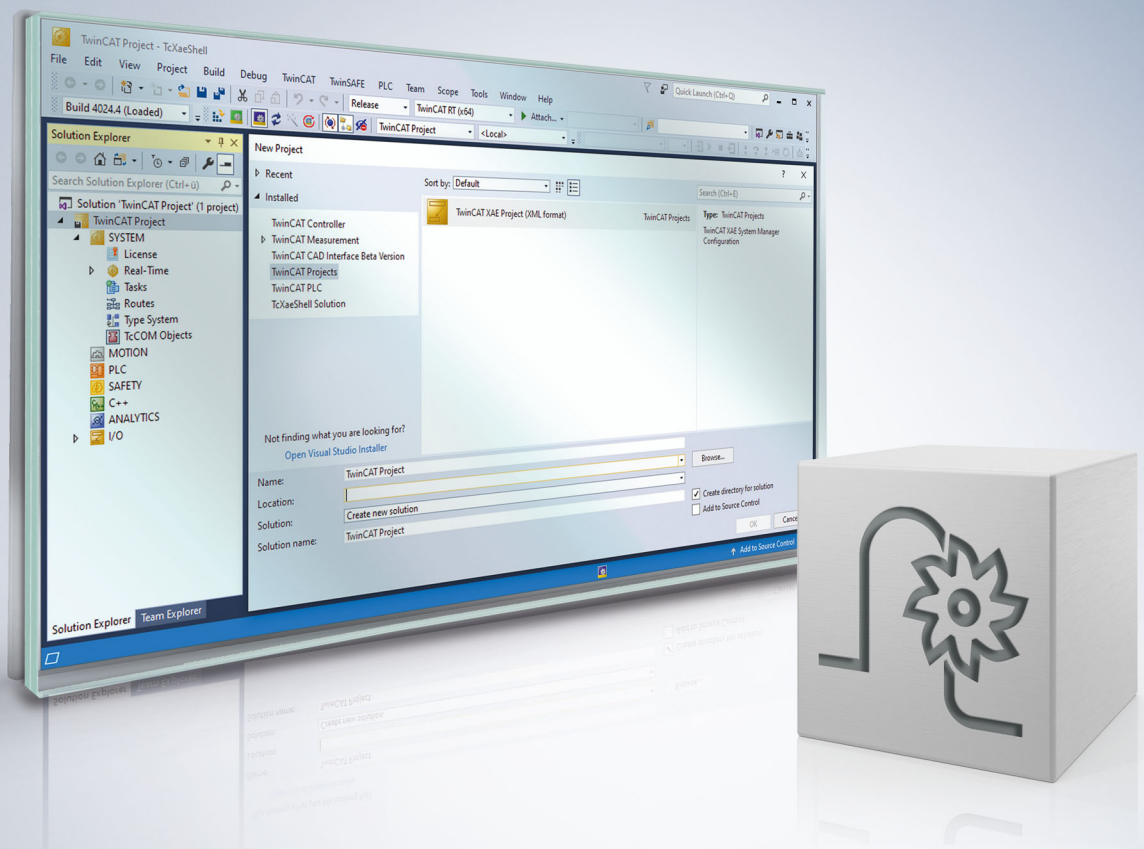
BECKHOFF New Automation Technology

Handbuch | DE

TF5270/71 | TwinCAT 3 CNC

Virtual NCK



21.12.2023 | Version: 1.1.57

Hinweise zur Dokumentation

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs- und Automatisierungstechnik, das mit den geltenden nationalen Normen vertraut ist.
Zur Installation und Inbetriebnahme der Komponenten ist die Beachtung der Dokumentation und der nachfolgenden Hinweise und Erklärungen unbedingt notwendig.
Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zu dem betreffenden Zeitpunkt veröffentliche Dokumentation zu verwenden.

Das Fachpersonal hat sicherzustellen, dass die Anwendung bzw. der Einsatz der beschriebenen Produkte alle Sicherheitsanforderungen, einschließlich sämtlicher anwendbaren Gesetze, Vorschriften, Bestimmungen und Normen erfüllt.

Disclaimer

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte werden jedoch ständig weiter entwickelt.
Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.
Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Marken

Beckhoff®, TwinCAT®, TwinCAT/BSD®, TC/BSD®, EtherCAT®, EtherCAT G®, EtherCAT G10®, EtherCAT P®, Safety over EtherCAT®, TwinSAFE®, XFC®, XTS® und XPlanar® sind eingetragene und lizenzierte Marken der Beckhoff Automation GmbH.
Die Verwendung anderer in dieser Dokumentation enthaltenen Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Patente

Die EtherCAT-Technologie ist patentrechtlich geschützt, insbesondere durch folgende Anmeldungen und Patente:
EP1590927, EP1789857, EP1456722, EP2137893, DE102015105702
mit den entsprechenden Anmeldungen und Eintragungen in verschiedenen anderen Ländern.

EtherCAT® ist eine eingetragene Marke und patentierte Technologie lizenziert durch die Beckhoff Automation GmbH, Deutschland

Copyright

© Beckhoff Automation GmbH & Co. KG, Deutschland.
Weitergabe sowie Vervielfältigung dieses Dokuments, Verwertung und Mitteilung seines Inhalts sind verboten, soweit nicht ausdrücklich gestattet.
Zuwiderhandlungen verpflichten zu Schadenersatz. Alle Rechte für den Fall der Patent-, Gebrauchsmuster- oder Geschmacksmustereintragung vorbehalten.

Allgemeine- und Sicherheitshinweise

Verwendete Symbole und ihre Bedeutung

In der vorliegenden Dokumentation werden die folgenden Symbole mit nebenstehendem Sicherheitshinweis und Text verwendet. Die (Sicherheits-) Hinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

Symbole im Erklärtext

1. Gibt eine Aktion an.

⇨ Gibt eine Handlungsanweisung an.

⚠ GEFAHR

Akute Verletzungsgefahr!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, besteht unmittelbare Gefahr für Leben und Gesundheit von Personen!

⚠ VORSICHT

Schädigung von Personen und Maschinen!

Wenn der Sicherheitshinweis neben diesem Symbol nicht beachtet wird, können Personen und Maschinen geschädigt werden!

HINWEIS

Einschränkung oder Fehler

Dieses Symbol beschreibt Einschränkungen oder warnt vor Fehlern.

Tipps und weitere Hinweise

Dieses Symbol kennzeichnet Informationen, die zum grundsätzlichen Verständnis beitragen oder zusätzliche Hinweise geben.

Allgemeines Beispiel

Beispiel zu einem erklärten Sachverhalt.

NC-Programmierbeispiel

Programmierbeispiel (komplettes NC-Programm oder Programmsequenz) der beschriebenen Funktionalität bzw. des entsprechenden NC-Befehls.

Spezifischer Versionshinweis

Optionale, ggf. auch eingeschränkte Funktionalität. Die Verfügbarkeit dieser Funktionalität ist von der Konfiguration und dem Versionsumfang abhängig.

Inhaltsverzeichnis

Hinweise zur Dokumentation ... **3**

Allgemeine- und Sicherheitshinweise ... **5**

1 CNC-Kern als DLL kernelv - Einleitung ... **12**

2 Beschreibung ... **13**

2.1 DLL-Version ... 13
2.2 Komponenten ... 13
2.3 Abhängigkeiten ... 14
2.3.1 kernelv ab V300 ... 14
2.4 Mehrfachinstanzierung ... 14
2.4.1 Diagnose bei Mehrfachinstanzierung ... 14
2.5 Lizenzschutz ... 15
2.5.1 Lizenzierung unter TwinCAT 2 ... 15
2.5.2 Lizenzierung unter TwinCAT 3 ... 15
2.6 Allgemeines ... 17
2.6.1 Identifikation von Achsen und Kanälen ... 17
2.6.2 Koordinatensysteme ... 17
2.7 Benutzung von kernelv ... 18
2.7.1 Start ... 18
2.7.2 Zyklischer Betrieb ... 18
2.7.3 Beenden/Neustart ... 18
2.7.4 Betriebsarten von kernelv ... 19
2.7.5 Quittieren von Technologiefunktionen ... 20
2.7.6 Suchpfad für NC-Programme ... 21
2.7.7 kernelv Demoapplikation ... 21
2.8 Konfiguration ... 24
2.8.1 Pfad der Konfigurationslisten ... 24
2.9 Fehlermeldungstexte ... 24
2.10 Werkzeugverwaltung ... 25

3 kernelv API Funktionen ... **26**

3.1 kernelv_get_api_version() ... 26
3.2 kernelv_get_cnc_version() ... 27
3.3 kernelv_get_cycletime() ... 28
3.4 kernelv_startup() ... 29
3.5 kernelv_startup_instance() ... 30
3.6 kernelv_do_cycle() ... 31
3.7 kernelv_ch_program_start() ... 32
3.8 kernelv_ch_reset() ... 33
3.9 kernelv_ch_suspend() ... 34
3.10 kernelv_ch_resume() ... 35
3.11 kernelv_ch_get_override() ... 36
3.12 kernelv_ch_set_override() ... 37
3.13 kernelv_ch_get_blocknumber() ... 38
3.14 kernelv_ch_get_filename() ... 39

3.15 kernelv_ch_get_programname() ... 40
3.16 kernelv_ch_get_state() ... 41
3.17 kernelv_ch_get_fileoffset() ... 42
3.18 kernelv_ch_get_techno_data() ... 43
3.19 kernelv_ch_get_new_techno_data() ... 44
3.20 kernelv_ch_get_techno_data2() ... 45
3.21 kernelv_ch_get_new_techno_data2() ... 46
3.22 kernelv_ch_get_finished_nc_lines() ... 47
3.23 kernelv_ax_get_techno_data() ... 48
3.24 kernelv_ax_get_new_techno_data() ... 49
3.25 kernelv_ax_get_techno_data2() ... 50
3.26 kernelv_ax_get_new_techno_data2() ... 51
3.27 kernelv_ax_set_position() ... 52
3.28 kernelv_get_acs_command_positions() ... 53
3.29 kernelv_get_acs0_command_positions() ... 54
3.30 kernelv_get_acs_actual_positions() ... 55
3.31 kernelv_get_acs_target_positions() ... 56
3.32 kernelv_get_acs_start_positions() ... 57
3.33 kernelv_get_wcs_command_positions() ... 58
3.34 kernelv_get_wcs_target_positions() ... 59
3.35 kernelv_get_wcs_start_positions() ... 60
3.36 kernelv_get_prg_target_positions() ... 61
3.37 kernelv_get_axis_channel_number() ... 62
3.38 kernelv_ch_get_variable_value() ... 63
3.39 kernelv_ch_set_variable_value() ... 65
3.40 kernelv_get_channel_count() ... 67
3.41 kernelv_get_axis_count() ... 68
3.42 kernelv_sync_read_request() ... 69
3.43 kernelv_sync_write_request() ... 70
3.44 kernelv_sync_read_write_req() ... 71
3.45 kernelv_get_axis_names() ... 73
3.46 kernelv_control_techno_func_duration() ... 75
3.47 kernelv_ch_set_techno_func_duration() ... 76
3.48 kernelv_ch_set_techno_func_user_ackn() ... 77
3.49 kernelv_ch_ackn_techno_func() ... 78
3.50 kernelv_ax_ackn_techno_func() ... 79
3.51 kernelv_get_license_info() ... 80
3.52 kernelv_set_options() ... 81
3.53 kernelv_ch_get_decoder_positions() ... 82
3.54 kernelv_ch_get_prog_start_mode() ... 84
3.55 kernelv_ch_set_cont_visu_grid() ... 85
3.56 kernelv_ch_set_cont_visu_rel_curvature_error() ... 86
3.57 kernelv_ch_set_cont_visu_abs_curvature_error() ... 87
3.58 kernelv_ch_get_cont_visu_data() ... 88
3.59 kernelv_ch_get_active_g_codes() ... 89
3.60 kernelv_get_active_g_group() ... 90

3.61 kernelv_ch_get_command_feed() ... 91
3.62 kernelv_ch_get_active_feed() ... 92
3.63 kernelv_set_call_ratio() ... 93
3.64 CNC Fehlermeldungen mit kernelv ... 94
3.64.1 Fehlermeldung als Zeichenkette auslesen kernelv_get_error() ... 94
3.64.2 Allgemeine Informationen zu Fehlermeldungen ... 95
3.64.3 Fehlermeldungen die durch NC-Programme verursacht werden ... 104
3.65 Koordinatensysteme und Versätze ... 109
3.65.1 kernelv_ch_get_cs_name() ... 110
3.65.2 kernelv_ch_get_cs_rot_matrix() ... 111
3.65.3 kernelv_ch_get_cs_shift_vector() ... 112
3.65.4 kernelv_ch_get_cs_count() ... 113
3.65.5 kernelv_ch_get_active_cs_index() ... 114
3.65.6 kernelv_ch_axis_get_offsets() ... 114
3.65.7 kernelv_ch_get_total_cs_rot_matrix() ... 115
3.65.8 kernelv_ch_get_total_cs_offset ... 116
3.65.9 kernelv_ch_get_total_cs_def() ... 118
3.65.10 kernelv_ch_get_coord_sys_active() ... 119
3.66 Kinematische Transformationen ... 119
3.66.1 kernelv_ch_get_kin_trafo_active() ... 119
3.66.2 kernelv_ch_get_active_kin_id() ... 120
3.67 Externe Messhardware ... 120
3.67.1 kernelv_ax_get_ext_latch_command() ... 121
3.67.2 kernelv_ax_acknowledge_ext_latch_command() ... 122
3.67.3 kernelv_ax_set_ext_latch_event_pos() ... 122
3.67.4 kernelv_ax_set_ext_latch_event() ... 123
3.68 kernelv_ch_get_timer() ... 124
3.69 kernelv_get_production_time() ... 124
3.70 kernelv_diagnosis_upload() ... 125

4 kernelv API Typen ... 126
4.1 Enum KERNELV_RETURN ... 126
4.2 KERNELV_CHANNEL_STATE ... 128
4.3 Enum E_KERNELV_TECHNO_TYPE ... 130
4.4 Struct KERNELV_TECHNO_DATA ... 130
4.5 KERNELV_CHANNEL_TECHNO_DATA_ARRAY ... 131
4.6 KERNELV_CHANNEL_TECHNO_DATA_ARRAY2 ... 132
4.7 KERNELV_AXIS_TECHNO_DATA_ARRAY ... 133
4.8 Struct KERNELV_TECHNO_DATA2 ... 133
4.9 KERNELV_CHANNEL_TECHNO_DATA_ARRAY2 ... 134
4.10 KERNELV_AXIS_TECHNO_DATA_ARRAY2 ... 135
4.11 Union U_KERNELV_TECHNO_PARAM ... 135
4.12 Union U_KERNELV_TECHNO_PARAM2 ... 136
4.13 Struct M_H_CODE_DATA ... 136
4.14 Struct M_H_CODE_DATA2 ... 136
4.15 Enum E_KERNELV_SPINDLE_TYPE ... 136
4.16 Struct S_CODE_DATA ... 137

4.17 Struct T_CODE_DATA ... 138
4.18 Struct KERNELV_NC_LINE_DATA ... 138
4.19 Enum E_KERNELV_VAR_TYPE ... 139
4.20 Union U_KERNELV_VAR_VALUE ... 140
4.21 Struct KERNELV_VARIABLE ... 141
4.22 Struct KERNELV_NC_LINE_DATA ... 141
4.23 Struct KERNELV_LICENSE_INFO ... 142
4.24 Struct KERNELV_DECODER_POSITION_HEADER ... 142
4.25 Struct KERNELV_DECODER_POSITION_DATA ... 143
4.26 Enum E_KERNELV_PROG_START_MODE ... 143
4.27 Struct ACTIVE_G_CODES ... 143
4.28 Enum E_KERNELV_G_GROUP_TYPE ... 144
4.29 Datentypen der Konturvisualisierung ... 146
4.29.1 Struct CONTOUR_VISU ... 146
4.29.2 Union CONTOUR_VISU_DATA ... 147
4.29.3 Struct CONTOUR_VISU_DATA_V0 ... 147
4.29.4 Struct CONTOUR_VISU_DATA_V1 ... 148
4.29.5 Struct CONTOUR_VISU_DATA_V2 ... 148
4.29.6 Struct CONTOUR_VISU_DATA_V3 ... 148
4.29.7 Struct CONTOUR_VISU_DATA_V4 ... 149
4.29.8 Struct CONTOUR_VISU_DATA_V5 ... 149
4.29.9 Struct CONTOUR_VISU_DATA_V6 ... 149
4.29.10 Struct CONTOUR_VISU_DATA_V7 ... 150
4.29.11 Struct CONTOUR_VISU_DATA_V8 ... 150
4.29.12 Struct CONTOUR_VISU_DATA_V9 ... 150
4.29.13 Struct CONTOUR_VISU_DATA_V10 ... 151
4.29.14 Struct CONTOUR_VISU_DATA_V11 ... 151
4.29.15 Struct CONTOUR_VISU_CH_DATA ... 152
4.29.16 Struct CONTOUR_VISU_CH_DATA_V1 ... 153
4.29.17 Struct CONTOUR_VISU_CH_DATA_V2 ... 154
4.29.18 Struct CONTOUR_AXIS_DATA ... 154
4.29.19 Struct CONTOUR_AXIS_DATA_V1 ... 155
4.29.20 Struct CONTOUR_AXIS_DATA_V2 ... 155
4.29.21 Enum E_CONTOUR_TECHNO_TYPE ... 155
4.29.22 Struct CONTOUR_M_H_PROCESS ... 156
4.29.23 Struct CONTOUR_M_H_PROCESS_V1 ... 156
4.29.24 Enum E_CONTOUR_S_CMD ... 156
4.29.25 Struct CONTOUR_S_PROCESS ... 157
4.29.26 Struct CONTOUR_TOOL_PROCESS ... 157
4.29.27 Struct CONTOUR_DATA_TECHNO ... 157
4.29.28 Struct CONTOUR_DATA_TECHNO_V1 ... 158
4.30 Datentypen der Fehlerausgabe ... 159
4.30.1 Struct KERNELV_ERROR_VALUE ... 159
4.30.2 KERNELV_ERROR_VALUE_ARRAY ... 159
4.30.3 Enum E_KERNELV_ERR_VAL_TYPE ... 160
4.30.4 Enum E_KERNELV_ERR_VAL_DIMENSION ... 161

4.30.5 Enum E_KERNELV_ERR_VAL_MEANING ... 162
4.31 Enum KERNELV_AXIS_OFFSET_TYPES ... 163
4.32 Externe Messhardware ... 164
4.32.1 Struct KERNELV_EXT_LATCH_COMMAND_DATA ... 164
4.32.2 Enum E_KERNELV_EXT_LATCH_ORDER ... 164
4.32.3 E_KERNELV_MEAS_ACTIVE_EDGE ... 164
4.33 Fertigungszeitberechnung ... 165

5 kernelv API Konstanten ... 166
5.1 KERNELV_VAR_STRING_LEN ... 166
5.2 KERNELV_FILE_NAME_LENGTH ... 166
5.3 KERNELV_VAR_NAME_LENGTH ... 166
5.4 KERNELV_OPTION_LICENSE_CHECK_VERBOSE ... 166
5.5 CONTOUR_MAX_DATA_V0 ... 167
5.6 CONTOUR_MAX_DATA_V1 ... 167
5.7 CONTOUR_MAX_DATA_V2 ... 167
5.8 CONTOUR_MAX_DATA_V3 ... 167
5.9 CONTOUR_MAX_DATA_V4 ... 167
5.10 CONTOUR_MAX_DATA_V5 ... 168
5.11 CONTOUR_MAX_DATA_V6 ... 168
5.12 CONTOUR_MAX_DATA_V7 ... 168
5.13 CONTOUR_MAX_DATA_V8 ... 168
5.14 CONTOUR_MAX_DATA_V9 ... 168
5.15 CONTOUR_MAX_DATA_V10 ... 169
5.16 CONTOUR_MAX_DATA_V11 ... 169
5.17 CONTOUR_MAX_M_H_DATA ... 169
5.18 CONTOUR_MAX_SPDL_DATA ... 169
5.19 CONTOUR_AXIS_PER_CHANNEL ... 169
5.20 KERNELV_ERROR_VALUE_COUNT ... 170
5.21 KERNELV_ERR_MSG_ STRING_LENGTH ... 170
5.22 KERNELV_CHANNEL_TECHNO_DATA_COUNT ... 170
5.23 KERNELV_AXIS_TECHNO_DATA_COUNT ... 170
5.24 KERNELV_ERROR_VALUE_COUNT ... 170
5.25 KERNELV_INSTANCE_PREFIX_MAX_LEN ... 171

6 Support und Service ... 172

Stichwortverzeichnis ... 173

Abbildungsverzeichnis

Abb. 1 Auswählen einer TwinCAT 3 Testlizenz ... 16
Abb. 2 Aktivieren einer TwinCat 3 Testlizenz ... 16
Abb. 3 Startbildschirm der Demoapplikation ... 22
Abb. 4 Anzeige der Achspositionen in der Demoapplikation ... 23
Abb. 5 Schematischer Ablauf einer Messfahrt ... 121
Abb. 6 Zustandsdiagramm eines CNC-Kanals ... 129

1 CNC-Kern als DLL kernelv - Einleitung

Der ISG CNC-Kern wird für eine Standard-CNC in eine Echtzeitumgebung eingebunden, um ein echtzeitfähiges, deterministisches Verhalten zu erreichen.

Für viele Anwendungen im Umfeld einer CNC-Steuerung ist jedoch diese Echtzeitumgebung nicht erforderlich oder ggf. auch nicht gewünscht, z.B.:

- eine Fertigungszeitberechnung,
- die Vorab-Kollisionskontrolle oder
- eine Visualisierung.

Durch kernelv, die CNC-Simulations-DLL, soll dem Anwender eine Möglichkeit gegeben werden, eine virtuelle CNC innerhalb einer eigenen Nichtechtzeitanwendung verwenden zu können.

Bezüglich der vorhandenen Funktionalitäten bietet die Simulations-DLL weitgehend dieselben Möglichkeiten wie der Echtzeitkernel.

Bekannte Einschränkungen:

- Zugriffe auf Hardware werden nicht durchgeführt.
- Die Achsen werden von der CNC als Simulationsachsen behandelt.
- Eine SPS kann nicht verwendet werden, die API-Schnittstellen verwenden das HLI zur Beauftragung des CNC-Kernels.
- Keine Echtzeit erforderlich bzw. möglich.

Obligatorischer Hinweis zu Verweisen auf andere Dokumente

Zwecks Übersichtlichkeit wird eine verkürzte Darstellung der Verweise (Links) auf andere Dokumente bzw. Parameter gewählt, z.B. [PROG] für Programmieranleitung oder P-AXIS-00001 für einen Achsparameter.

Technisch bedingt funktionieren diese Verweise nur in der Online-Hilfe (HTML5, CHM), allerdings nicht in PDF-Dateien, da PDF keine dokumentenübergreifenden Verlinkungen unterstützt.

2 Beschreibung

2.1 DLL-Version

Die vorliegende Beschreibung bezieht sich auf die API-Version der Simulations-DLL, welche im Dokumententitel genannt wird.

Der Versionsstring der DLL kann mittels der Funktion kernelv_get_api_version() abgefragt werden.

2.2 Komponenten

Die Simulations-DLL besteht aus 3 Komponenten:

1. der Simulations-DLL kernelv_mt.dll
2. der Header-Datei kernelv.h
3. der Lib-Datei kernelv_mt.lib zum impliziten Binden.

Außerdem ist zur Benutzung der DLL eine gültige Lizenz nötig.

Diagnose

Für Diagnosezwecke ist das Programm "ahmi.exe" enthalten.

Mithilfe dieses Programms können interne Diagnosedaten der kernelv-DLL abgerufen und in eine Textdatei abgespeichert werden.

Das Programm ahmi.exe ist ausschließlich für Diagnosezwecke vorgesehen und sollte nicht als Bedienoberfläche für die kernelv-DLL verwendet werden.

Zusätzlich können die interne Diagnosedaten mithilfe der Funktion kernelv_diagnosis_upload() [▶ 125] auch in eine Datei abgespeichert werden.

2.3 Abhängigkeiten

Bei der Verwendung der kernelv-Dll bestehen folgende Abhängigkeiten von Komponenten anderer Hersteller.

2.3.1 kernelv ab V300

Zur Verwendung der kernelv-Dll müssen die folgenden zusätzlichen Dll's auf dem Applikationsrechner verfügbar sein:

- TcAdsDll.DLL , abhängig von der verwendeten Version der kernelv-Dll (32- oder 64 Bit), jeweils entsprechende Version dieser Dll.
- VCRUNTIME140.DLL

Die TcAdsDll.DLL wird automatisch bei der Installation von TwinCAT installiert.

Wenn VCRUNTIME140.DLL fehlt deutet dies darauf hin, dass „Visual C++ Re-distributable für Visual Studio 2015/2017/2019“ installiert werden sollte.

Diese Pakete können kostenlos bei Microsoft© heruntergeladen werden.

Hierbei ist zu beachten, dass die korrekte Version (32- oder 64-Bit) installiert wird.

2.4 Mehrfachinstanzierung

Pro Applikation kann nur eine kernelv-Instanz gestartet werden. Es ist jedoch möglich, auf einem PC mehrere Applikationen mit jeweils einer kernelv-Instanz laufen zu lassen.

Falls versucht wird, innerhalb einer Applikation 2 Instanzen von kernelv zu starten, z.B. durch 2 Aufrufe von kernelv_startup() oder kernelv_startup_instance(), wird der Startversuch der 2. Instanz verhindert und die Fehlerkennung ERR_DOUBLE_INSTANCE zurückgegeben

Um mehrere kernelv-Instanzen zu starten, muss die Funktion kernelv_startup_instance() verwendet werden. Hierbei muss die übergebene Instanzkennung für jede zu startende kernelv-Instanz computerweit eindeutig sein.

Falls versucht wird, 2 kernelv-Instanzen mit derselben Instanzkennung zu starten, wird beim Startversuch der 2. Instanz die Fehlerkennung ERR_DOUBLE_KERNEL zurückgegeben.

2.4.1 Diagnose bei Mehrfachinstanzierung

Auch bei mehreren Instanzen der kernelv-DLL kann das Programm ahmi.exe zur Diagnose verwendet werden.

Hierbei muss dann beim Start des Programms ahmi.exe angegeben werden, mit welcher Instanz der kernelv-DLL verbunden werden soll.

Hierzu gibt es 2 Möglichkeiten:

1. Angabe der Instanzkennung als Kommandozeilenparameter beim Programmstart. Es muss dabei dieselbe Instanzkennung verwendet werden, die beim Start der kernelv-Instanz mittels kernelv_startup_instance() [▶ 30] verwendet wurde. Die kernelv-Instanz wird dabei durch den Parameter **–instance_prefix** angegeben.
   Beispiel: ahmi.exe -instance_prefix 1_
2. Durch den Kommandozeilenparameter **–query_instance_prefix** wird erreicht, dass beim Start des Programms die Instanzkennung der kernelv-Instanz angegeben werden kann.
   Beispiel: ahmi.exe -query_instance_prefix
   Die Funktion kernelv_diagnosis_upload() [▶ 125] startet den Upload der Diagnosedaten für die Instanz, in welcher der Aufruf der Funktion stattfindet.

2.5 Lizenzschutz

Die DLL und sämtliche damit verbundene Funktionen stehen nur in TwinCAT-Systemen zur Verfügung.

Zur Benutzung der DLL ist eine gültige Lizenz nötig.

Ab DLL-Version V1.0.22.1 ist zur Nutzung der DLL mindestens eine Basislizenz erforderlich.

Die Nutzung nur über ein Optionspaket ist nicht möglich.

Die Lizenzprüfung erfolgt beim Start des Simulationskerns in der Funktion kernelv_startup(). Falls keine gültige Lizenz gefunden wird, wird der Start des Simulationskerns abgebrochen und der Fehlercode ERR_NO_LICENSE zurückgegeben.

Bei Fehlern beim Zugriff auf die Lizenzinformationen wird der Fehlercode ERR_REGISTRY_ACCESS zurückgegeben. Dies ist z.B. der Fall, wenn

- unter TwinCAT 2 die Installation des VNCK nicht korrekt durchgeführt wurde oder
- unter TwinCAT 3 keine Testlizenz erzeugt wurde.

Zusätzlich sind zur Verwendung bestimmter Funktionen, wie z.B. Transformationen oder Achsanzahlen > 8, ebenfalls Lizenzpakete nötig. Das Vorhandensein dieser Lizenzpakete wird während der Laufzeit von kernelv geprüft und gegebenenfalls eine CNC-Fehlermeldung ausgegeben.

2.5.1 Lizenzierung unter TwinCAT 2

Unter TwinCAT 2 werden die Lizenzinformationen bei der Installation der VNCK-Installation in die Windows-Registry geschrieben und werden dort beim Start der kernelv-DLL abgefragt.

Nach einer erfolgten Installation sind keine weiteren Aktionen des Anwenders notwendig.

2.5.2 Lizenzierung unter TwinCAT 3

TwinCAT 3 Testlizenz

Wie bei allen anderen TwinCAT 3 Softwaremodulen kann für die kernelv-DLL eine 7-Tage-Testlizenz erzeugt werden. Hierzu ist es allerdings notwendig, dass eine TwinCAT 3 Installation auf dem Rechner vorhanden ist.

Um die Testlizenz zu erzeugen, sind folgende Aktionen nötig:

1. Start TwinCAT XAE
2. Ein neues, leeres TwinCAT-Projekt erzeugen.
3. In der Baumansicht in der linken Seite von TwinCAT XAE den Knoten System\License auswählen.
4. In den nun erscheinenden Karteikarten die Karte 'Manage Licenses' auswählen und in der Liste die Optionen 'TC3 CNC Virtual NCK Basis' (TF5270) und eventuell 'TC3 CNC Virtual NCK Options' auswählen.
5. Zum Aktivieren der Testlizenz auf der Karteikarte 'Order Information (Runtime)' den Button '7 Days Trial License' betätigen und den angeforderten Security Code eingeben.

