formatted via the current data channel (field 10 characters wide with two decimal places as well as CR and LF)
16. `G521.R2.10.0`  ;Outputs the content of R2 right-justified and formatted via the current data channel (field 10 characters wide with no decimal places as well as CR and LF)
17. `END`
3.55 G5?? Text- / Value transfer from the current data channel

G5??...

Introduction:
Registers, the status of outputs, flags or character strings can be read via the actual data channel during the program run (automatic mode) by means of this command group.

3.55.1 G531 - Value transfer

G531.

G531.(Operand) (Label)

Description:
G531. waits for characters to be received at the current serial port. The reception is considered to be terminated with the transmission of CR. The received characters are further processed according to the operand.

Time monitoring is not implemented in the G531 command. If required, the commands G401, G421, G422 and G423 can be used for this purpose.

If a transmission error (parity error,...) or an illegal character string is detected during reception, the program is continued at the label.

Command format:
G531.Rn Label
G531.Nn Label
G531.On Label
G531.Mn Label
G531.Sn Label
### Example

1. **G500.2.6.1.0**
   
   ;Initialisation of the current port

2. **G531.R2 ERROR**
   
   ;Waits for the reception of a message at the current serial port and assigns the information to register R2. If there is a transmission error, the program is continued at the label "ERROR"

3. **G531.M3 ERROR**
   
   ;Waits for the reception of a message at the current serial port and sets flag 3 according to the received information

4. **G531.S2 ERROR**
   
   ;Waits for the reception of a message at the current serial port and assigns the information to character buffer S2 (in this case the global character buffer). If there is a transmission error, the program is continued at the label "ERROR"

5. **JMP END**

6. **$ERROR**
   
   ;Error message program section

7. ..................

8. ..................

9. **$END**

10. **END**

### Operand	| Status	| Output (ASCII characters)
---|---|---
Output	| Set output	| 1
Output	| Reset output	| 0
Flag	| Set	| 1
Flag	| Set	| 1
Flag	| Reset	| 0
Real number register	| 546.09	| 546.09
Integer register	| 34	| 34
3.55.2 G532 - Character transfer to the local character buffer S0

G532.

G532.(End criterion).(Label)

Description:
The G532.n command receives characters (max. 80) from the current serial port until the end criterion "n" is received or an error occurred. The characters are stored in local character buffer S0. The end criterion "n" is a number greater than 0 and less than 255 and represents the corresponding ASCII character. If an error occurs during the transmission, the transfer is cancelled and the program continues at the label.

Possible errors:
• Parity error, overrun error, framing error during the character transmission
• Character buffer full

NOTE The program stays in the program line (in the following example line 2) until the end criterion has been detected or a transmission error occurs.

Command format:
G532.n label

Example

1   G500.2.6.1.0 ;Initialisation of serial port COM2
2   G532.13 U_ERROR ;Receives characters at the serial port and stores them in local character buffer S0 until the end criterion 13 = 0Dhex = CR has been received
3   N3:=COPY.2.5 C_ERROR ;Convert the next 5 characters starting from the 2nd character in the character buffer to a number and store it in integer register 3
4
5   JMP END
6   $U_ERROR
7   SUB Error_1
8   JMP END
9   $C_ERROR
10  SUB Error_2
11  $END
12  END
3.55.3 G533 - Character transfer in the background

G533. (End criterion)

Description:
The G533.n command receives characters in the background (max. 80) from the current serial port until the end criterion "n" has been received or an error occurred. The characters remain in the receive buffer of the serial port.

The end criterion "n" is a number greater than 0 and less than 255 and represents the corresponding ASCII character.

The addition "in the background" means that PA-CONTROL starts a system routine which accepts the characters. The next command of the program is subsequently executed.

The command Ni:=CHN can be used to check whether the character string was received in full.

Possible errors: (see also Section 3.55.5 page 221):
- Parity error, overrun error, framing error during the character transmission
- Character buffer full

The characters can be copied from the receive buffer of the serial port to the local character buffer by means of the command "S0:=CHN".

NOTE
If communication is set-up via the serial port, data loss may occur.

G510. Please send
... ; Warning: If the communication partner transmits immediately, data may be lost!
G533.13 ; PAC is ready for reception from this point

A better solution is:
G533.13 ; PAC is ready for data reception
G510. Please send! ; PAC is ready to receive

Command format:
G533.n
Example

1. \texttt{G500.2.6.1.0} ; Initialisation of serial port COM2
2. \texttt{G533.13} Receives characters in the background at the serial port and stores them in the receive buffer until the end criterion 13 = \text{0D} \text{hex} = \text{CR} is received
3. \texttt{$\$\text{LOOP}$}
4. \texttt{SUB WORK}
5. \texttt{N1:=CHN} ; Character string received completely?
6. \texttt{CASE.JMP.N1} ; Evaluation of the transfer
7. \texttt{(CHAR\_RCD)} ; End criterion was received
8. \texttt{(U\_ERROR)} ; Error during the transfer of a character
9. \texttt{(P\_FULL)} ; Character buffer is full and the end criterion has not yet been received
10. \texttt{ELSE LOOP} ; End criterion has not yet been received
11. \texttt{$\$\text{CHAR\_RCD}$}
12. \texttt{S0:=CHN} ; Copy content of the receive buffer of the current data channel to the local character buffer S0
13. \texttt{N3:=COPY.2.5 C\_ERROR} ; Convert the next 5 characters starting from the 2nd character in the character buffer to a number and store it in integer register 3
14. \texttt{JMP END}
15. \texttt{$\$U\_ERROR$}
16. \texttt{O16:=1} ; Error display on
17. \texttt{I16.1} ; Error acknowledged
18. \texttt{O16:=0} ; Error display off
19. \texttt{JMP END}
20. \texttt{$\$P\_FULL$}
21. \texttt{G510.BUFFER\ FULL} ; Output of the error message
22. \texttt{$\$\text{END}$}
23. \texttt{END}
3.55.4 G534 - Character transfer in the background

G534.(No. of characters)

Description:
The G534.n command receives characters in the background (max. 80) from the current serial port. The characters remain in the receive buffer of the serial port.
The end criterion "n" is a number greater than 0 and less than 81 and represents the number of characters to be received.
The addition "in the background" means that PA-CONTROL starts a system routine which accepts the characters. The next command of the program is subsequently executed.
The command Ni:=CHN can be used to check whether the character string was received in full.

Possible errors: (see also Section 3.55.5 page 221):
- Parity error, overrun error, framing error during the character transmission
- Character buffer full

The characters can be copied from the receive buffer of the serial port to the local character buffer by means of the command "S0:=CHN".

Command format:
G534.n
Example

1  G500.2.6.1.0 ;Initialisation of serial port 2
2  G534.4 ;Receives 4 characters in background at the serial port and stores them in the receive buffer
3  $LOOP
4  SUB WORK
5  N1:=CHN ;Character string received completely?
6  CASE.JMP.N1 ;Evaluation of the transfer
7  (CHAR_RCD) ;4 characters were received
8  (U_ERROR) ;Error during the transfer of a character
9  (P_VOLL) ;Character buffer is full
10 ELSE LOOP ;End criterion has not yet been received
11 $ZEI_DA
12 S0:=CHN ;Copy content of the receive buffer of the current data channel to local character buffer S0
13 N3:=COPY.1.4 C_ERROR ;Convert the next 5 characters starting from the 2nd character in the character buffer to a number and store it in integer register 3
14  JMP END
15  $U_ERROR
16  SUB Error_1
17  JMP END
18  $P_FULL
19  SUB Error_2
20  $END
21  END
3.55.5 CHN - Check whether character transfer in the background has been completed

CHN

Description:
The receive status of a G533 or G534 is retrieved by means of the Nn:=CHN command. Branching can then be executed in the program depending on the result. The result in the integer register is interpreted as follows:

<table>
<thead>
<tr>
<th>Content of Ni</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>End criterion not yet received</td>
</tr>
<tr>
<td>1</td>
<td>End criterion received</td>
</tr>
<tr>
<td>2</td>
<td>Transmission error at the serial port (parity, …)</td>
</tr>
<tr>
<td>3</td>
<td>Receive buffer full, but end criterion not yet received</td>
</tr>
</tbody>
</table>

Command format:
Nn:=CHN
3.56  Data transfer to and from the local character buffer

Characters can be received via ports and transferred to the local character buffer. Dependent on the port and the transmission protocol, the received characters, texts or numbers can have various representational forms. The following commands are available for further processing these characters in PA-CONTROL:

- POS,
- COPY
- GET,
- GETI.

However, it may be necessary to combine the data which is stored in registers or flags in PA-CONTROL into one message and then send it via a port. The following commands are provided for this:

- PUT,
- PUTI
- S0:="Text".

3.56.1  Sn - Copying character strings

Sn:= Si

Description:

Strings are used to store up to 80 ASCII characters which are received e.g. via a serial port. Each process can access the global character buffers S1 to S16 (strings). In addition, each process has its own local character buffer S0.

The contents of S0 can be copied to a global character buffer by means of the command Sn:=S0. It is also possible to copy from a global character buffer to the local character buffer S0.

Command format:

S0:=Sn
Sn:=S0
3.56.2 S0 - Copying character strings

S0:= CHN

Description:
Strings are used to store up to 80 ASCII characters which are received e.g. via a serial port. Each process can access the global character buffers S1 to S16 (strings). In addition, each process has its own local character buffer S0.

The local character buffer S0 is used for processing and analysing character strings (POS, COPY, etc.). The command S0:=CHN copies the contents of the serial receive buffer of the current data channel to the local character buffer. The end criterion for the transfer is the character 0.

The two command formats differ in that the command S0:=CHN is used to copy the contents of the entire serial receive buffer, in the other case, a defined number of characters is copied.

Command format:
S0:=CHN
S0:=CHN.i
Sn:=S0
### Example

1. G500.2.6.1.0 ;Initialisation of serial port 2
2. G533.13 ;Receives characters in the background at the serial port and stores them in the character buffer until end criterion 13 = 0Dhex = CR has been received
3. $LOOP
4. SUB WORK
5. N1:=CHN ;Character string received completely?
6. CASE.JMP,N1 ;Evaluation of the transfer
7. (CHAR_REC) ;End criterion has been received
8. (U_ERROR) ;Error in the transfer of a character
9. (P_FULL) ;Character buffer is full and the end criterion has not yet been received
10. ELSE LOOP ;End criterion has not yet been received
11. $(CHAR_REC)
12. S0:=CHN ;Copy content of the receive buffer of the current data channel to the local character buffer S0
13. N3:=COPY.2.5 C_ERROR ;Convert the first 5 characters starting from the 2nd character from the local character buffer S0 to a number and store this number in integer register 3
14. S8:=S0 ;Copy contents of the local character buffer S0 to the global character buffer S8
15. S0:=S16 ;Copy the contents of the global character buffer S16 to the local character buffer S0
16. JMP END
17. $U_ERROR
18. SUB Error_1
19. JMP END
20. $P_FULL
21. SUB Error_2
22. $END
23. END
Example

1 G500.2.6.1.0 ;Initialisation of serial port 2
2 G533.13 ;Receives characters in the background at the serial port and stores them in the character buffer until end criterion 13 = 0Dhex = CR has been received
3 $LOOP
4 SUB WORK
5 N1:=CHN ;Character string received completely?
6 CASE.JMP.N1 ;Evaluation of the transfer
7 (CHAR_REC) ;End criterion has been received
8 (U_ERROR) ;Error in the transfer of a character
9 (P_FULL) ;Character buffer is full and the end criterion has not yet been received
10 ELSE LOOP ;End criterion has not yet been received
11 $(CHAR_REC) ;Copy contents of the receive buffer of the current data channel to the local character buffer S0
12 N3:=COPY.2.5 C_ERROR ;Convert the first 5 characters starting from the 2nd character from the character buffer to a number and store this number in integer register 3
13 $U_ERROR
14 SUB Error_1
15 JMP END
16 $P_FULL
17 SUB Error_1
18 $END
19 END
20
3.56.3 POS - Search for character position in local character buffer S0

**POS.** *(Search position).*(Character 0-255)*

*Description:*
A specific character ("m") can be searched for in the local character buffer S0 by means of the **POS** command.

"m" is a number greater than 0 and less than 255 and represents the corresponding ASCII character.

The search in the local character buffer S0 begins at the position "n". If the character is found, the position (1 to n) is stored in the integer register. If the character is not found, the value "0" is stored in the integer register.

**Command format:**

Ni:=POS.n.m

Ni:=POS.Nn.m

Ni:=POS.Nn.Nm

**Example**

1. G500.2.6.1.0 ;Initialisation of serial port 2
2. G532.13 U_ERROR ;Receives characters in the background at the serial port and stores them in the local character buffer S0 until the end criterion 13 = 0Dhex = CR has been received
3. N2:=POS.1.35 ;Search in local character buffer S0 for the character 35= 23hex = # and store the position in N2
4. M1:=N2<1 ;Character found?
5. G21 M1.1 Z_NO ;No!
6. N3:=COPY.N2.5 C_ERROR ;Convert the next 5 characters starting from the value of N2 from the local character buffer S0 to a number and store this number in integer register 3

7. JMP END
8. $U_ERROR
9. SUB Error_1
10. JMP END
11. $C_ERROR
12. SUB Error_2
13. JMP END
14. Z_NO
15. SUB Error_3
16. $END
17. END
3.56.4 Copy - Convert characters from the local character buffer

**COPY**

*Copy.(Start position).(No. of characters)[[Label]]*

**Description:**
A maximum of 80 characters from the local character buffer (S0) can be converted to a number and stored in an integer or real number register by means of the command COPY.

COPY starts from the "nth" character and considers the following "m" characters provided that there are characters available in the local character buffer (S0). If there are less than "m" characters available, the conversion process is ended without an error.

If an error occurs, the conversion process is aborted. The register is unchanged and the program is continued at the label.

**Possible errors:**
- "n" is less than 1 or greater than 80
- "m" less than 1 or greater than 80
- No legal characters were found in the conversion process.

<table>
<thead>
<tr>
<th>Legal characters</th>
<th>N Register</th>
<th>R Register</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leading blank characters</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>0...9</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>Decimal separator &quot;.&quot;</td>
<td>X</td>
<td>X</td>
</tr>
<tr>
<td>E</td>
<td>-</td>
<td>X,</td>
</tr>
</tbody>
</table>

If there is at least one number between 0...9 in front, or two numbers separated by the decimal separator ".". At least one number (1...9) must follow the E!

(Number format for the R register)

**NOTE**
Leading blank characters are included for determination of the "nth" character!

**Command formats:**
- Ni:=COPY.n.m Label
- Ri:=COPY.n.m Label
- Ni:=COPY.Nn.Nm Label
- Ri:=COPY.Nn.Nm Label
- Ni:=COPY.N!n.N!m Label
- Ri:=COPY.N!n.N!m Label
Example

1  G500.2.6.1.0 ;Initialisation of serial port 2
2  G532.13 U_ERROR ;Receives characters in background at the serial port and stores them in the local character buffer S0 until the end criterion 13 = 0Dhex = CR has been received
3  N3:=COPY.2.5 C_ERROR ;Converts the next 5 characters starting from the 2nd character from the local character buffer S0 to a number and stores this number in integer register 3
4  JMP END
5  $U_ERROR
6  SUB Error_1
7  JMP END
8  $C_ERROR
9  SUB Error_2
10 $END
11 END

An S0 string is also copied to another string (S0 ... S16) by means of the COPY command.

Command formats:
Sn:=COPY.(Start position).(No. of characters)

Examples:
S1:=COPY.4.5
S3:=COPY.N5.N7
S5:=COPY.N!3.N!9

Example

1  G500.2.6.1.0 ;Initialisation of serial port 2
2  G532.13 U_ERROR ;Receives characters in background at the serial port and stores them in the local character buffer S0 until the end criterion 13 = 0Dhex = CR has been received
3  S3:=COPY.2.5 ;Converts the next 5 characters starting from the 2nd character from the local character buffer S0 to a number and stores this number in string 3
4  JMP END
5  $U_ERROR
6  SUB Error_1
7  $END
8  END
3.56.5 Merging strings

The commands of the following command group are used to create a string or to merge two strings.

**Description:**
- Two strings are merged to form one string
- A string and a number are merged to form one string

No format specifications can be made for the conversion of numbers. N registers are simply converted, R registers are converted by means of the function "ConvertFloatToASCII" (see G500 commands).

If the command results in a string which is longer than 80 characters, system error "E754, String too long" is displayed.

**Command format:**

\[
\begin{align*}
S_i &= S_n + S_m \\
S_i &= S_n + N_m \\
S_i &= S_n + R_m \\
S_i &= N_m + S_n \\
S_i &= R_m + S_n \\
\end{align*}
\]

**Examples:**

\[
\begin{align*}
S_0 &= S_1 + S_3 \\
S_0 &= S_0 + S_7 \\
S_4 &= S_3 + S_4 \\
S_1 &= S_3 + N_6 \\
S_3 &= S_0 + R_7 \\
S_0 &= N_4 + S_0 \\
\end{align*}
\]

3.56.6 Get the length of a string

**LENGTH**

The length of a string, i.e. the current number of characters, can be determined by means of the following command.

**Command format:**

\[
N_i = \text{LENGTH}.S_n
\]

**Examples:**

\[
\begin{align*}
N_6 &= \text{LENGTH}.S_0 \\
N_{17} &= \text{LENGTH}.S_4 \\
\end{align*}
\]
3.56.7 Write the content of a register (number) to a string

A numerical value is converted to a string by means of this command.
- The number of places and the number of decimal places is optional (as in G520)
- If the numerical value is less than the number of specified places, leading blank characters are output
- If the number is greater than the number of places, the string becomes longer

**Command format:**

\[ Sn := Ni \]
\[ Sn := Ni.(No. \ of \ places) \]
\[ SN := Ri \]
\[ SN := Ri.(No. \ of \ places).(No. \ of \ decimal \ places) \]

**Examples:**

\[ S0 := N7 \]

\[ S0 := N8.5 \]
\[ S2 := N!7.N9 \]
\[ S4 := R2 \]
\[ S5 := R5.9.2 \]
\[ S4 := R6.N3.N7 \]
\[ S7 := R1.N!3.N!7 \]
3.56.8 GET - Transfer the content of the local character buffer S0 to an integer register

GET

GET.(Start Position).(No. of Characters)

Description:
Up to four characters can be transferred to an integer register from the local character buffer S0 by means of the command GET.
GET begins from the "Start Position" specified in the command and considers the "Number of Characters" defined in the command.

Possible errors:
- "Start Position" is less than 1 or greater than 80
- "No. of Characters" is less than 1 or greater than 4
- "Start Position" + "No. of Characters" is greater than 80

Command format:
Ni:=GET.n.m
Ni:=GET.Nn.Nm
Ni:=GET.Nn.Nm

Example

1  N1:=GET.1.2 ;Transfer two characters of the local character buffer S0 starting from the first place and store them in N1

n-1
n  END

Local character buffer S0 | Byte 1 | Byte 2 | Byte 3 | Byte 4 | ....... | Content of N1
-------------------------------|--------|--------|--------|--------|----------|------------------
3E8_{hex} | 03 | E8 | 00 | 00 | ....... | 1000
64_{hex} | 00 | 64 | 00 | 00 | | 100
3.56.9 GETI - Transfer the content of the local character buffer S0 to an integer register in INTEL format

GETI

\[ Ni := \text{GETI}(\text{Start position}).(\text{No. of characters}) \]

**Description:**

Up to four characters can be transferred to an integer register from the local character buffer S0 by means of the command GETI.

GETI begins from the "Start Position" specified in the command and considers the "No. of Characters" defined in the command.

**Possible errors:**

- "Start Position" is less than 1 or greater than 80
- "No. of Characters" is less than 1 or greater than 4
- "Start Position" + "No. of Characters" is greater than 80

**Application:**

Process data objects can be transferred in INTEL format via the CANopen bus. The stored information has to be converted for further processing.

**Command format:**

\[ Ni := \text{GETI}.n.m \]
\[ Ni := \text{GETI}.Nn.Nm \]
\[ Ni := \text{GETI}.N!n.N!m \]

**Example**

1. \[ N1 := \text{GETI}.1.2 \]

   ;Transfer two characters of the local character buffer S0 starting from the first place from the INTEL format and store them in N1

   \begin{tabular}{c|ccccc}
   \hline
   Local character buffer S0 & Byte 1 & Byte 2 & Byte 3 & Byte 4 & Content of N1 \\
   \hline
   3E8_{hex} & E8 & 03 & 00 & 00 & 1000 \\
   64_{hex}  & 64 & 00 & 00 & 00 & 100 \\
   \hline
   \end{tabular}
3.56.10 PUT – Transfer the content of an N register to the local character buffer

**PUT**

*PUT.(Start position).(No. of characters).(N Register)*

**Description:**
The contents of an N register can be transferred byte-by-byte to the local character buffer by means of the command PUT. The characters are stored in the local character buffer starting from the "Start Position". The number of characters specified as the "No. of Characters" is stored.

**Possible errors:**
- "Start Position" is less than 1 or greater than 80
- "No. of Characters" is less than 1 or greater than 4
- "Start Position" + "No. of Characters" is greater than 80

**Command format:**
PUT.n.m.Ni
PUT.Nn.Nm.Ni
PUT.N!n.N!m.Ni

**Example:**

1  N1:=100
2  PUT.1.4.N1 ;Load the number 100 to integer register 1
       ;Write the contents of N1 from the first place as a binary number with a length of 4 places (4 bytes) to the local character buffer S0

\[
\begin{array}{cccccc}
\text{Local character buffer } & S0 & 1 & 2 & 3 & 4 \\
100_{\text{dec}} & \rightarrow & 64_{\text{hex}} & 00 & 00 & 00 & 64 \\
1000_{\text{dec}} & \rightarrow & 3E8_{\text{hex}} & 00 & 00 & 03 & E8 \\
100000_{\text{dec}} & \rightarrow & 186A0_{\text{hex}} & 00 & 01 & 86 & A0 \\
\end{array}
\]
3.56.11 PUTI - Transfer the content of an N register in the INTEL format to the local character buffer S0

**PUTI**

**PUTI.(Start position).(No. of characters).(N Register)**

**Description:**
The contents of an N register can be transferred byte-by-byte to the local character buffer by means of the command **PUTI**. The characters are stored in the local character buffer starting from the "**Start Position**". The number of characters specified as the "**No. of Characters**" is stored. The storage of numbers in the INTEL format is thereby considered.

**Possible errors:**
- "**Start Position**" is less than 1 or greater than 80
- "**No. of Characters**" is less than 1 or greater than 4
- "**Start Position**" + "**No. of Characters**" is greater than 80

**Application:**
The fact that process data objects may be transferred in the INTEL format via the CANopen bus must be considered when accessing the S0 string.

**Command format:**

- PUTI.n.m.Ni
- PUTI.Nn.Nm.Ni
- PUTI.N!n.N!m.Ni

**Example**

1  \( N1 := 100 \)  ;Load the number 100 to integer register 1  
2  PUTI.1.4.N1  ;Write the contents of N1 from the first place as a binary number with a length of 4 places (4 bytes) to the local character buffer S0

\( n-1 \)  .....  
\( n \)  END

**Example:**

<table>
<thead>
<tr>
<th>Local character buffer S0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
</tr>
</thead>
<tbody>
<tr>
<td>( 100_{\text{dec}} \rightarrow 64_{\text{hex}} )</td>
<td>64</td>
<td>00</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>( 1000_{\text{dec}} \rightarrow 3E8_{\text{hex}} )</td>
<td>E8</td>
<td>03</td>
<td>00</td>
<td>00</td>
</tr>
<tr>
<td>( 100000_{\text{dec}} \rightarrow 186A0_{\text{hex}} )</td>
<td>A0</td>
<td>86</td>
<td>01</td>
<td>00</td>
</tr>
</tbody>
</table>
3.57  Character transfer from the keyboard

3.57.1  G540 - Check if a key has been actuated

G540

**G540.(N Register)**

*Description:*
The G540 command checks if a key on the front plate of PA-CONTROL has been actuated.
If this is the case, the key code is stored in the N register. If this is not the case the N register is set to 0.
The keyboard can be polled while a program section is being processed by means of this command, in order to react to operator requests.

**NOTE**
The command only functions with the front plate of PA-CONTROL (LC display, membrane keyboard).
Please refer to the "PAC key code" table in the "Technical Appendix" chapter for the key codes.

**Command format:**
G540.Nn

**Example**

1  G11.0 ;Switch off run displays
2  G500.0 ;Switches over the G500 commands to the current display medium (LC display or serial port in case of "Simulation of front plate")
3  G501 ;Clear display
4  N1:=0
5  $LOOP
6  O1:=1 T10
7  O1:=0 T10
8  G540.N1 ;Checks if a key has been actuated and stores the key code in N1; otherwise N1 is set to 0
9  M1:=N1>0 ;Has a key been actuated?
10  G21 M1.0 LOOP ;No!
11  G503.4.1 ;Place cursor in the 4th column of the 1st line
12  G502 ;Delete up to end of line
13  G510.N1
14  G512.32
15  G520.N1
16  JMP LOOP
17  END
3.57.2 G541 - Get a character from the keyboard

G541

G541.(N Register)

Description:
The G541 command waits until a key on the front panel of PA-CONTROL has been actuated and stores the key code in the N register.
The G541 command can be used to create individual menus in PA-CONTROL.

NOTE

The command only functions with the front plate of PA-CONTROL (LC display, membrane keyboard).

Please refer to the "PAC key code" table in the "Technical Appendix" chapter for the key codes.

Command format:
G541.Nn

Example

1  G11.0 ;Switch off run displays
2  G500.0 ;Switches over the G500 commands to the current display medium (LC display or serial port in case of "Simulation of front plate")
3  G501 ;Clear display
4  N1:=0
5  $LOOP
6  G541.N1 ; Waits until a key has been actuated and stores the key code in N1
7  G503.4.1 ; Place cursor in the 4th column of the 1st line
8  G502 ; Delete up to end of line
9  G510.N1
10  G512.32
11  G520.N1
12  JMP LOOP
13  END
3.57.3 G542 - Entry of a value via the keyboard

**G542.(Column).(Line).(Length).(Target)**

**Description:**
The G542.s.z.l.Ni command enables the operator to enter a value via the keyboard while the program is active.

The command opens an input field (see the chapter on "User interface"). The length and position of the input field are determined by the operators (parameters) of the command.

The system flag SM5 is polled to check whether the G542 command has been exited by means of the Enter key (i.e. transfer of the value) or by means of the ESC key (entry is cancelled and the previous value is reset).

**The following applies:**

<table>
<thead>
<tr>
<th>s</th>
<th>→ Stands for the column</th>
</tr>
</thead>
<tbody>
<tr>
<td>z</td>
<td>→ Stands for the line</td>
</tr>
<tr>
<td>l</td>
<td>→ Stands for the length of the entry field</td>
</tr>
<tr>
<td>Target</td>
<td>Register or character buffer (storage)</td>
</tr>
</tbody>
</table>

**Command format:**

- G542.s.z.l.Ni
- G542.s.z.l.Ri
- G542.s.z.l.Si *
- G542.Ns.Nz.Nl.Si *

*Si can be S0, S1 ...S16
Example

1  G11.0 ;Switch off run displays
2  G500.0 ;Switches over the G500 commands to the current display medium (LC display or serial port in case of “Simulation of front plate”)
3  G501 ;Clear display
4  $LOOP
5  G503.1.1
6  G510.Entry Target
7  G542.30.1.3.N1 ;Entry of the N register via the front plate. The input field is displayed in the 30th column of line 1 with a length of 3 characters
8  G501
9  G503.1.2
10 G510.Time for output 1 is
11 G512.32
12 G520.N1
13 G512.32
14 G510. * 10ms
15 O1:=1 TN1 O1:=0
16 JMP LOOP
17 END
3.58 G6?? - Image to register / Image from register

Introduction:
An image of the contents of registers can be transferred to outputs or flags by means of this command group during the automatic mode. An image of inputs or flags can also be loaded to a register.

General command structure:
G60?.Rn.m.i
G60?.Nn.m.i

<table>
<thead>
<tr>
<th>Rn / Nn:</th>
<th>Source / Target</th>
</tr>
</thead>
<tbody>
<tr>
<td>m:</td>
<td>First element of target / source address (corresponds to the least significant bit)</td>
</tr>
<tr>
<td>i:</td>
<td>Number of elements to / from which an image is to be created.</td>
</tr>
<tr>
<td></td>
<td>( 1 &lt;= i &lt;= 32)</td>
</tr>
</tbody>
</table>

NOTE  
\( m + i \) must be less than or equal to the maximum number of elements used.

The content of the register is used as follows:
- Decimal places are truncated
- An image of a max. of 32 bits is created

Various formats are available:
- Binary image
- BCD image
3.58.1 G600 - Binary notation to outputs

G600.(Register).(1st output).(No. of outputs)

Command format:
G600.Rn.m.i
G600.Nn.m.i

Example 1

1  R3:=10.7
2  G600.R3.5.8 ;Content of R3 is represented in binary code with 8 bits starting from output 5
3  END

Status of outputs after execution of the above commands:

Output 5 → reset (0)
Output 6 → set (1)
Output 7 → reset (0)
Output 8 → set (1)
Output 9 → reset (0)
Output 10 → reset (0)
Output 11 → reset (0)
Output 12 → reset (0)

Example 2

1  N3:=18
2  G600.N3.2.4 ;Content of N3 is represented in binary code with 4 bits starting from output 2
3  END

Status of outputs after execution of the above commands:

Output 2 → reset (0)
Output 3 → set (1)
Output 4 → reset (0)
Output 5 → reset (0)
3.58.2 G601 - BCD notation to outputs

G601.(Register).(1st output).(No. of outputs)

Command format:
G601.Rn.m.i
G601.Nn.m.i

Example 1
1  R3:=10.7
2  G601.R3.5.8 ;Content of R3 is represented in BCD code with 8 bits starting from output 5
3  END

Status of outputs after execution of the above commands:

Output 5 → reset (0)
Output 6 → reset (0)
Output 7 → reset (0)
Output 8 → reset (0)
Output 9 → set (1)
Output 10 → reset (0)
Output 11 → reset (0)
Output 12 → reset (0)

Example 2
1  N3:=12
2  G601.N3.2.4 ;Content of N3 is represented in BCD code with 4 bits starting from output 2
3  END

Status of outputs after execution of the above commands:

Output 2 → reset (0)
Output 3 → set (1)
Output 4 → reset (0)
Output 5 → reset (0)
3.58.3  G602 - Binary notation to flags

**G602.**

*G602.(Register).(1st flag).(No. of flags)*

**Command format:**

G602.Rn.m.i  
G602.Nn.m.i

**Example 1**

1  
R3:=11.3

2  
G602.R3.4.8  ;Content of R3 is represented in binary code with 8 bits starting from flag 4

3  
END

Status of the flags after execution of the above commands:

- Flag 4  →  set (1)
- Flag 5  →  set (1)
- Flag 6  →  reset (0)
- Flag 7  →  set (1)
- Flag 8  →  reset (0)
- Flag 9  →  reset (0)
- Flag 10  →  reset (0)
- Flag 11  →  reset (0)

**Example 2**

1  
R3:=65535

2  
G602.R3.17.32  ;Content of R3 is represented in binary code with 32 bits starting from flag 17

3  
END

Status of flags 17 to 49 after execution of the above commands:

- Flags 17 – 32  →  are set
- Flags 33 – 49  →  are reset
Example 3

1  N3:=11
2  G602.N3.1.4 ;Content of N3 is represented in binary code with 4 bits starting from flag 1
3  END

Status of the flags after execution of the above commands:

Flag 1  →  set (1)
Flag 2  →  set (1)
Flag 3  →  reset (0)
Flag 4  →  set (1)
3.58.4 G603 - Inputs in binary format to register

**G603.**

_G603.(Register). (1st input).(No. of inputs)_

Command format:
G603.Rn.m.i
G603.Nn.m.i

**Example 1**

1. G603.R3.2.8
   ;Inputs 2 to 9 (2=least significant bit) are interpreted as a binary number and the value transferred to register 3

2. END

Status of inputs after execution of the above commands:

<table>
<thead>
<tr>
<th>Input</th>
<th>Status</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>cold (0)</td>
<td>Value = 1</td>
</tr>
<tr>
<td>3</td>
<td>hot (1)</td>
<td>Value = 2</td>
</tr>
<tr>
<td>4</td>
<td>cold (0)</td>
<td>Value = 4</td>
</tr>
<tr>
<td>5</td>
<td>hot (1)</td>
<td>Value = 8</td>
</tr>
<tr>
<td>6</td>
<td>cold (0)</td>
<td>Value = 16</td>
</tr>
<tr>
<td>7</td>
<td>cold (0)</td>
<td>Value = 32</td>
</tr>
<tr>
<td>8</td>
<td>cold (0)</td>
<td>Value = 64</td>
</tr>
<tr>
<td>9</td>
<td>hot (1)</td>
<td>Value = 128</td>
</tr>
</tbody>
</table>

This is equivalent to the number "10001010" in binary code and the value 138 in register 3.
3.58.5 G604 - Flags in binary format to register

**G604.**

\[ \text{G604.(Register).(1st flag).(No. of flags)} \]

Command format:
- G604.Rn.m.i
- G604.Nn.m.i

**Example 1**

1. \texttt{G604.R3.2.8} ;Flags 2 to 9 (2=least significant bit) are interpreted as a binary number and the value transferred to register 3

2. \texttt{END}

Status of flags after execution of the above commands:

<table>
<thead>
<tr>
<th>Flag</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>3</td>
<td>2</td>
</tr>
<tr>
<td>4</td>
<td>4</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>32</td>
</tr>
<tr>
<td>8</td>
<td>64</td>
</tr>
<tr>
<td>9</td>
<td>128</td>
</tr>
</tbody>
</table>

This is equivalent to the number "10001111" in binary code and the value 143 in register 3.
3.59 Arithmetic operations

Description:
Registers can be directly and indirectly addressed for value assignments and fundamental arithmetic operations.

Direct addressing:
- A number (1-1024), which identifies the desired register, follows the letter R(N).

Indirect addressing:
- An "!" and then a number (1-1024), which refers to the register whose content identifies the desired register, follows the letter R(N).

The following generally applies for assignments:

<table>
<thead>
<tr>
<th>Target</th>
<th>Source</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
<td>Constant (K)</td>
</tr>
<tr>
<td></td>
<td>→ Register</td>
</tr>
<tr>
<td></td>
<td>→ Absolute position of an axis (only for real number registers)</td>
</tr>
</tbody>
</table>

The following generally applies for arithmetic operations:

<table>
<thead>
<tr>
<th>Target operand</th>
<th>Source operand 1</th>
<th>Operation</th>
<th>Source operand 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Register</td>
<td>Register</td>
<td>+ - * /</td>
<td>Unsigned constant Register</td>
</tr>
</tbody>
</table>

NOTE
The result is dependent on the operation and the sign character of the register contents.

