BECKHOFF New Automation Technology

Functional description | EN

TF5200 | TwinCAT 3 CNC

Error management

Name: TwinCAT Project

Location:

Solution: Create new solution

Solution name: TwinCAT Project

7/20/2021 | Version: 1.01

Notes on the documentation

This description is only intended for the use of trained specialists in control and automation engineering who are familiar with the applicable national standards.
It is essential that the documentation and the following notes and explanations are followed when installing and commissioning the components.
It is the duty of the technical personnel to use the documentation published at the respective time of each installation and commissioning.

The responsible staff must ensure that the application or use of the products described satisfy all the requirements for safety, including all the relevant laws, regulations, guidelines and standards.

Disclaimer

The documentation has been prepared with care. The products described are, however, constantly under development.
We reserve the right to revise and change the documentation at any time and without prior announcement.
No claims for the modification of products that have already been supplied may be made on the basis of the data, diagrams and descriptions in this documentation.

Trademarks

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® and XPlanar® are registered trademarks of and licensed by Beckhoff Automation GmbH.
Other designations used in this publication may be trademarks whose use by third parties for their own purposes could violate the rights of the owners.

Patent Pending

The EtherCAT technology is patent protected, in particular by the following applications and patents:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
with corresponding applications or registrations in various other countries.

EtherCAT® is registered trademark and patented technology, licensed by Beckhoff Automation GmbH, Germany

Copyright

General and safety instructions

Icons used and their meanings

This documentation uses the following icons next to the safety instruction and the associated text. Please read the (safety) instructions carefully and comply with them at all times.

Icons in explanatory text

1. Indicates an action.

⇨ Indicates an action statement.

DANGER

Acute danger to life!

If you fail to comply with the safety instruction next to this icon, there is immediate danger to human life and health.

CAUTION

Personal injury and damage to machines!

If you fail to comply with the safety instruction next to this icon, it may result in personal injury or damage to machines.

NOTE

Restriction or error

This icon describes restrictions or warns of errors.

Tips and other notes

This icon indicates information to assist in general understanding or to provide additional information.

General example

Example that clarifies the text.

NC programming example

Programming example (complete NC program or program sequence) of the described function or NC command.

Specific version information

Optional or restricted function. The availability of this function depends on the configuration and the scope of the version.

Table of contents

List of figures

1 Overview

Task

The error management functionality consists of the following topics:

- Output location and scope of the error output
- Error message filters, both global and channel- or axis-specific
- Adding additional information to error messages
- Outputting user error messages

This functionality has been available since CNC Build V3.00.xx.

Parameter definition

Error output parameters are defined in the start-up parameters. Error messages are filtered and supplemented depending on the level on which the filter is intended to act.

Programming

The Parameters [▶ 30] section contains a detailed description of each parameter.

The user can force user error messages via the #ERROR [▶ 20] command in the NC program.

Links to other documents

For the sake of clarity, links to other documents and parameters are abbreviated, e.g. [PROG] for the Programming Manual or P-AXIS-00001 for an axis parameter.

For technical reasons, these links only function in the Online Help (HTML5, CHM) but not in pdf files since pdfs do not support cross-linking.

2 Description

If a channel, single axis or platform error occurs, a corresponding message is output. Errors can be pre-processed by a CNC diagnostics instance and supplemented with additional information (access to database or file system). The user can define parameters for the response (filter) to individual error messages or set them up via the PLC. The extended error message is forwarded to the higher-level diagnostics system via a diagnostics instance.

The occurrence of an error is immediately reported to other devices in the controller (e.g. PLC) in a compact message containing the error ID, error class and time. These devices can then directly derive an internal error reaction.

By default, the CNC takes care of reading and pre-processing CNC error messages.

Forwarding errors through the communication system to a higher-level visualisation system is specifically incorporated by the user.

Managing error messages

The error management functionality allows the user to control the output of error messages. The CNC then acts as a collection point for all information provided.

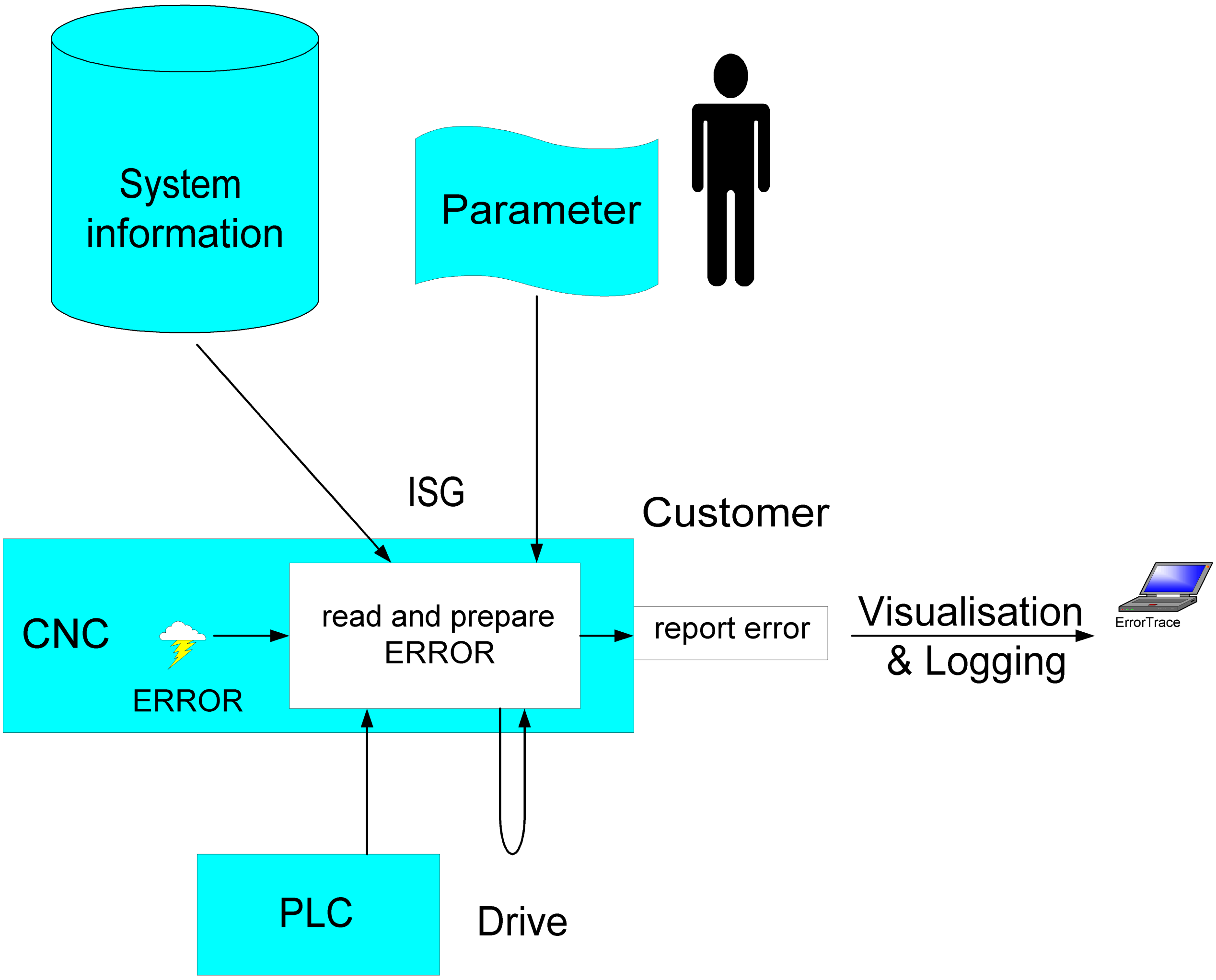


Figure 1: Overview of reading and evaluating an error message within the CNC

Process

- Record and collect all errors of the CNC
- Filter: Suppress and filter errors
- Add. information: Add external/internal error information
- Format: Pre-process and format an error
- Report & log: Log errors to file (LOG) -> log file
- Report & log: Report pre-processed errors (REPORT) -> on-screen display

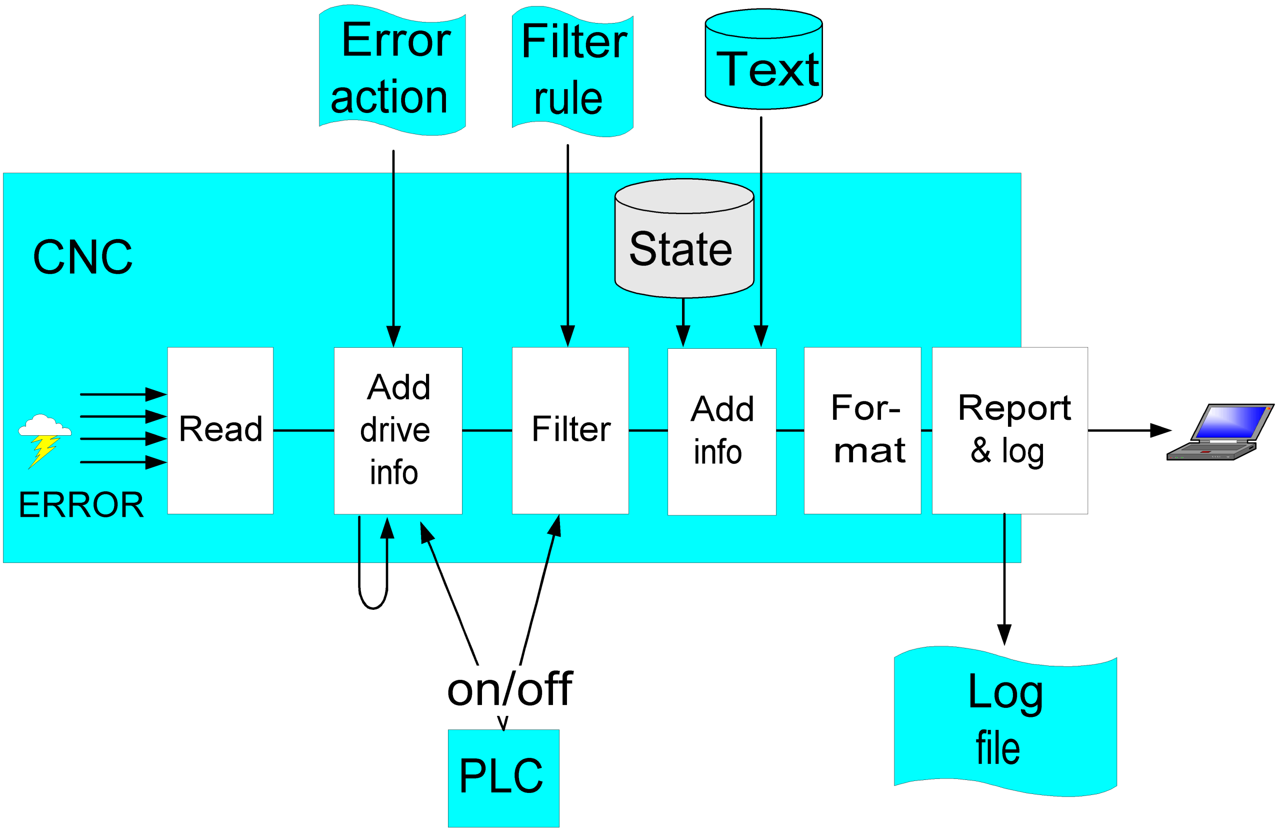


Figure 2: Process and pre-process error messages in the CNC

2.1 Reporting and recording errors

The output of error messages is defined globally by the P-STUP-00167 [▶ 32] parameter.

All the required options must be set in P-STUP-00167.

If you set NO_WARNINGS exclusively, this will result in no output.

The output control mode can also read and write via CNC objects.

- Read via cnc_error_manager_mode_r [▶ 45]
- Write via cnc_error_manager_mode_w [▶ 46]

Output options

Basically, the following 3 parallel output options are available:

1. Print: Output to screen/shell
2. Log: Output to log file
3. Report: Application-specific output

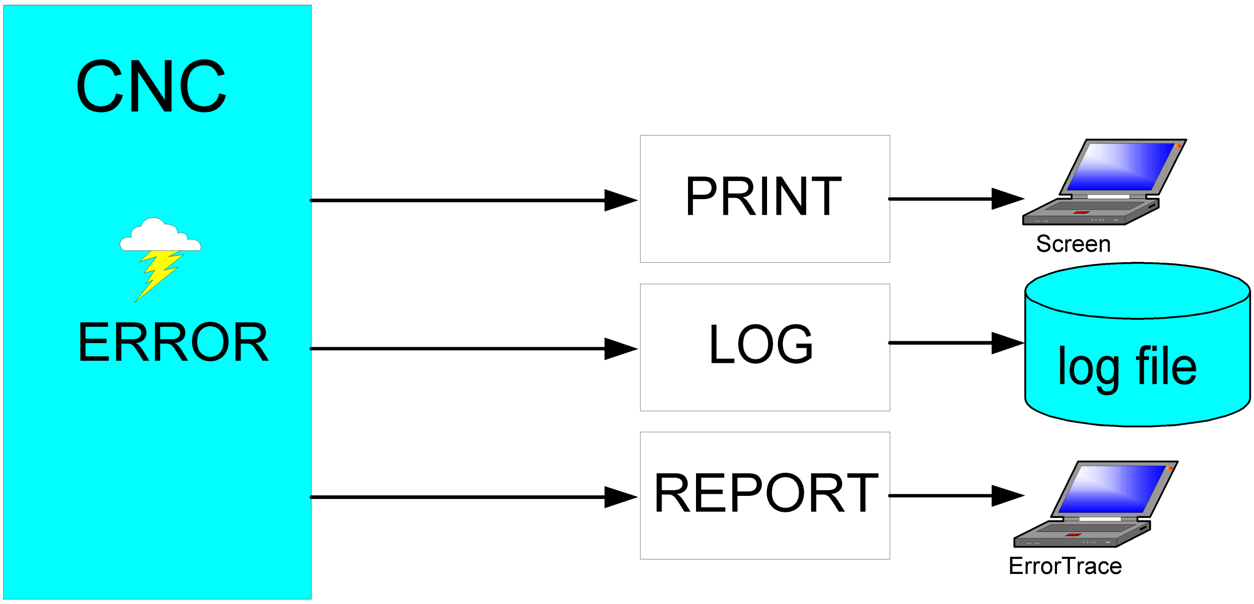


Figure 3: Overview of the 3 output options

Print

Depending on the application, errors can be output directly to screen or to the shell.

