Albatros

3.2.3

Numerische Steuerung

Tecnologie e Prodotti per l'Automazione

Inhaltsverzeichins

1 Einleitung

1.1 Bestimmungsgemäßer Gebrauch

Die vorliegende Anleitung beschreibt die Funktionen der numerischen Steuerung Albatros.
Ziel dieser Anleitung ist, den Bediener zum Verständnis des Systems und seiner Verwendung anzuleiten.

Die Hauptpunkte der einzelnen Abschnitte dieser Gebrauchsanleitung sind die folgenden:
- Die Fenster und Hilfsmittel von Albatros.
- Die Beschreibung der typischen Architektur eines Albatros-Systems.
- Die Anzeige der Vorrichtungen und deren Bearbeitung mittels der Funktionen Manuell und Diagnostik unter Zuhilfenahme des synoptischen Schemas.
- Die Anzeige der Technologischen, Geometrischen und Werkzeugparameter sowie deren Bearbeitung und Visualisierung.
- Die Anzeige der Vorrichtungen und deren Bearbeitung mittels der Funktionen Manuell und Diagnostik.

Um den Umfang der vorliegenden Hilfeanleitung in Grenzen zu halten, wird der Leser gebeten, die Handbücher des Windows-Betriebssystems, heranzuziehen, wo der Gebrauch von *Maus, Menü, Symbolleiste* und anderen klassischen Windows-Funktionen vertieft werden kann.

1.2 Die Arbeitsfenster

Es gibt verschiedene Arbeitsfenster, die von den Vorgang, die man ausführen möchte abhängig sind, und die gleichzeitig offen sein können.

Die Fenster sind im Wesentlichen in folgende Typen aufgeteilt:

Fenster	**Beschreibung**
Hauptfenster	Hauptfenster von Albatros. Das dient dem Aufruf von Funktionen, und ist der Behälter der anderen Fenster, deren Inhalt von der jeweiligen Anwendung abhängt.
Synoptisch	enthält eine grafische Darstellung der Maschine oder von Teilen der Maschine, und erlaubt, darauf einzuwirken.
Technologische Parameter	dient der Anzeige und Änderung von technologischen Parametern
Werkzeugparameter	dient der Anzeige und Änderung von Werkzeugparametern.
Diagnostik	dient der Anzeige des Zustandes der Vorrichtungen, und auch darauf einzuwirken, sofern erlaubt.
Systemfehler	Fenster mit der Liste der zuletzt aufgetretenen Systemfehler. Man kann auch Zyklusfehler und Nachrichten angezeigt werden.
Systemkonfiguration	dient der Anzeige und die Bearbeitung der logischen und physischen Maschinenvorrichtungen.

2 Aufbau des Systems

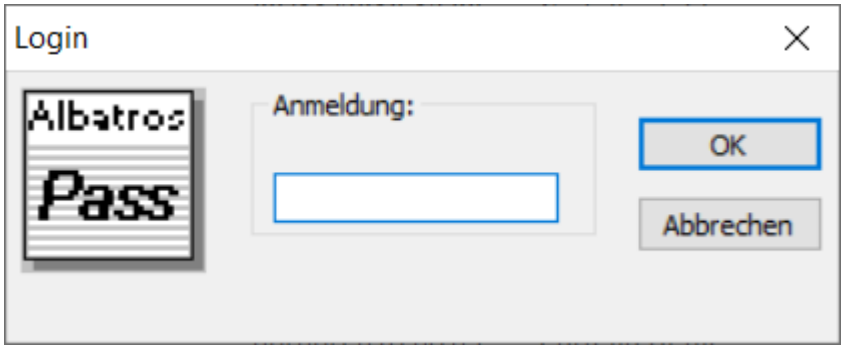
2.1 Zugangsebenen zum System

Albatros sieht vier Zugangsebenen zum System vor:

- Benutzer*:* Es handelt sich um die Ebene mit den meisten Zugangseinschränkungen. Keine der Vorgaben betreffs der Vorrichtungen kann geändert werden. Diese Ebene entspricht den Bearbeitungen und normalen Vorgängen an der Maschine. Beim Einschalten des Systems wird automatisch diese Zugangsebene aktiviert.
- Wartung*:* Es handelt sich um die Ebene für die ordentliche Wartung der Maschine. Der Bediener kann einige Konfigurationsparameter ändern, aber nicht die Struktur der Maschine.
- Hersteller*:* Es handelt sich um die Ebene zur Konfiguration von Anlagen und Maschinen. Auf dieser Ebene können fast alle möglichen Änderungen vorgenommen werden. Diese Ebene wird von Programmierern benutzt.
- Tpa: Es handelt sich um die höchste Zugangsebene zum System. Sie dient dem Schutz von einigen besonders kritischen Einstellungen, deren Änderung eine vertiefte Kenntnis von Albatros voraussetzt. Wird äußerst selten benutzt. Das Passwort für diese Ebene ist direkt bei TPA zu erfragen.

Um auf einer höheren als der Benutzerebene Zugang zum System zu erhalten oder um das System wieder zur Benutzerebene zu bringen, nachdem auf einer höheren Ebene Änderungen vorgenommen worden sind, ist das entsprechende Passwort notwendig.

Um das Fenster zur Eingabe des Passwortes aufzurufen, verwenden Sie den Shortcut **Strg+ *** (Stern). Alternativ befindet sich rechts auf der **Taskleiste** von Windows das Symbol. Durch Klicken auf das Symbol mit der rechten Maustaste kann ein Menü, in dem **Change pass level** erscheint, veranschaulicht werden.
Es erscheint folgendes Fenster:



Login-Fenster

Nun ist das Passwort einzugeben und durch Druck auf **[OK]** zu bestätigen. Anstelle der Zeichen des Passwortes erscheinen die Zeichen "*", so dass niemand das gerade eingegebene Passwort lesen kann. Sofort nach Eingabe des Passwortes befindet man sich auf der entsprechenden Zugangsebene. Um die Zugangsebene bestätigt zu bekommen, wählen Sie die Option **Informationen über Albatros** im Menü **Hilfe**.

Bei Eingabe eines falschen Passwortes erscheint die Fehlermitteilung "Achtung! Falsches Passwort!".

2.2 Mehrsprachige Unterstützung

Albatros erlaubt die Anzeige des Textes in verschiedenen Sprachen.

Sprachwechsel
Der Sprachwechsel kann auf jeder beliebigen Zugangsebene zum System erfolgen. Um die gewählte Sprache zu ändern, verwenden Sie den Shortcut **Strg** **+/** oder wählen Sie innerhalb der "**Anwendungsleiste**" von Windows das Symbol  aus. Nun wählen Sie die Sprache aus und klicken Sie die Schaltfläche **[OK]** an.
Der Sprachwechsel erfolgt nicht sofort, sondern jedes Mal, wenn Albatros neu gestartet wird.

2.3 Typische Architektur eines Albatros-Systems

Da zahlreiche Aspekte sowohl der graphischen Darstellung als auch der Struktur der Grunddaten der Maschine - zusätzlich zu einigen allgemeinen Angaben - stark vom Maschinentyp abhängig sind, stellt die vorliegende Hilfeanleitung eine beispielhafte Beschreibung eines typischen Systems zur Verfügung

Die tatsächlichen Angaben, die Schemata sowie die Grafikseiten des wirklichen Systems hängen logischerweise von der jeweiligen Anwendung ab und obliegen daher dem Hersteller der Werkzeugmaschine.

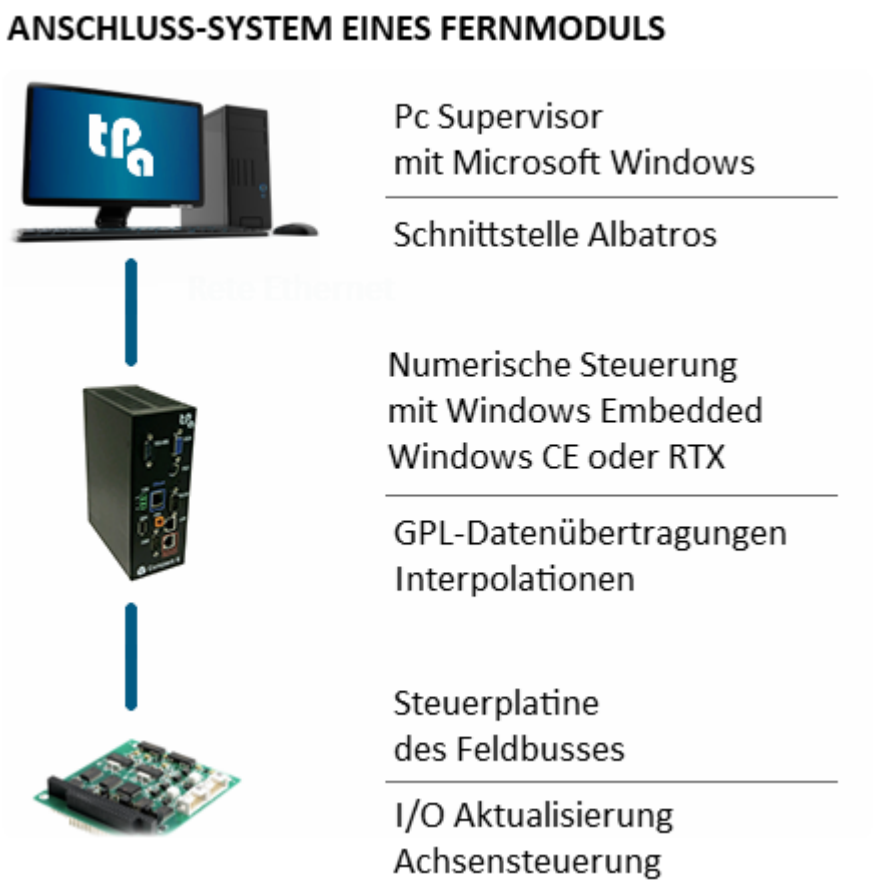
Die numerische Steuerung Albatros besteht aus einer auf einem PC basierenden Steuereinheit, das die Schnittstelle zwischen Bediener und Maschine darstellt, und einer veränderlichen Zahl von Modulen (von 1 bis 16) zur Verwaltung und Kontrolle aller Betriebsmittel der Werkzeugmaschine oder der Anlage.

Es sind also zwei Versionen möglich:

Monomodul besteht aus einem einzigen Modul, das direkt an den PC-Bus angeschlossen ist.

Multimodul besteht aus mind. 1 und max. 16 Modulen und ist normalerweise bei Anwendungen auf Anlagen oder Linien mit mehreren Maschinen vorgesehen; die PC-Einheit ist in diesen Fällen örtlich von den Mainframe-Modulen getrennt, die an verschiedenen Punkten der Linie oder der Anlage aufgestellt sein können.

In beiden Versionen bestehen die Mainframe-Module aus Multiprozessor-Platten zur direkten Kontrolle der Maschinenachsen und zur logischen Verwaltung der Ein- und Ausgänge.
Bei der Monomodul-Version sind die Platinen der Achsen direkt im Supervisor-PC installiert, während sie bei der Multimodul-Version in einem industriellen PC (mit oder ohne Bildschirm und Tastatur) installiert sind, der per Ethernet-Netz mit dem Supervisor-PC verbunden ist. Folgende Abbildung zeigt ein Anschluss-Schema zwischen einem Supervisor-PC und einem Fernmodul (Clipper). Außerdem sind die Haupttätigkeiten der verschiedenen Komponenten angegeben.



Intelligente Fernvorrichtungen steuern E/A-Vorrichtungen und Achsen (Fern-TRS-AX) unmittelbar an der Maschine. Diese Vorrichtungen dienen dem Ablesen der digitalen oder analogen Eingangsleitungen (ON/OFF) sowie der Auffrischung der digitalen oder analogen Ausgangsleitungen, und sind mittels GreenBUS (seriellem Bus RS485 - 1 Mbaud), Bus CAN und EtherCAT mit den Modulen verbunden.
Der Funktionsablauf in der Maschine von Albatros ist von einem USB - Hardwareschlüssel geschützt, der von TPA konfiguriert wird.

2.4 Organisation und logische Konfiguration

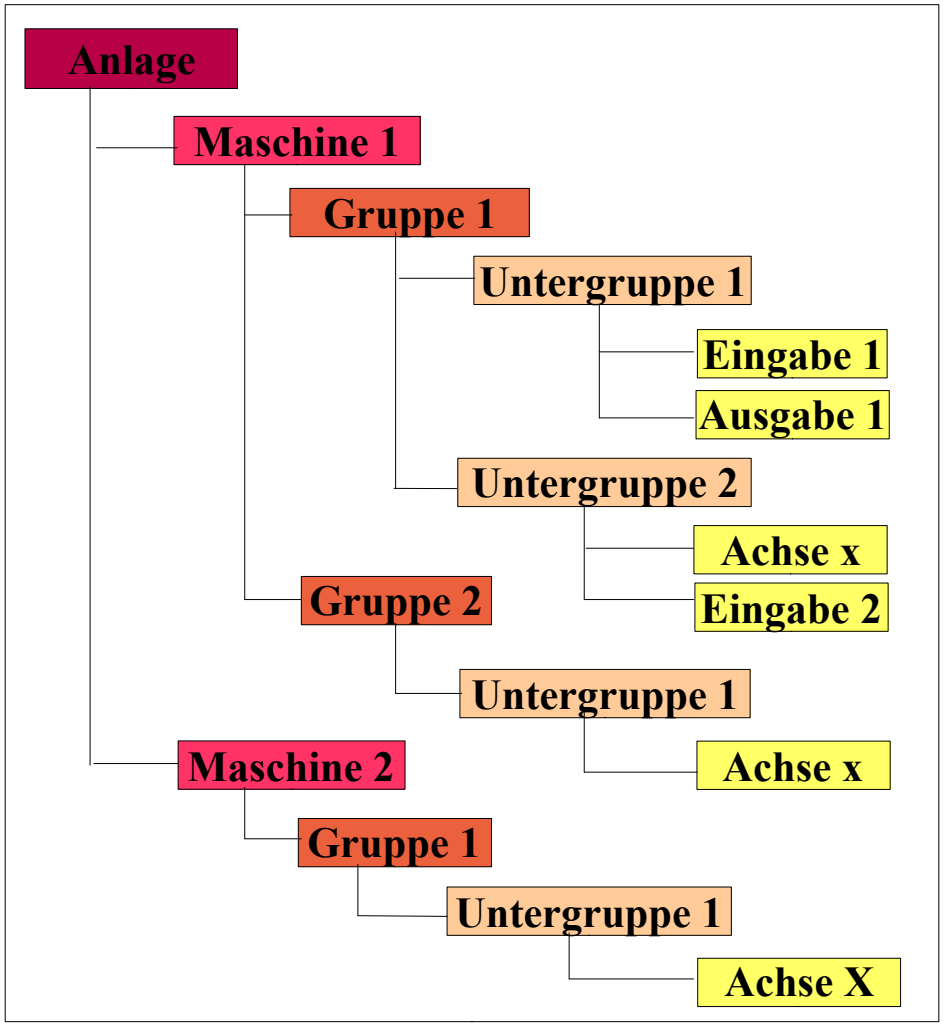
Im System Albatros ist die Beschreibung der Anlage oder der einzelnen Werkzeugmaschine in einem technologischen Archiv mit hierarchischer Baumstruktur organisiert.

Dieser Aufbau beruht auf der Anforderung, die etwaige modulare Struktur der Maschine bezüglich Konfigurationsdaten und Zugangsmodalitäten zu bewahren, indem sie nach dynamischer Zusammenstellung von verschiedenen Modulen, Aggregaten und Vorrichtungen klassifiziert wird, die anwendungsspezifisch eingesetzt oder ausgeschlossen werden können.