K (constant) is used both for the description of integers, as well as for real numbers. Only the permissible number type can be used for the real application.

The source operands remain unchanged. The exception is if the target operand is also the source operand.

2 Examples

R1:=5
R2:=3
R3:= R1+R2
According to this, the registers have the following values: R1=5, R2=3, R3=8.
R1:=R1+R2
According to this, the registers have the following values: R1=8, R2=3.
3.59.1 Rn/Nn:= Load register

<table>
<thead>
<tr>
<th>Rn:=</th>
<th>Nn:=</th>
</tr>
</thead>
</table>

**Description:**
Values can be loaded into the registers by means of these commands.

**NOTE** No spaces are permitted in the command.

**Command format:** (selection from the possible combinations)

<table>
<thead>
<tr>
<th>Nn:=K</th>
<th>Nln:=K</th>
<th>Nn:=Nm</th>
<th>Nn:=Rm</th>
<th>Nn:=Ai</th>
<th>Nn:= SNn</th>
</tr>
</thead>
<tbody>
<tr>
<td>Rn:=K</td>
<td>Rln:=K</td>
<td>Rn:=Rm</td>
<td>Rn:=Nm</td>
<td>Rn:=Ai</td>
<td>Rn:=SRn</td>
</tr>
</tbody>
</table>

**Example**

1. \( R2:=45 \) ; Load real number register 2 with 45
2. \( R3:=R2 \) ; Load real number register 3 with value of real number register 2
3. \( R!2:=100.79 \) ; Load real number register 45 with 100.79
5. \( R5:=A1 \) ; Load real number register 5 with absolute position of the A1 axis
6. \( N7:=87 \) ; Load integer register 7 with value 87
7. \( N65:=N7 \) ; Load integer register 65 with value from integer register 7
8. \( N1:=R45 \) ; Load integer register 1 with value from real number register 45, value is rounded
9. \( R25:=N65 \) ; Load real number register 25 with value from integer register 65
10. \( N2:=4 \) ; Load integer register 2 with 4
11. \( N!2:=17 \) ; Load integer register 4 \( (N2=4) \) with 17
12. \( N24:=SN12 \) ; Transfer of the number of the defective axis
13. \( R12:=SR32 \) ; Transfer of the absolute position of axis A2 to real number register R11
14. END
3.59.2 Rn:=Ran - Load register with axis parameters

\[ \text{Rn:=RAi} \]

**Description:**
The axis parameters can be loaded into the registers by means of this command. The following applies here:

\[
\begin{array}{|c|c|}
\hline
\text{RA1.i} & \rightarrow \text{"i" axis parameter axis 1} \\
\text{RA2.2} & \rightarrow \text{"2" axis parameter axis 2} \\
\vdots & \rightarrow \\
\text{RAn.i} & \rightarrow \text{"i" axis parameter axis n} \\
\hline
\end{array}
\]

The selection of the desired parameter "i" ensues through input as per the following table: It includes a selection of the most important axis parameters. See Chapter 5, Axis parameters

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>→</td>
<td>Traversing speed</td>
</tr>
<tr>
<td>2</td>
<td>→</td>
<td>Reference speed</td>
</tr>
<tr>
<td>3</td>
<td>→</td>
<td>Manual speed</td>
</tr>
<tr>
<td>4</td>
<td>→</td>
<td>Creep speed</td>
</tr>
<tr>
<td>5</td>
<td>→</td>
<td>Start/stop speed</td>
</tr>
<tr>
<td>6</td>
<td>→</td>
<td>Acceleration</td>
</tr>
<tr>
<td>7</td>
<td>→</td>
<td>Gear factor</td>
</tr>
<tr>
<td>8</td>
<td>→</td>
<td>min. traversing range</td>
</tr>
<tr>
<td>9</td>
<td>→</td>
<td>max. traversing range</td>
</tr>
</tbody>
</table>

**Command format:**
Rn:=RAm.i

**Example**

1  G90.A0 ;Approach to reference point A1 axis
2  G25.A1 ;Set traversing speed to 1000AE/s
3  FA1:=1000 ;Traverse to position A1:=100
4  A1:=100 ;Load real number register 34 with traversing speed of the A1 axis
5  R34:=RA1.1 ;Set traversing speed to parameter default
6  FA1:=R34 ;Traverse to position A1:=10
7  A1:=10
8  END
3.59.3 Request for the current line number of a program

**Ni:=LINE**

**Description:**

The line number of a program can be determined by means of the LINE command, in order to subsequently branch back to this point in the program by means of the command "JMP-LINE...".

The command "Ni:=LINE" determines the line number of its own program, i.e. the line in which this command is located.

The commands "Ni:=LINE.name" or Ni:=LINE.Si search for the program name in all TASKS and their sub-program levels.

If the program is not active, the result is 0. However, if the program does not exist, error message "E513=Program does not exist" is output.

**NOTE**

Only S1 to S16 can be used as a string!

**Command format:**

Ni:=LINE
Ni:=LINE.Name
Ni:=LINE.Si
3.59.4 Addition

Command format:
- \( R_n := R_m + K \)
- \( R^n := R_m + K \)
- \( R_n := R_m + R_i \)
- \( R^n := R^n + R_i \)
- \( N_n := N_m + K \)
- \( N^n := N_m + N_i \)
- \( N^n := N^n + N_i \)

Example

1. \( R_2 := R_2 + 57 \); Adds 57 to the content of \( R_2 \)
2. \( R_2 := R_4 + R_3 \); Adds the content of \( R_4 \) and the content of \( R_3 \) and assigns the result to \( R_2 \)
3. \( R_1 := 5 \)
4. \( R_5 := 100 \)
5. \( R_{24} := R_1 + R_2 \); Adds the content of \( R_5 \) and the content of \( R_2 \) and assigns the result to register \( R_{24} \). The content of \( R_1 = 5 \). The content of \( R_5 \) is thus accessed by indirect addressing. The content of \( R_5 \) and \( R_2 \) is added and the result assigned to \( R_{24} \). \( R_1, R_5 \) and \( R_2 \) remain unchanged.
6. \( N_3 := N_3 + 87 \); Adds 87 to the content of \( N_3 \)
7. \( N_3 := N_3 + N_5 \); Adds the content of \( N_5 \) to the content of \( N_3 \)
8. END
### 3.59.5 Subtraction

Command format:

- \( R_n := R_m - K \)
- \( R!_n := R_m - K \)
- \( R_n := R!_m - R_i \)
- \( R!_n := R!_m - R_i \)
- \( N_n := N_m - K \)
- \( N_n := N_m - N_i \)
- \( N!_n := N!_m - N_i \)

**Example**

1. \( R2 := R2 - 57 \); Subtracts 57 from the content of \( R2 \)
2. \( R2 := R2 - R3 \); Subtracts the content of \( R3 \) from the content of \( R2 \)
3. \( R1 := 5 \)
4. \( R5 := 100 \)
5. \( R24 := R!1 - R2 \); Subtracts the content of \( R2 \) from the content of \( R5 \) and assigns the result to register \( R24 \). The content of \( R1 \) is 5. The content of \( R5 \) is thus accessed by indirect addressing. The content of \( R2 \) is now substracted from this value and the result assigned to \( R24 \). \( R1 \), \( R5 \) and \( R2 \) remain unchanged.
6. \( N23 := N23 - 99 \); Subtracts 99 from the content of \( N23 \)
7. \( N23 := N!23 - N4 \); Subtracts the content of \( N4 \) from the content of \( N23 \)
8. \( END \)
3.59.6 Multiplication

Command format:
Rn:=Rm*K
Rln:=Rm*K
Rn:=Rm*ki
Rn:=Rm*ri
Nn:=Nm*K
Nn:=Nm*ki
Nln:=Nm*ri

Example
1  R2:=R2*4 ;Multiplies the content of R2 by 4 and assigns the result to R2
2  R4:=R2*R3 ;Multiplies the content of R2 by the content of R3 and assigns the result to R4.
3  R1:=5
4  R5:=100
5  R24:=R1*R2 ;Multiplies the content of R5 by the content of R2 and assigns the result to register R24.
6  N2:=N2*4 ;Multiplies the content of N2 by 4 and assigns the result to N2.
7  N2:=N3*N6 ;Multiplies the content of N3 by the content of N6 and assigns the result to N2.
8  END
3.59.7 Division

NOTE A system error occurs in the case of division by 0. The message "E005" is output to the display.

Command format:
- \( R_n := R_m / K \)
- \( R!_n := R_m / K \)
- \( R_n := R_m / R_i \)
- \( R!_n := R!_m / R_i \)
- \( N_n := N_m / K \)
- \( N!_n := N_m / K \)
- \( N_n := N!_m / N_i \)
- \( N!_n := N!_m / N_i \)

Example

1. \( R_2 := R_2 / 3 \) ;Divides the register content of \( R_2 \) by 3 and assigns the result to \( R_2 \)
2. \( R_4 := R_2 / R_3 \) ;Divides the register content of \( R_2 \) by the register content of \( R_3 \) and assigns the result to \( R_4 \)
3. \( R_1 := 5 \)
4. \( R_5 := 100 \)
5. \( R_{24} := R_{1} / R_{2} \) ;Divides the register content of \( R_5 \) by the register content of \( R_2 \) and assigns the result to \( R_{24} \)
6. \( N_{34} := N_{34} / 2 \) ;Divides the content of \( N_{34} \) by 2 and assigns the result to \( N_{34} \), the result is rounded (if necessary)
7. \( N_{25} := N_{67} / N_{88} \) ;Divides the content of \( N_{67} \) by the content of \( N_{88} \) and assigns the result to \( N_{25} \), the result is rounded (if necessary)
8. END
Trigonometrical functions

\[ \text{SIN} \quad \text{COS} \quad \text{TAN} \]

Introduction:
The trigonometrical functions sine, cosine and tangent and their inverse functions arc sine, arc cosine and arc tangent are available. The value is input in degrees. The trigonometrical functions can be used in conjunction with R registers and constant values.

3.59.8 SIN/ASIN - Sine functions

**SIN / ASIN**

Command format:
- \( R_n:=\text{SIN}.K \)
- \( R_n:=\text{SIN}.R_i \)
- \( R_n:=\text{SIN}.R!i \)
- \( R_n:=\text{ASIN}.K \)
- \( R_n:=\text{ASIN}.R_i \)
- \( R_n:=\text{ASIN}.R!i \)

Example
1. \( R_{43}:=70 \)
2. \( R_{70}:=45 \)
3. \( R_{100}:=\text{SIN}.10 \); Write sine of the angle 10° to register R100 (R100=0.174)
4. \( R_{101}:=\text{SIN}.R_{43} \); Write sine of the angle 70° to register R101 (R101=0.940)
5. \( R_{102}:=\text{SIN}.R!{43} \); Write sine of the angle 45° to register R102 (R102=0.707)
6. \( R_{200}:=\text{ASIN}.R_{100} \); Write arc sine of the register content of R100 to register R200 (R200:=10)
7. END
3.59.9 COS/ACOS - Cosine functions

**COS / ACOS**

**Command format:** (selection from the possible combinations)

- \( R_n := \text{COS.K} \)
- \( R_n := \text{COS.Ri} \)
- \( R_n := \text{COS.R!i} \)
- \( R_n := \text{ACOS.K} \)
- \( R_n := \text{ACOS.Ri} \)
- \( R_n := \text{ACOS.R!i} \)

**Example**

1. \( R43 := 70 \)
2. \( R70 := -210 \)
3. \( R100 := \text{COS.10} \); Write cosine of the angle 10° to register R100 \( (R100 = 0.985) \)
4. \( R101 := \text{COS.R43} \); Write cosine of the angle 70° to register R101 \( (R101 = 0.342) \)
5. \( R102 := \text{COS.R!43} \); Write cosine of the angle -210° to register R102 \( (R102 = -0.866) \)
6. \( R200 := \text{ACOS.R101} \); Write arc cosine of the register content of R101 to register R200 \( (R200 = 70) \)
7. \( \text{END} \)
3.59.10 TAN/ATAN - Tangent functions

**NOTE**
The value range for tangent must be defined by the user himself. Non-permissible values (90° + x * 180°, for x = 0, 1, 2,...) result in an undefined register value.

**Command format:** (selection from the possible combinations)
- Rn:=TAN.K
- Rn:=TAN.Ri
- Rn:=TAN.Rli
- Rn:=ATAN.K
- Rn:=ATAN.Ri
- Rn:=ATAN.Rli

**Example**

1. R43:=70
2. R70:=-210
3. R100:=TAN.10; Write tangent of the angle 10° to register R100 (R100= 0.176)
4. R101:=TAN.R43; Write tangent of the angle 70° to register R101 (R101=2.747)
5. R102:=TAN.R!43; Write tangent of the angle -210° to register R102 (R102=-0.577)
6. R200:=ATAN.R102; Write arc tangent of the register content of R101 to register R200 (R200=-30)
7. END
3.59.11 SQRT - Root function

NOTE The numerical value being squared may not be negative! The numerical value for the calculation must always be taken from a register!

Command format:
Rn:= SQRT.Ri
Rn:= SQRT.R!i

Example
1 R43:=70
2 R70:=210
3 R101:=SQRT.R43 ;Write square root of the number 70 to register R101 (R101=8.367)
4 R102:=SQRT.R!43 ;Write square root of the number 210 to register R102 (R102=14.491)
5 END
3.59.12 INT - Integer component

INT

Command format:
Rn:=INT.Rn

Example
1  R43:=70
2  R70:=210.23445
3  R10:=INT.R70 ;Stores the integer component of real number register R70 in real number register R10 (Result: 210)

4  END
3.59.13 FRAC - Decimal component of a real number

**FRAC**

Command format:

Rn:=FRAC.Rn

**Example**

1  R43:=70
2  R70:=210.23445
3  R10:=FRAC.R70 ;Stores the decimal component of real number register R70 in real number register R10 (Result: 0.23445)
4  END
3.59.14 ABS - Absolute value of a real number / integer

**ABS**

**Command format:**

Rn:=ABS.Rn
Nn:=ABS.Nn

**Example**

1. \( R70:=-210.23445 \)
2. \( R10:=\text{ABS}.R70 \); Stores the absolute value of real number register \( R70 \) in real number register \( R10 \) (Result: \( \mid 210.23445 \mid \))
3. \( N23:=-452 \)
4. \( N11:=\text{ABS}.N23 \); Stores the absolute value of integer register \( N23 \) in integer register \( N11 \) (Result: \( \mid 452 \mid \))
5. \( \text{END} \)
3.59.15 Bit-by-bit processing of N registers

**AND, OR, EXCLUSIVE-OR**

The possibilities for the use of N registers have been extended by the following bit-by-bit operating functions:

<table>
<thead>
<tr>
<th>Function</th>
<th>Operator</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit-by-bit</td>
<td>AND &amp;</td>
</tr>
<tr>
<td>Bit-by-bit</td>
<td>OR</td>
</tr>
<tr>
<td>Bit-by-bit</td>
<td>EXCLUSIVE-OR ^</td>
</tr>
</tbody>
</table>

The commands have been extended for:

**Command formats**

- \( N_i := N_n \& N_m \)
- \( N_i := N_n \& \text{Constant} \)
- \( N_i := N_n \lor N_m \)
- \( N_i := N_n \lor \text{Constant} \)
- \( N_i := N_n \oplus N_m \)
- \( N_i := N_n \oplus \text{Constant} \)

**NOTE**

The N registers are stored in PA-CONTROL with 32 bits (signed long int). The bit-by-bit processing takes place for all 32 bits!

The following notation results from this:

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hex</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0000 0000</td>
<td>0000 0000 0000 0000 0000 0000 0000 0000 0000</td>
</tr>
<tr>
<td>1</td>
<td>0000 0001</td>
<td>0000 0000 0000 0000 0000 0000 0000 0000 0001</td>
</tr>
<tr>
<td>100</td>
<td>0000 0064</td>
<td>0000 0000 0000 0000 0000 0000 0000 0110 0100</td>
</tr>
<tr>
<td>-1</td>
<td>FFFF FFFF</td>
<td>1111 1111 1111 1111 1111 1111 1111 1111</td>
</tr>
<tr>
<td>-2</td>
<td>FFFF FFFE</td>
<td>1111 1111 1111 1111 1111 1111 1111 1110</td>
</tr>
</tbody>
</table>

**Example 1**

- \( N_2 := 6 \); Binary 0000 0000 0000 0000 0000 0000 0000 0010
- \( N_3 := 3 \); Binary 0000 0000 0000 0000 0000 0000 0000 0011
- \( N_{12} := N_2 \& N_3 \); Binary (Result=2) 0000 0000 0000 0000 0000 0000 0000 0010

**Example 2**

- \( N_4 := 3 \); Binary 0000 0000 0000 0000 0000 0000 0000 0011
- \( N_6 := 10 \); Binary 0000 0000 0000 0000 0000 0000 0000 1010
- \( N_3 := N_4 \lor N_6 \); Binary (Result=11) 0000 0000 0000 0000 0000 0000 0000 1011

**Example 3**

- \( N_4 := 3 \); Binary 0000 0000 0000 0000 0000 0000 0000 0011
- \( N_6 := 10 \); Binary 0000 0000 0000 0000 0000 0000 0000 1010
- \( N_3 := N_4 \oplus N_6 \); Binary (Result=9) 0000 0000 0000 0000 0000 0000 0000 1001
3.60 Comparison operations

> = < <>

**Introduction:**

Comparison operations can be performed between registers and between registers and constants. The result of the comparison operation (logical 0 or 1) is assigned to a flag. The flag is always processed according to the result.

<table>
<thead>
<tr>
<th>Result true (logical 1)</th>
<th>Flag is set (logical 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Result untrue (logical 0)</td>
<td>Flag is reset (logical 0)</td>
</tr>
</tbody>
</table>

In general, the following applies:

<table>
<thead>
<tr>
<th>Target</th>
<th>Operand 1</th>
<th>Operation</th>
<th>Operand 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flag</td>
<td>Register</td>
<td>&lt; = &gt;</td>
<td>Register</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Constant</td>
</tr>
</tbody>
</table>

**NOTE**

The operation "=" must be handled with the utmost of care. If limits of inputs or loop conditions are to be checked, the operations ">" or "<" are safer, because they also provide a clear result in the case of truncation problems.
3.60.1 Comparisons

Command format:

<table>
<thead>
<tr>
<th>Direct addressing</th>
<th>Indirect addressing</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mn:=Rm=K</td>
<td>M!n:=Rm=K</td>
</tr>
<tr>
<td>Mn:=Rm&gt;</td>
<td>M!n:=Rm&gt;K</td>
</tr>
<tr>
<td>Mn:=Rm&lt;</td>
<td>M!n:=Rm&lt;K</td>
</tr>
<tr>
<td>Mn:=Rm=Ri</td>
<td>M!n:=Rm=Ri</td>
</tr>
<tr>
<td>Mn:=Rm=Nm=</td>
<td>M!n:=Rm=Nm=</td>
</tr>
<tr>
<td>Mn:=Rm&gt;Ni</td>
<td>M!n:=Rm&gt;Ni</td>
</tr>
<tr>
<td>Mn:=Nn&lt;&gt;K</td>
<td></td>
</tr>
</tbody>
</table>

Example

1. M34:=R1=56.77 ; Flag 34 is set if the content of R1 is equal to 56.77, otherwise flag 34 is reset
2. M35:=R1>56.34 ; Flag 35 is set if the content of R1 is greater than 56.34, otherwise flag 35 is reset
3. M36:=R1<56 ; Flag 36 is set if the content of R1 is less than 56, otherwise flag 36 is reset
4. R2:=10
5. M12:=R!2=0 ; Flag 12 is set if the content of R10 is equal to 0, otherwise flag 12 is reset
6. M34:=N1=56 ; Flag 34 is set if the content of N1 is equal to 56, otherwise flag 34 is reset
7. M35:=N1>56 ; Flag 35 is set if the content of N1 is greater than 56, otherwise flag 35 is reset
8. M36:=N1<56 ; Flag 36 is set if the content of N1 is less than 56, otherwise flag 36 is reset
9. N37:=255
10. M!37:=N100=510 ; Flag 255 is set by the indirect addressing of flag 37 if the content of N100 is equal to 510, otherwise flag 100 is reset
11. END
3.60.2 Complex examples

Example 1
Load real number register 10 to 20 with 0 at program start (resetting of counters)

1  \( R1 := 10 \); R1 is used for the indirect addressing
2  \$ LOOP
3  \( R1 := 0.0 \); Assign the value to the register pointed to by the content of R1
4  \( R1 := R1 + 1 \); Next register
5  \( M1 := R1 < 20 \); All 10 registers (R10-R20) loaded with the value?
6  G21 M1.1 LOOP ; Jump if all registers have not yet been loaded
7  END

Example 2
Production process for a pallet with unequal distances

1  \( R11 := 1 \); Switch off run display
2  G11.0
3  G500.0 ; LC display is the current data channel
4  G501 ; Clear display
5  $READ
6  G503.1.1 ; Position cursor
7  G510. ENTER PALLET POS.; Text output "ENTER PALLET POS."
8  G542.25.1.6.R!11 ; Entry of the pallet position in the indirectly addressed register R11 via the front plate. The input field is displayed in the 1st line of the 25th column with a length of 6 characters
9  \( R11 := R11 + 1 \); Pointer to the next pallet position
10 \( M1 := R11 > 5 \); Have all pallet positions been read?
11 $NEXT
12 A1 := 100 ; Approach "get" position
13 SUB GET PART ; Pick up part
14 A1 := R11 ; Approach "drop" position
15 SUB DROP PART ; Drop part
16 \( R11 := R11 + 1 \); Position pointer to the next pallet position
17 \( M1 := R1 > 6 \); All pallet positions used?
18 G21 M1.0 NEXT ; No!

Program: GET PART
1 I1.1 ; Wait until part is in pick-up position
\( n \) END

Program: DROP PART
1 I7.1 ; Pallet available
\( N \) END
3.6.1 Logic operations

Introduction:

A logic operation must always start with an LD command and end with an OUT command. Only commands of logic operations may be applied within a logical chain (starting with LD, ending with OUT).

Logic operations are processed according to the following functional mechanism.

In order to be able to also apply the notation of the commands "Wait for the status of an element" (e.g. I3.1, M12.0) to logic operations, both the element, such as inputs, flags, system flags or outputs, and the number, as well as the desired status of the element were assigned to the LD, AND and OR commands. If the current status of the element (I, M, SM, O) corresponds to the status of the element for the logical command, processing is continued with a logical "1". After the command (LD, AND, OR) has been converted. Otherwise processing is continued with a logical "0".

The result of the logic operations LD, AND and OR is stored in the bit accumulator. In an LD command, the status of the bit-accu is first of all transferred to the bit accumulator stack (a depth of 7 bits) and the result of the check of the element is then stored in the bit accumulator.

The logic operations AND-LD and OR-LD link the bit accumulator and the bit accumulator stack and store the result in the bit accumulator.

The OUT command transfers the status of the bit accumulator to the corresponding element. If the status of the bit accumulator is logical "1", the output or the flag is set, otherwise it is reset. Multiple OUT commands can directly follow each other.

The programmer can invert the status of the bit accumulator and transfer it to the element, output or flag to be controlled by means of the NOT command, i.e. the logical status "1" leads to a RESET of the addressed element.

Each parallel run has its own bit accumulator and bit accumulator stack with a stack depth of 7 bits for complex logic operations.

![Fig. 9: General logic operation mechanism](image-url)
Command: LD

Function catalogue for logic operations

Check status of element

Status of element as noted in the command

Yes

No

Example:
LD I3.1

Passive path
Active path

Bit accumulator stack

Logik-operator
“AND”
“OR”
“XOR”

Bit accumulator stack

Ausgabe
O
M

Fig. 10: Logic operation mechanism for the command LD

Command: AND or OR

Function catalogue for logic operations

Check status of element

Status of element as noted in the command

Yes

No

Example:
AND M6.0
OR I4.1

Passive path
Active path

Bit accumulator stack

Logik-operator
“AND”
“OR”
“XOR”

Bit accumulator stack

Ausgabe
O
M

Fig. 11: Logic operation mechanism for the commands AND and OR
Command: AND-LD or OR-LD

Function catalogue for logic operations

<table>
<thead>
<tr>
<th>Check status of element</th>
<th>Status of element as noted in the command</th>
<th>Example:</th>
<th>Passive path</th>
<th>Active path</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>?</td>
<td>AND-LD OR-LD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>No</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Bit accumulator stack

7 6 5 4 3 2 1

Logik-operator
“AND”
“OR”
“XOR”

Bit accumulator stack

Output
O M

Fig. 12: Logic operation mechanism for the commands AND-LD and OR-LD

The logic operations AND-LD and OR-LD retrieve the last stored bit from the bit accumulator stack, i.e. bit 1 of the bit accumulator stack is moved to the logic operator. This bit is now linked with bit 0 of the bit accumulator stack. The result of this logic operation is then in bit 0 of the bit accumulator stack.

NOTE

When bit 1 is read from the bit accumulator stack, the complete bit accumulator stack is moved one position to the right.
Since a maximum of 7 bits can be stored in the bit accumulator stack, the commands AND-LD or OR-LD can only be practically applied a maximum 7 times in succession!
Command: OUT

Function catalogue for logic operations

<table>
<thead>
<tr>
<th>Check status of element</th>
<th>Status of element as noted in the command</th>
</tr>
</thead>
<tbody>
<tr>
<td>Yes</td>
<td>?</td>
</tr>
<tr>
<td>No</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit accumulator stack</th>
<th>Logik-operator &quot;AND&quot; &quot;OR&quot; &quot;XOR&quot;</th>
<th>Bit accumulator stack</th>
</tr>
</thead>
<tbody>
<tr>
<td>7 6 5 4 3 2 1</td>
<td></td>
<td>Bit 0</td>
</tr>
</tbody>
</table>

Example: OUT 07

---

Passive path
Active path

---

Fig. 13: Logic operation mechanism for the command OUT
3.61.1 LD/AND/OUT/NOT - Logic AND operation

LD, AND, OUT, NOT

Command format:

AND li.n
AND Oi.n
AND Mi.n
AND Mli.n
AND SMi.n

Application:

Creation of logic AND operations between inputs, flags, system flags and outputs. The result of the logic operation can be assigned to a flag or an output.

Example

1  LD I3.1
2  AND I4.0
3  OUT O6
4  NOT O7

Implementation as a function plan (FUP)

or as a contact plan (COP)

5  END
### 3.61.2 LD/OR/OUT/NOT - Logic OR operation

**Command format:**
- OR Ii.n
- OR Oi.n
- OR Mli.n
- OR Mi.n
- OR Smi.n

**Application:**
Setting up logic OR operations between inputs, flags, system flags and outputs. The result of the logic operation can be assigned to a flag or an output.

**Example**

1. LD I3.1
2. OR I4.0
3. OUT O6
4. NOT O7
5. END

---

Implementation as a function plan (FUP):

![Function Plan Diagram](image)

Implementation as a contact plan (COP):

![Contact Plan Diagram](image)
3.61.3 SET/RES - Supplemental instructions for logic operations

**SET / RES**

**Command format:**

- SET Mn
- RES Mn
- SET On
- RES On

**Application:**

Simpler handling of results of logic operations arising from logic commands.

**Example 1**

1. LD I3.1
2. OR I4.0
3. SET O6
4. END

Implementation as a function plan (FUP)

Implementation as a contact plan (COP)
Example 2

1. LD I3.1
2. OR I4.0
3. RES M6
4. SET M56
5. END

Implementation as a function plan (FUP)

or as a contact plan (COP)
### 3.61.4 EOR logic operation (Exclusive-OR)

**XOR**

**Command format:**
- XOR In.i
- XOR On.i
- XOR Mn.i
- XOR SMn.i

<table>
<thead>
<tr>
<th><strong>Example 1 with truth table</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>States</strong></td>
</tr>
<tr>
<td>LD I1.1</td>
</tr>
<tr>
<td>XOR I2.1</td>
</tr>
<tr>
<td>OUT O1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th><strong>Example 2 with truth table</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>States</strong></td>
</tr>
<tr>
<td>LD I1.1</td>
</tr>
<tr>
<td>XOR I2.1</td>
</tr>
<tr>
<td>XOR I3.1</td>
</tr>
<tr>
<td>OUT O1</td>
</tr>
</tbody>
</table>

**NOTE**

In the case of EOR logic operations with more than two variables, the EOR logic operation of the two first variables is first of all implemented and the result is then implemented with the next EOR logic operation (see the next example), and so on.
3.61.5 Multi-level logic AND operation

**AND-LD**

Command format:

**AND-LD**

**Example**

<table>
<thead>
<tr>
<th>Input</th>
<th>I3</th>
<th>I4</th>
<th>I5</th>
<th>I6</th>
<th>I7</th>
<th>I8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log. Status</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Implementation as a contact plan (COP)

Status of bit accumulation stack

<table>
<thead>
<tr>
<th>Bit</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>L D I3.1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>2</td>
<td>OR I4.0</td>
<td>≥1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>3</td>
<td>LD I5.1</td>
<td>≥1</td>
<td>1</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>4</td>
<td>OR I6.1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>5</td>
<td>LD I7.0</td>
<td>&amp;</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>6</td>
<td>AND I8.0</td>
<td></td>
<td>1</td>
<td>0</td>
<td>x</td>
<td>x</td>
<td>x</td>
<td>x</td>
</tr>
<tr>
<td>7</td>
<td>OR-LD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>AND-LD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>OUT O6</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>END</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Application:

Creation of complex (multi-level) logic AND operations between inputs, flags, system flags and outputs. The bit accumulator stack has a depth of 7 bits.

The result of the logic operation can be assigned to a flag or an output.
3.61.6  OR-LD - Multi-level logic OR operation

Command format:

OR-LD

Example

<table>
<thead>
<tr>
<th>Input</th>
<th>I3</th>
<th>I4</th>
<th>I5</th>
<th>I6</th>
<th>I7</th>
<th>I8</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log. Status</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

Implementation as a contact plan (COP)

Status of bit accumulation stack

1  LD I3.1
2  AND I4.0
3  LD I5.1
4  AND I6.1
5  LDI7.1
6  AND I8.1
7  OR-LD
8  OR-LD
9  OUT O6
10  END
Application:

Creation of complex (multi-level) logic OR operations between inputs, flags, system flags and outputs. The bit accumulator stack has a depth of 7 bits.

The result of the logic operation can be assigned to a flag or an output.
3.61.7 Complex logic operation

**Description:**

If a multi-level logic operation is executed, the result of a logic operation must be temporarily stored, since it is repeatedly overwritten in the bit accumulator stack. The bit accumulator stack has a depth of 8 bits.

The example shows a multi-level logic operation. If the programming is executed without buffering in flag 82, the bit accumulator stack is overwritten after the second LD command and the result of the logic operation is incorrect.

### Example

<table>
<thead>
<tr>
<th>Input</th>
<th>I4</th>
<th>I5</th>
<th>I14</th>
<th>I15</th>
<th>I24</th>
<th>I25</th>
<th>I34</th>
<th>I35</th>
</tr>
</thead>
<tbody>
<tr>
<td>Log. Status</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

**Implementation as a contact plan (COP)**

```
I5
I15
I4
I14
I25
I24
I35
I34
M82
```

**Status of bit accumulation stack**

<table>
<thead>
<tr>
<th>Bit</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
<td>X</td>
</tr>
</tbody>
</table>

1. **LD I5.1**
2. **OR I4.0**
3. **LD I15.1**
4. **OR I14.0**
5. **LD I25.1**
6. **OR I24.0**
7. **LD I35.1**
8. **OR I34.0**
9. **AND-LD**
10. **AND-LD**
11. **AND-LD**
12. **OUT M82**
13. **END**
3.62 Analog-to-digital converter

Description:
As an option, PA-CONTROL can be equipped with A-D converters (see A-D converter manual).

3.62.1 Get A-D values

**ADI**

Description:
The value of the A-D converter can be transferred directly. The user can transfer the bit value of the A-D converter to an integer register or transfer the result to a real number register parameterized to the full-scale deflection.

Command format:
Nn:=Adi
Rn:=Adi

Application:
Acquisition of temperatures, forces, speeds, etc.

**Example**

```
1 $BEGINNING
2 N100:=2250 ;Specify limit
3 $LOOP ;
4 N101:=AD1 ;The bit value is transferred to register N101.
5 M100:=N100<N101
6 G21 M100.1 LOOP
7 O8:=1 ;Error lamp on
8 I8.1 I8.0 ;Acknowledgement key for errors
9 O8:=0 ;Error lamp off
10 R1:=AD1 ;The value converted to the specified full-scale deflection (voltage which corresponds to a pressure value, temperature, etc.) is transferred to real number register R1

11 JMP BEGINNING
12 END
```
3.62.2 G18x – Acquisition of several A-D values

Introduction:
The G180 command group is used to acquire A-D converter values. The A-D converter values are retrieved from the A-D converter in accordance with the acquisition criteria position/path or time and stored in the N registers. The stored values can then be further processed.

3.62.3 G180 - A-D values synchronous to the axis movement

\[ G180.(Axis).(Position).(Resolution).(AD\ No).(1st\ N\ Register).(Number) \]

**Command format:**
G180.A1.R27.R31.5.100.25 (see the following table)

- **Axis** / e.g. A1: Axis which is monitored, always as a constant
- **Position** / e.g.:
  - **R Reg. 27**: The axis position from which the A-D values are acquired is stored in this R register. The information as to whether the acquisition is to be started if the position is larger or smaller is derived from the sign character of the resolution. An R register must be used.
- **Resolution** / e.g.:
  - **R Reg. 31**: The resolution, i.e. the path according to which the next A-D value is to be retrieved, is stored in this R register. The sign character of the resolution applies as a criterion as to whether the acquisition is to be started if the position is larger or smaller. An R register must be used.
- **AD No.** / e.g.:
  - **A-D converter 5**: Specifies the number (from 1 to 8) of the A-D converter, from which the values are retrieved.
- **1st N Register** / e.g.:
  - **N Reg. 100**: Specifies the number of the first N register, starting from which the A-D values are stored. The address of the N register is increased by 1 with each new measured value. The specification can ensue as a constant or via an N register.
- **Number** / e.g.:
  - **max. number = 25**: Specifies the maximum number of A-D values which are to be acquired. This number also specifies the number of integer registers required for storage of the measured values. The specification can ensue as a constant or via an N register.