Logging to file

Errors are logged to a log file. They can be defined by name via the P-STUP-00170 [▶ 33] parameter and their maximum size can be defined in P-STUP-00171 [▶ 33]. Once the log file overwrites the maximum size defined, the original file is copied to a backup file and the original file is deleted.

Application-specific report

An application-specific output is additionally possible.

"Direct" output to PLC

In addition to the output of a pre-processed error message via PRINT/LOG/REPORT, the error message is sent in "raw format" directly to the PLC after it occurs. As required, this output can be fully disabled via the logging mode P-STUP-00167 [▸ 32] in the Error Manager.

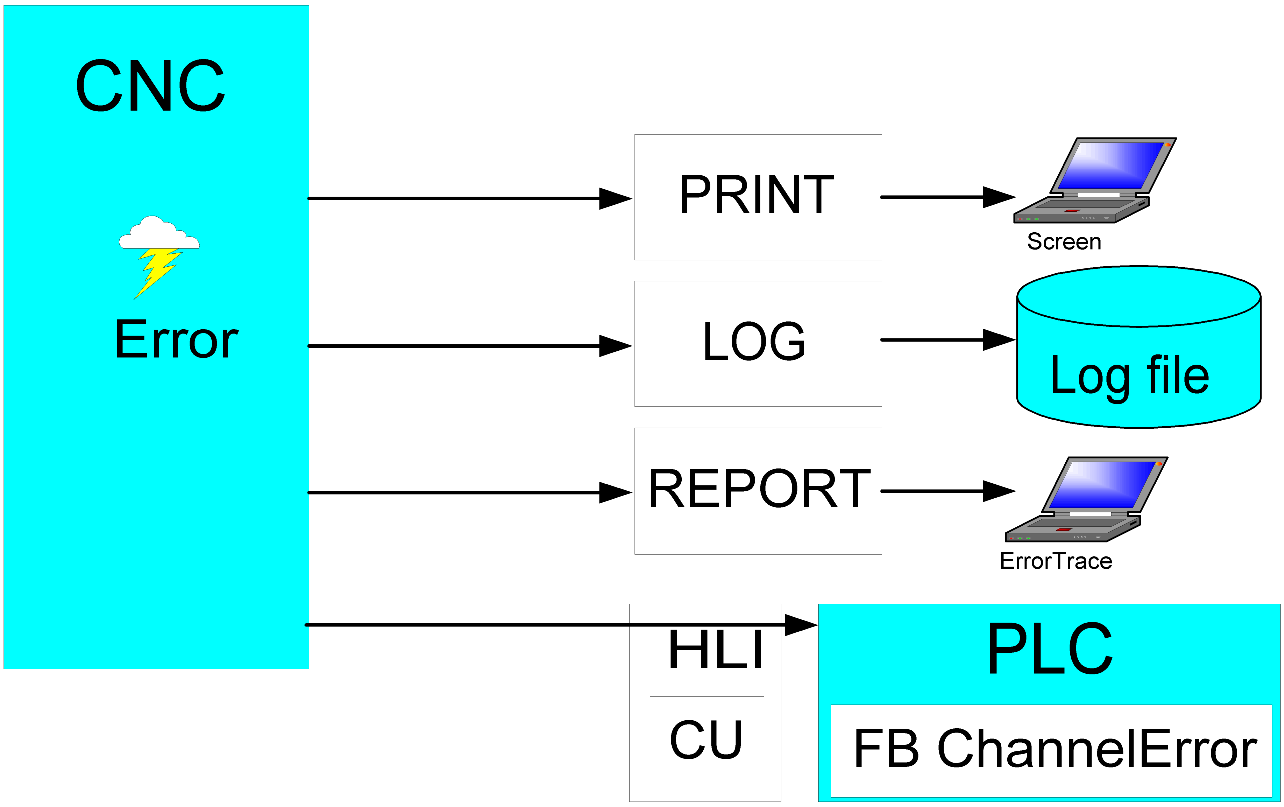


Figure 4: Overview of output options with the PLC block interface

If the PLC is no longer required to evaluate other CNC error messages, the output to the PLC can be completely skipped via the ControlUnit (Cu) on the HLI.

```
error_protocol_mode PRINT | LOG | REPORT | SEND_NOT_TO_PLC
```

Outputs under TwinCAT 3

In TwinCAT the PRINT channel is interfaced to the Event Logger output:

2.1.1 Suppressing all warnings

To avoid annoying end-users with a barrage of warnings, it may be useful to suppress them in the user display and log them to log file (P-STUP-00170 [▸ 33]).

When the controller is reset, a warning is generated by default. However, the warning can be suppressed via the P-STUP-00166 [▸ 32] parameter. This requires a change to the logging mode setting:

Logging mode: Filtering warnings – TC3_EventLogger

Warnings only to log file

```
error_protocol_mode LOG | PRINT | REPORT | TC3_EVENT_LOGGER | PRINT_NO_WARNINGS | REPORT_NO_WARNINGS
```

Suppress warnings even to log file

```
error_protocol_mode LOG | PRINT | REPORT | TC3_EVENT_LOGGER | PRINT_NO_WARNINGS | REPORT_NO_WARNINGS
| LOG_NO_WARNINGS
```

Suppress all warnings: This also includes suppressing the output of warnings to the PLC.

```
error_protocol_mode LOG | PRINT | REPORT | TC3_EVENT_LOGGER | NO_WARNINGS
```

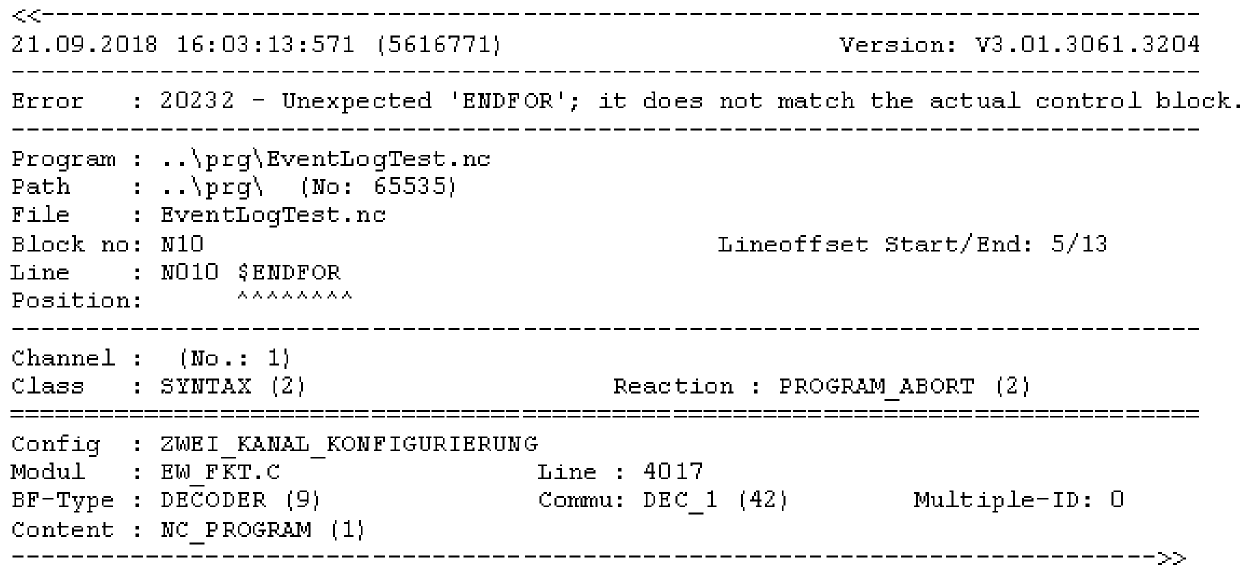
This is identical with:

```
error_protocol_mode LOG | PRINT | REPORT | TC3_EVENT_LOGGER | PRINT_NO_WARNINGS | REPORT_NO_WARNINGS
| LOG_NO_WARNINGS | SEND_TO_PLC_NO_WARNINGS
```

2.1.2 Output examples

Logging to file

The output of an error message to the logging file takes place as follows:



```
<<-----------------------------------------------------------------------------
21.09.2018 16:03:13:571 (5616771)                        Version: V3.01.3061.3204
-------------------------------------------------------------------------------
Error   : 20232 - Unexpected 'ENDFOR'; it does not match the actual control block.
-------------------------------------------------------------------------------
Program : ..\prg\EventLogTest.nc
Path    : ..\prg\  (No: 65535)
File    : EventLogTest.nc
Block no: N10                                    Lineoffset Start/End: 5/13
Line    : N010 $ENDFOR
Position:      ^^^^^^^^
-------------------------------------------------------------------------------
Channel :  (No.: 1)
Class   : SYNTAX (2)                      Reaction : PROGRAM_ABORT (2)
===============================================================================
Config  : ZWEI_KANAL_KONFIGURIERUNG
Modul   : EW_FKT.C                 Line : 4017
BF-Type : DECODER (9)              Commu: DEC_1 (42)          Multiple-ID: 0
Content : NC_PROGRAM (1)
----------------------------------------------------------------------------->>
```

Figure 5: Output example - log file

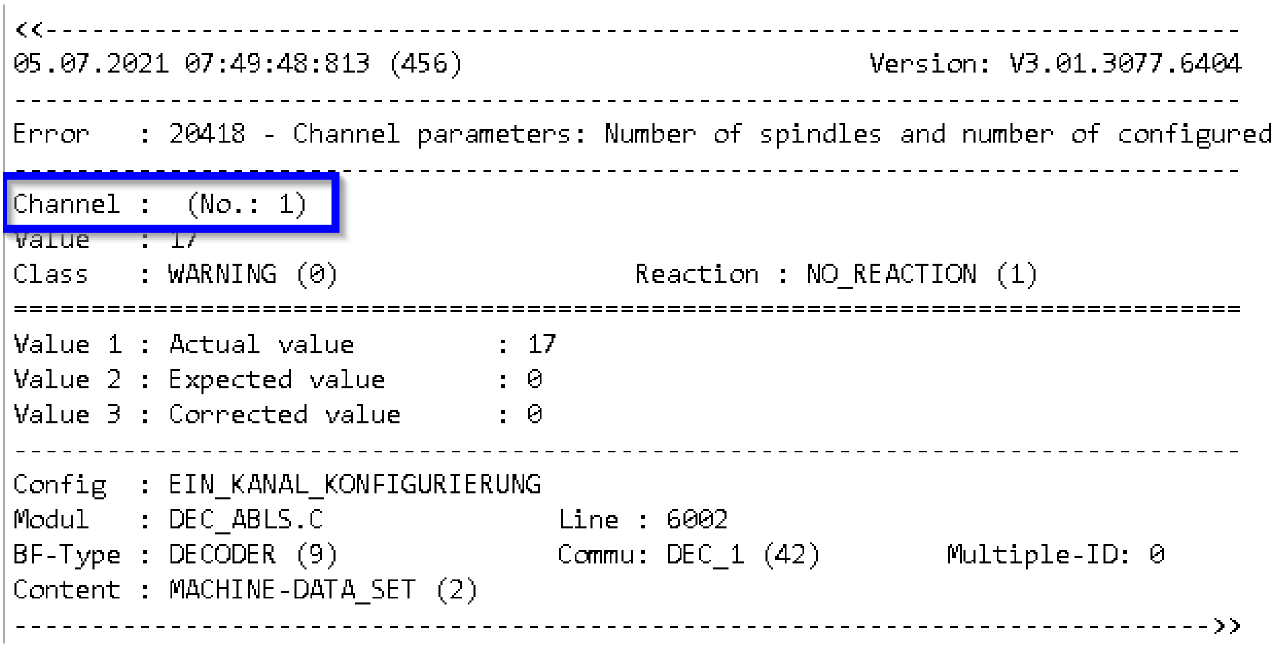
2.2 Filtering error messages

The output of error messages can be filtered globally or specifically by channel or axis

Identifying the filter level

The type of message in the log file indicates the level on which the error message or warning must be filtered.