Entsprechend dieser logischen Struktur ist die oberste hierarchische Ebene im allgemeinsten und komplexesten Fall wie folgt zusammengesetzt:

1. **Anlage** nichts weiter als verschiedene Maschinen. Es handelt sich um die Summe der Betriebselemente, die von der numerischen Steuerung verwaltet werden. Die Anlage ist immer vorhanden, auch bei einer einzigen Maschine, und braucht nicht definiert zu werden.
2. **Maschine** wird von einem "logischen" Gesichtspunkt aus wie eine Summe von Vorrichtungen (Achse, Timer, usw.) und Kontrollzyklen betrachtet, d.h. eines GPL-Codes, der die Kontroll-Logarithmen der Maschine selbst implementiert. Normalerweise sind in einer Maschine zahlreiche Vorrichtungen vorhanden, die in Gruppen geordnet sind.
3. **Gruppen** sind "Behälter", die der Organisation der Komponenten einer Maschine nach logischen Kriterien dienen. Man kann z.B. eine Gruppe "Achsen" definieren, die alle Achse der Maschine, die Endschalter, die Zyklen zur Ausführung des Nullstellungs der Achsen, usw. enthält.
4. **Untergruppen** sind eine weitere Unterteilung einer Gruppe. Die Gruppe "Achsen" könnte z.B. in "digitale Achsen" und "Achsen im Schrittbetrieb" unterteilt werden.
5. **Vorrichtungen** sind die unterste Ebene der Hierarchie. Sie sind eine von der Hardware unabhängige, logische Darstellung der elektrischen und mechanischen Bauteile der Maschine.

Folgende Abbildung zeigt schematisch die Struktur einer erfundenen, aus zwei Maschinen bestehenden Anlage:



Beispiel der hierarchischen Organisation des Systems.

ANMERKUNG: Die Gruppen müssen nicht unbedingt in Untergruppen aufgeteilt sein und können auch direkt aus Vorrichtungen bestehen.
Um in einer Anlage mit mehreren Maschinen gewisse Funktionen wie Diagnostik, Systemkonfiguration, Parameter aufzurufen, ist es notwendig, die Maschine zu wählen, deren Daten man einsehen möchte.

2.5 Vorrichtungen

Die Vorrichtungen können nach zwei Kategorien gruppiert werden: Physikalische Vorrichtungen und logische Vorrichtungen. Im System ist jeder Vorrichtung ein Name zur Kennzeichnung ihres Gebrauchs zugeordnet.

Physikalische Vorrichtungen

Unter physikalischen Vorrichtungen versteht man all die Elemente, die auf die elektrischen oder pneumatischen Maschinenteile einwirken oder deren Zustand übermitteln. Diese sind:

Symbol	***Vorrichtung***	***Funktion***
	Digitaler Eingang	liest den "AN" oder "AUS"-Zustand einer Vorrichtung ab. Z.B. den Sicherheitsschalter einer Schutzabdeckung.
	Digitaler Ausgang	Aktiviert oder deaktiviert eine Vorrichtung, die sich also entweder im Zustand "AN" oder "AUS" befindet. Er kann z.B. zur Steuerung eines Elektromagnetventils eingesetzt werden.

	*Analoger Eingang*	liest den Wert einer Eingangsspannung an der jeweiligen Klemme ab, z.B. den vom Tacho-Dynamo hervorgerufenen Wert.
	*Analoger Ausgang*	verleiht der jeweiligen Klemme eine Ausgangsspannung. Er kann z.B. zur Steuerung eines Inverters verwendet werden.
	*Eingangs-Port*	besteht aus acht digitalen Eingangsleitungen
	*Ausgangs-Port*	besteht aus acht digitalen Ausgangsleitungen
	*Achse*	verwaltet die Bewegung einer elektrischen Achse. Es können Achsen verschiedener Art verwaltet werden: analoge, digitale, Schrittmotoren, Zählachsen (nur Encoder ablesen).

Logische Vorrichtungen

Die logischen Betriebsmittel sind Elemente, die ausschließlich innerhalb der Arbeitsprogramme funktionieren und der Ausführung aller Trieb- und Kontrolleingriffe auf physikalischen Vorrichtungen dienen:

Symbol	**Vorrichtung**	**Funktion**
	*Timer*	Element, das die Zeit zählt. Maßeinheit ist die Sekunde. Die Auflösung beträgt 4 ms. Kann nur positive Werte ausdrücken, wobei der Höchstwert 8.589.934 Sekunden beträgt (99 Tage ungefähr) (Realtime bei 250 Hz). Sein Wert wird im nichtflüchtigen Speicher der Achsenplatine gespeichert.
	*Zähler*	Element, das die Vorgänge zählt. Kann einen zwischen -2.147.483.648 und +2.147.483.647 begriffenen Wert annehmen. Sein Wert wird in dem nichtflüchtigen Speicher der Achsenplatine gespeichert.
	*Flag Bit*	Element, das den Wert "an" oder "aus" annehmen kann
	*Flag Switch*	Es handelt sich um besondere Flags, die einigen Schaltflächen der Symbolleiste zugeordnet werden können, wie z.B. das Startflag.
	*Flag-Port*	besteht aus 8 Flag-Leitungen.
	*Variable*	globale Variable des Typs *Integer* des GPL-Codes
	*Variable*	globale Variable des Typs *Char* des GPL-Codes
	Variable	globale Variable des Typs *Float* des GPL-Codes
	*Variable*	globale Variable des Typs *Double* des GPL-Codes
	Variable	globale Variable des Typs *String* des GPL-Codes
	*Variable*	globale Variable des Typs *Array* des GPL-Codes
	*Variable*	globale Variable des Typs *Matrix* des GPL-Codes

3 Synoptische Darstellung

3.1 Gebrauch der synoptischen Darstellung

Während der Ausführungsphase der Maschine kann das Fenster *Synoptische Darstellung* geöffnet werden, in dem der Zustand der wichtigsten Vorrichtungen angezeigt wird.

Die Informationen in synoptischer Darstellung sind dieselben wie im Fenster Diagnostik. In letzterer werden die Informationen jedoch in einer Baumstruktur (mit allen Vorrichtungen der Maschine) dargestellt, während das synoptische Schema die Informationen grafisch darstellt (z.B. durch Anzeige eines Bildes von der Maschine und Angabe der Achsenhöhen in der Nähe der jeweiligen Achse). Außerdem können die wichtigsten Informationen ausgewählt und die weniger wichtigen in andere, vom Benutzer aufrufbare Bildschirmseiten gruppiert werden.

3.2 Bedienung der synoptischen Darstellung

Der Bediener kann zu Diagnosezwecken durch doppeltes Klicken mit der Maus auf eines der von einem Rechteck hervorgehobenen Maschinenfelder, sog. "heiße Zonen", die entsprechende Fläche der Maschine wählen und auf einer Extra-Seite anzeigen.
Um auf eine "heiße Zone", eine Vorrichtung oder eine Achse zu weisen, ist es ausreichend, sich mit der Maus auf dessen Abbildung zu bewegen. Während man die Maus bewegt, erscheint auf der Statusleiste der Name der Vorrichtung, auf die der Zeiger der Maus weist.

Der Zeiger der Maus nimmt außerdem verschiedene Formen an, je nachdem auf welche Gegenstände er weist und gibt dadurch die jeweils am Gegenstand zugelassenen Eingriffe an:

 *Vergrößerungsglas*	wenn es eine "heiße Zone" ist
 *Hand*	wenn es eine Ausgangsvorrichtung ist
 *Textkursor*	wenn es ein Feld ist, in das ein Wert eingegeben werden kann

3.3 Einwirken auf die Vorrichtungen

Um auf die Vorrichtungen einzuwirken, mit dem Zeiger der Maus darauf weisen und je nach Vorrichtung wie nachfolgend angegeben weitermachen:

Darstellungsart	**Vorgang**	**Vorrichtung**
Vorrichtungssymbol	darauf zeigen und *klicken*	Digitaler Ausgang Flag Switch Flat Bit
Feld zur Eingabe eines Werts	zeigen, *klicken* und Wert eingeben	Analog-Ausgang Ausgangs-Port Flag-Port Achsenhöhe Timer Zähler

3.4 Manuelle Achsenbewegung

Um die Funktion der manuellen Achsenbewegung aufzurufen, ist die passende Zugangsebene notwendig. Diese wird vom Hersteller der Maschine vergeben.

Um mit einer der Achsen zu kommunizieren, einfach mit der Maus auf dem Anzeigefeld der Höhen der gewünschten Achse *doppelt klicken*. Damit wird das Fenster der Achsenbewegung geöffnet. Im Fall von Achsen des Typs Virtuell, Schrittschaltung und Zähler enthält das Fenster eine kleinere Anzahl Informationen. Handelt es sich z.B. um eine Achse des Typs Zähler, werden nur die Werte des Echten Maßes und der Geschwindigkeit angezeigt.

Das Fenster ist in folgende zwei Bereiche unterteilt:

Bereich Anzeige

- Drei Felder mit der Anzeige des *absoluten Maßes* der Achse [mm], ihrer *laufenden Geschwindigkeit* sowie des *Loop-Fehlers*.
- Zwei Schaltfelder zur Anzeige des *Achsenzustands* (*Free* = mit offenem Ring, z.B. aufgrund eines Systemfehlers, *Normal* = mit geschlossenem Ring, d.h. im normalen Positionskontrollzustand). Diese Schaltfelder dienen auch der Auswahl des Zustands.
- Die Mitteilung des Achsenzustands während der Bewegungsphasen (z.B. Geschwindigkeitszunahme).
- Zwei Schaltflächen, um die Achsenbewegung in negativer ⊟ oder positiver ⊞ *Richtung* auszuführen.
- Die Schaltfläche ■, um die Achsenbewegung in der Bewegungsart Absolut und Step jederzeit zum Stillstand zu bringen.

Bereich Bewegung

- Zwei Felder zur Einstellung einer *Negativen Höhe* und einer *Positiven Höhe*, die in der Modalität Absolut ausgeführt werden.
- Ein Feld zur Einstellung der Achsen*geschwindigkeit* während der manuellen Bewegungen.
- Drei Schaltfelder zur Auswahl der Bewegungsart unter: *Jog*, *Absolute Höhe* oder *Step.*
- Ein Feld zur Vorgabe des *Schritt*werts in der Bewegungsart *Step*.

Um eine Achse zu bewegen, sind die beschriebenen Parameter passend einzustellen. Die Bewegungsart ist zu wählen und die Schaltfläche ⊞ (zur Bewegung der Achse in positiver Richtung) oder die Schaltfläche ⊟ (zur Bewegung der Achse in negativer Richtung) zu drücken.
In der Bewegungsart *Jog* bewegt sich die Achse, solange die Schaltfläche ⊞ oder ⊟ gedrückt werden.
In der Bewegungsart *Step* bewegt sich die Achse um den im Feld "Schritt" angegebenen Wert, jedes Mal wenn die Schaltfläche ⊞ oder ⊟ gedrückt wird.
In der Bewegungsart *Absolut* erreicht die Achse direkt das im Feld positives oder negatives Maß angegebene Maß.

Alternativ zu den Schaltflächen ⊞, ⊟ und ■ können die Tasten "+" (oder Ctrl+P), "-" (oder Ctrl+M) und "Leertaste" der Tastatur verwendet werden.

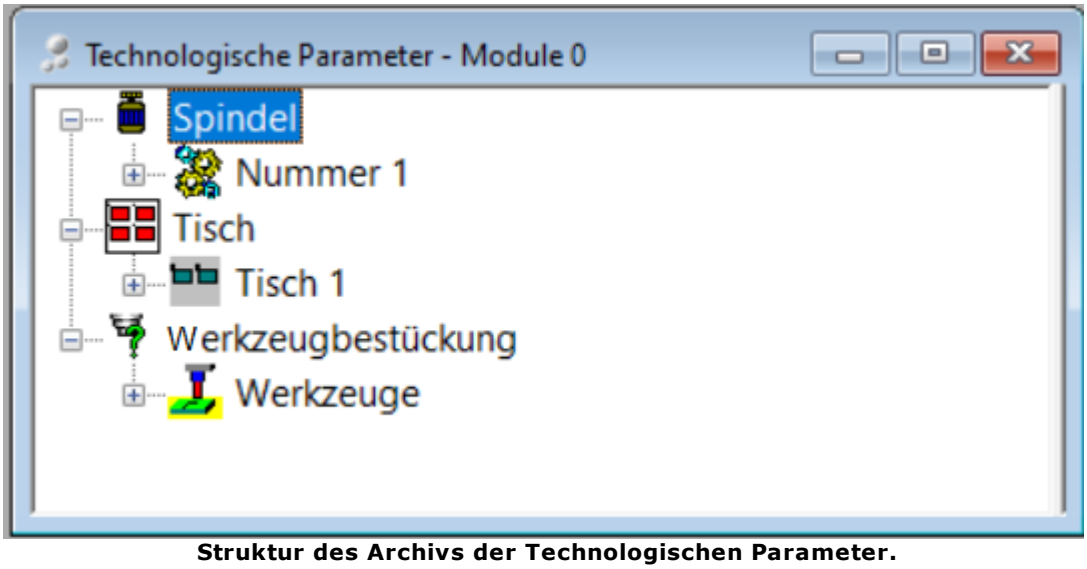
4 Technologische Parameter und Werkzeuge

4.1 Das Fenster Technologische Parameter

Das Archiv der Technologischen Parameter dient dem Speichern von geometrischen und technologischen Informationen einer Maschine. Es handelt sich dabei um Informationen, die der numerischen Steuerung notwendig sind, um den Betrieb der Maschine korrekt verwalten zu können. Das wird von Menü ***Datei->Technologische Parameter öffnen.***

Die Technologischen Parameter der Maschine sind normalerweise in Gruppen/Untergruppen organisiert (Generell sind die Gruppen und Untergruppen der Technologischen Parameter nicht an die Gruppen und Untergruppen der Vorrichtungen der Maschine gebunden). Die Anzeige wird vom Hersteller der Maschine bestimmt und je nach Anwendung unterschieden.
Normalerweise werden die angegebenen Werte während der Einstellung der Maschine vom Hersteller bestimmt und können auch nur in seltenen Fällen vom Endkunden geändert werden. Es kann auch vorkommen, dass gewisse Daten durch ein Passwort geschützt sind, um zufällige Änderungen zu vermeiden, die den korrekten Betrieb des Systems beeinträchtigen könnten.

Das Fenster der Technologischen Parameter zeigt in einer Baumstruktur alle Parameter-Gruppen und -Untergruppen an, die zusammen die technologischen und geometrischen Parameter ausmachen (siehe Abbildung):



Struktur des Archivs der Technologischen Parameter.

Das Fenster zeigt die Parameter-Gruppen und– Untergruppen in einer Baumstruktur an. Die Baumstruktur kann durch Einwirken auf die Schaltflächen ⊞ und ⊟ jedes Knotens erweitert oder reduziert werden. Das Öffnen und Schließen der Baumteile kann auch anhand der Tasten: **+**, **-**, Pfeiltasten nach rechts und nach links.

Bedienung der Technologischen Parameter
Nach Öffnen des Baums an der gewünschten Gruppe / Untergruppe, erscheint die Seite mit den Daten.
Die Daten können in tabellarischer Form oder als Text- bzw. Auswahlfelder erscheinen; das hängt vom Datentyp und von den Vorgaben des Herstellers ab.
Wenn die Daten geändert werden, ist die Schaltfläche **[OK]*** zu drücken, um die Änderungen zu speichern.