**NOTE**
The number of the acquired A-D values is stored in the system N registers SN51 – 56 (SN51 for axis 1, SN52 for axis 2, etc.) and is available for polling, see the example on page 280.
Maximum sampling rate:

**NOTE**
- PA-CONTROL MP with A-D converter 2x : 0.2ms
- PA-CONTROL MP with A-D converter 8x : 1ms
- PA-CONTROL SINGLE : 2ms
- PA-CONTROL COMPACT : 2ms
- PA-CONTROL STEUER : 2ms

---

**Example**

2. R1:=10 ; Acquisition starting from axis position 10mm
3. R2:=0.5 ; An A-D value is to be acquired every 0.5mm
4. G180.A1.R1.R2.1.100.200 ; Starting from the 10mm position, acquire at the next positioning of axis 1, a maximum of 200 values every 0.5mm from A-D converter 1 and store these in the memory starting from N register 100
5. N1000:=SN51 ; Load system N register 51 to N register 1000
6. M55:=N1000>180 ; Compare contents of N Reg. 1000 with 180
7. G22 M55.1 ERROR ; Jump to UP ERROR if <SN51> >180
8. A1:=1000 ; Traverse to position 1000
9. END

**Program: ERROR**

2. O32:=1 ; Switch on error lamp
3. I32.1 ; Wait for acknowledgement
4. I32.0 ;
5. O32:=0 ; Switch off error lamp
6. END
3.62.4 G181 - Acquire A-D values in a defined time base

G181

\[(G181.\text{Time Base.AD no.Start Register.Number})\]

**Description:**
The A-D values are acquired at a specified interval.

**Command format:**
G181.5.1.100.50  

**Application:**
Acquisition of dynamic A-D values, e.g. torque values for screwing or force values for joining and pressing.

**Example 1**

1. G210.A1 ;Activates the start positioning mode for axis 1  
2. A1:=100 ;Approach position 100  
3. G230.1.A1.50 ;Wait until position >50.0 has been reached  
4. **G181.5.1.100.50** ;Values of A-D converter 1 are read at an interval of 5ms and stored from N register 100 to N register 149.  
5. G213.A1 ;Go to the normal positioning mode  
6. END
3.63 Digital-to-analog converter

Brief description:
As an option, PA-CONTROL can be equipped with up to 16 D-A converters on a CANopen bus. Only 12 bit converters are used. In each case, 4 D-A converters are combined to form a group (hardware module).

<table>
<thead>
<tr>
<th>DA 1</th>
<th>DA 2</th>
<th>DA 3</th>
<th>DA 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>DA 5</td>
<td>DA 6</td>
<td>DA 7</td>
<td>DA 8</td>
</tr>
<tr>
<td>DA 9</td>
<td>DA10</td>
<td>DA11</td>
<td>DA12</td>
</tr>
<tr>
<td>DA13</td>
<td>DA14</td>
<td>DA15</td>
<td>DA16</td>
</tr>
</tbody>
</table>

The entire group is addressed via the CANopen bus in one message frame and is assigned with permanent PDO addresses (PDO, Process Data Objects).

<table>
<thead>
<tr>
<th>Rx PDO</th>
<th>D-A converters</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>DA 1, DA 2, DA 3, DA 4</td>
</tr>
<tr>
<td>34</td>
<td>DA 5, DA 6, DA 7, DA 8</td>
</tr>
<tr>
<td>35</td>
<td>DA 9, DA10, DA11, DA12</td>
</tr>
<tr>
<td>36</td>
<td>DA13, DA14, DA15, DA16</td>
</tr>
</tbody>
</table>

This assignment is implemented by overwriting the assignment in the CAN.bus node, also referred to as variable mapping.

Example:
Murr CANOpen module "MBM-C" with
- Inputs (directly on bus node or DI 8),
- Outputs (DO 8/0,5A) and
- D-A converters (AO 4).

The bus node is set to node ID (CANaddress) 17 (see Chapter 6, Section on CANopen port).

<table>
<thead>
<tr>
<th>Inputs /Outputs / D-A converters</th>
<th>Tx/RxPDO(hex)</th>
<th>PA-CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Directly on bus node (8 inputs)</td>
<td>211</td>
<td>I1025 – I1032</td>
</tr>
<tr>
<td>DI 8 (8In)</td>
<td>211</td>
<td>I1033 – I1040</td>
</tr>
<tr>
<td>DO 8 / 0.5A</td>
<td>211</td>
<td>O1025 – O102</td>
</tr>
<tr>
<td>AO4 (first module)</td>
<td>221</td>
<td>DA1 – DA4</td>
</tr>
<tr>
<td>AO4 (second module)</td>
<td>222</td>
<td>DA5 – DA8</td>
</tr>
</tbody>
</table>

NOTE
If the automatic mode is exited, the D-A converters are set to the value 0!

Diagnosis
Diagnosis of the D-A converters is possible via the PA-CONTROL front plate or IEF control console (see Chapter 2, Section 2.7.4, “CANOpen”).

NOTE
The output values of the D-A converters are set to 0 each time this menu item is entered.

The input field is opened by means of the enter key and a new D-A value can be entered.
3.63.1 Output of D-A values

**DAi**

**Command format:**
DAi:=Nn  
DAi:=Rn

**Application:**
Output of analog values

**Example**

1  $BEGINNING
2  N2:=1024  ;Load the number 1024 to the N2 register
3  DA1:=N2  ;Output contents of N2 as an analog value, here the number 1024 results in an output voltage of 2.5V at a resolution of 12Bit/10V
4
5  R2:=5.2  ;Load the numerical value 5.2 to the real number register R2
6  DA2:=R2  ;Output contents of R2 as an analog value, here the numerical value 5.2 results in an output voltage of 5.2V at a scaling of 10V as a maximum output value of the converter

n  END
3.64 Commands for devices on the CANopen bus

3.64.1 General

All PA-CONTROL devices from version 4.60 are equipped with the CANopen interface. Various devices, which are divided into groups, can be connected to the CANopen bus. The devices are assigned with station addresses, i.e. their ID numbers, for identification.

<table>
<thead>
<tr>
<th>ID number</th>
<th>Device</th>
<th>Manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>ID 1</td>
<td>Driving axis 1 to driving axis 16</td>
<td>LV-servoTEC, PA-CONTROL MP</td>
</tr>
<tr>
<td>ID17</td>
<td>Digital input / output modules</td>
<td>Murr, Beckhoff, FESTO</td>
</tr>
<tr>
<td>ID49</td>
<td>Analog input / output modules</td>
<td></td>
</tr>
<tr>
<td>ID52</td>
<td>Any desired CANopen device</td>
<td>Examples</td>
</tr>
<tr>
<td>ID60</td>
<td>Reserve</td>
<td></td>
</tr>
<tr>
<td>ID63</td>
<td>4th control console to 1st control console</td>
<td>SÜTRON</td>
</tr>
</tbody>
</table>

Only devices as per the above table can be connected as devices with the ID numbers 1 to 48. These devices are automatically detected by PA-CONTROL during booting and are handled by the operating system.

Any desired CANopen device can be connected as device 49 to 52. Commands are made available for operating these devices in the automatic mode. These commands enable the implementation of communication in the respective application by means of PNC/PTX or PAB programs.

A diagnosis menu and facilities for checking and simple operation are available in the WINPAC program for the initial start-up and service.

**NOTE**
A control console can be connected as device 60 to 63. Communication ensues via PDOs directly by means of the operating system. Control consoles of SÜTRON & Co. are used.

**NOTE**
Always follow the information and specifications of the manufacturers of devices which are used on the CANopen bus in your particular case. This documentation also provides you with essential additional information on the commands described in the sections below.
The command set for CANopen devices consists of the following command groups:
- Network management commands
- Emergency commands
- SDO commands
- PDO commands

The command set is implemented by the command group G7xx.

### 3.64.2 Network management commands

The various operating statuses (see Fig. 14) of a device on the CANopen bus can be achieved by means of the 5 commands of this command group.

#### 3.64.2.1 Start Remote Node

**G701**

- **G701.n**
- **Command format:**
  
  G701.49
  
  G701.50
- **Description:**
  - The changeover from the operating state "Pre-Operational" to the "Operational" state is implemented by means of this command (see Fig. 14, path 3).
  - The master sends the NMT frame "Start Remote Node" via the CANopen bus to execute this command.
- **Example:**
  
  You will find an application for this command at the end of the section as part of a complex example.
3.64.2.2 Stop Remote Node

**G702**

**G702.n**

**Command format:**
G702.49
G702.50

**Description:**
- The changeover from the operating state "Operational" to the "Stopped" state (Fig. 14, Path 8) is implemented by means of this command.
- The master sends the NMT frame "Stop Remote Node" via the CANopen bus to execute this command.

**Example:**
You will find an application for this command at the end of the section as part of a complex example.

3.64.2.3 Reset Remote Node

**G703**

**G703.n**

**Command format:**
G703.49
G703.50

**Description:**
- The changeover from the operating state "Operational" to the "Initialisation" state (Fig. 14, Path 9) is implemented by means of this command. After a successful initialisation, the device independently switches to the "Pre-Operational" state.
- The master sends the NMT frame "Reset Remote Node" via the CANopen bus to execute this command.

**Example:**
You will find an application for this command at the end of the section as part of a complex example.
3.64.2.4 Enter Pre-Operational State

**G704**

**G704.n**

**Command format:**
- G704.49
- G704.50

**Description:**
- The changeover from the operating state "Operational" to the "Pre-Operational" state (Fig. 14, Path 4) is implemented by means of this command.
- The master sends the NMT frame "Enter-Pre-Operational-State" via the CANopen bus to execute this command.

**Example:**
You will find an application for this command at the end of the section as part of a complex example.

3.64.2.5 Reset Communication

**G705**

**G705.n**

**Command format:**
- G705.49
- G705.50

**Description:**
- The changeover from the operating state "Operational" to the "Initialisation" state (Fig. 14, Path 9) is implemented by means of this command. After a successful initialisation, the device independently switches to the "Pre-Operational" state.
- The master sends the NMT frame "Reset Communication" via the CANopen bus to execute this command.

**Example:**
You will find an application for this command at the end of the section as part of a complex example.
3.64.3 Functional monitoring of the devices

G711

G711.(ID).Nn (Label)

Command format:
G711.49.12 ERROR

Application:
The Node Guarding Protocol is used for the functional monitoring of the devices. In this connection, the device is to be addressed by PA-CONTROL, the CANopen master, at the same intervals by means of a "NODE GUARDING frame" and a message requested from the device.

This type of device monitoring is provided when the communication is processed only via PDOs after initialization. The maximum time interval between two requests to the device, also referred to as a Node Guard frame, results from the product of the monitoring time (Guard Time) and a factor (Life Time Factor). If one of these values is 0, the response monitoring is deactivated.

If the device is not addressed within the monitoring time, refer to the description of device for its reaction.

Description:
The G711 command requests a monitoring message from a selected device by means of a data request frame (remote frame). The answer is stored in the N register specified in the command. The content of this encoded message can be found in the manual of the device.

If no monitoring message is received from the device within 200ms, the PA-CONTROL program branches to the programmed label.

Example:
You will find an application for this command at the end of the section as part of a complex example.
3.64.4 Commands for checking device errors

Device error messages ("Emergency Messages") are initiated by internal device errors. They have a high priority, in order to ensure prompt processing.

A device error message has a prescribed structure:
- Error field with pre-defined error number (2 bytes)
- Error register (1 byte)
- Error category (1 byte)
- Additional information

The higher order byte of the error number specifies the error class, the lower order byte specifies the error number in the class. Error numbers from xx00h to xx7Fh are defined in the communication profile.

The error numbers from xx80h to xxFFh result from specifications of the manufacturers. The error categories allow the occurring errors to be classified according to their importance.

3.64.4.1 Check whether a device error message is present

<table>
<thead>
<tr>
<th>Byte</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>LowByte</td>
<td>HighByte</td>
<td>Error field, specified by the manufacturer</td>
<td>Error register (Object 1001)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Refer to the manual of the defective CANopen device for the possible content of the device error message.

**Command format:**

G721.49 EMERGENCY
G722.50 GRIPPER

**Example for the evaluation of errors:**

...  
N1:=GETI.1.2 ; Error code  
N2:=GETI.3.1 ; Error register (Object 1001hex)
3.64.5 Commands for processing service data objects

In the same way as in other familiar field bus systems, CANopen makes a distinction between two basic data transfer mechanisms, i.e. the high-speed exchange of short process data via so-called process data objects (PDOs) and the access to entries of the object data directory via service data objects (SDOs). The service data objects are primarily used for the transfer of parameters during the device configuration and for the transfer of lengthy data areas.

The service data objects are accessed with a variable length.

NOTE
In principle, a request frame and an answer frame are required for the reading or writing of a service data object. As a result, a device should be accessed in only one parallel run.

If this is disregarded, data errors may occur if the request frame comes from run 1 and the answer frame from run 2.

3.64.5.1 Read the content of a service data object (Initiate Domain Upload Rq)

Command format:

\[ G730.<ID>.<Index(hex)>.<Subindex>.<No. Data Bytes>.<Error>.<Read Data> <Label> \]

If an error occurs during the read operation, e.g. an incorrect index or incorrect subindex, the error code is stored in the specified integer register. The program branches to the \(<Label>\). If the command proceeds normally, the data is stored in the register for the read data and the program run continued with the next command.

If the request to read is not acknowledged within 1 second, PA-CONTROL branches after the label with error 9, "IEF time-out".

NOTE
The index is transferred as a hex number.

Example:

\[ G730.49.1000.0.4.N1.N2 ERROR_SDO \]

In this example command, the device definition (device type) of CANopen device 49 is read and stored in integer register N2. In the case of an error, the error would be stored in register N1 and the program continued at the label "ERROR_SDO".
3.64.5.2 Write the content of a service data object (Initiate Domain Upload Rq)

**G731**

Command format:

```
```

If an error occurs during the write operation, the error code is stored in the specified integer register. The program branches to the `<Label>`. If the command proceeds normally, the program run is continued with the next command.

If the request is not acknowledged within 1 second, PA-CONTROL branches after the label with error 9, "IEF time-out".

**NOTE** The index is transferred as a hex number.

Command format:

```
G731.49.100C.0.2.N1.N2 FEHLER_SDO
```

In this example, the monitoring time (Guard Time) of CANopen device 49 is overwritten with the content of register N2. In the case of an error, the error is stored in register N1 and the program continued at the label "ERROR_SDO".
3.64.6 Commands for the processing of process data objects

As described in the previous section, CANopen makes a distinction between two basic data transfer mechanisms, i.e. the high-speed exchange of short process data via so-called process data objects (PDOs) and the access to entries of the object data directory via service data objects (SDOs).

Process data objects can be optionally transferred event-driven or synchronized. The event-driven mode enables the constant transfer of the complete process image to be dispensed with. Only the changes need to be transferred. In this way, the bus loading and the reaction time can be reduced to a minimum. A high communication performance is obtained at a comparatively low baud rate.

Process data objects can be sent and received. Sending and receiving is always regarded from the point of view of the slave, e.g. an I/O module.

Four process data objects are assigned to each device on the CANopen bus. There is a command and one storage location for each of them in PA-CONTROL.

The CANopen protocol is based on the CAN Application Layer (CAL) and makes standardised device profiles available. The communication profile DS-301 forms the basis for communication in the Canopen bus. It defines the permissible communication mechanisms between the devices on the bus. The communication profile is the basis for diverse device profiles which enable standardised access to all parameters of a device. The profiles listed below have been implemented so far:

<table>
<thead>
<tr>
<th>Profile</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>DS-301</td>
<td>CANopen communication profile</td>
</tr>
<tr>
<td>DS-401</td>
<td>Device profile for digital and analog I/O subassemblies</td>
</tr>
<tr>
<td>DS-402</td>
<td>Device profile for drive</td>
</tr>
<tr>
<td>DS-403</td>
<td>Device profile for control stations</td>
</tr>
<tr>
<td>DS-404</td>
<td>Device profile for sensors and controllers</td>
</tr>
<tr>
<td>DS-405</td>
<td>Interface to programmable systems (IEC1131)</td>
</tr>
<tr>
<td>DS-406</td>
<td>Device profile for rotary encoders and encoders</td>
</tr>
<tr>
<td>DS-407</td>
<td>Application profile for local public transport</td>
</tr>
</tbody>
</table>

According to the definition of DEVICE-PROFILE 401 "I/O modules", the process data objects are assigned as follows. The order can be changed by "variable mapping".

<table>
<thead>
<tr>
<th>Receive PDO</th>
<th>PDO no.</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
<td>Maximum of 64 digital outputs (O1 – O64)</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>Maximum of 4 analog outputs (DA1 – DA4)</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>Free</td>
</tr>
<tr>
<td></td>
<td>4</td>
<td>Free</td>
</tr>
</tbody>
</table>
Transmit PDO

<table>
<thead>
<tr>
<th>PDO no.</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Maximum of 64 digital inputs (I1 – I64)</td>
</tr>
<tr>
<td>2</td>
<td>Maximum of 4 analog inputs (AD1 – AD4)</td>
</tr>
<tr>
<td>3</td>
<td>Free</td>
</tr>
<tr>
<td>4</td>
<td>Free</td>
</tr>
</tbody>
</table>

The third and fourth process data object is not permanently defined in the standard. The manufacturers of I/O modules apply the next digital and analogue inputs and outputs to these PDOs depending on the hardware configuration of the I/O module. Refer to the technical documentation of the manufacturer for further information about the application case.

Example: Murr I/O modul with
- 16 digital inputs
- 8 digital outputs
- 4 analog inputs (12 bit, +/-10V)
- 4 analog outputs (12 bit, +/-10V)

Receive PDO

<table>
<thead>
<tr>
<th>PDO no</th>
<th>Data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Maximum of 64 digital outputs (O1 – O8)</td>
</tr>
<tr>
<td>2</td>
<td>Maximum of 4 analog outputs (DA1 – DA4)</td>
</tr>
<tr>
<td>3</td>
<td>Free</td>
</tr>
<tr>
<td>4</td>
<td>Free</td>
</tr>
</tbody>
</table>

Commands for all four objects are available for accessing the process data objects (PDO).
- PDO1
- PDO2
- PDO3
- PDO4

The number of data bytes of a PDO can vary between 1 and 8.
3.64.6.1 Read process data

**G74x**

**G74x.(ID). (No. data bytes)**

The Canopen device sends its process data to PA-CONTROL cyclically or after synchronisation, depending on the configuration, i.e. if an input is changed. The received data is stored in the assigned PDO receive buffer (PDO1, PDO2,...).

The command "Read process data" causes the data to be copied from the receive buffer to the local S0 character buffer.

**NOTE** The process data must always be transferred via the CANopen bus in the INTEL format for evaluation of the data. The data in the local S0 character buffer is read out using the "GETI command" (GET-INTEL format), page 232 and is then available for evaluation.

**Command format:**

- G741.(ID).(No. data bytes) PDO1
- G742.(ID).(No. data bytes) PDO2
- G743.(ID).(No. data bytes) PDO3
- G744.(ID).(No. data bytes) PDO4

**Examples:**

- G741.49.4 The first four bytes of the data received from the first process data object (PDO1) are copied from CANopen device 49 to the S0 string.
- G742.50.2 The first two bytes of the data received from the second process data object (PDO1) are copied from CANopen device 50 to the S0 string.
3.64.6.2 Send process data

**G75x**

*G74x.(ID). (No. data bytes)*

Two steps are required to send process data. The data to be sent is entered in the S0 string in the first step. This is followed by the actual transmission to the CANopen device.

**NOTE**

The process data is transferred via the CANopen bus in the INTEL format. This must always be considered when entering the data in the local S0 character buffer. The PUTI command (PUT-INTEL format), page 234 writes the content of an N register to the S0 string.

**Command format:**

- \texttt{G751.(ID).(No. data bytes) PDO1}
- \texttt{G752.(ID).(No. data bytes) PDO2}
- \texttt{G753.(ID).(No. data bytes) PDO3}
- \texttt{G754.(ID).(No. data bytes) PDO4}

**Example:**

- \texttt{G751.49.4 Sends the first four bytes from the S0 string to the first process first data object (PDO1) of CANopen device 49.}
3.64.7 Typical applications

3.64.7.1 Example 1 for MURR I/O module

The CANopen module 55900 of Murr & Co. is used in the example.

- 55920: DI8 ➔ 8 inputs ➔ from flag M1 to M8
- 55922: DO8/0,5A ➔ 8 outputs ➔ from flag M9 to M16

Example 1: Start-SDO.pnc

```plaintext
1 G701.49 ; START REMOTE NODE
2 T50 ; wait until CAN modules are in OPERATION mode
3 ;
4 $A
5 G721.49 EMERGENCY ; Error message there?
6 ;
7 G730.49.6200.2.1.N1.N2 READ_SDO_ERROR ; Read SDO ➔ inputs
8 G602.N2.1.B ; Image to flag M1 – M8
9 ;
10 G604.N3.9.8 ; Image from flag M9 – M16
11 G731.49.6200.1.1.N1.N3 WRITE_SDO_ERROR ; Write SDO ➔ outputs
12 JMP A
13 ;
14 $READ_SDO_ERROR
15 $WRITE_SDO_ERROR
16 O1:1 ; Report error
17 I1.1 ; SDO error
18 BREAK ;
19 ;
20 $EDITOR
21 N5:=GETI.1.2 ; Error code
22 N6:=GETI.3.1 ; Error register object 0x1001
23 M100:=N5=0 ; Error <>0?
24 G21 M100.1 A ; NO, no error!
25 I1.1 ; Emergency error
26 Break ;
27 ;
28 END
```

**NOTE**

The NODE GUARDING, by means of which the device is addressed via SDOs, can be dispensed with in the above example.
3.64.7.2 Example 2 for MURR I/O module

The CANopen module 55900 of Murr & Co. is used in the example.

55920: DI8  8 inputs  from flag M1 to M8
55922: DO8/0,5A  8 outputs  from flag M9 to M16

Example 2: Start-pdo.pnc

1 G701.49 ; START REMOTE NODE
2 T50 ; wait until CAN modules are in OPERATION mode
3 ;
4 $A
5 G721.49 EMERGENCY ; Error message there?
6 G711.49 NODE_GUARD ;
7 G741.49.2 ; Read PDO1 → S0
8 N2:=GETI.2.1 ; Get 2 bytes from S0
9 G602.N2.1.8 ; Image to flag M1 – M8
10 ;
11 G604.N3.9.8 ; Get image from flag M9 – M16
12 PUTI.1.1.N3 ; Insert in S0
13 G751.49.1 ; S0 → write PDO1
14 JMP A
15 ;
16 $EMERGENCY
17 N5:=GETI.1.2 ; Error code
18 N6:=GETI.3.1 ; Error register Object 0x1001
19 M100:=N5=0 ; Error <>0 ?
20 G21 M100.1 A ; NO, no error!
21 I1.1 ; Emergency error
22 BREAK ;
23 ;
24 $NODE_GUARD , Device has not reported to NODE GUARD
25 I1.1
26 BREAK
27 ;
28 END
4 Startup

4.1 Important Information

PA-CONTROL may only be operated in combination with motors approved by IEF Werner. The earthing connection of the motors must be executed correctly. The cables and lead wires used must comply with EN60204.

All power lines and signal lines must be shielded. The shielding must have low impedance (large surface connection). Warranties for trouble-free operation only apply if original IEF cables are used.

Control cables and power cables must be laid separately at a minimum distance of 10 cm apart. Electrical connections on the device may only be disconnected in a no-voltage condition.

NOTE Installation and startup may only be carried out by qualified personnel in accordance with EN60204.

4.2 Installation of a PA-CONTROL servoTEC

Ambient conditions such as temperature, dirt or humidity influence the perfect operation of a positioning and execution control system. Limit values (temperature 0°C to 40°C) and certain conditions must be considered for the installation of PA-CONTROL. PA-CONTROL may not be placed near strong electric or magnetic fields (e.g. welding transformer) or be subject to mechanic interferences (e.g. vibrations).

Ensure that air can circulate freely (air inlet and outlet ports must be kept free).

NOTE Safety instructions must be followed to the letter (see Chapter 1, "Technical Short Information").

WARNING Protection from electric shock! The controller must be installed in accordance with EN 60204, in order to prevent personnel from coming into direct contact with it.

NOTE The fitting dimensions and instructions are described in the Chapter "Technical Short Information".

4.3 Wiring of the connections

Please refer to the chapter "Technical Appendix" for the assignment of the individual connectors. Connection examples for the individual connectors are also shown there.

The following connections must be made.

The grey fields indicate the connections which have to be made:
<table>
<thead>
<tr>
<th>Required</th>
<th>Important</th>
<th>Optional</th>
</tr>
</thead>
<tbody>
<tr>
<td>24V for CPU</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IEF wiring module</td>
<td></td>
<td></td>
</tr>
<tr>
<td>External 24V for I/O</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Inputs/outputs</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Stop input</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Diagnosis line</td>
<td></td>
<td></td>
</tr>
<tr>
<td>IEF control console</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Profibus DP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>COM1,2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>A-D converter</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Original IEF cables must be used to ensure trouble-free operation. Furthermore, EMC-compatible screw connections (e.g. on motors) and EMC-compatible Sub-D housings (metallized) must be used.

**CAUTION**

Connectors may only be inserted or removed in a no-voltage condition.
4.3.1 Wiring structure of the PA-CONTROL servoTEC

Fig. 15: Wiring of the PA-CONTROL servoTEC
4.3.2 Cables

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>IEF No. Standard length</th>
<th>IEF No. Special length</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Interface cable (PC ↔ PA-CONTROL servoTEC)</td>
<td>231766</td>
<td>1002839</td>
</tr>
<tr>
<td>2</td>
<td>Interface cable &quot;Console&quot; (for IEF control console)</td>
<td>231766</td>
<td>1002839</td>
</tr>
<tr>
<td>4</td>
<td>Interface cable &quot;Diagnosis&quot; (for IEF control console)</td>
<td>231585</td>
<td>231586</td>
</tr>
<tr>
<td>4</td>
<td>CANopen cable (PA-CONTROL ↔ LV servoTEC via IEF CANopen adapter)</td>
<td>1025142 (1m)</td>
<td>1026674</td>
</tr>
<tr>
<td>5</td>
<td>CANopen cable (LV servoTEC to LV servoTEC)</td>
<td>1025141 (0.35m)</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>I/O cable (PA-CONTROL servoTEC)</td>
<td>1029394</td>
<td>1029391</td>
</tr>
</tbody>
</table>

4.3.3 Optional accessories

<table>
<thead>
<tr>
<th>Description</th>
<th>IEF No.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Option IEF control console for surface-mounting on walls</td>
<td>231764</td>
</tr>
<tr>
<td>Option IEF control console for flush-mounting in walls</td>
<td>1022752</td>
</tr>
<tr>
<td>Option SÜTRON control console via CANopen bus</td>
<td></td>
</tr>
<tr>
<td>Option Port for control console</td>
<td>527452</td>
</tr>
<tr>
<td>Option A-D converter</td>
<td>527451</td>
</tr>
<tr>
<td>Option RS 232 port</td>
<td>527450</td>
</tr>
<tr>
<td>Option Profibus DP Slave Interface</td>
<td>527184</td>
</tr>
<tr>
<td>Option Varioface module</td>
<td>1030818</td>
</tr>
</tbody>
</table>
4.4 Connector Assignments

4.4.1 Inputs and outputs

Fig. 16: Assignment of the inputs and outputs of PA-CONTROL servoTEC

4.4.2 Limit switches

Axis limit switches are wired to the terminals of the power amplifier in the PA-CONTROL servoTEC (see operating instructions of LV servoTEC, Chapter 3.7).

4.4.3 Diagnosis port

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DCD, Reception level</td>
</tr>
<tr>
<td>2</td>
<td>RXD, Receive Data</td>
</tr>
<tr>
<td>3</td>
<td>TXD, Transmit Data</td>
</tr>
<tr>
<td>4</td>
<td>DTR, Terminal ready</td>
</tr>
<tr>
<td>5</td>
<td>GND</td>
</tr>
<tr>
<td>6</td>
<td>N.C.</td>
</tr>
<tr>
<td>7</td>
<td>RTS, switch on transmitter</td>
</tr>
<tr>
<td>8</td>
<td>CTS, ready for transmission</td>
</tr>
<tr>
<td>9</td>
<td>RI, incoming call</td>
</tr>
</tbody>
</table>

Fig. 17: Diagnosis port, Sub-D, 9 poles; pins

NOTE The assignment corresponds to the asynchronous standard of the RS232 port. A shielded line must be used to ensure the troublefree exchange of data.
4.4.4 Rotation monitoring

Resolvers or encoders of servo motors are wired to the connections of the power amplifier in the PA-CONTROL servoTEC (see Chapter 3.7 of the operating instructions of LV servoTEC).

4.4.5 Brake connector

The connections for a brake are wired to the terminals of the power amplifier in the PA-CONTROL servoTEC (see Chapter 3.7 of the operating instructions of LV servoTEC).

---

CAUTION

Connectors may only be inserted or removed in a no-voltage condition!
4.4.6 COM1 RS232

4.4.6.1 COM1 Port

COM1 is a serial asynchronous port of the type RS232 and is available on the CPU5 as an option via an IEF module. The port can be set by the user within a wide range of limits. Other devices in the plant can be communicated with in automatic mode by means of this port.

![Diagram of Connector Assignment for COM1: Sub-D, 9 poles; pins](image)

1. N.C.
2. RXD, Receive data
3. TXD, Transmit data
4. N.C.
5. GND
6. N.C.
7. RTS, switch on transmitter
8. CTS, ready for transmission
9. N.C.

**Fig. 18: Connector assignment for COM1: Sub-D, 9 poles; pins**

The assignment corresponds to asynchronous standard RS232.

**NOTE**

A shielded line must be used to ensure the troublefree exchange of data.
4.4.7 CAN BUS

Communication with other devices is possible via the CAN bus, e.g. LV-servoTEC.

![Diagram of CAN bus connector assignment]

**Fig. 19: Connector assignment for CAN bus, Sub-D, 9 poles; pins**

**NOTE**

DIN ISO 11898 must be complied with when the CAN bus is installed.
4.4.8 Profibus Option

PA-CONTROL can be integrated in a Profibus network by means of the Profibus option. Please refer to the "Interface manual" for further information.

Fig. 20: Connector assignment for Profibus: Sub-D, 9 poles; socket

* These signals are required if signal amplification is necessary owing to a bus extension.

Fig. 21: External connection circuit

NOTE EN 50170 and the Profibus-DP installation guidelines must be complied with when the Profibus option is installed.
4.5 Function and status check

4.5.1 Functional check via a connected IEF control console

The inputs and outputs of PA-CONTROL can be checked using the sub-menu item "Diagnosis" (See Chapter 2 "User Interface → Diagnosis" for a description).

The following checks must be carried out without fail:
- Check of the function of the limit switches
- Check of the statuses of the connected inputs
- Check of the statuses of the connected outputs
- Check of the standby of the LV servotec

4.5.1.1 Parameter Settings

Please refer to Chapter 5 "Parameters" for the parameter settings.

**CAUTION**

The parameters must be checked in accordance with the prevailing conditions!

4.5.1.2 Creating a PNC program

The main menu must be exited to create a new PNC program. The PA-CONTROL program editor opens after selection of the menu items "Programming → Create new program" and entry of a program name of the type *.PNC.

An example of a simple PNC program, which carries out a small test using the A1 axis, is shown below.

The PNC program is to execute the following functions:
- Approach to reference point with axis A1
- Set the absolute measuring system
- Set positioning speed to 500 mm/s \([\text{AE/s}]\)
- Traverse with axis A1 to position 100
- Wait 2 seconds
- Traverse with axis A1 to position 10, wait 1 second

The program will be named "EXAMPLE":

**CAUTION**

Check without fail that the program can be executed on the machine in question!
Use the following keys in the program editor to create this program:

**Program: EXAMPLE**

**Program:**

<table>
<thead>
<tr>
<th>Key</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>G90.A1</td>
<td>G 9 0 ENTER</td>
</tr>
<tr>
<td>FA1:=500</td>
<td>F A 1 = 5 0 0 ENTER</td>
</tr>
<tr>
<td>A1:=100</td>
<td>A 1 = 1 0 0 ENTER</td>
</tr>
<tr>
<td>T200</td>
<td>T 2 0 0 ENTER</td>
</tr>
<tr>
<td>A1:=10</td>
<td>A 1 = 1 0 ENTER</td>
</tr>
<tr>
<td>T100</td>
<td>T 1 0 0 ENTER</td>
</tr>
<tr>
<td>END</td>
<td>END ENTER</td>
</tr>
</tbody>
</table>

The program input is concluded and exited by pressing the "ESC" key and then confirming storage of the program.

**ESC**
- End editor Enter "1"
- Store program Enter "1"

If errors or incorrect commands were input during the program input, relevant error messages are output to the display.

These errors must be remedied, otherwise the program cannot be stored.

4.5.1.3 Specification of the start program

Before a created program can be processed, PA-CONTROL must be informed about which PNC program is to be used to start the automatic run.

This is specified in the menu item "Run definition → Start program" (see the Chapter on "Operator interface").
4.5.1.4 Program execution

The PAC must be switched to automatic mode to run a program (main menu 1st line). The program is executed after pressing the start key. It can be stopped by means of the stop key.

The following is displayed during the program run:

1st line: Program name
2nd line: Current program line number with the commands

Example:

```
EXAMPLE
1 G90.A1
```

The program EXAMPLE from Section 4.6 executes the following steps:

Display (2nd line) Execution of PA-CONTROL
G90.A1 The following positionings are executed in the absolute dimension system.
3 FA1:=500 The traversing speed of axis A1 is set to 500 AE/sec.
4 A1:=100 PA-CONTROL moves to position 100 with axis A1.
5 T200 PA-CONTROL waits 2 seconds before the next command is executed.
6 A1:=10 The PA-CONTROL moves to position 10 with axis A1.
7 T100 PA-CONTROL waits 1 second before the next command is executed.
8 END The automatic run is ended.