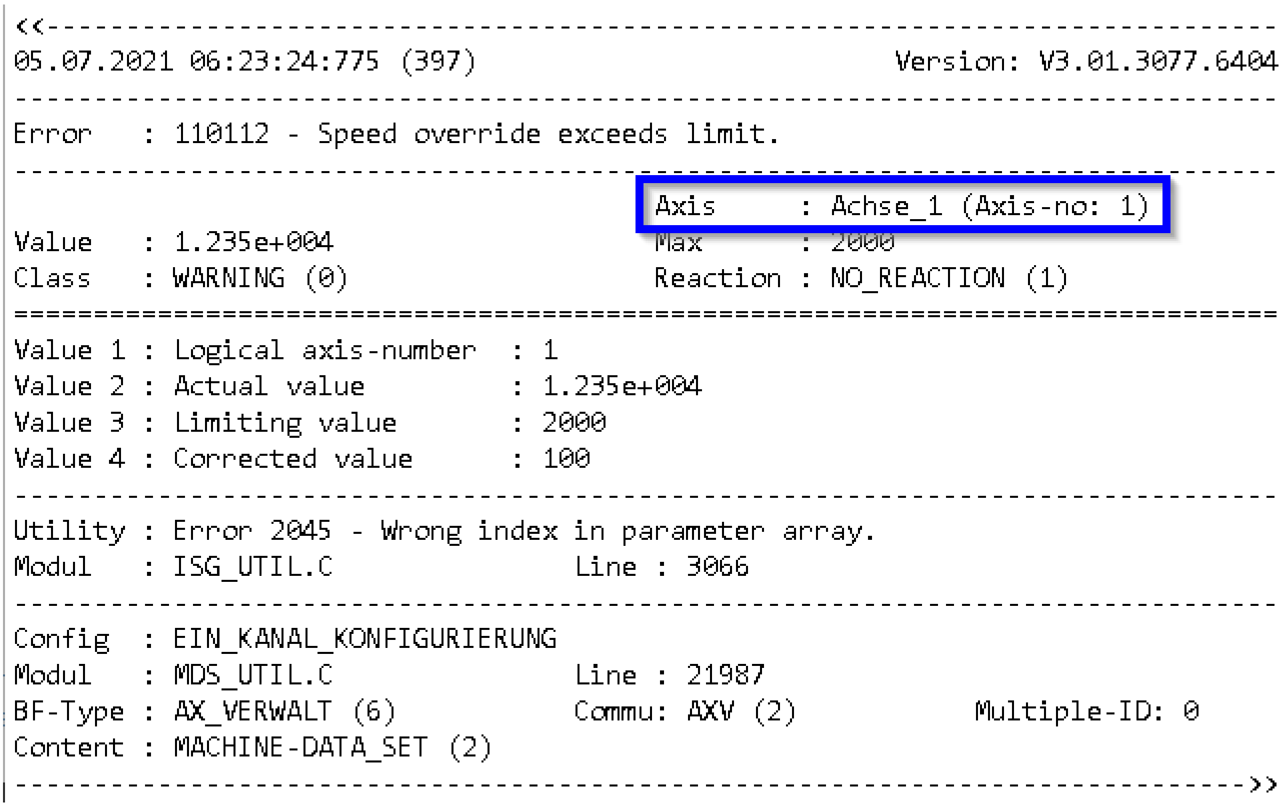
Channel error messages or warnings are identified as follows:



```
<<-------------------------------------------------------------------------------
05.07.2021 07:49:48:813 (456)                             Version: V3.01.3077.6404
---------------------------------------------------------------------------------
Error    : 20418 - Channel parameters: Number of spindles and number of configured
---------------------------------------------------------------------------------
Channel :   (No.: 1)
Value    : 17
Class    : WARNING (0)                        Reaction : NO_REACTION (1)
=================================================================================
Value 1 : Actual value            : 17
Value 2 : Expected value          : 0
Value 3 : Corrected value         : 0
---------------------------------------------------------------------------------
Config   : EIN_KANAL_KONFIGURIERUNG
Modul    : DEC_ABLS.C                   Line : 6002
BF-Type : DECODER (9)                   Commu: DEC_1 (42)         Multiple-ID: 0
Content : MACHINE-DATA_SET (2)
------------------------------------------------------------------------------->>
```

Figure 6: Message from the channel

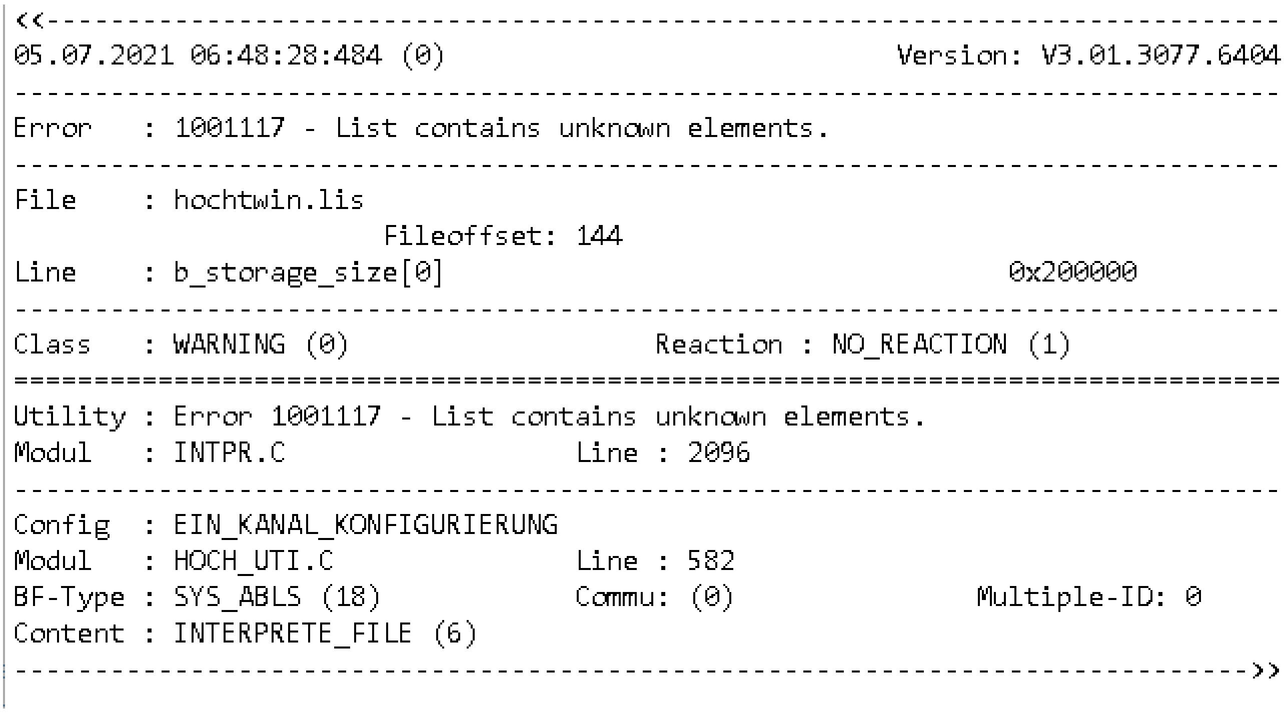
Axis error messages or warnings are identified as follows:



```
<<-----------------------------------------------------------------------------
05.07.2021 06:23:24:775 (397)                          Version: V3.01.3077.6404
-------------------------------------------------------------------------------
Error    : 110112 - Speed override exceeds limit.
-------------------------------------------------------------------------------
                                        Axis     : Achse_1 (Axis-no: 1)
Value   : 1.235e+004                    Max      : 2000
Class   : WARNING (0)                   Reaction : NO_REACTION (1)
===============================================================================
Value 1 : Logical axis-number  : 1
Value 2 : Actual value         : 1.235e+004
Value 3 : Limiting value       : 2000
Value 4 : Corrected value      : 100
-------------------------------------------------------------------------------
Utility : Error 2045 - Wrong index in parameter array.
Modul   : ISG_UTIL.C               Line : 3066
-------------------------------------------------------------------------------
Config  : EIN_KANAL_KONFIGURIERUNG
Modul   : MDS_UTIL.C               Line : 21987
BF-Type : AX_VERWALT (6)           Commu: AXV (2)           Multiple-ID: 0
Content : MACHINE-DATA_SET (2)
----------------------------------------------------------------------------->>
```

Figure 7: Message from an axis

A message from the start-up has neither an axis nor a channel entry:



```
<<------------------------------------------------------------------------------
05.07.2021 06:48:28:484 (0)                             Version: V3.01.3077.6404
--------------------------------------------------------------------------------
Error    : 1001117 - List contains unknown elements.
--------------------------------------------------------------------------------
File     : hochtwin.lis
                       Fileoffset: 144
Line     : b_storage_size[0]                                    0x200000
--------------------------------------------------------------------------------
Class    : WARNING (0)                  Reaction : NO_REACTION (1)
================================================================================
Utility : Error 1001117 - List contains unknown elements.
Modul    : INTPR.C                 Line : 2096
--------------------------------------------------------------------------------
Config   : EIN_KANAL_KONFIGURIERUNG
Modul    : HOCH_UTI.C              Line : 582
BF-Type : SYS_ABLS (18)            Commu: (0)                Multiple-ID: 0
Content : INTERPRETE_FILE (6)
------------------------------------------------------------------------------>>
```

Figure 8: Message from the start-up

2.2.1 Programming example of filtering error messages

As an example, step 1 should be to filter the warning **ID 20048** and step 2 the filter should be extended by an additional test.

This is a warning in the channel; then define the following parameters:

Filtering a single error message

```
error_filter[0].reason 20048 (P-CHAN-00378)
(HIDE - no message output)
error_filter[0].conditional_action HIDE (P-CHAN-00381)
```

These settings suppress the output of the warning ID 20048.

In order to use this filter for multiple messages, such as IDs 20048, 20622 and 2169, the line would have to be extended as follows:

```
error_filter[0].reason 20048, 20622 , 21691
```

A maximum of 5 error IDs can be assigned in a filter.

Changing specific warnings for errors

```
error_filter[0].reason 20048 (P-CHAN-00378)
error_filter[0].action NONE (P-CHAN-00379 no action)

(Action: output as syntax error)
error_filter[0].conditional_action F_SYNTAX (P-CHAN-00381)
```

These settings change the warning ID 20048 in the output into an error of recovery class 2.

2.3 Activating filter rules

Filter rules can be dynamically activated or deactivated from the GUI or the PLC. Activation/deactivation can be carried out at platform level or specifically for a channel or axis,

depending on whether the corresponding bit is set in the “activation_bit” data item when the filter rules are defined.

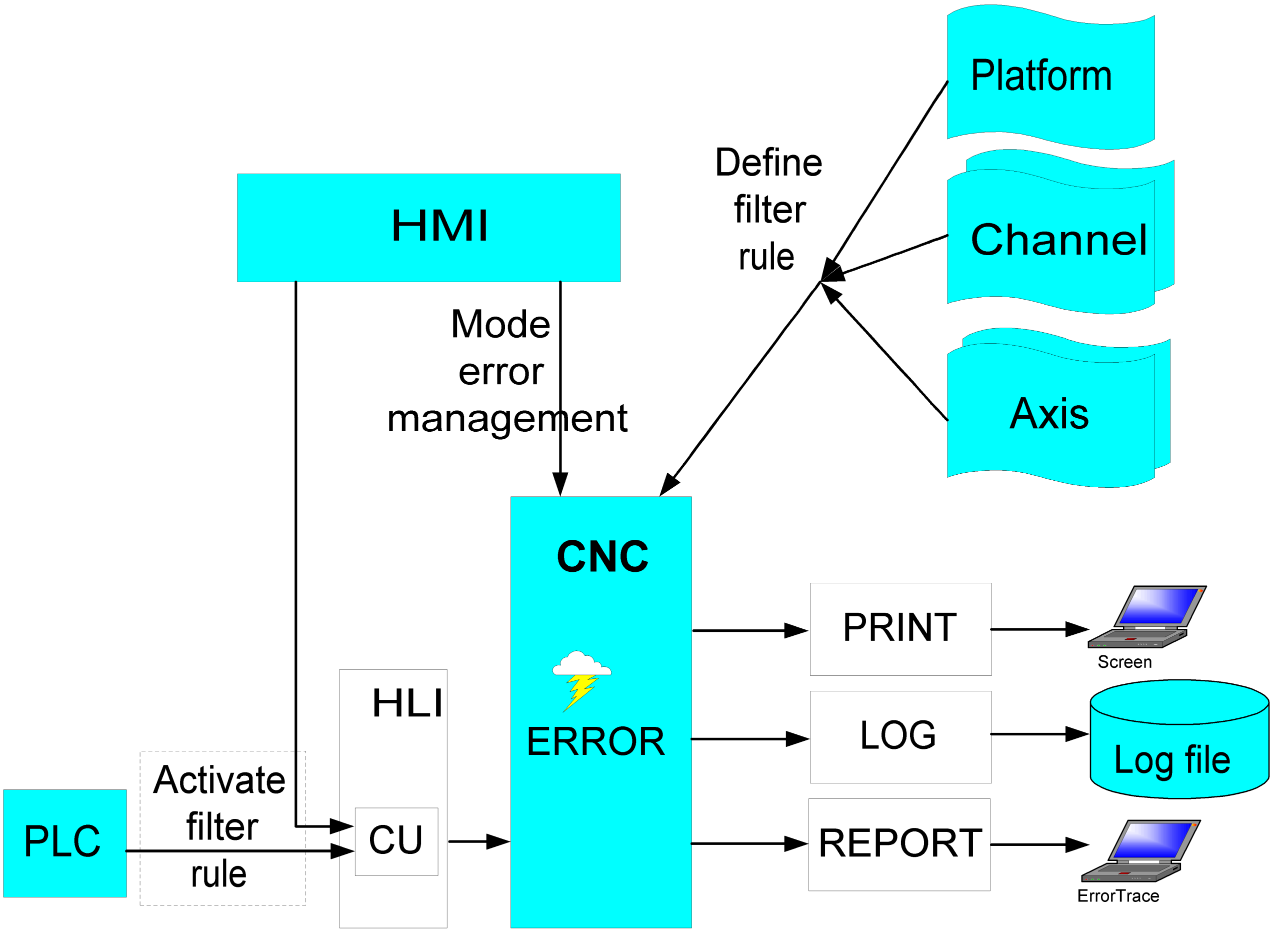


Figure 9: Activating filter rules

Activation from the GUI

This is dependent on the interaction between parameter and CNC object.

Consider the following pairings for use:

- Platform: P-STUP-00188 and cnc_error_filter_w [▶ 45]
- Channel-specific: P-CHAN-00380 and mc_error_error_filter_w [▶ 45]
- Axis-specific: P-AXIS-00629 and ac_<i>_error_filter_w [▶ 45]

Activation from the PLC

This is dependent on the interaction between parameter and the corresponding ControlUnit (CU).

Consider the following pairings for use:

- Platform: : P-STUP-00188 and CU error_filter [▶ 47]
- Channel-specific: P-CHAN-00380 and CU error_filter [▶ 47]
- Axis-specific: P-AXIS-00629 and CU error_filter [▶ 47]

2.4 Outputting user error messages

The user has the option of outputting user error messages in the NC program via the #ERROR [▶ 20] NC command.

The related error texts are saved to a text file with parameters defined in P-STUP-00169 [▶ 33]. The output is sent to the log file defined via P-STUP-00170 [▶ 33].

Error IDs saved to text file together with the related error texts are only written to the log file.

The range for possible error IDs is 1 – 1000.