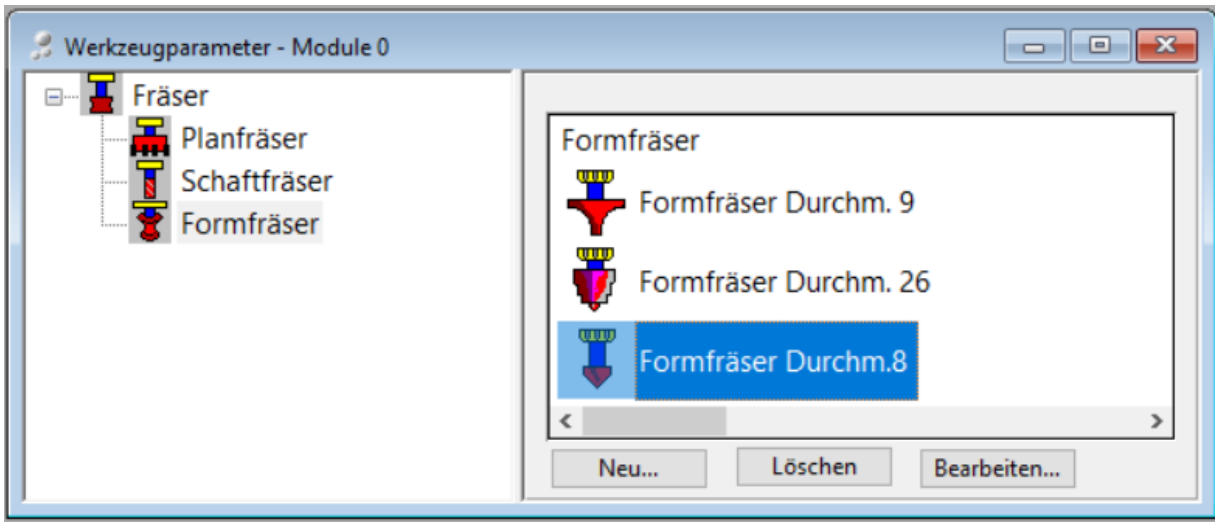
Ausrüstung
Ein Sonderfall unter den Daten einer Maschine ist der der Ausrüstungen. Normalerweise werden die Informationen, die die Werkzeuge der Maschine (Ausrüstung) betreffen, im Archiv der Technologischen Parameter gespeichert. Die Informationen, die die Werkzeuge betreffen, werden hingegen im Archiv der Werkzeugparameter gespeichert.
Um die Ausrüstung zu definieren, ist es also notwendig, die Informationen der zwei Archive zu verbinden. Wenn die Anwendung es vorsieht, wird es also möglich sein, aus dem Archiv der Technologischen Parameter die Informationen des Archivs Werkzeugparameter aufzurufen.

Normalerweise wird die Verbindung mittels einer Schaltfläche hergestellt, deren Symbol der folgenden ähnelt.

Markiert man das Symbol und klickt man mit der linken Taste der Maus doppelt darauf, erscheint ein Fenster mit der Liste der im Archiv Werkzeugparameter definierten Werkzeuge, aus denen das gewünschte Werkzeug gewählt werden kann. Nach dem Markieren wechselt das Symbol der Schaltfläche und zeigt das dem spezifischen Werkzeug zugeordnete Symbol an.
Außerdem können die Daten des Werkzeugs durch Doppelklicken mit der rechten Taste der Maus auf dem Symbol angezeigt werden.

4.2 Das Fenster Werkzeugparameter

Das Fenster der Werkzeugparameter wird vom Menü ***Datei->Werkzeugparameter öffnen*** geöffnet.
Die Werkzeugparameter, die vom Hersteller auf die Bearbeitungsmöglichkeiten der Maschine abgestimmt werden, sind gewöhnlich so strukturiert wie in der folgenden Abbildung angezeigt:



Beispiel eines Fensters Werkzeugparameter.

Das Fenster Werkzeugparameter besteht aus zwei Bereichen:

- Der *linke Bereich* enthält die Werkzeuggruppen mit den zugehörigen Untergruppen im Bild einer Baumstruktur. Die Baumstruktur kann durch Einwirken auf die Schaltflächen ⊞ und ⊟ jedes Knotens erweitert oder reduziert werden. Die Gruppe Fräsen kann z.B. aus Untergruppen von Fräsen mit verschiedenen Eigenschaften wie Profilfräsen, Schwenkfräsen, usw. bestehen. Jeder Untergruppe sind ein oder mehrere Werkzeuge zugeordnet, deren Eigenschaften in einem vom Hersteller definierten Dialogfenster zugewiesen werden. Die in jeder Untergruppe vorhandenen Werkzeuge werden in der rechten Bildschirmhälfte angezeigt.
- Der *rechte Bereich* mit dem Namen der gewählten Untergruppe als Titel, enthält die Symbole der Werkzeuge, die zur Untergruppe gehören. Die an dieser Stelle definierten Werkzeuge sind nicht notwendigerweise auf der Maschine vorhanden. Die Zuordnung von Werkzeug und Position in der Maschine (Ausrüstung) wird normalerweise im Archiv der Technologischen Parameter gemacht.

Bedienung der Werkzeugparameter
Editing-Eingriffe sind anhand der Schaltfelder im unteren Teil des Fensters möglich:

[Neu...] erlaubt das Einfügen eines neuen Werkzeugs in die Untergruppe. Es öffnet das Dialogfeld "Neues Werkzeug", in dem Sie die folgenden Daten zuordnen können:
- *Beschreibung:* Das ist eine Nachricht, die das Werkzeug identifiziert. Die Beschreibung kann aus den bereits in der Liste vorhandenen ausgewählt werden, sofern sie nicht bereits einem anderen Werkzeug zugeordnet wurde, oder Sie können ein neues Werkzeug definieren.
- *Bild:* Das ist ein Symbol, das das Werkzeug identifiziert. Es kann aus den bereits in der Liste vorhandenen ausgewählt werden, oder es kann mit der

Schaltfläche **[Bild]** aus einem Ordner aufgerufen werden. Das Werkzeug wird in der alphabetischen Reihenfolge der Beschreibungen eingegeben.

[Löschen] dient dem Löschen eines Werkzeugs aus einer Untergruppe, sofern bestätigt; die zugehörige Beschreibung wird nicht gelöscht und bleibt für ein anderes Werkzeug verfügbar.

[Bearbeiten...] dient dem Ändern der Beschreibung oder des Symbols des gewählten Werkzeugs; mit der selben Verfahrensweise erscheint dasselbe Fenster wie unter **[Neu...]** beschrieben.

5 Diagnostik

5.1 Das Diagnostikfenster

Während der Ausführungsphase der Maschine kann das *Diagnostik*-Fenster geöffnet werden, wodurch der Bediener den Betrieb der Maschine mittels der Überwachung des logischen Zustands der digitalen E/A-Signale, des Werts der analogen E/A, der Zähler und Timer und evtl. der Bewegung einer Achse prüfen kann.
Außerdem kann je nach vom Hersteller zugewiesener Zugangsebene auch der Zustand der Vorrichtungen geändert werden.

Wenn die Zugangsebene es zulässt, kann man in real time:

- den Zustand (ON/OFF) aller digitalen Ein- und Ausgangssignale anzeigen
- die digitalen Ausgangssignale aktivieren bzw. deaktivieren
- die laufenden Werte (im Intervall +/-10V) der analogen Eingänge anzeigen
- allen analogen Ausgängen einen Wert (im Intervall +/-10V) zuweisen
- eine Achse manuell bewegen und deren Geschwindigkeit, Wert des Schrittes oder absolutes Endmaß auswählen sowie die tatsächliche Position, die Geschwindigkeit und den Loop Fehler anzeigen
- die globalen Variablen anzeigen und ändern

In den folgenden Kapiteln werden die Vorrichtungen und die globalen Variablen sowie deren grafische Darstellung detailliert beschrieben.

ANMERKUNG: Im Diagnostik-Fenster werden ausschließlich die Vorrichtungen angezeigt, die auf der laufenden Zugangsebene aktiviert sind.

5.2 Aufbau des Fensters

Unter Bezugnahme auf die Struktur "Gruppen / Untergruppen", die bereits im Kapitel Systemaufbau, beschrieben worden ist, können die Vorrichtungen abgerufen werden, die dann in einer Baumstruktur angezeigt werden.
An oberster Stelle der Struktur befindet sich die Maschine, dargestellt durch das Symbol.

 , Gefolgt vom Namen und Kommentar.

Die Struktur wird erweitert oder reduziert durch *Klicken* auf die Schaltflächen ⊞ oder ⊟. Das Öffnen und Schließen der Baumteile kann auch anhand der Tasten: **+**, **- Nach-links/rechts-Taste** erfolgen.

Beim Öffnen einer Gruppe erscheinen im Baum:

- Die "Liste der Vorrichtungen" der Gruppe, mit dem Symbol 
- Etwaige zur Gruppe gehörende Untergruppen.

Öffnet man eine der Untergruppen, erscheinen die darin enthaltenen Vorrichtungen.

5.3 Darstellung der Vorrichtungen

Von allen angezeigten Vorrichtungen werden folgende Informationen angegeben:

- Das grafische Symbol;
- Der aktuelle Zustand oder Wert;
- Der Name;
- Der Kommentar.

Im Folgenden wird die grafische Darstellung der Vorrichtungen, deren Typ und was davon in real time angezeigt wird, aufgelistet.

Der Zustand der digitalen Ein- und Ausgänge wird grafisch von einem LED dargestellt, das je nach ein- oder ausgeschaltetem Zustand eine andere Farbe annimmt.

Im Fall von Ports, d.h. von mehreren Linien, die gleichzeitig dargestellt werden (8), erscheint eine LED-Reihe, wobei die erste Linie der Gruppe vom ersten LED rechts dargestellt wird und die letzte Linie vom letzten LED links.

Vorrichtungstyp	Symbol	Zustand	Anzeige in Echtzeit
Digitaler Eingang			Zustand: Aktiv = GRÜN, N. aktiv = GRAU
Digitaler Ausgang			Zustand: Aktiv = ROT, N. aktiv = GRAU
Analoger Eingang		22.000	Laufender Wert
Analoger Ausgang		22.000	Laufender Zahlenwert in Volt
Eingangs-Port			Zustand jeder Linie (als digitaler Eingang). Zustand: Aktiv = GRÜN, N. aktiv = GRAU
Ausgangs-Port			Zustand jeder Linie (als digitaler Ausgang). Zustand: Aktiv = ROT, N. aktiv = GRAU
Achse		100.000	Laufende absolute Position
Timer		12.000	Laufender Wert in Sekunden
Zähler		58	Laufender Zahlenwert
Flag Bit			Zustand: Aktiv = GELB, N. aktiv = GRAU
Flag Switch			Zustand (als Flag Bit). Zustand: Aktiv = GELB, N. aktiv = GRAU
Flag-Port			Zustand jeder Linie (als Flag Bit). Zustand: Aktiv = GELB, N. aktiv = GRAU
Globale Variable		2	globale Variable des Typs Integer des GPL-Codes
Globale Variable		127	globale Variable des Typs Char des GPL-Codes
Globale Variable		50.00000000	globale Variable des Typs Float des GPL-Codes
Globale Variable		200.0000000	globale Variable des Typs Double des GPL-Codes
Globale Variable		Area	globale Variable des Typs String des GPL-Codes
Globale Variable		[256]	globale Variable des Typs Array des GPL-Codes
Globale Variable		[10][3]	globale Variable des Typs Matrix des GPL-Codes

5.4 Abfragen der Vorrichtungen

Aus diagnostischen Gründen kann eine Vorrichtung abgefragt werden, um deren Zustand abzulesen oder deren Wert zu ändern.

Bei einigen Vorrichtungen, wie den Eingängen oder den vom Hersteller geschützten ist dies allerdings nicht möglich; versucht der Bediener einen solchen Eingriff vorzunehmen, erscheint eine Mitteilung, die ihn darauf hinweist.

Nachdem die Vorrichtung gewählt worden ist, mit der Maus doppelt klicken oder die **Eingabetaste** oder die **Leertaste** drücken, um das Fenster abzurufen, in dem der Zustand oder der Wert der Vorrichtung geändert werden können.
Wenn es sich um einen **digitalen Ausgang** oder um ein **Flag Bit** handelt, erscheint kein Fenster, sondern es wird direkt deren Zustand geändert. Das Funktionieren des Ausgangs wird von der geänderten Farbe des entsprechenden LEDs angezeigt.

Im Fall eines **Ausgangs-Ports** oder muss man mit dem Zeiger der Maus auf das LED weisen, das dem gewünschten Ausgang entspricht und doppelt klicken, um dessen Zustand zu ändern.
Das gilt ebenfalls für **Flag Switch** und **Flag-Port**.

Bei **analogen Ausgängen**, **Timern** und **Zählern** erscheint ein Dialogfenster zum Visualisieren des derzeitigen Wertes und zum Vorgeben des neuen Werts für die Vorrichtung.

Das Abfragen einer **Achse** ist im Abschnitt Manuelle Achsenbewegung beschrieben.

5.5 Liste der Tastenkombinationen für die Navigation in einer Baumstruktur

Drucktaste	Beschreibung
Nach-oben-Taste Nach-unten-Taste	verschiebt die Auswahl in die vorherige oder in die nächste Zeile