PA-CONTROL returns to the main menu.
### 4.5.2 Functional check of the device with WINPAC

The following information is provided to ensure that start-up is executed in a suitable sequence without danger for man or machine.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Check installation</td>
<td>In a no-voltage condition</td>
</tr>
<tr>
<td>Check protective conductor system</td>
<td>Protective conductor measurement to EN50178</td>
</tr>
<tr>
<td>Connect PA-CONTROL servoTEC to the C and start WINPAC</td>
<td>The correct PC port must be entered in the WINPAC program.</td>
</tr>
<tr>
<td>Switch on 24V supply for inputs/outputs</td>
<td>When using the IEF wiring module, the LEDs IN (green) and OUT (red) are lit.</td>
</tr>
<tr>
<td>Switch on AC input power supply</td>
<td>If the wiring is correct, this message appears in the display of PA-CONTROL</td>
</tr>
<tr>
<td>Connect the PC and PA-CONTROL via the &quot;Blue Hand&quot; Button</td>
<td>If the connection was successful, the message &quot;ONLINE&quot; appears at the bottom edge of the screen.</td>
</tr>
<tr>
<td>Set the project folder and motor parameters and transfer.</td>
<td>See “Chapter 5, “Parameters”</td>
</tr>
<tr>
<td>Check inputs/outputs and test stop input</td>
<td>See “Chapter 5, “Parameters”, these inputs can be used as an option</td>
</tr>
<tr>
<td>Traverse axes manually</td>
<td>See Chapter 2, Section 2.5, “Manual” on axes manually via front plate</td>
</tr>
<tr>
<td>Put optional subassemblies into operation.</td>
<td>See documentation of the subassemblies.</td>
</tr>
<tr>
<td>Create program and transfer to PA-CONTROL</td>
<td>See Chapter 3, “Commands of the PA-CONTROL Family”, for programming tips.</td>
</tr>
</tbody>
</table>

Good luck!
4.5.2.1 Creating a project folder

After wiring has been completed and the connections checked, the controller can be put into operation via "WINPAC", the program development system for PA-CONTROL.

To do this, a current project folder must first of all be created.

Fig. 22: WINPAC, setting the project folder

General settings for the configuration of WINPAC are stored in the new project folder. Refer to the "WINPAC" operating instructions for details of the exact procedure.
4.5.2.2 Parameters of PA-CONTROL

After a project folder has been successfully created, the hardware configuration must be retrieved from PA-CONTROL and set if necessary. The following parameters are to be set:

- Axis parameters
- Traversing parameters
- Motor parameters

---

**Fig. 23: Main screen, setting the PA-CONTROL parameters**

**Fig. 24: Setting the axis parameters**
Refer to Chapter 5, “Parameters”, for a description of the individual parameters.

After the desired parameters have been set, a connection to PA-CONTROL must be set up, in order to transfer the set parameters.

4.5.2.3 Function and status check

The function and status check is carried out by means of the program development system WINPAC or the optional IEF control console.

By means of WINPAC, you can check the following points for their function, value and status via the "Diagnosis" sub-menu:

**Function check:**
- Programs
- Manual traversing
- System inputs (functions of the limit switches)
- Error message

**Status check:** ("0" “OFF” or "1" “ON”)
- Inputs
- Outputs (can be changed with right mouse button)
- Flags (can be changed with right mouse button)

**Value check:**
- N register (integer register)
- R register (real number register)
CAUTION

Ensure that no danger to the machine or personnel can arise if an output is incorrectly set or the drive does not move as intended.

CAUTION

Before the initial start-up of the motor, the function and position of the axis limit switches must be checked to ensure that they are correct.

Fig. 26: Changing an output, setting

Fig. 27: Changing an output, resetting
Fig. 28: Example: Manual traverse of an axis

The motor can be manually moved continuously to the left or the right by clicking the two buttons << and >> with the left mouse button. The axis can be moved within the range of the two axis limit switches.

4.5.2.4 Error diagnosis

Errors which occur during operation or start-up are stored in PA-CONTROL and can be read out and printed under WINPAC.

See Fig. Diagnosis of PA-CONTROL on page 311 of this chapter.
4.5.2.5 **First Aid**

The table below is for "First aid". Depending on the conditions of use, there may be other causes of malfunctions.

<table>
<thead>
<tr>
<th>Error</th>
<th>Possible cause</th>
</tr>
</thead>
<tbody>
<tr>
<td>No connection to PA-CONTROL possible</td>
<td>Diagnosis cable defective/not plugged in</td>
</tr>
<tr>
<td></td>
<td>Incorrect PC port set in the project parameters in</td>
</tr>
<tr>
<td></td>
<td>WINPAC</td>
</tr>
<tr>
<td></td>
<td>PA-CONTROL is not switched on</td>
</tr>
<tr>
<td>Motor does not turn</td>
<td>Mechanical problems</td>
</tr>
<tr>
<td></td>
<td>Motor cable not plugged in</td>
</tr>
<tr>
<td></td>
<td>Motor incorrectly connected</td>
</tr>
<tr>
<td></td>
<td>Limit switches A1+/A1- not active/transposed</td>
</tr>
<tr>
<td></td>
<td>Axis is at limit switch</td>
</tr>
<tr>
<td></td>
<td>Brake not opened</td>
</tr>
<tr>
<td></td>
<td>Incorrect motor parameters</td>
</tr>
<tr>
<td>Motor stops after only a few steps</td>
<td>Motor incorrectly connected</td>
</tr>
<tr>
<td></td>
<td>Rotary encoder not/incorrectly connected</td>
</tr>
<tr>
<td></td>
<td>Axis is at limit switch</td>
</tr>
<tr>
<td>No connection to I/O level</td>
<td>I/O cable not plugged in or is defective</td>
</tr>
<tr>
<td></td>
<td>Connect and/or check external 24V supply for I/O</td>
</tr>
<tr>
<td></td>
<td>level</td>
</tr>
<tr>
<td></td>
<td>IEF Vario module defective</td>
</tr>
</tbody>
</table>

Please refer to *Chapter 7, “Technical Appendix”,* for further information on error messages.
5 Parameters

5.1 General information on parameters

PA-CONTROL has several types of parameters:

- System parameters
  Parameters that concern the general system, such as user language, serial port, etc.
- Drive parameters:
  Parameters that concern the axes, such as traversing speed, motor steps, acceleration, etc.
- Option parameters:
  Profibus, RS232, etc.

Various basic settings have to be made before an appropriate program run can take place. This means that differentiated settings have to be made for various axes which are variously dimensioned and have varying loads.

The conditions for acceleration, maximum speed, gear factor etc. also have to be set for the individual axes. There are several motor and axis parameters available depending on the number of axes.

All the parameters can be modified using the Winpac program, some of them on the front plate of the device. The parameters of the diagnosis port are an exception, as they cannot be altered.

NOTE Parameters in the Winpac program which are displayed on a grey background are not available depending on the version of PA-Control.
5.2 Parameter description

This interface will be used for description purposes as all parameters can be altered using WINPAC. Access to the parameters via the front plate of the device is similar, but not possible for certain parameters.

The configuration window is called by the following menu items to edit the parameters:

![WINPAC menu bar, selected setting, PA-CONTROL](image)

**Fig. 29: WINPAC menu bar, selected setting, PA-CONTROL**

<table>
<thead>
<tr>
<th>Setting</th>
<th>System parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Diagnosis</td>
</tr>
<tr>
<td></td>
<td>Inputs and outputs</td>
</tr>
<tr>
<td></td>
<td>CANopen Bus</td>
</tr>
<tr>
<td></td>
<td>RS23 Port2</td>
</tr>
</tbody>
</table>

![Configuration window](image)

**Fig. 30: Configuration window**

All the parameters implemented in **PA-CONTROL** are edited in this configuration window.
5.2.1 System parameters

Fig. 31: Selection of the system parameters and language setting

(1) Language

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Language</td>
<td>1</td>
<td>German</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>English</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>French</td>
</tr>
</tbody>
</table>

(2) Autostart

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Parameter value</th>
<th>Meaning</th>
</tr>
</thead>
</table>
| Autostart | 1              | If the parameter Autostart is set to 1, **PA-CONTROL** begins to run the start program after being switched on (reset), i.e. it automatically switches into the automatic mode. **Prerequisites:**  
  - Start program must be defined  
  - An input must be defined as an external stop and be hot  
  **Note:**  
  The Autostart function is only executed once after switch-on (reset). If the program is to be executed several times, either an endless loop must be programmed or the program must be called again via a start input. |
(3) Access level

The system parameter "Control console access level" regulates the operator actions which can executed via a keyboard integrated in the device or via an IEF control console.

The system parameter "Access via diagnosis port" regulates the operator actions which can executed via the diagnosis port by means of the WINPAC program.

Both parameters are set to "0" on "Boot of PA-CONTROL". The parameter value can be altered in this status either via the integrated keyboard, the IEF control console or by means of the WINPAC program. If it is not equal to "0", it can only be reset by "Boot of PA-CONTROL". There are two possibilities here:

- By means of the WINPAC program or
- After switch-on, if a change in the hardware configuration is detected and the key-operated switch is in the "Program" position.

**Control console access level**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Par. Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>All menu items can be accessed (no restrictions)</td>
<td></td>
</tr>
</tbody>
</table>
| 1         | 3 = Programming (except for Change program *.PNX)  
            | 5 = Run definitions  
            | 6 = Parameters  
            | 7 = Basic settings  
            | The following are restricted in menu "9 = Communication via modem"  
            | 4 = Edit modem settings  
            | 5 = Select modem connection type |
| 2         | See level 1 and also  
            | 4 = Diagnosis |
| 3         | See level 1 and 2, also  
            | 2 = manual |
| 4         | See level 1, 2 and 3, also  
            | "9 = Communication via modem", Change program *.PNX |
| 5         | No function at present |
| ...       | ... |
| 255       | No function at present |

**Diagnosis port access level**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Par. Value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No restrictions</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Programs cannot be loaded from PA-CONTROL</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>No function at present</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td></td>
</tr>
<tr>
<td>255</td>
<td>No function at present</td>
<td></td>
</tr>
</tbody>
</table>
5.2.1.1 System I/O

Fig. 32: Display and change of the System I/Os

NOTE

Any input or output in the system can be used for the system I/Os.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>External start, input no.</td>
<td>Any input can be specified for the external start signal of PA-CONTROL. If 0 is entered, which is the standard value, there is no possibility for an external start. The following applies for the start: The transition from cold to hot triggers the start, i.e. NO contact function, positive edge evaluation.</td>
</tr>
<tr>
<td>External stop, input no.</td>
<td>Any input can be specified for the external stop signal of PA-CONTROL. If 0 is entered, which is the standard value, there is no possibility for an external stop. The following applies for the stop: If an external stop is defined, the selected input must be hot. Automatic mode is interrupted when the voltage supply is interrupted (principle of wire-break protection), i.e. NC contact function, status-driven. The stop button stops only the tasks started as a PNC program as well as all positionings. Tasks started as a PAB program are not affected and continue to run!</td>
</tr>
</tbody>
</table>

CAUTION
## 5.2.1.2 Run definitions

### Parameter

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manual release, input no.</td>
<td>Only active with &quot;Manual via front plate&quot;!</td>
</tr>
<tr>
<td>Standby, output no.</td>
<td>Any output can be specified for the output of stand-by in <strong>PA-CONTROL</strong>.</td>
</tr>
<tr>
<td>Malfunction, output no.</td>
<td>Any output can be specified for the output of a malfunction which has occurred in automatic mode in <strong>PA-CONTROL</strong> (e.g. limit switch actuated, value too high, etc.). If a malfunction occurs in automatic mode, the output is set. If the automatic mode is aborted and you return to the main menu, the output is reset.</td>
</tr>
<tr>
<td>Activate teleservice, input no.</td>
<td>Any input can be specified in <strong>PA-CONTROL</strong> for activation of the teleservice function. The function is activated when the input is hot.</td>
</tr>
<tr>
<td>Teleservice connected, output no.</td>
<td>Any output in <strong>PA-CONTROL</strong> can be specified for the status display of the teleservice function. Flashing indicates the connection set-up, steady light indicates connection.</td>
</tr>
</tbody>
</table>

---

**Fig. 33: Display and change of the run definition**
NOTE

Each program may only be registered once

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Start program</td>
<td><strong>PA-CONTROL</strong> starts in automatic mode with this program. In principle, this program is the main program.</td>
</tr>
<tr>
<td>Program after STOP</td>
<td>Special actions (e.g. closing valves) can be carried out with this assignment when &quot;STOP&quot; is detected in automatic run. This program is run after the axes have been stopped.</td>
</tr>
<tr>
<td>Program START after STOP</td>
<td>If &quot;STOP&quot; was actuated during the automatic run of PA-CONTROL and the program is to be restarted with &quot;START&quot; (automatic mode has not been exited), the program &quot;Start after Stop&quot; will be executed before continuation of the interrupted program. This option can be used to reset actions which were triggered by a stop.</td>
</tr>
<tr>
<td>Program at MALFUNCTION</td>
<td>Special actions (e.g. closing valves) can be implemented with this assignment when a malfunction is detected during the program run (error messages, e.g. value too large).</td>
</tr>
</tbody>
</table>
| Program in the initial position | The user can execute a program in the "initial position", e.g. in order to  
  • operate the light tower of an installation  
  • set the "correct window" of a SÜTRON control console  
  • communicate with a control station via COM1  
  See the section on "Explanatory information on the parameters" on the following pages for further information.                                      |
| Title                      | The title PA-CONTROL Vx.xx is replaced by the first line of the assigned *.PTX file.                                                                                                               |

NOTE

Any PNC, PNX or PAB programs can be registered in each case.  
**EXCEPTION:** Only a PTX program is allowed for the title!
Explanatory information on the parameters

Start program

PA-CONTROL starts in automatic mode with this program. In principle, this program is the main program.

Program at Stop

Special actions (e.g. closing valves) can be carried out with this assignment when "STOP" is detected in automatic run. This program is run after the axes have been stopped.

---

**CAUTION**

The command set has the following restrictions in the Program after Stop:

1. This program may not invoke any further programs as subroutines or programs to be processed in parallel.
2. Positioning commands are not permissible in this program.
3. The commands "Wait for logical status of..." inputs, outputs and flags may not be used.
4. Any time monitoring which is still active is reset by the operating system without a message and is not reactivated at "Start".

---

**Note**

The following commands may not be used in the program "after STOP":

- SUB, CASE.SUB
- RUN, CASE.RUN
- G22
- Ai:=n, Ai:=Rn
- G01
- G212, G222

PA-CONTROL checks in automatic mode and generates the errors E506, E507 or E508 if an error is found.
Program Start after Stop

If "STOP" was actuated during the automatic run of PA-CONTROL and the program is then to be restarted with "START" (automatic mode has not been exited), the program "Start after Stop" is executed before continuation of the interrupted program. This option can be used to reset actions which were triggered by a stop.

Note
This program may not invoke any further programs as subroutines or programs to be processed in parallel. Positioning commands are not permissible in this program.

```
SUB, CASE.SUB
RUN, CASE.RUN
G22
Ai:=n, Ai:=Rn
G01
G25.A, G26.A
G212, G222
```

PA-CONTROL checks in automatic mode and generates the errors E506, E507 or E508 if an error is found

Program at Malfunction

Special actions (e.g. closing valves) can be implemented with this assignment when a malfunction is detected during the program run (e.g. error messages such as: Value too large, power circuit not ready...).

Note
This program may not invoke any further programs as subroutines or programs to be processed in parallel. Positioning commands are not permissible in this program.

Program in initial position

The "Program in initial position" is activated via the window "Display and change of the Run definition" (Fehler! Verweisquelle konnte nicht gefunden werden., see page Fehler! Textmarke nicht definiert.).

The following are not allowed in the "Program in initial position":
- Traversing of axes (G25, ..., A1:=, ...)
- Activation of parallel TASKS (RUN, ...)
- Use of subroutines (SUB, G22, ...)

The program itself is started in the following situations:
- After switch-on
- On exiting the automatic mode
- On exiting the manual mode
- On exiting the diagnosis

Note
System flag SM67 is set after the initial position program has been processed
5.2.1.3 **CANopen control console**

**NOTE**

Only devices of SÜTRON & Co. are specified as control consoles on the CANopen bus of PA-CONTROL!

**Fig. 34: Display and change**

System parameters for the "poll range" are provided for control console 1 (CANopen address 63) and control console 2 (CANopen address 62) in PA-CONTROL.

**Default settings after reinitialization of PA-CONTROL**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Type</th>
<th>Console 1</th>
<th>Console 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coordination byte</td>
<td>Number of flag word</td>
<td>224</td>
<td>0</td>
</tr>
<tr>
<td>Serial signalling channel</td>
<td>Number of N register</td>
<td>2048</td>
<td>0</td>
</tr>
<tr>
<td>LEDs function keys</td>
<td>Number of flag word</td>
<td>218</td>
<td>0</td>
</tr>
</tbody>
</table>

A tab for changing the default setting is available in the WinPAC program (see Fig. 34).

**NOTE**

Please refer to the "SÜTRON control console" documentation for further information on the control console on the CANopen bus.
5.2.2 Diagnosis

5.2.2.1 System and diagnosis port

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port</td>
<td>Parameter is permanently set and can only be displayed and not changed</td>
</tr>
<tr>
<td>Baud rate</td>
<td>From V4.68 or higher, PA-CONTROL automatically adapts the baud rate of</td>
</tr>
<tr>
<td></td>
<td>the port to the baud rate pre-selected in the WINPAC program. The baud</td>
</tr>
<tr>
<td></td>
<td>rate can either be 19.2 or 57.6 Kbaud. The preset value is 19.2 Kbaud.</td>
</tr>
<tr>
<td>Data bits</td>
<td>Parameter is permanently set and can only be displayed and not changed</td>
</tr>
<tr>
<td>Stop bits</td>
<td>Parameter is permanently set and can only be displayed and not changed</td>
</tr>
<tr>
<td>Parity</td>
<td>Parameter is permanently set and can only be displayed and not changed</td>
</tr>
</tbody>
</table>

Fig. 35: System and diagnosis port
5.2.2.2 Teleservice

![Fig. 36: Setting the teleservice parameters](image)

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Type</td>
<td>Selected modem type</td>
</tr>
<tr>
<td>Initialise</td>
<td>Initialisation of modem</td>
</tr>
<tr>
<td>Service telephone no.:</td>
<td>Telephone number which is dialled by the modem</td>
</tr>
<tr>
<td>Dial prefix</td>
<td>String before telephone number</td>
</tr>
<tr>
<td>Dial suffix</td>
<td>String after telephone number</td>
</tr>
</tbody>
</table>
5.2.3 CANopen-Bus

Fig. 37: Setting the baud rate of the CANopen bus

Baud rate of the CANopen bus

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud rate</td>
<td>Standard= 500Kbaud, Selection: 125, 250, 500, 1000Kbaud</td>
</tr>
</tbody>
</table>
### 5.2.4 RS232 port

**Fig. 38: Setting the RS232/1 port**

**COM1 port**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud rate</td>
<td>Standard = 9600Bd, Selection: 110, 300, 1200, 2400, 4800, 9600, 19200.</td>
</tr>
</tbody>
</table>
| Port format     | Standard = 8B,nP,1S  
                 | Selection: 8B,nP,1S or 7B,nP,1S or 7B,eP,1S or 7B,oP,1S                                                           |
| Hardware        | Standard: no  
                 | Selection: yes/no                                                 |
| handshake       | Mode                                                                                                              |
|                 | This port can be changed over for various modes.                                                                  |
|                 | - 0 = Commands of the G500.xx group use this port  
                 | - 1 = ONLINE command port (see interface manual)  
                 | - 2 = Modem port (only for CPU4)                                |
### COM2 port

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baud rate</td>
<td>Standard = 9600Bd, Selection: 110, 300, 1200, 2400, 4800, 9600, 19200.</td>
</tr>
<tr>
<td>Port format</td>
<td>Standard = 8B,nP,1S or 7B,nP,1S or 7B,eP,1S or 7B,oP,1S</td>
</tr>
<tr>
<td>Hardware handshake</td>
<td>Standard : no, Selection : yes/no</td>
</tr>
<tr>
<td>Mode</td>
<td>Only commands of the G500.xx group use this port</td>
</tr>
</tbody>
</table>
5.2.5 Profibus

Fig. 39: Setting the Profibus-DP address

Setting the Profibus-DP address

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Profibus-DP address</td>
<td>Standard = 0, i.e. no Profibus-DP module activated</td>
</tr>
<tr>
<td></td>
<td>Selection: 2 to 126</td>
</tr>
<tr>
<td></td>
<td><strong>IMPORTANT</strong>: Address 126 is only permitted for commissioning. User data may not be transferred using address 126!</td>
</tr>
</tbody>
</table>
5.2.6 **Axis and drive parameters**

**NOTE**

Please refer to the HTML document "German ASCII object reference" for further explanatory information on the parameters presented in this section.

5.2.6.1 **Axis parameters**

---

**Fig. 40: Axis parameters of PA-CONTROL servoTEC**

**Active Parameters**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Display device</td>
<td>Setting for the desired display device. It is possible to add further units. The gear factor must be considered when setting the display device.</td>
</tr>
<tr>
<td>Gear factor</td>
<td>Setting for the gear factor</td>
</tr>
<tr>
<td>Traversing (min.) range</td>
<td>Lower traversing limit (software limit switch)</td>
</tr>
<tr>
<td>Traversing (max.) range</td>
<td>Upper traversing limit (software limit switch)</td>
</tr>
</tbody>
</table>
5.2.6.2 Traversing parameters

Active Parameters

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Traversing speed</td>
<td>Maximum permissible traversing speed.</td>
</tr>
<tr>
<td>Reference speed</td>
<td>Speed at which the approach to reference point is executed</td>
</tr>
<tr>
<td>Manual speed</td>
<td>Speed at which manual traversing is executed</td>
</tr>
<tr>
<td>Creep speed</td>
<td>Speed at which the limit switch is cleared during the approach to reference point</td>
</tr>
<tr>
<td>Acceleration</td>
<td>Change in speed per sec.</td>
</tr>
<tr>
<td>Acceleration ramp</td>
<td>2 setting possibilities: linear or sin^2 (t)</td>
</tr>
</tbody>
</table>
5.2.6.3 Limit switches / Approach to reference point

**Fig. 42: Limit switches and approach to reference point for PA-CONTROL servoTEC**

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Axis limit switch exchanged</td>
<td>Positive and negative limit switches can be exchanged without changes to hardware.</td>
</tr>
<tr>
<td>Limit switch type</td>
<td>Selection of the contact type used:</td>
</tr>
<tr>
<td></td>
<td>- Normally closed contact</td>
</tr>
<tr>
<td></td>
<td>- Normally open contact</td>
</tr>
<tr>
<td>Distance of the reference switch zero point</td>
<td>Entry for zero shift after approach to reference point</td>
</tr>
<tr>
<td>Acceleration ramp</td>
<td>Default value 200ms</td>
</tr>
<tr>
<td>Braking ramp</td>
<td>Default value 200ms</td>
</tr>
</tbody>
</table>
5.2.6.4 Drive parameters for positioning

![Image of Drive parameters for positioning](image)

**Active Parameters**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>In Position</td>
<td>PEINPOS</td>
<td>Status message for traversing job</td>
</tr>
<tr>
<td>Standstill threshold</td>
<td>VEL0</td>
<td>Speed threshold for standstill message</td>
</tr>
<tr>
<td>$V_{max}$</td>
<td>PVMAX</td>
<td>Maximum speed which may not be exceeded by a traversing set</td>
</tr>
<tr>
<td>Resolution</td>
<td>PGEARI</td>
<td>Increments per motor revolution</td>
</tr>
</tbody>
</table>
5.2.6.5  Drive parameters for position controller

![Parameters diagram]

**Active Parameters**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>KV</td>
<td>GP</td>
<td>Position controller: proportional gain</td>
</tr>
<tr>
<td>KP</td>
<td>GPV</td>
<td>Amplification of the speed controller when using the PI position controller</td>
</tr>
<tr>
<td>Tn [ms]</td>
<td>GPTN</td>
<td>Position controller: integral-action time</td>
</tr>
<tr>
<td>Max. tracking error</td>
<td>PEMAX</td>
<td>Tracking error monitoring</td>
</tr>
</tbody>
</table>
### 5.2.6.6 Drive parameters for speed controller

**Active Parameters**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Maximum speed</td>
<td>VLIM</td>
<td>Maximum speed for speed control</td>
</tr>
</tbody>
</table>
| Direction of rotation | DIR                                    | DIR=1, positive counting direction – positive rotational speed, speed and current settings cause the motor shaft to turn in a clockwise direction  
DIR=0, negative counting direction |
| Overspeed         | VOSPD                                    | Definition of the tripping threshold for error message F08 (overspeed) |
| SW ramp +         | ACC                                      | Acceleration ramp of the speed controller in msec |
| SW ramp -         | DEC                                      | Braking ramp of speed controller in msec |
| Emergency ramp    | DECSTOP                                  | Ramp for emergency stop  
- Error  
- Emergency stop input |
| Disable ramp      | DECDIS                                   | Speed of braking ramp on disabling the output stage, enable=0 |
5.2.6.7 Drive parameters for current controller

Fig. 46: Drive parameters for current controller

**Active Parameters**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>( I_{ms} )</td>
<td>( \text{ICONT} )</td>
<td>The constant current required by the application, corresponds to 100% ( I^2T ) current</td>
</tr>
<tr>
<td>( I_{\text{peak}} )</td>
<td>( \text{IPEAK} )</td>
<td>Sets the desired maximum current</td>
</tr>
<tr>
<td>Ref.-( I_{\text{peak}} )</td>
<td>( \text{REFIP} )</td>
<td>Definition of the peak current for the reference point approach to stop</td>
</tr>
<tr>
<td>( I^t ) message</td>
<td>( \text{I2TLIM} )</td>
<td>( I^t ) message threshold</td>
</tr>
</tbody>
</table>
5.2.6.8  Drive parameters for motor

**Active Parameters**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>I₀</td>
<td>MICONT</td>
<td>Rated current of the motor</td>
</tr>
<tr>
<td>I₀ max</td>
<td>MIPEAK</td>
<td>Peak current of the motor</td>
</tr>
<tr>
<td>Number of poles</td>
<td>MPOLES</td>
<td>Number of motor poles</td>
</tr>
<tr>
<td>L</td>
<td>L</td>
<td>Stator inductivity of the motor</td>
</tr>
<tr>
<td>Limit speed</td>
<td>MSPEED</td>
<td>Maximum speed of the motor</td>
</tr>
<tr>
<td>Motor no. name</td>
<td>MNUMBER/MNAME</td>
<td>Loading of a motor data set / motor name</td>
</tr>
<tr>
<td>Current advance</td>
<td>MTANGLP</td>
<td>Current advance</td>
</tr>
<tr>
<td>Final value Phi</td>
<td>MVANGLF</td>
<td>Speed-dependent advance (Final value Phi)</td>
</tr>
<tr>
<td>Service</td>
<td>MVANGLB</td>
<td>Speed-dependent advance (Service Phi)</td>
</tr>
<tr>
<td>Brake</td>
<td>MBRAKE</td>
<td>Preselection for motor brake</td>
</tr>
</tbody>
</table>
5.2.6.9 Drive parameters for feedback

![Configuration Interface](image)

**Fig. 48: Drive parameters for feedback**

**Active Parameter**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Feedback</td>
<td>FBTYPE</td>
<td>Preselection of the feedback unit</td>
</tr>
</tbody>
</table>
5.2.6.10 Drive parameters for basic settings

---

**Active Parameters**

<table>
<thead>
<tr>
<th>WINPAC Parameters</th>
<th>Designation in the ASCII object reference</th>
<th>Short description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ballast performance</td>
<td>PBALMAX</td>
<td>Maximum ballast performance</td>
</tr>
<tr>
<td>Max. supply voltage</td>
<td>VBUSBAL</td>
<td>Maximum supply voltage</td>
</tr>
<tr>
<td>Check supply system BTB</td>
<td>NONBTB</td>
<td>Check BTB for supply system on/off</td>
</tr>
<tr>
<td>Undervoltage monitoring</td>
<td>UVLTMODE</td>
<td>Undervoltage mode</td>
</tr>
<tr>
<td>Warning/fault mask</td>
<td>WMASK</td>
<td>Warning/fault mask</td>
</tr>
<tr>
<td>Initialization type for amplifier</td>
<td>BOOT</td>
<td>Type of initialization at the start of the amplifier</td>
</tr>
<tr>
<td>Mains phase missing</td>
<td>PMODE</td>
<td>Mains phase mode</td>
</tr>
<tr>
<td>Brake reaction time</td>
<td>TBRAKE</td>
<td>Disable delay time for braking</td>
</tr>
<tr>
<td>Deactivate</td>
<td>TBRAKE0</td>
<td>Brake release time</td>
</tr>
</tbody>
</table>

---

Fig. 49: Drive parameters for basic setting

---

Parameter File Name: STANDARD.PAR
6 Options

6.1 Options of the PA-CONTROL servoTEC

The functional scope of the PA-CONTROL servoTEC can be extended by numerous options. These expansions are implemented by means of plug-in IEF modules.

In line with the basic equipment of the device and the free installation space, not all options are possible simultaneously.

<table>
<thead>
<tr>
<th>Description of the option</th>
<th>CPU5</th>
</tr>
</thead>
<tbody>
<tr>
<td>CANopen</td>
<td>X</td>
</tr>
<tr>
<td>COM 1 / 2 port</td>
<td>X</td>
</tr>
<tr>
<td>COM 3 / 4 port</td>
<td>X</td>
</tr>
<tr>
<td>Profibus-DP</td>
<td>X</td>
</tr>
<tr>
<td>A-D converter</td>
<td>X</td>
</tr>
<tr>
<td>SSI</td>
<td>X</td>
</tr>
</tbody>
</table>

6.2 Plug-in connections of the PA-CONTROL servoTEC

The PA-CONTROL servoTEC is implemented by inserting a CPU5 in an LV servoTEC.

The connection to the LV servoTEC is effected by a ribbon cable, which is placed on the CAN (2) connector of the CPU5.
The diagnosis and CANopen bus cables are connected on the front plate. The connections for the inputs and outputs, the power supply and the expansion options are located on top of the PA-CONTROL.

**NOTE**

X26 is the preferred location for the IEF control console.

![Diagram](BILD701E)

**Fig. 51: Connections of the CPU5**

**NOTE**

Not all options can be combined at the same time, as the free installation space available varies depending on the basic equipment of the device.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>X26 (internal slot J2)</td>
<td>All IEF modules except Profibus</td>
</tr>
<tr>
<td>X25 (internal slot J3)</td>
<td>All IEF modules</td>
</tr>
<tr>
<td>Preferably X26 (internal slot J6 )</td>
<td>RS 232 port driver for operating terminal</td>
</tr>
</tbody>
</table>

**NOTE**

You will find information about the IEF modules from page 350 in this chapter.
6.3 CANopen interface

PA-CONTROL has been provided with a CANopen interface. This interface is used for the optional expansion of PA-CONTROL.

Technical Data:

- **Baud rate**: 500kBits/s, can be altered
- **Max. bus cable length**: 66m

To ensure operator-friendliness and an uncomplicated initial start-up (Plug&Play), some specifications have been made for the connection of instruments to the CANopen bus and their addresses (ID numbers).

### 6.3.1 CANopen devices with a permanent assignment

PA-CONTROL assumes the role of master as it were for the devices with a permanent assignment in the CANopen. It detects the devices on the CANopen bus after reinitialization, transfers them to the hardware configuration of PA-CONTROL and monitors the devices in the various modes.

The devices with a permanent assignment are controlled by the operating system of PA-CONTROL and made available to the user as an axis, input or output.

### 6.3.2 CANopen devices without a permanent assignment

Any CANopen device which meets the technical requirements (address, baud rate, ...) can be connected as a device without a permanent assignment.

These devices are not addressed and not supported by the operating system. Commands (G700 group) are available in the AUTOMATIC mode for communication with such devices. The devices can be

- initialized (NMT)
- parameterized (SDO)
- monitored (EMCY)
- written to (PDO) and
- read (PDO)

by the programs in AUTOMATIC mode by means of these commands.

The WINPAC program provides various windows for support in diagnosis and commissioning.
### 6.3.3 Overview of the CAN IDs

<table>
<thead>
<tr>
<th>CAN ID</th>
<th>Function in PA-CONTROL</th>
<th>Possible component</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Driving axis 1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Driving axis 2</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Driving axis 3</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Driving axis 4</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Driving axis 5</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Driving axis 6</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Driving axis 7</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Driving axis 8</td>
<td>- LV-servoTEC</td>
</tr>
<tr>
<td>9</td>
<td>Driving axis 9</td>
<td>- PA-CONTROL-MP</td>
</tr>
<tr>
<td>10</td>
<td>Driving axis 10</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Driving axis 11</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>Driving axis 12</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>Driving axis 13</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>Driving axis 14</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>Driving axis 15</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>Driving axis 16</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>Digital inputs/outputs (I/O1025 - I/O1088) Analog inputs/outputs (AD1 - AD4 / DA1 - DA4)</td>
<td>- BECKHOFF &quot;BK5120&quot;, &quot;BK5150&quot; „IL2301-B510“</td>
</tr>
<tr>
<td>18</td>
<td>Digital inputs/outputs (I/O1089 - I/O1152) Analog inputs/outputs (AD5 - AD8 / DA5 - DA8)</td>
<td>- MURR Elektronik &quot;MBM55900&quot;</td>
</tr>
<tr>
<td>19</td>
<td>Digital inputs/outputs (I/O1153 - I/O1216)</td>
<td>- FESTO &quot;CPV, CPX-FB14.&quot;</td>
</tr>
<tr>
<td>20</td>
<td>Digital inputs/outputs (I/O1217 - I/O1280)</td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>Digital inputs/outputs (I/O1281 - I/O1344)</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>Digital inputs/outputs (I/O1345 - I/O1408)</td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>Digital inputs/outputs (I/O1409 - I/O1472)</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>Digital inputs/outputs (I/O1473 - I/O1536)</td>
<td></td>
</tr>
<tr>
<td>49</td>
<td>Access via G700 commands in AUTOMATIC</td>
<td>- Any CANopen device</td>
</tr>
<tr>
<td>50</td>
<td>Access via G700 commands in AUTOMATIC</td>
<td></td>
</tr>
<tr>
<td>51</td>
<td>Access via G700 commands in AUTOMATIC</td>
<td></td>
</tr>
<tr>
<td>52</td>
<td>Access via G700 commands in AUTOMATIC</td>
<td></td>
</tr>
<tr>
<td>60</td>
<td>BUS access to PA-CONTROL (SÜTRON PDO protocol)</td>
<td>- SÜTRON control console</td>
</tr>
<tr>
<td>61</td>
<td>BUS access to PA-CONTROL (SÜTRON PDO protocol)</td>
<td></td>
</tr>
<tr>
<td>62</td>
<td>BUS access to PA-CONTROL (SÜTRON PDO protocol) System parameters available for &quot;polling range&quot;</td>
<td></td>
</tr>
<tr>
<td>63</td>
<td>BUS access to PA-CONTROL (SÜTRON PDO protocol) System parameters available for &quot;polling range&quot; Time monitoring for access</td>
<td></td>
</tr>
</tbody>
</table>
6.4 RS 232 port driver

The RS 232 port driver only executes the signal level adjustment from 5V to the RS232 level of the COM port.