Customer-specific error text in log file

Content of user-specific error text file:

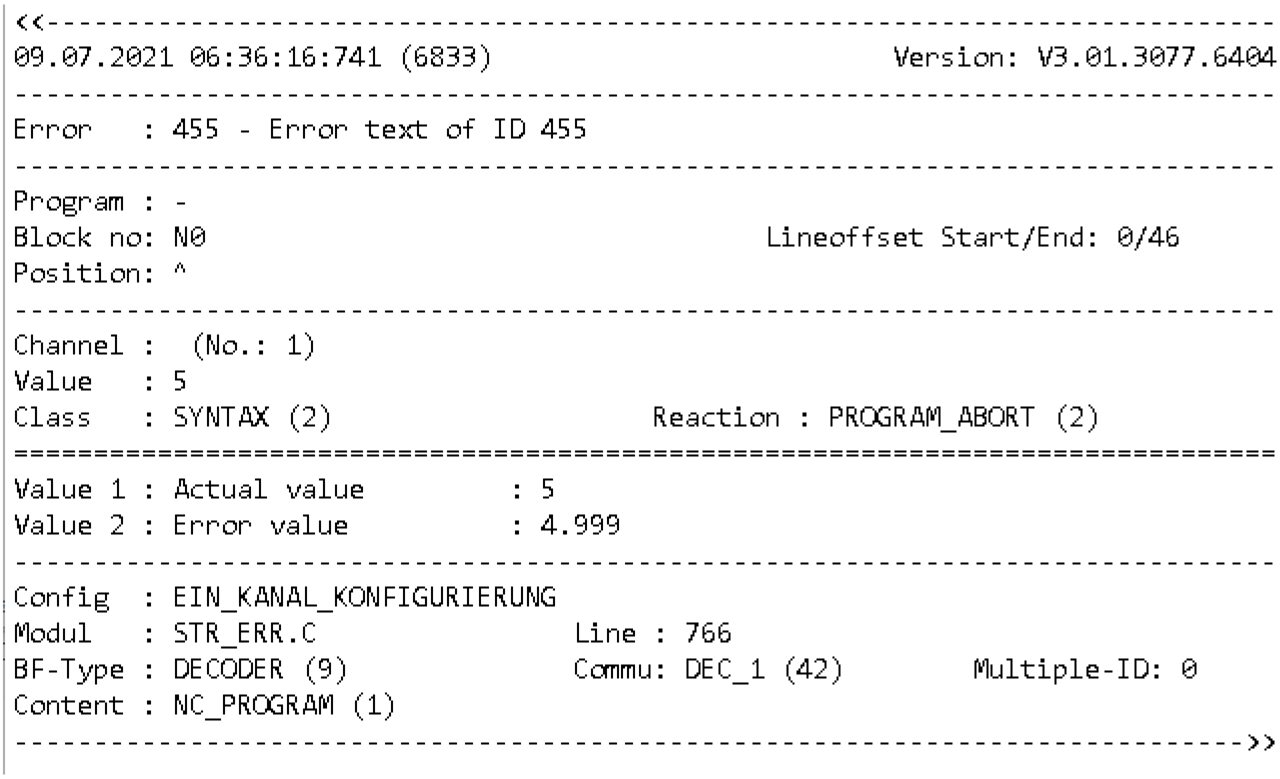
```
455 error text to ID455
```

The call in the NC program then looks like this:

```
#ERROR [ID455 RC2 PV1=5 PV2=4.999 PM1=2 PM2=3]
```

Other information on the #ERROR command: [PROG://User-defined error output [▶ 20]]

The output in the log file:



```
<<-----------------------------------------------------------------------
09.07.2021 06:36:16:741 (6833)                        Version: V3.01.3077.6404
-------------------------------------------------------------------------
Error    : 455 - Error text of ID 455
-------------------------------------------------------------------------
Program : -
Block no: N0                                   Lineoffset Start/End: 0/46
Position: ^
-------------------------------------------------------------------------
Channel :  (No.: 1)
Value   : 5
Class   : SYNTAX (2)                     Reaction : PROGRAM_ABORT (2)
=========================================================================
Value 1 : Actual value         : 5
Value 2 : Error value          : 4.999
-------------------------------------------------------------------------
Config  : EIN_KANAL_KONFIGURIERUNG
Modul   : STR_ERR.C                 Line : 766
BF-Type : DECODER (9)               Commu: DEC_1 (42)       Multiple-ID: 0
Content : NC_PROGRAM (1)
----------------------------------------------------------------------->>
```

Figure 10: Output of user error texts to log file

For user-defined error messages under TwinCAT3, see: Output of user-defined error messages to HMI [▶ 23]

Also see about this

- Output of user-defined error messages to HMI [▶ 23]

2.4.1 User-defined error output (#ERROR)

The NC command #ERROR allows the output of user-defined error messages which are further processed by the higher-level GUI (GUI = Graphical User Interface). Additional parameters offer the option of specifying the error more precisely.

#ERROR [[**ID**<*expr*>] [**RC**<*expr*>] [**MID**<*expr*>] {**PV**<*I*><*expr*>} {**PM**<*I*><*expr*>} {**PIV**<*i*><*expr*>} **]** (non-modal)

ID<*expr*> Error number:

Valid values	Meaning
1...1000	The numerical value determines the user-specific error number to be output.

RC<*expr*> Error class

Valid values	Meaning
0	Warning: No transition to error state. Program execution is continued.
2	Error: Transition to error state. Can be cleared with NC–RESET.
7	Fatal error: Transition to ‘system error’ state. Requires restart of controller.

MID<*expr*> Multiple ID. Counter acts as a distinguishing feature if the #ERROR command with the same error number (ID) is used several times in an NC program. MID must be a positive integer.

PV<*i*><*expr*> Max. 5 (1 <= i <= 5) user-specific numerical values (PV1...PV5) in real format can also be output in the error message.

PM<*i*><*expr*> The maximum of 5 (1 <= i <= 5) PM parameters (PM1...PM5) specify the meaning of the PV parameters more precisely.

Valid values	Meaning
0	IGNORE, value has no meaning
1	Limit value
2	Current value
3	Error value
4	Expected value
5	Corrected value
6	Logical axis number
7	Drive type
8	Logical control element number
9	State
10	Transition
11	Sender
12	Class
13	Instance
14	Identification number
15	Status
16	Ring number
17	Block number
18	Lower limit value

19	Upper limit value
20	Initial value
21	Final value

PIV<*i*><*expr*> The maximum of 4 (1 <= i <= 4) PIV parameters (PIV1...PIV4) transfer additional information in real format.

For non-programmed parameters, the following default values are valid:

ID	1
RC	0
MID	0
PV1...PV5	0.0
PM1…PM5	1
PIV1...PIV4	0.0

User-defined error output

```
:
#ERROR [ID100 RC0 MID10]    ;Warning 100, multiple id 10
:
#ERROR [ID455 RC2 PV1=5 PV2=4.999 PM1=2 PM2=3]    ;Error 455 with
                                                  ;parameters
:
#ERROR [ID455 RC2 MID2 PV1 5 PV2 4.999 PM1 2 PM2 3]    ;Error 455
                                                       ;Multiple id 2
                                                       ;with parameters
:
#ERROR [ID999 RC7]          ;Fatal error 999
:
```

2.4.2 Output of user-defined error messages to HMI

The output of user-defined error messages via the #ERROR [▶ 20] command can be displaced on the GUI of the TwinCAT3 system. The range for possible error IDs is 1 – 1000.

Error texts can be saved in multiple languages.

The the errors texts specified in the P-STUP-00169 file are not linked to the output to the HMI.

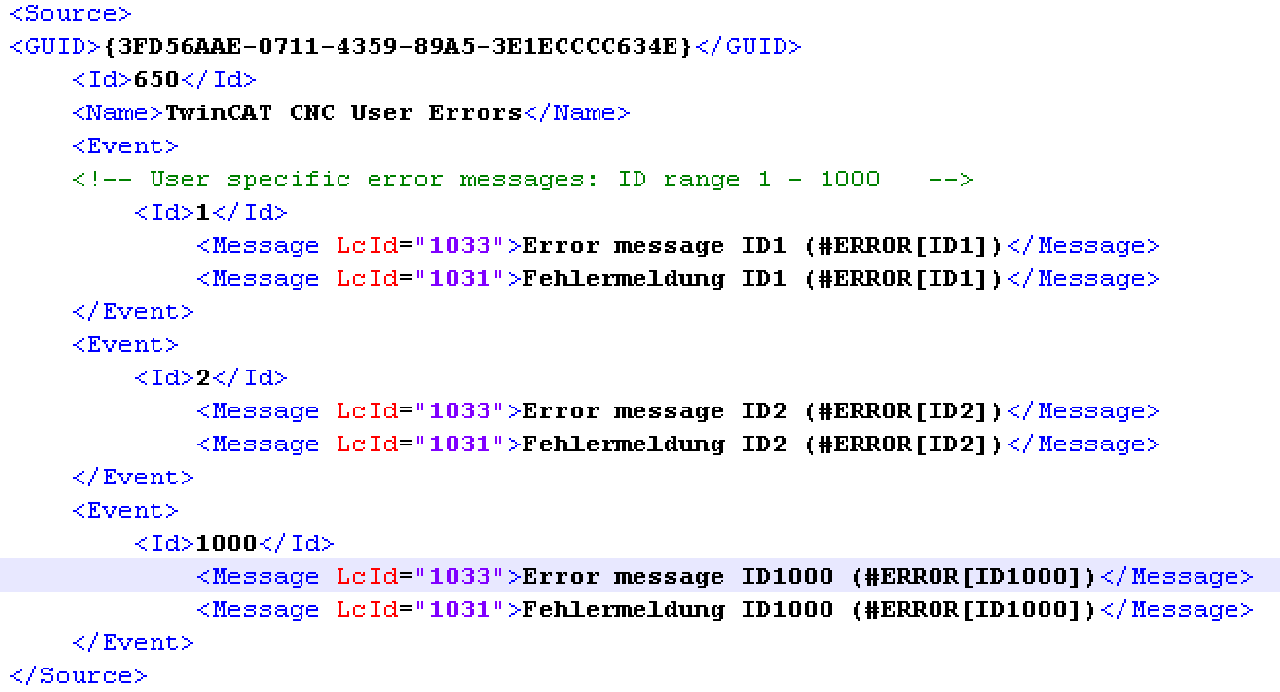
The required error texts must be integrated in both files.

Error texts for output to HMI must be integrated in the

C:\TwinCAT\3.1\Target\Resource\TcCncUserEvents.xml

file.

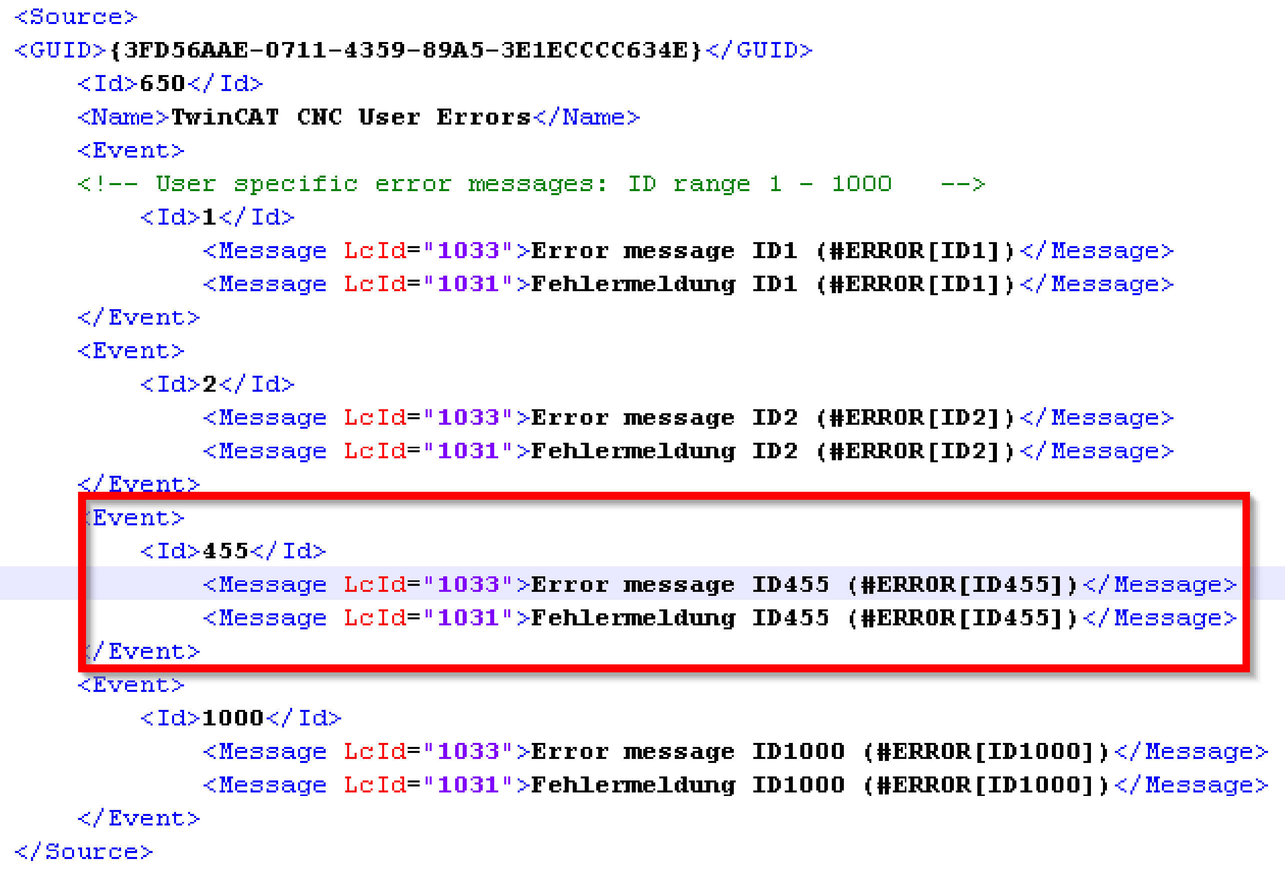
Integrating a message in TcCncUserEvents.xml



```
<Source>
<GUID>{3FD56AAE-0711-4359-89A5-3E1ECCCC634E}</GUID>
    <Id>650</Id>
    <Name>TwinCAT CNC User Errors</Name>
    <Event>
    <!-- User specific error messages: ID range 1 - 1000   -->
        <Id>1</Id>
            <Message LcId="1033">Error message ID1 (#ERROR[ID1])</Message>
            <Message LcId="1031">Fehlermeldung ID1 (#ERROR[ID1])</Message>
    </Event>
    <Event>
        <Id>2</Id>
            <Message LcId="1033">Error message ID2 (#ERROR[ID2])</Message>
            <Message LcId="1031">Fehlermeldung ID2 (#ERROR[ID2])</Message>
    </Event>
    <Event>
        <Id>1000</Id>
            <Message LcId="1033">Error message ID1000 (#ERROR[ID1000])</Message>
            <Message LcId="1031">Fehlermeldung ID1000 (#ERROR[ID1000])</Message>
    </Event>
</Source>
```