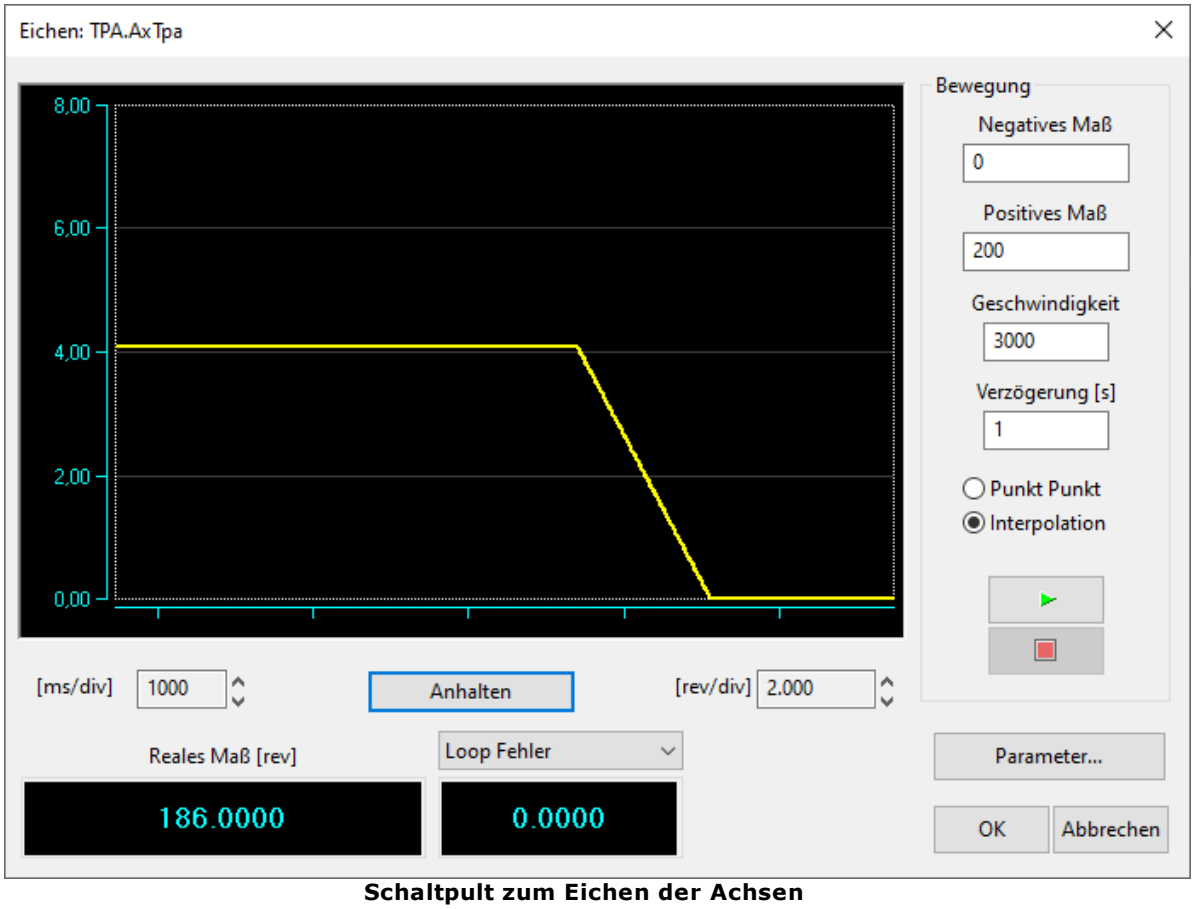
Nach-rechts-Taste	erweitert zu einer Ebene die ausgewählte Verzweigung und, falls sie bereits erweitert worden ist, verschiebt die Auswahl zur nachfolgenden Verzweigung
Nach-links-Taste	schließt die ausgewählte Verzweigung und falls sie bereits geschlossen worden ist, verschiebt die Auswahl in die vorherige Verzweigung
+	erweitert die ausgewählte Verzweigung um eine Ebene
-	schließt die ausgewählte Verzweigung
*	erweitern alle Niveaus der ausgewählten Verzweigung
STRG+Alt+Shift und Eingabe	zeigt die mit einer Achse verbundenen Tabellen der Linearitätskorrektoren. Falls die Tastenkombination aktiviert wird, während im Gerätebaum eines
STRG+Alt+Shift und Linksklick	Moduls eine Achse ausgewählt ist, dann werden alle mit einer Achse verbundenen Korrektoren visualisiert, als wären sie eine Matrix, in der die Spalten die damit verbundenen Achsen einerseits (die erste Spalte ist die der Autokorrektur), und die Zeilen die Korrekturwerte andererseits sind. Die eventuell geänderten Werte sind während der Achsenbewegung übernommen, aber auf der Festplatte nicht gespeichert.

5.6 Linearitätskorrektoren

In Diagnostik können Sie die Linearitätskorrektoren der ausgewählten Achse anzeigen und ändern, indem Sie das Kontextmenü öffnen und **Linearitätskorrektoren** auswählen**.** Das Element ist nur sichtbar, wenn in der Konfiguration die Linearitätskorrektoren für die Achse definiert wurden. Als Alternative zum Menü können Sie die Tastenkombination **[Strg+Umsch.+Eingabe]** verwenden.

5.7 Schaltpult zum Eichen der Achsen

Das Schaltpult zum Eichen der Achsen ermöglicht die Änderung der Konfigurationsparameter einer Achse und die gleichzeitige Bewegung der Achse, deren Verhalten auf einem virtuellen Oszilloskop angezeigt wird.
Um das Schaltpult zum Eichen der Achsen aufzurufen, ist die Zugangsebene "Hersteller" oder höher erforderlich. Zum Aufruf des Eich-Schaltpults aus dem Modus Diagnostik oder Manuell auf der zu eichenden Achse doppelt klicken und gleichzeitig die Taste "shift" gedrückt halten. Um Zugriff auf das Schaltpult der Achsenkalibrierung aus dem Modus Diagnostik oder Manuell zu verwenden, doppelklicken Sie auf die zu kalibrierender Achse, während Sie gleichzeitig die Taste, während Sie gleichzeitig die Taste **[Shift]** gedrückt halten oder wenn sie aus dem Kontextmenü den Eintrag "Eichen" auswählen.
Es erscheint das in folgender Abbildung dargestellte Eich-Schaltpult:



Schaltpult zum Eichen der Achsen

Um das Verhalten der Achse bei sich ändernden Parametern zu prüfen, wird die Achse kontinuierlich zwischen zwei Grenzmaßen namens **Positives/Negatives Maß** bewegt. Außer diesen zwei Parametern ist die **Geschwindigkeit** der Achse anzugeben. In den Anfangsphasen der Eichung wird die Verwendung einer niederen Geschwindigkeit nahegelegt. Es kann auch eine **Verzögerung** zwischen einer Bewegung und der folgenden eingestellt werden.

Im Fenster des Oszilloskops wird die Linie des Loop Fehlers oder einer der anderen Größen der Achse angezeigt. Analog zu Tisch-Oszilloskopen besteht die Möglichkeit, die Größe der Grafik dem Fenster anzupassen oder Details zu vergrößern. Die Maus und die Steuerungstaste ermöglichen die letzte Eichenminute nochmals anzuzeigen, einen oder zwei Cursors zum Abmessen und Überprüfen der Stichprobendaten anzuzeigen, einen Bereich des Diagramms zu vergrößern, um Einzelheiten der Stichprobendaten zu analysieren, Offset und Skalierung sowohl auf den Abszissen als auch auf den Ordinaten zu ändern.

Durch Druck auf die Schaltfläche **Anhalten** kann außerdem die Bewegung der Grafik angehalten werden, um sie in Ruhe zu betrachten, ohne die Achse anhalten zu müssen.
Außer der Grafik werden in zwei Feldern das echte Maß (links) und die grafisch verarbeitete Größe (rechts) angezeigt. Diese kann anhand der ComboBox über dem Anzeigefeld eingestellt werden.

Zur Änderung der Achsenparameter, Schaltfläche **[Parameter...]** drücken. Ein Fenster erscheint, in dem sofort die meisten Achsenparameter geändert werden können. Nach Änderung von einem oder zwei Parametern kann die Änderung durch Druck auf die Schaltfläche **[Übernehmen]** angewendet werden. Mit dieser Schaltfläche können Sie sofort die Auswirkungen einer Änderung des dynamischen Verhaltens der Achse sehen. Falls Sie **[Ok]** drücken, werden die Änderungen weiterverwendet; falls sie dagegen **[Abbrechen]** drücken, so werden die Originalwerte vor der Eröffnung des Fensters **[Parameter...]** wiederherstellt.

Die Parameter, auf die man gewöhnlich einwirkt, sind:

- **Proportional:** Proportionaler Koeffizient
- **Integral:** Integraler Koeffizient
- **Ableitung:** Derivativer Koeffizient
- **Vorschub vorwärts:** Prozentsatz der laufenden Geschwindigkeit, die direkt vom Antrieb stammt (unabhängig vom Loop Fehler)

- **Vorschub vorwärts Beschleunigung:** Prozentsatz des Geschwindigkeitsbezugs, der während der Beschleunigungs- und Verzögerungsphasen direkt vom Antrieb stammt (zusätzlich zum Vorschub vorwärts)
- **Beschleunigung:** Dauer der Beschleunigungsrampe
- **Verzögerung:** Dauer der Verzögerungsrampe

Der Cursor im graphischen Bereich

Der Cursor ist ein Werkzeug, der ermöglicht, einige Daten der Spur zu bemessen und zu visualisieren. Er besteht aus einer senkrechten Linie derselben Farbe der ausgewählten Spur. Man kann sie innerhalb des Graphen mit den Pfeiltasten oder mit der Maus bewegen. In einem Tooltip der senkrechten Linie kann man verschiedene Werte visualisieren, die aus einem Popupmenü ausgewählt werden können, das durch Anklicken der rechten Maustaste in der Nähe des Cursors aufgerufen werden kann.

Man kann im Popupmenü folgende Optionen auswählen:

- **Kanal**: Zeigt die Liste der im Graph gezeigten Spuren und markiert mittels eines Häkchens die Spur, auf die sich der Cursor bezieht. Außerdem ist es auch möglich, eine neue mit dem Cursor zu verbundener Spur auszuwählen.
- **Stil**: Zeigt eine Liste von Daten, die im Tooltip-Rechteck visualisiert werden können.
- **X-Y-Wert**: Zeigt den Wert auf der X-Achse und auf der Y-Achse des Punktes der Spur, in der der Cursor sich befindet.
- **X-Wert**: Zeigt den Wert auf der X-Achse des Punktes, in der sich der Cursor befindet.
- **Y-Wert**: Zeigt den Wert auf der Y-Achse des Punktes der Spur, in der sich der Cursor befindet.
- **Periode**: Zeigt den Wert des Abstands zwischen zwei visualisierten Cursors, wenn die Messung dem X-Achse entlang durchgeführt wird.
- **Spitze-Spitze**: Zeigt den Wert des Abstandes zwischen zwei visualisierten Cursors, wenn die Messung zwischen zwei Punkten der Y-Achse entlang durchgeführt wird.
- **Frequenz**: Zeigt das Inverse des Abstandes der X-Achse entlang.
- **Optionen**: Konfiguriert die grafische Darstellung des Cursors und des damit verbundenen Tooltips.
- **Die Farbe des Kanals verwenden**: Wenn aktiv, wird der Cursor mit derselben Farbe der abgetasteten Daten markiert, auf die er sich bezieht; andernfalls, wird der Cursor zufällig gefärbt.
- **Tipp bei Freigabe ausblenden**: Wenn aktiv, wird der Tooltip nur gezeigt, bis dass die linke Maustaste gedrückt gehalten wird, dann wird er ausgeblendet.
- **Ausrichtung umkehren**: Der Tooltip wird gewöhnlich auf der rechten Seite des Cursors visualisiert. Wird die Option aktiviert, dann wird der Cursor auf der rechten Seite des Cursors visualisiert.
- **Platzierung an der Abtastung**: Wenn aktiv, wird der Cursor nur auf den abgetasteten Werten platziert.

Eichen der Achse
Das Eichen einer Achse ist ein heikler Vorgang, der mit höchster Vorsicht vorzunehmen ist.

Durch die Option CalibSampleTime im Abschnitt [Albatros] in Tpa.ini kann man die Probeannahmezeit der Daten einer Achse für das Kalibrierungsfenster. Der Wert ist in Millisekunden und kann weder kleiner als die Frequenz der Achsenkontrolle noch größer als 100.

Vor dem Eichen der Achse aus diesem Schaltpult sind alle Konfigurationsparameter sowie der Höchstwert der Antriebsgeschwindigkeit einzustellen. Der Spannungswert, dem in Albatros die Höchstgeschwindigkeit entspricht, liegt bei 9 Volt.

Um Schäden an der Maschine aufgrund von falschen Parametern zu vermeiden, empfiehlt sich die Einstellung einer niederen Geschwindigkeit, z.B. 10% der für die Achse vorgesehene Höchstgeschwindigkeit. Auf diese Weise vermeidet man auch bei Vorgabe eines übermäßigen Gewinns heftige Reaktionen der Achse.

Generell wird die erste Eichung für Punkt Punkt - Bewegungen ausgeführt und erst dann die Eichung für Interpolationsbewegungen.

Der erste Schritt besteht in der Einstellung der Beschleunigungs- und Verzögerungszeiten, sofern sie nicht schon in der Konfiguration gesetzt worden sind. Je größer diese Zeiten sind, desto kleiner ist die Beschleunigung, der die Achse untersteht.

Dann ist zur Einstellung eines minimalen Gains zur Bewegung der Achse überzugehen. Dies dient der Überprüfung der korrekten Eichung des Antriebs. Albatros ist so konfiguriert, dass bei der in der Achsenkonfiguration eingestellten Höchstgeschwindigkeit ein Bezug von 9 Volt erscheint. Wird die Achse z.B. bei 10% der Höchstgeschwindigkeit bewegt und ist der Antrieb korrekt geeicht, erscheint auch eine

Bezugsspannung, die 10% des Höchstwertes entspricht, also 0,9 Volt. Wenn nicht diese Bezugsspannung erhalten wird, ist auf den Höchstwert des Antriebs einzuwirken.

Nach Eichung des Antriebs lässt man langsam und vorsichtig den Gain des Position Loops ansteigen. Es ist jedes Mal zu prüfen, dass kein Überschwingen oder Instabilität auftritt. Die Geschwindigkeit in dieser Phase hat immer kleiner oder gleich 10% der Höchstgeschwindigkeit zu sein. In dieser Phase ist es außerdem angebracht, das erhaltene Profil der Geschwindigkeit anhand des virtuellen Oszilloskops vertieft zu betrachten, indem das Bild so weit vergrößert wird, dass die Details gut sichtbar werden.

Wenn die Reaktion der Achse gleichzeitig rasch und stabil geworden ist, kann die Geschwindigkeit nach und nach erhöht werden, wobei jedes Mal das Verhalten der Achse zu prüfen und der Gain, falls nicht zufriedenstellend, anzupassen ist. Gain und Geschwindigkeit sind niemals jäh zu erhöhen. Scheinbar stabile Eichbedingungen bei niederer Geschwindigkeit bleiben bei größerer Geschwindigkeit nicht unbedingt gleich.

Nachdem der optimale Gain-Wert bestimmt worden ist, können, falls erforderlich, der integrative und derivative Koeffizient und schließlich der Vorschub vorwärts progressiv erhöht werden, um den Loop Fehler einzuschränken und unter dem Gesichtspunkt der Genauigkeit der Achse innerhalb akzeptabler Grenzen zu führen. Der Vorschub vorwärts gleicht den Loop Fehler während der Betriebsgeschwindigkeit der Bewegung fast vollständig aus, jedoch nicht während der Beschleunigungs- und Verzögerungsphase. Um den Loop Fehler in diesen Phasen weiter zu senken, kann die Feed Forward Beschleunigung erhöht werden. Normalerweise führen schon sehr niedrige Werte dieses Parameters zu guten Ergebnissen.

Was das Eichen einer Achse für Interpolationsbewegungen angeht, können die bereits für Punkt Punkt - Bewegungen ermittelten Werte verwendet werden, jedoch sind die anderen Achsen der Maschine zu berücksichtigen. Um eine zufriedenstellende Genauigkeit während der Interpolationsbewegungen zu erhalten, ist es notwendig, die Loop Fehler der Achsen abzuwägen. Es ist also notwendig, die Achse mit dem größten Loop Fehler (bei gleicher Geschwindigkeit) zu bestimmen und die Eichung der anderen so zu "verschlechtern" (nur auf Interpolationsparameter beschränkt), dass die Loop Fehler identisch sind.

6 Fehler und Signalisierungen

6.1 Einführung

Albatros verwaltet Fehlerereignisse und Berichte

Systemfehler

Es handelt sich um Fehler, die das System *Albatros* sowohl während der Ausführung der Programme als auch während der Wartungs- und Diagnose-Eingriffe an der Anlage automatisch erfassen kann.

Diese Fehler sind verschiedener Art und gehen von Schwierigkeiten bei der Verwaltung von Achsen bis zu Problemen, die bei der Ausführung des Programms auftauchen können

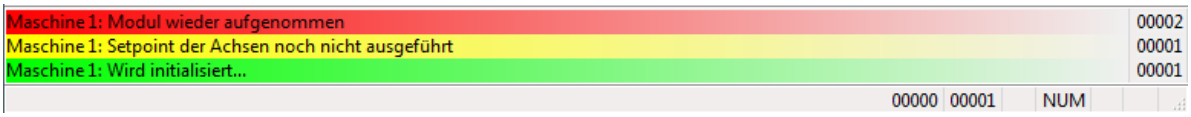
Die Systemfehler können innerhalb der Arbeitsprogramme anhand der Anweisung ONERRSYS, verwaltet werden. Im gegenteiligen Fall wird bei deren Erscheinen die Ausführung der Programme am Modul, wo sich der Fehler ereignet hat, beendet. Die Erklärung jedes Systemfehlers wird auf den folgenden Seiten dieses Handbuchs beschrieben.