![Fig. 52: RS 232 port driver](image)

![Fig. 53: Assignment of the RS 232 port (Sub-D, 9 poles; pins)](image)

Use:

<table>
<thead>
<tr>
<th>Use on:</th>
<th>Application for:</th>
<th>Connection to:</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU4</td>
<td>COM2</td>
<td>J3</td>
</tr>
<tr>
<td>IEF module RS232 port</td>
<td>COM2 / COM4</td>
<td>J2</td>
</tr>
<tr>
<td>PA-CONTROL servoTEC</td>
<td>Control console connection</td>
<td>X6</td>
</tr>
<tr>
<td>PA-CONTROL-MP</td>
<td>Control console connection</td>
<td>X5</td>
</tr>
</tbody>
</table>
6.5 **IEF module RS 232 port**

A COM1 and 2 or COM3 and 4 port can be implemented by means of the IEF module RS 232 port. This module occupies an IEF module slot.

A COM2 or COM4 port comes into being by means of the RS232 port driver, which is connected to the RS232 port module via connector J2.

![Fig. 54: RS 232 port module](image)

![Fig. 55: Assignment of the RS 232 port (Sub-D, 9 poles; pins)](image)
6.6 IEF module Profibus DP

PA-CONTROL can be integrated as a slave in a Profibus network by means of the Profibus option. Please refer to the "Profibus documentation" and the interface manual for further information.

![fig56](image1)

**Fig. 56: IEF module Profibus DP**

![fig57](image2)

**Fig. 57: Connector assignment (Sub-D, 9 poles, socket)**

- These signals are required when the signal must be amplified for a bus extension.

**NOTE**

EN50170 and the general installation guidelines for Profibus DP must be complied with when the Profibus option is installed.
6.7 IEF module A-D converter

This module is available in executions with 2 or 8 channels. In principle, the connector assignment is the same for both executions. See also the A-D converter manual.

Technical data:
- 12 bit converter
- 160 µsec/channel
- Input voltage ranges: (0-10) V or +/-10V
- Input resistance: 1MΩ in the 10V range

![IEF module A-D converter diagram](image)

**Fig. 58: IEF module A-D converter**

<table>
<thead>
<tr>
<th>Pin</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Analog input value channel 1</td>
</tr>
<tr>
<td>2</td>
<td>Analog input value channel 2</td>
</tr>
<tr>
<td>3</td>
<td>Analog input value channel 3</td>
</tr>
<tr>
<td>4</td>
<td>Analog input value channel 4</td>
</tr>
<tr>
<td>5</td>
<td>Analog input value channel 5</td>
</tr>
<tr>
<td>6</td>
<td>Analog input value channel 6</td>
</tr>
<tr>
<td>7</td>
<td>Analog input value channel 7</td>
</tr>
<tr>
<td>8</td>
<td>Analog input value channel 8</td>
</tr>
<tr>
<td>9</td>
<td>GND</td>
</tr>
</tbody>
</table>

**Fig. 59: Connector assignment (Sub-D, 9 poles, pins)**

**NOTE** Inputs not connected to signal sources must be connected to GND.
6.8 IEF module SSI interface

6.8.1 General

PA-CONTROL offers the user the option of assigning the axes with an absolute positioning system. As a result, the approach to reference point for the respective axis can be dispensed with and the position of the axes taken from the absolute value system.

The absolute positioning system is implemented by an IEF module.

Fig. 60: IEF module SSI interface

Up to two absolute positioning systems can be implemented with one module. The first one, Channel A, is wired up to plug-in connector J1. The second one, Channel B, is wired up to plug-in connector J2. From there, the signals are transmitted to an exterior 9 pole SUB D socket connector via a ribbon cable.

Assignment of the plug-in connectors J1 /J2 (Channel A/B)

<table>
<thead>
<tr>
<th>Type</th>
<th>SUB D socket connector, 9 poles</th>
</tr>
</thead>
</table>

<table>
<thead>
<tr>
<th>Pin.No.</th>
<th>Signal direction</th>
<th>Signal name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>➔</td>
<td>SSIClk+</td>
</tr>
<tr>
<td>2</td>
<td>➔</td>
<td>SSIClk-</td>
</tr>
<tr>
<td>3</td>
<td>←</td>
<td>SSIData+</td>
</tr>
<tr>
<td>4</td>
<td>←</td>
<td>SSIData-</td>
</tr>
<tr>
<td>5</td>
<td></td>
<td>Gnd</td>
</tr>
<tr>
<td>6</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>➔</td>
<td>+5VDC (supply for rotary encoder max. 100mA)</td>
</tr>
</tbody>
</table>
6.8.2 Detection and assignment

The "Absolute positioning system SSI interface" modules can be inserted in all possible IEF module slots in PA-CONTROL. Since there are 2 channels in a module, the number of modules which are detected and subsequently processed is restricted to 8. As a result, an absolute positioning system can be assigned to each of the 16 possible axes.

The modules are detected and applied during the reinitialisation of PA-CONTROL. The modules are checked every time PA-CONTROL is switched on.

6.8.2.1 Sequence for the detection of the modules

PA-CONTROL searches in the following order:

- IEF module slot 1 (CPU4/MP/servoTEC)
- IEF module slot 2 (CPU4/MP/servoTEC)
- Euro supporting board 1, IEF module slot 1
- Euro supporting board 1, IEF module slot 2
e.tc. up to
- Euro supporting board 4, IEF module slot 2

In doing so, PA-CONTROL first of all executes the following assignment between the absolute positioning systems and the axes.

<table>
<thead>
<tr>
<th>Detected module</th>
<th>Assigned axis</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number 1, Channel A</td>
<td>Axis 1</td>
</tr>
<tr>
<td>Number 1, Channel B</td>
<td>Axis 2</td>
</tr>
<tr>
<td>Number 2, Channel A</td>
<td>Axis 3</td>
</tr>
<tr>
<td>Number 3, Channel B</td>
<td>Axis 4</td>
</tr>
<tr>
<td>etc. up to</td>
<td></td>
</tr>
<tr>
<td>Number 8, Channel A</td>
<td>Axis 15</td>
</tr>
<tr>
<td>Number 8, Channel B</td>
<td>Axis 16</td>
</tr>
</tbody>
</table>

The assignment can be changed by means of WINPAC when setting the parameters.
### 6.8.3 Parameters

#### 6.8.3.1 Device parameters

**Fig. 61: Selection of absolute position encoders**

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>No. of absolute positioning systems</strong></td>
<td>is the number of the existing absolute positioning systems in PA-CONTROL (No. of modules * 2) (Can only be read via the diagnosis port)</td>
</tr>
<tr>
<td><strong>Assigned axis number</strong></td>
<td>Access number associated with this absolute positioning system (Channel).</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>none</td>
</tr>
<tr>
<td>1</td>
<td>Axis 1</td>
</tr>
<tr>
<td>2</td>
<td>Axis 2</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
</tbody>
</table>

(Can be read and written to via the diagnosis port)
Fig. 62: Assignment of the absolute positioning system

**Slot number**

Number of the slot in which the module is inserted. (Can only be read via the diagnosis port)
6.8.3.2 Axis parameters

Fig. 63: Activation of the absolute positioning system

<table>
<thead>
<tr>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Absolute positioning system activated</strong></td>
<td>Activation desired yes/no</td>
</tr>
<tr>
<td><strong>AbsPoSys Transmission clock pulse scaler</strong></td>
<td>Setting the frequency of the clock pulse of the SSI. Formula: $F = \frac{2\text{MHz}}{2n}$</td>
</tr>
<tr>
<td>0</td>
<td>1MHz</td>
</tr>
<tr>
<td>1</td>
<td>500KHz</td>
</tr>
<tr>
<td>2</td>
<td>333KHz</td>
</tr>
<tr>
<td>...</td>
<td></td>
</tr>
<tr>
<td>max. 32</td>
<td></td>
</tr>
<tr>
<td><strong>AbsPoSys Pause time scaler</strong></td>
<td>Setting the clock pulse space after which a value package was retrieved Formula: $T = n * 0.5\mu s$</td>
</tr>
<tr>
<td>2</td>
<td>1\mu s</td>
</tr>
<tr>
<td>4</td>
<td>2\mu s</td>
</tr>
<tr>
<td>20</td>
<td>10\mu s</td>
</tr>
<tr>
<td>max. 128</td>
<td></td>
</tr>
</tbody>
</table>
| **AbsPoSys Evaluation code variant** | The data format of an absolute positioning system can vary.  
| Common data formats:  
| 0 | Gray code  
| 1 | Binary code  
| 2 | BCD code  
| 3 | Gray excess code  |

| **AbsPoSys Multiturn** | There are absolute positioning systems for one or more revolutions  
| 0 | Singleturn - one revolution  
| 1 | Multiturn – several revolutions  |

| **AbsPoSys Number of data bits** | States the number of data bits which are to be received via the SSI interface  
| e.g.:  
| 25 | Indramat ECODRIVE3  |

| **AbsPoSys Number of data bits for position** | States the number of data bits which are used for position evaluation.  
| **AbsPoSys Function of data bit 25** | Data bit 25 can have different meanings in different absolute positioning systems.  
| 0 | No separate evaluation, is data bit 2^0  
| 1 | Ignore, do not use as a data bit  
| 2 | Parity bit  
| 3 | Power Fail Bit (PFB)  |

| **AbsPoSys Resolution factor** | Depending on the application, the increments of the absolute positioning system are to be variably evaluated.  
| The resolution factor is the conversion factor, so that the position of the absolute positioning system is identical to the axis position.  
| **AbsPoSys Offset to the reference point** | The zero point of the absolute positioning system and the axis is synchronized by this value.  
| **AbsPoSys Direction inverted** | The counting direction of the absolute positioning system can be inverted and the counting direction of the axis can be aligned by this means.  
| **AbsPoSys Increments per revolution** | Resolution in increments for one revolution of the absolute position encoder  
| **AbsPoSys max. number of revolutions** | Maximum number of revolutions of the absolute position encoder |
6.8.4 Application

On entering a traversing mode (MANUAL, AUTOMATIC, ONLINE), PA-CONTROL checks the settings in relation to the absolute positioning system.

- If an absolute positioning system has been activated in the parameters for an axis, but no system has been assigned, an error message is output and the operating mode cannot be executed.
- If the absolute positioning system does not function correctly after initialisation (SSI module does not report READY), an error message is output and the operating mode cannot be executed.
- On reception of the axis parameters, PA-Control checks the change of the parameter "Absolute positioning system activated". If the Parameter changes from "activated" to "deactivated", the "Reference flag" of this axis is reset.

6.8.4.1 Reinitialisation of PA-Control

The following data is reset after reinitialisation of PA-Control and may have to be set again.

- Assignment of the absolute positioning System to an axis
- Axis parameters for the absolute positioning system
- Flag of absolute positioning system is "referenced"

6.8.4.2 Referencing the absolute positioning system

The synchronization between an axis and the absolute positioning system is executed by an approach to reference point. The approach to reference point can be carried out in any traversing mode.

An automatic alignment between the axis system and the absolute positioning system is carried out at the end of an "APPROACH TO REFERENCE POINT". The calculated offset is stored in the axis parameter "AbsPoSys Offset to the reference point". The absolute positioning system is set to "referenced".

If the position data cannot be correctly transferred from the absolute positioning system at the end of the "APPROACH TO REFERENCE POINT" (no encoder connected, encoder data not complete, etc.), an error message is output. If PA-Control is in the AUTOMATIC or ONLINE mode, an "Exxx" message is output and the operating mode cannot be continued.

These settings and data are retained and are only deleted through a reinitialisation of PA-Control.

6.8.4.3 Traversing axes manually when absolute positioning system is activated

On entering the "MANUAL" mode, a check is carried out as to whether the absolute positioning system has been "referenced".

Yes: The current position is transferred from the absolute positioning system.
No: The position of the axes is not changed.

6.8.4.4 Axes in the AUTOMATIC or ONLINE mode when absolute positioning system is activated

On entering the "AUTOMATIC" or "ONLINE" mode, a check is carried out as to whether the absolute positioning system has been "referenced".

Yes: The current position is transferred from the absolute positioning system. Traversing can be executed without further referencing of the axis.
No: The position of the axes is not changed. The axis is not referenced and cannot be traversed without further referencing of the axis.
6.8.4.5 Measuring mode "G140 / G141" when absolute positioning system is activated

If an absolute positioning system has been installed and activated for an axis, the absolute positioning system has priority over the rotary encoder.

Irrespective of the settings for the measuring mode (synchronisation to rotary encoder, transfer of the rotary encoder position, etc.), the position is transferred from the absolute positioning system.

The actions for "Synchronisation to rotary encoder" (if activated) are not executed.

NOTE See the detailed description of the commands in Chapter 3, G140 / G141 commands
6.8.5 WINPAC and the absolute positioning System

6.8.5.1 Diagnosis

The diagnosis for the absolute positioning system takes place in the diagnosis window for the "Axis position" via the button "Extension of the absolute positioning system". The counter (incremental value) and the position of the absolute positioning system are displayed in this extended window.

![Display of the incremental value and position](image)

Fig. 64: Display of the incremental value and position
6.8.5.2 Setting the axis parameters

The tab "Absolute positioning system" is available in the field "Axis" in the "PA-CONTROL configuration" window.

Fig. 65: Selection of the tab "Absolute positioning system"

The set parameters are displayed in this tab.

Fig. 66: Display of the set parameters
Setting the parameter values:

These parameters can only be changed by selecting another absolute positioning system (see Fig. 66 on page 362).

A selection can be made from a number of absolute position encoders stored in a special file in the "Selection of absolute position encoders" window. The selection file was created by IEF Werner and is regularly updated.

**Fig. 67: Selection of a stored positioning system**

**Selection of the slot**

**Fig. 68: Selection of the slot on a CPU4**
6.9  IEF control console

6.9.1  General

The IEF control console is one of the numerous options for PA-CONTROL. It is used in the versions of PA-CONTROL which do not have their own keyboard.

In addition to the connection for the internal power supply (circular plug connection series 718 M8x1 / M8), the IEF control console has another three possibilities for connection via a 9 pole Sub-D plug connection:

- Diagnosis port, input / cable connection to the PC (9 pole connector on the front)
- Diagnosis port, output / cable connection to PA-CONTROL (9 pole socket at the bottom or on the back)
- Port for control console (9 pole socket at the bottom or on the back).

The respective connections must be provided on the controllers to enable connection of the IEF control console to the various versions of PA-CONTROL.

<table>
<thead>
<tr>
<th>PA-CONTROL Single, Compact, Steuer</th>
<th>Additional front plate on EURO bus with Sub-D plug connection required</th>
</tr>
</thead>
<tbody>
<tr>
<td>PA-CONTROL EP / MP</td>
<td>Expansion with RS232 port driver required</td>
</tr>
<tr>
<td>PA-CONTROL servoTEC</td>
<td>Expansion with RS232 port driver required</td>
</tr>
</tbody>
</table>
The following are part of the scope of supply of the IEF control console:

<table>
<thead>
<tr>
<th>Description</th>
<th>IEF no. (standard length, 3m)</th>
<th>IEF no. (special length, max. 20m)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Interface cable (connection of the diagnosis port to the PC)</td>
<td>Not in the scope of supply</td>
<td>Not in the scope of supply</td>
</tr>
<tr>
<td>Interface cable (data to and from the console)</td>
<td>231 766</td>
<td>on inquiry</td>
</tr>
<tr>
<td>Interface cable (connection of the diagnosis port to PA-CONTROL)</td>
<td>231 585</td>
<td>on inquiry</td>
</tr>
<tr>
<td>Power supply cable (5m)</td>
<td>732145</td>
<td>on inquiry</td>
</tr>
</tbody>
</table>

Pin assignment of cables 1 to 3:

<table>
<thead>
<tr>
<th>IEF control console</th>
<th>PA-CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pin 1 SCLK</td>
<td>Pin 4</td>
</tr>
<tr>
<td>Pin 2 MOSI</td>
<td>Pin 3</td>
</tr>
<tr>
<td>Pin 3 MISO</td>
<td>Pin 2</td>
</tr>
<tr>
<td>Pin 4 /STOP</td>
<td>Pin 1</td>
</tr>
<tr>
<td>Pin 5 GND</td>
<td>Pin 5</td>
</tr>
<tr>
<td>Pin 6</td>
<td></td>
</tr>
<tr>
<td>Pin 7 BER</td>
<td>Pin 8</td>
</tr>
<tr>
<td>Pin 8 SEL</td>
<td>Pin 7</td>
</tr>
</tbody>
</table>
Pin assignment of the 24VDC supply (looking at the contact pins of the console)

- Pin 1: 24VDC
- Pin 3: GND
- Pin 4: Not assigned

The following colour assignments apply if the IEF power supply cable, order number 732145, is used:

- Brown: 24VDC
- Blue: GND
- Black: not assigned

The IEF control console is available in two modifications, for installation in a control panel and in an aluminium housing as a surface-mounting device. In both cases, the cable connections can point vertically downwards or to the rear. This possibility is provided by a rotatable cover.

Fig. 71: IEF control console, connections
6.9.2  IEF control console - version for panel mounting

Fig. 72: IEF control console – mounting specification
6.9.3 IEF control console - standard version

Fig. 73: IEF control console, mounting dimensions

Fig. 74: IEF control console, hole pattern for standard version
6.10 SÜTRON control console via CANopen Bus

An interface for the connection of up to 4 control consoles is provided on the CAN bus of PA-CONTROL for the connection of the control consoles of SÜTRON & Co. to PA-CONTROL.

All variants of the control consoles of SÜTRON & Co. equipped with a CAN bus interface (e.g. BT8, HT06, TP22, ...) can be connected to PA-CONTROL.

The user interface is generated on the control consoles by means of the "TSwin" software (V2.35 or more recent) of SÜTRON & Co.

![Fig. 75: SÜTRON TP32](image)

6.10.1 CAN addresses and monitoring

Up to four control consoles can be connected to PA-CONTROL at the same time.

<table>
<thead>
<tr>
<th>Number of control console</th>
<th>CANopen address</th>
<th>Number of the system flag &quot;Control console active&quot;</th>
<th>Time monitoring of access by PA-CONTROL</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>63</td>
<td>27</td>
<td>Yes</td>
</tr>
<tr>
<td>2</td>
<td>62</td>
<td>28</td>
<td>No</td>
</tr>
<tr>
<td>3</td>
<td>61</td>
<td>29</td>
<td>No</td>
</tr>
<tr>
<td>4</td>
<td>60</td>
<td>30</td>
<td>No</td>
</tr>
</tbody>
</table>

System flags are provided, so that a check can be made in PA-CONTROL in automatic mode as to whether a SÜTRON control console has been connected and is active. The system flag assigned to a console is set every time it is accessed and reset five seconds after the last access.

The control console with the ID63 is time-monitored by PA-CONTROL (5 seconds). If the control console was active and becomes inactive for any reason (cable breakage, switched off, ...), error "E322 = TIME OUT with the SÜTRON control console" is set.
6.10.2 TSwin Configuring software

Templates are provided for the operating consoles of SÜTRON & Co.”.

A template is provided by IEF Werner for each of the different control consoles.

6.10.2.1 Directory structure for storage of the IEF templates

The user should create a directory "IEF" under "...\Programm\Tswin...\Templates" and copy the IEF templates to this. The IEF templates are stored on the IEF CD and can be copied from there.

Fig. 76: Directory structure

6.10.2.2 New project with IEF template

The IEF templates are offered for selection when a new project is created.

Fig. 77: Template selection
6.10.2.3 CAN bus settings with the TSwin configuring software

The settings for the CANopen communication have already been made in all IEF templates. Only the CAN address, "Terminal module number" or the baud rate need to be customized as required.

![Protocol parameters CAN](image)

Fig. 78: Protocol parameters

![Communication relationships](image)

Fig. 79: Communication relationships
Communication between the control console and PA-CONTROL only takes place via PDO1 (no SDO communication, no node guard, no ...).

The strings and display lines are accessed "in byte mode" in communication 2. Further access takes place in word mode in communication 1.

### 6.10.3 List of variables in the control console

Only a fraction of the possible variables for the inputs, outputs, flags, N registers and R registers are entered in the IEF templates. All other variables are complete in accordance with the version.

<table>
<thead>
<tr>
<th>Variable name</th>
<th>Address</th>
<th>Com. relations</th>
<th>Low bit no.</th>
<th>High bit no.</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>CAN-PA-CONTROL</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>Betriebssart</td>
<td>DW 8182:1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>Automatik-STAN</td>
<td>DW 8182:2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>Automatik-STOP</td>
<td>DW 8182:3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Automatik-AEBO</td>
<td>DW 8182:4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Poldbereich</td>
<td>W 8208:0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Fehlertext-Syst</td>
<td>BY 8224:0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Fehlertext-Abschal</td>
<td>BY 8225:0</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>N-Register</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>N0001</td>
<td>DW 12269:0</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>N0002</td>
<td>DW 12269:1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>N0003</td>
<td>DW 12269:2</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>N0004</td>
<td>DW 12269:3</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>N0005</td>
<td>DW 12269:4</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>N0006</td>
<td>DW 12269:5</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>N0007</td>
<td>DW 12269:6</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>N0008</td>
<td>DW 12269:7</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>N0009</td>
<td>DW 12269:8</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>N0010-Summen</td>
<td>DW 12269:9</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>N0011-Guetteil-Z</td>
<td>DW 12269:10</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**Fig. 80: List of variables**

The lists of all variables for PA-CONTROL are available as EXCEL files. The supplements to the variables can be copied from this file as required.
• TSwin-ControlVariable-Outputs-PA-CONTROL-V1-00.xls
• TSwin-ControlVariable-Inputs-PA-CONTROL-V1-00.xls
• TSwin-ControlVariable-FlagWords-PA-CONTROL-V1-00.xls
• TSwin-ControlVariable-FlagWords-PA-CONTROL-V1-00.xls
• TSwin-ControlVariable-NRegister-PA-CONTROL-V1-00.xls
• TSwin-ControlVariable-RRegister-PA-CONTROL-V1-00.xls

Fig. 81: Reference for variables
### 6.10.3.1 Object list through access of the data of PA-CONTROL

#### Object list Part 1

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Index (dec)</th>
<th>Sub-index (hex)</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>2000</td>
<td>8192</td>
<td>0</td>
<td>Number of SUB index 2000</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Operating status of PA-CONTROL</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>Automatic START</td>
<td>Signed 32</td>
<td>Wo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>Automatic STOP</td>
<td>Signed 32</td>
<td>Wo</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>Automatic ABORT</td>
<td>Signed 32</td>
<td>Wo</td>
</tr>
</tbody>
</table>

Polling range:

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Index (dec)</th>
<th>Sub-index (hex)</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>2010</td>
<td>8208</td>
<td>0</td>
<td>Coordination byte</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1</td>
<td>Serial signalling channel</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2</td>
<td>LEDs of function keys 1-8</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>3</td>
<td>LEDs of function keys 9-16</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>4</td>
<td>LEDs of function keys 17-24</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>5</td>
<td>LEDs of function keys 25-32</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>6</td>
<td>LEDs of function keys 33-40</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td>7</td>
<td>LEDs of function keys 41-48</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td>2020</td>
<td>8224</td>
<td>0- (3C)</td>
<td>System error text</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>2021</td>
<td>8225</td>
<td>0 - (3C)</td>
<td>Run error text</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>2100</td>
<td>8448</td>
<td>0</td>
<td>Number of input words</td>
<td>Unsigned 16</td>
<td>Ro</td>
</tr>
<tr>
<td>2101</td>
<td>8449</td>
<td>0-7F</td>
<td>Input word 1-128 (I1 - I2048)</td>
<td>Unsigned 16</td>
<td>Ro</td>
</tr>
<tr>
<td>2200</td>
<td>8704</td>
<td>0</td>
<td>Number of output words</td>
<td>Unsigned 16</td>
<td>Ro</td>
</tr>
<tr>
<td>2201</td>
<td>8705</td>
<td>0-7F</td>
<td>Output word 1-128 (O1 - O2048)</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td>2800</td>
<td>10240</td>
<td>0</td>
<td>Number of flag words</td>
<td>Unsigned 16</td>
<td>Ro</td>
</tr>
<tr>
<td>2801</td>
<td>10241</td>
<td>0-FF</td>
<td>Flag word flag 1-256 (M1 - M4096)</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td>2810</td>
<td>10256</td>
<td>0 - FF</td>
<td>Flag word 1-256 (M1 - M4096)</td>
<td>Unsigned 16</td>
<td>RW</td>
</tr>
<tr>
<td>3000</td>
<td>12288</td>
<td>0</td>
<td>Number of N registers</td>
<td>Unsigned 16</td>
<td>Ro</td>
</tr>
<tr>
<td>3001</td>
<td>12289</td>
<td>0-FF</td>
<td>N1 - N256</td>
<td>Signed 32</td>
<td>RW</td>
</tr>
<tr>
<td>3002</td>
<td>12290</td>
<td>0-FF</td>
<td>N257 - N512</td>
<td>Signed 32</td>
<td>RW</td>
</tr>
<tr>
<td>3002</td>
<td>12291</td>
<td>0-FF</td>
<td>N513 - N768</td>
<td>Signed 32</td>
<td>RW</td>
</tr>
</tbody>
</table>

...
### Object list Part 2

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Index (dec)</th>
<th>Sub-index (hex)</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>3008</td>
<td>12296</td>
<td>0-FF</td>
<td>N1793 - N2048</td>
<td>Signed 32</td>
<td>RW</td>
</tr>
<tr>
<td>3081</td>
<td>12417</td>
<td>0</td>
<td>N1 - N256</td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3082</td>
<td>12419</td>
<td>0</td>
<td>N257 - N512</td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3083</td>
<td>12419</td>
<td>0</td>
<td>N513 - N768</td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3084</td>
<td>12420</td>
<td>0</td>
<td></td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3085</td>
<td>12421</td>
<td>0</td>
<td></td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3086</td>
<td>12422</td>
<td>0</td>
<td></td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3087</td>
<td>12423</td>
<td>0</td>
<td></td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3088</td>
<td>12424</td>
<td>0</td>
<td>N1793 - N2048</td>
<td>Signed 16</td>
<td>RW</td>
</tr>
<tr>
<td>3100</td>
<td>12544</td>
<td>0</td>
<td>Number of R registers</td>
<td>Unsigned 16</td>
<td>Ro</td>
</tr>
<tr>
<td>3101</td>
<td>12545</td>
<td>0-FF</td>
<td>R1 - R256</td>
<td>Float 32</td>
<td>RW</td>
</tr>
<tr>
<td>3102</td>
<td>12546</td>
<td>0-FF</td>
<td>R257 - R512</td>
<td>Float 32</td>
<td>RW</td>
</tr>
<tr>
<td>3102</td>
<td>12547</td>
<td>0-FF</td>
<td>R513 - R768</td>
<td>Float 32</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3108</td>
<td>12552</td>
<td>0-FF</td>
<td>R1793 - R2048</td>
<td>Float 32</td>
<td>RW</td>
</tr>
<tr>
<td>3200</td>
<td>12800</td>
<td>0</td>
<td>Number of PA-CONTROL strings</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td>3201</td>
<td>12801</td>
<td>0 - (3C)</td>
<td>S1 (global String 1)</td>
<td>Unsigned 8</td>
<td>RW</td>
</tr>
<tr>
<td>3202</td>
<td>12802</td>
<td>0 - (3C)</td>
<td>S2 (global String 2)</td>
<td>Unsigned 8</td>
<td>RW</td>
</tr>
<tr>
<td>3203</td>
<td>12803</td>
<td>0 - (3C)</td>
<td>S3 (global String 3)</td>
<td>Unsigned 8</td>
<td>RW</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3210</td>
<td>12816</td>
<td>0 - (3C)</td>
<td>S16 (global String 16)</td>
<td>Unsigned 8</td>
<td>RW</td>
</tr>
<tr>
<td>3300</td>
<td>13056</td>
<td>0</td>
<td>Number of system flags</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td>3301</td>
<td>13057</td>
<td>0 - ...</td>
<td>SM1 - ...</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>3400</td>
<td>13312</td>
<td>0</td>
<td>Number of system N registers</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td>3401</td>
<td>13313</td>
<td>0 - ...</td>
<td>SN1 - ...</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td>3500</td>
<td>13568</td>
<td>0</td>
<td>Number of system R registers</td>
<td>Signed 32</td>
<td>RO</td>
</tr>
<tr>
<td>3501</td>
<td>13569</td>
<td>0 - ...</td>
<td>SR1 - ...</td>
<td>Float 32</td>
<td>RO</td>
</tr>
<tr>
<td>3801</td>
<td>14337</td>
<td>0 - (28)</td>
<td>Display PA-CONTROL front plate line 1</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
</tbody>
</table>
### Object list Part 3

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Index (dec)</th>
<th>Sub-index (hex)</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>3802</td>
<td>14338</td>
<td>0 - (28)</td>
<td>Display PA-CONTROL front plate line 2</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>3803</td>
<td>14339</td>
<td>0</td>
<td>Key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
<tr>
<td>3804</td>
<td>14340</td>
<td>0</td>
<td>STOP key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
<tr>
<td>3805</td>
<td>14341</td>
<td>0</td>
<td>Control console is activated</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>3806</td>
<td>14342</td>
<td>0</td>
<td>SHIFT key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
<tr>
<td>4000</td>
<td>16384</td>
<td>0</td>
<td>Current number of axes</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>4001</td>
<td>16385</td>
<td>0 - F</td>
<td>Current axis position</td>
<td>Float 32</td>
<td>RO</td>
</tr>
<tr>
<td>4002</td>
<td>16386</td>
<td>0 - F</td>
<td>Standby of the axis</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>4003</td>
<td>16387</td>
<td>0 - F</td>
<td>Axis status (in position / running / stopped)</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>4004</td>
<td>16388</td>
<td>0 - F</td>
<td>Status of negative limit switches</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>4005</td>
<td>16389</td>
<td>0 - F</td>
<td>Status of positive limit switches</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
</tbody>
</table>
6.10.4 Serial signalling system with polling range

The serial signalling system of SÜTRON control consoles consists of the following elements:
- Writing of coordination byte
- Serial signalling channel and
- Status of the LEDs of the function keys

The three elements are combined in the variable "Polling range" for communication via CAN bus.

The serial signalling system of SÜTRON control consoles consists of the following elements:
- Writing of coordination byte
- Serial signalling channel and
- Status of the LEDs of the function keys

The three elements are combined in the variable "Polling range" for communication via CAN bus.

```
<table>
<thead>
<tr>
<th>Control console</th>
<th>Number of flag word &quot;Write&quot; Coordination byte</th>
<th>Number of N register Serial signalling channel</th>
<th>Number of flag word LEDs of function keys</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 (ID63)</td>
<td>224</td>
<td>2048</td>
<td>218</td>
</tr>
<tr>
<td>2 (ID62)</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
```

A tab is available in the WinPAC program for setting the parameters for the polling range (see Fig. 83 page 378)
6.10.4.1 Coordination byte

The coordination byte exists as "Read coordination byte" and "Write coordination byte". Both use a flag word in PA-CONTROL.

- Write coordination byte means that PA-CONTROL writes this byte (flag).
- Read coordination byte means that PA-CONTROL reads this byte (flag).

6.10.4.1.1 Write coordination byte

"Write coordination byte" is associated with the polling range. The number of the flag word in which the coordination byte "Write" is stored is specified in a system parameter of PA-CONTROL (see above).

Flag word 224 (M3569 - M3584) is used in PA-CONTROL as the default setting for "Write coordination byte". This results in the following assignment:

<table>
<thead>
<tr>
<th>Write coordination byte (PA-CONTROL writes)</th>
<th>M3584</th>
<th>M3583</th>
<th>M3582</th>
<th>M3581</th>
<th>M3580</th>
<th>M3579</th>
<th>M3578</th>
<th>M3577</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Free</td>
<td>Free</td>
<td>Free</td>
<td>Data set download enable (DDF bit)</td>
<td>Ready flag (LM bit)</td>
<td>Reset password, delete (PL bit)</td>
<td>Refresh acknowledgement (RQ bit)</td>
<td>External data enabling (ED bit)</td>
</tr>
</tbody>
</table>
6.10.4.1.2 Read coordination byte

The "Read coordination byte" is set with its own variable in the TSwin program. A flag word should be used as a variable.

![Fig. 84: Setting the read coordination byte](image)

The flag word 217 (M3457 ... M3472) is used in the above example. This results in the following assignment:

<table>
<thead>
<tr>
<th>Read coordination byte (PA-CONTROL reads)</th>
</tr>
</thead>
<tbody>
<tr>
<td>M3464</td>
</tr>
<tr>
<td>M3463</td>
</tr>
<tr>
<td>M3462</td>
</tr>
<tr>
<td>M3461</td>
</tr>
<tr>
<td>M3460</td>
</tr>
<tr>
<td>M3459</td>
</tr>
<tr>
<td>M3458</td>
</tr>
<tr>
<td>M3457</td>
</tr>
</tbody>
</table>
6.10.4.2 Serial signalling channel

The serial signalling channel is stored in an N register. N register N2048 is used as the default setting in PA-CONTROL.

PA-CONTROL stores a 16 bit message number in this N register. The operator panel cyclically retrieves the entire polling range from PA-CONTROL and transmits the serial message at the same time.

As soon as a message (message number > 0) is detected, it is stored in the internal message store of the operator panel, and the N register in PA-CONTROL is reset to zero as an "acknowledgment".

External masks and message masks are addressed in the same way. As soon as the transferred number corresponds to a mask number, it is shown on the display. If the number has a mask and a message text, the mask (message mask, full-page error text) is shown on the display and the associated message text entered in the message store.

6.10.4.3 LEDs of function keys

Number of the flag word starting from which the status of the LEDs of the function keys is stored.

The LEDs of the function keys use a varying number of flag words depending on the setting in the control console.

![Fig. 85: Setting the polling range](image)

**Note on size:**

- 12 bytes → 6 Words: 1 Flag word for coordination byte
  1 N register for signalling channel
  4 Flag words for LEDs of function keys
- 10 bytes → 5 Words: 1 Flag word for coordination byte
  1 N register for signalling channel
  3 Flag words for LEDs of function keys

The flag word 218 (M3488 - M3568) is used as the default setting in PA-CONTROL.
### Assignment of the LEDs of the function keys to the flags of PA-CONTROL in the default

<table>
<thead>
<tr>
<th>MW218</th>
<th>MW219</th>
</tr>
</thead>
<tbody>
<tr>
<td>LED 1 on/off</td>
<td>LED 1 on/off</td>
</tr>
<tr>
<td>LED 2 on/off</td>
<td>LED 2 on/off</td>
</tr>
<tr>
<td>LED 3 on/off</td>
<td>LED 3 on/off</td>
</tr>
<tr>
<td>LED 4 on/off</td>
<td>LED 4 on/off</td>
</tr>
<tr>
<td>LED 5 on/off</td>
<td>LED 5 on/off</td>
</tr>
<tr>
<td>LED 6 on/off</td>
<td>LED 6 on/off</td>
</tr>
<tr>
<td>LED 7 on/off</td>
<td>LED 7 on/off</td>
</tr>
<tr>
<td>LED 8 on/off</td>
<td>LED 8 on/off</td>
</tr>
</tbody>
</table>

### 6.10.5 Parallel signalling system

The parallel signalling system is applied in the flag area (flag words) of PA-CONTROL.