TwinCAT 3 Dauerlizenz

Das Anfordern einer Dauerlizenz erfolgt entsprechend der bei TwinCAT 3 üblichen Vorgehensweise.

Verwendung der kernelv-DLL mit Lizenzprüfung

Beim Start des CNC-Kernels durch die Funktion kernelv_startup() werden die benötigten Lizenzinformationen beim TwinCAT 3 Lizenzserver abgefragt. Die Kommunikation zwischen CNC-kernel und dem Lizenzserver erfolgt per ADS. Es ist daher nötig, dass die Applikation, die die kernelv-DLL verwendet, Zugriff auf die Bibliothek TcAdsDll.dll hat. Diese Bibliothek ist in der TwinCAT-Installation enthalten.

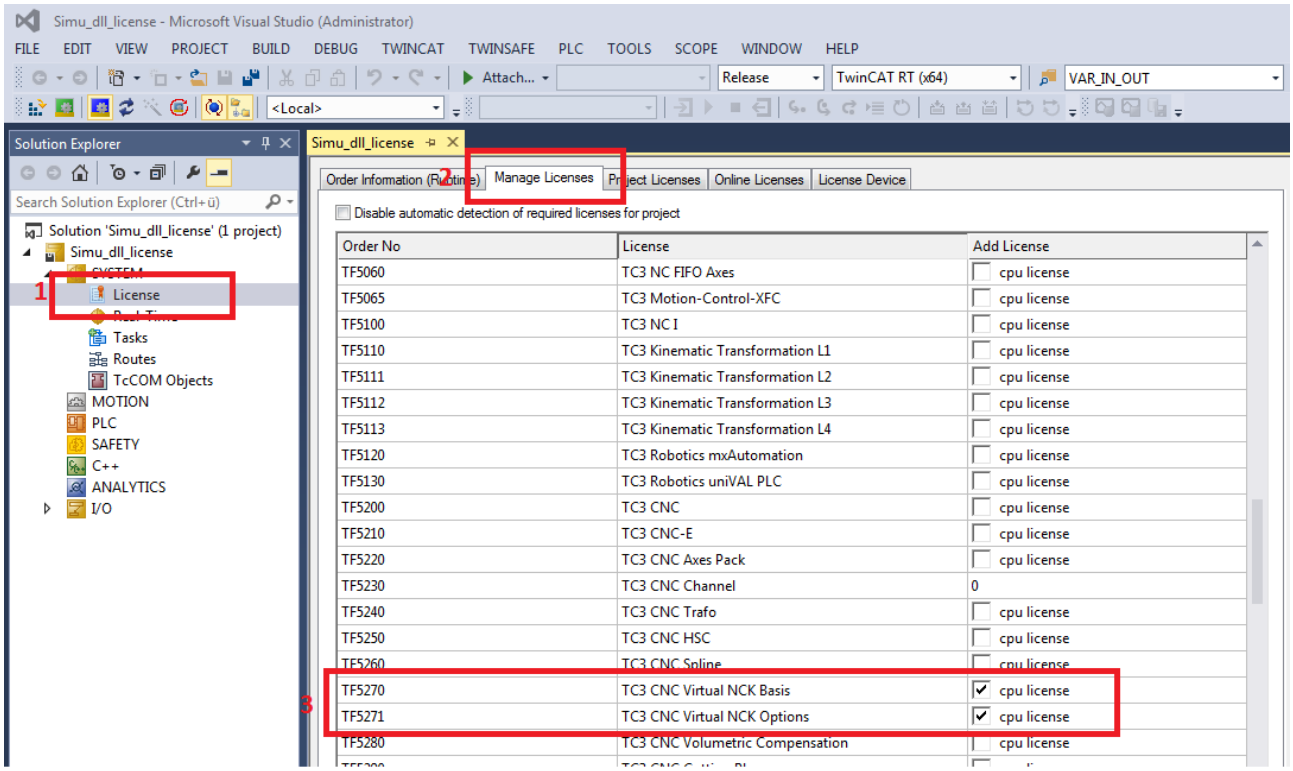


Abb. 1: Auswählen einer TwinCAT 3 Testlizenz

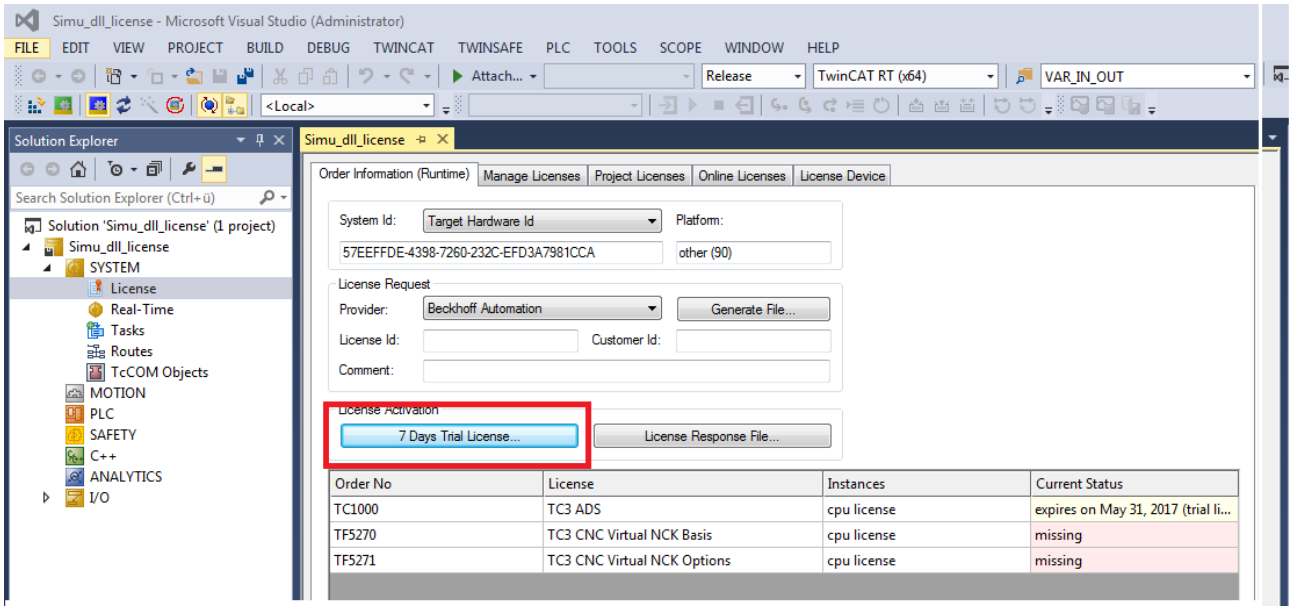


Abb. 2: Aktivieren einer TwinCat 3 Testlizenz

2.6 Allgemeines

2.6.1 Identifikation von Achsen und Kanälen

Kanalidentifikation

Kanäle werden durch ihren Index auf der SPS-Schnittstelle identifiziert. Die Reihenfolge der Kanäle entspricht der Konfigurationsreihenfolge.

Der erste konfigurierte Kanal wird mit dem Index 0 angesprochen, der letzte konfigurierte Kanal von n Kanälen mit dem Index n – 1.

Wird einer Funktion ein ungültiger Kanalindex übergeben, so wird der Rückgabewert ERR_INVALID_CHAN (definiert in kernelv.h) zurückgegeben.

Falls von Funktionen Kanalnummern als Rückgabewert geliefert werden, kann durch die Beziehung

Kanalindex = Kanalnummer - 1

der zugehörige Kanalindex ermittelt werden.

Achsidentifikation

Achsen werden durch ihren Index auf der SPS-Schnittstelle identifiziert. Die Reihenfolge der Achsen entspricht der Konfigurationsreihenfolge.

Die erste konfigurierte Achse wird mit dem Index 0 angesprochen, die letzte konfigurierte Achse von m Achsen mit dem Index m – 1.

Wird einer Funktion ein ungültiger Achsindex übergeben, so wird der Rückgabewert ERR_INVALID_AX (definiert in kernelv.h) zurückgegeben.

2.6.2 Koordinatensysteme

ACS-Koordinaten

Unter ACS-Koordinaten (ACS = **a**xis **c**oordinate **s**ystem) werden die Koordinaten der physikalischen Achsen verstanden.

Funktionsnamen von Funktionen, die ACS-Koordinaten zurückliefern, beginnen mit dem Präfix "kernelv_get_acs_".

WCS-Koordinaten

Unter WCS-Koordinaten (WCS = **w**orkpiece **c**oordinate **s**ystem) wird das Programmierkoordinatensystem verstanden.

Das Programmierkoordinatensystem kann gegenüber dem Achskoordinatensystem, z.B. durch Bezugspunktverschiebungen und die Definition von Bearbeitungskoordinatensystemen, verschoben und verdreht sein.

Funktionsnamen von Funktionen, die WCS-Koordinaten zurückliefern, beginnen mit dem Präfix "kernelv_get_wcs_".

Dimensionen

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1x10^{-4°}$ für Spindeln oder Moduloachsen.

2.7 Benutzung von kernelv

Die DLL kann von der Anwenderapplikation sowohl durch implizites als auch durch explizites (dynamisches) Binden verwendet werden.

Voraussetzungen bei TwinCAT 3

Bei kernelv-Versionen, die für eine TwinCAT 3 Umgebung erstellt wurden (Buildnummer der CNC-Version >= 3000), ist es nötig, dass die Applikation die die kernelv-DLL verwendet, Zugriff auf die TwinCAT-Bibliothek TcAdsDll.dll hat. Diese Bibliothek wird im Rahmen einer TwinCAT-Installation bereitgestellt.

2.7.1 Start

kernelv_startup()

Die Simulations-CNC wird gestartet, indem die API-Funktion kernelv_startup() aufgerufen wird. Aufrufparameter sind die Zykluszeit in Mikrosekunden sowie die Achs- und Kanalanzahl.

Wenn der Start der Simulations-CNC erfolgreich war, gibt die Funktion den Wert RET_FINISHED zurück, andernfalls einen Fehlercode.

Es kann immer nur eine Instanz der Simulations-CNC auf demselben Rechner laufen. Wird die Funktion kernelv_startup() aufgerufen während bereits eine Instanz der Simulations-CNC läuft, wird der Fehlercode ERR_DOUBLE_KERNEL zurückgegeben.

Des Weiteren ist zur Benutzung der Simulations-CNC eine gültige Lizenz nötig. Falls beim Start der Simulations-CNC keine gültige Lizenz gefunden wird, wird der Start der Simulations-CNC abgebrochen und der Fehlercode ERR_NO_LICENSE zurückgegeben.

2.7.2 Zyklischer Betrieb

kernelv_do_cycle()

Durch Aufruf der Funktion kernelv_do_cycle() wird ein Zyklus der Simulations-CNC für alle Kanäle durchgerechnet.

Als Zykluszeit wird die beim Start der Simulations-CNC übergebene Zykluszeit verwendet.

Ein Aufruf in Echtzeit ist nicht notwendig.

2.7.3 Beenden/Neustart

Zum Beenden der Simulations-CNC steht aktuell keine API-Funktion zur Verfügung.

Falls die Simulations-CNC neu gestartet werden soll, kann im Falle der dynamischen Bindung die DLL entladen und danach wieder neu geladen werden. Danach kann die Simulations-CNC erneut gestartet werden.

2.7.4 Betriebsarten von kernelv

Ein NC-Programm kann in 2 verschiedenen Kanalbetriebsarten gestartet werden. Abhängig von der Betriebsart stehen unterschiedliche Funktionalitäten der kernelv-DLL zur Verfügung. Die Betriebsart kann kanalspezifisch und ist beim Programmstart anzugeben.

Aktuell stehen die folgenden Betriebsarten zur Verfügung:

Name	E_KERNELV_PROG_START_MODE	Bedeutung
Normalbetriebsart	KERNELV_START_MODE_NORMAL	Standardbetriebsart, die Achsbewegungen werden in der korrekten Geschwindigkeit interpoliert, und die Bewegung der Physikalischen Achsen simuliert.
Sollkonturvisualisierung	KERNELV_START_MODE_CONTOUR_VISU	Betriebsart Sollkonturvisualisierung, es werden nur Interpolationsstützpunkte entsprechend dem eingestellten Visualisierungsraster berechnet.

Normalbetriebsart

In der Normalbetriebsart wird das NC-Programm mit der beim Start der kernelv-DLL angegebenen Zykluszeit abinterpoliert. Die erzeugten Interpolationsdaten entsprechen denen einer in Echtzeit laufenden Steuerung. Die für die Achsen eingestellten dynamischen Grenzwerte werden eingehalten.

Betriebsart Sollkonturvisualisierung

Die Betriebsart Sollkonturvisualisierung dient dem Erzeugen einer im Vergleich zur Normalbetriebsart gröberen Stützpunktfolge, zur schnellen Visualisierung der programmierten Kontur.

Das Ausgaberaster kann für gerade und gekrümmte Konturelemente mit den Funktionen
kernelv_ch_set_cont_visu_grid(),
kernelv_ch_set_cont_visu_rel_curvature_error() und
kernelv_ch_set_cont_visu_abs_curvature_error()
eingestellt werden. Unabhängig vom eingestellten Ausgaberaster werden die Endpunkte eines Bewegungssatzes immer ausgegeben.

Das gröbere Ausgaberaster wird erreicht, indem die Geschwindigkeit, mit der ein Konturelement abinterpoliert wird, so berechnet wird, dass sich das eingestellte Ausgaberaster ergibt. Die dynamischen Daten der Achsen werden deshalb nicht eingehalten, die interpolierten Sollwerte werden ebenfalls nicht an den Lageregler weitergegeben, sodass die ACS-Koordinaten sich in dieser Betriebsart nicht ändern.

Anwahl einer Betriebsart

Die Betriebsart ist beim Aufruf der Funktion kernelv_ch_program_start() anzugeben.

```
kernelv_ch_program_start (0,
                prog_name,
                KERNELV_START_MODE_NORMAL);
```

Ausgabedaten der Konturvisualisierung

Die in der Betriebsart Konturvisualisierung berechneten Stützpunkte werden in einen FIFO-Speicher geschrieben und sind mit der Funktion kernelv_ch_get_cont_visu_data() zu lesen.

Um zu verhindern, dass Visualisierungsdaten verloren gehen, wird die Interpolation angehalten, wenn der interne FIFO voll ist. Um eine möglichst schnelle Abarbeitung eines NC-Programms zu erreichen, muss also die Funktion kernelv_ch_get_cont_visu_data() ausreichend oft aufgerufen werden.

Ausgabeformat der Konturvisualisierung

Die bei aktiver Konturvisualisierung ausgegeben Daten können mittels des Parameters contour_visu_ifc_version (P-STUP-00039) in der Hochlaufliste systemweit eingestellt werden.

Abhängig von der eingestellten Interfaceversion ändert sich das Format der von der Funktion kernelv_ch_get_cont_visu_data() zurückgegebenen Daten. Siehe dazu auch Strukturdefinition CONTOUR_VISU.

Die interpolierten WCS-Koordinaten können mit den Funktionen kernelv_get_wcs_ gelesen werden.

2.7.5 Quittieren von Technologiefunktionen

Durch Technologiefunktionen (z. B. M- oder H-Funktionen) werden bei einer realen CNC-Steuerung Informationen zwischen dem NC-Programm und den umgebenden Softwarekomponenten (z. B. SPS) ausgetauscht.

Dies beinhaltet die eigentlichen Technologieinformationen, die von der CNC ausgegeben werden, sowie Quittierungen die von den umgebenden Softwarekomponenten an die CNC übergeben werden, um Programmabläufe innerhalb und außerhalb der CNC zu synchronisieren. Bei der Abarbeitung eines NC-Programmes können diese Synchronisierungen zu Verzögerungen im Ablauf des NC-Programmes führen.

Standardverhalten

Standardmäßig werden von der kernelV-DLL alle ausgegebenen Technologiefunktionen sofort automatisch quittiert.

Bearbeitungszeitsimulation

Falls die Bearbeitungszeit von Technologiefunktionen bei der Ausführung eines NC-Programmes berücksichtigt werden soll, kann mit der API-Funktion kernelv_control_techno_func_duration() [▶ 75] die Bearbeitungszeitsimulation für Technologiefuntkionen aktiviert werden.

Bei aktiver Bearbeitungszeitsimulation werden Technologiefunktionen nach einer einstellbaren Zeit automatisch quittiert.

Die Ausführungszeit kann durch die Funktion kernelv_ch_set_techno_func_duration() [▶ 76] oder durch Einträge in der Kanalparameterliste (P-CHAN-00040, P-CHAN-00026) eingestellt werden.

Anwenderquittierung

Durch Verwendung der Anwenderquittierung hat der Anwender selbst volle Kontrolle über den Quittierungszeitpunkt der Technologiefunktion.

Hierzu muss zunächst die Bearbeitungszeitsimulation durch Aufruf der Funktion kernelv_control_techno_func_duration() [▶ 75] aktiviert werden.

Technofunktionen, für die die Anwenderquittierung aktiviert werden soll sind mit der Funktion kernelv_ch_set_techno_func_user_ackn() [▶ 77] zu markieren. Alle anderen Technofunktionen werden nach Ablauf der jeweiligen Bearbeitungszeit quittiert.

Durch Aufruf der Funktionen kernelv_ch_get_new_techno_data() [▶ 44] bzw. kernelv_ax_get_new_techno_data() [▶ 49] ist zu prüfen, ob die jeweilige Technofunktion ausgegeben wurde.

Zum Quittierungszeitpunkt ist die Technofunktion durch Aufruf der Funktion kernelv_ch_ackn_techno_func() [▶ 78] bzw. kernelv_ax_ackn_techno_func() [▶ 79] zu quittieren.

2.7.6 Suchpfad für NC-Programme

NC-Programme werden relativ zum Startverzeichnis der Applikation gesucht. Zusätzlich können Suchpfade für Programme und globale Unterprogramme in der Hochlaufliste (siehe Dokumentation CNC-Kern) angegeben werden.

2.7.7 kernelv Demoapplikation

Als Beispiel für die Benutzung der kernelv-DLL ist eine Demoapplikation inklusive eines Demoparametersatzes verfügbar, bei der die kernelv-DLL explizit geladen wird. Der Anwender hat über eine einfache ASCII-Oberfläche die Möglichkeit NC-Programme zu starten und Achspositionen anzuzeigen.

Die Demoapplikation wird in Form eines Zip-Archives ausgeliefert. Die eigentliche kernelv-DLL wird in einer separaten Installation geliefert und ist nicht im Zip-Archiv enthalten.

2.7.7.1 Projektstruktur

Beim Entpacken der Demoapplikation wird die folgende Ordnerstruktur angelegt:

```
kernelv_demo
|
├----\listen
└----\prg
```

Im Verzeichnis kernelv_demo sind dabei die folgenden Dateien enthalten:

kernelv_demo.dsw, kernelv_demo.dsp	Projektfiles Demoapplikation Visual Studio 6
kernelv_demo.c	C-Quelltext der Demoapplikation
err_text_version.txt	Datei mit Fehlermeldungstexten
kernelv_demo.bat	Batch-Datei zum Starten der Demoapplikation (Debug-Version).

Im Unterverzeichnis \listen sind dabei die Konfigurationslisten des CNC-Kernels gespeichert. Das Unterverzeichnis \prg enthält ein einfaches Beispielprogramm.

2.7.7.2 Benutzung der Demoapplikation

Zur Benutzung der Demoapplikation sind die folgenden Schritte notwendig:

1. Demo Entpacken.
2. Kernelv-DLL und kernelv.h in das beim Entpacken erzeugte Verzeichnis kernelv_demo kopieren.
3. Projekt kompilieren und starten. Der Start kann entweder im Debugger des Visualstudios oder mittels der Datei kernelv_demo.bat erfolgen.

Startbildschirm

Nach dem Start ist folgender Bildschirm zu sehen:

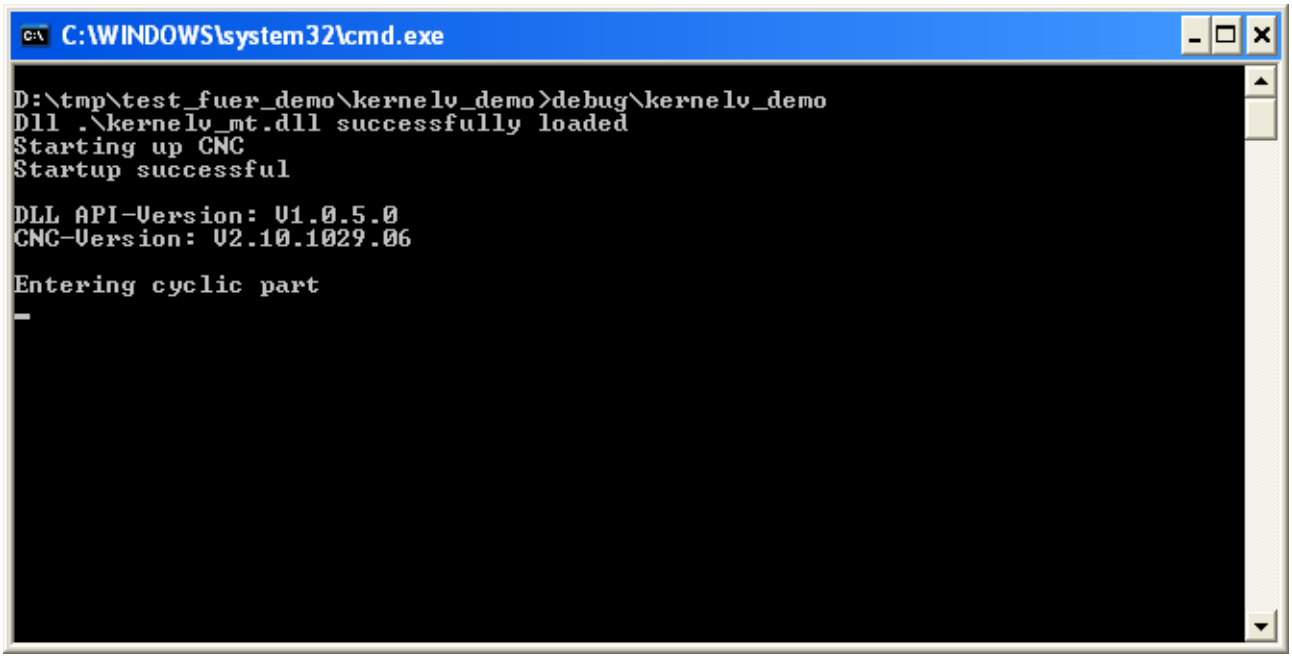


Abb. 3: Startbildschirm der Demoapplikation

Tastaturbefehle

Eingabe	Wirkung
S	Programm starten. Wird kein Programmname eingegeben, wird das NC-Programm 'test1.nc' versucht zu starten. Dieses wird im Verzeichnis /prg gesucht.
R	CNC-Reset ausführen.
P	Achspositionen auslesen und anzeigen.
Q	Applikation beenden.
'Enter'	Statusanzeige

Anzeige Achspositionen

Nach der Abarbeitung des mitgelieferten Testprogrammes und Anzeige der Achspositionen ergibt sich folgende Anzeige:

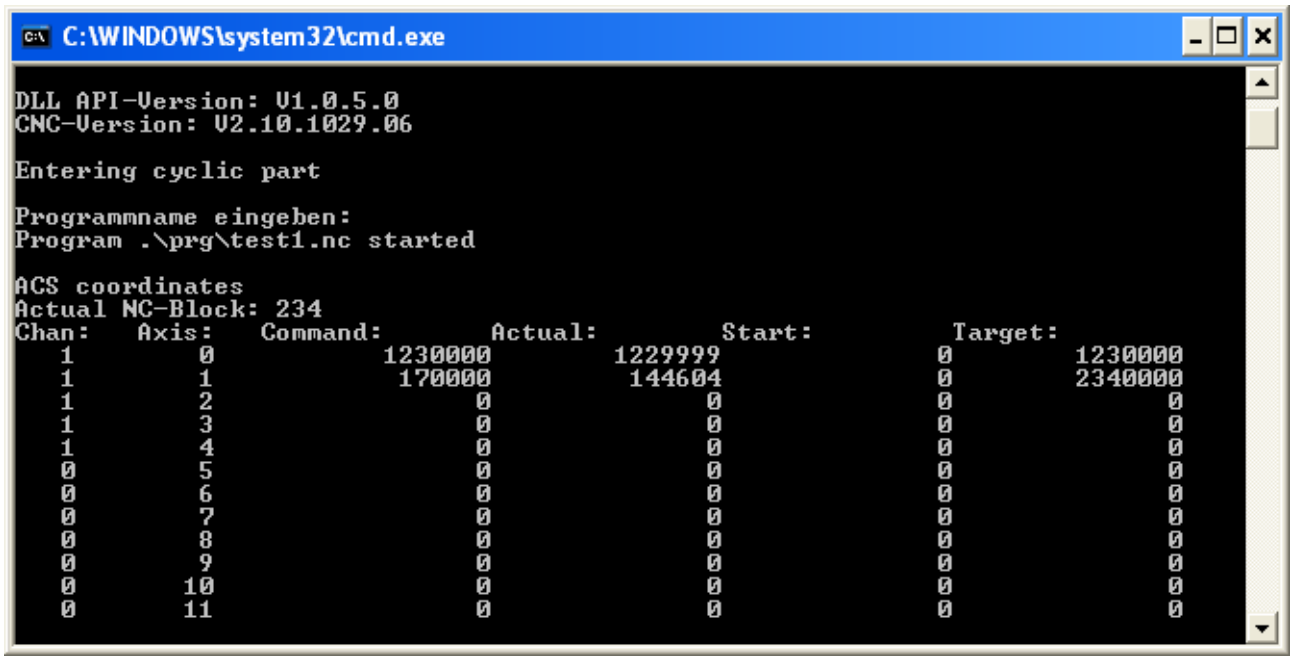


Abb. 4: Anzeige der Achspositionen in der Demoapplikation

2.7.7.3 Erläuterungen zur Demoapplikation

Im Quellcode der Demoapplikation werden sämtliche notwendigen Schritte zur Benutzung der kernelv-DLL durchgeführt. Weitergehende Informationen zu den aufgerufenen Betriebssystemfunktionen können dabei der jeweiligen Compilerdokumentation entnommen werden.

DLL Laden:

```
hDll = LoadLibrary(dll_name);
```

Funktionszeiger abfragen:

```
if ( !(pCycle = (KERNELV_CYCLE) GetProcAddress(hDll,
                                "kernelv_do_cycle"))  )
{
  printf("Missing pointer to kernelv_do_cycle()\n");
  return -2;
}
```

Falls das angefragte Symbol nicht in der DLL definiert ist, wird eine Meldung ausgegeben und die Applikation beendet. Einzelheiten zur Verwendung der einzelnen Funktionen kann dem Abschnitt ‚kernelv API dieses Dokuments entnommen werden.

Hauptschleife zur Behandlung von Benutzereingaben sowie zyklischer Aufruf von kernelv_do_cycle(). Hierbei wird ein CNC-Zyklus durchgerechnet:

```
(pCycle)();
```

2.8 Konfiguration

In der vorliegenden Version der Simulations-DLL wird der CNC-Kern durch ASCII-Listen konfiguriert. Diese ASCII-Listen können durch das Programm "Listenexporteur" aus einer TwinCAT-Konfigurationsdatei erzeugt werden.

2.8.1 Pfad der Konfigurationslisten

Startupfile

Das Startup-File enthält die grundlegende Konfiguration des CNC-Kerns wie z.B.:

- Anzahl der Achsen,
- Anzahl der Kanäle, sowie
- den Zugriffspfad auf die jeweiligen achs- oder kanalspezifischen Konfigurationsdateien.

Pfad und Dateiname des Startup-Files (Default: hochlauf.lis) müssen der API-Funktion kernelv_startup übergeben werden.

Innerhalb der Hochlaufliste sind die Zugriffspfade zu weiteren Parameterlisten anzugeben:

- Relative Pfade innerhalb der Hochlaufliste sind relativ zum Speicherort der Hochlaufliste anzugeben.
- Absolute Pfade werden unverändert übernommen.

Es wird empfohlen, die Parameterlisten entweder durch Exportieren der Listen im TwinCAT-Systemmanager zu erzeugen oder durch Verwendung des Tools ‚Listenexporteur'.

2.9 Fehlermeldungstexte

Zuordnung Fehlermeldungsnummer zu Fehlermeldungstext

Bei der Ausgabe einer Fehlermeldung wird vom CNC-Kern nur ein Fehlermeldungscode zusammen mit einigen Parametern ausgegeben. Die Zuordnung eines Fehlermeldungstextes erfolgt erst zu einem späteren Zeitpunkt.

Die Zuordnung von Fehlermeldungsnummer zu Fehlermeldungstext wird durch die Datei ‚err_text_version.txt' durchgeführt. Diese Datei muss sich im Arbeitsverzeichnis der Applikation, die die Simulations-DLL verwendet, befinden.

2.10 Werkzeugverwaltung

Interne/Externe Werkzeugvewaltung

Bei der Verwendung des CNC-Kerns als Simulations-DLL kann sowohl die interne als auch die externe Werkzeugverwaltung (Umschaltung durch P-CHAN-00016) verwendet werden.

Werkzeugdaten

Bei der internen Werkzeugverwaltung sind die Werkzeugdaten kanalspezifisch in der Werkzeugdatenliste zu hinterlegen.

Bei Verwendung der externen Werkzeugverwaltung werden die Werkzeugdaten stets global für die gesamte CNC verwaltet. In diesem Fall sind die Werkzeugdaten in die Werkzeugdatenliste des 1. Kanals einzutragen.