Zyklusfehler
Es handelt sich dabei um Fehler, die während der Ausführung des Programms auftreten, die es aber in der Regel erlauben, das Programm nach der Fehlerbeseitigung fortzusetzen. Zyklusfehler können mit der GPL-ERROR - Anweisung erzeugt werden.

Nachricht
Es handelt sich dabei um Warnmeldungen, die in bestimmten Situationen während der Programmausführung erzeugt werden, oder um Signale einer Anforderung zum Eingreifen des Bedieners, die jedoch die Programmausführung nicht stoppen. Nachrichten können durch die GPL-Anweisung MESSAGE generiert werden.

Berichte
Es handelt sich um allgemeine Berichte, die nicht mit den Zyklen der Maschine zusammenhängen und von Albatros geschrieben wurden. Sie werden in Albatros nicht angezeigt. Die Beschreibung der einzelnen Berichte finden Sie auf den folgenden Seiten dieses Handbuchs.

Fehlerleiste
In der Fehlerleiste erscheint der zuletzt aufgetretene Systemfehler zusammen mit dem letzten Zyklusfehler und der letzten Nachricht.
Systemfehler sind rot gekennzeichnet.
Zyklusfehler sind gelb gekennzeichnet. Es handelt sich um Fehler, die während der Ausführung des Programms auftreten, aber diese nicht unterbrechen, sofern die Fehler beseitigt werden.
Nachrichten sind grün gekennzeichnet.



Fehlerleiste

Die Fehler, die ab dem Einschalten des Systems aufgetreten sind, können in einem Fenster angezeigt werden, das mittels Doppelklicks der Maus auf der *Fehlerleiste* oder aus dem Menü ***Ansicht*** geöffnet wird. In diesem Fenster werden außerdem zusätzliche Informationen zu den Systemfehlern angezeigt.

Das Fenster ist in folgende Bereiche unterteilt: Im **oberen Teil** sind folgende Informationen angezeigt:

- Typ bezeichnet den Fehlertyp (Systemfehler, Nachricht- und Zyklusfehler)
- Zeit&Datum sind die Uhrzeit und das Datum, zu dem der Fehler aufgetreten ist.
- Beschreibung ist die Beschreibung des Fehlers.
- Code ist die Nummer der Fehlermitteilung.
- Task ist der Name des Tasks, der den Fehler hervorgerufen hat (nicht in der Fehlerleiste vorhanden).

Durch *Doppelklick* auf eine dieser Spalten werden die Informationen aufgrund des Inhalts der Spalte geordnet.
Im **unteren Teil** befinden sich die Felder:

- Zyklusfehler wenn aktiviert, werden auch diese Fehler angezeigt.
- Nachrichten wenn aktiviert, werden auch die Nachrichten angezeigt.

- Alle wenn aktiviert, erscheinen die Nachrichten aller Module des Systems in Bezug auf die angezeigten Informationen.
- Feld des Modulnamens zeigt den Namen des Moduls, dessen Anzeigen erfolgen und erlaubt außerdem bei einem System mit mehreren Modulen die Wahl des Moduls, dessen Informationen anzuzeigen sind.

Schließlich gibt es folgende **Befehlsschaltflächen**:

- **[Alle entfernen]** löscht alle angezeigten Informationen aus dem Speicher, aber nicht aus dem Archiv.
- **[Entfernen]** löscht die laufende Information aus dem Speicher, aber nicht aus dem Archiv.
- **[OK]** schließt das Fenster.

Speichern von Fehlern und Berichten in einer Berichtsdatei
Alle Fehler werden in einer Datei zur historischen Rekonstruktion gespeichert. Dies ist eine Textdatei im TSV-Format. Der Dateiname lautet MONTH (laufende Monatsnummer).TER. In der Albatros-Suite wird ein ViewRER-Programm bereitgestellt, das die .TER-Dateien lädt und anzeigt.

6.2 Systemfehler

6.2.1 Von der Achsenverwaltung ausgelöste Fehler

1 Achsenname: Falscher Encoder-Anschluss

Ursache:
Bei stehender Achse hat sich ein Unterschied zwischen theoretischer und realer Achsenhöhe ergeben, die größer als 1024 Encoderschritte ist.
Dies geschieht meist, während der Inbetriebnahme der Achse, wenn die Phasen des Encoders umgekehrt sind. Während des normalen Betriebs geschieht dies, wenn die Achse bei ausgeschaltetem Antrieb manuell bewegt wird, ohne vorher auf FREE gesetzt worden zu sein, oder wenn die Achse aufgrund falschen Eichens beim Erreichen des Maßes überschwingt (Overshoot).
Nach diesem Fehler wird das Bezugssystem auf Null gestellt und die Achse wird in den Zustand FREE versetzt.

Lösung:
Während der Inbetriebnahme der Achse, prüfen Sie den Anschluss der Encoder-Phasen der betroffenen Achsen, (aktivieren Sie eventuell die Option zur Umkehr der Encoder-Phasen in der Achsenkonfiguration). Prüfen Sie die Eichung der Achse anhand der dafür vorgesehenen Funktion in der Diagnostik.

2 Achsenname: Bewegung nicht beendet

Ursache:
Am Ende der Bewegung, 5 Sekunden nach Abschluss der theoretischen Bewegung war der Unterschied zwischen theoretischer und realer Achsenhöhe größer als der im Konfigurationsfenster angegebene. Es kann einfach daran liegen, dass der Antrieb aus oder nicht aktiv ist oder an einer schlechten Einstellung des Offsets des Antriebs. Es kann allerdings auch von mechanischem Spiel an der Achse oder vom extrem niedrigen Gewinns des Positions-rings der Achse abhängen

Lösung:
Prüfen Sie, ob der Antrieb an und aktiv ist. Die Eichung der Achse prüfen und den Offsetwert des Antriebs der betroffenen Achse einstellen.

3 Achsenname: Servoerror

Ursache:
Während jeder Art von Bewegung ist der Unterschied zwischen theoretischer und realer Achsenhöhe größer geworden als die maximale in Konfiguration angegebene Abweichung oder als die mit der Anweisung SETMAXER gesetzte Abweichung.
Dieser Fehler hängt normaler-weise von der fehlerhaften Einstellung des Gewinns des Positionsrings oder vom Vollausschlag der Geschwindigkeit oder von einer zu großen Achsenträgheit ab.

Lösung
Prüfen Sie die Einstellung des Gewinns und den Vollausschlagswert der Antriebgeschwindigkeit.
Prüfen Sie die korrekte Funktionsweise des Encoders und der Einheit Motor/Antrieb.
Außerdem, prüfen Sie etwaige mechanische Blockierungen.

4 Achsenname: Positive Grenze überschritten

Ursache:
Die theoretische Achsenhöhe hat das in der Konfiguration festgelegte, positive Grenzmaß überschritten oder das mit der Anweisung SETLIMPOS gesetzte Grenzmaß.

Lösung:
Korrigieren Sie im Programm die Höhe, die über das positive Grenzmaß geht.

5 Achsenname: Negative Grenze überschritten

Ursache:
Die theoretische Achsenhöhe hat das in der Konfiguration festgelegte, negative Grenzmaß überschritten oder das mit der Anweisung SETLIMNEG gesetzte Grenzmaß.

Lösung:
Korrigieren Sie im Programm die Höhe, die über das negative Grenzmaß geht oder die Grenzmaße der Achse neu definieren.

10 Achsenname: Die Echtzeit-Ausfürung ist schneller als die Profilbildung

Ursache:
Die Realtime-Ausführung des Bewegungsprofils ist schneller als die GPL-Erzeugung desselben Profils. Der Lookahead wird schneller als seine Erfüllung entleert. Der Fehler kann zwei in der Regel gleichzeitige Ursachen haben:
- Die Interpolationsgeschwindigkeit ist zu hoch bezüglich der zurückzulegenden Streckenlänge.
- Die zurückzulegenden Strecken sind zu kurz.

Lösung:
Prüfen Sie die programmierte Interpolationsgeschwindigeit, damit sie nicht im Bezug zu den zurückzulegenden Strecken zu hoch ist. Es ist auch zu überprüfen dass die zurückzulegenden Interpolationsstrecken nicht zu kurz sind.

6.2.2 Von der Verwaltung entfernter E/A ausgelöste Fehler

2049 EmpfängerNummer: Falsche Konfiguration

Ursache:
Der Fernempfänger hat eine andere Konfiguration von E/A-Erweiterungen im Vergleich zu den abgelesenen erhalten. Dies kann auftreten, wenn das Ferngerät ungleich ist dem, der in der Hardware-Konfiguration Albatros ausgewählt wurde. Zum Beispiel, der Fern-Empfänger ist ein Albre16 und in Albatros wurde ein Fern-Albre24 (GreenBUS v.3.0) konfiguriert, oder noch ein TRS-IO mit einer falschen Erweiterungszahl TRS-IO-E (GreenBUS v4.0).

Lösung:
Prüfen Sie die Hardware Konfiguration.

2050 EmpfängerNummer: Getrennt

Ursache:
Der Fern-Empfänger folgt nicht den Befehlen des Senders.

Lösung:
Prüfen Sie die Stromzufuhr und den seriellen Anschluss des Empfängers.

2051 EmpfängerNummer: Wieder angeschlossen

Ursache:
Die Verbindung zwischen Sender und Empfänger ist wiederhergestellt worden.

2052 EmpfängerNummer: Fehler beim neuen Lesen des nicht angeschlossenen Ausgang AusgangNummer

Ursache:
Der angegebene digitale Ausgang hat das Ansprechen des Überstromschutzes oder einen Kurzschluss hervorgerufen; auf jeden Fall befindet er sich in einem anderen Zustand als von der Steuerung vorgesehen. Der Ausgang ist in der virtuell-physikalischen Konfiguration keiner logischen Vorrichtung zugeordnet, was Zeichen einer Abweichung zwischen Konfiguration und tatsächlicher Verkabelung der Maschine ist.

Lösung:
Prüfen Sie die Virtuell-physikalische Konfiguration. Den Kurzschluss beseitigen oder prüfen Sie, dass die angeschlossene Last die Höchstgrenzen nicht überschreitet (technische Dokumentation zu Rate ziehen).

2054 EmpfängerNummer: Falscher Typ

Ursache:
Während der Initialisierung der Fern-Empfänger ist an einer bestimmten Adresse ein anderer Fern-Empfänger ermittelt worden als in der Konfiguration angegeben.

Lösung:
Prüfen Sie, dass die Hardwarekonfiguration den Einstellungen der Fern-Empfänger entspricht.

2055 EmpfängerNummer: Initialisiert

Ursache:
Der Fern-Empfänger ist wieder mit dem Sender verbunden, nachdem die Verbindung aufgrund eines Ausfalls der Stromzufuhr abgebrochen war.

2056 EmpfängerNummer: Stormversorgungsfehler +24 Vcc

Ursache:
Die Niederspannung-Stromzufuhr (+24 Vcc) eines E/A-Fernmoduls ist nicht aktiv oder funktioniert nicht einwandfrei.

Lösung:
Prüfen Sie das Funktionieren der Stromzufuhr +24 Vcc.

2057 GreenBUS-Stormversorgungsfehler

Ursache:
Die Stromzufuhr des Busses, der die E/A-Fernmodule mit der Steuerung verbindet, funktioniert nicht einwandfrei. Diese Stromzufuhr muss einen nominalen Wert von +12 Vcc ausweisen und wird von der Steuerung geliefert.

Lösung:
Prüfen Sie, ob die GreenBUS-Versorgung anwesend ist, überprüfen Sie die GreenBUS-Käbel. Eventuell, tauschen Sie die Achsenplatine aus/ein.

2058 EmpfängerNummer: Fehler beim erneuten Lesen VorrichtungTyp VorrichtungName

Ursache:
Der Zustand des angegebenen Ausgangs entspricht nicht der Einstellung. Das kann an einem Kurzschluss, am Anspringen des Schutzes (übermäßige Last) oder einfach am Fehlen der Niederspannung-Stromzufuhr. Der angegebene Ausgang kann ein Digitalausgang, ein Analogausgang, ein Ausgang der Achsenkontrolle sein. Bei der Fehleranzeige wird der Ausgangstyp spezifiziert.

Lösung:
Bei einem digitalen Ausgang: Prüfen Sie die Stromversorgung +24V (Feldseite). Entfernen Sie den eventuellen Kurzschluss oder die übermäßigen Ausgangsabsorption (technische Dokumentation nachschlagen). Bei einem Analogausgang oder einem Ausgang der Achsenkontrolle: Prüfen Sie das Dasein und den Wert der auf dem Ausgang eingestellten Spannung (Tester oder Oszilloskop).

Entfernen Sie den eventuellen Kurzschluss oder die übermäßige Ausgangsabsorption (technische Dokumentation nachschlagen Dokumentation). Prüfen Sie das Funktionieren der Stromzufuhr +24 Vcc, eventuell den Kurzschluss beseitigen oder prüfen Sie, dass die Last die Höchstgrenzen nicht überschreitet (technische Dokumentation nachschlagen).

2059 Test des Dual-Port-Memory des Senders fehlgeschlagen

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten. Insbesondere ist die Initialisierung des Senders GreenBUS (Mikrosteuerung i296) gescheitert. Es kann an einer falschen Konfiguration der E/A und IRQ-Adressen der Platine oder an einem Konflikt mit anderen entfernten Einheiten des Systems liegen. Es kann auch die Folge einer Beschädigung der Achsenplatine sein.

Lösung:
Prüfen Sie die Konfiguration der Platine, schließen Sie die Konflikte mit anderen entfernten Einheiten aus. Verwendet man ein Fernmodul, ist die Firmware neu ans Modul zu übertragen. Fachleute können einen Hardware-Test des Dual-Port-Memory der Mikrosteuerung i296 ausführen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

2060 Fehler beim Initialisieren des Senders

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten. Insbesondere ist die Übertragung der Firmware an den Sender GreenBUS (Mikrosteuerung i296) gescheitert. Es kann an einer falschen Konfiguration der E/A und IRQ-Adressen der Platine oder an einem Konflikt mit anderen entfernten Einheiten des Systems liegen. Es kann auch die Folge einer Beschädigung der Achsenplatine sein.

Lösung:
Prüfen Sie die Konfiguration der Platine, schließen Sie Konflikte mit anderen entfernten Einheiten aus. Verwendet man ein Fernmodul, ist die Firmware neu ans Modul zu übertragen. Fachleute können einen Hardware-Test des Führen Sie die Dual-Port-Memory der Mikrosteuerung i296 aus. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

2061 Fehler beim Übertragen der Firmware an den Sender

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten Insbesondere ist die Übertragung der Konfiguration der I/O-Fernmodule an den Sender GreenBUS (Mikrosteuerung i296) gescheitert.

Lösung:
Prüfen Sie die Hardwarekonfiguration; verwendet man ein Fernmodul, ist die Firmware neu ans Modul zu übertragen. Fachleute können einen Hardware-Test des RAM der Mikrosteuerung i296 ausführen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

2062 Fehler beim Übertragen der Konfiguration an den Sender

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten. Insbesondere ist die Initialisierung der E/A-Fernmodule gescheitert.