The user must ensure that the settings are possible in PA-CONTROL and that the ranges for status and acknowledgement do not overlap.

256 parallel messages are implemented in the following example:

![Fig. 86: Setting the parallel signalling system](image-url)
This results in the following assignment:

<table>
<thead>
<tr>
<th>Status of the parallel signalling system</th>
</tr>
</thead>
<tbody>
<tr>
<td>M3585</td>
</tr>
<tr>
<td>M3586</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>M3601</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>M3839</td>
</tr>
<tr>
<td>M3840</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Acknowledgement of the parallel signalling system</th>
</tr>
</thead>
<tbody>
<tr>
<td>M3841</td>
</tr>
<tr>
<td>M3842</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>M3857</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>M4095</td>
</tr>
<tr>
<td>M4096</td>
</tr>
</tbody>
</table>
6.10.6 Simulation of the PA-CONTROL front plate

Special access options were created, in order to display the front plate of PA-CONTROL, and therefore all the operating possibilities provided by the front plate of PA-CONTROL, on the user interface of the control console.

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Index (dec)</th>
<th>Sub-index (hex)</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>3801</td>
<td>14337</td>
<td>0 - (28)</td>
<td>Display PA-CONTROL front plate line 1</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>3802</td>
<td>14338</td>
<td>0 - (28)</td>
<td>Display PA-CONTROL front plate line 2</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>3803</td>
<td>14339</td>
<td>0</td>
<td>Key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
<tr>
<td>3804</td>
<td>14340</td>
<td>0</td>
<td>STOP key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
<tr>
<td>3805</td>
<td>14341</td>
<td>0</td>
<td>Control console is activated</td>
<td>Unsigned 8</td>
<td>RO</td>
</tr>
<tr>
<td>3806</td>
<td>14342</td>
<td>0</td>
<td>SHIFT key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
</tbody>
</table>

Any existing PA-CONTROL front plate is disabled by accessing (reading) the display lines (Index 3801/3802).

PA-CONTROL front plates are implemented ready for use in the IEF templates:

![Fig. 87: Template for "IEF front plate"](image)

**NOTE** The font for the display line must be set to "COURIER 10", so that the special characters (ä, ü, ...) are displayed correctly
Fig. 88: Setting the font

Key code for cursor keys

<table>
<thead>
<tr>
<th>Index (hex)</th>
<th>Index (dec)</th>
<th>Sub-index (hex)</th>
<th>Name</th>
<th>Type</th>
<th>Access</th>
</tr>
</thead>
<tbody>
<tr>
<td>3804</td>
<td>14340</td>
<td>0</td>
<td>STOP key control console</td>
<td>Unsigned 8</td>
<td>WO</td>
</tr>
</tbody>
</table>

Since the control console can only generate one code between 0 and 255 for keys, the cursor keys were assigned with the following values. The standard ASCII characters are transferred with their standard code.
<table>
<thead>
<tr>
<th>Key designation</th>
<th>Code (dec)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CURUP</td>
<td>1</td>
</tr>
<tr>
<td>CURDOWN</td>
<td>2</td>
</tr>
<tr>
<td>CURRIGHT</td>
<td>3</td>
</tr>
<tr>
<td>CURLEFT</td>
<td>4</td>
</tr>
<tr>
<td>BACKSPACE</td>
<td>8</td>
</tr>
<tr>
<td>POS1</td>
<td>9</td>
</tr>
<tr>
<td>POSEND</td>
<td>10</td>
</tr>
<tr>
<td>PGUP</td>
<td>11</td>
</tr>
<tr>
<td>PGDOWN</td>
<td>12</td>
</tr>
<tr>
<td>ENTER</td>
<td>13</td>
</tr>
<tr>
<td>INS</td>
<td>14</td>
</tr>
<tr>
<td>DEL</td>
<td>15</td>
</tr>
<tr>
<td>ESC</td>
<td>27</td>
</tr>
<tr>
<td>ASCII character for example</td>
<td>set:</td>
</tr>
<tr>
<td>STARTKEY</td>
<td>222</td>
</tr>
</tbody>
</table>
7 Technical Appendix

7.1 Messages of the PA-CONTROL servoTEC

Operating status display

PA-CONTROL continuously monitors the function of its components and the program execution. Status and operating conditions as well as error and fault signals are generated and displayed as a result of this monitoring.

This information can be requested in all devices by means of the WINPAC program development system (see Chapter 4 Start-up and Documentation of WINPAC).

The error and fault signals are displayed in plain language on the operator terminal of devices which are equipped with this.

The CPU5 uses the two right-hand places of the three-place display of the LV servoTEC for the output of its operational status messages and system error messages. The left-hand place of the three display elements is provided exclusively for output of the operating statuses of the LV servoTEC.

7.1.1 Status messages of the LV servoTEC

The display elements of the 7-segment display are used as follows:

<table>
<thead>
<tr>
<th>Operating status LV servoTEC</th>
<th>Display</th>
</tr>
</thead>
<tbody>
<tr>
<td>24VDC</td>
<td>Lower cross-bar</td>
</tr>
<tr>
<td>24VDC + Power ON</td>
<td>Lower and middle cross-bar</td>
</tr>
<tr>
<td>24VDC + Power ON + Enable</td>
<td>All cross-bars</td>
</tr>
</tbody>
</table>

Fig. 89: PA-CONTROL servoTEC with display elements
7.1.2 Status LED of the CPU5

The CPU5 board has two LEDs. The LED mounted near slots J2 and J3 is used to check the supply voltage. The second LED mounted near the connector for the inputs indicates the status of the CPU5. The LED has not been led to the outside. It can be seen through the slotted hole for the CAN ribbon cable which connects the CPU5 and the LV servoTEC.

![Fig. 90: CPU5, Position of the status LED](image)

<table>
<thead>
<tr>
<th>Condition of the LED</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>OFF</td>
<td>Everything OK, the status messages are output via the two right-hand places of the display in which the CPU5 is inserted.</td>
</tr>
<tr>
<td>Flashes 1x / s</td>
<td>The LV servoTEC in which the CPU5 is inserted cannot be addressed.</td>
</tr>
<tr>
<td>Flashes 3x / s</td>
<td>A system error has occurred</td>
</tr>
<tr>
<td>Flashes 5x / s</td>
<td>No operating system is loaded in the CPU5</td>
</tr>
</tbody>
</table>

A status or error code is also output via the two right-hand places of the 7-segment display of the LV servoTEC.

The messages are displayed as numbers. In this connection, the two digit positions of the code are statically displayed in the status and operating status display (see the Section 7.1.3 table of operating statuses on page 390 in this chapter).

In the case of error and fault signals (see the tables from page 391 onwards), the individual digits of the respective code are displayed alternately in groups of two digit positions together with the letter E (error).

**Example of an operating status display:**

![Fig. 91: Display for "Automatic stopped"](image)

- Bar is only lit when the amplifier is in the "Enable" status
- See table on page 1 for the meaning of the bar
Example of a system error message:

![Error Message](image)

< alternating>

Fig. 92: Display for "external stop" cold
### 7.1.3 Status and operational status messages

<table>
<thead>
<tr>
<th>No.</th>
<th>Operational status</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dec</td>
<td>Hex</td>
<td></td>
</tr>
<tr>
<td>00</td>
<td>00</td>
<td>no communication</td>
</tr>
<tr>
<td>01</td>
<td>01</td>
<td>Initial position</td>
</tr>
<tr>
<td>02</td>
<td>02</td>
<td>Manual operation</td>
</tr>
<tr>
<td>03</td>
<td>03</td>
<td>Manual operation</td>
</tr>
<tr>
<td>10</td>
<td>0A</td>
<td>Automatic started</td>
</tr>
<tr>
<td>11</td>
<td>0B</td>
<td>Automatic running</td>
</tr>
<tr>
<td>12</td>
<td>0C</td>
<td>Automatic and &quot;STOP&quot; detected</td>
</tr>
<tr>
<td>13</td>
<td>0D</td>
<td>Automatic and process &quot;STOP&quot; program</td>
</tr>
<tr>
<td>15</td>
<td>0F</td>
<td>Automatic and stopped</td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>Automatic and process program &quot;START&quot; after &quot;STOP&quot;</td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>Automatic and malfunction detected</td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>Automatic and process &quot;Malfunction program&quot;</td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>Automatic with malfunction and stopped</td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>Manual traversing via serial port</td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>Online</td>
</tr>
<tr>
<td>35</td>
<td>23</td>
<td>Online stopped</td>
</tr>
<tr>
<td>39</td>
<td>27</td>
<td>Online stopped with malfunction</td>
</tr>
<tr>
<td>PA-CONTROL MP CANopen slave mode</td>
<td></td>
<td></td>
</tr>
<tr>
<td>50</td>
<td>32</td>
<td>CANopen slave mode started</td>
</tr>
<tr>
<td>51</td>
<td>33</td>
<td>CANopen slave mode, running</td>
</tr>
<tr>
<td>52</td>
<td>34</td>
<td>CANopen slave mode, STOP detected</td>
</tr>
<tr>
<td>55</td>
<td>37</td>
<td>CANopen slave mode, stopped</td>
</tr>
<tr>
<td>56</td>
<td>38</td>
<td>PA-CONTROL MP is working in measuring mode</td>
</tr>
<tr>
<td>57</td>
<td>39</td>
<td>CANopen slave mode, malfunction detected</td>
</tr>
<tr>
<td>59</td>
<td>3B</td>
<td>CANopen slave mode, malfunction detected and stopped</td>
</tr>
</tbody>
</table>
### 7.1.4 Error and fault signals

**CPU error message:**

<table>
<thead>
<tr>
<th>Code</th>
<th>CPU error messages</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>002</td>
<td>Bus error</td>
<td>CPU faulty</td>
</tr>
<tr>
<td>003</td>
<td>Address error</td>
<td>see above</td>
</tr>
<tr>
<td>004</td>
<td>Unimplemented command</td>
<td>see above</td>
</tr>
<tr>
<td>005</td>
<td>Division by zero</td>
<td>see above</td>
</tr>
<tr>
<td>006</td>
<td>Command chk</td>
<td>see above</td>
</tr>
<tr>
<td>007</td>
<td>Command Trapv</td>
<td>see above</td>
</tr>
<tr>
<td>008</td>
<td>Privilege violation</td>
<td>see above</td>
</tr>
<tr>
<td>009</td>
<td>Trace</td>
<td>see above</td>
</tr>
<tr>
<td>010</td>
<td>Emulator 1</td>
<td>see above</td>
</tr>
<tr>
<td>011</td>
<td>Emulator 2</td>
<td>see above</td>
</tr>
<tr>
<td>012</td>
<td>reserved</td>
<td>see above</td>
</tr>
<tr>
<td>013</td>
<td>Uninitialized interrupt</td>
<td>see above</td>
</tr>
<tr>
<td>014</td>
<td>Incorrect interrupt</td>
<td>see above</td>
</tr>
<tr>
<td>015</td>
<td>Uninitialized auto-vector1</td>
<td>see above</td>
</tr>
<tr>
<td>016</td>
<td>Uninitialized auto-vector2</td>
<td>see above</td>
</tr>
<tr>
<td>017</td>
<td>Uninitialized auto-vector3</td>
<td>see above</td>
</tr>
<tr>
<td>018</td>
<td>Uninitialized auto-vector4</td>
<td>see above</td>
</tr>
<tr>
<td>019</td>
<td>Uninitialized auto-vector5</td>
<td>see above</td>
</tr>
<tr>
<td>020</td>
<td>Uninitialized auto-vector6</td>
<td>&quot;Boot&quot; PA-CONTROL on switch-on</td>
</tr>
<tr>
<td>021</td>
<td>Uninitialized auto-vector7</td>
<td>&quot;Boot&quot; PA-CONTROL on switch-on</td>
</tr>
<tr>
<td>022</td>
<td>Uninitialized nonauto interrupt</td>
<td>&quot;Boot&quot; PA-CONTROL on switch-on</td>
</tr>
<tr>
<td>023</td>
<td>Uninitialized trap vector</td>
<td>&quot;Boot&quot; PA-CONTROL on switch-on</td>
</tr>
<tr>
<td>024</td>
<td>Uninitialized co-processor vector</td>
<td>&quot;Boot&quot; PA-CONTROL on switch-on</td>
</tr>
<tr>
<td>099</td>
<td>Incompatible operating system</td>
<td>The version of the boot system and the loaded operating system do not match. Eliminate the error by loading another version of the operating system</td>
</tr>
</tbody>
</table>

[The version of the boot system and the loaded operating system do not match. Eliminate the error by loading another version of the operating system]
## System error Hardware Part 1:

<table>
<thead>
<tr>
<th>Code</th>
<th>Error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E100</td>
<td>Keyboard port time-out</td>
<td>Keyboard removed, keyboard switched off</td>
</tr>
<tr>
<td>E101</td>
<td>Keyboard not ready</td>
<td>see above</td>
</tr>
<tr>
<td>E102</td>
<td>Short circuit in stepping motor output stage</td>
<td>Motor cable defective, motor defective</td>
</tr>
<tr>
<td>E103</td>
<td>Motor voltage incorrect / absent</td>
<td>230VAC not connected to the system? Bridge inserted in the motor connector? Motor power supply defective?</td>
</tr>
<tr>
<td>E104</td>
<td>Auxiliary voltages incorrect</td>
<td>Power supply defective</td>
</tr>
<tr>
<td>E105</td>
<td>Temperature in heat sink too high</td>
<td>Ventilation defective, ventilation clogged</td>
</tr>
<tr>
<td>E106</td>
<td>Motor rotation monitoring actuated</td>
<td>Motor run against mechanical stop, motor acceleration too high, rotary encoder wiring</td>
</tr>
<tr>
<td>E107</td>
<td>Phase sequence monitoring – no motor</td>
<td>Check motor wiring</td>
</tr>
<tr>
<td>E108</td>
<td>Axis type not defined</td>
<td>Non-existent axis addressed</td>
</tr>
<tr>
<td>E109</td>
<td>Axis type incorrectly defined</td>
<td>Hardware configuration changed</td>
</tr>
<tr>
<td>E110</td>
<td>Hardware configuration M-module slot</td>
<td>Module was exchanged, &quot;boot&quot; PA-CONTROL</td>
</tr>
<tr>
<td>E111</td>
<td>Hardware configuration IEF slot 1</td>
<td>Module was exchanged, &quot;boot&quot; PA-CONTROL</td>
</tr>
<tr>
<td>E112</td>
<td>Hardware configuration IEF slot 2</td>
<td>Module was exchanged, &quot;boot&quot; PA-CONTROL</td>
</tr>
<tr>
<td>E113</td>
<td>PLS7 interface different version</td>
<td>PLS7 is more recent than the operating system, match versions</td>
</tr>
<tr>
<td>E114</td>
<td>Axis power circuit not ready</td>
<td>LV-UNIT rotation monitoring error, wiring defective, LV-servoTEC supply voltage</td>
</tr>
<tr>
<td>E115</td>
<td>Axis module PLS7 does not exist</td>
<td>Hardware not connected</td>
</tr>
<tr>
<td>E116</td>
<td>PLS7 reports system error no.xxx</td>
<td>See table on page for further information</td>
</tr>
<tr>
<td>E117</td>
<td>PLS7 TIMEOUT</td>
<td>PLS7 card defective, exchange card</td>
</tr>
<tr>
<td>E118</td>
<td>Hardware configuration ASi Master 1</td>
<td>ASi master card defective or not connected</td>
</tr>
<tr>
<td>E119</td>
<td>Hardware configuration ASi Master 2</td>
<td>ASi master card defective or not connected</td>
</tr>
<tr>
<td>E120</td>
<td>Hardware configuration ASi Master 3</td>
<td>ASi master card defective or not connected</td>
</tr>
<tr>
<td>E121</td>
<td>Hardware configuration ASi Master 4</td>
<td>ASi master card defective or not connected</td>
</tr>
<tr>
<td>E122</td>
<td>Axis module PLS6 does not exist</td>
<td>Check card</td>
</tr>
</tbody>
</table>
### System error Hardware Part 2:

<table>
<thead>
<tr>
<th>Code</th>
<th>Error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E123</td>
<td>Axis module PLS8 does not exist</td>
<td>Check card</td>
</tr>
<tr>
<td>E124</td>
<td>PLS6 / PLS8 TIMEOUT on loading PIC</td>
<td>Hardware error in the PLS card, exchange card</td>
</tr>
<tr>
<td>E125</td>
<td>More than 16 axes detected</td>
<td>Check axis configuration</td>
</tr>
<tr>
<td>E126</td>
<td>Another axis type detected</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E127</td>
<td>New axis type detected</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E128</td>
<td>Different axis types detected</td>
<td>Different cards which are not permissible together are present in system.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>(PLS6 combined with PLS7 or servo-axis with axis number of a PLS7 card)</td>
</tr>
<tr>
<td>E129</td>
<td>PLS6 / PLS8 PIC not ready</td>
<td>Hardware error in the PLS card, exchange card</td>
</tr>
<tr>
<td>E130</td>
<td>PLS6- / PLS8 PIC error checksum</td>
<td>Hardware error in the PLS card, exchange card</td>
</tr>
<tr>
<td>E131</td>
<td>EURO TP1 IEF module 1 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E132</td>
<td>EURO TP1 IEF module 2 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E133</td>
<td>EURO TP2 IEF module 1 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E134</td>
<td>EURO TP2 IEF module 2 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E135</td>
<td>EURO TP3 IEF module 1 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E136</td>
<td>EURO TP3 IEF module 2 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E137</td>
<td>EURO TP4 IEF module 1 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E138</td>
<td>EURO TP4 IEF module 2 not possible</td>
<td>Module not permissible on this slot</td>
</tr>
<tr>
<td>E139</td>
<td>EURO TP1 IEF module 1 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E140</td>
<td>EURO TP1 IEF module 2 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E141</td>
<td>EURO TP2 IEF module 1 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E142</td>
<td>EURO TP2 IEF module 2 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E143</td>
<td>EURO TP3 IEF module 1 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
</tbody>
</table>
## System error Hardware Part 3:

<table>
<thead>
<tr>
<th>Code</th>
<th>Error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E144</td>
<td>EURO TP3 IEF module 2 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E145</td>
<td>EURO TP4 IEF module 1 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E146</td>
<td>EURO TP4 IEF module 2 change</td>
<td>Check hardware configuration, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E147</td>
<td>Permissible internal temperature of the device exceeded</td>
<td>See technical data</td>
</tr>
<tr>
<td>E148</td>
<td>Too many absolute positioning systems detected</td>
<td>More IEF SSI modules are connected than the number of axes</td>
</tr>
<tr>
<td>E149</td>
<td>Warning: discharge degree of the battery</td>
<td>Check battery and exchange if necessary</td>
</tr>
</tbody>
</table>

## System error RAM contents group:

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E150</td>
<td>System parameters outside range</td>
<td>&quot;Boot&quot; system parameters, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E151</td>
<td>Axis parameters outside range</td>
<td>&quot;Boot&quot; axis parameters, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E152</td>
<td>Horizontal checksum error in the ASCII program</td>
<td>Delete program, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
<tr>
<td>E153</td>
<td>Horizontal checksum error in the Code program</td>
<td>Delete program, &quot;boot&quot; PA-CONTROL if necessary</td>
</tr>
</tbody>
</table>

## Field bus error messages:

<table>
<thead>
<tr>
<th>Code</th>
<th>BUS error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E200</td>
<td>Profibus-DP error IO length</td>
<td>Incorrect configuration (GSD file?)</td>
</tr>
<tr>
<td>E201</td>
<td>Profibus-DP init error</td>
<td>Incorrect configuration (GSD file?)</td>
</tr>
<tr>
<td>E202</td>
<td>Profibus-DP unknown error</td>
<td>Incorrect configuration (GSD file?)</td>
</tr>
<tr>
<td>E203</td>
<td>RS232-CMD init error</td>
<td>RS 232 setting incorrect</td>
</tr>
<tr>
<td>E204</td>
<td>RS232-CMD hardware interrupt</td>
<td>Check transmission path (cable &amp; connector)</td>
</tr>
<tr>
<td>E205</td>
<td>Rotary switch outside range</td>
<td>Incorrect setting of the rotary switch</td>
</tr>
<tr>
<td>E206</td>
<td>Malfunction on reading the rotary switch</td>
<td>Hardware error (CPU, rotary switch)</td>
</tr>
</tbody>
</table>
### ASi system error and CMD error messages:

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E250</td>
<td>ASi : Error on execution of host job</td>
<td>ASi master could not execute the command</td>
</tr>
<tr>
<td>E251</td>
<td>ASi : No error on execution of host job</td>
<td>ASi master could execute the command</td>
</tr>
<tr>
<td>E252</td>
<td>ASi : Slave does not exist</td>
<td>Device with the address xx does not exist</td>
</tr>
<tr>
<td>E253</td>
<td>ASi : Slave address 0 exists</td>
<td>The address 0 is not permissible in automatic or online mode</td>
</tr>
<tr>
<td>E254</td>
<td>ASi : Slave address already in use</td>
<td>Selection of another address for programming</td>
</tr>
<tr>
<td>E255</td>
<td>ASi : The ASi slave cannot be programmed to address 0</td>
<td>Check error on execution of host job</td>
</tr>
<tr>
<td>E256</td>
<td>ASi : The address could not be assigned to the ASi slave</td>
<td>Check address</td>
</tr>
<tr>
<td>E257</td>
<td>ASi : Address could not be stored in the EEPROM of the ASi slave</td>
<td>Problem with the programming of the address, repeat operation</td>
</tr>
<tr>
<td>E258</td>
<td>ASi : Unknown error</td>
<td>Repeat</td>
</tr>
<tr>
<td>E259</td>
<td>ASi : Timeout for transfer of command</td>
<td>Check ASI master card</td>
</tr>
<tr>
<td>E260</td>
<td>ASi : No supply voltage</td>
<td>Check ASI power supply</td>
</tr>
<tr>
<td>E261</td>
<td>ASi: Slave 0 is here</td>
<td>Invalid address for automatic and online mode</td>
</tr>
<tr>
<td>E262</td>
<td>ASi: Configuration error</td>
<td>Slaves on the bus do not coincide with the stored configuration</td>
</tr>
</tbody>
</table>
## CANopen bus, Communication errors / Warnings Part 1

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E300</td>
<td>Bus offline</td>
<td>Too many bad CANopen frames, implement EMC-compatible system design</td>
</tr>
<tr>
<td>E301</td>
<td>Timeout for SDO communication with CANopen module (axis, IO module)</td>
<td>Axis switched off, CAN connector removed</td>
</tr>
<tr>
<td>E302</td>
<td>Error during SDO communication with CANopen module (axis, IO module)</td>
<td>Axis switched off, CAN connector removed</td>
</tr>
<tr>
<td>E303</td>
<td>Unknown warning at the axis</td>
<td>Device sets warning flag, but does not describe the error</td>
</tr>
<tr>
<td>E304</td>
<td>Unknown error at the axis</td>
<td>Device sets warning flag, but does not describe the error</td>
</tr>
<tr>
<td>E305</td>
<td>Axis does not answer the SYNC frame</td>
<td>Axis switched off, CAN connector removed</td>
</tr>
<tr>
<td>E306</td>
<td>Timeout at stop of an axis</td>
<td>Optimize parameter &quot;VEL0&quot;</td>
</tr>
<tr>
<td>E307</td>
<td>Axis not ready</td>
<td>Axis switched off, motor supply voltage unavailable</td>
</tr>
<tr>
<td>E308</td>
<td>Version of LV servoTEC operating system incorrect</td>
<td>The version of the operating system of LV-servoTEC is not up-to-date</td>
</tr>
<tr>
<td>E309</td>
<td>Different parameters in PA-CONTROL and LV-servoTEC</td>
<td>Match parameters</td>
</tr>
<tr>
<td>E310</td>
<td>Communication error with CANopen module (axis, IO module)</td>
<td>Axis switched off, CAN connector removed</td>
</tr>
<tr>
<td>E311</td>
<td>IO module does not answer NODE GUARD frame</td>
<td>IO module switched off, break in the cabling</td>
</tr>
<tr>
<td>E312</td>
<td>Index of the object does not exist</td>
<td>Incorrect command/parameter</td>
</tr>
<tr>
<td>E313</td>
<td>SUB-index of the object does not exist</td>
<td>Incorrect command/parameter</td>
</tr>
<tr>
<td>E314</td>
<td>Parameter value too high</td>
<td>Check parameter value in the file of the servoTEC parameters</td>
</tr>
<tr>
<td>E315</td>
<td>Parameter value too low</td>
<td>Check parameter value in the file of the servoTEC parameters</td>
</tr>
<tr>
<td>E316</td>
<td>Service parameters inconsistent</td>
<td>Error in the operating system</td>
</tr>
<tr>
<td>E317</td>
<td>Object may only be read</td>
<td>Error in the operating system</td>
</tr>
<tr>
<td>E318</td>
<td>Counter TX error too high</td>
<td>Bus malfunction (terminating resistor, cable length)</td>
</tr>
<tr>
<td>E319</td>
<td>Counter RX error too high</td>
<td>Bus malfunction (terminating resistor, cable length)</td>
</tr>
<tr>
<td>E320</td>
<td>&quot;SYNC message&quot; cannot be sent</td>
<td>Bus malfunctions,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Error in the operating system</td>
</tr>
<tr>
<td>E321</td>
<td>Not all CAN axes or CAN IO modules have been initialized</td>
<td>CAN device connected ?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Not switched on ?</td>
</tr>
<tr>
<td></td>
<td></td>
<td>at error ?</td>
</tr>
</tbody>
</table>
### CANopen bus, Communication errors / Warnings Part 2

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
</table>
| E322 | Timeout for the communication with the control console | • Control console connected?  
• Not switched on?  
• at error? |
| E323 | Emergency message received from IO module | • Power supply switched off  
• Error in IO module  
• Short circuit of an output  
Please note error code of the module, e.g.  
30xx-xx : Voltage, ...  
4xxx-xx : Temperature, ...  
81xx-xx : Communication, ...  
90xx-xx : External error, ... |
| E324 | CAN axis at "EMERGENCY STOP" on traversing | EMERGENCY STOP input of the axis actuated |
### CANopen bus error axis (servoTEC) Part 1

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E400</td>
<td>Unknown error, not detailed</td>
<td></td>
</tr>
<tr>
<td>E401</td>
<td>Error reset, error no longer present</td>
<td></td>
</tr>
<tr>
<td>E402</td>
<td>Generic error</td>
<td></td>
</tr>
<tr>
<td>E403</td>
<td>Not ready</td>
<td>e.g. initialization not yet completed, ready (BTB) signal is not available on switch-on</td>
</tr>
<tr>
<td>E404</td>
<td>Fault to ground (F22)</td>
<td>Only with 40/70 ampere devices</td>
</tr>
<tr>
<td>E405</td>
<td>No supply system, not ready (F16)</td>
<td>Controller enabling was present, although no supply voltage was present, check mains supply.</td>
</tr>
<tr>
<td>E406</td>
<td>Overvoltage DC link (F02)</td>
<td>Ballast performance limit was reached, use external ballast resistor with higher performance and alter ballast performance parameters. Supply voltage too high, use mains transformer.</td>
</tr>
<tr>
<td>E407</td>
<td>Undervoltage DC link (F05)</td>
<td>Supply voltage not available or too low for enabled servo amplifier, enable servo amplifier with at least 500ms delay.</td>
</tr>
<tr>
<td>E408</td>
<td>Loss of a phase of the supply system (F19)</td>
<td>Can be switched off for operation on two phases.</td>
</tr>
<tr>
<td>E409</td>
<td>Internal temperature exceeded (F13)</td>
<td>Upgrade ventilation</td>
</tr>
<tr>
<td>E410</td>
<td>Temperature of heat sink exceeded (F01)</td>
<td>Limit value permanently set at 80°C by the manufacturer. Upgrade ventilation</td>
</tr>
<tr>
<td>E411</td>
<td>Motor temperature exceeded (F06)</td>
<td>Thermostatic switch of motor has actuated, allow motor cool to down and check why motor is overheating. Connector of the feedback unit loose or feedback line interrupted, secure connector or replace feedback line.</td>
</tr>
<tr>
<td>E412</td>
<td>Error in auxiliary supply (F07)</td>
<td>The auxiliary voltage internally generated in the LV-servoTEC is defective, send LV-servoTEC to the manufacturer for repair.</td>
</tr>
</tbody>
</table>
CANopen bus error axis (servoTEC) Part 2

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E413</td>
<td>Error A-D converter (F17)</td>
<td>Error in the A-D conversion, usually an EMC disturbance, reduce EMV disturbances and check screen and grounding Possibly also hardware defect of the regulator</td>
</tr>
<tr>
<td>E414</td>
<td>Output stage error (F14)</td>
<td>Motor supply cable has a short circuit, exchange cable Motor has a ground fault or short circuit, exchange motor Output stage module has overheated; upgrade ventilation Defect of the output stage module Send servo amplifier to the manufacturer for repair Ground fault or short circuit in the circuit of the external ballast resistor, eliminate ground fault or short circuit.</td>
</tr>
<tr>
<td>E415</td>
<td>Error ballast resistor (F18)</td>
<td>Ballast circuit defective or setting incorrect</td>
</tr>
<tr>
<td>E416</td>
<td>Error in application of the AS option (F27)</td>
<td>Activation of the AS option and the ENABLE signal are present at the same time</td>
</tr>
<tr>
<td>E417</td>
<td>Checksum serial EEPROM (F09)</td>
<td>Might be remedied by re-storage of the parameters (SAVE)</td>
</tr>
<tr>
<td>E418</td>
<td>Flash EEPROM (F10)</td>
<td>Checksum error</td>
</tr>
<tr>
<td>E419</td>
<td>Watchdog caused software reset (F32)</td>
<td>System software does not react correctly</td>
</tr>
<tr>
<td>E420</td>
<td>BCC error (Table)</td>
<td>1)</td>
</tr>
<tr>
<td>E421</td>
<td>BCC error, (system macro)</td>
<td>E420-E426: lead to F32, causes of the various breakdown of the various causes only through output via the serial port</td>
</tr>
<tr>
<td>E422</td>
<td>BCC error, (EEPROM serial)</td>
<td></td>
</tr>
<tr>
<td>E423</td>
<td>FPGA error</td>
<td></td>
</tr>
<tr>
<td>E424</td>
<td>Error (Table)</td>
<td></td>
</tr>
<tr>
<td>E425</td>
<td>User software BCC</td>
<td></td>
</tr>
<tr>
<td>E426</td>
<td>Defective user software BCC (F32)</td>
<td>System software does not react correctly</td>
</tr>
<tr>
<td>E427</td>
<td>Parameter error</td>
<td>Incorrect parameter value via PDO</td>
</tr>
</tbody>
</table>

1) Defective drive firmware, might be fixed by download of firmware.