Der Zugriffspfad auf die Werkzeugdatenliste ist in der Hochlaufliste anzugeben.

Das Format der Werkzeugdatenliste ist in [TOOL] beschrieben.

3 kernelv API Funktionen

3.1 kernelv_get_api_version()

Prototyp

```
KERNELV_RETURN    kernelv_get_api_version (char* versionString,
                                           unsigned long maxStringLength,
                                           unsigned long* returnSize);
```

Beschreibung

Versionsstring der API lesen.

Parameter

Name	Typ	Bedeutung
versionString	char*	Zeiger auf den Speicherort des Versionsstrings, der Speicher ist von der Applikation bereitzustellen.
maxStringLength	unsigned long	Länge des von der Applikation bereitgestellten Speichers (in Bytes).
returnSize	unsigned long*	Länge des zurückgelieferten Versionsstrings. Falls ein Fehler aufgetreten ist, wird der Wert 0 zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes inklusive der terminierenden Null zurückgegeben.

3.2 kernelv_get_cnc_version()

Prototyp

```
KERNELV_RETURN  kernelv_get_cnc_version(char* versionString,
                                        unsigned long maxStringLength,
                                        unsigned long* returnSize);
```

Beschreibung

Versionsstring der CNC lesen.

Parameter

Name	Typ	Bedeutung
versionString	char*	Zeiger auf den Speicherort des Versionsstrings, der Speicher ist von der Applikation bereitzustellen.
maxStringLength	unsigned long	Länge des von der Applikation bereitgestellten Speichers (in Bytes).
returnSize	unsigned long*	Länge des zurückgelieferten Versionsstrings. Falls ein Fehler aufgetreten ist, wird der Wert 0 zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes inklusive der terminierenden Null zurückgegeben.

3.3 kernelv_get_cycletime()

Prototyp

```
KERNELV_RETUR kernelv_get_cycletime (unsigned long* cycleTime);
N
```

Beschreibung

Zykluszeit der CNC auslesen.

Parameter

Name	Typ	Bedeutung
cycleTime	unsigned long*	Zykluszeit in us.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.4 kernelv_startup()

Prototyp

```
KERNELV_RETURN  kernelv_startup( unsigned long cycleTime,
                                 char* startupFile);
```

Beschreibung

Simulations-CNC starten.

Parameter

Name	Typ	Bedeutung
cycleTime	unsigned long	Zykluszeit in us..
startupFile	char *	Pfad und Name des Startup-Files.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_DOUBLE_KERNEL	-6	Es läuft bereits eine Instanz der Simulations-CNC.
ERR_SHM_STARTUP	-7	Beim Start des CNC-Kerns konnten intern verwendete Shared memories nicht angelegt werden.
ERR_STARTUP	-8	Beim Start der Simulations-CNC ist ein Fehler aufgetreten. Mögliche Ursachen sind fehlende Parameterlisten oder fehlerhafte Einträge in Parameterlisten.
ERR_NO_LICENSE	-17	Es wurde keine Lizenz für die Verwendung der kernelv-DLL gefunden.
ERR_REGISTRY_ACCESS	-19	Beim Versuch Werte aus der Windows-Registry zu lesen ist ein Fehler aufgetreten.
ERR_PREFIX_TOO_LONG	-23	Beim Aufruf der Funktion kernelv_startu_prefix() ist die übergebene Instanzkennung zu lang, sodass die intern generierten Namen für die verwendeten Shared Memories nicht mehr in den dafür vorgesehenen Speicher passen.
ERR_STARTUP_CHAN_INIT	-31	Beim Start der kernelev-Dll konnte die Initialisierung der konfigurierten NC-Kanäle nicht durchgeführt werden.

Abhängig von der Achs- und Kanalanzahl der verwendeten Konfiguration kann der Start des CNC-Kerns 20 - 30 Sekunden dauern.

3.5 kernelv_startup_instance()

Prototyp

```
KERNELV_RETUR  kernelv_startup_instance (unsigned long cycleTime,
N                                        char* startupFile
                                         char* instancePrefix);
```

Beschreibung

Simulations-CNC starten:

Mit dieser Funktion ist es möglich, auf einem Computer mehrere Applikationen laufen zu lassen. Jede nutzt eine einzelne und damit eigene Instanz der kernelv-DLL. Hierzu muss im Aufrufparameter instancePrefix eine computerweit eindeutige Instanzkennung übergeben werden.

Die maximale Länge der Zeichenkette, die als Instanzkennung übergeben werden darf ist durch die Konstante KERNELV_INSTANCE_PREFIX_MAX_LEN definiert. Falls eine längere Zeichenkette übergeben wird, wird der Hochlauf nicht durchgeführt und die Funktion gibt den Wert ERR_PREFIX_TOO_LONG (-23) zurück.

Es ist nicht möglich, innerhalb einer Applikation mehrere Instanzen der kernelv-DLL laufen zu lassen.

Parameter

Name	Typ	Bedeutung
cycleTime	unsigned long	Zykluszeit in us..
startupFile	char *	Pfad und Name des Startup-Files.
instancePrefix	char *	Eindeutige Instanzkennung.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_DOUBLE_KERNEL	-6	Es läuft bereits eine Instanz der Simulations-CNC.
ERR_SHM_STARTUP	-7	Beim Start des CNC-Kerns konnten intern verwendete Shared memories nicht angelegt werden.
ERR_STARTUP	-8	Beim Start der Simulations-CNC ist ein Fehler aufgetreten. Mögliche Ursachen sind fehlende Parameterlisten oder fehlerhafte Einträge in Parameterlisten.
ERR_NO_LICENSE	-17	Es wurde keine Lizenz für die Verwendung der kernelv-DLL gefunden.
ERR_REGISTRY_ACCESS	-19	Beim Versuch Werte aus der Windows-Registry zu lesen ist ein Fehler aufgetreten.
ERR_PREFIX_TOO_LONG	-23	Beim Aufruf der Funktion kernelv_startu_prefix() ist die übergebene Instanzkennung zu lang, sodass die intern generierten Namen für die verwendeten Shared Memories nicht mehr in den dafür vorgesehenen Speicher passen.
ERR_STARTUP_CHAN_INIT	-31	Beim Start der kernelev-Dll konnte die Initialisierung der konfigurierten NC-Kanäle nicht durchgeführt werden.

Abhängig von der Achs- und Kanalanzahl der verwendeten Konfiguration kann der Start des CNC-Kerns 20 - 30 Sekunden dauern.

3.6 kernelv_do_cycle()

Prototyp

```
KERNELV_RETUR  kernelv_do_cycle();
N
```

Beschreibung

Einen Zyklus der Simulations-CNC durchrechnen.
Für die internen Berechnungen wird die beim Start als Parameter übergebene Zykluszeit verwendet. Sollte in der Simulations-CNC ein Fehler auftreten, so kann dies mit der Funktion kernelv_get_error () geprüft und der Fehlermeldungsstring ausgelesen werden.

Parameter

Keine

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_NO_LICENSE	-17	Es wurde keine Lizenz für die Verwendung der kernelv-DLL gefunden.

3.7 kernelv_ch_program_start()

Prototyp

```
KERNELV_RETURN    kernelv_ch_program_start (unsigned long chanIndex,
                                            char* name,
                                            unsigned long mode);
```

Beschreibung

Im angegebenen Kanal ein NC-Programm starten.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals in dem das Programm gestartet werden soll.
name	char*	Name des zu startenden Programms.
mode	unsigned long*	Bearbeitungmodus, in dem das Programm gestartet wird. Mögliche Bearbeitungsmodi, siehe E_KERNELV_PROG:START_MODE.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_PROG_NAME_LENGTH	-2	Der übergebene Programmname ist länger als zulässig.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INVALID_START_MODE	-25	Beim Aufruf der Funktion kernelv_ch_program_start() wurde ein ungültiger Startmode als Parameter übergeben. Gültige Werte für den Bearbeitungsmodus, siehe E_KERNELV_PROG_START_MODE.

3.8 kernelv_ch_reset()

Prototyp

```
KERNELV_RETURN  kernelv_ch_reset (unsigned long chanIndex);
```

Beschreibung

In dem angegebenen Kanal einen CNC-Reset ausführen.

Durch den Reset werden CNC-interne Fehler zurückgesetzt. Ein eventuell während des Resets laufendes Programm wird abgebrochen.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals in dem das Programm gestartet werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.9 kernelv_ch_suspend()

Prototyp

```
KERNELV_RETURN          kernelv_ch_suspend (unsigned long int chanIndex);
```

Beschreibung

Das aktuell im Kanal laufende Programm wird angehalten. Der Kanalzustand wechselt nach SIMU_CNC_STATE_HOLD.

Das Anhalten eines Programms ist nur möglich, wenn im Kanal gerade ein NC-Programm abgearbeitet wird, also wenn sich der Kanal im Zustand SIMU_CNC_STATE_ACTIVE befindet. Wird die Funktion aufgerufen, während sich der Kanal in einem anderen Zustand befindet, gibt die Funktion den Wert ERR_INVALID_STATE zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals der angehalten werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INVALID_STATE	-5	Der CNC-Kanal ist im falschen Zustand, um eine Funktion auszuführen.

3.10 kernelv_ch_resume()

Prototyp

```
KERNELV_RETURN kernelv_ch_resume (unsigned long int chanIndex);
```

Beschreibung

Das aktuell im Kanal laufende Programm wird fortgesetzt. Der Kanalzustand wechselt nach SIMU_CNC_STATE_ACTIVE.

Das Fortsetzen eines Programms ist nur möglich, wenn sich der Kanal im Zustand SIMU_CNC_STATE_HOLD befindet. Wird die Funktion aufgerufen, wenn sich der Kanal in einem anderen Zustand befindet, gibt die Funktion den Wert ERR_INVALID_STATE zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals, der fortgesetzt werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INVALID_STATE	-5	Der CNC-Kanal oder die Achse ist im falschen Zustand, um eine Funktion auszuführen.

3.11 kernelv_ch_get_override()

Prototyp

```
KERNELV_RETURN          kernelv_ch_get_override (unsigned long int chanIndex,
                                                 unsigned short int *override);
```

Beschreibung

Lesen des aktuellen Override-Wertes für den Kanal.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals, dessen Override gelesen werden soll..
override	unsigned short*	Zeiger auf den Wert, in den der Override geschrieben werden soll. Zurückgegeben wird der aktuelle Overridewert in 0,1%.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.12 kernelv_ch_set_override()

Prototyp

```
KERNELV_RETURN  kernelv_ch_set_override (unsigned long int chanIndex,
                                         unsigned short int override);
```

Beschreibung

Setzen des aktuellen Override Wertes für den Kanal.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals, dessen Override gesetzt werden soll.
override	unsigned short	Zu setzender Overridewert in 0,1%.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.13 kernelv_ch_get_blocknumber()

Prototyp

```
KERNELV_RETURN kernelv_ch_get_blocknumber (unsigned long int chanIndex,
                                           signed long int *blocknumber);
```

Beschreibung

Lesen der aktuell abgearbeiteten Satznummer eines NC-Programms.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals, dessen Satznummer gelesen werden soll.
blocknumber	signed long*	Zeiger auf den Wert, in den die Satznummer geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.14 kernelv_ch_get_filename()

Prototyp

```
KERNELV_RETURN kernelv_ch_get_filename (unsigned long int chanIndex,
                                       char *filename,
                                       unsigned short int nameLength,
                                       unsigned short int* returnLength);
```

Beschreibung

Lesen des Dateinamens des aktuell im Kanal aktiven Programms.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals dessen Dateiname gelesen werden soll.
filename	char*	Zeiger auf den Speicherplatz für den Dateinamen.
nameLength	unsigned short	Länge des Speicherbereiches für den Dateinamen.
returnLength	unsigned short*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Es wird die Anzahl der Zeichen des Dateinamens +1 zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird in diesem Fall der benötigte Speicher inklusive der terminierenden 0 zurückgegeben.

3.15 kernelv_ch_get_programname()

Prototyp

```
KERNELV_RETURN        kernelv_ch_get_programname (unsigned long int chanIndex,
                                                  char *programname,
                                                  unsigned short int nameLength,
                                                  unsigned short int* returnLength);
```

Beschreibung

Lesen des Dateinamens des aktuell im Kanal aktiven Programms. Der Programmname wird am Anfang des NC-Programms angegeben (siehe auch Programmieranleitung).

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals, dessen Programmname gelesen werden soll.
programname	char*	Zeiger auf den Speicherplatz für den Programmnamen.
nameLength	unsigned short	Länge des Speicherbereiches für den Programmnamen.
returnLength	unsigned short*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Es wird die Anzahl der Zeichen des Dateinamens +1 zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird in diesem Fall der benötigte Speicher zurückgegeben.

3.16 kernelv_ch_get_state()

Prototyp

```
KERNELV_RETURN kernelv_ch_get_state (unsinged long int chanIndex,
                                     KERNELV_CHANNEL_STATE *state);
```

Beschreibung

Lesen des aktuellen Zustandes des Kanals.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex.
state	KERNELV_CHANNEL_STATE *	Zeiger auf den zurückzugebenden Zustand.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten.

3.17 kernelv_ch_get_fileoffset()

Prototyp

```
KERNELV_RETURN kernelv_ch_get_fileoffset (unsigned long int chanIndex,
                                          signed long int *fileoffset);
```

Beschreibung

Liefert den aktuellen Fileoffset in der Programmdatei.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex
fileoffset	signed long int *	Zeiger auf den Wert, in den der Dateioffset geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.18 kernelv_ch_get_techno_data()

Prototyp

```
KERNELV_RETURN kernelv_ch_get_techno_data (unsigned long int chanindex,
                                           KERNELV_TECHNO_DATA *
                                           technoData
                                           unsigned long int technoLength
                                           unsigned long int * returnLength);
```

Beschreibung

Gibt die während des letzten Aufrufes von kernelv_do_cycle() für den angegebenen Kanal quittierten Technologiefunktionen (M/H-Funktionen) zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex
technoData	KERNELV_TECHNO_DATA*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.

3.19 kernelv_ch_get_new_techno_data()

Prototyp

```
KERNELV_RETURN          kernelv_ch_get_new_techno_data (unsigned long int chanindex,
                                                        KERNELV_TECHNO_DATA * technoData
                                                          unsigned long int technoLength
                                                          unsigned long int * returnLength);
```

Beschreibung

Gibt die beim letzten Aufruf von kernelv_do_cycle() neu ausgegebenen Technologiefunktionen (M/H-Funktionen) für den jeweiligen Kanal zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex
technoData	KERNELV_TECHNO_DATA*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.

3.20 kernelv_ch_get_techno_data2()

Prototyp

```
KERNELV_RETUR  kernelv_ch_get_techno_data2 (unsigned long int chanindex,
N                                           KERNELV_TECHNO_DATA2 * technoData2
                                              unsigned long int technoLength
                                              unsigned long int * returnLength);
```

Beschreibung

Gibt die während des letzten Aufrufes von kernelv_do_cycle() für den angegebenen Kanal quittierten Technologiefunktionen (M/H-Funktionen) zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex
technoData	KERNELV_TECHNO_DATA2*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.

3.21 kernelv_ch_get_new_techno_data2()

Prototyp

```
KERNELV_RETURN      kernelv_ch_get_new_techno_data2 (unsigned long int chanindex,
                                                     KERNELV_TECHNO_DATA2 * technoData2
                                                       unsigned long int technoLength
                                                       unsigned long int * returnLength);
```

Beschreibung

Gibt die beim letzten Aufruf von kernelv_do_cycle() neu ausgegebenen Technologiefunktionen (M/H-Funktionen) für den jeweiligen Kanal zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex
technoData	KERNELV_TECHNO_DATA2*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.

3.22 kernelv_ch_get_finished_nc_lines()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_finished_nc_lines(unsigned long int chanIndex,
                                                   KERNELV_NC_LINE_DATA *ncLineData,
                                                   unsigned long int maxByteSize,
                                                   unsigned long int* returnLength);
```

Beschreibung

Gibt die während des letzten Aufrufes von kernelv_do_cycle() für den angegebenen Kanal ausgeführten NC-Zeilen zurück. Es können bis zu 20 NC-Sätze pro Aufruf von kernelv_do_cycle() abgearbeitet werden.

Zurückgegeben wird ein Array von Strukturen des Typs KERNELV_NC_LINE_DATA.

Die Anzahl der zurückgegebenen Einträge kann durch returnLength/sizeof(KERNELV_NC_LINE_DATA) berechnet werden.

Der Aufbau der Struktur ist im Abschnitt Struct KERNELV_NC_LINE_DATA beschrieben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex
ncLineData	KERNELV_NC_LINE_DATA*	Zeiger auf den Speicherbereich, in den die Daten der abgearbeiteten NC-Zeilen geschrieben werden sollen.
maxByteSize	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Falle wird in returnLength die Anzahl der tatsächlich benötigten Bytes zurückgeliefert

3.23 kernelv_ax_get_techno_data()

Prototyp

```
KERNELV_RETURN          kernelv_ax_get_techno_data (unsigned long int axisIndex,
                                                    KERNELV_TECHNO_DATA * technoData
                                                    unsigned long int technoLength
                                                    unsigned long int *returnLength);
```

Beschreibung

Gibt die während des letzten Aufrufes von kernelv_do_cycle() für die angegebene Achse quittierten Technologiefunktionen (M/H-Funktionen) zurück.

Parameter

Name	Typ	Bedeutung
axisIndex	unsigned long	Index der Achse.
technoData	KERNELV_TECHNO_DATA*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_INVALID_AX	-9	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen – 1.

3.24 kernelv_ax_get_new_techno_data()

Prototyp

```
KERNELV_RETURN          kernelv_ax_get_new_techno_data (unsigned long int axisIndex,
                                                        KERNELV_TECHNO_DATA * technoData
                                                        unsigned long int technoLength
                                                        unsigned long int *returnLength);
```

Beschreibung

Gibt die beim letzten Aufruf von kernelv_do_cycle() neu ausgegebenen Technologiefunktionen (M/H-Funktionen) für die jeweiligen Achsen zurück.

Parameter

Name	Typ	Bedeutung
axisIndex	unsigned long	Index der Achse.
technoData	KERNELV_TECHNO_DATA*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_INVALID_AX	-9	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen – 1.

3.25 kernelv_ax_get_techno_data2()

Prototyp

```
KERNELV_RETURN      kernelv_ax_get_techno_data2 (unsigned long int axisIndex,
                                                 KERNELV_TECHNO_DATA2 * technoData
                                                 unsigned long int technoLength
                                                 unsigned long int *returnLength);
```

Beschreibung

Gibt die während des letzten Aufrufes von kernelv_do_cycle() für die angegebene Achse quittierten Technologiefunktionen (M/H-Funktionen) zurück.

Parameter

Name	Typ	Bedeutung
axisIndex	unsigned long	Index der Achse.
technoData	KERNELV_TECHNO_DATA2*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_INVALID_AX	-9	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen – 1.

3.26 kernelv_ax_get_new_techno_data2()

Prototyp

```
KERNELV_RETURN      kernelv_ax_get_new_techno_data (unsigned long int axisIndex,
                                                    KERNELV_TECHNO_DATA2 * technoData
                                                    unsigned long int technoLength
                                                    unsigned long int *returnLength);
```

Beschreibung

Gibt die beim letzten Aufruf von kernelv_do_cycle() neu ausgegebenen Technologiefunktionen (M/H-Funktionen) für die jeweiligen Achsen zurück.

Parameter

Name	Typ	Bedeutung
axisIndex	unsigned long	Index der Achse.
technoData	KERNELV_TECHNO_DATA2*	Zeiger auf den Speicherbereich, in den die Technologiedaten geschrieben werden sollen.
technoLength	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
returnLength	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnLength wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_INVALID_AX	-9	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen – 1.

3.27 kernelv_ax_set_position()

Prototyp

```
KERNELV_RETURN    kernelv_ax_set_position (unsigned long int axisIndex,
                                           signed long int position);
```

Beschreibung

Setzt die Istposition der Achse auf die im Parameter position angegebene Position. Diese Funktion kann nur dann ausgeführt werden, wenn in dem Kanal, zu dem die Achse aktuell gehört, kein NC-Programm aktiv ist. Wird bei aktivem NC-Programm versucht die Achsposition zu setzen, wird die Übernahme der Position verweigert und der Rückgabewert ERR_INVALID_STATE zurückgegeben.

Parameter

Name	Typ	Bedeutung
axisIndex	unsigned long	Index der Achse.
position	unsigned long	Neue Istposition der Achse in 0,1 µm.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC Kern ist noch nicht initialisiert.
ERR_INVALID_STATE	-5	Der CNC-Kanal der die Achse ist im falschen Zustand um eine Funktion auszuführen.
ERR_INVALID_AX	-9	Die übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen – 1 oder Null.
ERR_INVALID_AX	-9	Die übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen – 1.
ERR_AXIS_ERROR	-10	Die CNC-Achse zeigt einen Fehler an. Es wird zusätzlich von der CNC eine Fehlermeldung ausgegeben.

3.28 kernelv_get_acs_command_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_acs_command_positions (unsigned long* positions,
                                                     unsigned long maxByteSize,
                                                     unsigned long* returnSize);
```

Beschreibung

Es werden die ACS-Sollpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben. Falls eine achsspezifische Sollwerttransformation für eine Achse konfiguriert ist, wird von dieser Funktion der transformierte Sollwert für die jeweilige Achse zurückgeliefert.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}°$ für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

3.29 kernelv_get_acs0_command_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_acs0_command_positions (unsigned long* positions,
                                                      unsigned long maxByteSize,
                                                      unsigned long* returnSize);
```

Beschreibung

Es werden die ACS-Sollpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben. Falls eine achsspezifische Sollwerttransformation für eine Achse konfiguriert ist, wird von dieser Funktion der untransformierte Sollwert für die jeweilige Achse zurückgeliefert.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}°$ für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

3.30 kernelv_get_acs_actual_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_acs_actual_positions (unsigned long* positions,
                                                    unsigned long maxByteSize,
                                                    unsigned long* returnSize);
```

Beschreibung

Es werden die ACS-Istpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Zur Berechnung der Istpositionen wird CNC-intern ein Lageregelkreis simuliert, wobei das mechanische Verhalten der Achse durch ein PT2-Glied nachgebildet wird.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}°$ für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

3.31 kernelv_get_acs_target_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_acs_target_positions (unsigned long* positions,
                                                    unsigned long maxByteSize,
                                                    unsigned long* returnSize);
```

Beschreibung

Es werden die ACS-Zielpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Die Zielposition ist die Achsposition am Ende des gerade bearbeiteten Bewegungssatzes.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4°}$ für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird in diesem Fall der benötigte Speicher zurückgegeben.

3.32 kernelv_get_acs_start_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_acs_start_positions (unsigned long* positions,
                                                   unsigned long maxByteSize,
                                                   unsigned long* returnSize);
```

Beschreibung

Es werden die ACS-Startpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Die Startposition ist die Achsposition die die Achse am Anfang des gerade bearbeiteten Bewegungssatzes hatte.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}$° für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

3.33 kernelv_get_wcs_command_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_wcs_command_positions (unsigned long* positions,
                                                     unsigned long maxByteSize,
                                                     unsigned long* returnSize);
```

Beschreibung

Es werden die WCS-Sollpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}$° für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird in diesem Fall der benötigte Speicher zurückgegeben.

3.34 kernelv_get_wcs_target_positions()

Prototyp

```
KERNELV_RETURN kernelv_get_wcs_target_positions (unsigned long* positions,
                                                 unsigned long maxByteSize,
                                                 unsigned long* returnSize);
```

Beschreibung

Es werden die WCS-Zielpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Die Zielposition ist die Achsposition am Ende des gerade bearbeiteten Bewegungssatzes.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4°}$ für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird in diesem Fall der benötigte Speicher zurückgegeben.

3.35 kernelv_get_wcs_start_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_wcs_start_positions (unsigned long* positions,
                                                   unsigned long maxByteSize,
                                                   unsigned long* returnSize);
```

Beschreibung

Es werden die WCS-Startpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Die Startposition ist die Achsposition die die Achse am Anfang des gerade bearbeiteten Bewegungssatzes hatte.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}$° für Spindeln oder Moduloachsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird in diesem Fall der benötigte Speicher zurückgegeben.

3.36 kernelv_get_prg_target_positions()

Prototyp

```
KERNELV_RETURN    kernelv_get_prg_target_positions (signed long int *positions,
                                                    unsigned long int maxByteSize,
                                                    unsigned long int *returnSize);
```

Beschreibung

Es werden die im NC-Programm programmierten Zielpositionen aller in der CNC vorhandenen Achsen in einem Array zurückgegeben.

Die zurückgegebenen Positionen sind die im gerade abgearbeiteten NC-Satz programmierten Positionen.

Falls durch die CNC Bewegungssätze eingefügt, bzw. programmierte Bewegungssätze aufgeteilt werden, (z.B. beim Überschleifen) wird für alle erzeugten NC-Sätze die Zielposition des Ursprungssatzes ausgegeben.

Die Auflösung der Positionswerte beträgt 0,1 µm für translatorische Achsen bzw. $1*10^{-4}$° für Spindeln oder rotatorische Achsen.

Falls der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe aller Werte ausreicht, wird der Fehlercode ERR_CNC_RET_MEMORY zurückgegeben.

Zur Rückgabe aller Positionswerte müssen mindestens Achsanzahl * sizeof(signed long int) Bytes vom Aufrufer bereitgestellt werden.

Die Reihenfolge der Achspositionen im zurückgegebenen Array ist gleich der Konfigurationsreihenfolge der Achsen.

Falls eine Achse gerade keinem NC-Kanal zugeordnet ist, wird für diese Achse der Wert Null zurückgegeben.

Parameter

Name	Typ	Bedeutung
positions	unsigned long*	Zeiger auf Speicherbereich für die zurückzugebenden Achspositionen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird in diesem Fall der benötigte Speicher zurückgegeben.

3.37 kernelv_get_axis_channel_number()

Prototyp

```
KERNELV_RETURN    kernelv_get_axis_channel_number (unsigned short* chanNumbers,
                                                   unsigned long maxByteSize,
                                                   unsigned long* returnSize);
```

Beschreibung

Durch Konfiguration bzw. durch Achstausch-Befehle während eines NC-Programmes kann eine Achse von unterschiedlichen CNC-Kanälen bewegt werden. Durch diese Funktion kann die Nummer des Kanals der die jeweilige Achse bewegt abgefragt werden.

Zwischen der Kanalnummer und dem Kanalindex besteht die Beziehung

Kanalindex = Kanalnummer –1.

Parameter

Name	Typ	Bedeutung
chanNumbers	unsigned short*	Zeiger auf Speicherbereich für die zurückzugebenden Kanalnummern.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achspositionen.
returnSize	unsigned long*	Anzahl der in positions zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird in diesem Fall der benötigte Speicher zurückgegeben.