Figure 11: Code extract – starting point

As an example, the error with ID 455 is integrated in the file:



```
<Source>
<GUID>{3FD56AAE-0711-4359-89A5-3E1ECCCC634E}</GUID>
    <Id>650</Id>
    <Name>TwinCAT CNC User Errors</Name>
    <Event>
    <!-- User specific error messages: ID range 1 - 1000   -->
        <Id>1</Id>
            <Message LcId="1033">Error message ID1 (#ERROR[ID1])</Message>
            <Message LcId="1031">Fehlermeldung ID1 (#ERROR[ID1])</Message>
    </Event>
    <Event>
        <Id>2</Id>
            <Message LcId="1033">Error message ID2 (#ERROR[ID2])</Message>
            <Message LcId="1031">Fehlermeldung ID2 (#ERROR[ID2])</Message>
    </Event>
    <Event>
        <Id>455</Id>
            <Message LcId="1033">Error message ID455 (#ERROR[ID455])</Message>
            <Message LcId="1031">Fehlermeldung ID455 (#ERROR[ID455])</Message>
    </Event>
    <Event>
        <Id>1000</Id>
            <Message LcId="1033">Error message ID1000 (#ERROR[ID1000])</Message>
            <Message LcId="1031">Fehlermeldung ID1000 (#ERROR[ID1000])</Message>
    </Event>
</Source>
```

Figure 12: Code extract with integrated error text

With TwinCAT Build 4022.23 and higher, the configuration must be re-activated after integrating a user-defined error text.

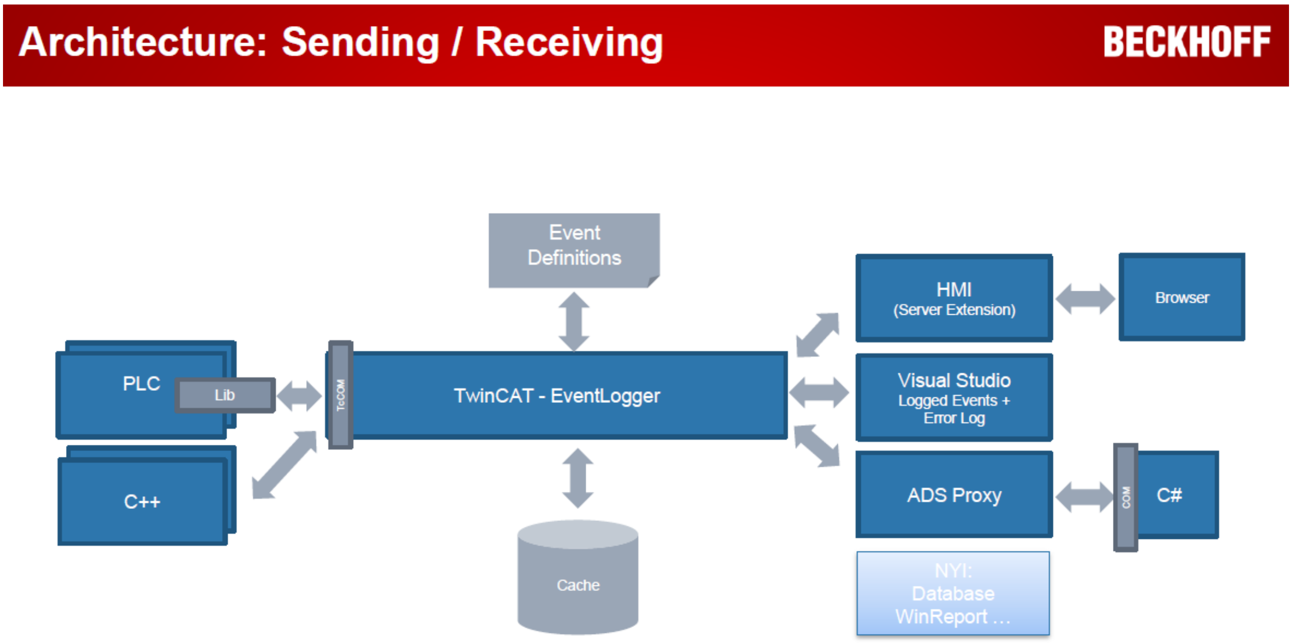
With TwinCAT versions < 4022.23, a computer reboot is required.

2.5 TwinCAT3 error output

Errors under TwinCAT3 are output

- via the PLC or
- directly in the TwinCAT Event Logger.

The overview diagram below shows the general function of the Event Logger:



Architecture: Sending / Receiving

Figure 13: Architecture - overview of the TwinCAT Event Logger

Outputs under TwinCAT 3

In TwinCAT the PRINT channel is interfaced to the Event Logger output:

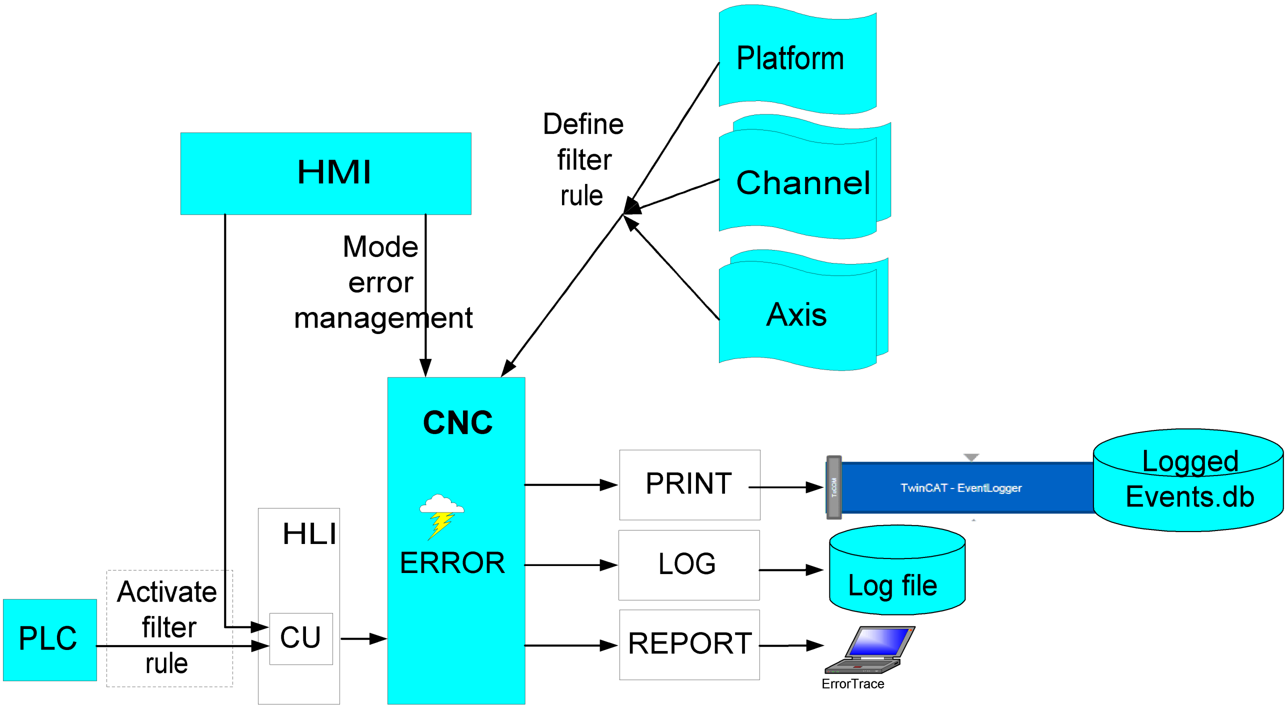


Figure 14: Overview with TwinCAT Event Logger

Default parameter definition

By default, errors are output to the PLC together with logging to the Event Logger via the original ChannelError function block.

If an existing CNC configuration is migrated from TwinCAT2 to TwinCAT3, the start-up parameters for error management require no adaptation in P-STUP-00167 [▶ 32] (error_protocol_mode).

Parameter	Parameter name	Setting
P-STUP-00167	error_protocol_mode	PRINT | LOG | REPORT
P-STUP-00168	error_text_of_id	<TC3-Install>\components\mc\cnc\diagnostics\err_text_version_eng.txt
P-STUP-00169	error_text_user_of_id	not assigned
P-STUP-00170	error_log_file_name	<TC3-Install> \components\mc\cnc\diagnostics\error.log
P-STUP-00171	error_log_file_max_size	100000 # maximum length in bytes
P-STUP-00172	error_plc_wait_cycles	5 # waiting cycles [IPO ticks]
P-STUP-00173	error_ao_name	(not assigned

Error texts

With a CNC release, language-specific error texts are automatically installed by the TcCncErrors-xml file.

Also see about this

- CNC objects [▶ 46]

2.5.1 Output only via PLC

If an output is to be sent to the Event Logger only via the ChannelError() function block (FB) of the PLC, this function can be selected by writing the "log errors [▶ 46]" CNC object. If the PLC assumes the logging function for error messages, activation normally takes place when the PLC is started.

Output via the ChannelError() FB

If errors are reported to the PLC via the PLC interface, errors can be evaluated within the PLC and reported to the Event Logger via a ChannelError() FB. Besides outputting the error, the PLC may execute other indirect reactions to a single error.

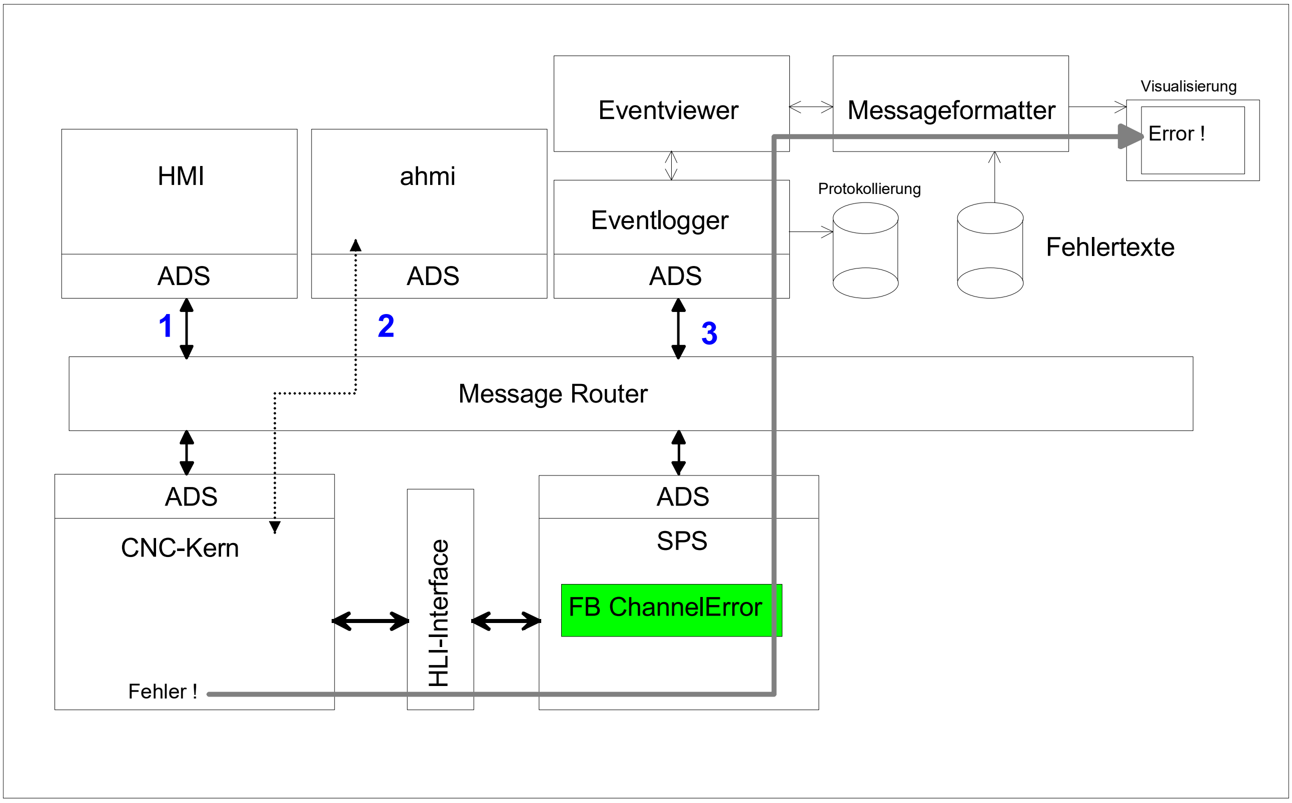


Figure 15: ChannelError() FB - positioning in the system

This output is identical with a TwinCAT2 output.