Lösung:
Prüfen Sie die Hardwarekonfiguration; verwendet man ein Fernmodul, ist die Firmware neu ans Modul zu übertragen. Fachleute können einen Hardware-Test des RAM der Mikrosteuerung i296 ausführen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

2063 Fehler beim Übertragen der Konfiguration an den Empfänger

Ursache:
Bei der Initialisierung eines Fernmoduls ist ein Fehler aufgetreten.

Lösung:
Die Hardware-Konfiguration prüfen. Fachleute können einen Hardware-Test des Fernmoduls ausführen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

2064 EmpfängerNummer: Firmware-Version nicht kompatibel

Ursache:
Der Fern-Empfänger verfügt über eine Firmware-Version, die nicht mit der Firmware-Version der Steuerung kompatibel ist.

Lösung:
Prüfen Sie die Installation der Steuerung. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

2065 EmpfängerNummer: Fehler bei einer asynchronen Verbindung

Ursache:
Ein Fehler oder eine fehlende Antwort bei der Verbindung eines Befehls mit einem Ferngerät ist entstanden (GreenBUS v4.0).

Lösung:
Verbindungen und GreenBUS-Stromversorgung überprüfen. Bleibt das Problem bestehen, nehmen Sie mit dem technischen Kundendienst Kontakt auf.

2066 EmpfängerNummer: Allgemeiner Fehler

Ursache:
Ein allgemeiner Fehler bei der Verbindung eines Ereignisses oder eines Alarms von einem Ferngerät (GreenBUS v4.0) ist aufgetreten.

Lösung:
Überprüfen Sie Verbindungen und GreenBUS-Stromversorgung. Bleibt das Problem bestehen, nehmen Sie mit dem technischen Kundendienst Kontakt auf.

2067 EmpfängerNummer: Fehler beim Übertragen der Konfiguration

Ursache:
Beim Übertragen der Konfigurationsdaten an eine Fernsteuerung (GreenBUS v4.0) ist ein Kommunikationsfehler aufgetreten.

Lösung:
Überprüfen Sie Verbindungen und GreenBUS-Stromversorgung. Schalten Sie aus und ein. Bleibt das Problem bestehen, nehmen Sie mit dem technischen Kundendienst Kontakt auf.

2068 EmpfängerNummer: Interner Fehler Nr. FehlerNummer

Ursache:
Ein interner Fehler ist auf der angegebenen Fernsteuerung aufgetreten.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

2069 EmpfängerNummer: Stormversorgung + 24 Vcc Fehler Werkbank Nummer

Ursache:
Die Stromversorgung (+24 Vcc) einer Output-Gruppe mit demselben Stromklemme ist nicht aktiv oder funktioniert nicht einwandfrei.

Lösung:
Prüfen Sie den Stromversorgungsbetrieb +24 Vcc.

6.2.3 Von der Verwaltung MECHATROLINK-II ausgelöste Fehler

2308 Platine PlatinenNummer: Wegen einer inkorrekten Einstellung eines Konfigurationsparameter fiel die Initialisierung aus

Ursache:
In Konfiguration Virtuell-Physisch wurde keine Achse (logische Vorrichtung) an eine Platine mit dem MECHATROLINK-II-Bus (physischer Vorrichtung) angeschlossen.

Lösung:
Prüfen Sie die Anschlüsse in Konfiguration Virtuell-Physisch.

2341 Platine PlatinenNummer: Höchstzahl des Servoantriebes über den gültigen Wert

Ursache:
Hinsichtlich der eingestellten Konfiguration wurde an eine Platine mit MECHATROLINK-II-Bus eine hohe Servoantriebszahl angeschlossen.

Lösung:
Prüfen Sie den Wert von Frequenz Achsenkontrolle in Systemkonfiguration.

In der folgenden Tabelle sind die richtigen Werte angezeigt, die nach der Anzahl der von der Platine gesteuerten Servoantriebe einzustellen sind.

Platine	Frequenz Achsenkontrolle (Hz)	Höchstzahl der Servoantriebe
AlbMech	1000	8
AlbMech	<=500	16
DualMech Mono	1000	8
DualMech Mono	500	20
DualMech Mono	250	30
DualMech	1000	16
DualMech	500	40
DualMech	250	60

2342 Platine PlatinenNummer: Die Hardware-Adresse des Servoantriebes Servo überschreitet den Höchstzahl

Ursache:
In einer Platine mit MECHATROLINK-II-Bus wurde eine Achse (logische Vorrichtung) an eine Hardware Adresse verbunden (physische Vorrichtung) die höher als die Zahl der von der Platine steuerbaren Servoantriebe ist.

Lösung:
Den Wert von Frequenz Achsenkontrolle in Systemkonfiguration überprüfen.
In der folgenden Tabelle sind die richtigen Werte angezeigt, die nach den Zahl der von der Platine gesteuerten Servoantriebe einzustellen sind.

Platine	Frequenz Achsenkontrolle (Hz)	Höchstzahl der Servoantriebe
AlbMech	1000	8
AlbMech	<=500	16
DualMech Mono	1000	8
DualMech Mono	500	20

DualMech Mono	250	30
DualMech	1000	16
DualMech	500	40
DualMech	250	60

In der Konfiguration Virtuell-Physisch prüfen Sie den Anschluss zwischen logischer Vorrichtung und physischer Vorrichtung Zum Beispiel, sollte die Höchstzahl der Servoantriebe bei 8 liegen, dann muss der Anschluss zwischen logischer Vorrichtung und physischer Vorrichtung innerhalb den ersten 8 Schritten (von Ax1 bis Ax8) liegen.

2349 Platine PlatinenNummer: Der Servoantrieb Servo ist nicht angeschlossen

Ursache:
Die physikalische Verbindung dem Servoantrieb ist unterbrochen.

Lösung:
Prüfen Sie die Verkabelungen von MECHATROLINK-II-Bus und Servoantrieb.

6.2.4 Von der Verwaltung CAN-Bus ausgelöste Fehler

2761 KnotenNummer: Getrennt

Ursache:
Der angegebene Knoten CAN scheint mit dem Feldbus, der sich auf die angezeigte Platine bezieht, nicht angeschlossen zu sein, trotz seiner Anwesenheit in der Konfiguration.

2762 KnotenNummer: Wieder angeschlossen

Ursache:
Der angegebene Knoten CAN scheint mit dem Feldbus soeben angeschlossen zu sein, der mit der angezeigten Platine bezogen ist.

2763 Fehlgeschlagene Übertragung Fehler

Ursache:
Die Übertragung von einem oder mehreren Paketen auf dem angegebenen CAN-Knoten ist fehlgeschlagen

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

2764 KnotenNummer: Fehlgeschlagene Empfang Fehler

Ursache:
Der Empfang von einem oder mehreren aus dem CAN-Knoten ist fehlgeschlagen.

Lösung:
Anschluss und Stromversorgung der angegebenen CAN-Vorrichtung. Prüfen Sie die Verkabelung der ganzen CAN-Linie. Prüfen Sie den Anschluss an die Numerische Steuerung. Prüfen Sie die Kohärenz unter den Protokolleingaben der angegebenen CAN-Vorrichtung hinsichtlich der Eingaben des in der numerischen Steuerung vorhandenen Senders (Baud Rate, Adresse, spezifischer Eingaben des übernommenen Protokolls).

2765 KnotenNummer: Initialisiert

Ursache:
Der angegebene CAN-Knoten wurde mit dem Feldbus angeschlossen. Dann wurde er einwandfrei initialisiert.

2766 Störungszustand auf der CAN-Schnittstelle

Ursache:
Störungsmeldung der internen Stromversorgung der Vorrichtung von CAN-Schnittstelle.

Lösung:
Nehmen Kontakt Sie mit dem Hersteller auf.

2767 Fehler: Zustandverlust in CANopen

Ursache:
Aufgrund eines ernsten Problems ist der CAN-Sender nicht mehr in Betrieb (Operational).

Lösung:
Nehmen Kontakt Sie mit dem Hersteller auf.

2768 KnotenNummer: PDO wurde nicht empfangen

Ursache:
Die numerische Steuerung empfing das PDO, das vom CAN-Knoten wartete, nicht.

Lösung:
Prüfen Sie:
- die Stromversorgung des Knotens;
- dass der Knoten nicht im präoperativen Modus geblieben ist.
- Prüfen Sie die PDO- und CAN-Bus_Daten, die in Albatros konfiguriert sind.

2769 KnotenNummer: Ein nicht konfigurierter Knoten wurde empfangen

Ursache:
Ein Knoten, der in der Hardware-Konfiguration Albatros nicht deklariert wurde, wurde in der CAN-Netz erkannt.

Lösung:
Überprüfen Sie die Hardware-Knotenadresse und die in der Hardware-Konfiguration eingestellte Knotenadresse. Überprüfen Sie dass der Knoten in der Hardware-Konfiguration deklariert wurde, ansonsten muss man ihn hinzufügen.

2770 KnotenNummer: Falsche Konfiguration

Ursache:
Die Beschreibung der Daten von RPDO und TPDO ist falsch.

Lösung:
In der Hardware-Konfiguration korrigieren Sie die Beschreibung der Daten der Sende- und Empfang-PDOs.

2771 KnotenNummer: Fehler auf SDO-Kommunikation

Ursache:
Der angegebene CAN-Knoten hat in einer asynchronen Kommunikation (SDO) nicht geantwortet.

Lösung:
Prüfen Sie den Zustand des Knotenanschlusses. Wenn das Problem weiterhin besteht, nehmen Kontakt Sie mit dem Hersteller auf.

2772 Timeout auf CAN-Zyklus beim Abfragen der Knoten

Ursache:
Während der Datenabfrage CAN der Knoten ist ein Timeout-Fehler aufgetreten

Lösung:

In der Hardware-Konfiguration ändern Sie den Wert der eingestellten Abtastzeit.

3073 KnotenNummer: Emergency Fehler Nr. FehlerNummer

Ursache:
Die CANopen-Vorrichtung hat beim Knoten einen Fehler gefunden, der vom angezeigten Code definiert ist. Es handelt sich von Fehlersituationen, die dem einzelnen Knoten betreffen und die mit dem standardmäßigen CiA DS301 - Protokoll EMERGENCY konform sind.

Lösung:
Verweisen Sie sich auf die Dokumentation über die Knoten.

3074 KnotenNummer: Allgemeiner CAN-Fehler Nr. FehlerNummer

Ursache:
Auf dem angegebenen Knoten ist ein interner Fehler aufgetreten.

Lösung:
Nehmen Sie mit Tpa S.r.l Kontakt auf.

3088 CAN-PlatinenNummer: Knoten KnotenNummer: Fehler auf SDO Kommunication SDO Nr. FehlerNummer - Beschreibung

Ursache:
In einer READDICTIONARY oder WRITEDICTIONARY - Anweisung konnten eine oder mehrere Anforderungen von SDO-Lesen und Schreiben nicht ausgeführt werden. Mögliche Gründen:

- Es wurde angefordert, ein in der zugrunde liegenden Vorrichtung nicht implementiertes CANOpen-Objekt zu lesen, oder
- in einem CANOpen-Register werden keine mit dem Objekt-Typ kompatiblen Daten (zum Beispiel, es wird versucht, in einem Integer-Objekt-Typ eine Zeichenfolge zu schreiben) geschrieben. Der angegebene Fehlercode stimmt mit den DS402-Spezifikationen überein und neben dem numerischen Code wird auch die Textbeschreibung angegeben. Der Fehlercode ist eine hexadezimale Zahl.

Lösung:
Überprüfen Sie dass die in der Hardware-Konfiguration eingestellten Parameter von BaudRate und Abtastzeit etc. und dass die Parameter von in der GPL-Code eventuell vorhandenen READDICTIONARY und/oder WRITEDICTIONARY-Anweisungen, korrekt sind.

6.2.5 Von der Verwaltung EtherCAT-Bus ausgelöste Fehler

3329 Fehler beim Initialisieren des Kommunikation-Sockets

Ursache:
Die Firmware konnte nicht mit der Netzwerkkarte kommunizieren.

Lösung:
Wenn die Karte in einem RTX-System konfiguriert wurde, überprüfen Sie, ob die .ini-Dateien im Albatros FW-Unterordner korrekt geschrieben sind. Zur Überprüfung der Dateisyntax lesen Sie das "Albatros Installationshandbuch" (InstallationRTXGuide.pdf).

3330 Fehler beim Scannen des EtherCAT-Netzwerkes

Ursache:
Bei dem einleitenden Abtasten des EtherCAT - Netzwerkes, hat der Master keine Antwort von einigen oder von allen konfigurierten Slaves erhalten oder entspricht die Konfiguration nicht dem effektiven EtherCAT - Netzwerk des Feldes.

Lösung:
Überprüfen Sie die Verkabelung zwischen EtherCAT - Master und Slave.

Überprüfen Sie die Beschreibungen der Geräte in der Hardware-Konfiguration. Das Fenster von Diagnostik-Hardware hilft den Fehler zu finden. Hier sind die anwesenden Knoten visualisiert und, falls sie falsch konfiguriert sind, wird neben dem Namen des gefundenen Gerätes der Name des erwarteten Gerätes visualisiert.

3331 Fehler beim Konfigurieren der Übertragung-Mailbox

Ursache:
Der EtherCAT-Knoten hat auf den durch den Master erteilten Befehl aus verschiedenen Ursachen, wie abwesender Kommunikation, fehlerhaften Knoten, usw, nicht reagiert.

Lösung:
Prüfen Sie die Verkabelung und den Fernbetrieb.

3332 Fehler beim Konfigurieren der empfangenden Mailbox

Ursache:
Der EtherCAT-Knoten hat auf den durch den Master erteilten Befehl aus verschiedenen Ursachen, wie abwesender Kommunikation, fehlerhaften Knoten, usw., nicht reagiert.

Lösung:
Prüfen Sie die Verkabelung und den Fernbetrieb.

3333 Platine EtherCAT-Nummer: Fehler beim Erweiterungstyp des Knotens KnotenNummer

Ursache:
Der auf einem EtherCAT konfigurierte Erweiterungstyp in der Hardware-Konfiguration entspricht nicht dem wirklich anwesenden Erweiterungstyp. (Zum Beispiel, es wurde in der Hardware-Konfiguration ein TRS-CAT mit einer Erweiterung TRS-IO-E definiert, während im System ein TRS-CAT mit einer TRS-AN-E-Erweiterung anwesend ist).

Lösung:
Überprüfen Sie, dass die in der Hardware-Konfiguration definierten Geräte mit den anwesenden entsprechen.

3334 Fehler beim Konfigurieren der PDOs

Ursache:
Der EtherCAT-Knoten, für den hat man versucht die PDOs zu konfigurieren, ist auf dem Netz nicht vorhanden oder ist defekt.

Lösung:
Überprüfen Sie, dass die Konfiguration des Netzes EtherCAT, wie in der Konfiguration Albatros beschrieben ist, der physikalischer Konfiguration des Netzes entspricht.

3335 Knoten KnotenNummer ist alarmiert (FehlerNummer)

Ursache:
Der angezeigte Knoten befindet sich in einem Alarmzustand.