3.38 kernelv_ch_get_variable_value()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_variable_value (unsigned long int chanIndex,
                                                 char* varName,
                                                 KERNELV_VARIABLE *var);
```

Beschreibung

Lesen einer Variablen aus dem CNC-Kernel. Aktuell können die folgenden Variablentypen gelesen werden:

Typ	Beispiel
Externe Variablen	V.E.VAR_1
Globale Variablen, siehe (1)	V.G.CNC_RELEASE
Achsspezifische Variablen	V.A.+SWE.X
Programmglobale, eigendefinierte Variablen	V.P.VAR_1
Programmübergreifende, eigendefinierte Variablen	V.S.VAR_1
Programmlokale, eigendefinierte Variablen	V.L.VAR_1

Variablen des Typs V.G.WZ[]. können aktuell nicht gelesen werden.

Die Identifikation der zu lesenden Variablen erfolgt anhand ihres Namens und des Kanalindex.

Als Name ist der vollständige Name (inklusive Präfix V.E. und Arrayindex bei Arrayvariablen) anzugeben.

Beispiel: „V.E.VAR_FLOAT_ARRAY[3]“

Die Rückgabe des Wertes erfolgt in der Struktur KERNELV_VARIABLE *var.

Falls beim Lesen der Variablen ein Fehler auftrat, wird ein Fehlercode zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
varName	char*	Zeiger auf den Variablennamen.
var	KERNELV_VARIABLE*	Zeiger auf Struktur, in die der Variablenwert und Typ geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher.
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten.
ERR_UNKNOWN_VARIABLE	-12	Der Variablenname ist im CNC-Kern nicht bekannt.
ERR_VARIABLE_SYNTAX	-13	Der Variablenname ist syntaktisch nicht korrekt, z.B. schließende Klammer bei Arrayvariablen fehlt.
ERR_VAR_NAME_LENGTH	-18	Der an die Funktion übergebene Variablenname überschreitet die maximal zulässige Länge (KERNELV_VAR_NAME_LENGTH).

ERR_ARRAY_NOT_SUPPORTED	-21	Bei manchen Echtzeit Varianten der CNC ist es möglich ein Array ‚en block' zu lesen, indem beim Zugriff der Arrayindex weggelassen wird, diese Zugriffsart wird aktuell von der kernelv-DLL nicht unterstützt.

3.39 kernelv_ch_set_variable_value()

Prototyp

```
KERNELV_RETURN    kernelv_ch_set_variable_value (unsigned long int chanIndex,
                                                 char* varName,
                                                 KERNELV_VARIABLE *var);
```

Beschreibung

Schreiben einer CNC-Kernel definierten Variablen.

Das Schreiben einer Variablen ist nur möglich, wenn diese auch vom NC-Programm aus beschreibbar ist. Externe Variablen können unabhängig von der konfigurierten Zugriffsart immer geschrieben werden.

Aktuell können die folgenden Variablentypen geschrieben werden:

Typ	Beispiel
Externe Variablen	V.E.VAR_1
Globale Variablen, siehe (1)	V.G.WZ_AKT.R
Achsspezifische Variablen	V.A.WCS.X
Programmglobale, eigendefinierte Variablen	V.P.VAR_1
Programmübergreifende, eigendefinierte Variablen	V.S.VAR_1
Programmlokale, eigendefinierte Variablen	V.L.VAR_1

Variablen des Typs V.G.WZ[]. können aktuell nicht geschrieben werden.

Die Identifikation der zu schreibenden Variablen erfolgt anhand ihres Namens und des Kanalindex.

Als Name ist der vollständige Name (inklusive Präfix V.E. und Arrayindex bei Arrayvariablen) anzugeben.

Beispiel: „V.E.VAR_FLOAT_ARRAY[3]“

Der zu schreibende Variablenwert ist in der Struktur KERNELV_VARIABLE *var zu übergeben.

Falls beim Schreiben der Variablen ein Fehler auftrat, wird ein Fehlercode zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
varName	char*	Zeiger auf den Variablennamen.
var	KERNELV_VARIABLE*	Zeiger auf Struktur, in die der Variablenwert und Typ geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher.
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten.
ERR_UNKNOWN_VARIABLE	-12	Der Variablenname ist im CNC-Kern nicht bekannt.
ERR_VARIABLE_SYNTAX	-13	Der Variablenname ist syntaktisch nicht korrekt, z.B. schließende Klammer bei Arrayvariablen fehlt.

ERR_DATA_TYPE_MISMATCH	-14	Bei einem Schreibzugriff auf eine Variable passt stimmt der übergebene Datentyp nicht mit dem CNC-intern verwendeten Datentyp überein.
ERR_VAR_NAME_LENGTH	-18	Der an die Funktion übergebene Variablenname überschreitet die maximal zulässige Länge (KERNELV_VAR_NAME_LENGTH).
ERR_ARRAY_NOT_SUPPORTED	-21	Bei manchen Echtzeit Varianten der CNC ist es möglich ein Array ‚en block' zu lesen bzw. zu schreiben, indem beim Zugriff der Arrayindex weggelassen wird, diese Zugriffsart wird aktuell von der kernelv-DLL nicht unterstützt.
ERR_VAR_NOT_WRITEABLE	-22	Es wurde versucht eine nicht schreibbare Variable zu beschreiben. Für den Schreibzugriff auf Variable gelten dieselben Zugriffsregeln, wie sie auch innerhalb eines NC-Programms gelten. Einzige Ausnahme hierbei sind V.E-Variable, diese können, unabhängig von den konfigurierten Zugriffsrechten, immer beschrieben werden.

3.40 kernelv_get_channel_count()

Prototyp

```
KERNELV_RETURN kernelv_get_channel_count (unsigned long int* channelCount);
```

Beschreibung

Liest die Anzahl der konfigurierten Kanäle der CNC-Steuerung.

Falls die Funktion aufgerufen wird, wenn der CNC-Kern noch nicht gestartet ist, wird ein Fehlercode zurückgegeben.

Parameter

Name	Typ	Bedeutung
channelCount	unsigned long*	Zeiger auf Speicher für die zurückzugebende Kanalanzahl.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.41 kernelv_get_axis_count()

Prototyp

```
KERNELV_RETURN kernelv_get_axis_count (unsigned long int* axisCount);
```

Beschreibung

Liest die Anzahl der konfigurierten Achsen der CNC-Steuerung.

Falls die Funktion aufgerufen wird, wenn der CNC-Kern noch nicht gestartet ist, wird ein Fehlercode zurückgegeben.

Parameter

Name	Typ	Bedeutung
axisCount	unsigned long*	Zeiger auf Speicher für die zurückzugebende Achsanzahl.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.42 kernelv_sync_read_request()

Prototyp

```
signed long int    kernelv_sync_read_request (unsigned short port,
                                              unsigned long int indexGroup,
                                              unsigned long int indexOffset,
                                              unsigned long int length,
                                              void* data);
```

Beschreibung

Synchrones Lesen von Variablen. Die Variablen werden dabei durch Port, indexGroup und indexOffset identifiziert.

Aktuell werden die folgenden Ports unterstützt:

Portnummer	CNC-Task
551	Task GEO
552	Task SDA
553	Task COM

Die Bedeutung von indexGroup und indexOffset ist abhängig vom adressierten Port und kann der Dokumentation entnommen werden.

Falls die Funktion aufgerufen wird, wenn der CNC-Kern noch nicht gestartet ist, wird ein Fehlercode zurückgegeben.

Parameter

Name	Typ	Bedeutung
port	unsigned short	Port von dem die Daten zu lesen sind.
indexGroup	unsigned long*	Indexgroup des zu lesenden Datums.
indexOffset	unsigned long*	Indexoffset des zu lesenden Datums.
length	unsigned long*	Größe des Speichers für den zu lesenden Wert in Bytes.
data	void*	Zeiger auf Speicher für den zu lesenden Wert.

Rückgabewerte

Symbol	Wert	Bedeutung
	0	Die Funktion wurde fehlerfrei durchgeführt.
	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
	6	Die übergebene Portnummer ist nicht bekannt.
	0x701	Der angeforderte Dienst wird nicht unterstützt.
	0x702	Indexgroup ist ungültig
	0x703	Indexoffset ist ungültig.
	0x704	Die adressierte Variable darf nicht gelesen werden.
	0x705	Der bereitgestellte Speicher ist zu klein für den zurückzugebenden Wert.

3.43 kernelv_sync_write_request()

Prototyp

```
signed long int        kernelv_sync_write_request (unsigned short port,
                                                   unsigned long int indexGroup,
                                                   unsigned long int indexOffset,
                                                   unsigned long int length,
                                                   void* data);
```

Beschreibung

Synchrones Schreiben von Variablen. Die Variablen werden dabei durch Port, indexGroup und indexOffset identifiziert.

Aktuell werden die folgenden Ports unterstützt:

Portnummer	CNC-Task
551	Task GEO
552	Task SDA
553	Task COM

Die Bedeutung von indexGroup und indexOffset ist abhängig vom adressierten Port und kann der Dokumentation entnommen werden.

Falls die Funktion aufgerufen wird, wenn der CNC-Kern noch nicht gestartet ist, wird ein Fehlercode zurückgegeben.

Abhängig vom verwendeten Indexgroup und Indexoffset kann ein Schreibvorgang mehrere NC-Zyklen dauern. In diesem Fall gibt die Funktion den Wert 1 (RET_BUSY) zurück.

Ist dies der Fall ist die ist diese Funktion nach jedem Aufruf von kernelv_do_cycle() erneut aufzurufen, bis entweder der Rückgabewert 0 (RET_FINISHED) oder eine Fehlerkennung zurückgegeben wird.

Parameter

Name	Typ	Bedeutung
port	unsigned short	Port über den die Daten zu schreiben sind.
indexGroup	unsigned long	Indexgroup des zu schreibenden Datums.
indexOffset	unsigned long	Indexoffset des zu schreibenden Datums.
length	unsigned long*	Größe des Speichers für den zu schreibenden Wert in Bytes.
data	void*	Zeiger auf Speicher für den zu schreibenden Wert.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
	6	Die übergebene Portnummer ist nicht bekannt.
	0x701	Der angeforderte Dienst wird nicht unterstützt.
	0x702	Indexgroup ist ungültig
	0x703	Indexoffset ist ungültig.
	0x704	Die adressierte Variable darf nicht geschrieben werden.
	0x705	Die zu schreibenden Daten passen nicht in die adressierte Variable.

3.44 kernelv_sync_read_write_req()

Prototyp

```
signed long int        kernelv_sync_write_req (unsigned short port,
                                               unsigned long int indexGroup,
                                               unsigned long int indexOffset,
                                               unsigned long int *readLength,
                                                unsigned long int writeLength,
                                               void* data);
```

Beschreibung

Synchrones Schreiben und Lesen von Variablen. Die Variablen werden dabei durch Port, indexGroup und indexOffset identifiziert.

Beim Aufruf beinhaltet der Zeiger data die zu schreibenden Daten, in writeLength ist die Länge des zu schreibenden Datenbereiches in Bytes anzugeben, die Anzahl der Bytes die gelesen werden sollen ist im Zeiger readLength einzutragen.

Die gelesenen Daten werden in den Speicher, auf den der Zeiger data zeigt abgelegt, und die Anzahl der geschriebenen Bytes in *readLength eingetragen.

Aktuell werden die folgenden Ports unterstützt:

Portnummer	CNC-Task
551	Task GEO
552	Task SDA

Die Bedeutung von indexGroup und indexOffset ist abhängig vom adressierten Port und kann der Dokumentation entnommen werden.

Falls die Funktion aufgerufen wird, wenn der CNC-Kern noch nicht gestartet ist, wird ein Fehlercode zurückgegeben.

Parameter

Name	Typ	Bedeutung
port	unsigned short	Port von dem die Daten zu lesen sind.
indexGroup	unsigned long	Indexgroup des zu lesenden Datums.
indexOffset	unsigned long	Indexoffset des zu lesenden Datums.
readLength	unsigned long*	Zeiger auf Rückgabewert, für die Anzahl der zurückgelieferten Bytes.
writeLength	unsigned long	Länge des zu schreibenden Datenbereiches.
data	void*	Zeiger auf Speicher für den zu schreibenden bzw. zu lesenden Daten.

Rückgabewerte

Symbol	Wert	Bedeutung
	0	Die Funktion wurde fehlerfrei durchgeführt.
	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher.
	6	Die übergebene Portnummer ist nicht bekannt.
	0x701	Der angeforderte Dienst wird nicht unterstützt.
	0x702	Indexgroup ist ungültig
	0x703	Indexoffset ist ungültig.

	0x704	Die adressierte Variable darf nicht geschrieben werden.
	0x705	Die zu schreibenden Daten passen nicht in die adressierte Variable.

3.45 kernelv_get_axis_names()

Prototyp

```
KERNELV_RETURN    kernelv_get_axis_names(char* axisNames,
                                         unsigned long int maxByteSize,
                                         unsigned long int *returnSize,
                                         unsigned long int *nAxis);
```

Beschreibung

Liefert die in den jeweiligen Kanälen verwendeten Achsnamen einer Achse. Der Achsname ist der Name, mit dem die Achse in einem NC-Programm angesprochen wird. Nur Achsen, die in einem Kanal sind haben einen Achsnamen.

Zurückgeliefert wird ein Array von Zeichenketten fester Länge. Die Länge einer einzelnen Zeichenkette (inklusive terminierender Null) kann wie folgt ermittelt werden:

Namenslänge = returnSize / Anzahl der konfigurierten Achsen.

Die Anzahl der konfigurierten Achsen kann mit der Funktion kernelv_get_axis_count() ermittelt werden.

Die Indizes der zurückgegebenen Achsnamen entsprechen der Konfigurationsreihenfolge. Siehe auch Abschnitt 1.2.3.1.

Falls der Achsname kürzer ist als Namenslänge – 1 Zeichen, wird die Zeichenkette mit dem Wert \0 aufgefüllt.

Die Indizes der zurückgegebenen Achsnamen entsprechen der Konfigurationsreihenfolge. Siehe auch Abschnitt 1.2.3.1.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebenden Werte ist, wird der Wert ERR_CNC_RET_MEMORY zurückgegeben, in die Variable returnSize wird in diesem Falle der zur Rückgabe minimal benötigte Speicher in Bytes eingetragen.

Parameter

Name	Typ	Bedeutung
axisNames	char*	Zeiger auf Speicherbereich für die zurückzugebenden Achsnamen.
maxByteSize	unsigned long	Größe des Speicherbereiches für die Achsnamen.
returnSize	unsigned long*	Anzahl der in axisNames zurückgegebenen Bytes.
nAxis	unsigned long*	Anzahl der in axisNames zurückgegebenen Achsnamen.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

Es sind 4 Achsen konfiguriert, nur die Achsen mit den Indizes 0 (Achsname ‚X2') und 2 (Achsname ‚C') sind einem Kanal zugeordnet:Ein Aufruf von kernelv_get_axis_names() liefert folgende Werte zurück:

Parameter	Wert	Kommentar
returnSize	80	Anzahl der konfigurierten Achsen * Namenslänge

nAxis	2	Anzahl der Achsen, die aktuell einem Kanal zugeordnet sind.
axisNames		Siehe folgende Tabelle.

Die Länge (nameLength) einer einzelnen Zeichenkette ist damit:

namelength = returnSize / Anzahl konf. Achsen = 80 / 4 = 20

axisNames hat den folgenden Aufbau:

Offset	**Wert**
0	'**X**', '**2**'; \0, \0, \0 ….
20 (1 * nameLength)	\0, \0, \0, …..
40 (2 * nameLength)	'**C**'; \0, \0, \0 ….
60 (3 * nameLength)	\0, \0, \0, …..

3.46 kernelv_control_techno_func_duration()

Prototyp

KERNELV_RETURN kernelv_control_techno_func_duration (unsigned char onOff);

Beschreibung

Aktiviert bzw. deaktiviert die Bearbeitungszeitsimulation für Technologiefunktionen.

Standardmäßig ist die Bearbeitungszeitsimulation ausgeschaltet, d. h. Technologiefunktionen werden sofort quittiert. Bei aktiver Bearbeitungszeitsimulation wird die Quittierung der Technologiefunktionen verzögert, µm die tatsächliche Ausführungszeit der Technologiefunktionen zu simulieren.

Die Ausführungszeit kann durch die Funktion kernelv_ch_set_techno_func_duration() oder durch Einträge in der Kanalparameterliste (P-CHAN-00040, P-CHAN-00026) eingestellt werden.

Wird die Bearbeitungszeitsimulation deaktiviert während noch nicht quittierte Technologiefunktionen anstehen, werden diese unabhängig von der eingestellten Bearbeitungszeit sofort quittiert.

Parameter

Name	Typ	Bedeutung
onOff	unsigned char	Bei einem Wert von onOff > 0 wird die Bearbeitungszeitsimulation aktiviert, ein Wert von 0 deaktiviert die Bearbeitungszeitsimulation.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.

3.47 kernelv_ch_set_techno_func_duration()

Prototyp

```
KERNELV_RETURN    kernelv_ch_set_techno_func_duration (unsigned long int chanIndex,
                                                       E_KERNELV_TECHNO_TYPE type,
                                                       unsigned long int number,
                                                       unsigned long int time_us);
```

Beschreibung

Setzt die Bearbeitungszeit für die übergebene Technologiefunktion.

Die Bearbeitungszeit ist dabei in Mikrosekunden anzugeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long int	Kanalindex.
type	E_KERNELV_TECHNO_TYPE	Typ der Technologiefunktion, deren Bearbeitungszeit geschrieben werden soll.
number	unsigned long int	Nummer der Technologiefunktion, deren Bearbeitungszeit geschrieben werden soll.
time_us	unsigned long int	Bearbeitungszeit in us.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten. Der Wert konnte nicht geschrieben werden
ERR_INVALID_TECHNO_PARAM	-16	Beim Setzten der Bearbeitungszeit für eine Technologiefunktion wurde ein ungültiger Parameter Übergeben, z.B. übergebene Nummer der M- bzw. H-Funktion ist größer als die maximal zulässige Anzahl.
ERR_UNKNOWN_TECHNO_TYPE	-15	Beim Setzen der Bearbeitungszeit für eine Technologiefunktion wurde ein ungültiger Typ für die Technologiefunktion angegeben.

3.48 kernelv_ch_set_techno_func_user_ackn()

Prototyp

```
KERNELV_RETURN    kernelv_ch_set_techno_func_user_ackn (unsigned long int chanIndex,
                                                        E_KERNELV_TECHNO_TYPE type,
                                                        unsigned long int number);
```

Beschreibung

Schaltet bei aktiver Bearbeitungszeitsimulation die automatische Quittierung nach Ablauf der gesetzten Bearbeitungszeit aus. Die Technologiefunktion ist vom Anwender durch Aufruf einer der Funktionen kernelv_ch_ackn_techno_func() bzw, kernelv_ax_ackn_techno_func() zu quittieren.

Die Quittierung durch den Anwender ist nur möglich, wenn Bearbeitungszeitsimulation aktiv ist.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long int	Kanalindex.
type	E_KERNELV_TECHNO_TYPE	Typ der Technologiefunktion, die durch den Anwender quittiert werden soll.
number	unsigned long int	Nummer der Technologiefunktion.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten. Der Wert konnte nicht geschrieben werden
ERR_INVALID_TECHNO_PARAM	-16	Beim Aktivieren der Anwenderquittierung wurde ein ungültiger Parameter Übergeben, z.B. übergebene Nummer der M- bzw. H-Funktion ist größer als die maximal zulässige Anzahl.
ERR_UNKNOWN_TECHNO_TYPE	-15	Beim Aktivieren der Anwenderquittierung für eine Technologiefunktion wurde ein ungültiger Typ für die Technologiefunktion angegeben.

3.49 kernelv_ch_ackn_techno_func()

Prototyp

```
KERNELV_RETURN    kernelv_ch_ackn_techno_func(unsigned long int chanIndex,
                                              E_KERNELV_TECHNO_TYPE type,
                                              unsigned long int number);
```

Beschreibung

Quittiert eine Technologiefunktion für die die Anwenderquittierung aktiviert wurde.

Falls die angegebene Technologiefunktion nicht offen ist (zur Quittierung ansteht), wird der Rückgabewert ERR_TECHNO_NOT_FOUND zurückgegeben, andernfalls RET_FINISHED.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long int	Kanalindex.
type	E_KERNELV_TECHNO_TYPE	Typ der Technologiefunktion, die quittiert werden soll.
number	unsigned long int	Nummer der Technologiefunktion.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Technologiefunktion wurde quittiert.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_TECHNO_NOT_FOUND	-29	Die angegebene Technologiefunktion wurde nicht gefunden.

3.50 kernelv_ax_ackn_techno_func()

Prototyp

```
KERNELV_RETURN    kernelv_ax_ackn_techno_func(unsigned long int axIndex,
                                              E_KERNELV_TECHNO_TYPE type,
                                              unsigned long int number);
```

Beschreibung

Quittiert eine Technologiefunktion für die die Anwenderquittierung aktiviert wurde.

Falls die angegebene Technologiefunktion nicht offen ist (zur Quittierung ansteht), wird der Rückgabewert ERR_TECHNO_NOT_FOUND zurückgegeben, andernfalls RET_FINISHED.

Parameter

Name	Typ	Bedeutung
axdex	unsigned long int	Achsindex.
type	E_KERNELV_TECHNO_TYPE	Typ der Technologiefunktion, die quittiert werden soll.
number	unsigned long int	Nummer der Technologiefunktion.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Technologiefunktion wurde quittiert.
ERR_INVALID_CHAN	-1	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_TECHNO_NOT_FOUND	-29	Die angegebene Technologiefunktion wurde nicht gefunden.

3.51 kernelv_get_license_info()

Prototyp

```
KERNELV_RETURN    kernelv_get_license_info (KERNELV_LICENSE_INFO *pLicenseInfo);
```

Beschreibung

Liest die vorhandenen Lizenzinformationen aus.

Die Funktion kann erst nach erfolgreichem Aufruf der Funktion kernelv_startup() verwendet werden.

Die Funktion schreibt die vorhandenen Lizenzinformationen in den durch den Aufrufparameter pLicenseInfo referenzierten Speicher.

Parameter

Name	Typ	Bedeutung
pLicenseInfo	KERNELV_LICENSE_INFO*	Speicher für die zurückzugebenden Lizenzinformationen.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.52 kernelv_set_options()

Prototyp

KERNELV_RETURN kernelv_set_options (unsigned long int optionsMask);

Beschreibung

Mit dieser Funktion kernelv-interne Funktionen wie z.B. die Ausgabe von Meldungen während der Lizenzprüfung aktiviert werden.

Abhängig von der zu aktivierenden Funktion muss diese Funktion eventuell vor der Funktion kernelv_startup() aufgerufen werden.

Jeder Funktionalität ist ein Bit im Aufrufparameter dieser Funktion zugeordnet. Die Funktionalität wird durch Setzen des entsprechenden Bits aktiviert.

Sollen mehrere Funktionalitäten aktiviert werden, so ergibt sich die resultierende Bitleiste aus der ODER-Verknüpfung der einzelnen Bitmasken.

Falls in der der Funktion übergebenen Bitleiste Bits gesetzt sind, denen keine Funktionalität zugeordnet ist, werden die unbekannten Bits ignoriert und der Wert ERR_UNKNOWN_OPTION zurückgegeben. Auch in diesem Fall werden die bekannten Bits ausgewertet und die zugeordneten Funktionalitäten aktiviert.

Parameter

Name	Typ	Bedeutung
optionsMask	unsigned long	Bitleiste mit Optionen.

Mögliche Werte für die Bitmaske sind:

Symbol	Wert	Bedeutung
KERNELV_OPTION_ LICENSE_CHECK_VERBOSE	0x1	Während der Lizenzprüfung beim Hochlauf werden Meldungen zum Ablauf der Lizenzprüfung ausgegeben. Dient zur Fehlersuche bei Lizenzierungsproblemen.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.53 kernelv_ch_get_decoder_positions()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_decoder_positions (unsigned long int chanIndex,
                                                    void* ret_buffer,
                                                    unsigned long int  buffer_size,
                                                    unsigned long int *ret_length));
```

Beschreibung

Mit dieser Funktion können vom Decoder die gerade aktiven Achspositionen im Maschinenkoordinaten sowie in Programmierkoordinaten angefordert werden.

Die von der Funktion zurückgelieferten Werte werden dabei in den Speicherbereich, der auf den buffer zeigt, geschrieben. In der Variablen buffer_size ist dabei die Größe dieses Speicherbereiches anzugeben. In ret_length wird die tatsächliche Größe der zurückgelieferten Daten zurückgegeben. Wenn die zurückzuliefernden Daten nicht in den bereitgestellten Speicherbereich passen, wird der Fehlerkode ERR_CNC_RET_MEMORY zurückgegeben. In diesem Fall beinhaltet ret_length die zur Rückgabe benötigte Speichergröße.

Die zurückgelieferten Daten haben die folgende Struktur, die einzelnen Datenelemente der Strukturen liegen dabei gepackt im Speicher:

Byte-Index	Typ	Bedeutung
0	KERNELV_DECODER_POSITION_HEADER	Allgemeine Daten über die gelieferten Decoderpositionen, siehe 2.2.24 [▶ 142]. Im Element axis_count ist die Anzahl der folgenden Strukturen des Typs KERNELV_DECODER_POSITION_DATA eingetragen.
11	KERNELV_DECODER_POSITION_DATA	Achspositionen und Achsnummer der 1. Achse. Strukturdefinition siehe 2.2.25 [▶ 143]
25	KERNELV_DECODER_POSITION_DATA	Achspositionen und Achsnummer der 2. Achse. Strukturdefinition siehe 2.2.25 [▶ 143]
39	…	...

Da aktuell die maximal mögliche Anzahl von Achsen im Kanal auf 32 begrenzt ist, beträgt die maximale Größe des Rückgabewertes 459 Bytes.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Index des Kanals
ret_buffer	void*	Zeiger auf den Speicherbereich, in den die Rückgabedaten geschrieben werden sollen.
buffer_size	unsigned long	Größe des bereitgestellten Speicherbereiches in Byte.
ret_Length	unsigned long*	Zeiger auf den Speicherbereich, in den die Anzahl der tatsächlich zurückgelieferten Bytes geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1

ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher.

3.54 kernelv_ch_get_prog_start_mode()

Prototyp

```
KERNELV_RETURN      kernelv_ch_get_prog_start_mode  (unsigned long int chanIndex,
                                                     E_KERNELV_PROG_START_MODE* mode);
```

Beschreibung

Liest den Bearbeitungsmodus des aktuell laufenden Programms.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
mode	E_KERNELV_PROG _START_MODE*	Zeigen auf Enum für den zurückgelieferten Bearbeitungsmodus.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.55 kernelv_ch_set_cont_visu_grid()

Prototyp

```
KERNELV_RETURN    kernelv_ch_set_cont_visu_grid  (unsigned long int chanIndex,
                                                  unsigned long int  grid);
```

Beschreibung

Setzt das Ausgaberaster für Linearsätze bei der Sollkonturvisualisierung.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
grid	unsigned long	Ausgaberaster für Linearsätze in 0,1 um.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.56 kernelv_ch_set_cont_visu_rel_curvature_error()

Prototyp

```
KERNELV_RETURN      kernelv_ch_set_cont_visu_rel_curvature_error
                                (unsigned long int chanIndex,
                                 unsigned long int  rel_error);
```

Beschreibung

Setzt den relativen Krümmungsfehler der Sollkonturvisualisierung für gekrümmte Konturen (Kreissegmente und Polynome).

Der relative Krümmungsfehler gibt den zulässigen Sekantenfehler bei der Abtastung der gekrümmten Kontur als prozentualen Wert des Krümmungsradius an.

Beispiel: Bei einem Kreis ist der Krümmungsradius gleich dem Kreisradius, bei einem Kreisradius von 100 mm und einem relativen Krümmungsfehler von 1 % ergibt sich der zulässige Sekantenfehler zu 100 mm * 1% = 1 mm.

Der für die Abtastung eines Konturelements wirksame Sekantenfehler wird durch das Minimum des absoluten und relativen Sekantenfehlers bestimmt.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
rel_error	unsigned long	Relativer Krümmungsfehler in 0,1 %.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.57 kernelv_ch_set_cont_visu_abs_curvature_error()

Prototyp

```
KERNELV_RETURN    kernelv_ch_set_cont_visu_abs_curvature_error
                                (unsigned long int chanIndex,
                                 unsigned long int  rel_error);
```

Beschreibung

Setzt den absoluten Krümmungsfehler der Sollkonturvisualisierung für gekrümmte Konturen (Kreissegmente und Polynome).

Der absolute Krümmungsfehler gibt den zulässigen Sekantenfehler bei der Abtastung der gekrümmten Kontur an.

Der für die Abtastung eines Konturelements wirksame Sekantenfehler wird durch das Minimum des absoluten und relativen Sekantenfehlers bestimmt.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
rel_error	unsigned long	Relativer Krümmungsfehler in 0,1 %.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.58 kernelv_ch_get_cont_visu_data()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_cont_visu_data (unsigned long int chanIndex,
                                                 unsigned char* pData,
                                                 unsigned long int maxByteSize,
                                                 unsigned long int* retLength);
```

Beschreibung

Liest die Konturvisualisierungsdaten eines Kanals aus. Das Format der zurückgelieferten Daten wird durch den Parameter P-STUP-00039 eingestellt. Zurückgeliefert wird eine Struktur des Typs CONTOUR_VISU, gefolgt von Strukturen des Typs COUNTOUR_VISU_DATA_V0 … _V8.

Sollte der vom Aufrufer bereitgestellte Speicher nicht zur Rückgabe der Struktur CONTOUR_VISU ausreichen wird ERR_CNC_RET_MEMORY zurückgegeben und in retLength die Mindestgröße des benötigten Speicherbereichs zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
pData	unsigned char*	Zeiger auf den Speicherbereich für Rückgabewerte.
maxByteSize	unsigned long	Größe des Rückgabespeichers.
retLength	unsigned long*	Anzahl der in pData zurückgegebenen Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

3.59 kernelv_ch_get_active_g_codes()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_active_g_codes (unsigned long int chanIndex,
                                                 ACTIVE_G_CODES* pGCodes);
```

Beschreibung

Liest die im angegebenen Kanal aktiven G-Funktionsgruppen. Zurückgeliefert wird eine Struktur des Typs ACTIVE_G_CODES. In dieser Struktur sind die aktiven G-Funktionsgruppen eingetragen. Falls in einem Eintrag der Wert -1 steht, bedeutet dies, dass der Eintrag nicht belegt ist.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
pGCodes	ACTIVE_G_ CODES*	Zeiger auf die Struktur, in der die Werte zurückgegeben werden.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC- Kern ist noch nicht initialisiert.

3.60 kernelv_get_active_g_group()

Prototyp

```
signed short        kernelv_get_active_g_group (ACTIVE_G_CODES* pGCodes,
                                                E_KERNELV_G_GROUP_TYPE Type);
```

Beschreibung

Nimmt eine Struktur des Typs ACTIVE_G_CODES entgegen und gibt den Inhalt der in E_KERNELV_G_GROUP_TYPE angegebenen Gruppe zurück.

Utilityfunktion für Auswertung der von kernelv_ch_get_active_g_codes() zurückgelieferten Struktur.

Parameter

Name	Typ	Bedeutung
pGCodes	ACTIVE_G_CODES*	Zeiger auf die Struktur, aus der eine G-Gruppe gelesen werden soll.
Type	E_KERNELV_G_GROUP_TYPE	Type-Id der G-Gruppe, die gelesen werden soll..

Rückgabewerte

short int: Aktive G-Funktion der angefragten Gruppe oder -1, falls in der angefragten Gruppe kein gültiger Wert eingetragen ist.

3.61 kernelv_ch_get_command_feed()

Prototyp

```
KERNELV_RETURN  kernelv_ch_get_command_feed  (unsigned long int chanIndex,
                                              signed long int* command_feed);
```

Beschreibung

Gibt den programmierten Vorschub in µm/s zurück.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
command_feed	signed long int*	Zeiger auf Variable für den Rückgabewert.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.62 kernelv_ch_get_active_feed()

Prototyp

```
KERNELV_RETURN      kernelv_ch_get_active_feed  (unsigned long int chanIndex,
                                                 signed long int* command_feed);
```

Beschreibung

Gibt den tatsächlich gefahrenen Vorschub in µm/s zurück..

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals von dem die Variable gelesen werden soll.
command_feed	signed long int*	Zeiger auf Variable für den Rückgabewert.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle - 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.63 kernelv_set_call_ratio()

Prototyp

```
KERNELV_RETURN    kernelv_set_call_ratio (unsigned short int dec_calls,
                                          unsigned short  int interpolator_calls);
```

Beschreibung

Legt das Verhältnis von Decoder-Aufrufen zu Interpolator-Aufrufen fest. Pro Aufruf von kernelv_do_cycle() wird ein Interpolatoraufruf durchgeführt. In vielen Echtzeitumgebungen kann die Zykluszeit für die Bahnplanungs-Task unabhängig von der Interpolator-Zykluszeit eingestellt werden. Mit dieser Funktion kann das Verhältnis von Bahnplanungszyklen zu Interpolatorzyklen für die kernelv-DLL eingestellt werden.