2.5.2 Direct output to Event Logger

In order to use the output directly to the “new” Event Logger, specify the keyword TC3_EVENT_LOGGER in P-STUP-00167 [▸ 32]. Errors are then logged directly by the CNC in the new format and no longer via the ChannelError() PLC block. Errors continue to be output to the PLC

```
error_protocol_mode PRINT | LOG | REPORT | TC3_EVENT_LOGGER
```

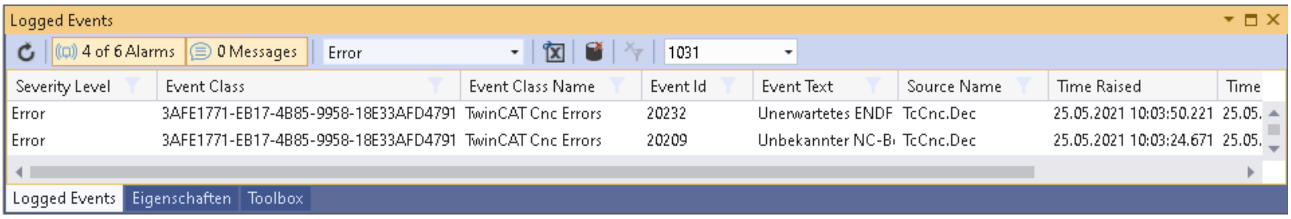
If the PLC is no longer required to evaluate CNC error messages, error output to the PLC can be suppressed by:

```
error_protocol_mode PRINT | LOG | REPORT | TC3_EVENT_LOGGER | SEND_NOT_TO_PLC
```

Alternatively, the mode can be set accordingly via the cnc_error_manager_mode_w [▸ 46] CNC object

2.5.2.1 Output format of Event Logger

In Visual Studio, the Event Logger can be output to a separate window.



Severity Level	Event Class	Event Class Name	Event Id	Event Text	Source Name	Time Raised	Time
Error	3AFE1771-EB17-4B85-9958-18E33AFD4791	TwinCAT Cnc Errors	20232	Unerwartetes ENDF	TcCnc.Dec	25.05.2021 10:03:50.221	25.05.
Error	3AFE1771-EB17-4B85-9958-18E33AFD4791	TwinCAT Cnc Errors	20209	Unbekannter NC-B	TcCnc.Dec	25.05.2021 10:03:24.671	25.05.

Figure 16: Event Logger output

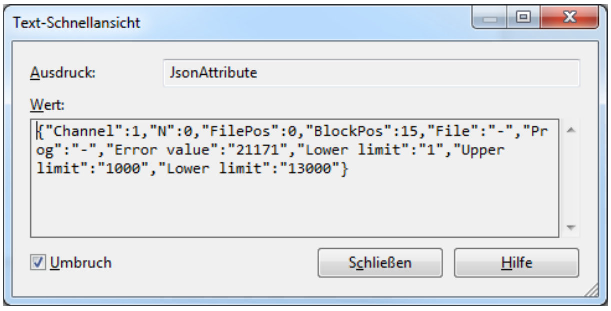
Overview output format of Event Logger

Tab name		Meaning

Severity level	Classification	Warning (0) / Error (1-6) / Critical (7)
EventClassName	Name of error class	TwinCAT Cnc Errors
SourceName	Error source	TcCNC Config / Diag / PosCtrl / PathPrep / AxesMgr / Commu / DrvMgr / Dec / TRC / Filter / FileMgr / SAI / IPO / Manual / SysMgr / Application
EventId	ID of the event	CNC-specific ID of the message occurring
Text	Error text	Multilingual error text
Time Raised	Release time	Date and time
Json Attributes	Json attributes	

Json attributes

Additional parameters of an error messages are output in Json format as key/value pairings. In particular, this provides more detailed error information about the NC channel or reference to the NC program.



Ausdruck: JsonAttribute

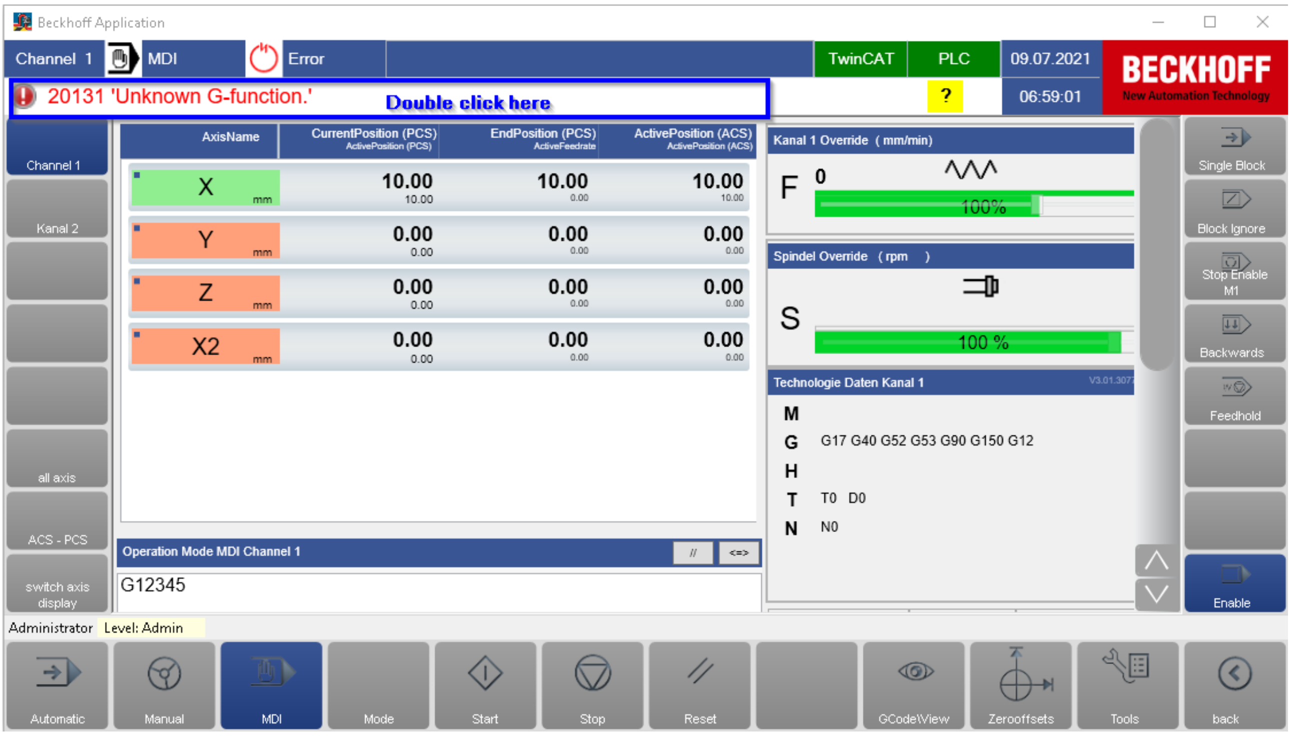
Wert:

```
{"Channel":1,"N":0,"FilePos":0,"BlockPos":15,"File":"-","Pr
og":"-","Error value":"21171","Lower limit":"1","Upper
limit":"1000","Lower limit":"13000"}
```

Figure 17: Output of Json attributes

2.5.3 Example of output to HMI

Error messages logged in the Event Logger are displayed accordingly in the GUI.



20131 'Unknown G-function.'

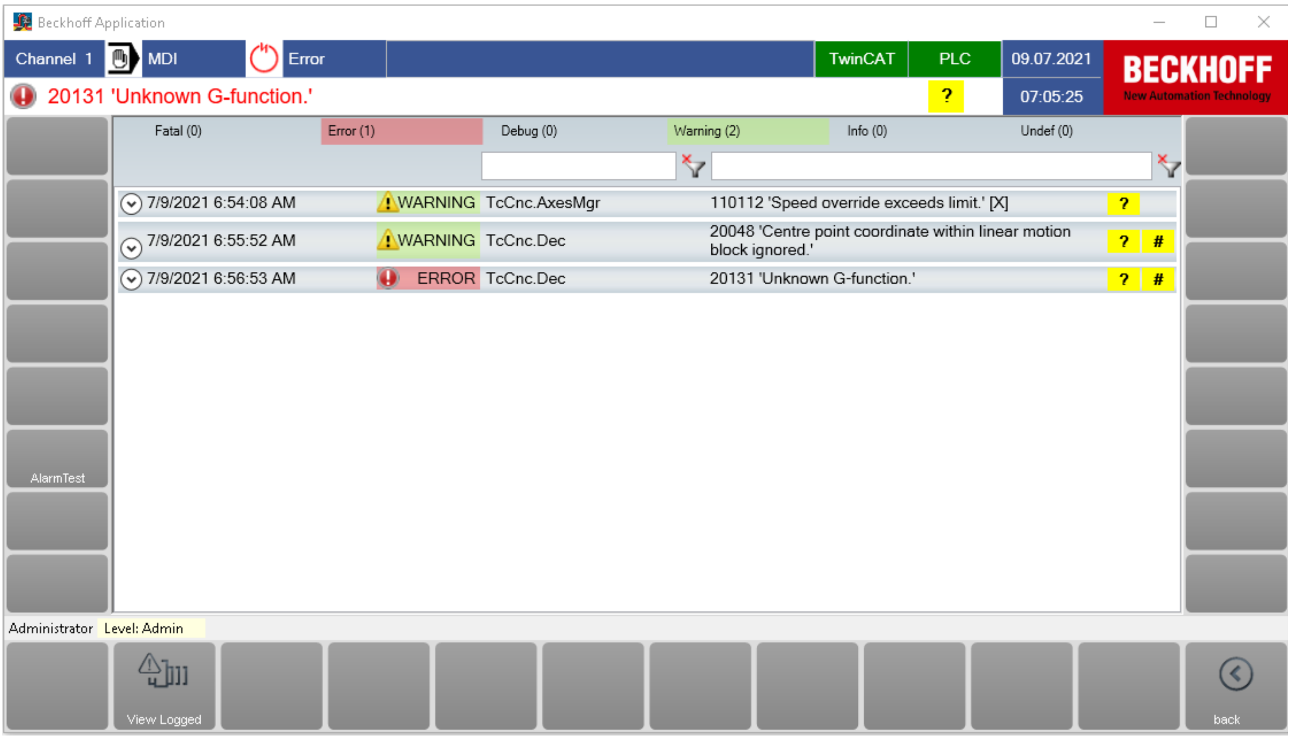
Double click here

AxisName	CurrentPosition (PCS) ActivePosition (PCS)	EndPosition (PCS) ActiveFeedrate	ActivePosition (ACS) ActivePosition (ACS)
X mm	10.00 10.00	10.00 0.00	10.00 10.00
Y mm	0.00 0.00	0.00 0.00	0.00 0.00
Z mm	0.00 0.00	0.00 0.00	0.00 0.00
X2 mm	0.00 0.00	0.00 0.00	0.00 0.00

M
G G17 G40 G52 G53 G90 G150 G12
H
T T0 D0
N N0

G12345

Figure 18: On-screen output of error message to the Beckhoff HMI



20131 'Unknown G-function.'

Fatal (0)	Error (1)	Debug (0)	Warning (2)	Info (0)	Undef (0)

7/9/2021 6:54:08 AM	WARNING	TcCnc.AxesMgr	110112 'Speed override exceeds limit.' [X]	?	
7/9/2021 6:55:52 AM	WARNING	TcCnc.Dec	20048 'Centre point coordinate within linear motion block ignored.'	?	#
7/9/2021 6:56:53 AM	ERROR	TcCnc.Dec	20131 'Unknown G-function.'	?	#

Figure 19: On-screen output after double-click

3 Parameter

3.1 Overview

Start-up parameters

ID	Parameter	Description
P-STUP-00166	no_error_message_at_reset	Logging a CNC reset
P-STUP-00167	error_protocol_mode	Logging mode
P-STUP-00168	error_text_of_id	Name of the file for error message texts
P-STUP-00169	error_text_user_of_id	Name of the file for user-specific error message texts
P-STUP-00170	error_log_file_name	Name of the error log file
P-STUP-00171	error_log_file_max_size	Maximum size of the error log file
P-STUP-00172	error_plc_wait_cycles	Waiting cycles before evaluation of PLC activation
P-STUP-00173	error_ao_name	Additional descriptive text (AO name)
P-STUP-00186	error_filter[i].reason	Identification of the error (cause)
P-STUP-00187	error_filter[i].action	Error action to be performed
P-STUP-00188	error_filter[i].conditional_activation	Activation via PLC/HMI
P-STUP-00189	error_filter[i].conditional_action	Action that is only activated after enabling by PLC/HMI
P-STUP-00190	error_filter[i].conditional_param	Additional comparison parameter in the event of a drive error message
P-STUP-00191	error_filter[i].conditional_output	Individual additional error information for output

Channel parameters

ID	Parameter	Description
P-CHAN-00338	mdi_log_file	Name of the manual block log file
P-CHAN-00339	mdi_log_file_max_size	Maximum size of the manual block log file
P-CHAN-00378	error_filter[i].reason	Identification of the error (cause)
P-CHAN-00379	error_filter[i].action	Error action to be performed
P-CHAN-00380	error_filter[i].conditional_activation	Activation via PLC/HMI
P-CHAN-00381	error_filter[i].conditional_action	Action that is only to be activated after enabling by PLC/HMI
P-CHAN-00382	error_filter[i].conditional_param	Additional comparison parameter in the event of a drive error message
P-CHAN-00383	error_filter[i].conditional_output	Individual additional error information for output

Axis parameter

ID	Parameter	Description
P-AXIS-00627	error_filter[i].reason	Identification of the error (cause)
P-AXIS-00628	error_filter[i].action	Error action to be performed
P-AXIS-00629	error_filter[i].conditional_activation	Activation via PLC/HMI
P-AXIS-00630	error_filter[i].conditional_action	Action that is only activated after enabling by PLC/HMI
P-AXIS-00631	error_filter[i].conditional_param	Additional comparison parameter in the event of a drive error message

P-AXIS-00632	error_filter[i].conditional_output	Individual additional error information for output

3.2 Description

3.2.1 Start-up parameters

P-STUP-00166	**Logging a CNC reset as event in error message output**
Description	This parameter defines whether the CNC reset triggered by the user is included as an event in the error message log. Previous error messages are acknowledged when the CNC is reset. This always occurs regardless of the setting of P-STUP-00166.
Parameter	no_error_message_at_reset
Data type	BOOLEAN
Data range	0: a CNC reset is logged as warning 270076 in the error message output. 1 a CNC reset is not logged
Dimension	----
Default value	0
Remarks	Parameter only effective in TwinCAT systems.