The error codes F01 to F32 in brackets correspond to error messages of LV-servoTEC. See the LV-servoTEC operating instructions, Chap. 6.3 Error messages.
## CANopen bus error axis (servoTEC) Part 3

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E428</td>
<td>Error brake (F11)</td>
<td>Cable break, short circuit or ground fault of the brake line</td>
</tr>
<tr>
<td>E429</td>
<td>Commutation error (F25)</td>
<td>Motor commutation not correct, check motor cable, check parameters MPHASE, VCOMM</td>
</tr>
<tr>
<td>E430</td>
<td>LV-servoTEC could not be enabled</td>
<td>HW Enable unavailable, or error status of the servo</td>
</tr>
<tr>
<td>E431</td>
<td>Command only allowed if LV-servoTEC not enabled</td>
<td></td>
</tr>
<tr>
<td>E432</td>
<td>Defective feedback unit (F04)</td>
<td>Cable break, short circuit or ground fault</td>
</tr>
<tr>
<td>E433</td>
<td>Handling error (F21)</td>
<td>Software error of the expansion card</td>
</tr>
<tr>
<td>E434</td>
<td>Response monitoring</td>
<td>Node-guarding error, can be reset via control word</td>
</tr>
<tr>
<td>E435</td>
<td>CANopen bus not ready</td>
<td>Faulty CAN bus communication?</td>
</tr>
<tr>
<td>E436</td>
<td>State machine is not operation-enabled</td>
<td>Control word used incorrectly</td>
</tr>
<tr>
<td>E437</td>
<td>Incorrect mode</td>
<td>Preselect correct mode</td>
</tr>
<tr>
<td>E438</td>
<td>Torque setting incorrect (F15)</td>
<td>Preset RMS current value exceeded, e.g. as a result of a stiff mechanical system, check mechanical system</td>
</tr>
<tr>
<td>E439</td>
<td>Overspeed (F08)</td>
<td>Set overspeed reached, motor has possibly run away</td>
</tr>
<tr>
<td>E440</td>
<td>Tracking error (F03)</td>
<td>Message of the position controller</td>
</tr>
<tr>
<td>E441</td>
<td>Invalid traverse job number started</td>
<td>Selected traversing set was not defined</td>
</tr>
<tr>
<td>E442</td>
<td>Error external trajectory (F28)</td>
<td>Setpoint step change too large Only in SERCOS systems</td>
</tr>
<tr>
<td>E443</td>
<td>Fatal exception error (F32)</td>
<td>System software does not react correctly</td>
</tr>
<tr>
<td>E444</td>
<td>Error in a PDO component</td>
<td>Incorrect value</td>
</tr>
<tr>
<td>E445</td>
<td>Incorrect operating mode</td>
<td>Only for Rx-PDO 22</td>
</tr>
<tr>
<td>E446</td>
<td>Slot error (F20)</td>
<td>Error of the expansion card Only in servos with slot cards</td>
</tr>
<tr>
<td>E447</td>
<td>Warning display as error (F24)</td>
<td>Warning display is evaluated as an error (user-defined)</td>
</tr>
<tr>
<td>E448</td>
<td>Error in approach to reference point (F26)</td>
<td>Hardware limit switch reached</td>
</tr>
<tr>
<td>E449</td>
<td>Serco error (F29)</td>
<td>Only in SERCOS systems</td>
</tr>
<tr>
<td>E450</td>
<td>Sercoes</td>
<td>Only in SERCOS systems</td>
</tr>
</tbody>
</table>

The error codes F01 to F32 in brackets correspond to error messages of LV-servoTEC. See the LV-servoTEC operating instructions, Chap. 6.3 Error messages.
### CANopen bus error axis (servoTEC) Part 4

<table>
<thead>
<tr>
<th>Code</th>
<th>Hardware error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E460</td>
<td>I²T message threshold crossed</td>
<td>If necessary, reduce motor current, acceleration or traversing speed</td>
</tr>
<tr>
<td>E461</td>
<td>Ballast performance reached</td>
<td></td>
</tr>
<tr>
<td>E462</td>
<td>Tracking error</td>
<td>Change drive parameter &quot;tracking error&quot; or, if necessary, reduce motor current, acceleration or traversing speed</td>
</tr>
<tr>
<td>E463</td>
<td>Response monitoring active</td>
<td></td>
</tr>
<tr>
<td>E464</td>
<td>No mains phase</td>
<td>Check mains wiring or deactivate mains monitoring (PMODE)</td>
</tr>
<tr>
<td>E465</td>
<td>Software limit switch 1 has actuated</td>
<td></td>
</tr>
<tr>
<td>E466</td>
<td>Software limit switch 2 has actuated</td>
<td></td>
</tr>
<tr>
<td>E467</td>
<td>Bad traversing job started</td>
<td>Error in the operating system</td>
</tr>
<tr>
<td>E468</td>
<td>No reference point</td>
<td>Reference axis (G25…)</td>
</tr>
<tr>
<td>E469</td>
<td>Positive limit switch active</td>
<td></td>
</tr>
<tr>
<td>E470</td>
<td>Negative limit switch active</td>
<td></td>
</tr>
<tr>
<td>E471</td>
<td>Motor default values have been loaded</td>
<td>Setting of the motor parameters not yet complete</td>
</tr>
<tr>
<td>E472</td>
<td>Expansion card is not functioning correctly</td>
<td></td>
</tr>
<tr>
<td>E473</td>
<td>Motor phase</td>
<td></td>
</tr>
<tr>
<td>E474</td>
<td>Incorrect VCT entry</td>
<td></td>
</tr>
<tr>
<td>E475</td>
<td>Warning n17 – n31</td>
<td>Display warning using the WINPAC-program Diagnosis -&gt; servoTEC -&gt; error statistics</td>
</tr>
<tr>
<td>E481</td>
<td>Positive limit switch actuated</td>
<td></td>
</tr>
<tr>
<td>E482</td>
<td>Short circuit</td>
<td>Motor cable defective, motor defective</td>
</tr>
<tr>
<td>E483</td>
<td>Voltage for stepping motor output stage incorrect</td>
<td>230VAC not connected to the supply? Bridge inserted in the motor connector? Motor power supply defective?</td>
</tr>
<tr>
<td>E484</td>
<td>Auxiliary voltages incorrect</td>
<td>Power supply defective</td>
</tr>
<tr>
<td>E485</td>
<td>Temperature in the heat sink too high</td>
<td>Ventilation defective, ventilation clogged</td>
</tr>
<tr>
<td>E486</td>
<td>Motor rotation monitoring was actuated</td>
<td>Motor run against mechanical stop, motor acceleration too high, rotary encoder wiring</td>
</tr>
<tr>
<td>E487</td>
<td>Phase sequence monitoring! – no motor</td>
<td>Check motor wiring</td>
</tr>
</tbody>
</table>
## Error messages in automatic mode Part 1:

<table>
<thead>
<tr>
<th>Code</th>
<th>Run error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E501</td>
<td>External stop input not defined</td>
<td>Parameter</td>
</tr>
<tr>
<td>E502</td>
<td>External stop input cold</td>
<td>Stop input not connected ?</td>
</tr>
<tr>
<td>E503</td>
<td>STOP program unavailable</td>
<td>Run definition</td>
</tr>
<tr>
<td>E504</td>
<td>START-AFTER-STOP prog. unavailable</td>
<td>Run definition</td>
</tr>
<tr>
<td>E505</td>
<td>MALFUNCTION program unavailable</td>
<td>Run definition</td>
</tr>
<tr>
<td>E506</td>
<td>STOP program illegal command</td>
<td>Only use permissible commands (see Chapter 3)</td>
</tr>
<tr>
<td>E507</td>
<td>START after STOP prog. illegal command</td>
<td>Only use permissible commands (see Chapter 3)</td>
</tr>
<tr>
<td>E508</td>
<td>MALFUNCTION program illegal command</td>
<td>Only use permissible commands (see Chapter 3)</td>
</tr>
<tr>
<td>E509</td>
<td>Reserve</td>
<td></td>
</tr>
<tr>
<td>E510</td>
<td>Reserve</td>
<td></td>
</tr>
<tr>
<td>E511</td>
<td>Start program not defined</td>
<td>Run definition</td>
</tr>
<tr>
<td>E512</td>
<td>Start program unavailable</td>
<td>Run definition</td>
</tr>
<tr>
<td>E513</td>
<td>Program unavailable</td>
<td>Run definition</td>
</tr>
<tr>
<td>E514</td>
<td>Unknown command</td>
<td>Syntax error</td>
</tr>
<tr>
<td>E515</td>
<td>Nesting depth too large</td>
<td>Too many subroutines</td>
</tr>
<tr>
<td>E516</td>
<td>Too many processes started</td>
<td>More than 31 programs opened</td>
</tr>
<tr>
<td>E517</td>
<td>Value too low</td>
<td>Value less than e.g. the min. traverse</td>
</tr>
<tr>
<td>E518</td>
<td>Value too high</td>
<td>Value greater than e.g. the max. traverse</td>
</tr>
<tr>
<td>E519</td>
<td>Flag number too low</td>
<td>Value outside range 1-1024</td>
</tr>
<tr>
<td>E520</td>
<td>Flag number too high</td>
<td>Value outside range 1-1024</td>
</tr>
<tr>
<td>E521</td>
<td>Register number too low</td>
<td>Value outside range 1-1024</td>
</tr>
<tr>
<td>E522</td>
<td>Register number too high</td>
<td>Value outside range 1-1024</td>
</tr>
<tr>
<td>E523</td>
<td>Axis still running</td>
<td>Traversing command is still being processed</td>
</tr>
<tr>
<td>E524</td>
<td>Axis overrun</td>
<td>Axis outside traversing range</td>
</tr>
<tr>
<td>E525</td>
<td>No axis reference point</td>
<td>Reference switch defective?</td>
</tr>
<tr>
<td>E526</td>
<td>Data channel not initialized</td>
<td>Port available ?</td>
</tr>
<tr>
<td>E527</td>
<td>Display unavailable</td>
<td>IEF control terminal available</td>
</tr>
<tr>
<td>E528</td>
<td>Command at the data channel not possible</td>
<td>Port available ?</td>
</tr>
<tr>
<td>E529</td>
<td>Positive limit switch actuated</td>
<td>Switch actuated or cable break ?</td>
</tr>
<tr>
<td>E530</td>
<td>Negative limit switch actuated</td>
<td>Switch actuated or cable break ?</td>
</tr>
</tbody>
</table>
## Error messages in automatic mode Part 2:

<table>
<thead>
<tr>
<th>Code</th>
<th>Run error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E531</td>
<td>Limit switch actuated</td>
<td>Switch actuated or cable break ?</td>
</tr>
<tr>
<td>E532</td>
<td>Value outside range</td>
<td>Value outside traversing range</td>
</tr>
<tr>
<td>E533</td>
<td>System error, cause unknown</td>
<td>Error message of the operating system</td>
</tr>
<tr>
<td>E534</td>
<td>Profibus-DP communication switched off</td>
<td>Interruption of Profibus (connector, cable)</td>
</tr>
<tr>
<td>E535</td>
<td>AP traversing speed too high</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E536</td>
<td>AP traversing speed too low</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E537</td>
<td>AP reference speed too high</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E538</td>
<td>AP reference speed too low</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E539</td>
<td>AP manual speed too high</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E540</td>
<td>AP manual speed too low</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E541</td>
<td>AP creep speed too high</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E542</td>
<td>AP creep speed too low</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E543</td>
<td>AP Start-Stop speed too high</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E544</td>
<td>AP Start-Stop speed too low</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E545</td>
<td>AP acceleration too high</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E546</td>
<td>AP acceleration too low</td>
<td>Check setting of the parameters</td>
</tr>
<tr>
<td>E547</td>
<td>SLEEP program not running</td>
<td>Program is not being executed at present</td>
</tr>
<tr>
<td>E548</td>
<td>Assignment of axis type for interpolation</td>
<td>Interpolation not possible with this hardware (only with PLS7, PLS9)</td>
</tr>
<tr>
<td>E549</td>
<td>PLS7 negative acknowledgement at card command</td>
<td>Hardware error</td>
</tr>
<tr>
<td>E550</td>
<td>PLS7 incorrect card mode</td>
<td>Hardware error</td>
</tr>
<tr>
<td>E551</td>
<td>COM port busy</td>
<td>COM1 used as RS232-OnlineCMD</td>
</tr>
<tr>
<td>E552</td>
<td>Axes already used by interpolation</td>
<td>Interpolation command uses all four axes of this PLS card during interpolation</td>
</tr>
<tr>
<td>E553</td>
<td>IPO – path traversing speed too high</td>
<td>Check parameters</td>
</tr>
<tr>
<td>E554</td>
<td>IPO – path traversing speed too low</td>
<td>Check parameters</td>
</tr>
<tr>
<td>E555</td>
<td>IPO – path Start-Stop speed too high</td>
<td>Check parameters</td>
</tr>
<tr>
<td>E556</td>
<td>IPO – path Start-Stop speed too low</td>
<td>Check parameters</td>
</tr>
<tr>
<td>E557</td>
<td>IPO – path acceleration too high</td>
<td>Check parameters</td>
</tr>
<tr>
<td>E558</td>
<td>IPO – path acceleration too low</td>
<td>Check parameters</td>
</tr>
<tr>
<td>E559</td>
<td>Input field too small</td>
<td>Error in user program</td>
</tr>
<tr>
<td>E560</td>
<td>Rotary encoder error on synchronization</td>
<td>Hardware error</td>
</tr>
<tr>
<td>Code</td>
<td>Run error message</td>
<td>Further information - Troubleshooting</td>
</tr>
<tr>
<td>------</td>
<td>-------------------</td>
<td>--------------------------------------</td>
</tr>
<tr>
<td>E561</td>
<td>Program already running</td>
<td>Program can only be started once</td>
</tr>
<tr>
<td>E562</td>
<td>RS232 communication was disconnected</td>
<td>Check cabling, handshake lines</td>
</tr>
<tr>
<td>E563</td>
<td>Manuell approach to reference point: positive limit switch actuated</td>
<td>Rotational and limit switch direction incorrect</td>
</tr>
<tr>
<td>E564</td>
<td>Incorrect program type</td>
<td>Call of an incorrect program (PTX)</td>
</tr>
<tr>
<td>E565</td>
<td>INTERBUS_S emergency stop</td>
<td>INTERBUS_S was disconnected</td>
</tr>
<tr>
<td>E566</td>
<td>Axis not ready</td>
<td>Check standby power section</td>
</tr>
<tr>
<td>E567</td>
<td>Increments of axis range greater than 31 bits</td>
<td>Check definition of traversing range</td>
</tr>
<tr>
<td>E568</td>
<td>Approach to reference point not possible</td>
<td>Check servoTEC communication</td>
</tr>
<tr>
<td>E569</td>
<td>STORE command executed incorrectly</td>
<td>Check placeholder for characters in the program</td>
</tr>
<tr>
<td>E570</td>
<td>Number of revolutions greater than maximum</td>
<td>Check definition of traversing range</td>
</tr>
</tbody>
</table>

*Formula for calculation of the speed N:*
\[
N = \frac{\text{traversing range max} - \text{traversing range min}}{\text{PGEARI}} \times \text{Gearfactor} 
\]

*(PGEARI – servoTEC parameters)*

**IMPORTANT:** The following applies up to version V4.72

No. of rev. < 1024

| E571 | Command not possible with this axis | Program error |
| E572 | Reserve, not assigned at present | |
| E573 | Any CANopen device SDO TIME-OUT | Check whether CANopen device connected |
| E574 | String too long | The result string would be longer than 80 characters if strings are added |
| E575 | Traversing speed greater than 16 bits | The value would be greater than a 16 bit value in the case of a traversing command for a servoTEC axis |
| E576 | Value cannot be read stably | |
| E577 | Incorrect jump target | The commands JMP-LINE or JMP-LINE-IPO jump to an invalid program line |
| E578 | Traverse of the servoTEC axis too long | The traverse for a traversing command is limited to a number of revolutions

- Up to V4.73I : 1022 revolutions
- from 4.74 or higher |
### Error messages in automatic mode Part 4:

<table>
<thead>
<tr>
<th>Code</th>
<th>Run error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
</table>
| E579 | Command not allowed in the initial position program | A command is to be executed in the “initial position” program which is not enabled for this program type e.g:  
  - A1:=  
  - RUN |
## Communication errors:

<table>
<thead>
<tr>
<th>Code</th>
<th>Run error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E600</td>
<td>IEF module slot number not allowed</td>
<td>Insert module in permissible slot</td>
</tr>
<tr>
<td>E601</td>
<td>COM port number not allowed</td>
<td>Programming error</td>
</tr>
<tr>
<td>E602</td>
<td>COM port not installed</td>
<td>Addressed COM port unavailable</td>
</tr>
<tr>
<td>E603</td>
<td>COM port not initialized</td>
<td>Programming error</td>
</tr>
<tr>
<td>E604</td>
<td>COM port not empty for sending</td>
<td>Previous transmission not yet ended</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programming error, Port disabled, hardware handshake</td>
</tr>
<tr>
<td>E605</td>
<td>COM port hardware handshake disabled</td>
<td>Check parameter setting and device</td>
</tr>
<tr>
<td>E606</td>
<td>COM port data format incorrect</td>
<td>Parameter setting</td>
</tr>
<tr>
<td>E607</td>
<td>COM port hardware not empty for sending</td>
<td>Previous transmission not yet ended</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Programming error, Port disabled, hardware handshake</td>
</tr>
</tbody>
</table>
Errors of the absolute dimension system:

<table>
<thead>
<tr>
<th>Code</th>
<th>Run error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E620</td>
<td>Absolute positioning system unavailable</td>
<td>SSI hardware error</td>
</tr>
<tr>
<td>E621</td>
<td>Absolute positioning system not referenced</td>
<td>Carry out approach to reference point</td>
</tr>
<tr>
<td>E622</td>
<td>Reserve, not assigned at present</td>
<td></td>
</tr>
<tr>
<td>E623</td>
<td>Absolute positioning system SSI module not ready</td>
<td>Check SSI module (connection and wiring)</td>
</tr>
<tr>
<td>E624</td>
<td>Absolute positioning system DIN-T0 error</td>
<td>Parameter error</td>
</tr>
<tr>
<td>E625</td>
<td>Absolute positioning system DIN-Tn+ error</td>
<td>Parameter error</td>
</tr>
<tr>
<td>E626</td>
<td>Absolute positioning system initialization timeout error</td>
<td>Check SSI module (connection and wiring)</td>
</tr>
</tbody>
</table>

CANopen LV-servoTEC error:

<table>
<thead>
<tr>
<th>Code</th>
<th>Run error message</th>
<th>Further information - Troubleshooting</th>
</tr>
</thead>
<tbody>
<tr>
<td>E900</td>
<td>Different parameters in PA-CONTROL and LV-servoTEC</td>
<td>Adjust parameter value</td>
</tr>
<tr>
<td>E901</td>
<td>Different obligatory parameters in PA-CONTROL and LV-servoTEC</td>
<td>Adjust parameter value</td>
</tr>
<tr>
<td>E902</td>
<td>Different reference parameters in PA-CONTROL and LV-servoTEC</td>
<td>Adjust parameter value</td>
</tr>
</tbody>
</table>
### PLS7- Run error message:

<table>
<thead>
<tr>
<th>Code</th>
<th>CPU-Run error message</th>
<th>Weitere Informationen - Fehlerbeseitigung</th>
</tr>
</thead>
<tbody>
<tr>
<td>2</td>
<td>Busfehler</td>
<td>Change the PLS7-card</td>
</tr>
<tr>
<td>3</td>
<td>Adressfehler</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>nicht implementierter befehl</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>Division durch Null</td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>Befehlcheck</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>Privilegverletzung</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>Trace Trap</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>Emulator 1</td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>Emulator 2</td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>Reserviert</td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>nicht initialisierter Interrupt</td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>falscher Interrupt</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>Autovektor 1 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>Autovektor 2 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>Autovektor 3 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>Autovektor 4 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>Autovektor 5 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>Autovektor 6 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>107</td>
<td>Autovektor 7 nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>„NON Autovektor“ nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>„Trap-Vektor“ nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>Co-Prozessor-Vektor nicht initialisiert</td>
<td></td>
</tr>
<tr>
<td>401</td>
<td>Generator 1 laden “Time out”</td>
<td></td>
</tr>
<tr>
<td>402</td>
<td>Generator 2 laden “Time out”</td>
<td></td>
</tr>
<tr>
<td>403</td>
<td>Generator 3 laden “Time out”</td>
<td></td>
</tr>
<tr>
<td>404</td>
<td>Generator 4 laden “Time out”</td>
<td></td>
</tr>
<tr>
<td>501</td>
<td>Quersummenfehler Generator 1 laden</td>
<td></td>
</tr>
<tr>
<td>502</td>
<td>Quersummenfehler Generator 2 laden</td>
<td></td>
</tr>
<tr>
<td>503</td>
<td>Quersummenfehler Generator 3 laden</td>
<td></td>
</tr>
<tr>
<td>504</td>
<td>Quersummenfehler Generator 4 laden</td>
<td></td>
</tr>
<tr>
<td>999</td>
<td>Software-Watchdog</td>
<td></td>
</tr>
</tbody>
</table>
7.2 Basic board

The PA-CONTROL CPU5 is implemented on a printed circuit board.

Fig. 93: CPU5, position of the plug-in connections

Fig. 94: CPU5, IEF modules and inputs and outputs

The functional scope can be extended by means of various IEF modules. The PA-CONTROL servoTEC can only be expanded by the modules Profibus-DP, A-D converter and RS232. These modules are connected to the CPU5 via connector locations X25 or X26. Please refer to Chapter 6 of this documentation from the section RS232_Port Driver for further information on the IEF modules.
7.2.1 Voltage supply

The CPU5 is supplied with 24VDC via the connector X24, the plug-in connection for the outputs of the CPU5, which is reverse polarity protected. The supply voltages for the CPU and the CANopen bus are generated in the CPU5. All voltages are galvanically isolated from each other.

The 24VDC supply is fuse-protected on the CPU board.

7.3 Connector assignments

7.3.1 Inputs and outputs

The inputs can be polled and the outputs changed by means of the WINPAC program (Menu items Settings / PA-CONTROL / System I/O).
### 7.4 List of accessories and spare parts

<table>
<thead>
<tr>
<th>No.</th>
<th>Description</th>
<th>IEF No. (Standard length)</th>
<th>IEF No. (Special length)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Interface cable PC ↔ PA-CONTROL</td>
<td>231 766</td>
<td>-</td>
</tr>
<tr>
<td>2</td>
<td>IEF control console</td>
<td>231 764</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Port for IEF control console</td>
<td>527 452</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>Interface cable IEF control console</td>
<td>231 766</td>
<td>1002839</td>
</tr>
<tr>
<td>3</td>
<td>WINPAC (prog. development system)</td>
<td>231 726</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>Software update PA-CONTROL</td>
<td>230 714</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Option RS 232 port</td>
<td>527 450</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Option Profibus-DP interface</td>
<td>527 184</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Option A-D converter</td>
<td>527 451</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Option IEF control console</td>
<td>231 764</td>
<td></td>
</tr>
</tbody>
</table>
7.5 Service

CAUTION

The controller must be deenergized when carrying out assembly, disassembly and repair work or replacing individual components. Safety instructions must be followed to the letter (See Chapter 4 "Start-Up").

7.5.1 Exchange of the battery

The battery is used to buffer the static RAM. A soldered p.c.b. battery is used ex works. The service life of this battery is approx. 5 years.

Procedure:

- Order an exchange CPU5 board from IEF Werner (new battery)
- Carry out a data backup (programs and parameters) by means of the WINPAC program
- Switch off the device and remove the power plug
- Exchange the CPU5 board
- Switch on the device and follow the instructions on the two-line display, boot if required
- Transfer the program and parameters from the PC using WINPAC (download)

Remark:

Your own service personnel can exchange the battery on the CPU5 board, of course. We do not recommend this, however. The basic procedure corresponds to the above description.
7.6 Service

**CAUTION**

The controller must be deenergized when carrying out assembly, disassembly and repair work or replacing individual components. Safety instructions must be followed to the letter (See Chapter 4 "Start-Up").

7.6.1 Exchange of the battery

The battery is used to buffer the static RAM. A soldered p.c.b. battery is used ex works. The service life of this battery is approx. 5 years.

**Procedure:**

- Order an exchange CPU5 board from IEF Werner (new battery)
- Carry out a data backup (programs and parameters) by means of the WINPAC program
- Switch off the device and remove the power plug
- Exchange the CPU5 board
- Switch on the device and follow the instructions on the two-line display, boot if required
- Transfer the program and parameters from the PC using WINPAC (download)

**Remark:**

Your own service personnel can exchange the battery on the CPU5 board, of course. We do not recommend this, however. The basic procedure corresponds to the above description.
7.7 PA-CONTROL key code (+ ASCII character set to ISO/IEC 8859-15)

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hex</th>
<th>Character</th>
<th>PAC Keyboard</th>
<th>PAC Keyboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>NUL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>SOH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>2</td>
<td>STX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>ETX</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>4</td>
<td>EOT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>5</td>
<td>ENQ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>6</td>
<td>ACK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>7</td>
<td>BEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>8</td>
<td>BS</td>
<td>Shift+DEL</td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>9</td>
<td>HT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>A</td>
<td>LF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>B</td>
<td>VT</td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>C</td>
<td>FF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>D</td>
<td>CR</td>
<td>ENTER</td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>E</td>
<td>SO</td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>F</td>
<td>SI</td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>10</td>
<td>DLE</td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>11</td>
<td>DC1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>12</td>
<td>DC2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>13</td>
<td>DC3</td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>14</td>
<td>DC4</td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>15</td>
<td>NAK</td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>16</td>
<td>SYN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>17</td>
<td>ETB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>18</td>
<td>CAN</td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>19</td>
<td>EM</td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>1A</td>
<td>SUB</td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>1B</td>
<td>ESC</td>
<td>ESC/Shift+ESC</td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>1C</td>
<td>FS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>1D</td>
<td>GS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>1E</td>
<td>RS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>1F</td>
<td>US</td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>20</td>
<td>SP</td>
<td>SPACE BAR</td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>21</td>
<td>!</td>
<td>Shift+1</td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>22</td>
<td>_</td>
<td>Shift+2</td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>23</td>
<td>#</td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>24</td>
<td>$</td>
<td>Shift+4</td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>25</td>
<td>%</td>
<td>Shift+5</td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>26</td>
<td>&amp;</td>
<td>Shift+6</td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>27</td>
<td>'</td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>28</td>
<td>(</td>
<td>Shift+8</td>
<td></td>
</tr>
<tr>
<td>41</td>
<td>29</td>
<td>)</td>
<td>Shift+9</td>
<td></td>
</tr>
<tr>
<td>42</td>
<td>2A</td>
<td>*</td>
<td>Shift+*</td>
<td></td>
</tr>
<tr>
<td>43</td>
<td>2B</td>
<td>+</td>
<td>+</td>
<td></td>
</tr>
<tr>
<td>44</td>
<td>2C</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>45</td>
<td>2D</td>
<td>-</td>
<td>-</td>
<td></td>
</tr>
<tr>
<td>46</td>
<td>2E</td>
<td>.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dec</td>
<td>Hex</td>
<td>Character</td>
<td>PAC Keyboard</td>
<td></td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-----------</td>
<td>--------------</td>
<td></td>
</tr>
<tr>
<td>94</td>
<td>5E</td>
<td>^</td>
<td></td>
<td></td>
</tr>
<tr>
<td>95</td>
<td>5F</td>
<td>_</td>
<td>Shift+ -</td>
<td></td>
</tr>
<tr>
<td>96</td>
<td>60</td>
<td>‘</td>
<td></td>
<td></td>
</tr>
<tr>
<td>97</td>
<td>61</td>
<td>a</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>98</td>
<td>62</td>
<td>b</td>
<td>B</td>
<td></td>
</tr>
<tr>
<td>99</td>
<td>63</td>
<td>c</td>
<td>C</td>
<td></td>
</tr>
<tr>
<td>100</td>
<td>64</td>
<td>d</td>
<td>D</td>
<td></td>
</tr>
<tr>
<td>101</td>
<td>65</td>
<td>e</td>
<td>E</td>
<td></td>
</tr>
<tr>
<td>102</td>
<td>66</td>
<td>f</td>
<td>F</td>
<td></td>
</tr>
<tr>
<td>103</td>
<td>67</td>
<td>g</td>
<td>G</td>
<td></td>
</tr>
<tr>
<td>104</td>
<td>68</td>
<td>h</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>105</td>
<td>69</td>
<td>i</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>106</td>
<td>6A</td>
<td>j</td>
<td>J</td>
<td></td>
</tr>
<tr>
<td>107</td>
<td>6B</td>
<td>k</td>
<td>K</td>
<td></td>
</tr>
<tr>
<td>108</td>
<td>6C</td>
<td>l</td>
<td>L</td>
<td></td>
</tr>
<tr>
<td>109</td>
<td>6D</td>
<td>m</td>
<td>M</td>
<td></td>
</tr>
<tr>
<td>110</td>
<td>6E</td>
<td>n</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>6F</td>
<td>o</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>70</td>
<td>p</td>
<td>P</td>
<td></td>
</tr>
<tr>
<td>113</td>
<td>71</td>
<td>q</td>
<td>Q</td>
<td></td>
</tr>
<tr>
<td>114</td>
<td>72</td>
<td>r</td>
<td>R</td>
<td></td>
</tr>
<tr>
<td>115</td>
<td>73</td>
<td>s</td>
<td>S</td>
<td></td>
</tr>
<tr>
<td>116</td>
<td>74</td>
<td>t</td>
<td>T</td>
<td></td>
</tr>
<tr>
<td>117</td>
<td>75</td>
<td>u</td>
<td>U</td>
<td></td>
</tr>
<tr>
<td>118</td>
<td>76</td>
<td>v</td>
<td>V</td>
<td></td>
</tr>
<tr>
<td>119</td>
<td>77</td>
<td>w</td>
<td>W</td>
<td></td>
</tr>
<tr>
<td>120</td>
<td>78</td>
<td>x</td>
<td>X</td>
<td></td>
</tr>
<tr>
<td>121</td>
<td>79</td>
<td>y</td>
<td>Y</td>
<td></td>
</tr>
<tr>
<td>122</td>
<td>7A</td>
<td>z</td>
<td>Z</td>
<td></td>
</tr>
<tr>
<td>123</td>
<td>7B</td>
<td>{</td>
<td>Ctrl+7</td>
<td></td>
</tr>
<tr>
<td>124</td>
<td>7C</td>
<td></td>
<td></td>
<td>Ctrl+0</td>
</tr>
<tr>
<td>125</td>
<td>7D</td>
<td>}</td>
<td></td>
<td></td>
</tr>
<tr>
<td>126</td>
<td>7E</td>
<td>~</td>
<td></td>
<td></td>
</tr>
<tr>
<td>127</td>
<td>7F</td>
<td>DEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>128</td>
<td>80</td>
<td>PAD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>129</td>
<td>81</td>
<td>HOP</td>
<td></td>
<td></td>
</tr>
<tr>
<td>130</td>
<td>82</td>
<td>BPH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>131</td>
<td>83</td>
<td>NBH</td>
<td></td>
<td></td>
</tr>
<tr>
<td>132</td>
<td>84</td>
<td>IND</td>
<td></td>
<td></td>
</tr>
<tr>
<td>133</td>
<td>85</td>
<td>NEL</td>
<td></td>
<td></td>
</tr>
<tr>
<td>134</td>
<td>86</td>
<td>SSA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>135</td>
<td>87</td>
<td>ESA</td>
<td></td>
<td></td>
</tr>
<tr>
<td>136</td>
<td>88</td>
<td>HTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>137</td>
<td>89</td>
<td>HTJ</td>
<td></td>
<td></td>
</tr>
<tr>
<td>138</td>
<td>8A</td>
<td>VTS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>139</td>
<td>8B</td>
<td>PLD</td>
<td></td>
<td></td>
</tr>
<tr>
<td>140</td>
<td>8C</td>
<td>PLU</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hex</th>
<th>Character</th>
<th>PAC Keyboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>141</td>
<td>8D</td>
<td>RI</td>
<td></td>
</tr>
<tr>
<td>142</td>
<td>8E</td>
<td>SS2</td>
<td></td>
</tr>
<tr>
<td>143</td>
<td>8F</td>
<td>SS3</td>
<td></td>
</tr>
<tr>
<td>144</td>
<td>90</td>
<td>DCs</td>
<td></td>
</tr>
<tr>
<td>145</td>
<td>91</td>
<td>PU1</td>
<td></td>
</tr>
<tr>
<td>146</td>
<td>92</td>
<td>PU2</td>
<td></td>
</tr>
<tr>
<td>147</td>
<td>93</td>
<td>STS</td>
<td></td>
</tr>
<tr>
<td>148</td>
<td>94</td>
<td>CCH</td>
<td></td>
</tr>
<tr>
<td>149</td>
<td>95</td>
<td>MW</td>
<td></td>
</tr>
<tr>
<td>150</td>
<td>96</td>
<td>SPA</td>
<td></td>
</tr>
<tr>
<td>151</td>
<td>97</td>
<td>EPA</td>
<td></td>
</tr>
<tr>
<td>152</td>
<td>98</td>
<td>SOS</td>
<td></td>
</tr>
<tr>
<td>153</td>
<td>99</td>
<td>SGC1</td>
<td></td>
</tr>
<tr>
<td>154</td>
<td>9A</td>
<td>SCI</td>
<td></td>
</tr>
<tr>
<td>155</td>
<td>9B</td>
<td>CSI</td>
<td></td>
</tr>
<tr>
<td>156</td>
<td>9C</td>
<td>ST</td>
<td></td>
</tr>
<tr>
<td>157</td>
<td>9D</td>
<td>OSC</td>
<td></td>
</tr>
<tr>
<td>158</td>
<td>9E</td>
<td>PM</td>
<td></td>
</tr>
<tr>
<td>159</td>
<td>9F</td>
<td>APC</td>
<td></td>
</tr>
<tr>
<td>160</td>
<td>A0</td>
<td>NBSP</td>
<td></td>
</tr>
<tr>
<td>161</td>
<td>A1</td>
<td>¡</td>
<td></td>
</tr>
<tr>
<td>162</td>
<td>A2</td>
<td>¥φ</td>
<td></td>
</tr>
<tr>
<td>163</td>
<td>A3</td>
<td>£</td>
<td></td>
</tr>
<tr>
<td>164</td>
<td>A4</td>
<td>€</td>
<td></td>
</tr>
<tr>
<td>165</td>
<td>A5</td>
<td>¥§</td>
<td></td>
</tr>
<tr>
<td>166</td>
<td>A6</td>
<td>$</td>
<td></td>
</tr>
<tr>
<td>167</td>
<td>A7</td>
<td>§</td>
<td></td>
</tr>
<tr>
<td>168</td>
<td>A8</td>
<td>$</td>
<td></td>
</tr>
<tr>
<td>169</td>
<td>A9</td>
<td>©®</td>
<td></td>
</tr>
<tr>
<td>170</td>
<td>AA</td>
<td>°</td>
<td></td>
</tr>
<tr>
<td>171</td>
<td>AB</td>
<td>«</td>
<td></td>
</tr>
<tr>
<td>172</td>
<td>AC</td>
<td>¬</td>
<td></td>
</tr>
<tr>
<td>173</td>
<td>AD</td>
<td>SHY</td>
<td></td>
</tr>
<tr>
<td>174</td>
<td>AE</td>
<td>®</td>
<td></td>
</tr>
<tr>
<td>175</td>
<td>AF</td>
<td>~</td>
<td></td>
</tr>
<tr>
<td>176</td>
<td>B0</td>
<td>°</td>
<td></td>
</tr>
<tr>
<td>177</td>
<td>B1</td>
<td>±</td>
<td></td>
</tr>
<tr>
<td>178</td>
<td>B2</td>
<td>²</td>
<td></td>
</tr>
<tr>
<td>179</td>
<td>B3</td>
<td>³</td>
<td></td>
</tr>
<tr>
<td>180</td>
<td>B4</td>
<td>Ž</td>
<td></td>
</tr>
<tr>
<td>181</td>
<td>B5</td>
<td>μ</td>
<td></td>
</tr>
<tr>
<td>182</td>
<td>B6</td>
<td>¶</td>
<td></td>
</tr>
<tr>
<td>183</td>
<td>B7</td>
<td>·</td>
<td></td>
</tr>
<tr>
<td>184</td>
<td>B8</td>
<td>ž</td>
<td></td>
</tr>
<tr>
<td>185</td>
<td>B9</td>
<td>'</td>
<td></td>
</tr>
<tr>
<td>186</td>
<td>BA</td>
<td>°</td>
<td></td>
</tr>
<tr>
<td>187</td>
<td>BB</td>
<td>»</td>
<td></td>
</tr>
<tr>
<td>Dec</td>
<td>Hex</td>
<td>Character</td>
<td>PAC Keyboard</td>
</tr>
<tr>
<td>-----</td>
<td>-----</td>
<td>-----------</td>
<td>--------------</td>
</tr>
<tr>
<td>188</td>
<td>BC</td>
<td>Œ</td>
<td></td>
</tr>
<tr>
<td>189</td>
<td>BD</td>
<td>œ</td>
<td></td>
</tr>
<tr>
<td>190</td>
<td>BE</td>
<td>Ŷ</td>
<td></td>
</tr>
<tr>
<td>191</td>
<td>BF</td>
<td>ç</td>
<td></td>
</tr>
<tr>
<td>192</td>
<td>C0</td>
<td>Á</td>
<td></td>
</tr>
<tr>
<td>193</td>
<td>C1</td>
<td>Á</td>
<td></td>
</tr>
<tr>
<td>194</td>
<td>C2</td>
<td>Á</td>
<td></td>
</tr>
<tr>
<td>195</td>
<td>C3</td>
<td>Á</td>
<td></td>
</tr>
<tr>
<td>196</td>
<td>C4</td>
<td>Å</td>
<td>Alt+A</td>
</tr>
<tr>
<td>197</td>
<td>C5</td>
<td>A</td>
<td></td>
</tr>
<tr>
<td>198</td>
<td>C6</td>
<td>Æ</td>
<td></td>
</tr>
<tr>
<td>199</td>
<td>C7</td>
<td>Ç</td>
<td></td>
</tr>
<tr>
<td>200</td>
<td>C8</td>
<td>É</td>
<td></td>
</tr>
<tr>
<td>201</td>
<td>C9</td>
<td>Ê</td>
<td></td>
</tr>
<tr>
<td>202</td>
<td>CA</td>
<td>É</td>
<td></td>
</tr>
<tr>
<td>203</td>
<td>CB</td>
<td>É</td>
<td></td>
</tr>
<tr>
<td>204</td>
<td>CC</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>205</td>
<td>CD</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>206</td>
<td>CE</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>207</td>
<td>CF</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>208</td>
<td>D0</td>
<td>Ø</td>
<td></td>
</tr>
<tr>
<td>209</td>
<td>D1</td>
<td>N</td>
<td></td>
</tr>
<tr>
<td>210</td>
<td>D2</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>211</td>
<td>D3</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>212</td>
<td>D4</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>213</td>
<td>D5</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>214</td>
<td>D6</td>
<td>́</td>
<td>Alt+O</td>
</tr>
<tr>
<td>215</td>
<td>D7</td>
<td>x</td>
<td></td>
</tr>
<tr>
<td>216</td>
<td>D8</td>
<td>Ø</td>
<td></td>
</tr>
<tr>
<td>217</td>
<td>D9</td>
<td>Ü</td>
<td></td>
</tr>
<tr>
<td>218</td>
<td>DA</td>
<td>Ü</td>
<td></td>
</tr>
<tr>
<td>219</td>
<td>DB</td>
<td>Ü</td>
<td></td>
</tr>
<tr>
<td>220</td>
<td>DC</td>
<td>Ü</td>
<td>Alt+U</td>
</tr>
<tr>
<td>221</td>
<td>DD</td>
<td>¦</td>
<td></td>
</tr>
<tr>
<td>222</td>
<td>DE</td>
<td>START</td>
<td></td>
</tr>
<tr>
<td>223</td>
<td>DF</td>
<td>́</td>
<td></td>
</tr>
<tr>
<td>224</td>
<td>E0</td>
<td>à</td>
<td></td>
</tr>
<tr>
<td>225</td>
<td>E1</td>
<td>á</td>
<td></td>
</tr>
<tr>
<td>226</td>
<td>E2</td>
<td>á</td>
<td></td>
</tr>
<tr>
<td>227</td>
<td>E3</td>
<td>á</td>
<td></td>
</tr>
<tr>
<td>228</td>
<td>E4</td>
<td>á</td>
<td>Ctrl+A</td>
</tr>
<tr>
<td>229</td>
<td>E5</td>
<td>á</td>
<td></td>
</tr>
<tr>
<td>230</td>
<td>E6</td>
<td>æ</td>
<td></td>
</tr>
<tr>
<td>231</td>
<td>E7</td>
<td>ç</td>
<td></td>
</tr>
<tr>
<td>232</td>
<td>E8</td>
<td>é</td>
<td></td>
</tr>
<tr>
<td>233</td>
<td>E9</td>
<td>é</td>
<td></td>
</tr>
</tbody>
</table>
## Additions