Ein Verhältnis von 5 Bahnplanungsaufrufen zu 2 Interpolatoraufrufen wird durch kernelv_set_call_ratio(5, 2) erreicht.

Die Parameter dec_calls und interpolator_calls dürfen beide nicht 0 sein, das Verhältnis der beiden Parameter muss im Bereich [0,05 , 20] liegen.

Parameter

Name	Typ	Bedeutung
dec_calls	unsigned short	Anzahl der Bahnplanungszyklen.
interpolator_calls	unsigned short	Anzahl der Interpolatorzyklen.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_ PARAMETER	-30	Einer der übergebenen Parameter ist ungültig. Es gelten die folgenden Bedingungen: dec_calls, interpolator_calls <> 0 0,05 ≤ dec_calls/interpolator_calls ≤ 20.

3.64 CNC Fehlermeldungen mit kernelv

Von der CNC ausgegebene Fehlermeldungen können mit der kernelv-DLL ausgelesen und angezeigt werden. Hierzu gibt es 2 Möglichkeiten:

1. Auslesen der fertig formatierten Fehlermeldung als Zeichenkette. Hier wird die Fehlermeldung im selben Format, mit der sie im Fehlermeldungslog eingetragen wird als Zeichenkette zurückgegeben. Hierzu kann die Funktion kernelv_get_error() verwendet werden.
2. Interne Speicherung der Fehlermeldung und Abfrage von Teilen der Fehlermeldung zur weiteren Bearbeitung und Anzeige in einer Oberfläche. Hierzu sind können die restlichen in diesem Kapitel beschriebenen Funktionen verwendet werden.

3.64.1 Fehlermeldung als Zeichenkette auslesen kernelv_get_error()

Prototyp

```
KERNELV_RETURN    kernelv_get_error (unsigned long* errorId,
                                     char* messageString,
                                     unsigned long maxStringLength,
                                     unsigned long* returnLength);
```

Beschreibung

Fehlermeldungen der Simulations-CNC auslesen: Es werden die Fehlermeldungen sämtlicher Kanäle ausgelesen. Da während eines CNC-Zyklus mehrere Fehlermeldungen auftreten können, ist die Funktion zyklisch aufzurufen, bis die Funktion die Error-Id 0 zurückgibt.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebende Zeichenkette ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben. In Parameter returnLength steht in diesem Fall die zur Rückgabe der Zeichenkette benötigte Größe in Bytes.

Dem Parameter "errorId" wird in jedem Fall ein Wert zugewiesen.

Bei der gleichzeitigen Verwendung von dieser Funktion mit kernelv_read_error() ist zu beachten, dass diese Funktion intern ebenfalls kernelv_read_error() aufruft. Um bei Verwendung der Funktion kernelv_read_error() einen Fehlerstring zu lesen, sollte die Funktion kernelv_get_error_message_string() verwendet werden.

Parameter

Name	Typ	Bedeutung
errorId	unsigned long*	Zeiger auf Fehlermeldungs-ID.
messageString	char*	Zeiger auf Zeichenkette für Fehlermeldungsstring. Der Speicher ist vom Aufrufer bereitzustellen.
maxStringLength	unsigned long*	Größe des Speichers für den Fehlermeldungsstring. Wenn der von der Simulations-CNC erzeugte Fehlermeldungsstring länger ist als der bereitgestellte Speicher, wird kein String zurückgegeben.
returnLength	unsigned long*	Größe des erwarteten Speichers.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter returnSize wird die Anzahl der benötigten Bytes zurückgegeben.

3.64.2 Allgemeine Informationen zu Fehlermeldungen

Um spezifische Informationen zu einer aufgetretenen Fehlermeldung abzufragen ist wie folgt vorzugehen:

Aufruf der Funktion kernelv_read_error(). Hiermit wird geprüft, ob eine Fehlermeldung vorliegt und diese gegebenenfalls zur weiteren Auswertung DLL-intern zwischengespeichert wird.

Falls eine Fehlermeldung vorliegt, können zu dieser Fehlermeldung weitere Informationen abgefragt werden.

3.64.2.1 kernelv_read_error()

Prototyp

```
KERNELV_RETURN      kernelv_read_error (void);
```

Beschreibung

Prüfen ob eine Fehlermeldung der Simulations-CNC vorliegt und diese intern zwischenspeichern. Es werden die Fehlermeldungen sämtlicher Kanäle geprüft.

Falls eine Fehlermeldung vorliegt, können mit weiteren Funktionen Details für diese spezifische Fehlermeldung abgefragt werden. Durch einen erneuten Aufruf dieser Funktion wird gegebenenfalls eine neue Fehlermeldung ausgelesen. Die Fehlermeldungseigenschaften einer vorher anstehenden Fehlermeldung können dann nicht mehr abgefragt werden.

Da während eines CNC-Zyklus mehrere Fehlermeldungen auftreten können, ist die Funktion aufzurufen, bis sie RET_FINISHED zurückgibt.

Parameter

-

Rückgabewerte

Name	Wert	Bedeutung
RET_FINISHED	0	Es wurde keine Fehlermeldung gelesen.
RET_BUSY	1	Eine Fehlermeldung wurde gelesen.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.64.2.2 kernelv_get_error_id()

Prototyp

```
unsigned long int        kernelv_get_error_id(void);
```

Beschreibung

Liefert die Fehlernummer einer zuvor mit kernelv_read_error() gelesenen Fehlermeldung zurück. Falls kernelv_read_error() nicht aufgerufen wurde bzw. keine Fehlermeldung vorliegt, wird der Wert 0 zurückgegeben.

Parameter

-

Rückgabewerte

Typ: unsigned long int

0 falls keine Fehlermeldung vorliegt, andernfalls die Fehlernummer.

Die einzelnen Fehlernummern sind in der Diagnoseanleitung der CNC ([DIAG]) beschrieben.

3.64.2.3 kernelv_get_error_reaction()

Prototyp

signed short int

kernelv_get_error_reaction(void);

Beschreibung

Liefert die Fehlerreaktionsklasse einer zuvor mit kernelv_read_error() gelesenen Fehlermeldung zurück. Falls keine Fehlermeldung gelesen wurde bzw. keine Fehlermeldung vorliegt, wird der Wert -1 zurückgegeben.

Parameter

-

Rückgabewerte

Typ: signed short int

-1 falls keine Fehlermeldung vorliegt, andernfalls die Fehlerreaktionsklasse (siehe auch [DIAG]).

Fehlerreaktionsklasse	interne Fehlerreaktion
1	**Keine Reaktion.** Nur möglich bei Warnung (Fehlerklasse 1).
2	**Abbruch der NC-Programmbearbeitung und Übergang in Fehlerzustand.** Tritt ein Fehler im Bereich der NC-Satzaufbereitung auf, so werden vom Interpolator die bereits aufbereiteten NC-Sätze fertig bearbeitet. In diesem Fall ist die Zeit zwischen dem Auftreten des Fehlers und dem Stillstand der Maschine von der Art und der Anzahl der gepufferten NC-Sätze abhängig. Die fehlermeldende BF geht in einen Fehlerzustand.
3	**Abbruch der Auftragsbearbeitung und Übergang in Normalzustand.** BF, die Dienste für andere BF erbringen (Server), wie ACHSVERWALTUNG, DATEIVERWALTUNG, HANDBETRIEB, ..., brechen nach einer Fehlermeldung die Auftragsbearbeitung ab und gehen in den Normalzustand zurück.
4	**Bewegungsstopp (Feedhold) für die gesamte Achsgruppe und Übergang in Fehlerzustand**.
5	**Abrupter Achsstopp für fehlerhafte Achse, Feedhold für übrige Achsen der Achsgruppe und Übergang in einen Fehlerzustand.**
6	**Abrupter Achsstopp für alle Achsen und Übergang in einen Fehlerzustand.** Lageregelung geht in Fehlerzustand.
7	**Geregelter Achsstopp für fehlerhafte Achse, Feedhold für übrige Achsen der Achsgruppe und Übergang in einen Fehlerzustand**.
8	**Gesteuerter Achsstopp für fehlerhafte Achse, Feedhold für übrige Achsen der Achsgruppe und Übergang in einen Fehlerzustand.** Der Lageregelkreis der fehlerhaften Achse wird geöffnet.

3.64.2.4 kernelv_get_error_severity()

Prototyp

signed short int

kernelv_get_error_severity(void);

Beschreibung

Liefert die Fehlerbehebungsklasse einer zuvor mit kernelv_read_error() gelesenen Fehlermeldung zurück. Falls keine Fehlermeldung gelesen wurde bzw. keine Fehlermeldung vorliegt, wird der Wert -1 zurückgegeben.

Parameter

-

Rückgabewerte

Typ: unsigned short int

-1 falls keine Fehlermeldung vorliegt, andernfalls die Fehlerbehebungsklasse (siehe auch [DIAG]).

Fehlerbehebungsklasse	**interne Fehlerbehebung**
0	Fehlermeldung dient als Warnung, es erfolgt **automatisch** eine **interne Fehlerbehebung.**
2	Kompletter **Reset** des NC-Kanals notwendig.
5	Kompletter **Reset** des NC-Kanals notwendig.
6	Neuer **Hochlauf** des CNC-Kerns notwendig.
7	Neuer **Hochlauf** des gesamten NC-Kerns nach Ausschalten notwendig.

3.64.2.5 kernelv_get_error_channel()

Prototyp

signed short int

kernelv_get_error_channel(void);

Beschreibung

Liefert die Kanalnummer des Kanals, in dem eine zuvor mit kernelv_read_error() gelesene Fehlermeldung aufgetreten ist, zurück. Falls keine Fehlermeldung gelesen wurde bzw. keine Fehlermeldung vorliegt, wird der Wert -1 zurückgegeben.

Parameter

-

Rückgabewerte

Typ: unsigned short int

0 falls keine Fehlermeldung vorliegt, andernfalls die Kanalnummer.

3.64.2.6 kernelv_get_error_message_string()

Prototyp

```
KERNELV_RETURN      kernelv_get_error_message_string(char * string,
                                                     unsigned long int *length);
```

Beschreibung

Liefert die formatierte Zeichenkette einer zuvor mit kernelv_read_error() gelesenen Fehlermeldung als Zeichenkette zurück. Die zurückgegebene Zeichenkette ist identisch mit der von der Funktion kernelv_get_error() zurückgegebenen Zeichenkette.

Beim Aufruf ist im Parameter 'string' ein Zeiger auf den Speicher für die zurückzugebende Zeichenkette zu übergeben, in 'length' die Größe dieses Speicherbereiches. Falls der bereitgestellte Speicher ausreichend für die Rückgabe der Zeichenkette ist, wird in 'length' die Länge der zurückgegebenen Zeichenkette zurückgegeben.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebende Zeichenkette ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben. In Parameter 'length' steht in diesem Fall die zur Rückgabe der Zeichenkette benötigte Größe in Bytes.

Parameter

Name	Typ	Bedeutung
string	char*	Zeiger auf Zeichenkette für Fehlermeldungsstring. Der Speicher ist vom Aufrufer bereitzustellen.
length	unsigned long*	Größe des Speichers für den Fehlermeldungsstring. Wenn der von der Simulations-CNC erzeugte Fehlermeldungsstring länger ist als der bereitgestellte Speicher, wird kein String zurückgeben. Zurückgegeben wird entweder die Länge der zurückgegebenen Zeichenkette oder die zur Rückgabe benötigte Größe des Speicherbereiches.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter 'returnSize' wird die Anzahl der benötigten Bytes zurückgegeben.

3.64.2.7 kernelv_get_error_id_text()

Prototyp

```
KERNELV_RETURN  kernelv_get_error_message_string(char * string,
                                                 unsigned long int *length);
```

Beschreibung

Liefert den zur aktuellen Fehlernummer gehörenden Fehlermeldungstext zurück einer zuvor mit kernelv_read_error() gelesenen Fehlermeldung. Beim Aufruf ist im Parameter 'string' ein Zeiger auf den Speicher für die zurückzugebende Zeichenkette zu übergeben, in 'length' Größe dieses Speicherbereiches. Falls der bereitgestellte Speicher ausreichend für die Rückgabe der Zeichenkette ist, wird in 'length' die Länge der zurückgegebenen Zeichenkette zurückgegeben.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebende Zeichenkette ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben. In Parameter 'length' steht in diesem Fall die zur Rückgabe der Zeichenkette benötigte Größe in Bytes.

Parameter

Name	Typ	Bedeutung
string	char*	Zeiger auf Zeichenkette für Fehlermeldungsstring. Der Speicher ist vom Aufrufer bereitzustellen.
length	unsigned long*	Größe des Speichers für den Fehlermeldungsstring. Wenn der von der Simulations-CNC erzeugte Fehlermeldungsstring länger ist als der bereitgestellte Speicher, wird kein String zurückgeben. Zurückgegeben wird entweder die Länge der zurückgegebenen Zeichenkette oder die zur Rückgabe benötigte Größe des Speicherbereiches.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter 'returnSize' wird die Anzahl der benötigten Bytes zurückgegeben.

3.64.2.8 kernelv_get_error_message_values()

Prototyp

```
KERNELV_RETURN  kernelv_get_error_message_string (KERNELV_ERROR_VALUE * p_values,
                                                  unsigned long int *length);
```

Beschreibung

Liefert die in einer Fehlermeldung ausgegebenen Werte zurück.

Vor Anwendung dieser Funktion muss durch Aufruf der Funktion kernelv_read_error() geprüft werden, ob ein Fehler vorliegt.

Die Werte werden in einem Array von Strukturen des Typs KERNELV_ERROR_VALUE zurückgegeben. Die Arraygröße beträgt KERNELV_ERROR_VALUE_COUNT. Im Aufrufparameter 'length' ist die Größe des Speicherbereiches für die Fehlermeldungswerte anzugeben.

Falls das zurückzugebende Array nicht in den bereitgestellten Speicher passt, wird der Rückgabewert ERR_CNC_RET_MEMORY ausgegeben und 'length' enthält die zur Rückgabe benötigte Speichergröße in Bytes.

Parameter

Name	Typ	Bedeutung
p_values	KERNELV_ERROR_VALUE*	Zeiger auf Zeichenkette für Fehlermeldungswerte. Der Speicher ist vom Aufrufer bereitzustellen.
length	unsigned long*	Größe des Speichers für die Fehlermeldungswerte. Wenn der bereitgestellte Speicher nicht ausreicht, wird die benötigte Speichergröße in Bytes zurückgegeben, andernfalls die Anzahl der zurückgelieferten Bytes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter 'length' wird die Anzahl der benötigten Bytes zurückgegeben.

3.64.2.9 kernelv_get_error_cycle_time()

Prototyp

```
unsigned long long int    kernelv_get_error_cycle_time(void);
```

Beschreibung

Liefert den CNC Zyklus in dem eine zuvor mit kernelv_read_error() gelesenen Fehlermeldung aufgetreten ist zurück. Falls keine Fehlermeldung gelesen wurde bzw. keine Fehlermeldung vorliegt, wird der Wert 0 zurückgegeben.

Parameter

-

Rückgabewerte

Typ: unsigned long int

0 falls keine Fehlermeldung vorliegt, andernfalls den CNC Zyklus, in dem die Fehlermeldung aufgetreten ist.

3.64.3 Fehlermeldungen die durch NC-Programme verursacht werden

Bei Fehlermeldungen, die durch ein NC-Programm verursacht wurden, besteht die Möglichkeit, zusätzliche Informationen zum NC-Programm in dem der Fehler aufgetreten ist, abzufragen. Diese Informationen können dann verwendet werden, um die fehlerhafte Stelle im NC-Programm näher zu lokalisieren.

Hierzu ist zunächst zu prüfen, ob die NC-Programm spezifischen Informationen vorliegen, dies erfolgt durch Aufruf der Funktion kernelv_is_program_err(). Wenn diese Funktion den Wert 1 zurückgibt, wurde die aktuelle Fehlermeldung durch ein NC-Programm verursacht, und weitere Informationen über die fehlerhafte Stelle im NC-Programm können durch Aufruf der folgenden Funktionen abgerufen werden.

3.64.3.1 kernelv_error_is_program_error()

Prototyp

unsigned char

kernelv_error_is_program_error(void);

Beschreibung

Gibt an, ob der aktuelle Fehler durch ein CNC-Programm verursacht wurde.

Parameter

-

Rückgabewerte

Wert	Bedeutung
0	Die aktuelle Fehlermeldung wurde nicht durch ein NC-Programm verursacht oder es liegt kein Fehler vor.
1	Die aktuelle Fehlermeldung wurde durch ein NC-Programm verursacht.

3.64.3.2 kernelv_program_error_get_path

Prototyp

```
KERNELV_RETURN      kernelv_program_error_get_path(char *return_string,
                                                   unsigned long * return_length);
```

Beschreibung

Liefert den Programmpfad der verwendet wurde um das aktive Programm zu starten zurück.

Wenn das Programm durch die absolute Angabe eines Dateinamens gestartet wurde wird ein Leerstring zurückgegeben, falls das gerade aktive Programm ein Handsatz ist, wird "-" (Ohne Anführungszeichen) zurückgegeben.

Zurückgegeben wird eine nullterminierte Zeichenkette, in return_length wird die Anzahl der zurückgelieferten Bytes zurückgegeben, d. h. die terminierende Null wird mitgezählt.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebende Zeichenkette ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben. In Parameter returnLength steht in diesem Fall die zur Rückgabe der Zeichenkette benötigte Größe in Bytes.

Parameter

Name	Typ	Bedeutung
return_string	char*	Zeiger auf Zeichenkette für den Programmpfad. Der Speicher ist vom Aufrufer bereitzustellen.
return_length	unsigned long*	Größe des Speichers für den Rückgabewert. Wenn die zurückzugebende Zeichenkette länger ist als der bereitgestellte Speicher, wird nichts zurückgeben. Zurückgegeben wird entweder die Länge der zurückgegebenen Zeichenkette oder die zur Rückgabe benötigte Größe des Speicherbereiches.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter length wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. Es liegen keine NC-programmspezifischen Daten zu einem CNC-Fehler vor.

3.64.3.3 kernelv_program_error_get_program_name

Prototyp

```
KERNELV_RETURN      kernelv_program_error_get_program_name
                                    (char *return_string,
                                     unsigned long * returnLength);
```

Beschreibung

Liefert den Programmnamen des aktiven NC-Programms zurück.

Der Programmname wird am Anfang des Hauptprogrammes durch ein "%"-Zeichen angegeben, weitere Informationen siehe [PROG]. Falls für das NC- Programm kein Programmname angegeben wurde, wird ein Leerstring zurückgegeben.

Zurückgegeben wird eine nullterminierte Zeichenkette, in return_length wird die Anzahl der zurückgelieferten Bytes zurückgegeben, d. h. die terminierende Null wird mitgezählt.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebende Zeichenkette ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben. In Parameter returnLength steht in diesem Fall die zur Rückgabe der Zeichenkette benötigte Größe in Bytes.

Parameter

Name	Typ	Bedeutung
return_string	char*	Zeiger auf Zeichenkette für den Programmname. Der Speicher ist vom Aufrufer bereitzustellen.
return_length	unsigned long*	Größe des Speichers für den Rückgabewert. Wenn die zurückzugebende Zeichenkette länger ist als der bereitgestellte Speicher, wird nichts zurückgeben. Zurückgegeben wird entweder die Länge der zurückgegebenen Zeichenkette oder die zur Rückgabe benötigte Größe des Speicherbereiches.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter length wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar.

3.64.3.4 kernelv_program_error_get_file_name

Prototyp

```
KERNELV_RETURN    kernelv_program_error_get_file_name(char *return_string,
                                                      unsigned long * returnLength);
```

Beschreibung

Liefert den Dateinamen des aktiven NC-Programms zurück.

Falls das Programm durch Angabe eines absoluten Programmnamens gestartet wurde, wird der komplette Programmname zurückgegeben.

Falls die CNC-Steuerung das NC-Programm unter Benutzung eines Suchpfades öffnete, wird von dieser Funktion der beim Programmstart angegebene Dateiname zurückgegeben, der verwendete Suchpfad kann mit der Funktion kernelv_program_error_get_path() abgefragt werden.

Die Funktion liefert die Anzahl der zurückgelieferten Bytes in returnLength zurück, d. h. die den String terminierende Null wird mitgezählt.

Falls der vom Aufrufer bereitgestellte Speicher zu klein für die zurückzugebende Zeichenkette ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben. In Parameter returnLength steht in diesem Fall die zur Rückgabe der Zeichenkette benötigte Größe in Bytes.

Parameter

Name	Typ	Bedeutung
return_string	char*	Zeiger auf Zeichenkette für den Programmpfad. Der Speicher ist vom Aufrufer bereitzustellen.
return_length	unsigned long*	Größe des Speichers für den Rückgabewert. Wenn die zurückzugebende Zeichenkette länger ist als der bereitgestellte Speicher, wird nichts zurückgeben. Zurückgegeben wird entweder die Länge der zurückgegebenen Zeichenkette oder die zur Rückgabe benötigte Größe des Speicherbereiches.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. Im Parameter length wird die Anzahl der benötigten Bytes zurückgegeben.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. Es liegen keine NC-programmspezifischen Daten zu einem CNC-Fehler vor.

3.64.3.5 kernelv_program_error_get_fileoffset

Prototyp

KERNELV_RETURN kernelv_program_error_get_fileoffset(unsigned long * fileoffset);

Beschreibung

Liefert den Fileoffset der fehlerhaften Stelle in der Programmdatei zurück.

Parameter

Zeiger auf Speicherort für den Fileoffset.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. Es liegen keine NC-programmspezifischen Daten zu einem CNC-Fehler vor.

3.64.3.6 kernelv_program_error_get_lineoffset

Prototyp

KERNELV_RETURN kernelv_program_error_get_linenoffset(unsigned short int*);

Beschreibung

Liefert Offset der fehlerhaften Stelle innerhalb der NC-Zeile zurück

Parameter

Zeiger auf Speicherort für Zeilenoffset.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. Es liegen keine NC-programmspezifischen Daten zu einem CNC-Fehler vor.

3.64.3.7 kernelv_program_error_get_tokenoffet

Prototyp

KERNELV_RETURN kernelv_program_error_get_tokenoffset(unsigned short int*);

Beschreibung

Liefert Offset der fehlerhaften Stelle innerhalb des geparsten NC_Tokens zurück.

Parameter

Zeiger auf Speicherort für Tokenoffset.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. Es liegen keine NC-programmspezifischen Daten zu einem CNC-Fehler vor.

3.64.3.8 kernelv_program_error_get_linenumber

Prototyp

KERNELV_RETURN kernelv_program_error_get_linenumber(signed long * linenumber);

Beschreibung

Liefert die programmierte NC-Zeilennummer der fehlerhaften Stelle in der Programmdatei zurück.

Parameter

Zeiger auf Speicherort für Zeilennumer

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. Es liegen keine NC-programmspezifischen Daten zu einem CNC-Fehler vor.

3.65 Koordinatensysteme und Versätze

Innerhalb eines NC-Programmes kann das Programmierkoordinatensystem durch NC-Befehle an die jeweiligen Anforderungen angepasst werden, indem es z.B. gedreht oder verschoben wird. Die hierzu verwendeten NC-Befehle lauten unter anderem #CS DEF/#CS ON oder #ACS DEF/#ACS ON. Einzelheiten hierzu können der Programmieranleitung Kapitel 'Koordinatenssysteme' entnommen werden.

Durch die wiederholte Anwendung dieser Befehle können Koordinatensysteme auch verkettet werden. Das resultierende Koordinatensystem wird dabei durch einen Stapel von unterlagerten Koordinatensystemen gebildet.

Durch weitere NC-Befehle (#CS ADD*) können diesem Koordinatensystem-Stapel weitere Koordinatensysteme hinzugefügt werden, ohne dass diese wirksam werden, also das resultierende Koordinatensystem beeinflussen. Durch den Befehl #CS SELECT* kann das zu verwendende Koordinatensystem festgelegt werden.

Innerhalb jedes Koordinatensystems können für jede im Kanal vorhandene Achse Versätze definiert werden, z.B. durch die Befehle G54 … G59 (Nullpunktverschiebung) oder G92 (Bezugspunktverschiebung).

Durch die bereitgestellten API-Funktionen können die folgenden Informationen ausgelesen werden:

Anzahl der definierten Koordinatensysteme

Index des aktiven Koordinatensystems im Koordinatensystem-Stack

Informationen über ein Koordinatensystems an einem bestimmten Index im Stack

Informationen über achsweisen Versätze innerhalb eines Koordinatensystems im Stack

*Dieser NC-Befehl ist nicht in allen Versionen verfügbar.

3.65.1 kernelv_ch_get_cs_name()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_cs_name(unsigned long int chanIndex,
                                         unsigned short csIndex,
                                         char *name,
                                         unsigned long int bufferSize,
                                         unsigned long int *retBytes);
```

Beschreibung

Liefert die im NC-Programm definierte Bezeichnung des Koordinatensystems zurück.

Falls an der durch csIndex definierten Stelle im Koordinatensystem-Stack kein Koordinatensystem definiert ist, wird eine leere Zeichenkette zurückgegeben und der Rückgabewert der Funktion ist ERR_CNC_NO_DATA.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
csIndex	unsigned short	Index des Koordinatensystems im Koordinatensystem-Stack.
name	char*	Zeiger auf den Speicherplatz für den Koordinatensystemnamen.
nameLength	unsigned long	Länge des Speicherbereiches für den Koordinatensystemnamen.
returnLength	unsigned long*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Es wird die Anzahl der Zeichen des Koordinatensystemnamens + 1 zurückgegeben. Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.

ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. An der durch den Aufrufparameter csIndex angegebenen Stelle im Koordinatensystem-Stack steht kein definiertes Koordinatensystem. Es wird eine leere Zeichenkette zurückgegeben.
ERR_INVALID_PARAMETER	-30	Es wurde ein ungültiger Parameter übergeben: Der in Parameter csIndex übergebene Koordinatensystemindex ist größer als der maximal mögliche Index im Koordinatensystem-Stack.

3.65.2 kernelv_ch_get_cs_rot_matrix()

Prototyp

```
KERNELV_RETURN  kernelv_ch_get_cs_rot_matrix(unsigned long int chanIndex,
                                             unsigned short csIndex,
                                             double *matrix,
                                             unsigned long int bufferSize,
                                             unsigned long int *retBytes);
```

Beschreibung

Liefert die Rotationsmatrix des durch den Parameter csIndex angegebenen Koordinatensystems zurück.

Es wird eine 3x3 Drehmatrix zurückgegeben, mit der das Koordinatensystem aus dem unterlagerten Koordinatensystem erzeugt werden kann.

Es wird die Rotationsmatrix zurückgegeben, die z.B. durch die Parameter **ϕ1, ϕ2 ϕ3** des folgenden NC-Befehls erzeugt wird:

#CS DEF [CS1] [*<v1>*,*<v2>*,*<v3>*,**<ϕ1>,<ϕ2>,<ϕ3>]**

Falls an dem angegebenen Index im Koordinatensystems-Stack kein Koordinatensystem definiert ist, wird die Einheitsmatrix zurückgegeben und der Rückgabewert der Funktion ist ERR_CNC_NO_DATA.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
csIndex	unsigned short	Index des Koordinatensystems im Koordinatensystem-Stack.
matrix	double*	Zeiger auf den Speicherplatz für die Rotationsmatrix.
bufferSize	unsigned long	Länge des Speicherbereiches für die Rotationsmatrix muss mindestens 3 x 3 x sizeof(double) sein.
returnLength	unsigned long*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1

ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.
ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. An der durch den Aufrufparameter csIndex angegebenen Stelle im Koordinatensystem-Stack steht kein definiertes Koordinatensystem. Es wird die Einheitsmatrix zurückgegeben.
ERR_INVALID_PARAMETER	-30	Es wurde ein ungültiger Parameter übergeben: Der in Parameter csIndex übergebene Koordinatensystemindex ist größer als der maximal mögliche Index im Koordinatensystem-Stack.

3.65.3 kernelv_ch_get_cs_shift_vector()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_cs_shift_vector(unsigned long int chanIndex,
                                                 unsigned short csIndex,
                                                 double *vector,
                                                 unsigned long int bufferSize,
                                                 unsigned long int *retBytes);
```

Beschreibung

Liefert die Verschiebung des Ursprungs des durch den Parameter csIndex angegebenen Koordinatensystems zurück.

Es wird ein Vektor mit drei Elementen zurückgegeben, der die Verschiebung des Koordinatensystem-Ursprungs zum Ursprung des unterlagerten Koordinatensystems angibt.

Es wird der Verschiebungsvektor zurückgegeben, der z.B. durch die Parameter **v1, v2, v3** des folgenden NC-Befehls erzeugt wird:

#CS DEF [CS1] [<v1>,<v2>,<v3>*,<ϕ1>,<ϕ2>,<ϕ3>* **]**

Falls an dem angegebenen Index im Koordinatensystems-Stack kein Koordinatensystem definiert ist, wird ein Nullvektor zurückgegeben und der Rückgabewert der Funktion ist ERR_CNC_NO_DATA.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
csIndex	unsigned short	Index des Koordinatensystems im Koordinatensystem-Stack.
vector	double*	Zeiger auf den Speicherplatz für den Verschiebungsvektor.
bufferSize	unsigned long	Länge des Speicherbereiches für die Rotationsmatrix muss mindestens 3 x sizeof(double) sein.
returnLength	unsigned long*	Zeiger auf den Wert in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.
ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. An der durch den Aufrufparameter csIndex angegebenen Stelle im Koordinatensystem-Stack steht kein definiertes Koordinatensystem. Es wird ein Nullvektor zurückgegeben.
ERR_INVALID_PARAMETER	-30	Es wurde ein ungültiger Parameter übergeben: Der in Parameter csIndex übergebene Koordinatensystemindex ist größer als der maximal mögliche Index im Koordinatensystem-Stack.