<table>
<tr><th><b>P-STUP-00167</b></th><th colspan="2"><b>Logging mode of error output</b></th></tr>
<tr><td>Description</td><td colspan="2">This parameter controls the output and scope of the error output.</td></tr>
<tr><td>Parameter</td><td colspan="2">error_protocol_mode</td></tr>
<tr><td>Data type</td><td colspan="2">STRING</td></tr>
<tr><td>Data range</td><td><b>Flag</b></td><td><b>Meaning</b></td></tr>
<tr><td></td><td>FILTER_OFF</td><td>No filters are evaluated</td></tr>
<tr><td></td><td>VERBOSE</td><td>Extended internal diagnostics</td></tr>
<tr><td></td><td>WITHOUT_ERROR_MANAGER</td><td>Direct output without error management</td></tr>
<tr><td></td><td>PRINT</td><td>Execute print output</td></tr>
<tr><td></td><td>LOG</td><td>Log output to log file</td></tr>
<tr><td></td><td>REPORT</td><td>Log output to log file</td></tr>
<tr><td></td><td>SEND_TO_PLC_NONE</td><td>Suppress output to the PLC</td></tr>
<tr><td></td><td>PRINT_EXTENDED</td><td>Extended print output</td></tr>
<tr><td></td><td>LOG_EXTENDED</td><td>Extended log output</td></tr>
<tr><td></td><td>REPORT_EXTENDED</td><td>Extended application-specific output</td></tr>
<tr><td></td><td>PRINT_NO_WARNINGS</td><td>Warnings are suppressed in the print output</td></tr>
<tr><td></td><td>LOG_NO_WARNINGS</td><td>Warnings are suppressed in the log output</td></tr>
<tr><td></td><td>REPORT_NO_WARNINGS</td><td>Warnings are suppressed in the report output</td></tr>
<tr><td></td><td>SEND_TO_PLC_NO_WARNINGS</td><td>Suppress warnings to PLC</td></tr>
<tr><td></td><td>STARTUP_NO_WARNINGS</td><td>Suppress warnings during controller start-up</td></tr>
<tr><td></td><td>NO_WARNINGS</td><td>Suppress all warnings</td></tr>
<tr><td></td><td>TC3_EVENT_LOGGER</td><td>Output to TC3 Event Logger</td></tr>
<tr><td>Unit</td><td colspan="2">----</td></tr>
<tr><td>Default value</td><td colspan="2">LOG | PRINT | REPORT</td></tr>
<tr><td>Remarks</td><td colspan="2"><b>Note</b>:
For example, to suppress warnings in the print output, the entire mode must be set accordingly.
error_protocol_mode LOG | PRINT | REPORT | PRINT_NO_WARNINGS</td></tr>
</table>

P-STUP-00168	**Name of the file for error message texts**

Description	Name of the file containing the error message texts which belong to the ID (error number). These can be used for output to the log file. This file is used to assign an error number to the related error message text. The file contains one line in the following format for each error ID: <Error-ID> TABULATOR <Error-Text> The default file 'err_text_version_eng.txt' is assumed if no file is specified.
Parameter	error_text_of_id
Data type	STRING
Data range	Maximum 256 characters
Dimension	----
Default value	-
Remarks	

P-STUP-00169	**Name of the file for user-specific error message texts**
Description	Comparable to default error texts (see P-STUP-00168), you can also specify user-specific texts in this file. These texts are used for error IDs in the range [1;1000] and for McCOM interface errors. This file is used to assign an error number to the related user-specific error message text. (ERR_KIN_TRAFO_CONFIG/-INITIALIZE/-FORWARD/-BACKWARD = 292030-292033). That is to say that, in this case, the returned error IDs of the McCOM methods are resolved.
Parameter	error_text_user_of_id
Data type	STRING
Data range	Maximum 256 characters
Dimension	----
Default value	-
Remarks	

P-STUP-00170	**Name of the error log file**
Description	Name of the error log file (including directory and path information). No log file is created if not name is specified. If the parameter is not configured, the error log file is generated with the default file name.
Parameter	error_log_file_name
Data type	STRING
Data range	Maximum 256 characters
Dimension	----
Default value	error.log
Remarks	

P-STUP-00171	**Maximum size of the error log file in bytes**
Description	This parameter defines the maximum size of the error log file.
Parameter	error_log_file_max_size
Data type	SGN32
Data range	> 0 :maximum size of the error log file. If this size is exceeded, the original file is copied to a backup file (extension: <name>.bak) and the contents of the original file are deleted. == 0 : no backup file is created.
Dimension	----
Default value	100000
Remarks	

P-STUP-00172	**Waiting cycles before evaluation of PLC activation**
Description	Waiting cycles in CNC ticks after an error has occurred before the PLC's activation mask for the filter rules is evaluated.
Parameter	error_plc_wait_cycles
Data type	UNS32
Data range	0 ... MAX(UNS32)
Dimension	----
Default value	-
Remarks	

P-STUP-00173	**Additional descriptive text (AO name)**
Description	Descriptive text (architecture object) that is additionally appended in the event of an error message.
Parameter	error_ao_name
Data type	STRING
Data range	Maximum 83 characters
Dimension	----
Default value	-
Remarks	

P-STUP-00186	**Cause of error**

Description	The individual error codes can be listed as numbers or texts, whereby the entire row must comply with the following syntax:
	(number | text) {, (number | text) }
	where:
	number:= CNC error number
	text:=" error-specific text "
	Example:
	error_filter[0].reason := "D012:", 123000, 123001
	If an error is logged, the program looks in the defined platform/channel/axis filters whether a user-specific filter rule is defined for it.

Parameter	error_filter[i].reason where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING
Data range	Maximum 96 characters
Dimension	----
Default value	-
Remarks	

P-STUP-00187	**Error action**
Description	Action that is to be performed if an error occurs.
Parameter	error_filter[i].action where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING

Data range	ACTION = NONE | DRIVE_STATE_REQ | PRE_RUN_STATE_REQ | RUN_STATE_REQ NONE: No action DRIVE_STATE_REQ: Read out drive status PRE_RUN_STATE_REQ: Error at start-up of the controller bus in PRE-run state RUN_STATE_REQ: Error at start-up of the controller bus in Run state
Dimension	----
Default value	-
Remarks	For SERCOS drive profiles: DRIVE_STATE_REQ: S-0-0095 diagnostic PRE_RUN_STATE_REQ: S-0-0021: list of unknown operation data in CP2 -> CP3, command 127 RUN_STATE_REQ: S-0-0022: list of unknown operation data in CP3 -> CP4, command 128 For ProfiDrive profiles: <all actions> Parameter 945 For CANopen profiles <all actions> Parameter ID603F

P-STUP-00188	**Conditional activation**
Description	This filter rule is activated when the applicable bit is set via the user interface or the PLC (HLI::Control Unit).
Parameter	error_filter[i].conditional_activation where i = 0 ... 3 (maximum Number of filters, application-specific)
Data type	UNS32
Data range	32-bit
Dimension	----
Default value	0
Remarks	Parameterisation example: *error_filter[0].conditional.action_activation 0x2* An activation bit = 0 means that the action is always executed.

P-STUP-00189	**Conditional action**
Description	Action that is to be executed if an error occurs and if the condition applies.
Parameter	error_filter[i].conditional_action where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING

Data range	ACTION = NONE | ([HIDE] [FORCE]) FORCE = F_WARNING | F_SYNTAX | F_ERROR | F_SEVERE | F_FATAL HIDE = [HIDE] [HIDE_LOG] [HIDE_PRINT] [HIDE_REPORT] NONE: no action HIDE: Suppress every error output HIDE_LOG: Error output to error log file is suppressed HIDE_DISPLAY: Error output is suppressed HIDE_REPORT: Application-specific error output is suppressed F_WARNING: Error is output as a WARNING (remedy class = 0) F_SYNTAX: Error is output as a syntax error (remedy class = 2) F_ERROR: Error due to NC program or other operator action (error remedy class = 5) F_SEVERE: Severe error, requires a warm start (remedy class = 6) F_FATAL: Severe error, requires a complete cold start (remedy class = 7)
Dimension	----
Default value	-
Remarks	

P-STUP-00190	**Conditional filter activation**

Description	The individual error codes can be listed as numbers or texts, whereby the entire row must comply with the following syntax:
	(number | text) {, (number | text) }
	
	where:
	number:= CNC error number
	text := **"** error-specific text **"**

Parameter	error_filter[i].conditional_param where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING
Data range	Maximum 96 characters
Dimension	----
Default value	-
Remarks	Parameterisation example: *error_filter[0].conditional.param* "D012:", 123, 1001 Individual error texts are only checked when the SERCOS drive error S95 is read out. Error numbers are only checked in case of SERCOS drive errors (S21 and S22) and in case of ProfiDrive drive errors (parameter 945).

P-STUP-00191	**Output of additional error information**
Description	This text is forwarded transparently via the CNC_ERROR_INFO data structure if the filter condition applies. This means the user has the option to output an additional error text.
Parameter	error_filter[i].conditional_output where i = 0 ... 3 (maximum number of filters, application-specific)

Data type	STRING
Data range	Maximum 32 characters
Dimension	----
Default value	-
Remarks	

3.2.2 Channel parameters

P-CHAN-00338	Name of the manual block log file
Description	If the name is specified, each manual block command of the NC channel is logged to this file. In addition to later diagnostics capability, this file is also used for error display. This means that, if a CNC error occurs within the manual block, the commanded manual block is immediately displayed in the error message.
Parameter	mdi_log_file
Data type	STRING
Data range	Maximum 256 characters
Dimension	----
Default value	*
Remarks	* Note: The default value of variable is a blank string.

P-CHAN-00339	Maximum size of the manual block log file
Description	As the log file grows with every new manual block, this parameter limits the file size. If the size of the log file is exceeded, it is first cleared automatically before the current manual block is logged.
Parameter	mdi_log_file_max_size
Data type	UNS32
Data range	0 ... MAX(UNS32)
Dimension	----
Default value	0 *
Remarks	* no size limitation

P-CHAN-00378	Error cause (filtering error messages in the channel)

Description	The individual error codes can be listed as numbers or texts, whereby the entire row must comply with the following syntax:
	(number | text) {, (number | text) }
	
	where:
	number:= CNC error number
	text:=" error-specific text "
	
	Example:
	error_filter[0].reason := "D012:", 123000, 123001
	
	If an error is logged, the program looks in the defined platform/channel/axis filters whether a user-specific filter rule is defined for it.

Parameter	error_filter[i].reason where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING
Data range	Maximum 96 characters
Dimension	----
Default value	*
Remarks	* Note: The default value of variable is a blank string.

P-CHAN-00379	Error action (filtering error messages in the channel)
Description	Action that is to be performed if an error occurs.
Parameter	error_filter[i].action where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING

<table>
<tr><td>Data range</td><td colspan="2">ACTION = NONE | DRIVE_STATE_REQ | PRE_RUN_STATE_REQ | RUN_STATE_REQ</td></tr>
<tr><td></td><td><b>Keyword</b></td><td><b>Meaning</b></td></tr>
<tr><td></td><td>NONE</td><td>No action</td></tr>
<tr><td></td><td>DRIVE_STATE_REQ</td><td>Reading out the drive status</td></tr>
<tr><td></td><td>PRE_RUN_STATE_REQ</td><td>Error at start-up of the controller bus in PRE-run state</td></tr>
<tr><td></td><td>RUN_STATE_REQ</td><td>Error at start-up of the controller bus in Run state</td></tr>
<tr><td>Unit</td><td colspan="2">----</td></tr>
<tr><td>Default value</td><td colspan="2">*</td></tr>
<tr><td>Remarks</td><td colspan="2">For SERCOS drive profiles:
• DRIVE_STATE_REQ S-0-0095 diagnostic
• PRE_RUN_STATE_REQ S-0-0021: list of unknown operation data in CP2 -> CP3, command 127
• RUN_STATE_REQ S-0-0022: list of unknown operation data in CP3 -> CP4, command 128
For ProfiDrive profiles:
• <all actions> Parameter 945
For CANopen profiles
• <all actions> Parameter ID603F
* Note: The default value of variable is a blank string.</td></tr>
</table>

P-CHAN-00380	**Conditional activation (filtering error messages in the channel)**
Description	This filter rule is activated when the corresponding bit is set via the user interface or the PLC (HLI::ControlUnit).
Parameter	error_filter[i].conditional_activation where i = 0 ... 3 (maximum Number of filters, application-specific)
Data type	UNS32
Data range	32 bits
Dimension	----
Default value	0
Remarks	Parameterisation example: *error_filter[0].conditional.action_activation 0x2* An activation bit = 0 means that the action is always performed.

P-CHAN-00381	**Conditional action (filtering error messages in the channel)**
Description	Action that is to be executed if an error occurs and if the condition applies.
Parameter	error_filter[i].conditional_action where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING

Data range	ACTION = NONE | ([HIDE] [FORCE]) FORCE = F_WARNING | F_SYNTAX | F_ERROR | F_SEVERE | F_FATAL HIDE = [HIDE] [HIDE_LOG] [HIDE_PRINT] [HIDE_REPORT] NONE: no action HIDE: Suppress every error output HIDE_LOG: Error output to error log file is suppressed HIDE_DISPLAY: Error output is suppressed HIDE_REPORT: Application-specific error output is suppressed F_WARNING: Error is output as a WARNING (remedy class = 0) F_SYNTAX: Error is output as a syntax error (remedy class = 2) F_ERROR: Error due to NC program or other operator action (error remedy class = 5) F_SEVERE: Severe error, requires a warm start (remedy class = 6) F_FATAL: Severe error, requires a complete cold start (remedy class = 7)
Dimension	----
Default value	*
Remarks	* Note: The default value of variable is a blank string.