<table>
<thead>
<tr>
<th>Dec</th>
<th>Hex</th>
<th>PAC Keyboard</th>
</tr>
</thead>
<tbody>
<tr>
<td>294</td>
<td>126</td>
<td>Alt+L</td>
</tr>
<tr>
<td>327</td>
<td>147</td>
<td>Shift+Arrow left</td>
</tr>
<tr>
<td>328</td>
<td>148</td>
<td>Arrow up</td>
</tr>
<tr>
<td>329</td>
<td>149</td>
<td>Shift+Arrow up</td>
</tr>
<tr>
<td>331</td>
<td>14B</td>
<td>Arrow left</td>
</tr>
<tr>
<td>333</td>
<td>14D</td>
<td>Arrow right</td>
</tr>
<tr>
<td>335</td>
<td>14F</td>
<td>Shift+Arrow right</td>
</tr>
<tr>
<td>336</td>
<td>150</td>
<td>Arrow down</td>
</tr>
<tr>
<td>337</td>
<td>151</td>
<td>Shift+Arrow down</td>
</tr>
<tr>
<td>338</td>
<td>152</td>
<td>INS</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Shift+INS</td>
</tr>
<tr>
<td>339</td>
<td>153</td>
<td>DEL</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ctrl+Arrow left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Ctrl+Arrow right</td>
</tr>
<tr>
<td>397</td>
<td>18D</td>
<td>Ctrl+DEL</td>
</tr>
</tbody>
</table>
## INDEX

### - A -

<table>
<thead>
<tr>
<th>Term</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A1</td>
<td>132</td>
</tr>
<tr>
<td>A1 (Axis)</td>
<td>132</td>
</tr>
<tr>
<td>ABS</td>
<td>260</td>
</tr>
<tr>
<td>Absolute dimension system</td>
<td>148</td>
</tr>
<tr>
<td>Absolute value</td>
<td></td>
</tr>
<tr>
<td>of a real number</td>
<td>260</td>
</tr>
<tr>
<td>of an integer</td>
<td>260</td>
</tr>
<tr>
<td>Accessories</td>
<td>411</td>
</tr>
<tr>
<td>Activate</td>
<td></td>
</tr>
<tr>
<td>Standard positioning mode</td>
<td>183</td>
</tr>
<tr>
<td>Activate the start positioning mode</td>
<td>178</td>
</tr>
<tr>
<td>Activating a menu item</td>
<td>25</td>
</tr>
<tr>
<td>Addition</td>
<td>250</td>
</tr>
<tr>
<td>ADi</td>
<td>278</td>
</tr>
<tr>
<td>Ambient temperature</td>
<td>18</td>
</tr>
<tr>
<td>Analog-to-digital converter</td>
<td>278</td>
</tr>
<tr>
<td>Acquisition of A-D values in a defined time base</td>
<td>281</td>
</tr>
<tr>
<td>Acquisition of several A-D values</td>
<td>279</td>
</tr>
<tr>
<td>Synchronous acquisition of A-D values</td>
<td>279</td>
</tr>
<tr>
<td>AND-LD</td>
<td>273, 274</td>
</tr>
<tr>
<td>Approach to reference point</td>
<td>143</td>
</tr>
<tr>
<td>Arithmetic operations</td>
<td>246</td>
</tr>
<tr>
<td>Addition</td>
<td>250</td>
</tr>
<tr>
<td>Division</td>
<td>253</td>
</tr>
<tr>
<td>Multiplication</td>
<td>252</td>
</tr>
<tr>
<td>Subtraction</td>
<td>251</td>
</tr>
<tr>
<td>Automatic mode</td>
<td>29</td>
</tr>
<tr>
<td>Automatic mode with STOP</td>
<td>75</td>
</tr>
<tr>
<td>Sequence of malfunction</td>
<td>76</td>
</tr>
<tr>
<td>Automatic run</td>
<td></td>
</tr>
<tr>
<td>start</td>
<td>123</td>
</tr>
<tr>
<td>stop</td>
<td>122</td>
</tr>
<tr>
<td>Axis</td>
<td></td>
</tr>
<tr>
<td>positioning</td>
<td>132</td>
</tr>
<tr>
<td>Axis and drive parameters</td>
<td>335</td>
</tr>
<tr>
<td>Axis parameters</td>
<td></td>
</tr>
<tr>
<td>Axis error number</td>
<td>85, 88</td>
</tr>
<tr>
<td>Axis parameters</td>
<td></td>
</tr>
<tr>
<td>Approach to reference point</td>
<td>337</td>
</tr>
<tr>
<td>Limit switches</td>
<td>337</td>
</tr>
<tr>
<td>Section</td>
<td>Page</td>
</tr>
<tr>
<td>--------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Motor parameters</td>
<td>338</td>
</tr>
<tr>
<td>Traversing parameters</td>
<td>336</td>
</tr>
<tr>
<td>Basic board</td>
<td>409</td>
</tr>
<tr>
<td>Basic settings</td>
<td></td>
</tr>
<tr>
<td>Axis parameters</td>
<td>53</td>
</tr>
<tr>
<td>Delete program memory</td>
<td>53</td>
</tr>
<tr>
<td>Load Axis parameter default values</td>
<td>53</td>
</tr>
<tr>
<td>Load system parameters default values</td>
<td>53</td>
</tr>
<tr>
<td>Overview of the menu</td>
<td>53</td>
</tr>
<tr>
<td>Reinitialize PA-CONTROL</td>
<td>54</td>
</tr>
<tr>
<td>System parameters</td>
<td>53</td>
</tr>
<tr>
<td><strong>- B -</strong></td>
<td></td>
</tr>
<tr>
<td>BCD notation to outputs</td>
<td>241</td>
</tr>
<tr>
<td>Binary notation to flags</td>
<td>242</td>
</tr>
<tr>
<td>Binary notation to outputs</td>
<td>240</td>
</tr>
<tr>
<td>Brake</td>
<td>304</td>
</tr>
<tr>
<td>Branch in the program</td>
<td></td>
</tr>
<tr>
<td>if time monitoring</td>
<td>191</td>
</tr>
<tr>
<td>BREAK</td>
<td>117</td>
</tr>
<tr>
<td><strong>- C -</strong></td>
<td></td>
</tr>
<tr>
<td>Cancel</td>
<td>73</td>
</tr>
<tr>
<td>Cancel automatic mode</td>
<td>117</td>
</tr>
<tr>
<td>CANopen bus</td>
<td>284</td>
</tr>
<tr>
<td>Checking device errors</td>
<td>289</td>
</tr>
<tr>
<td>Device error message is present</td>
<td>289</td>
</tr>
<tr>
<td>Enter Pre-Operational State</td>
<td>287</td>
</tr>
<tr>
<td>Functional monitoring of the devices</td>
<td>288</td>
</tr>
<tr>
<td>Network management commands</td>
<td>285</td>
</tr>
<tr>
<td>Processing of process data objects</td>
<td>292</td>
</tr>
<tr>
<td>Processing of service data objects</td>
<td>290</td>
</tr>
<tr>
<td>Read process data</td>
<td>294</td>
</tr>
<tr>
<td>Read service data object</td>
<td>290</td>
</tr>
<tr>
<td>Reset Communication</td>
<td>287</td>
</tr>
<tr>
<td>Send process data</td>
<td>295</td>
</tr>
<tr>
<td>Start-Remote Node</td>
<td>285</td>
</tr>
<tr>
<td>Stop-Remote Node</td>
<td>286</td>
</tr>
<tr>
<td>Typical applications</td>
<td>296</td>
</tr>
<tr>
<td>Write service data object</td>
<td>291</td>
</tr>
<tr>
<td>CANopen Bus</td>
<td>306</td>
</tr>
<tr>
<td>CANopen interface</td>
<td></td>
</tr>
<tr>
<td>Devices with a permanent assignment</td>
<td>347</td>
</tr>
<tr>
<td>Devices without a permanent assignment</td>
<td>347</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Overview of the CAN IDs</td>
<td>348</td>
</tr>
<tr>
<td>CANopen-Bus</td>
<td></td>
</tr>
<tr>
<td>Reset Remote Node</td>
<td>286</td>
</tr>
<tr>
<td>CASE.STORE</td>
<td>175</td>
</tr>
<tr>
<td>Changing the motor current</td>
<td>151</td>
</tr>
<tr>
<td>Character transfer</td>
<td></td>
</tr>
<tr>
<td>in the background</td>
<td>217, 219</td>
</tr>
<tr>
<td>to the local character buffer</td>
<td>216</td>
</tr>
<tr>
<td>Check</td>
<td></td>
</tr>
<tr>
<td>Character transfer in the background completed</td>
<td>221</td>
</tr>
<tr>
<td>Check if a key has been actuated</td>
<td>235</td>
</tr>
<tr>
<td>CHN</td>
<td>221</td>
</tr>
<tr>
<td>Clear the display</td>
<td>202</td>
</tr>
<tr>
<td>Clear up to end of line</td>
<td>203</td>
</tr>
<tr>
<td>COM1</td>
<td>305</td>
</tr>
<tr>
<td>COM1,2 parameters</td>
<td>332</td>
</tr>
<tr>
<td>Command code</td>
<td>89</td>
</tr>
<tr>
<td>Communication via modem</td>
<td>59</td>
</tr>
<tr>
<td>Activate connection</td>
<td>60</td>
</tr>
<tr>
<td>Display status of the connection</td>
<td>61</td>
</tr>
<tr>
<td>Edit Modem Settings</td>
<td>62</td>
</tr>
<tr>
<td>Overview</td>
<td>59</td>
</tr>
<tr>
<td>Selection of modem type</td>
<td>63</td>
</tr>
<tr>
<td>Sequence of communication</td>
<td>64</td>
</tr>
<tr>
<td>Terminate connection</td>
<td>61</td>
</tr>
<tr>
<td>Comparison operations</td>
<td>262</td>
</tr>
<tr>
<td>Comparisons</td>
<td>263</td>
</tr>
<tr>
<td>Conditional jump</td>
<td>106</td>
</tr>
<tr>
<td>Conditional Subroutine call</td>
<td>108</td>
</tr>
<tr>
<td>Connecting voltages</td>
<td>18</td>
</tr>
<tr>
<td>Connection Instructions</td>
<td>15</td>
</tr>
<tr>
<td>Connector assignment</td>
<td></td>
</tr>
<tr>
<td>Inputs and outputs</td>
<td>303</td>
</tr>
<tr>
<td>Inputs and outputs of PA-CONTROL Single</td>
<td>303</td>
</tr>
<tr>
<td>Rotation monitoring</td>
<td>304</td>
</tr>
<tr>
<td>Convert characters</td>
<td>227</td>
</tr>
<tr>
<td>Coordination byte</td>
<td>378</td>
</tr>
<tr>
<td>read</td>
<td>379</td>
</tr>
<tr>
<td>write</td>
<td>378</td>
</tr>
<tr>
<td>COPY</td>
<td>227</td>
</tr>
<tr>
<td>Current task number</td>
<td>85, 88</td>
</tr>
</tbody>
</table>
## - D -

<table>
<thead>
<tr>
<th>DAi</th>
<th>283</th>
</tr>
</thead>
<tbody>
<tr>
<td>Data transfer</td>
<td></td>
</tr>
<tr>
<td>to and from the local character buffer</td>
<td>222</td>
</tr>
<tr>
<td>Declaration</td>
<td>14</td>
</tr>
<tr>
<td>Diagnosis</td>
<td>37</td>
</tr>
<tr>
<td>CANopen</td>
<td>41</td>
</tr>
<tr>
<td>D-A Converter</td>
<td>42</td>
</tr>
<tr>
<td>Inputs, Outputs, Flags</td>
<td>38</td>
</tr>
<tr>
<td>Integer register</td>
<td>40</td>
</tr>
<tr>
<td>Limit switch</td>
<td>37</td>
</tr>
<tr>
<td>Real number register</td>
<td>40</td>
</tr>
<tr>
<td>Stand-by</td>
<td>37</td>
</tr>
<tr>
<td>Diagnosis port</td>
<td>303</td>
</tr>
<tr>
<td>Digital-to-analog converter</td>
<td>282</td>
</tr>
<tr>
<td>Brief description</td>
<td>282</td>
</tr>
<tr>
<td>Direct addressing</td>
<td>246</td>
</tr>
<tr>
<td>Display menu item</td>
<td>25</td>
</tr>
<tr>
<td>Division</td>
<td>253</td>
</tr>
</tbody>
</table>

## - E -

| Entry of a value via the keyboard | 237 |
| Expansions of PA-CONTROL CPU4 |  |
| IEF module SSI interface | 353 |
| Expansions of the PA-CONTROL CPU4 |  |
| IEF module A-D converter | 352 |
| IEF module Profibus DP | 351 |
| IEF module RS 232 port | 350 |
| RS 232 port driver | 349 |
| Explanations | 89 |
| External start input no. | 323 |
| External Stop input no. | 323 |

## - F -

<p>| F | 141 |
| Flags |  |
| to register | 245 |
| Flow diagram |  |
| PA-CONTROL in the initial position | 77 |
| Start automatic mode | 74 |
| Following error |  |
| LV-servoTEC | 137 |
| PA-CONTROL MP | 137 |
| Function and status check | 308 |</p>
<table>
<thead>
<tr>
<th>Code</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>G01</td>
<td>163</td>
</tr>
<tr>
<td>G100</td>
<td>150</td>
</tr>
<tr>
<td>G101</td>
<td>151</td>
</tr>
<tr>
<td>G123</td>
<td>152</td>
</tr>
<tr>
<td>G142</td>
<td>159</td>
</tr>
<tr>
<td>G143</td>
<td>160</td>
</tr>
<tr>
<td>G150</td>
<td>161</td>
</tr>
<tr>
<td>G170</td>
<td>169</td>
</tr>
<tr>
<td>G171</td>
<td>170</td>
</tr>
<tr>
<td>G172</td>
<td>171</td>
</tr>
<tr>
<td>G173</td>
<td>172</td>
</tr>
<tr>
<td>G180</td>
<td>279</td>
</tr>
<tr>
<td>G181</td>
<td>281</td>
</tr>
<tr>
<td>G210</td>
<td>178</td>
</tr>
<tr>
<td>G211</td>
<td>179</td>
</tr>
<tr>
<td>G212</td>
<td>181</td>
</tr>
<tr>
<td>G213</td>
<td>183</td>
</tr>
<tr>
<td>G221</td>
<td>184</td>
</tr>
<tr>
<td>G222</td>
<td>186</td>
</tr>
<tr>
<td>G230</td>
<td>188</td>
</tr>
<tr>
<td>G25.An</td>
<td>143</td>
</tr>
<tr>
<td>G26</td>
<td>145</td>
</tr>
<tr>
<td>G29</td>
<td>146</td>
</tr>
<tr>
<td>G401.1</td>
<td>194</td>
</tr>
<tr>
<td>G421.1</td>
<td>191</td>
</tr>
<tr>
<td>G422.1</td>
<td>192</td>
</tr>
<tr>
<td>G423.1</td>
<td>193</td>
</tr>
<tr>
<td>G500</td>
<td>200</td>
</tr>
<tr>
<td>G501</td>
<td>202</td>
</tr>
<tr>
<td>G502</td>
<td>203</td>
</tr>
<tr>
<td>G503/G504</td>
<td>204</td>
</tr>
<tr>
<td>G510</td>
<td>206</td>
</tr>
<tr>
<td>G511</td>
<td>207</td>
</tr>
<tr>
<td>G512</td>
<td>208</td>
</tr>
<tr>
<td>G515</td>
<td>209</td>
</tr>
<tr>
<td>G520</td>
<td>210</td>
</tr>
<tr>
<td>G521</td>
<td>212</td>
</tr>
<tr>
<td>G531</td>
<td>214</td>
</tr>
<tr>
<td>G532</td>
<td>216</td>
</tr>
<tr>
<td>G533</td>
<td>217</td>
</tr>
<tr>
<td>G534</td>
<td>219</td>
</tr>
</tbody>
</table>
G540 235
G541 236
G542 237
G600 240
G601 241
G602 242
G603 244
G604 245
G701 285
G702 286
G703 286
G704 287
G705 287
G711 288
G721 289
G722 289
G730 289
G731 291
G74x 294
G75x 295
G90 148
G91 149
GET 231
Get a character from the keyboard 236
Get A-D values 278
Get axis position 133
Get program status 131
GETI 232

- H -
Hold-up time 97

- I -
IEF control console 364
  Cable connections 365
  General 364
  Panel mounting 367
IEF Template
  IEF Template 370
Image from register 239
Image to register 239
In.m 91
Incremental dimension system 149
Indirect addressing 246
Initialisation of the ports
Input field
  Assignment of the keys in the input field
Inputs
Inputs and outputs of PA-CONTROL Single
Inputs in binary format to register
INT
Integer registers
  Value range
Interpolation
  Abort
  Activate outputs before starting
  Activation of outputs during interpolation
IPOEND

- J -
JMP
JMP-LINE.Ni
JMP-LINE-IPO.Ni
Jump distributor

- K -
Key
Keyboard
  Special keys

- L -
Limit switch monitoring
  switch off
  switch on
LINE command
Linear interpolation
List of system flags
List of system N registers
List of variables
Load register
Load register with axis parameters
Logic operations
  Command AND
  Command AND-LD
  Command LD
  Command OR
  command OR-LD
  Command OUT
  Complex logic operations
Logic AND operations 269
logic OR operations 270
Multi-level logic AND operations 274
Multi-level OR operations 275
Supplemental instructions 271
Loop with conditional jump 114, 116
LVservoTEC
Get following error 137

- M -
Main menu 24
MALFUNCTION 73
Manual 29
  Approach to reference point 29
  Single step 30
  via front plate 30
Manual release input no 324
Manual release input no. 324
Measuring mode 156
  activate 157
  deactivate 158
Menu item 24
Mn.M 93
Mn:=m 96
Modem
  Übersicht 59
Motor steps per revolution 338, 339, 340, 341, 342, 343, 344
Multiplication 252

- N -
Nesting 102
Ni.Value 94
Ni:= PEAn 137
Ni:=ENCn 138
Nn:=An 133

- O -
Object list 374
On.m 92
On:=m 95
Operating hours counter 85, 88
Operating tenths of a second counter 85, 88
Operator interface 21
Output of control characters 208
Output of D-A values 283
<table>
<thead>
<tr>
<th>Output of value</th>
<th>210, 212</th>
</tr>
</thead>
<tbody>
<tr>
<td>Outputs</td>
<td>18</td>
</tr>
<tr>
<td>Outside measurements</td>
<td>20</td>
</tr>
<tr>
<td>Overview of the command set</td>
<td>65</td>
</tr>
<tr>
<td>Overview of the CPU5</td>
<td>345</td>
</tr>
</tbody>
</table>

- **P** -

| PAB program                     | 33       |
| PA-CONTROL                      |         |
| CANopen interface               | 347      |
| General operation               | 23       |
| Inputs und Outputs              | 18       |
| Menu structure                  | 27       |
| Switch-on                       | 23       |
| PA-CONTROL front plate          |         |
| Simulation                      | 383      |
| PA-CONTROL key code             | 414      |
| PA-CONTROL servoTEC             |         |
| Cables                          | 302      |
| optional accessories            | 302      |
| Options                         | 13, 345  |
| Performance features            | 13       |
| Technical Short Information     | 13       |
| Wiring                          | 301      |
| Parallel run                    |         |
| starting with CASE              | 125      |
| Parameter                       |         |
| Start/Stop rotational speed     | 336      |
| Parameters                      | 46       |
| ASi parameters                  | 47       |
| Check axis parameters           | 313      |
| Check motor parameters          | 313      |
| Check traversing parameters     | 313      |
| External Start input no.        | 323      |
| External Stop input no.         | 323      |
| Manual release input no.        | 324      |
| Manual release input no.        | 324      |
| Motor full steps per revolution | 338, 339, 340, 341, 342 |
| Motor steps per revolution      | 338, 339, 340, 341, 342, 343, 344 |
| Overview of the menu            | 46       |
| Program after Stop              | 325      |
| Program at malfunction          | 325      |
| Program Start after Stop        | 325      |
Run definition 324
servoTEC-Parameters 50
Set axis parameters 46
Set system parameters 46
Standby output no. 324
Start program 325
Parameters of PA-CONTROL 319
CANopen-Bus 331
Diagnosis 329
General 319
Parameter description 320
Profibus 334
System parameters 321
PLS7
  Run error message 408
Plug-in connections 345
  Designations 20
  Position 20
PNC program 33
PNX-Programm 33
Polling range 377
Port 18
POS 226
Position the cursor 204
Position-conditional
  jump 179
  subroutine call 181, 186
Position-conditional jump 184
Power supply unit 410
Profibus Option 307
Program
  Create 33
  jump if time monitoring elapsed 191
  Modification 35
  Rename 35
Program after stop 43
Program after Stop 326
Program at malfunction 43, 44
Program at Malfunction 327
Program directory 33
Program in initial position 327
Program list 27
Program name 26
Program start after stop 43
Program Start after Stop 44, 327
Program status
  Standard request 131
Program structure 78
Program test 37
Programm
  Copy 35
  Delete 36
Programming 32
  Altering an existing program 35
  Copy program 35
  Delete program 36
  Memory occupancy 36
  Menu 32
  Program directory 33
  Rename program 35
Programming elements 79
  Axes 79
  Comment 79
  Direct addressing 79
  Flags 79
  Indirect addressing 79
  Inputs 79
  Integer registers 79
  Label 79
  Outputs 79
  Real number registers 79
  Strings 79
  System flags 79
  System integer registers 79
  System real number registers 79
  Times 79
Programming tips 71
  Cancel 73
  MALFUNCTION 73
  START after STOP 73
Programs in parallel run 118
  Ending 121
  Ending with Case 129
  Start 118
Stopping 120
Stopping with Case 127
Protection 18
PTX program 33
PUT 233
PUTI 234

- R -

Read character from PTX file 169
Real number register
  Decimal component 259
  integer component 258
Real number registers
  Value range 79
Real-time clock 195
  Read out 195
  Set date 197
  Set time 198
Request
  current path speed 134
  Current speed of an axis during interpolation 135
  Current traversing speed 134
Ri:= PEAn 137
Rn/Nn:= 247
Rn:=An 133
Rn:=ENCn 138
Rn:=FB 134
Rn:=Ran 248
Root function 257
Rotation monitoring
  Connector assignment 304
RS232 305
RS232 parameters 332
Run definition
  Deletion of assignments 45
  Program at malfunction 44
  Program at Malfunction 327
  Program at stop 43
  Program at Stop 326
  Program Start after Stop 44, 327
  Start program 43, 326
Run definitions 43
Run error number 85, 88
<table>
<thead>
<tr>
<th>Run interpreter</th>
<th>71</th>
</tr>
</thead>
<tbody>
<tr>
<td>S0 - Copying character strings</td>
<td>223</td>
</tr>
<tr>
<td>S0:= CHN</td>
<td>223</td>
</tr>
<tr>
<td>Safety Instructions</td>
<td>15</td>
</tr>
<tr>
<td>Search for character position</td>
<td>226</td>
</tr>
<tr>
<td>Selection of the data channel</td>
<td>200</td>
</tr>
<tr>
<td>Sequence number</td>
<td>85, 88</td>
</tr>
<tr>
<td>Serial signalling channel</td>
<td>380</td>
</tr>
<tr>
<td>ServoTEC Parameters</td>
<td></td>
</tr>
<tr>
<td>Read</td>
<td>155</td>
</tr>
<tr>
<td>ServoTEC-Parameters</td>
<td></td>
</tr>
<tr>
<td>Write</td>
<td>154</td>
</tr>
<tr>
<td>SET / RES</td>
<td>271</td>
</tr>
<tr>
<td>Set / reset</td>
<td></td>
</tr>
<tr>
<td>Flag</td>
<td>96</td>
</tr>
<tr>
<td>Set position to dimension</td>
<td>146</td>
</tr>
<tr>
<td>Set position to zero</td>
<td>145</td>
</tr>
<tr>
<td>Set/Reset</td>
<td></td>
</tr>
<tr>
<td>Output</td>
<td>95</td>
</tr>
<tr>
<td>Signalling system</td>
<td></td>
</tr>
<tr>
<td>parallel</td>
<td>381</td>
</tr>
<tr>
<td>serial</td>
<td>377</td>
</tr>
<tr>
<td>Sn - Copying character strings</td>
<td>222</td>
</tr>
<tr>
<td>Sn:= Si</td>
<td>222</td>
</tr>
<tr>
<td>Spare parts</td>
<td>411</td>
</tr>
<tr>
<td>Specifying the acceleration</td>
<td>150</td>
</tr>
<tr>
<td>SQRT</td>
<td>257</td>
</tr>
<tr>
<td>SSI interface</td>
<td></td>
</tr>
<tr>
<td>Assignment</td>
<td>354</td>
</tr>
<tr>
<td>Assignment of the plug-in connectors</td>
<td>353</td>
</tr>
<tr>
<td>assignment table</td>
<td>354</td>
</tr>
<tr>
<td>Detection</td>
<td>354</td>
</tr>
<tr>
<td>Device parameters</td>
<td>355</td>
</tr>
<tr>
<td>General</td>
<td>353</td>
</tr>
<tr>
<td>Parameters</td>
<td>355</td>
</tr>
<tr>
<td>Standard Subroutine call</td>
<td>101</td>
</tr>
<tr>
<td>Standby output no.</td>
<td>324</td>
</tr>
<tr>
<td>START after STOP</td>
<td>73</td>
</tr>
<tr>
<td>Start key test</td>
<td>58</td>
</tr>
<tr>
<td>Start of an interpolation at a specified program line</td>
<td>99, 100</td>
</tr>
<tr>
<td>Start program</td>
<td>43, 45, 326</td>
</tr>
<tr>
<td>Topic</td>
<td>Page</td>
</tr>
<tr>
<td>----------------------------------------------------------------------</td>
<td>------</td>
</tr>
<tr>
<td>Start/Stop rotational speed</td>
<td>336</td>
</tr>
<tr>
<td>Start-up program</td>
<td>27</td>
</tr>
<tr>
<td>Storage</td>
<td>18</td>
</tr>
<tr>
<td>Storage of values</td>
<td>173, 175</td>
</tr>
<tr>
<td>STORE</td>
<td>173</td>
</tr>
<tr>
<td>Store string (Sn) in a line of a PTX file</td>
<td>172</td>
</tr>
<tr>
<td>Subroutine call</td>
<td></td>
</tr>
<tr>
<td>extended form</td>
<td>103</td>
</tr>
<tr>
<td>Program name from register</td>
<td>104</td>
</tr>
<tr>
<td>Standard</td>
<td>101</td>
</tr>
<tr>
<td>Subroutine distributor</td>
<td>112</td>
</tr>
<tr>
<td>Subtraction</td>
<td>251</td>
</tr>
<tr>
<td>SÜTRON control console</td>
<td>302, 369</td>
</tr>
<tr>
<td>Switch display on / off</td>
<td>105</td>
</tr>
<tr>
<td>System Diagnosis</td>
<td>56</td>
</tr>
<tr>
<td>Clock</td>
<td>57</td>
</tr>
<tr>
<td>Display hardware configuration</td>
<td>56</td>
</tr>
<tr>
<td>Keyboard test</td>
<td>57</td>
</tr>
<tr>
<td>Test of key switch</td>
<td>58</td>
</tr>
<tr>
<td>Test of the COM Port</td>
<td>58</td>
</tr>
<tr>
<td>Test of the start key</td>
<td>58</td>
</tr>
<tr>
<td>Test of the stop key</td>
<td>58</td>
</tr>
<tr>
<td>System error</td>
<td>56</td>
</tr>
<tr>
<td>System error number</td>
<td>85, 88</td>
</tr>
<tr>
<td>System flags</td>
<td>81, 82, 83, 84</td>
</tr>
<tr>
<td>System functions</td>
<td>72</td>
</tr>
<tr>
<td>Cancel</td>
<td>73</td>
</tr>
<tr>
<td>MALFUNCTION</td>
<td>73</td>
</tr>
<tr>
<td>Program at MALFUNCTION</td>
<td>72</td>
</tr>
<tr>
<td>Program at START</td>
<td>72</td>
</tr>
<tr>
<td>Program at START after STOP</td>
<td>72</td>
</tr>
<tr>
<td>Program at STOP</td>
<td>72</td>
</tr>
<tr>
<td>START after STOP</td>
<td>73</td>
</tr>
<tr>
<td>System integer registers</td>
<td>85, 88</td>
</tr>
</tbody>
</table>

- **T** -

<table>
<thead>
<tr>
<th>Topic</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>T</td>
<td>97</td>
</tr>
<tr>
<td>Technical Appendix</td>
<td>387</td>
</tr>
<tr>
<td>Technical Data</td>
<td>18</td>
</tr>
<tr>
<td>Templates</td>
<td>370</td>
</tr>
<tr>
<td>Text and value output via the current data channel</td>
<td>199</td>
</tr>
<tr>
<td>Text output</td>
<td>206, 207, 209</td>
</tr>
<tr>
<td>Time monitoring</td>
<td></td>
</tr>
<tr>
<td>Conditional subroutine call</td>
<td>192</td>
</tr>
<tr>
<td>-------------------------------------</td>
<td>-----</td>
</tr>
<tr>
<td>jump if elapsed</td>
<td>191</td>
</tr>
<tr>
<td>start with jump</td>
<td>191</td>
</tr>
<tr>
<td>Time monitoring commands</td>
<td>190</td>
</tr>
<tr>
<td>Transfer encoder position</td>
<td>138</td>
</tr>
<tr>
<td>Transfer parameters</td>
<td>201</td>
</tr>
<tr>
<td>Transfer the content of an N register</td>
<td>234</td>
</tr>
<tr>
<td>in the INTEL format to S0</td>
<td></td>
</tr>
<tr>
<td>to the local character buffer</td>
<td>233</td>
</tr>
<tr>
<td>Transfer the content of the local character buffer</td>
<td>231</td>
</tr>
<tr>
<td>to an integer register</td>
<td></td>
</tr>
<tr>
<td>Transfer the content of the local character buffer S0</td>
<td>232</td>
</tr>
<tr>
<td>to an integer register in Intel format</td>
<td></td>
</tr>
<tr>
<td>Traverse as long as condition fulfilled</td>
<td>152</td>
</tr>
<tr>
<td>Traverse segment with Start-Stop</td>
<td>161</td>
</tr>
<tr>
<td>Traversing speed</td>
<td>141</td>
</tr>
<tr>
<td>Trigonometrical functions</td>
<td>254</td>
</tr>
<tr>
<td>Cosine</td>
<td>255</td>
</tr>
<tr>
<td>Sine</td>
<td>254</td>
</tr>
<tr>
<td>Tangent</td>
<td>256</td>
</tr>
<tr>
<td>TSwin Configuring software</td>
<td>370, 371</td>
</tr>
</tbody>
</table>

**- U -**

| Unconditional jump                   | 97  |
| User title                           | 45  |

**- V -**

| Value transfer                       | 214 |
| Value transfer from the current data channel | 214 |
| Voltage supply                       | 410 |

**- W -**

| Wait for position                    | 140 |
| from the SSI interface               |     |
| Wait until current position </> than value | 188 |
| Waiting for logical status           | 91  |
| Flag                                 | 93  |
| Input                                | 91  |
| N register                           | 94  |
| Output                               | 92  |
| WINPAC                               |     |
| Functional check                     | 311 |
| Write character to PTX file          | 170 |
| Write line from PTX file to string (Sn) | 171 |