3.65.4 kernelv_ch_get_cs_count()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_cs_count(unsigned long int chanIndex,
                                          unsigned short *count);
```

Beschreibung

Liefert die Anzahl der definierten Koordinatensysteme zurück.

Auch wenn im NC-Programm kein Koordinatensystem definiert ist, existiert immer ein Grundkoordinatensystem auf dem Koordinatensystem-Stacklevel 0, dessen Drehmatrix die Einheitsmatrix und dessen Verschiebungsvektor der Nullvektor ist.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
count	unsigned short*	Zeiger auf den Speicherplatz für die Koordinatensystem-Anzahl.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.

3.65.5 kernelv_ch_get_active_cs_index()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_cs_count(unsigned long int chanIndex,
                                          unsigned short *csIndex);
```

Beschreibung

Liefert den Index des aktiven Koordinatensystems zurück.

Auch wenn im NC-Programm kein Koordinatensystem definiert ist, existiert immer ein Grundkoordinatensystem auf dem Koordinatensystem-Stacklevel 0, dessen Drehmatrix die Einheitsmatrix und dessen Verschiebungsvektor der Nullvektor ist.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
csIndex	unsigned short*	Zeiger auf den Speicherplatz für den Koordinatensystem-Index.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.

3.65.6 kernelv_ch_axis_get_offsets()

Prototyp

```
KERNELV_RETURN    kernelv_ch_axis_get_offsets(unsigned long int chanIndex,
                                              unsigned long int axisIndex,
                                              unsigned short int csIndex,
                                              signed long int *offsets,
                                              unsigned long int bufferSize,
                                              unsigned long int *retBytes);
```

Beschreibung

Liefert die achsspezifischen Verschiebungen in dem durch den Parameter csIndex angegebenen Koordinatensystems zurück.

Der Parameter axisIndex kennzeichnet dabei den Index der Achse innerhalb des durch ChanIndex angegebenen NC-Kanals.

Es wird ein Vektor mit acht Elementen zurückgegeben, in dem für die durch axisIndex definierte Achse die durch die unterschiedlichen NC-Befehle eingeführten Offsets aufgeführt sind.

Die Zuordnung des Index innerhalb des Vektors zu den unterschiedlichen Verschiebungstypen ist durch die Enumeration KERNELV_AXIS_OFFSET_TYPES gegeben.

Parameter

Name	Typ	Bedeutung

chanIndex	unsigned long	Kanalindex des Kanals.
axisIndex	unsigned long	Index der Achse innerhalb des NC-Kanals.
csIndex	unsigned short	Index des Koordinatensystems im Koordinatensystem-Stack.
offsets	signed long*	Zeiger auf den Speicherplatz für den Offsetvektor.
nameLength	unsigned long	Länge des Speicherbereiches für die Rotationsmatrix muss mindestens 8 x sizeof (signed long int) sein.
returnLength	unsigned long*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.
ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar. An der durch den Aufrufparameter csIndex angegebenen Stelle im Koordinatensystem-Stack steht kein definiertes Koordinatensystem. Es wird ein Nullvektor zurückgegeben.
ERR_INVALID_PARAMETER	-30	Es wurde ein ungültiger Parameter übergeben: Der in Parameter csIndex übergebene Koordinatensystemindex ist größer als der maximal mögliche Index im Koordinatensystem-Stack.

3.65.7 kernelv_ch_get_total_cs_rot_matrix()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_total_cs_rot_matrix(unsigned long int chanIndex,
                                                     double *matrix,
                                                     unsigned long int bufferSize,
                                                     unsigned long int *retBytes);
```

Beschreibung

Liefert die durch die Verkettung aller aktiven Koordinatensysteme entstandene Rotationsmatrix zurück.

Es wird die Rotationsmatrix zurückgegeben, die z.B. durch die Parameter **ϕ1, ϕ2 ϕ3** des folgenden NC-Befehls erzeugt wird:

#CS DEF [CS1] [*<v1>*,*<v2>*,*<v3>*,**<ϕ1>,<ϕ2>,<ϕ3>]**

Falls kein Koordinatensystem aktiv ist, wird die Einheitsmatrix zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
matrix	double*	Zeiger auf den Speicherplatz für die Rotationsmatrix.
bufferSize	unsigned long	Länge des Speicherbereiches für die Rotationsmatrix muss mindestens 3 x 3 x sizeof(double) sein.
returnLength	unsigned long*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.

3.65.8 kernelv_ch_get_total_cs_offset

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_total_cs_offset(unsigned long int chanIndex,
                                                 double *vector,
                                                 unsigned long int bufferSize,
                                                 unsigned long int *retBytes);
```

Beschreibung

Liefert die Verschiebung des Ursprungs des durch die Verkettung aller aktiven Koordinatensysteme entstandenen resultierenden Koordinatensystems zurück.

Es wird ein Vektor mit drei Elementen zurückgegeben, der die Verschiebung des Koordinatensystem-Ursprungs zum Ursprung des Basiskoordinatensystems angibt.

Es wird der Verschiebungsvektor zurückgegeben, der z.B. durch die Parameter **v1, v2, v3** des folgenden NC-Befehls erzeugt wird:

#CS DEF [CS1] [<v1>,<v2>,<v3>,*<ϕ1>,<ϕ2>,<ϕ3>* **]**

Falls kein Koordinatensystem aktiv ist, wird ein Nullvektor zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
vector	double*	Zeiger auf den Speicherplatz für den Verschiebungsvektor.
bufferSize	unsigned long	Länge des Speicherbereiches für die Rotationsmatrix muss mindestens 3 x sizeof(double) sein.
returnLength	unsigned long*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll.

		Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.

3.65.9 kernelv_ch_get_total_cs_def()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_total_cs_def(unsigned long int chanIndex,
                                              double *vector,
                                              unsigned long int bufferSize,
                                              unsigned long int *retBytes);
```

Beschreibung

Liefert die Verschiebung und Drehwinkel des durch die Verkettung aller aktiven Koordinatensysteme entstandenen resultierenden Koordinatensystems zurück.

Es wird ein Vektor mit sechs Elementen zurückgegeben, die ersten drei Elemente enthalten die Verschiebung Koordinatensystem-Ursprungs zum Ursprung des Basiskoordinatensystems, die drei folgenden Vektorelemente enthalten die Drehwinkel in Grad, die benötigt werden um das Koordinatensystem aus dem Basiskoordinatensystem zu erzeugen. Die Reihenfolge mit der die Drehungen ausgeführt werden beträgt, analog zur Dokumentation des #CS-Befehls *ϕ3, . ϕ2,* ϕ1 in dieser Reihenfolge.

#CS DEF [CS1] [<v1>,<v2>,<v3>,<ϕ1>,<ϕ2>,<ϕ3>]

Index	Bedeutung im #CS-Befehl
0	<v1>
1	<v2>
2	<v3>
3	<ϕ1>
4	<ϕ2>
5	<ϕ3>

Falls kein Koordinatensystem aktiv ist, wird ein Nullvektor zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
vector	double*	Zeiger auf den Speicherplatz für den Verschiebungsvektor.
bufferSize	unsigned long	Länge des Speicherbereiches für die Rotationsmatrix muss mindestens 3 x sizeof(double) sein.
returnLength	unsigned long*	Zeiger auf den Wert, in den die tatsächlich zurückgegebene Anzahl Bytes geschrieben werden soll. Falls der übergebene Speicher zu klein für die Rückgabe des Wertes ist, wird der Rückgabewert ERR_CNC_RET_MEMORY zurückgegeben und es wird über diesen Parameter die benötigte Speichergröße zurückgegeben.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher. In diesem Fall wird über returnLength die zur Rückgabe benötigte Mindestgröße zurückgegeben.

3.65.10 kernelv_ch_get_coord_sys_active()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_coord_sys_active(unsigned long int chanIndex,
                                                  unsigned char *active);
```

Beschreibung

Liefert die Nummer der aktuell aktiven kinematischen Transformation zurück. Wenn keine kinematische Transformation aktiv ist, wird Null zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
active	unsigned short*	Zeiger auf den Speicherplatz für Rückgabewert.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.66 Kinematische Transformationen

3.66.1 kernelv_ch_get_kin_trafo_active()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_kin_trafo_active(unsigned long int chanIndex,
                                                  unsigned char *active);
```

Beschreibung

Liefert zurück, ob im angegebenen Kanal eine Koordinatensystem aktiv ist oder nicht.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
active	unsigned char*	Zeiger auf den Speicherplatz für Rückgabewert.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.66.2 kernelv_ch_get_active_kin_id()

Prototyp

```
KERNELV_RETURN    kernelv_ch_get_active_kin_id(unsigned long int chanIndex,
                                               unsigned short *kin_id);
```

Beschreibung

Liefert die Nummer der aktuell aktiven kinematischen Transformation zurück. Wenn keine kinematische Transformation aktiv ist, wird Null zurückgegeben.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long	Kanalindex des Kanals.
kin_id	unsigned short*	Zeiger auf den Speicherplatz für Rückgabewert.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.67 Externe Messhardware

Defaultmässig werden in der kernelv-Dll Messfahrten mit der für die jeweilge Achse parametrierten Messsimulation simuliert.

Bei der Messvariante "Messen mit externer Messhardware" wird für den Anwender die auf der HLI liegende Schnittstelle zur externen Messhardware über API-Funktionen zugänglich gemacht. Damit hat der Anwender die Möglichkeit eventuell in der SPS der Echtzeitsteuerung implementierte Messverfahren auch mit der kernelv-Dll zu verwenden.

Die Schnittstelle zur externen Messhardware besteht aus folgenden Teilen:

Kommandoschnittstelle von der CNC zur SPS bzw. zum Anwender der Dll.

Quittierungsschnittstelle von der SPS bzw. Anwender zur CNC.

Triggerschnittstelle für das Messereignis und gegebenenfalls den Messwert von der SPS zur CNC.

Der Ablauf einer Messfahrt unter Verwendung der externen Messschnittstelle ist wie folgt:

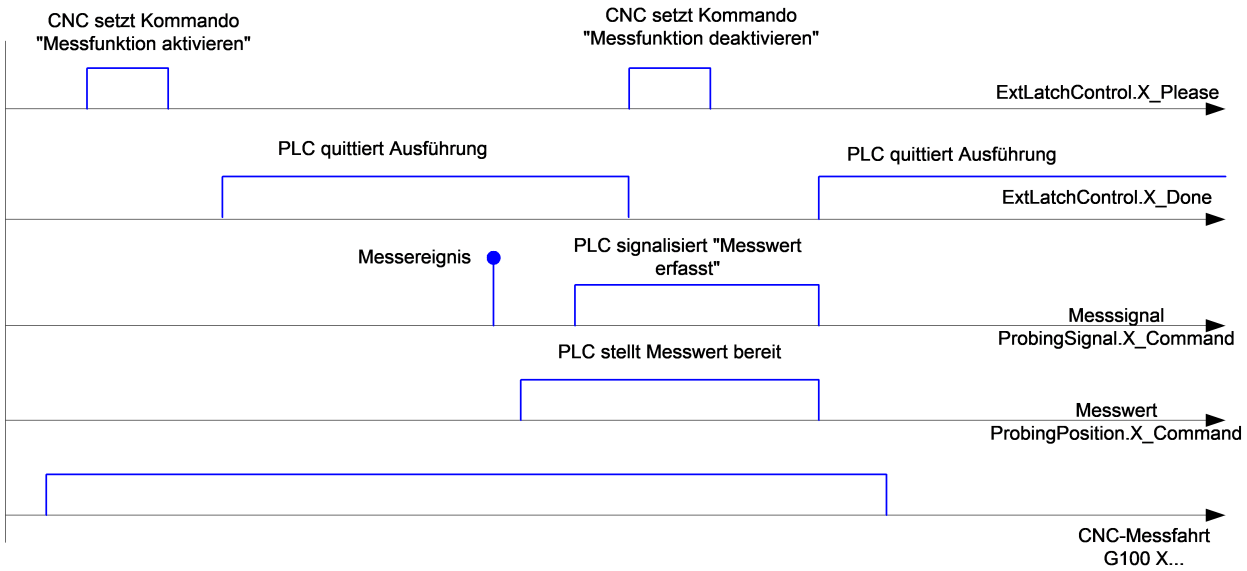


Abb. 5: Schematischer Ablauf einer Messfahrt

Das Abfragen der Kommandoschnittstelle erfolgt dabei über die Funktion **kernelv_ax_get_ext_latch_command()**, das Quittieren des Kommandos erfolgt mit **kernelv_ax_acknowledge_ext_latch_command()**, das Setzen des Latchereignisses sowie gegebenenfalls des Latch-Wertes verfolgt mit **kernelv_ax_set_ext_latch_event()** bzw. mit **kernelv_ax_set_ext_latch_event_pos()**.

Bei der Verwendung der externen Messschnittstelle muss die interne Messsimulation für die Achse durch Setzen des Achsparameters P-AXIS-00112 auf den Wert 0 deaktiviert werden.

3.67.1 kernelv_ax_get_ext_latch_command()

Prototyp

```
KERNELV_RETURN      kernelv_ch_get_active_kin_id(unsigned long int axIndex,
                                                 KERNELV_EXT_LATCH_COMMAND_DATA *data);
```

Beschreibung

Liefert den Inhalt der Kommadoschnittstelle für die externe Messhardare in der Struktur KERNELV_EXT_LATCH_COMMAND_DATA zurück.

Parameter

Name	Typ	Bedeutung
axIndex	unsigned long	Index der Achse.
data	KERNELV_EXT_LATCH_COMMAND_DATA *	Zeiger auf Struktur des Typs KERNELV_EXT_LATCH_COMMAND_DATA zur Rückgabe des Wertes.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
REST_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.

ERR_INVALID_AX	-9	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.67.2 kernelv_ax_acknowledge_ext_latch_command()

Prototyp

KERNELV_RETURN kernelv_ax_acknowledge_ext_latch_command(unsigned long int axIndex);

Beschreibung

Quittiert ein eventuell anstehendes Messkommando an die externe Messhardware. Falls ein Kommando aktiv ist und quittiert wurde, ist der Rückgabewerte der Funktion RET_FINISHED, andernfalls ist der Rückgabewert ERR_NO_DATA.

Parameter

Name	Typ	Bedeutung
axIndex	unsigned long	Index der Achse.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_INVALID_AX	-9	Der übergebene Achsindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar.

3.67.3 kernelv_ax_set_ext_latch_event_pos()

Prototyp

KERNELV_RETURN kernelv**_ax_**set_ext_latch_event_pos(unsigned long int axIndex,
signed long int DATA position);

Beschreibung

Zeigt der CNC das Auftreten des Latch-Ereignisses an und übergibt die gelatchte Position.

Die gelatchte Position ist dabei in 0,1 um bzw. 1E-4 ° zu übergeben.

Parameter

Name	Typ	Bedeutung
axIndex	unsigned long int	Index der Achse.
position	signed long int	Latchposition in 0,1 µm für translatorische Achsen bzw. $1*10^{-4}$° für Spindeln oder rotatorische Achsen.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_AX	-9	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NO_DATA	-28	Es ist kein Latch kommando aktiv.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.67.4 kernelv_ax_set_ext_latch_event()

Prototyp

```
KERNELV_RETURN      kernelv_ax_set_ext_latch_event(unsigned long int axIndex);
```

Beschreibung

Zeigt der CNC das Auftreten des Latch-Ereignisses an als gelatchter Wert wird die aktuelle Istposition der Achse übergeben.

Parameter

Name	Typ	Bedeutung
axIndex	unsigned long int	Index der Achse.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_AX	-9	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle – 1
ERR_CNC_NO_DATA	-28	Es ist kein Latch kommando aktiv.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

3.68 kernelv_ch_get_timer()

Prototyp

KERNELV_RETURN kernelv_ch_get_timer (unsigned long int chanIndex, unsigned short int timerId, unsigned long int* time);

Beschreibung

Gibt die Zeit in ms zurück, die in der Variable V.G.TIMER[<Zählernummer>] im angegebenen Kanal abgespeichert ist.

Parameter

Name	Typ	Bedeutung
chanIndex	unsigned long int	Index des Kanals
timerId	unsigned short int	Zählernummer für die Variable V.G.TIMER[]
Time	unsigned long int*	Zeiger auf den Speicherbereich, in den die Zeit in ms geschrieben wird.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INVALID_ PARAMETER	-30	Der übergebenen Parameter timerId ist ungültig. Der gültige Bereich für die timerID: 0<= timerID <= 127
ER_NULL_PARAMETER	-35	Der Zeiger auf Timer ist nicht referenziert.

3.69 kernelv_get_production_time()

Prototyp

KERNELV_RETURN kernelv_get_production_time(double* productionTime, KERNELV_PT_FILE files);

Beschreibung

Startet für jeden nicht leeren String in der Struktur KERNELV_PT_FILES ein NC-Programm im entsprechenden Kanal. Das erste Element wird im ersten Kanal gestartet, das zweite Element im zweiten Kanal usw.

Soll in einem Kanal kein Programm gestartet werden, muss der String am passenden Index 0 sein.

Es können nur Programme in Kanälen gestartet werden, die auch konfiguriert sind.

Es wird die gesamte Bearbeitungsdauer aller gestartet NC-Programme zurückgegeben.

Parameter

Name	Typ	Bedeutung
productionTime	double*	Zeiger auf den Speicherbereich, in den die Fertigungszeit in s geschrieben wird.

files	KERNELV_PT_FILES	Struktur in der die Namen der NC-Programme abgespeichert sind.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die übergebenen NC-Programme sind all erfolgreich beendet.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten. Der Wert konnte nicht geschrieben werden
ERR_NC_PROGRAM	-32	Im NC-Programm ist ein Fehler aufgetreten.
ERR_CH_ERROR_STATE	-33	Der Kanal befindet sich im Fehlerzustand.

3.70 kernelv_diagnosis_upload()

Prototyp

```
KERNELV_RETURN      kernelv_diagnosis_upload(char* filename);
```

Beschreibung

Startet den Upload der internen Diagnosedaten der DLL und schreibt diese in die Datei mit dem Namen filename. Wird kein Dateiname angegeben, so werden die Daten in diag_data.txt abgespeichert. Falls in filename kein Pfad angegeben wird, so wird die Datei in den aktuellen Pfad geschrieben.

Parameter

Name	Typ	Bedeutung
Filename	char*	Name der Datei, in die geschrieben werden soll.

Rückgabewerte

Symbol	Wert	Bedeutung
RET_FINISHED	0	Der Upload wurde beendet.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.

4 kernelv API Typen

Sämtliche aufgeführte Typen sind in der Datei **kernelv.h** definiert.

4.1 Enum KERNELV_RETURN

Beschreibung

Rückgabewerte und Fehlercodes von API-Funktionen.

Symbol	Wert	Bedeutung
RET_FINISHED	0	Die Funktion wurde fehlerfrei durchgeführt.
RET_BUSY	1	Die Funktion wird gerade ausgeführt, ist aber noch nicht abgeschlossen. Die API-Funktion muss weiter aufgerufen werden.
ERR_INVALID_CHAN	-1	Der übergebene Kanalindex ist größer als die Anzahl der konfigurierten Kanäle -1.
ERR_PROG_NAME_LENGTH	-2	Der übergebene Programmname ist länger als zulässig.
ERR_CNC_NOT_INIT	-3	Der Simulations-CNC-Kern ist noch nicht initialisiert.
ERR_CNC_RET_MEMORY	-4	Der oder die Rückgabewerte passen nicht in den bereitgestellten Speicher.
ERR_INVALID_STATE	-5	Der CNC-Kanal ist im falschen Zustand um eine Funktion auszuführen.
ERR_DOUBLE_KERNEL	-6	Es läuft bereits eine Instanz der Simulations-CNC die denselben Instanzprefix verwendet, dies kann passieren, wenn 2 Instanzen von kernelv mit dem Funktionsaufruf kernelv_startup() gestartet wurden, oder beim Aufruf von kernelv_startup_instance() mehrmals derselbe Instanzpräfix verwendet wurde.
ERR_SHM_STARTUP	-7	Beim Start konnten intern verwendete Shared memories nicht angelegt werden.
ERR_STARTUP	-8	Beim Start der Simulations-CNC ist ein Fehler aufgetreten. Mögliche Ursachen sind fehlende Parameterlisten oder fehlerhafte Einträge in Parameterlisten.
ERR_INVALID_AX	-9	Die übergebene Achsindex ist größer als die Anzahl der konfigurierten Achsen -1.
ERR_AXIS_ERROR	-10	Die CNC-Achse zeigt einen Fehler an. Es wird zusätzlich von der CNC eine Fehlermeldung ausgegeben.
ERR_INTERNAL_ERROR	-11	Es ist ein DLL-interner Fehler aufgetreten.
ERR_UNKNOWN_VARIABLE	-12	Der Variablenname ist im CNC-Kern nicht bekannt.
ERR_VARIABLE_SYNTAX	-13	Der Variablenname ist syntaktisch nicht korrekt, z.B. schließende Klammer bei Arrayvariablen fehlt.
ERR_DATA_TYPE_MISMATCH	-14	Bei einem Schreibzugriff auf eine Variable passt stimmt der übergebene Datentyp nicht mit dem CNC-intern verwendeten Datentyp überein.

Symbol	Wert	Bedeutung
ERR_UNKNOWN_TECHNO_TYPE	-15	Bein Setzten der Bearbeitungszeit für eine Technologiefunktion wurde ein ungültiger Typ für die Technologiefunktion angegeben.
ERR_INVALID_TECHNO_PARAM	-16	Bein Setzten der Bearbeitungszeit für eine Technologiefunktion wurde ein ungültiger Parameter übergeben, z.B. übergebene Nummer der M- bzw. H-Funktion ist größer als die maximal zulässige Anzahl.
ERR_NO_LICENSE	-17	Es wurde keine Lizenz für die Verwendung der kernelv-DLL gefunden.

ERR_VAR_NAME_LENGTH	-18	Der an die Funktion übergebene Variablenname überschreitet die maximal zulässige Länge (KERNELV_VAR_NAME_LENGTH).
ERR_REGISTRY_ACCESS	-19	Beim Versuch Werte aus der Windows-Registry zu lesen ist ein Fehler aufgetreten.
ERR_UNKNOWN_OPTION	-20	Der Funktion kernelv_set_options() wurde eine unbekannte Option übergeben.
ERR_ARRAY_NOT_SUPPORTED	-21	Bei manchen Echtzeit Varianten der CNC ist es möglich ein Array ‚en block' zu lesen bzw. zu schreiben, indem beim Zugriff der Arrayindex weggelassen wird, diese Zugriffsart wird aktuell von der kernelv-DLL nicht unterstützt.
ERR_VAR_NOT_WRITEABLE	-22	Es wurde versucht eine nicht schreibbare Variable zu beschreiben. Für den Schreibzugriff auf Variable gelten dieselben Zugriffsregeln, wie sie auch innerhalb eines NC-Programms gelten. Einzige Ausnahme hierbei sind V.E-Variable, diese können, unabhängig von den konfigurierten Zugriffsrechten, immer beschrieben werden.
ERR_PREFIX_TOO_LONG	-23	Beim Aufruf der Funktion kernelv_startu_prefix() ist die übergebene Instanzkennung zu lang, sodass die intern generierten Namen für die verwendeten Shared Memories nicht mehr in den dafür vorgesehenen Speicher passen. . Die zulässige Länge ist in dem Präprozessorkonstanten KERNELV_INSTANCE_PREFIX_MAX_LEN festgelegt.
ERR_DOUBLE_INSTANCE	-24	Aus dieser DLL wurde bereits eine kernelv-Instanz gestartet, es ist nicht möglich, aus einer Applikation mehrere Instanzen von kernelv zu starten.
ERR_INVALID_START_MODE	-25	Beim Aufruf der Funktion kernelv_ch_program_start() wurde ein ungültiger Startmode als Parameter übergeben. Gültige Werte für den Bearbeitungsmodus, siehe E_KERNELV_PROG_START_MODE.
ERR_READ_ERR	-26	Beim Lesen von Daten aus der kernelv-DLL ist ein Fehler aufgetreten.
ERR_WRITE_ERR	-27	Beim Schreiben von Daten in die kernelv-DLL ist ein Fehler aufgetreten.
ERR_CNC_NO_DATA	-28	Die angeforderten Daten sind nicht verfügbar.
ERR_TECHNO_NOT_FOUND	-29	Die angegebene Technologiefunktion wurde nicht gefunden.
ERR_INVALID_PARAMETER	-30	Es wurde ein ungültiger Parameter übergeben.
ERR_STARTUP_CHAN_INIT	-31	Beim Start der kernelev-Dll konnte die Initialisierung der konfigurierten NC-Kanäle nicht durchgeführt werden.

Symbol	**Wert**	**Bedeutung**
ERR_NC_PROGRAM	-32	Im NC-Programm ist ein Fehler aufgetreten.
ERR_CH_ERROR_STATE	-33	Der Kanal befindet sich im Fehlerzustand.
ERR_TIME_OUT	-34	Die Funktion konnte innerhalb des Zeitlimits nicht beendet werden.
ERR_NULL_PARAMETER	-35	Es wurde ein Zeiger übergeben, der nicht referenziert ist.

4.2 KERNELV_CHANNEL_STATE

Durch die Enumeration CNC_SIMU_CHANNEL_STATE wird der Zustand eines CNC-Kanals beschrieben.

Der Zustand eines CNC-Kanals wird durch das folgende Zustandsdiagramm beschrieben:

Symbol	Wert	Bedeutung
KERNELV_STATE _DESELECTED	1	Kanalzustand ist DESELECTED.
KERNELV_STATE _SELECTED	2	Kanalzustand ist SELECTED.
KERNELV_STATE_READY	3	Kanalzustand ist READY.
KERNELV_STATE_ACTIVE	4	Kanalzustand ist ACTIVE, es wird gerade ein NC-Programm abgearbeitet.
KERNELV_STATE_HOLD	5	Kanalzustand ist HOLD. Es wurde ein NC-Programm gestartet und danach angehalten.
KERNELV_STATE_ERROR	6	Kanalzustand ist ERROR.
KERNELV_STATE _SELECTING	7	Kanalzustand ist SELECTING.
KERNELV_STATE _DESELECTING	8	Kanalzustand ist DESELECTING.
KERNELV_STATE _PREPARING	9	Kanalzustand ist PREPARING.
KERNELV_STATE _CLEARING	10	Kanalzustand ist CLEARING.
KERNELV_STATE _STARTING	11	Kanalzustand ist STARTING. Es wird gerade ein NC-Programm gestartet.
KERNELV_STATE _ABORTING	12	Kanalzustand ist ABORTING. Es wird gerade ein NC-Programm abgebrochen.
KERNELV_STATE _STOPPING	13	Kanalzustand ist STOPPING. Einlaufendes NC-Programm wird angehalten.
KERNELV_STATE _RESUMING	14	Kanalzustand ist RESUMING. Ein angehaltenes NC-Programm wird weiterbearbeitet.
KERNELV_STATE _RESETTING	15	Kanalzustand ist RESETTING. Es wird gerade ein Reset durchgeführt.

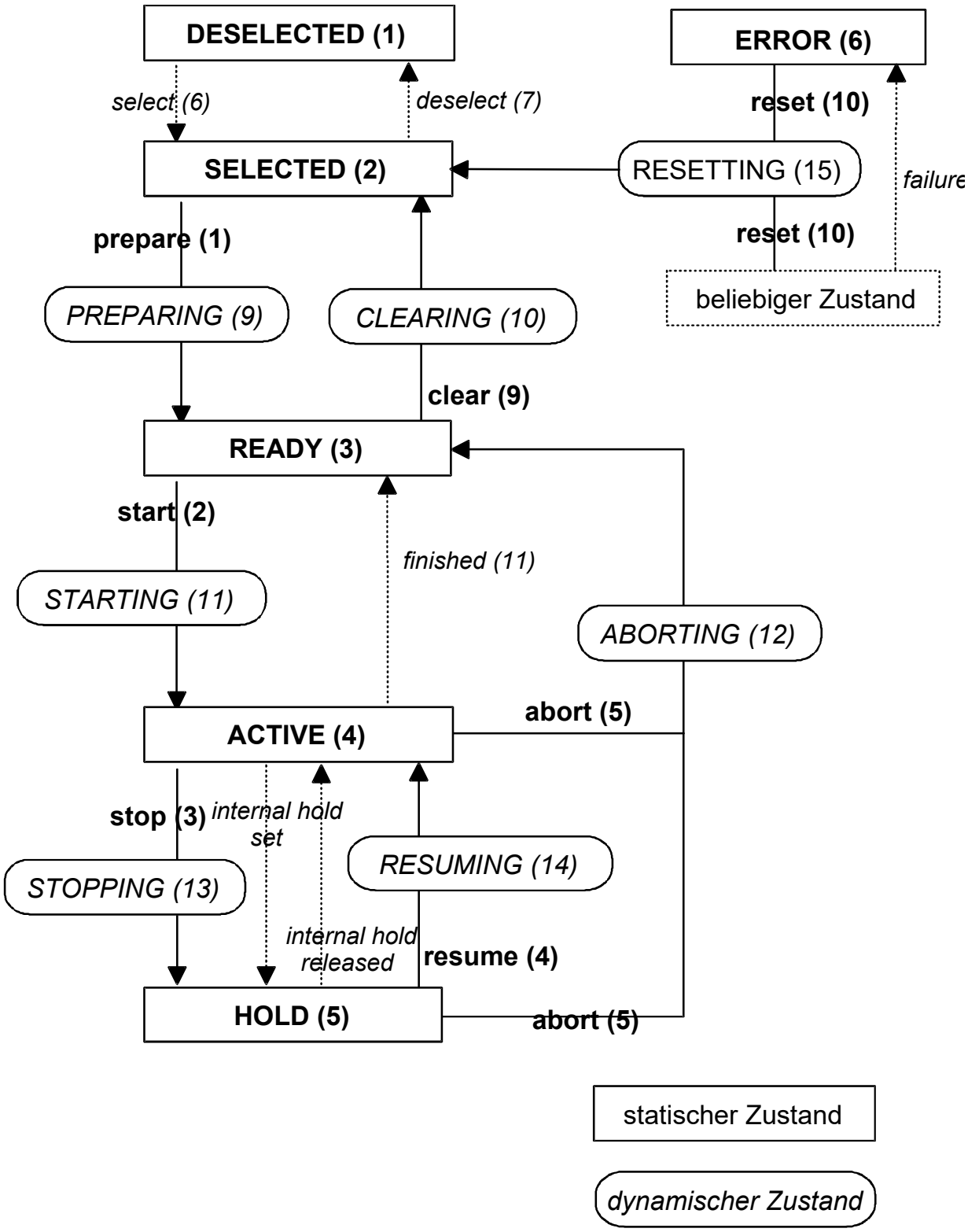


Abb. 6: Zustandsdiagramm eines CNC-Kanals

4.3 Enum E_KERNELV_TECHNO_TYPE

Beschreibung

Typ einer in der Struktur KERNELV_TECHNO_DATA gespeicherten Technologiefunktion.

Symbol	Wert	Bedeutung
KERNELV_TECHNO_EMPTY	0	Die Struktur enthält keine gültigen Werte.
KERNELV_TECHNO_M_CODE	1	Die Struktur enthält eine M-Funktion.
KERNELV_TECHNO_H_CODE	2	Die Struktur enthält eine H-Funktion.
KERNELV_TECHNO_S_CODE	3	Die Struktur enthält eine Spindel-Techno-Funktion, z.B. M3, M19 oder S.
KERNELV_TECHNO_T_CODE	4	Die Struktur enthält eine Werkzeug-Techno-Funktion.

4.4 Struct KERNELV_TECHNO_DATA

Beschreibung

Struktur mit vom CNC-Kern quittierten Technologiedaten.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher

Element	Type	Bedeutung
type	E_KERNELV_TECHNO_TYPE	Type der im Element param gespeicherten Technologiefunktion.
param	U_KERNELV_TECHNO_PARAM	Union mit den Daten der Technologiefunktion.

4.5 KERNELV_CHANNEL_TECHNO_DATA_ARRAY

Beschreibung

Definiert ein Array der Größe KERNELV_CHANNEL_TECHNO_DATA_COUNT von Strukturen des Typs KERNELV_TECHNO_DATA.