P-CHAN-00382	**Conditional filter activation (filtering error messages in the channel))**

Description	The individual error codes can be listed as numbers or texts, whereby the entire row must comply with the following syntax:
	(number | text) {, (number | text) }
	
	where:
	number:= CNC error number
	text := **"** error-specific text **"**

Parameter	error_filter[i].conditional_param where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING
Data range	Maximum 96 characters
Dimension	----
Default value	*
Remarks	Parameterisation example: *error_filter[0].conditional.param* "D012:", 123, 1001 Individual error texts are currently only checked when the SERCOS drive error S95 is read out. Error numbers are only checked in case of SERCOS drive errors (S21 and S22) and in case of ProfiDrive drive errors (parameter 945). * Note: The default value of variable is a blank string.

P-CHAN-00383	**Output of additional error information (filtering error messages in the channel))**

Description	This text is forwarded transparently via the CNC_ERROR_INFO data structure if the filter condition applies. This means that the user has the option to output an additional error text conditionally.
Parameter	error_filter[i].conditional_output where i = 0 ... 3 (maximum number of filters, application-specific)
Data type	STRING
Data range	Maximum 32 characters
Dimension	----
Default value	*
Remarks	* Note: The default value of variable is a blank string.

3.2.3 Axis parameter

P-AXIS-00627	**Cause of error (filtering of axis error messages)**	

Description	The individual error codes can be listed as numbers or texts, whereby the entire row must comply with the following syntax:
	(number | text) {, (number | text) }
	
	where:
	number:= CNC error number
	text:="error-specific text"
	
	Example:
	error_filter[0].reason:="D012:", 123000, 123001
	
	If an error is logged, the program looks in the defined platform/channel/axis filters whether a user-specific filter rule is defined for it.	

Parameter	error_filter[i].reason where i = 0 ... 3 (maximum number of filters, application-specific)	
Data type	STRING	
Data range	Maximum 96 characters	
Axis types	T, R, S	
Dimension	T: ----	R,S: ----
Default value	*	
Remarks	* Note: The default value of variable is a blank string.	

P-AXIS-00628	**Error action (filtering of axis error messages)**	
Description	Action that is to be performed in the event of the applicable error occurring.	
Parameter	error_filter[i].action where i = 0 ... 3 (maximum number of filters, application-specific)	
Data type	STRING	
Data range	ACTION = NONE | DRIVE_STATE_REQ | PRE_RUN_STATE_REQ | RUN_STATE_REQ NONE: no action DRIVE_STATE_REQ: reading out the drive status PRE_RUN_STATE_REQ: Error during start-up of the controller bus in PRE-Run-State RUN_STATE_REQ: Error during start-up of the controller bus in Run-State	
Axis types	T, R, S	
Dimension	T: ----	R,S: ----
Default value	*	

Remarks	For SERCOS drive profiles: DRIVE_STATE_REQ S-0-0095 diagnostic PRE_RUN_STATE_REQ S-0-0021: list of unknown operation data in CP2 -> CP3, command 127 RUN_STATE_REQ S-0-0022: list of unknown operation data in CP3 -> CP4, command 128 For ProfiDrive profiles: <all actions> Parameter 945 For CANopen profiles <all actions> Parameter ID603F * Note: The default value of variable is a blank string.

P-AXIS-00629	Conditional activation (filtering of axis error messages)	
Description	This filter rule is activated when the applicable bit is set via the user interface or the PLC (HLI::Control Unit).	
Parameter	error_filter[i].conditional_activation where i = 0 ... 3 (maximum Number of filters, application-specific)	
Data type	UNS32	
Data range	32-bit	
Axis types	T, R, S	
Dimension	T: ----	R,S: ----
Default value	0	
Remarks	Parameterisation example: *error_filter[0].conditional.action_activation 0x2* An activation bit = 0 means that the action is always executed.	

P-AXIS-00630	Conditional action (filtering of axis error messages)	
Description	Action that is to be performed in the event of the applicable error occurring and when the condition applies.	
Parameter	error_filter[i].conditional_action where i = 0 ... 3 (maximum number of filters, application-specific)	
Data type	STRING	

Data range	ACTION = NONE | ([HIDE] [FORCE])
	FORCE = F_WARNING | F_SYNTAX | F_ERROR | F_SEVERE | F_FATAL
	HIDE = [HIDE] [HIDE_LOG] [HIDE_PRINT] [HIDE_REPORT]	

	NONE:	no action
	HIDE:	suppress every error output
	HIDE_LOG:	error output to error log file is suppressed
	HIDE_DISPLAY:	error output is suppressed
	HIDE_REPORT:	application-specific error output is suppressed
	F_WARNING:	error is output as a WARNING (remedy class = 0)
	F_SYNTAX:	error is output as a syntax error (remedy class = 2)
	F_ERROR:	error due to NC program or other operating action (error remedy class = 5)
	F_SEVERE:	severe error, requires a warm start (remedy class = 6)
	F_FATAL:	severe error, requires a complete cold start (remedy class = 7)
Axis types	T, R, S	
Dimension	T: ----	R,S: ----

Default value	*	
Remarks	* Note: The default value of variable is a blank string.	

P-AXIS-00631	**Conditional filter activation (filtering of axis error messages)**	

Description	The individual error identifiers can be listed as numbers of texts and the entire line must comply with the following syntax:
	(number | text) {, (number | text) }
	where:
	number:= CNC error number
	text:=**"** error-specific text **"**	

Parameter	error_filter[i].conditional_param where i = 0 ... 3 (maximum number of filters, application-specific)	
Data type	STRING	
Data range	Maximum of 96 characters	
Axis types	T, R, S	
Dimension	T: ----	R,S: ----
Default value	*	
Remarks	Configuration example: *error_filter[0].conditional.param* "D012:", 123, 1001 Individual error texts are currently only checked when the SERCOS drive error S95 is read out. Error numbers are checked only in the case of SERCOS drive errors (S21 and S22) and in the case of ProfiDrive drive errors (parameter 945). * Note: The default value of variable is a blank string.	

P-AXIS-00632	**Output of additional error information (filtering of axis error messages)**	
Description	This text is forwarded transparently via the CNC_ERROR_INFO data structure if the filter condition applies, i.e. users have a possibility of conditionally also including additional error information in the output.	
Parameter	error_filter[i].conditional_output where i = 0 ... 3 (maximum number of filters, application-specific)	
Data type	STRING	
Data range	Maximum of 32 characters	
Axis types	T, R, S	
Dimension	T: ----	R,S: ----
Default value	*	
Remarks	* Note: The default value of variable is a blank string.	

3.3 CNC objects

Notes on addressing

<C_{ID}> Channel or channel ID starting with 1

<A_{ID}> Axis ID starting with 1

<S_{ID}> Spindle index starting with 0

Activating filter rules

Name	cnc_error_filter_w		
Description	The individual filter rules of the platform can be activated and deactivated via the GUI by setting the applicable bitmask. This is dependent on the parameters defined in P-STUP-00188. Therefore, up to 32 different filter rules can be switched. Example of the assigned filter definition: `error_filter[0].activation_bit        0x1`		
Task	COM (Port 553)		
Index group	0x120100	**Index offset**	0x26C
Data type	UNS32	**length**	4
Attributes	write	**Unit**	-
Remarks	See Error filter definition [▶ 18] in start-up list		

Name	mc_error_filter_w		
Description	The individual filter rules of a channel can be activated and deactivated via the GUI by setting the applicable bitmask. This is dependent on the parameters defined in P-CHAN-00380. Therefore, up to 32 different filter rules can be switched. Example of the assigned filter definition: `error_filter[0].activation_bit        0x1`		
Task	COM (Port 553)		
Index group	0x12010<C$_{ID}$>	**Index offset**	0x419
Data type	UNS32	**length**	4
Attributes	write	**Unit**	-
Remarks			

Name	ac_<A$_{ID}$>_error_filter_w		
Description	The individual filter rules of an axis can be activated and deactivated via the GUI by setting the applicable bitmask. This is dependent on the parameters defined in P-AXIS-00629. Therefore, up to 32 different filter rules can be switched. Example of the assigned filter definition: `error_filter[0].activation_bit        0x1`		
Task	COM (Port 553)		
Index group	0x120200	**Index offset**	0x<A$_{ID}$>0028
Data type	UNS32	**length**	4
Attributes	write	**Unit**	-
Remarks			

Access to logging mode

Name	cnc_error_manager_mode_r		
Description	CNC object to read the error management mode. See P-STUP-00167.		
Task	COM (Port 553)		
Index group	0x120100	**Index offset**	0x293
Data type	UN32	**Length/byte**	4
Attributes	read	**Unit**	-
Remarks	The assignments and meanings of the bits are described in the table [▶ 46].		

Name	cnc_error_manager_mode_w		
Description	CNC object to write the error management mode. See P-STUP-00167.		
Task	COM (Port 553)		
Index group	0x120100	**Index offset**	0x26D
Data type	UN32	**Length/byte**	4
Attributes	write	**Unit**	-
Remarks	The assignments and meanings of the bits are described in the table [▶ 46].		

Flag	**Bit assignment**	**Meaning**
FILTER_OFF	0x00004	No filters are evaluated
VERBOSE	0x00008	Extended internal diagnostics
WITHOUT_ERROR_MANAGER	0x00001	Direct output without error management
PRINT	0x00010	Execute print output
LOG	0x00020	Log output to log file
REPORT	0x00040	Log output to log file
SEND_TO_PLC_NONE	0x20000	Suppress output to the PLC
PRINT_EXTENDED	0x00100	Extended print output
LOG_EXTENDED	0x00200	Extended log output
REPORT_EXTENDED	0x00400	Extended application-specific output
PRINT_NO_WARNINGS	0x01000	Warnings are suppressed in the print output
LOG_NO_WARNINGS	0x02000	Warnings are suppressed in the log output
REPORT_NO_WARNINGS	0x04000	Warnings are suppressed in the report output
SEND_TO_PLC_NO_WARNINGS	0x08000	Suppress warnings to PLC
STARTUP_NO_WARNINGS	0x10000	Suppress warnings during controller start-up
NO_WARNINGS	0x1f000	Suppress all warnings
TC3_EVENT_LOGGER	0x80000	Output to TC3 Event Logger

Name	log errors		
Description	This object defines whether CNC error messages are output to the PLC: (default = TRUE)		
Task	GEO (Port 551)		
Index group	0x120300	**Index offset**	0x4
Data type	BOOLEAN	**Length**	1
Attributes	read/ write	**Unit**	[-]
Remarks	Values : TRUE/FALSE		

3.4 HLI parameters

Activating error filter rules - Platform	
Description	Here, the individual error filters can be activated/deactivated by the 32-bit mask according to their activation bits. For example the following rule is activated by setting the first bit (command_w = 0x00000001): `error_filter[0].activation_bit        0x1`
Data type	MC_CONTROL_UNS32_UNIT, see description of control unit
Unit	[-]
Access	PLC reads request_r + state_r and writes command_w + enable_w
ST Path	gpPform^.error_filter
Commanded and requested value	
ST Element	**.command_w** **.request_r**
Data type	UDINT
Return value	
ST Element	**.state_r**
Data type	UDINT
Redirection	
ST Element	**.enable_w**

Activating error filter rules - Channel	
Description	Here, the individual error filters can be activated/deactivated by the 32-bit mask according to their activation bits. For example the following rule is activated by setting the first bit (command_w = 0x00000001): `error_filter[0].activation_bit        0x1`
Data type	MC_CONTROL_UNS32_UNIT, see description of control unit
Unit	[-]
Access	PLC reads request_r + state_r and writes command_w + enable_w
ST Path	gpCh*[channel_idx]*^.head.error_filter
Commanded and requested value	
ST Element	**.command_w** **.request_r**
Data type	UDINT
Return value	
ST Element	**.state_r**
Data type	UDINT
Redirection	
ST Element	**.enable_w**

Activating error filter rules - Axis

Description	Here, the individual error filters can be activated/deactivated by the 32-bit mask according to their activation bits. For example the following rule is activated by setting the first bit (command_w = 0x00000001): `error_filter[0].activation_bit        0x1`
Data type	MC_CONTROL_UNS32_UNIT, see description of control unit
Unit	[-]
Access	PLC reads request_r + state_r and writes command_w + enable_w
ST Path	gpAx*[axis_idx]*^.head.error_filter
Commanded and requested value	
ST Element	**.command_w** **.request_r**
Data type	UDINT
Return value	
ST Element	**.state_r**
Data type	UDINT
Redirection	
ST Element	**.enable_w**

4 Support and Service

Beckhoff and their partners around the world offer comprehensive support and service, making available fast and competent assistance with all questions related to Beckhoff products and system solutions.

Beckhoff's branch offices and representatives

Please contact your Beckhoff branch office or representative for local support and service on Beckhoff products!

The addresses of Beckhoff's branch offices and representatives round the world can be found on her internet pages: https://www.beckhoff.com

You will also find further documentation for Beckhoff components there.

Beckhoff Support

Support offers you comprehensive technical assistance, helping you not only with the application of individual Beckhoff products, but also with other, wide-ranging services:

- support
- design, programming and commissioning of complex automation systems
- and extensive training program for Beckhoff system components

Hotline:	+49 5246 963 157
Fax:	+49 5246 963 9157
e-mail:	support@beckhoff.com

Beckhoff Service

The Beckhoff Service Center supports you in all matters of after-sales service:

- on-site service
- repair service
- spare parts service
- hotline service

Hotline:	+49 5246 963 460
Fax:	+49 5246 963 479
e-mail:	service@beckhoff.com

Beckhoff Headquarters

Beckhoff Automation GmbH & Co. KG

Huelshorstweg 20
33415 Verl
Germany

Phone:	+49 5246 963 0
Fax:	+49 5246 963 198
e-mail:	info@beckhoff.com
web:	https://www.beckhoff.com

Index

More information:
www.beckhoff.com/TF5200

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Germany
Phone: +49 5246 9630
info@beckhoff.com
www.beckhoff.com