Lösung:
Überprüfen Sie den Alarm-Code nach der folgenden Tabelle:

Alarm-Code	Beschreibung
0x0001	Unspecified error
0x0002	No memory
0x0011	Invalid requested state change
0x0012	Unknown requested state

0x0013	Bootstrap not supported
0x0014	No valid firmware
0x0015	Invalid mailbox configuration
0x0016	Invalid mailbox configuration
0x0017	Invalid sync manager configuration
0x0018	No valids inputs available
0x0019	No valid outputs
0x001A	Synchronization error
0x001B	Sync manager watchdog
0x001C	Invalid Sync Manager Types
0x001D	Invalid Output Configuration
0x001E	Invalid Input Configuration
0x001F	Invalid Watchdog Configuration
0x0020	Slave needs cold start
0x0021	Slave needs INIT
0x0022	Slave needs PREOP
0x0023	Slave needs SAFEOP
0x0024	Invalid input mapping
0x0025	Invalid output mapping
0x0026	Inconsistent settings
0x0027	Free-Run not supported
0x0028	Synchronization not supported
0x0029	Free-Run needs 3 buffer mode
0x002A	Background watchdog
0x002B	No valid inputs and outputs
0x002C	Fatal Sync error
0x002D	No Sync error
0x0030	Invalid DC SYNCH Configuration
0x0031	Invalid DC Latch Configuration
0x0032	PLL Error
0x0033	Invalid DC IO Error
0x0034	Invalid DC Timeout Error
0x0035	DC Invalid Sync Cycle Time
0x0036	DC Sync0 Cycle Time
0x0037	DC Sync1 Cycle Time
0x0041	MBX_AOE
0x0042	MBX_EOE
0x0043	MBX_COE
0x0044	MBX_FOE
0x0045	MBX_SOE
0x004F	MBX_VOE

0x0050	EEPROM no access
0x0051	EEPROM error
0x0060	Slave restarted locally

3336 EtherCAT Platine Nummer: Die Zahl der Erweiterungen des Knotens KnotenNummer ist falsch

Ursache:
Die auf einem EtherCAT-Knoten in Albatros konfigurierte Erweiterungszahl entspricht nicht der anwesenden Erweiterungszahl. (Zum Beispiel, wurde in der Hardware-Konfiguration ein TRS-CAT mit zwei TRS-IO-E-Erweiterungen definiert, während im System eine einzige Erweiterung existiert).

Lösung:
Überprüfen Sie, dass die in der Hardware-Konfiguration definierten Geräte mit den anwesenden entsprechen.

3337 EtherCAT Platine: Knoten KnotenNummer getrennt

Ursache:
Der Knoten EtherCAT hat der Steuerung noch keine Antwort zurückgegeben. Der Knoten könnte vom Netz getrennt oder abgeschaltet sein.

Lösung:
Überprüfen Sie die Verkabelung und dass der Knoten mit Strom versorgt wird.

3338 EtherCAT Platine: Knoten KnotenNummer wieder angeschlossen

Ursache:
Der angegebene Knoten EtherCAT wurde wieder dem Netzwerk verbunden und hat wieder angefangen, auf die Steueranforderungen zu antworten.

3340 EtherCAT Platine: Der Knoten KnotenNummer antwortete nicht der Anforderung (Code)

Ursache:
Das Knoten EtherCAT, der über den SDO-Dienst abgefragt wurde, hat auf die Anfrage nicht geantwortet. Außerdem, wenn die Vorrichtung einen Abbruchcode gegeben hat, wird dieser in der angezeigten Fehlermeldung (Code) gezeigt. Dieser Fehler kann während der Ausführung der Anweisungen SETPZERO, SETPFLY und FASTREAD auf EtherCAT-Achsen auftreten, da sie SDO-Kommunikation zum Konfigurieren des Antriebs verwenden. In diesem speziellen Fall kann die Fehlermeldung zusätzlich zu den Standardabbruchcodes die folgenden Codes enthalten:

- 1: Interner abgelaufener Timeout: Der Knoten hat innerhalb von 250 nicht geantwortet
- 00000005: falscher interner Mailboxindex.
- 00000006: empfangene Daten stimmen nicht mit der Anfrage.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

3341 EtherCAT Platine: Der Knoten KnotenNummer besteht nicht

Ursache:
Ein nicht existierender Knoten hat versucht, einen Befehl auszuführen.

Lösung:
Überprüfen Sie die Nummer der GPL-Anweisung, die den Fehler generiert hat. Verbinden Sie den Knoten.

3342 Kabel nicht verbunden

Ursache:
Das EtherCAT-Kabel ist nicht mit der Steuerung verbunden.

Lösung:
Überprüfen Sie, dass das Kabel korrekt an die Steuerung angeschlossen ist und nicht beschädigt ist.

3343 EtherCAT Platine Nummer: Knoten KnotenNummer: Der ändert seinen Zustand in SAFE-OPERATIONAL nicht (Code)

Ursache:
Der angegebene EtherCAT-Knoten wird nicht in den Zustand SAFE-OPERATIONAL versetzt.

Lösung:
Kontaktieren Sie den Hersteller und melden Sie die angezeigte Fehlernummer (Code), die den ALStatuscode-Code darstellt.

3344 EtherCAT Platine Nummer: Knoten KnotenNummer: Der ändert seinen Zustand in OPERATIONAL nicht (Code)

Ursache:
Der angegebene EtherCAT-Knoten wird nicht in den Zustand OPERATIONAL versetzt.

Lösung:
Wenden Sie sich an den Hersteller an und melden Sie die angegebene Fehlernummer (Code), die den ALStatuscode-Code darstellt.

3345 EtherCAT Platine: Instabile Kommunikation

Ursache:
Die Kommunikation auf EtherCAT-Netz ist instabil aufgrund von Störungen oder fehlerhaften Kabeln oder Knoten.

4400 Zu viele aktive Achsen in FASTREAD (Funktion: FunktionName Zeile: ZeileNummer)

Ursache:
Man versuchte, eine FASTREAD-Anweisung mit mehr Achsen als die zugelassene Anzahl zu benutzen. Es ist nötig, nicht zu vergessen, dass die mit zusätzlichen Encoder Achsen gleichbedeutend mit Doppelachsen sind (siehe Anweisung SWITCHENC).

Lösung:
Reduzieren Sie die benutzte Anzahl von Achsen mit zusätzlichem Encoder.

6.2.6 Von der Initialisierung ausgelöste Fehler

769 Fehlerhafte Software-Konfiguration

Ursache:
Die Hardware-Konfiguration des Fernmoduls entspricht nicht der in der Systemkonfiguration spezifizierten Software-Konfiguration.

Lösung:
Prüfen Sie die Kongruenz der Hardware-Parameter des Fernmoduls mit den Software-Parametern.

770 Falsch konfigurierte IRQ-Nummer

Ursache:
Der IRQ der Achsenplatine ist in der Modulkonfiguration falsch eingestellt worden. Normalerweise handelt es sich um einen Hardware-Konflikt mit anderen entfernten Einheiten des Systems.

Lösung:

In den BIOS-Einstellungen der Mutterkarte prüfen Sie, ob der von der Achsenplatine verwendete IRQ "Legacy ISA" reserviert ist. Prüfen Sie, dass keine sonstigen entfernten Einheiten denselben IRQ wie die Achsenplatine verwenden. Wenn möglich, den IRQ der entfernten Einheit ändern, die mit der Achsenplatine im Konflikt ist, ansonsten den IRQ der Achsenplatine ändern.

772 Fehler beim Lesen des Pufferspeicherbereichs während der Initialisierung

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten. Insbesondere ist der Test des Puffer-RAMs (Dallas) gescheitert. Es kann an einer falschen Konfiguration der E/A und IRQ-Adressen der Platine oder an einem Konflikt mit anderen entfernten Einheiten des Systems liegen. Es kann auch die Folge einer Beschädigung der Achsenplatine sein.

Lösung:
Prüfen Sie die Hardwarekonfiguration. Fachleute können einen Hardware-Test des RAM der Mikrosteuerung i296 ausführen. Dabei ist zu beachten, dass der Hardware-Test des RAM die Löschung aller darin gespeicherten Daten zur Folge hat. Im Puffer-RAM sind die Werte einiger Vorrichtungen gespeichert wie z.B. Zähler, Timer und Offsets des DAC der Achsen. Diese Werte sind vor Ausführung des Tests anderweitig zu speichern. Bleibt das Problem bestehen, nehmen Sie mit dem Hersteller Kontakt auf.

773 Höchstzahl konfigurierbarer Achsen erreicht

Ursache:
Man hat versucht, mehr Achsen zu konfigurieren als die dafür zulässige Höchstzahl.

Lösung:
Reduzieren Sie die Anzahl konfigurierter Achsen. Weitere Informationen sind über den Hersteller erhältlich.

774 Achsen-Echtzeit ist nicht in Betrieb gegangen

Ursache:
Die Firmware zur Verwaltung der Achsen ist initialisiert worden, funktioniert aber nicht einwandfrei. Normalerweise handelt es sich um einen Hardware-Konflikt mit anderen entfernten Einheiten des Systems.

Lösung:
Prüfen Sie, dass keine sonstigen entfernten Einheiten Konflikte verursachen, deren Konfiguration ändern oder sie aus dem System entfernen.

775 Nicht genug Zeit zur Ausführung des GPL

Ursache:
Die Ausführung eines Realtime-Tasks nimmt zu viel Zykluszeit ein. Dieser Fehler tritt auf, wenn ein Realtime-Task nicht vor Beginn der nächsten Achsen-Realtime endet (wenn z.B. ein unendlicher Zyklus geschaffen worden ist).

Lösung:
Ändern Sie den GPL-Code, so dass der Realtime-Task weniger lange dauert.

776 Zu hohe Zeit der Echtzeit-Ausführung

Ursache:
Die Ausführung eines Real Time-Tasks nimmt zu viel Zykluszeit ein. Die Ausführungszeit ist etwas länger als die dafür zulässige Höchstdauer.

Lösung:
Ändern Sie den GPL-Code um die um die Dauer des Realtime-Tasks zu verkürzen.

777 Watchdog abgelaufen

Ursache:
Die Firmware ist blockiert.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

778 Main Kode der Firmware ist blockiert

Ursache:
Die Firmware ist für mehr als 5 Realtime-Sperren gesperrt.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1025 Platine PlatinenNummer: Reagiert nicht auf den Befehl

Ursache:
Während der Initialisierung ist eine Achsen-Platine ermittelt worden, das aber nicht korrekt funktioniert.

Lösung:
Fachleute können einen Hardware-Test der Achsenplatine ausführen. Bleibt das Problem bestehen, mit dem Hersteller Kontakt aufnehmen.

1026 Platine PlatinenNummer: Fehler bei der Firmware-Übertragung an die Achsenplatine

Ursache:
Die Firmware konnte der Platine nicht übertragen werden.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1028 Platine PlatinenNummer: Firmware nicht vorhanden

Ursache:
Die auf der Platine vorhandenen Firmware stimmen nicht mit der abgelesenen Platine überein.

Lösung:
Die korrekte Firmware übertragen.

1029 Platine PlatinenNummer: Main blockiert

Ursache:
Die Firmware der Platine wurde nicht in Betrieb genommen.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1031 Platine PlatinenNummer: Fehler beim Initialisieren

Ursache:
Während der Initialisierungsprozedur der Achsenplatine ist ein Fehler aufgetreten.

Lösung:
Prüfen und beseitigen Sie die Ursachen der Systemfehler, die kurz vor dem jetzt entstandenen Fehler unterlaufen sind. Dann initialisieren Sie das System.

1032 Platine PlatinenNummer: Test des Dual-Port-Memory fehlgeschlagen

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten Insbesondere ist die Initialisierung des Dual-Port-Memory der Mikrosteuerung i960 gescheitert. Meistens ist die Ursache ein Hardware-Konflikt mit anderen entfernten Einheiten des Systems, aber es kann auch an einer Beschädigung der Platine liegen.

Lösung:
Prüfen sie die Konfiguration der Platine, und schließen Sie die Konflikte mit anderen entfernten Einheiten aus. Verwendet man ein Fernmodul, ist die Firmware neu ans Modul zu übertragen. Fachleute können einen Hardware-Test des Dual-Port-Memory ausführen. Bleibt das Problem bestehen, nehmen Sie mit dem Hersteller Kontakt auf.

1033 Platine PlatinenNummer: Der Boot-Code der Firmware ist nicht in Betrieb

Ursache:
Die Start-Firmware der Mikrosteuerung i960 ist initialisiert worden, funktioniert aber nicht richtig. Normalerweise handelt es sich dabei um einen Hardware-Konflikt mit anderen entfernten Einheiten des Systems.

Lösung:
Prüfen Sie die Konfiguration der Platine, und schließen Sie die Konflikte mit anderen entfernten Einheiten aus. Verwendet man ein Fernmodul, ist die Firmware neu ans Modul zu übertragen. Fachleute können einen Hardware-Test des Dual-Port-Memory der Mikrosteuerung i960 ausführen. Bleibt das Problem bestehen, mit dem Hersteller Kontakt aufnehmen.

1035 Platine PlatinenNummer: Nicht vorhanden

Ursache:
Während der Tests in der Initialisierungsphase der Achsenplatine ist ein Fehler aufgetreten. Insbesondere ist die Platine nicht ermittelt worden.

Lösung:
Prüfen, ob die Platine im System vorhanden und nicht beschädigt ist. Fachleute können einen Hardware-Test der Platine ausführen. Bleibt das Problem bestehen, mit dem Hersteller Kontakt aufnehmen.

1037 Platine PlatinenNummer: Dual-Port-Memory konnte nicht geöffnet werden

Ursache:
Das Dual-Port-Memory der Platine konnte nicht geöffnet werden.

Lösung:
Fachleute können einen Hardware-Test der Achsenplatine ausführen. Kontakt mit Tpa S.r.l aufnehmen.

1039 Platine PlatinenNummer: Watchdog abgelaufen

Ursache:
Die Firmware der Achsenkarte *PlatinenNummer* ist blockiert.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1040 Platine PlatinenNummer: Stromversorgungsfehler +24 Vcc

Ursache:
Die Niederspannung-Stromzufuhr (+24 Vcc) der Ausgänge ist nicht vorhanden oder funktioniert nicht einwandfrei.

Lösung:
Das Funktionieren der Stromzufuhr +24 Vcc prüfen.

1047 Platine PlatinenNummer: Software-Konfiguration nicht zulässig

Ursache:
Das Gerät hat eine Konfiguration erhalten, die nicht mit der verwendeten Hardware übereinstimmt. Es ist z.B. eine Achse konfiguriert worden, die in der Hardware gesperrt ist.

Lösung:
Die Übereinstimmung zwischen Hardware- und Software-Parametern der Platine prüfen.

1052 Platine PlatinenNummer: Der Bootcode wird ausgeführt

Ursache:
Das Gerät befindet sich aufgrund des Forcierens der Hardware im provisorischen Modus (nur der Bootcode ist aktiv).

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1053 Platine PlatinenNummer: Watchdog der Achsen abgelaufen

Ursache:
Bei dem Firmware-Ausführen der Platine der Achsenkontrolle ein schwerwiegender Fehler ist aufgetreten. Die Achsen werden deaktiviert und das eventuelle SYSOK- Signal wird gesenkt. Das System nicht neu starten.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1055 Watchdog auf Platine PlatinenNummer abgelaufen

Ursache:
Die Firmware der Platine *PlatinenNummer* ist blockiert

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1056 Platine PlatinenNummer: Stromversorgungsfehler bei der CAN-Schnittstelle

Ursache:
Versorgungsfehler der auf der Platine verwendeten Übertragungseinrichtung der CAN-Bus-Linie.
Ursachen: Kurzschluss, Verkabelungsfehler innerhalb des Bus-Systems oder Platine-Beschädigung.

Lösung:
Die Verkabelung der ganzen CAN-Linie überprüfen.
Anschluss an die Numerische Steuerung überprüfen.
Eventuellen Kurzschluss entfernen. Sollte die Verbindung nicht wieder hergestellt werden, nehmen Sie Kontakt mit dem Hersteller auf.