```
typedef KERNELV_TECHNO_DATA
KERNELV_CHANNEL_TECHNO_DATA_ARRAY[KERNELV_CHANNEL_TECHNO_DATA_COUNT];
```

Eine Variable dieses Typs kann den Funktionen kernelv_ch_get_techno_data() bzw. kernelv_ch_get_new_techno_data() übergeben werden um die Technologieinformationen zu lesen.

```
KERNELV_CHANNEL_TECHNO_DATA_ARRAY  ch_techno;

unsigned long int                  techno_len;

if ( kernelv_ch_get_techno_data(0,
                            ch_techno,
                            sizeof(ch_techno),
                            &techno_len) == RET_FINISHED)

{
for (int i = 0; i < KERNELV_CHANNEL_TECHNO_DATA_COUNT,i++)

.   .printf("Type: %d\n", ch_techno[i].type);

}
```

4.6 KERNELV_CHANNEL_TECHNO_DATA_ARRAY2

Beschreibung

Definiert ein Array der Größe KERNELV_CHANNEL_TECHNO_DATA_COUNT von Strukturen des Typs KERNELV_TECHNO_DATA2.

```
typedef KERNELV_TECHNO_DATA2

KERNELV_CHANNEL_TECHNO_DATA_ARRAY2[KERNELV_CHANNEL_TECHNO_DATA_COUNT];
```

Eine Variable dieses Typs kann den Funktionen kernelv_ch_get_techno_data2() bzw. kernelv_get_new_techno_data2() übergeben werden um die Technologieinformationen zu lesen.

```
KERNELV_CHANNEL_TECHNO_DATA_ARRAY2  ch_techno;

unsigned long int                   techno_len;

if ( kernelv_ch_get_techno_data2(0,

                            ch_techno,

                            sizeof(ch_techno),

                            &techno_len) == RET_FINISHED)

{

for (int i =0; i < KERNELV_CHANNEL_TECHNO_DATA_COUNT, i++)

.   .printf("Type: %d\n", ch_techno[i].type);

}
```

4.7 KERNELV_AXIS_TECHNO_DATA_ARRAY

Beschreibung

Definiert ein Array der Größe KERNELV_AXIS_TECHNO_DATA_COUNT von Strukturen des Typs KERNELV_TECHNO_DATA.

```
typedef KERNELV_TECHNO_DATA
KERNELV_AXIS_TECHNO_DATA_ARRAY[KERNELV_AXIS_TECHNO_DATA_COUNT];
```

Eine Variable dieses Typs kann den Funktionen kernelv_ax_get_techno_data() bzw. kernelv_ax_get_new_techno_data() übergeben werden um die Technologieinformationen zu lesen.

```
KERNELV_CHANNEL_TECHNO_DATA_ARRAY  ax_techno;

unsigned long int                  techno_len;


if ( kernelv_ax_get_techno_data(0,
                           ax_techno,
                           sizeof(ch_techno),
                           &techno_len) == RET_FINISHED)

{
for (int i =0; i < KERNELV_CHANNEL_TECHNO_DATA_COUNT ,i++)

.  .printf("Type: %d\n", ax_techno[i].type);

}
```

4.8 Struct KERNELV_TECHNO_DATA2

Beschreibung

Struktur mit vom CNC-Kern quittierten Technologiedaten.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher

Element	Type	Bedeutung
type	E_KERNELV_TECHNO_TYPE	Type der im Element param gespeicherten Technologiefunktion.
param	U_KERNELV_TECHNO_PARAM2	Union mit den Daten der Technologiefunktion.

4.9 KERNELV_CHANNEL_TECHNO_DATA_ARRAY2

Beschreibung

Definiert ein Array der Größe KERNELV_CHANNEL_TECHNO_DATA_COUNT von Strukturen des Typs KERNELV_TECHNO_DATA2.

```
typedef KERNELV_TECHNO_DATA2
KERNELV_CHANNEL_TECHNO_DATA_ARRAY2[KERNELV_CHANNEL_TECHNO_DATA_COUNT];
```

Eine Variable dieses Typs kann den Funktionen kernelv_ch_get_techno_data2() bzw. kernelv_ch_get_new_techno_data2() übergeben werden um die Technologieinforamtionen zu lesen.

```
KERNELV_CHANNEL_TECHNO_DATA_ARRAY2  ch_techno;

unsigned long int                   techno_len;

if ( kernelv_ch_get_techno_data2(0,

                            ch_techno,

                            sizeof(ch_techno),

                            &techno_len) == RET_FINISHED)

{

for (int i =0; i < KERNELV_CHANNEL_TECHNO_DATA_COUNT ,i++)

.   .printf("Type: %d\n", ch_techno[i].type);

}
```

4.10 KERNELV_AXIS_TECHNO_DATA_ARRAY2

Beschreibung

Definiert ein Array der Größe KERNELV_AXIS_TECHNO_DATA_COUNT von Strukturen des Typs KERNELV_TECHNO_DATA2.

```
typedef KERNELV_TECHNO_DATA2
KERNELV_AXIS_TECHNO_DATA_ARRAY2[KERNELV_AXIS_TECHNO_DATA_COUNT];
```

Eine Variable dieses Typs kann den Funktionen kernelv_ax_get_techno_data2() bzw. kernelv_ax_get_new_techno_data2() übergeben werden um die Technologieinformationen zu lesen.

```
KERNELV_CHANNEL_TECHNO_DATA_ARRAY2  ax_techno;

unsigned long int                   techno_len;

if ( kernelv_ax_get_techno_data2(0,
                            ax_techno,
                            sizeof(ch_techno),
                            &techno_len) == RET_FINISHED)

{
for (int i =0; i < KERNELV_AXIS_TECHNO_DATA_COUNT ,i++)

.   .printf("Type: %d\n", ch_techno[i].type);

}
```

4.11 Union U_KERNELV_TECHNO_PARAM

Beschreibung

Union mit den Daten einer Technologiefunktion.

Speicherausrichtung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Type	Bedeutung
m_h	M_H_CODE_DATA	Daten einer M/H-Technologiefunktion
spindle	S_CODE_DATA	Daten einer Spindel-Technologiefunktion
tool	T_CODE_DATA	Daten einer Werkzeug-Funktion.

4.12 Union U_KERNELV_TECHNO_PARAM2

Beschreibung

Union mit den Daten einer Technologiefunktion.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Type	Bedeutung
m_h	M_H_CODE_DATA2	Daten einer M/H-Technologiefunktion.
spindle	S_CODE_DATA	Daten einer Spindel-Technologiefunktion.
tool	T_CODE_DATA	Daten einer Werkzeug-Funktion.

4.13 Struct M_H_CODE_DATA

Beschreibung

Daten einer M/H-Funktion.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Type	Bedeutung
nr	unsigned long int	Nummer der M/H-Funktion.
duration	unsigned long int	Bei aktiver Bearbeitungssimulation eingestellte Bearbeitungszeit in us.

4.14 Struct M_H_CODE_DATA2

Beschreibung

Daten einer M/H-Funktion.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Type	Bedeutung
nr	unsigned long int	Nummer der M/H-Funktion.
duration	unsigned long int	Bei aktiver Bearbeitungssimulation eingestellte Bearbeitungszeit in us.
add_value	signed long int	Im NC-Programm zusätzlich programmierter Wert (M4711 = 123).
fillup	unsigned long int	Füllbytes für 8 Byte-Alignment.

4.15 Enum E_KERNELV_SPINDLE_TYPE

Beschreibung

Typ einer in der Struktur S_CODE_DATA gespeicherten Technologiefunktion.

Symbol	Wert	Bedeutung

KERNELV_TECHNO_S_EMPTY	0	Die Struktur enthält keine gültigen Werte.
KERNELV_TECHNO_S_M3	1	Die Struktur enthält eine M3 Spindelfunktion.
KERNELV_TECHNO_S_M4	2	Die Struktur enthält eine M4 Spindelfunktion.
KERNELV_TECHNO_S_M5	3	Die Struktur enthält eine M5 Spindelfunktion.
KERNELV_TECHNO_S_M19	4	Die Struktur enthält eine M19 Spindelfunktion.

4.16 Struct S_CODE_DATA

Beschreibung

Die Struktur enthält die zu einer Spindel-Technologiefunktion gehörigen Daten.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
type	E_KERNELV_SPINDLE_TYPE	Datentyp der Spindel-Technologiefunktion.
axis_number	unsigened short int	Achsnummer der Achse, an die die Technologiefunktion ausgegeben wurde..
revolutions	unsigned long int	Spindeldrehzahl.
position	signed long int	Zielposition der Spindel, wenn die Spindel positioniert wird.
duration	unsigned long int	Bei aktiver Bearbeitungssimulation eingestellte Bearbeitungszeit in us.

4.17 Struct T_CODE_DATA

Beschreibung

Die Struktur enthält die zu einer Werkzeug-Technologiefunktion gehörigen Daten.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
basic	signed long int	Basisnummer des Werkzeugs.
sister	signed long int	Nummer des Schwesterwerkzeuges.
variant	signed long int	Variantennummer des Werkzeuges.

4.18 Struct KERNELV_NC_LINE_DATA

Beschreibung

Die Struktur die zu einer abgearbeiteten NC-Programmzeile gehörenden Daten.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
fileoffset	unsigned long int	Dateioffset der abgearbeiteten Zeile in der aktuell aktiven .NC-Programmdatei.
ncLineNumber	unsigned long int	NC-Zeilennummer des der abgearbeiteten NC-Zeile, falls programmiert. Falls keine NC-Zeilennummer programmiert wurde, wird die letzte programmierte Zeilennummer eingetragen.
filename	char[KERNELV_FILE_NAME_-LENGTH + 1]	Dateiname der Datei, die die gerade abgearbeitete NC-Zeile enthält.

4.19 Enum E_KERNELV_VAR_TYPE

Beschreibung

Datentyp eines in der Struktur U_KERNELV_VAR_VALUE gespeicherten Datums.

Symbol	Wert	Bedeutung
KERNELV_VAR_TYPE_UNKNOWN	1	Die Struktur enthält keinen gültigen Datentyp.
KERNELV_VAR_TYPE_BOOLEAN	2	Datentyp ist unsigned char. Mögliche Werte 0 oder 1.
KERNELV_VAR_TYPE_UNS08	3	Datentyp ist unsigned char.
KERNELV_VAR_TYPE_SGN08	4	Datentyp ist signed char.
KERNELV_VAR_TYPE_UNS16	5	Datentyp ist unsigned short int.
KERNELV_VAR_TYPE_SGN16	6	Datentyp ist signed short int.
KERNELV_VAR_TYPE_UNS32	7	Datentyp ist unsigned long int.
KERNELV_VAR_TYPE_SGN32	8	Datentyp ist signed long int.
KERNELV_VAR_TYPE_DOUBLE	9	Datentyp ist double.
KERNELV_VAR_TYPE_STRING	10	Datentyp ist eine Zeichenkette mit maximal 127 (KERNELV_VAR_STRING_LEN) Zeichen (Gesamtlänge inklusive terminierender 0 KERNELV_VAR_STRING_LEN +1 Zeichen).

4.20 Union U_KERNELV_VAR_VALUE

Beschreibung

Union mit den möglichen Werten einer CNC-Variablen.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typkennung E_KERNELV_VAR_TYPE	Bedeutung
boolean	KERNELV_VAR_TYPE_BOOLEAN	Datentyp ist unsigned char. Mögliche Werte 0 oder 1.
uns08	KERNELV_VAR_TYPE_UNS08	Datentyp ist unsigned char.
sgn08	KERNELV_VAR_TYPE_SGN08	Datentyp ist signed char.
uns16	KERNELV_VAR_TYPE_UNS16	Datentyp ist unsigned short int.
sgn16	KERNELV_VAR_TYPE_SGN16	Datentyp ist signed short int.
uns32	KERNELV_VAR_TYPE_UNS32	Datentyp ist unsigned long int.
sgn32	KERNELV_VAR_TYPE_SGN32	Datentyp ist signed long int.
real64	KERNELV_VAR_TYPE_DOUBLE	Datentyp ist double.
string	KERNELV_VAR_TYPE_STRING	Datentyp ist eine Zeichenkette mit maximal 127 (KERNELV_VAR_STRING_LEN) Zeichen (Gesamtlänge inklusive terminierender Null KERNELV_VAR_STRING_LEN +1 Zeichen).

4.21 Struct KERNELV_VARIABLE

Beschreibung

Die Struktur enthält Wert und Type einer CNC-Variablen.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
type	E_KERNELV_VAR_TYPE	Datentyp des im Strukturelement value stehenden Datums.
value	U_KERNELV_VAR_VALUE	Union mit den für CNC-Variablen möglichen Datentypen.

4.22 Struct KERNELV_NC_LINE_DATA

Beschreibung

Die Struktur enthält Informationen über die im aktuellen Takt abgearbeiteten NC-Programmzeilen.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
fileoffset	unsigned long int	Fileoffset der Zeilen im NC-Programm.
ncLineNumber	unsigned long int	NC-Zeilennummer der Zeile. Es wird die im NC-Programm definierte Zeilennummer (N gefolgt von numerischem Wert) zurückgegeben. Falls in der abgearbeiteten Zeile keine NC-Zeilennummer angegeben wurde, wird die jeweils letzte programmierte Zeilennummer zurückgegeben.
filename	char[KERNELV_FILE_NAME_LENGTH + 1]	Dateiname des NC-Programmes das gerade abgearbeitet wird.

4.23 Struct KERNELV_LICENSE_INFO

Beschreibung

Die Struktur enthält Informationen über die lizensierten Optionen.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
CncSysIdOK	unsigned long int	Gültige Lizenz vorhanden.
CncAxesPack	unsigned long int	Maximal konfigurierbare Achsanzahl.
CncChannels	unsigned long int	Maximal konfigurierbare Kanalanzahl.
CncTrafo	unsigned long int	Transformationspaket ist lizensiert.
CncSpline	unsigned long int	Splinepaket ist lizensiert.
CncSpline	unsigned long int	Die Lizenz ist eine Exportlizenz (Funktionseinschränkungen aufgrund vor Exportbestimmungen).
CncDll	unsigned long int	Verwendung der kernelv-DLL ist lizensiert.

4.24 Struct KERNELV_DECODER_POSITION_HEADER

Beschreibung

Die Struktur enthält allgemeine Daten über vom Decoder gelesene Achspositionen.

Beim Aufruf der Funktion kernelv_ch_get_decoder_positions() wird zuerst eine Struktur des Typs KERNELV_DECODER_POSITION_HEADER zurückgegeben. Anschließend folgt für jede im Kanal vorhandene Achse eine Struktur des Typs KERNELV_DECODER_POSITION_DATA.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
data_valid	unsigned char	Gültigkennung für die folgenden Daten.
line_number	unsigned long int	NC Zeilennummer.
Block_count	unsigned long int	Satzzähler.
axis_count	unsigned short int	Anzahl der Achsen im Kanal, Anzahl der folgenden Strukturen des Typs KERNELV_DECODER_POSITION_DATA.

4.25 Struct KERNELV_DECODER_POSITION_DATA

Beschreibung

Die Struktur enthält enthält die Decoder Achspositionen einer Achse.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
mcs_pos	unsigned long int	Achsposition im Maschinenkoordinatensystem.
pcs_pos_no_offset	unsigned long int	Achsposition im Programmierkoordinatensystem ohne Verschiebungen.
pcs_pos_offset	unsigned long int	Achsposition im Programmierkoordinatensystem mit Verschiebungen.
axis_number	unsigned short int	Achsnummer der Achse.

4.26 Enum E_KERNELV_PROG_START_MODE

Beschreibung

Gibt den Bearbeitungsmodus des Programmstarts an.

Symbol	Wert	Bedeutung
KERNELV_START_MODE_NORMAL	0	Normale Bearbeitung.
KERNELV_START_MODE_CONTOUR_VISU	1	Bearbeitungsmodus Sollkonturvisualisierung.

4.27 Struct ACTIVE_G_CODES

Beschreibung

Enthält ein Array mit den aktiven G-Funktionen der jeweiligen G-Funktioonsgruppe.

Element	Typ	Bedeutung
group[KERNELV_G_MAX_G_GRP]	short int	Array der G-Gruppen.

4.28 Enum E_KERNELV_G_GROUP_TYPE

Beschreibung

Enumeration mit Kennungen für die unterschiedlichen G-Funktionstypen.

Symbol	Wert	Bedeutung
KERNELV_G_PATH_COND	0	Gruppe 0, Wegebedingung, mögliche aktive G-Funktionen: G00, G01, G02, G03, G04, G33, G63, G74, G98, G99, G301, G302, G160
KERNELV_G_PATH_FEED	1	Gruppe 1, mögliche aktive G-Funktionen: G08, G193
KERNELV_G_DEC	2	Gruppe 2, mögliche aktive G-Funktionen: G09, G900, G901
KERNELV_G_FEED_ADAPT	3	Gruppe 3, mögliche aktive G-Funktionen: G09, G900, G901
KERNELV_G_ACITVE_PLANE	4	Gruppe 4, mögliche aktive G-Funktionen: G17, G18, G19
KERNELV_G_MIRROR	5	Gruppe 5, mögliche aktive G-Funktionen: G20, G21, G22, G23, G24, G351
KERNELV_G_TRC_TRANSITION	6	Gruppe 6, mögliche aktive G-Funktionen: G25, G26
KERNELV_G_TOOL_RADIUS_COMP	7	Gruppe 7, mögliche aktive G-Funktionen: G40, G41, G42
KERNELV_G_DIAMETER_PROG	8	Gruppe 8, mögliche aktive G-Funktionen: G40, G41, G42
KERNELV_G_ZERO_POS_SHIFT	9	Gruppe 9, mögliche aktive G-Funktionen: G53-G59, G159
KERNELV_G_EXACT_STOP	10	Gruppe 10, mögliche aktive G-Funktionen: G60, G359, G360, G260, G261
KERNELV_G_OVERRRIDE_100	11	Gruppe 11, mögliche aktive G-Funktionen: G166
KERNELV_G_UNIT	12	Gruppe 12, mögliche aktive G-Funktionen: G70, G71
KERNELV_G_SUB_CALL	13	Gruppe 13, mögliche aktive G-Funktionen: G80-G89, G800-G819
KERNELV_G_ABS_REL	14	Gruppe 14, mögliche aktive G-Funktionen: G90, G91
KERNELV_G_POS_SHIFT	15	Gruppe 15, mögliche aktive G-Funktionen: G92
KERNELV_G_FEED_PROG	16	Gruppe 16, mögliche aktive G-Funktionen: G93, G94, G95, G194
KERNELV_G_SPINDEL_FEED	17	Gruppe 17, mögliche aktive G-Funktionen: G96, G97, G196
KERNELV_G_GEAR_CHANGE	18	Gruppe 18, mögliche aktive G-Funktionen: G112
KERNELV_G_LOOKAHEAD	19	Gruppe 19, mögliche aktive G-Funktionen: G115, G116, G117
KERNELV_G_ACC_WEIGHT	20	Gruppe 20, mögliche aktive G-Funktionen: G130, G131
KERNELV_G_FEEDFORWARD	21	Gruppe 21, mögliche aktive G-Funktionen: G135, G136, G137
KERNELV_G_TRC_SELCECTION	22	Gruppe 22, mögliche aktive G-Funktionen: G05, G138, G139, G237, G238, G239
KERNELV_G_CIRCE_CENTER	23	Gruppe 23, mögliche aktive G-Funktionen: G161, G162
KERNELV_G_RADIUS_PROGR	24	Gruppe 24, mögliche aktive G-Funktionen: G163
KERNELV_G_CIRC_CENTER_CORR	25	Gruppe 25, mögliche aktive G-Funktionen: G164, G165
KERNELV_G_MANUAL_MODE	26	Gruppe 26, mögliche aktive G-Funktionen: G200, G201, G202
KERNELV_G_RAMP_TIME_WEIGHT	27	Gruppe 27, mögliche aktive G-Funktionen: G132, G133, G134
KERNELV_G_SPLINE	28	Gruppe 28, mögliche aktive G-Funktionen: G150, G151
KERNELV_G_PROBING	29	Gruppe 29, mögliche aktive G-Funktionen: G100, G101, G102, G106, G107, G108
KERNELV_G_CORNER_DECEL	30	Gruppe 30, mögliche aktive G-Funktionen: G12, G13
KERNELV_G_CONTOUR_MASKING	31	Gruppe 31, mögliche aktive G-Funktionen: G140, G141
KERNELV_G_PROBING_INTERR	32	Gruppe 32, mögliche aktive G-Funktionen: G310

KERNELV_G_SPINDLE_OVERRRIDE	33	Gruppe 33, mögliche aktive G-Funktionen: G167
KERNELV_G_RAPID_FEED_WEIGHT	34	Gruppe 34, mögliche aktive G-Funktionen: G129
KERNELV_G_CONTOUR	35	Gruppe 35, mögliche aktive G-Funktionen: G301, G302
KERNELV_G_CYCLE_SYNC	36	Gruppe 36, mögliche aktive G-Funktionen: G66
KERNELV_G_MAX_G_GRP	37	Number of groups, size of array

4.29 Datentypen der Konturvisualisierung

4.29.1 Struct CONTOUR_VISU

Beschreibung

Daten der Konturvisualisierung, Rückgabewert der Funktion kernelv_ch_get_cont_visu_data().

Element	Typ	Bedeutung	
count	signed long int	Anzahl der im Element data enthaltenen Einträge mit Visualisierungsdaten.	
ifc_version	unsigned long int	Interfaceversion der Visualisierungsdaten, wird durch den Hochlaufparameter P-STUP-00039 eingestellt.	
		ifc_version	Datentyp in CONTOUR_VISU
		0	CONTOUR_VISU_DATA_V0
		1	CONTOUR_VISU_DATA_V1
		2	CONTOUR_VISU_DATA_V2
		3	CONTOUR_VISU_DATA_V3
		4	CONTOUR_VISU_DATA_V4
		5	CONTOUR_VISU_DATA_V5
		6	CONTOUR_VISU_DATA_V6
		7	CONTOUR_VISU_DATA_V7
		8	CONTOUR_VISU_DATA_V8
data	CONTOUR_VISU_DATA	Konturvisualisierungsdaten entsprechend dem in ifc_version angegebenen Interfacetyp.	

4.29.2 Union CONTOUR_VISU_DATA

Beschreibung

Union mit möglichen Werten der Konturvisualisierung..

Element	Typ	Bedeutung
visu_data_v0	CONTOUR_VISU_DATA_V0[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 0, Arraygröße CONTOUR_MAX_DATA_V0
visu_data_v1	CONTOUR_VISU_DATA_V1[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 1, Arraygröße CONTOUR_MAX_DATA_V1.
visu_data_v2	CONTOUR_VISU_DATA_V2[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 2, Arraygröße CONTOUR_MAX_DATA_V2.
visu_data_v3	CONTOUR_VISU_DATA_V3[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 3, Arraygröße CONTOUR_MAX_DATA_V3.
visu_data_v4	CONTOUR_VISU_DATA_V4[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 4, Arraygröße CONTOUR_MAX_DATA_V4.
visu_data_v5	CONTOUR_VISU_DATA_V5[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 5, Arraygröße CONTOUR_MAX_DATA_V5.
visu_data_v6	CONTOUR_VISU_DATA_V6[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 6, Arraygröße CONTOUR_MAX_DATA_V6.
visu_data_v7	CONTOUR_VISU_DATA_V7[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 7, Arraygröße CONTOUR_MAX_DATA_V7.
visu_data_v8	CONTOUR_VISU_DATA_V8[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 8, Arraygröße CONTOUR_MAX_DATA_V8.
visu_data_v9	CONTOUR_VISU_DATA_V9[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 9, Arraygröße CONTOUR_MAX_DATA_V9.
visu_data_v10	CONTOUR_VISU_DATA_V10[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 10, Arraygröße CONTOUR_MAX_DATA_V10.
visu_data_v11	CONTOUR_VISU_DATA_V11[]	Konturvisualisierungsdaten bei Verwendung von Interfaceversion 11, Arraygröße CONTOUR_MAX_DATA_V11.

4.29.3 Struct CONTOUR_VISU_DATA_V0

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 0.

Element	Typ	Bedeutung
ch_data	CONTOUR_VISU_CH_DATA	Konturvisualisierungsdaten eines Kanals.
ax_data[]	CONTOUR_AXIS_DATA	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.4 Struct CONTOUR_VISU_DATA_V1

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 1.

Element	Typ	Bedeutung
ch_data	CONTOUR_VISU_CH_DATA	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data[]	CONTOUR_AXIS_DATA	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.5 Struct CONTOUR_VISU_DATA_V2

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 2.

Element	Typ	Bedeutung
ch_data_v1	CONTOUR_VISU_CH_DATA_V1	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data[]	CONTOUR_AXIS_DATA	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.6 Struct CONTOUR_VISU_DATA_V3

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 3.

Element	Typ	Bedeutung
ch_data	CONTOUR_VISU_CH_DATA	Konturvisualisierungsdaten eines Kanals.
ax_data_v1[]	CONTOUR_AXIS_DATA_V1	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.7 Struct CONTOUR_VISU_DATA_V4

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 4.

Element	Typ	Bedeutung
ch_data	CONTOUR_VISU_CH_DATA	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data_v1[]	CONTOUR_AXIS_DATA_V1	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.8 Struct CONTOUR_VISU_DATA_V5

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 5.

Element	Typ	Bedeutung
ch_data_v1	CONTOUR_VISU_CH_DATA_V1	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data_v1[]	CONTOUR_AXIS_DATA_V1	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.9 Struct CONTOUR_VISU_DATA_V6

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 6.

Element	Typ	Bedeutung
ch_data	CONTOUR_VISU_CH_DATA	Konturvisualisierungsdaten eines Kanals.
ax_data_v2[]	CONTOUR_AXIS_DATA_V2	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.10 Struct CONTOUR_VISU_DATA_V7

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 7.

Element	Typ	Bedeutung
ch_data	CONTOUR_VISU_CH_DATA	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data_v2[]	CONTOUR_AXIS_DATA_V2	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.11 Struct CONTOUR_VISU_DATA_V8

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 8.

Element	Typ	Bedeutung
ch_data_v1	CONTOUR_VISU_CH_DATA_V1	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data_v2[]	CONTOUR_AXIS_DATA_V2	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.12 Struct CONTOUR_VISU_DATA_V9

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 9.

Element	Typ	Bedeutung
ch_data_v2	CONTOUR_VISU_CH_DATA_V2	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data[]	CONTOUR_AXIS_DATA	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.13 Struct CONTOUR_VISU_DATA_V10

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 10.

Element	Typ	Bedeutung
ch_data_v2	CONTOUR_VISU_CH_DATA_V2	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data_v1[]	CONTOUR_AXIS_DATA_V1	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.14 Struct CONTOUR_VISU_DATA_V11

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversion 7.

Element	Typ	Bedeutung
ch_data_v2	CONTOUR_VISU_CH_DATA_V2	Konturvisualisierungsdaten eines Kanals.
file_name	CONTOUR_VISU_FILE_NAME	Dateiname des aktiven NC-Programms. Arraygröße: FILE_NAME_LN + 1
ax_data_v2[]	CONTOUR_AXIS_DATA_V2	Achsspezifische Visualisierungsdaten. Arraygröße: CONTOUR_AXIS_PER_CHANNEL

4.29.15 Struct CONTOUR_VISU_CH_DATA

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversionen 0, 1, 3, 4, 6, 7.

Element	Typ	Bedeutung
nc_block_nr	signed long int	NC-Satznummer
fileoffset	signed long int	Dateioffset in aktueller Programmdatei
channel_nr	unsigned short int	Kanalnummer
g_function	signed short int	Interpolationsart: 0: Eilgang 1: Linearinterpolation 2; 3: Zirkularinterpolation 61: Polynominterpolation -1: Interpolationsart ist nicht belegt.
circle_radius	unsigned long int	Kreisradius bei Zirkularinterpolation in 0,1 um
circle_center_point[2]	double	Kreismittelpunkt bei Zirkularinterpolation in 0,1 um.

4.29.16 Struct CONTOUR_VISU_CH_DATA_V1

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversionen 2, 5, 8.

Element	Typ	Bedeutung
nc_block_nr	signed long int	NC-Satznummer
fileoffset	signed long int	Dateioffset in aktueller Programmdatei
channel_nr	unsigned short int	Kanalnummer
g_function	signed short int	Interpolationsart: 0: Eilgang 1: Linearinterpolation 2; 3 Zirkularinterpolation 61: Polynominterpolation -1. Interpolationsart ist nicht belegt.
circle_radius	unsigned long int	Kreisradius bei Zirkularinterpolation in 0,1 um
circle_center_point[2]	double	Kreismittelpunkt bei Zirkularinterpolation in 0,1 um.
v_prog	signed long int	Programmierte Geschwindigkeit in 1 um/s.
techno	CONTOUT_VISU_DATA_TECHNO	Technologiedaten.
fillup	unsigned long int	In kernelv-Versionen mit Buildnummer > 3000 Füllbytes zur Strukturausrichtung.

4.29.17 Struct CONTOUR_VISU_CH_DATA_V2

Beschreibung

Kanalspezifische Daten der Konturvisualisierung bei Verwendung der Interfaceversionen 9, 10, 11

Element	Typ	Bedeutung
nc_block_nr	signed long int	NC-Satznummer
fileoffset	signed long int	Dateioffset in aktueller Programmdatei
channel_nr	unsigned short int	Kanalnummer
g_function	signed short int	Interpolationsart: 0: Eilgang 1: Linearinterpolation 2; 3: Zirkularinterpolation 61: Polynominterpolation -1: Interpolationsart ist nicht belegt.
circle_radius	unsigned long int	Kreisradius bei Zirkularinterpolation in 0,1 um
circle_center_point[2]	double	Kreismittelpunkt bei Zirkularinterpolation in 0,1 um.
v_prog	signed long int	Programmierte Geschwindigkeit in 1 um/s.
techno	CONTOUT_VISU_DATA_TECHNO_V1	Technologiedaten.
fillup	unsigned long int	Alignmentbytes um 8 Byte Alignment zu erzwingen.

4.29.18 Struct CONTOUR_AXIS_DATA

Beschreibung

Achsspezifische Daten der Konturvisualisierung.

Element	Typ	Bedeutung
act_cmd_pos	signed long int	ACS-Sollposittion der Achse.
axis_nbr	unsigned short int	Achsnummer.
fillup	unsigned short int	Füllbytes zur Strukturausrichtung

4.29.19 Struct CONTOUR_AXIS_DATA_V1

Beschreibung

Achsspezifische Daten der Konturvisualisierung.

Element	Typ	Bedeutung
act_cmd_pos	signed long int	ACS-Sollposittion der Achse.
act_cmd_pos_wcs0	signed long int	WCS_0-Sollposittion der Achse. Wird nur berechnet, wenn der Kanalparameter P-CHAN-00145 den Wert1 hat und P-CHAN-00032 einen Wert > 0 hat.
axis_nbr	unsigned short int	Achsnummer.
fillup	unsigned short int	Füllbytes zur Strukturausrichtung

4.29.20 Struct CONTOUR_AXIS_DATA_V2

Beschreibung

Achsspezifische Daten der Konturvisualisierung.

Element	Typ	Bedeutung
act_cmd_pos	signed long int	ACS-Sollposition der Achse.
act_cmd_pos_wcs0	signed long int	WCS_0-Sollposition der Achse. (Sollposition der Achse im kartesischen Basiskoordinatensystem der Maschine). Wird nur berechnet, wenn der Kanalparameter P-CHAN-00145 den Wert1 hat und P-CHAN-00032 einen Wert > 0 hat.
act_cmd_pos_wcs	signed long int	WCS-Sollposition der Achse im gerade aktiven Koordinatensystem. Wird nur berechnet, wenn der Kanalparameter P-CHAN-00145 den Wert1 hat und P-CHAN-00032 einen Wert > 0 hat.
axis_nbr	unsigned short int	Achsnummer
fillup	unsigned short int	Füllbytes zur Strukturausrichtung.

4.29.21 Enum E_CONTOUR_TECHNO_TYPE

Beschreibung

Typ einer Technologiefunktion

Symbol	Wert	Bedeutung
TECHNO_UNKNOWN_TYPE_	0	Eintrag ist nicht belegt
TECHNO_M_TYPE	1	M-Funktion
TECHNO_H_TYPE	2	H-Funktion

4.29.22 Struct CONTOUR_M_H_PROCESS

Beschreibung

Daten einer ausgegebenen Technologiefunktion.

Element	Typ	Bedeutung
nr	unsigned long int	Nummer der Technologiefunktion.
sync	unsigned long int	Synchronisationsart der Technofunktion.
type	unsigned long int	Typ der Technologiefunktion, entsprechend E_CONTOUR_TECHNO_TYPE

4.29.23 Struct CONTOUR_M_H_PROCESS_V1

Beschreibung

Daten einer ausgegebenen Technologiefunktion.

Element	Typ	Bedeutung
nr	unsigned long int	Nummer der Technologiefunktion.
sync	unsigned long int	Synchronisationsart der Technofunktion.
type	unsigned long int	Typ der Technologiefunktion, entsprechend E_CONTOUR_TECHNO_TYPE
add_value	signed long int	Im NC Programm zusätzlich programmierter Wert (M4711 = 123).

4.29.24 Enum E_CONTOUR_S_CMD

Beschreibung

Typ einer Spindel-Technologiefunktion.

Symbol	Wert	Bedeutung
SPDL_CMD_UNKNOWN	0	Eintrag ist nicht belegt.
SPDL_CMD_M3	3	M3-Funktion (Spindel drehen im Uhrzeigersinn).
SPDL_CMD_M4	4	M4-Funktion (Spindel drehen im Gegenuhrzeigersinn).
SPDL_CMD_M5	5	M5-Funktion (Spindel Stopp).
SPDL_CMD_M19	19	M19-Funktion (Spindel positionieren).

4.29.25 Struct CONTOUR_S_PROCESS

Beschreibung

Daten einer ausgegebenen Spindelfunktion.

Element	Typ	Bedeutung
axis_nr	unsigned short int	Nummer der Spindelachse.
cmd	unsigned short int	Spindelkommando entsprechend E_CONTOUR_S_CMD.
sync	unsigned long int	Synchronisationsart der Spindelfunktion.
position	signed long int	Zielposition beim Spindelpositionieren in $0{,}1*10^{-3}°$
revolution	signed long int	Spindeldrehzahl in $1*10^{-3}°/s$

4.29.26 Struct CONTOUR_TOOL_PROCESS

Beschreibung

Daten einer ausgegebenen T-Funktion.

Element	Typ	Bedeutung
basic	signed long int	Basisnummer des Werkzeugs.
sister	signed long int	Schwesternummer des Werkzeugs. Der Wert -1 bedeutet nicht belegt.
variant	signed long int	Variantennummer des Werkzeugs. Der Wert -1 bedeutet nicht belegt.

4.29.27 Struct CONTOUR_DATA_TECHNO

Beschreibung

Daten der ausgegebenen Technologiefunktionen.

Element	Typ	Bedeutung
axis_nr	unsigned short int	Achsnummer bei achsspezifisch ausgegebenen Technologiefunktionen, 0 bei Kanalspezifisch ausgegebenen Technologiefunktionen.
fillup	unsigned short int	Füllbytes zur Strukturausrichtung.
m_h_count	unsigned long int	Anzahl der belegten Einträge im Array m_h_data[].
m_h_data[ ]	CONTOUR_M_H_PROCESS	M/H-Technologiedaten. Arraygröße: CONTOUR_MAX_M_H_DATA
s_count	signed long int	Anzahl der belegten Einträge im Array s_procm_h_data[].
s_data[CONTOUR_SPDL_COUNT]	CONTOUR_S_PROCESS	Daten der ausgegebenen Spindeltechnologiefunktionen. Arraygröße: CONTOUR_MAX_SPDL_DATA
tool	CONTOUR_TOOL_PROCESS	Technologiedaten eines Werkzeugs.

4.29.28 Struct CONTOUR_DATA_TECHNO_V1

Beschreibung

Daten der ausgegebenen Technologiefunktionen.

Element	Typ	Bedeutung
axis_nr	unsigned short int	Achsnummer bei achsspezifisch ausgegebenen Technologiefunktionen, 0 bei Kanalspezifisch ausgegebenen Technologiefunktionen.
fillup	unsigned short int	Füllbytes zur Strukturausrichtung.
m_h_count	unsigned long int	Anzahl der belegten Einträge im Array m_h_data[].
m_h_data[ ]	CONTOUR_M_H_PROCESS_V1	M/H-Technologiedaten. Arraygröße: CONTOUR_MAX_M_H_DATA
s_count	signed long int	Anzahl der belegten Einträge im Array s_procm_h_data[].
s_data[CONTOUR_SPDL_COUNT]	CONTOUR_S_PROCESS	Daten der ausgegebenen Spindeltechnologiefunktionen. Arraygröße: CONTOUR_MAX_SPDL_DATA
tool	CONTOUR_TOOL_PROCESS	Technologiedaten eines Werkzeugs.

4.30 Datentypen der Fehlerausgabe

4.30.1 Struct KERNELV_ERROR_VALUE

Beschreibung

In der Fehlermeldung ausgegebene zusätzliche Werte.

Element	Typ	Bedeutung
type	E_KERNELV_ERR_VAL_TYPE	Datentyp des im Element data enthaltenen Wertes.
dimension	E_KERNELV_ERR_VAL_DIMENSION	Dimension des im Element data enthaltenen Wertes.
meaning	E_KERNELV_ERR_VAL_MEANING	Bedeutung des im Element data enthaltenen Wertes.
fillup	unsigned long int	Füllbytes zur Strukturausrichtung.
data	U_KERNELV_ERR_VAL_DATA	Union mit den eigentlichen Nutdaten.

4.30.2 KERNELV_ERROR_VALUE_ARRAY

Beschreibung

Definiert ein Array der Größe KERNELV_ERROR_VALUE_COUNT von Strukturen des Typs KERNELV_KERNELV_ERROR_VALUE.

```
typedef KERNELV_ERROR_VALUE
KERNELV_ERROR_VALUE_ARRAY[KERNELV_ERROR_VALUE_COUNT];
```

Eine Variable dieses Typs kann der Funktionen kernelv_get_error_message_values() übergeben werden um die Fehlermeldungswerte zu lesen.

4.30.3 Enum E_KERNELV_ERR_VAL_TYPE

Beschreibung

Kennungen für den Datentype des Wertes in U_KERNELV_ERR_VAL_DATA.

Symbol	Wert	Bedeutung
ERR_VAL_TYPE_NONE	-1	Unbekannter Datentyp.
ERR_VAL_TYPE_BOOLEAN	0	Datentyp ist unsigned char 1 (8 Bit). Mögliche Werte 0 oder.
ERR_VAL_TYPE_UNS08	1	Datentyp ist unsigned char (8 Bit).
ERR_VAL_TYPE_SGN08	2	Datentyp ist signed char (8 Bit).
ERR_VAL_TYPE_UNS16	3	Datentyp ist unsigned short int (16 Bit).
ERR_VAL_TYPE_SGN16	4	Datentyp ist signed short int (16 Bit).
ERR_VAL_TYPE_UNS32	5	Datentyp ist unsigned long int (32 Bit).
ERR_VAL_TYPE_SGN32	6	Datentyp ist signed long int (32 Bit).
ERR_VAL_TYPE_UNS64	7	Datentyp ist unsigned long long int (64 Bit).
ERR_VAL_TYPE_SGN64	8	Datentyp ist signed long long int (64 Bit).
ERR_VAL_TYPE_REAL64	9	Datentyp ist eine 64 Bit Fließkommazahl.
ERR_VAL_TYPE_REAL32	10	Datentyp ist eine 32 Bit Fließkommazahl.
ERR_VAL_TYPE_CHAR	11	Datentyp ist ein character
ERR_VAL_TYPE_STRING	12	Datentyp ist eine Zeichenkette der Länge KERNELV_ERR_MSG_STRING_LENGTH (ohne terminierende 0).
ERR_VAL_TYPE_ADRESSE	13	Datentyp ist eine Adresse.
ERR_VAL_TYPE_IGNORE	14	Datentyp ist nicht belegt.
ERR_VAL_TYPE_A3_REAL64	15	Datentyp ist ein Array mit 3 64 Bit Fließkommazahlen.
ERR_VAL_TYPE_BITARRAY_32	16	Datentyp ist eine Bitleiste mit 32 Bit.
ERR_VAL_TYPE_BITARRAY_16	17	Datentyp ist eine Bitleiste mit 16 Bit.

4.30.4 Enum E_KERNELV_ERR_VAL_DIMENSION

Beschreibung

Dimension des Wertes in U_KERNELV_ERR_VAL_DATA.

Symbol	Wert	Bedeutung
ERR_VAL_DIM_UNKNOWN	-1	Unbekannte Dimensionskennung.
ERR_VAL_DIM_NO_DIM	0	Keine Dimensionsangabe.
ERR_VAL_DIM_POSITION	1	Position in [10^-4 mm bzw. °].
ERR_VAL_DIM_POSITION_HIG_RES	2	Position in [10^-7 mm bzw. °]
ERR_VAL_DIM_VELOCITY	3	Geschwindigkeit in 10^-3 mm/s bzw. 10^-3 °/s.
ERR_VAL_DIM_ACCELERATION	4	Beschleunigung in mm/s^2 bzw °/s^2.
ERR_VAL_DIM_JERK	5	Ruck in mm/s^3 bzw °/s^3.
ERR_VAL_DIM_TIME	6	Zeit in us.
ERR_VAL_DIM_PERMILL	7	Faktorielle Angabe in 1/1000 (Promille).
ERR_VAL_DIM_INKREMENTS	8	(Encoder-) Inkremente.
ERR_VAL_DIM_REV_FEED	9	Umdrehungsvorschub in 10^-4mm/U.
ERR_VAL_DIM_CUTTING_SPEED	10	Schnittgeschwindigkeit 10^-3 mm/s.
ERR_VAL_DIM_PATH_RESOLUTION	11	Wegauflösung in Inkremente / 10^-4 mm.
ERR_VAL_DIM_INCR_PER_REV	12	Inkremente / Umdrehung
ERR_VAL_DIM_BYTE	13	Byte.
ERR_VAL_DIM_PROPORTIONAL_GAIN	14	Proportionalverstärkung 0,01/s.
ERR_VAL_DIM_FREQUENCY	15	Frequenz in Hz.
ERR_VAL_DIM_LOAD	16	Last kg bzw. kg*m^2.

4.30.5 Enum E_KERNELV_ERR_VAL_MEANING

Beschreibung

Bedeutung des Wertes in U_KERNELV_ERR_VAL_DATA.

Symbol	Wert	Bedeutung
ERR_VAL_MEAN_UNKNOWN	-1	Unbekannte Bedeutung.
ERR_VAL_MEAN_LIMIT	0	Grenzwert
ERR_VAL_MEAN_ACT_VAL	1	aktueller Wert
ERR_VAL_MEAN_ERR_VAL	2	fehlerhafter Wert
ERR_VAL_MEAN_EXPECT_VAL	3	erwarteter Wert
ERR_VAL_MEAN_CORR_VAL	4	korrigierter Wert
ERR_VAL_MEAN_LOG_AXIS_NR	5	logische Achsnummer
ERR_VAL_MEAN_DRIVE_TYPE	6	Antriebstyp
ERR_VAL_MEAN_LOG_BED_ELEM_NR	7	logische Bedienelementnummer
ERR_VAL_MEAN_STATE	8	Zustand
ERR_VAL_MEAN_TRANSITION	9	Transition
ERR_VAL_MEAN_SENDER	10	Sender
ERR_VAL_MEAN_CLASS	11	Klasse
ERR_VAL_MEAN_INSTANCE	12	Instanz
ERR_VAL_MEAN_IDENT_NR	13	Identifikationsnummer
ERR_VAL_MEAN_STATUS	14	Status
ERR_VAL_MEAN_RING_NR	15	Ringnummer
ERR_VAL_MEAN_SATZ_NR	16	Satznummer
ERR_VAL_MEAN_MIN_LIMIT	17	unterer Grenzwert
ERR_VAL_MEAN_MAX_LIMIT	18	oberer Grenzwert
ERR_VAL_MEAN_START_VAL	19	Startwert
ERR_VAL_MEAN_TARGET_VAL	20	Endwert
ERR_VAL_MEAN_FILENAME	21	Dateiname
ERR_VAL_MEAN_LINE	22	Zeile (Text) in einer Datei
ERR_VAL_MEAN_LINE_NUMBER	23	Zeilennummer in einer Datei
ERR_VAL_MEAN_COLUMN_NUMBER	24	Spaltennummer in einer Datei
ERR_VAL_MEAN_ARGUMENT	25	Argument
ERR_VAL_MEAN_PARAMETER	26	Parameter
ERR_VAL_MEAN_AXIS	27	Achse (String)
ERR_VAL_MEAN_COMPENSATION	28	Kompensationsindex
ERR_VAL_MEAN_IDENTIFIER	29	Identifier
ERR_VA_MEAN_CHAIN	30	Kette

4.31 Enum KERNELV_AXIS_OFFSET_TYPES

Beschreibung

Zuordnung der unterschiedlichen Offsettypen zum Index des offsetvektors der Funktion kernelv_ch_axis_get_offsets ().

Symbol	Wert	Bedeutung
KERNELV_AXIS_OFFSET_UNKNOWN	-1	Unbekannter Offettyp.
KERNELV_AXIS_OFFSET_ZERO	0	Durch Nullpunktverschiebung (G54 … G59) verursachter Offset.
KERNELV_AXIS_OFFSET_ADD_ZERO	1	Durch Bezugspunktverschiebung (G92) verursachter Offset.
KERNELV_AXIS_OFFSET_PSET	2	Durch Istwertsetzen (#PSET) verursachter Offset.
KERNELV_AXIS_OFFSET_CLAMP	3	Durch Platzversatzdaten verursachter Offset.
KERNELV_AXIS_OFFSET_TOOL	4	Durch Werkezugdaten verursachter Offset.
KERNELV_AXIS_OFFSET_MEASURE	5	Durch G101 (Messoffset einrechnen) verursachter Offset.
KERNELV_AXIS_OFFSET_MAN_OP	6	Durch Handbetrieb verursachter Offset.
KERNELV_AXIS_OFFSET_TRACK_CS_OFFSET	7	Durch #CS TRACK* verursachter offset.
KERNELV_AXIS_OFFSET_MAX	8	Datentyp ist signed long long int (64 Bit).

*Dieser NC-Befehl ist nicht in allen Versionen verfügbar.

4.32 Externe Messhardware

4.32.1 Struct KERNELV_EXT_LATCH_COMMAND_DATA

Beschreibung

Die Struktur enthält Informationen über von der CNC ausgegebene Kommandos an die externe Messhardware.

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
order_type	E_KERNELV_EXT_LATCH_ORDER	Art des Kommandos an die externe Latchhardware.
input	unsigned long int	Nummer des Messeingangs der externen Messhardware der für die Messung verwendet werden soll.
edge	E_KERNELV_MEAS_ACTIVE_EDGE	Flanke des Messsignals, das für die Messung verwendet werden soll.

4.32.2 Enum E_KERNELV_EXT_LATCH_ORDER

Beschreibung

Typ des auszuführenden Messkommandos.

Symbol	Wert	Bedeutung
E_KERNELV_NO_ORDER	0	Kein Kommando aktiv
E_KERNELV_ENABLE_PROBE	1	Messhardware aktivieren
E_KERNELV_DISABLE_PROBE	2	Messhardware deaktivieren

4.32.3 E_KERNELV_MEAS_ACTIVE_EDGE

Beschreibung

Auszuwertende Flanke des Messsignals.

Symbol	Wert	Bedeutung
E_KERNELV_MEAS_SIGNAL_LOW_ACTIVE	1	Messen auf negative Flanke.
E_KERNELV_MEAS_SIGNAL_HIGH_ACTIVE	2	Messen auf positive Flanke.

4.33 Fertigungszeitberechnung

Beschreibung

Die Struktur enthält die Dateinamen der Programme, die im Fertigungszeitmodus in den einzelnen Kanälen gestartet werden sollen.

KERNELV_PT_FILE_NAME ist vom Typ CHAR[KERNELV_PT_PRG_NAME_LEN]

Speicherausrichung

Die einzelnen Strukturelemente liegen gepackt im Speicher.

Element	Typ	Bedeutung
file	KERNELV_PT_FILE_NAME [KERNELV_PT_MAX_CHAN]	Dateinamen der Programme, die im Fertigungszeitmodus in den einzelnen Kanälen gestartet werden sollen.

5 kernelv API Konstanten

Sämtliche aufgeführte Konstanten sind in der Datei **kernelv.h** definiert.

5.1 KERNELV_VAR_STRING_LEN

Beschreibung

Maximalanzahl der Zeichen in einem String der Union U_KERNELV_VAR_VALUE. Die Gesamtgröße der Zeichenkette (inklusive terminierender Null) beträgt KERNELV_VAR_STRING_LEN + 1 Bytes.

Wert

127

5.2 KERNELV_FILE_NAME_LENGTH

Beschreibung

Maximalanzahl der Zeichen in einem String der Struktur KERNELV_NC_LINE_DATA. Die Gesamtgröße der Zeichenkette (inklusive terminierender Null) beträgt KERNELV_FILE_NAME_LENGTH + 1 Bytes.

Wert

83

5.3 KERNELV_VAR_NAME_LENGTH

Beschreibung

Maximal zulässige Länge des Variablennamens, beim Lesen und Schreiben von Variablen mit den Funktionen kernelv_ch_get_variable_value()/kernelv_ch_get_variable_value().

Wert

255

5.4 KERNELV_OPTION_LICENSE_CHECK_VERBOSE

Beschreibung

Bitmaske zur Aktivierung von zusätzlichen Ausgaben bei der Lizenzprüfung. Muss vor dem Aufruf von kernelv_startup() gesetzt werden.

Wert

1 (0x1)

5.5 CONTOUR_MAX_DATA_V0

Beschreibung

Arraygröße des Unionelements visu_data_v0[] der Union CONTOUR_VISU_DATA.

Wert

15

5.6 CONTOUR_MAX_DATA_V1

Beschreibung

Arraygröße des Unionelements visu_data_v1[] der Union CONTOUR_VISU_DATA.

Wert

10

5.7 CONTOUR_MAX_DATA_V2

Beschreibung

Arraygröße des Unionelements visu_data_v2[] der Union CONTOUR_VISU_DATA.

Wert

5

5.8 CONTOUR_MAX_DATA_V3

Beschreibung

Arraygröße des Unionelements visu_data_v3[] der Union CONTOUR_VISU_DATA.

Wert

10

5.9 CONTOUR_MAX_DATA_V4

Beschreibung

Arraygröße des Unionelements visu_data_v4[] der Union CONTOUR_VISU_DATA.

Wert

7

5.10 CONTOUR_MAX_DATA_V5

Beschreibung

Arraygröße des Unionelements visu_data_v5[] der Union CONTOUR_VISU_DATA.

Wert

7

5.11 CONTOUR_MAX_DATA_V6

Beschreibung

Arraygröße des Unionelements visu_data_v6[] der Union CONTOUR_VISU_DATA.

Wert

6

5.12 CONTOUR_MAX_DATA_V7

Beschreibung

Arraygröße des Unionelements visu_data_v7[] der Union CONTOUR_VISU_DATA.

Wert

5

5.13 CONTOUR_MAX_DATA_V8

Beschreibung

Arraygröße des Unionelements visu_data_v8[] der Union CONTOUR_VISU_DATA.

Wert

3

5.14 CONTOUR_MAX_DATA_V9

Beschreibung

Arraygröße des Unionelements visu_data_v9[] der Union CONTOUR_VISU_DATA.

Wert

5

5.15 CONTOUR_MAX_DATA_V10

Beschreibung

Arraygröße des Unionelements visu_data_v10[] der Union CONTOUR_VISU_DATA.

Wert

4

5.16 CONTOUR_MAX_DATA_V11

Beschreibung

Arraygröße des Unionelements visu_data_v8[] der Union CONTOUR_VISU_DATA.

Wert

3

5.17 CONTOUR_MAX_M_H_DATA

Beschreibung

Arraygröße des Elements m_h_data[] der Struktur CONTOUR_TECHNO_DATA.

Wert

20

5.18 CONTOUR_MAX_SPDL_DATA

Beschreibung

Arraygröße des Elements s_data[] der Struktur CONTOUR_TECHNO_DATA.

Wert

6

5.19 CONTOUR_AXIS_PER_CHANNEL

Beschreibung

Arraygröße des Elements ax_data[] bzw. ax_data_v1[] der Strukturen CONTOUR_VISU_DATA_V0 … CONTOUR_VISU_DATA_V8.

Wert

32

5.20 KERNELV_ERROR_VALUE_COUNT

Beschreibung

Arraygröße des Arrays ERROR_VALUE_ARRAY, Anzahl der von von der Funktion kernelv_get_error_values() zurückgelieferten Strukturen das Typs KERNELV_ERROR_VALUE.

Wert

5

5.21 KERNELV_ERR_MSG_ STRING_LENGTH

Beschreibung

Länge eines Strings in der Union U_KERNELV_ERR_VAL_DATA ohne die terminierende Null.

Wert

23

5.22 KERNELV_CHANNEL_TECHNO_DATA_COUNT

Beschreibung

Anzahl der Elemente in einem Array KERNELV_CHANNEL_TECHNO_DATA_ARRAY oder KERNELV_CHANNEL_TECHNO_DATA_ARRAY2.

Maximalanzahl der Elemente, die die Funktionen kernelv_ch_get_(new_)techno_data() bzw. kernelv_ch_get_(new_)techno_data2() zurückliefern.

Wert

30

5.23 KERNELV_AXIS_TECHNO_DATA_COUNT

Beschreibung

Anzahl der Elemente in einem Array KERNELV_AXIS_TECHNO_DATA_ARRAY oder KERNELV_AXIS_TECHNO_DATA_ARRAY2..

Maximalanzahl der Elemente, die die Funktionen kernelv_ax_get_(new_)techno_data() bzw. kernelv_ax_get_(new_)techno_data2() zurückliefern.

Wert

30

5.24 KERNELV_ERROR_VALUE_COUNT

Beschreibung

Anzahl der Elemente in einem Array KERNELV_ERROR_VALUE_ARRAY.

Maximalanzahl der Elemente, die die Funktion kernelv_get_error_message_values() zurückliefert

Wert

5

5.25 KERNELV_INSTANCE_PREFIX_MAX_LEN

Beschreibung

Maximale Länge der Zeichenkette die als Präfix für den Aufruf der Funktion kernelv_startup_instance() übergeben werden darf.

Wert

60

6 Support und Service

Beckhoff und seine weltweiten Partnerfirmen bieten einen umfassenden Support und Service, der eine schnelle und kompetente Unterstützung bei allen Fragen zu Beckhoff Produkten und Systemlösungen zur Verfügung stellt.

Downloadfinder

Unser Downloadfinder beinhaltet alle Dateien, die wir Ihnen zum Herunterladen anbieten. Sie finden dort Applikationsberichte, technische Dokumentationen, technische Zeichnungen, Konfigurationsdateien und vieles mehr.

Die Downloads sind in verschiedenen Formaten erhältlich.

Beckhoff Niederlassungen und Vertretungen

Wenden Sie sich bitte an Ihre Beckhoff Niederlassung oder Ihre Vertretung für den lokalen Support und Service zu Beckhoff Produkten!

Die Adressen der weltweiten Beckhoff Niederlassungen und Vertretungen entnehmen Sie bitte unserer Internetseite: www.beckhoff.com

Dort finden Sie auch weitere Dokumentationen zu Beckhoff Komponenten.

Beckhoff Support

Der Support bietet Ihnen einen umfangreichen technischen Support, der Sie nicht nur bei dem Einsatz einzelner Beckhoff Produkte, sondern auch bei weiteren umfassenden Dienstleistungen unterstützt:

- Support
- Planung, Programmierung und Inbetriebnahme komplexer Automatisierungssysteme
- umfangreiches Schulungsprogramm für Beckhoff Systemkomponenten

Hotline:	+49 5246 963-157
E-Mail:	support@beckhoff.com

Beckhoff Service

Das Beckhoff Service-Center unterstützt Sie rund um den After-Sales-Service:

- Vor-Ort-Service
- Reparaturservice
- Ersatzteilservice
- Hotline-Service

Hotline:	+49 5246 963-460
E-Mail:	service@beckhoff.com

Beckhoff Unternehmenszentrale

Beckhoff Automation GmbH & Co. KG

Hülshorstweg 20
33415 Verl
Deutschland

Telefon:	+49 5246 963-0
E-Mail:	info@beckhoff.com
Internet:	www.beckhoff.com

Stichwortverzeichnis

Mehr Informationen:
www.beckhoff.de/TF5270/71

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