1057 Platine PlatinenNummer: Interner Fehler Nummer FehlerNummer

Ursache:
Fehler in der Hardware der Knoten.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

6.2.7 Von der Speicherverwaltung ausgelöste Fehler

1281 Fehler bei der Speicherallokation im Heap-Bereich

Ursache:
Der verfügbare RAM-Speicher ist kleiner als jene, die z.B. für eine globale Matrix notwendig ist.

Lösung:
Vermindern Sie die Größe der dem RAM zugeordneten globalen Variablen.

1286 Fehler bei der Heap-Verwaltung

Ursache:
Fehler bei der Verwaltung des Speichers seitens der Firmware.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1287 Zu viele Speicherdeallokationen aus Heap

Ursache:
Fehler bei der Verwaltung des Speichers seitens der Firmware.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1289 Fehler beim Schaffen der globalen Variablen

Ursache:
Es wurden zu viele globale Variablen oder es sind zu große globale Matrizen definiert worden.

Lösung:
Vermindern Sie die Anzahl der globalen Variablen oder die Größe der Matrizen.

1290 Fehler in der Dimension der nicht-flüchtigen Variablen

Ursache:
Es ist eine zu hohe Anzahl nichtflüchtiger Variablen oder es sind zu große nichtflüchtige Matrizen definiert worden.

Lösung:
Vermindern Sie die Anzahl der nichtflüchtigen Variablen oder die Größe der nichtflüchtigen Matrizen.

1291 Fehler in der Dimension der schreibgeschützten Variablen

Ursache:
Es ist eine zu hohe Anzahl schreibgeschützter Variablen oder es sind zu große schreibgeschützte Matrizen definiert worden.

Lösung:
Vermindern Sie die Anzahl der schreibgeschützten Variablen oder die Größe der schreibgeschützten Matrizen.

6.2.8 Von Faults ausgelöste Fehler

1559 Haltepunkte-Trace

Ursache:
Schwerwiegender Firmware-Fehler.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1569 Betriebscode des Mikroprozessors nicht gültig

Ursache:
Der Mikroprozessor ist auf eine unbekannte Anweisung gestoßen. Der Fehler kann an Hardware-Problemen des PCs liegen oder an der Beschädigung von Dateien mit der Firmware von Albatros.

Lösung:
Im Fall eines lokalen Moduls prüfen, dass die Dateien nicht beschädigt sind und evtl. Albatros neu installieren. Bei Clipper-Modulen ist die Firmware zu aktualisieren. Einen Hardware-Test des PCs, insbesondere des RAM ausführen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

1586 INTEGER-Wert durch Null geteilt

Ursache:
Man hat versucht, einen INTEGER-Wert durch Null zu teilen.

Lösung:
Prüfen Sie in den GPL-Funktionen, ob alle Teilungen korrekt sind.

1600 Überlauf beim Ergebnis einer Rechnung mit Gleitkomma

Ursache:
Das Ergebnis einer Rechnung zwischen FLOATs übertrifft die Kapazität des Empfängers:
±3,402823E+38 für Floats
±1,79769313486231E+308 für Doubles.

Lösung:
Prüfen Sie in den GPL-Funktionen, ob die Float-Rechnungen korrekt sind.

1601 Unterlauf beim Ergebnis einer Rechnung mit Gleitkomma

Ursache:
Das Ergebnis eines Gleitkommavorgangs zwischen FLOATs ist kleiner als die Kapazität des Empfängers:
1*2^126 für Floats
1*2^1022 für Doubles.

Lösung:
Prüfen Sie in den GPL-Funktionen prüfen, ob die Float-Rechnungen korrekt sind.

1602 Ungültiges Argument für eine Rechnung mit Gleitkomma

Ursache:
In einer Float-Vorgang ist ein nicht-floatender Operand verwendet worden.

Lösung:
In den GPL-Funktionen überprüfen, ob die Float-Rechnungen korrekt sind.

1603 Wert mit Gleitkomma durch Null geteilt

Ursache:
Man hat versucht, einen Float oder einen double durch Null zu teilen. Er wird gebildet, auch falls ein Logarithmus von Null ausgeführt wird.

Lösung:
In den GPL-Funktionen überprüfen, ob alle Teilungen korrekt sind.

1604 Falsches Ergebnis bei einer Rechnung mit Gleitkomma

Ursache:
Das Ergebnis eines Vorgangs zwischen Floats ist nicht richtig.

Lösung:
In den GPL-Funktionen überprüfen, ob die Float-Vorgänge korrekt sind.

1605 Ein falscher Gleitkommawert wurde verwendet

Ursache:
Ein Gleitkommawert kleiner als der minimal darstellbare Wert wurde verwendet:
± 1,401298E-45 für Floats
± 4,94065645841247E-324 für Doubles.

Lösung:
In den GPL-Funktionen überprüfen, ob die Float-Vorgänge korrekt sind.

1728 Man hat versucht, eine ungültige Adresse abzurufen

Ursache:
Das Programm hat einen ungültigen Speicherbereich abgerufen.

Lösung:
Die Übereinstimmung der globalen/lokalen Variablen prüfen; besteht das Problem weiter, auf die Störung hinweisen.

1735 Allgemeine Ausnahme

Ursache:
Es ist eine unbekannte Ausnahme eingetreten.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1736 Daten nicht ausgerichtet

Ursache:
Schwerwiegender Firmware-Fehler.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1801 Temperatur-Alarm

Ursache:
Die Temperatur der CPU der Steuerung hat die höchst zulässigen Werte überschritten.

Lösung:
Prüfen, dass keine Lüftungsprobleme herrschen oder Ursachen für Überhitzung bestehen. Bleibt das Problem bestehen, mit dem Hersteller Kontakt aufnehmen.

1802 Gebläse-Alarm

Ursache:
Das Gebläse der CPU der Steuerung funktioniert nicht einwandfrei. Das Problem führt in Kürze zur Überhitzung der CPU.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

1803 Die CPU-Frequenz ist instabil

Ursache:
Die Arbeitsfrequenz von CPU ist nicht stabil.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

6.2.9 Von GPL-Funktionen ausgelöste Fehler

4097 Die Vorrichtung VorrichtungTyp VorrichtungName ist nicht konfiguriert

Ursache:
Eine GPL-Anweisung hat eine nicht konfigurierte Vorrichtung benutzt, d.h. eine Vorrichtung ohne VirFisic-Anschluß. Der Fehler kann durch alle Anweisungen ausgelöst werden, die eine Vorrichtung als Parameter erhalten.

Lösung:
In den Konfigurationen der Steuerung überprüfen, dass alle von den Funktionen verwendeten Vorrichtungen über einen VirFisic-Anschluß verfügen.
Anschließend die Konfigurationen wieder an die Platine übertragen.

4098 Die globale Variable VariableName ist nicht vorhanden

Ursache:
Eine GPL-Anweisung hat eine nicht definierte globale Variable als Argument erhalten.
Geschieht meist, wenn die Steuerung nicht korrekt initialisiert worden ist.

Lösung:
Den gesamten GPL-Code neu kompilieren und die Steuerung erneut initialisieren.

4099 Funktion FunktionName nicht gefunden

Ursache:
Eine Funktion wurde abgerufen, die nicht auf der Platine vorhanden ist.
Das kann vorkommen, wenn nach einer Änderung des GPL-Codes keine Initialisierung der Steuerung ausgeführt worden ist.

Lösung:
Den gesamten GPL-Code neu kompilieren und die Steuerung erneut initialisieren.

4101 Unpassende Verwaltung der Achse AchseName

Ursache:
Es ist eine unpassende Bewegung auf einer Achse ausgeführt worden. Die Dokumentation zu den Bewegungen heranziehen.
Der Fehler kann von allen Anweisungen, die Achsen verwalten, ausgelöst werden. Normalerweise tritt er in folgenden Fällen auf:

- wenn man mit einer Achse, die bereits eine Punkt-zu-Punkt-Bewegung ausführt, eine Interpolation, eine Wicklung oder eine Koordination versucht (oder umgekehrt).
- Wenn man auf einer Achse im transparent mode eine Chain-, SetPFly- oder SetPTZero-Anweisung auszuführen versucht.
- Wenn man mit einer Achse, die Slave einer anderen Achse ist, eine Interpolation, eine Wicklung oder eine Koordination versucht.

Lösung:
Überprüfen, dass alle Achsenbewegungen mit einer Anweisung, auf einer bestimmten Höhe zu warten, enden; dies v.a., wenn sich verschiedene Bewegungen abwechseln (Punkt-zu-Punkt, Interpolation, usw.).

4105 Auf der Achse Achsename nicht ausführbare Anweisung

Ursache:
Man hat versucht, eine Anweisung auf einer Achse auszuführen, die sie nicht unterstützt. Z.B. eine Interpolationsanweisung auf einer Achse mit Schrittschaltung.

Lösung:
Den GPL-Code korrigieren.

4106 Das auf die Achse mit Schrittschaltung Achsenname bezogene Fernmodul ist nicht angeschlossen

Ursache:
Man hat versucht, auf eine Achse mit Schrittschaltung einzuwirken, die nicht an die Steuerung angeschlossen ist.

Lösung:
Den Anschluss des Fernmoduls zur Steuerung der Achse prüfen.

4107 SYSOK-Anweisung mit falschen Argumenten

Ursache:
Es ist eine SYSOK-Anweisung mit falschen Argumenten ausgeführt worden. Dieser Fehler tritt auf, wenn eine oder mehrere der als Argumente an die Anweisung übertragene digitale Ausgänge nicht korrekt konfiguriert sind.

Lösung:
Den GPL-Code und die Virtuell-physikalische Konfiguration prüfen.

4108 Die Achse Achsenname: Endhöhe über den Software-Grenzen

Ursache:
Man hat versucht, eine Achse außerhalb der im GPL-Code oder in der Konfiguration gesetzten Grenzen zu bewegen.

Lösung:
Das Bearbeitungsprogramm korrigieren, das den Fehler hervorgerufen hat. Eventuell den GPL-Code oder die Achsenkonfiguration korrigieren.

4110 Falsche Geschwindigkeit

Ursache:
Man hat versucht, einer Achse keine oder eine negative Geschwindigkeit zuzuweisen.

Lösung:
Den GPL-Code korrigieren.

4111 Negative Beschleunigung Achse Achsenname

Ursache:
Man hat versucht, einer Achse eine negative Beschleunigung zuzuweisen.

Lösung:
Den GPL-Code korrigieren.

4112 Negatives Abbremsen Achse Achsenname

Ursache:
Man hat versucht, einer Achse eine negative Verzögerung zuzuweisen.

Lösung:
Den GPL-Code korrigieren.

4114 Achse Achsenname: Zurücksetzung auf schnellem Eingang nicht ausgeführt

Ursache:
Das Nullstellen des Maßes am schnellen Eingang (Eil-Nullstellung) ist nicht korrekt vollendet worden. Dieses Verfahren erlaubt das Nullstellen des Maßes einer sich bewegenden Achse im selben Augenblick, in dem der zugehörige Fastread-Eingang seinen Zustand ändert. Beendet die Achse die angefangene Bewegung ohne Änderung des Eingangs, tritt dieser Systemfehler auf. Mögliche Ursachen

sind eine falsche Einstellung der Bewegungsparameter der Achse oder ein Verkabelungsproblem des Fastread-Eingangs.

Lösung:
Den GPL-Code prüfen, der den Eil-Nullstellung implementiert; die Verkabelung des Fastread-Eingangs prüfen.

4115 Achse Achsenname: Nullraste nicht gefunden

Ursache:
Das Nullstellen des Maßes an der Nullkerbe des Encoders ist nicht korrekt vollendet worden. Dieses Verfahren erlaubt das Nullstellen des Maßes einer sich bewegenden Achse im selben Augenblick, in dem die Nullkerbe des Encoders abgelesen wird. Erreicht die Achse das Maß der Suche nach der Kerbe, ohne diese zu finden, tritt dieser Systemfehler auf. Mögliche Ursachen sind eine falsche Einstellung der Bewegungsparameter der Achse oder ein Verkabelungsproblem des Kerbsignals (Phase C des Achsensteckers).

Lösung:
Den GPL-Code prüfen, der die Nullstellung an der Kerbe implementiert; die Verkabelung der Achse prüfen.

4353 Betriebscode unbekannte Anweisung (Funktion: FunktionName Zeile: ZeileNummer)

Ursache:
Bei Ausführung der Funktion ist eine unpassende Anweisung ausgeführt worden. Dies ist generell ein Zeichen dafür, daß die Dateien mit dem kompilierten GPL-Code beschädigt sind. Wenn die Software aktualisiert ist, prüfen Sie, den GPL-Code kompiliert zu haben. Die vorherige Version der Kontrollsoftware könnte denn für die neue Version alte Anweisungen enthalten.

Lösung:
Den gesamten GPL-Code erneut kompilieren und die Steuerung initialisieren. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

4354 Falscher mathematischer Vorgang (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine GPL-Anweisung hat versucht, einen falschen mathematischen Vorgang auszuführen, wie z.B. eine Teilung durch Null, oder einige der in die GPL-Anweisungen eingeführten Daten sind nicht kongruent. Dieser Fehler tritt normalerweise bei Interpolationen auf, weil es sich dabei um den Firmware-Teil handelt, der die meisten mathematischen Vorgänge ausführt, aber der Fehler kann auch durch folgende andere Anweisungen ausgelöst werden.

Lösung:
Prüfen, ob die an die Interpolationsanweisungen übertragenen Parameter korrekt sind. Wenn das Problem weiterbesteht, den Hersteller über die Störung in Kenntnis setzen.

4355 Falsche Matrix- oder Vektoradresse (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine GPL-Anweisung hat versucht, ein Array- oder Matrixelement außerhalb der maximalen Größe abzurufen, z.B. Abruf des Elements 10 eines Arrays mit 5 Elementen.
Kann von allen Anweisungen ausgelöst werden, die als Parameter einen Array oder eine Matrix akzeptieren.

Lösung:
Überprüfen Sie, dass sich alle an die Anweisungen übertragenen Matrix- und Array-Indexe innerhalb der Array- und Matrix-Größe befinden.

4356 Nicht von CALL aufgerufene RET-Anweisung (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Die Funktion hat eine RET-Anweisung ausgeführt, aber auf dem Stapelspeicher die entsprechende Rückkehradresse nicht gefunden. Die häufigste Ursache ist die Erklärung einer Unterprozedur vor der FRET-Anweisung zum Verlassen einer Funktion, ohne sie mit einem GOTO vor versehentlicher Ausführung geschützt zu haben. Oder es ist ein ungewünschter Sprung innerhalb einer Unterprozedur erfolgt.

Lösung:
Den Ablauf des GPL-Programms prüfen. Die Unterprozeduren vorzugsweise ans Ende der Funktionskörper (nach der FRET-Anweisung) setzen.

4357 Lokale Variable nicht vorhanden (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine GPL-Anweisung hat versucht, eine nicht zugeordnete lokale Variable abzurufen.

Lösung:
Alle Funktionen neu erstellen und wieder an die Platine übertragen. Wenn das Problem weiterbesteht, auf die Störung hinweisen.

4358 Sprungetikett nicht vorhanden (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine GPL-Anweisung ist zu einem nicht existierenden Sprungetikett gesprungen.
Kann durch GOTO, CALL, FCALL und allen IFs ausgelöst werden.

Lösung:
Alle Funktionen neu erstellen und wieder an die Platine übertragen. Wenn das Problem weiterbesteht, auf die Störung hinweisen.

4359 Makro-Argument ist fehlerhaft (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine GPL-Anweisung hat ungültige Argumente erhalten. Dieser Fehler kann von allen Anweisungen ausgelöst werden. Das GPL-System versucht jedoch in den meisten Fällen, diese Situation von selbst zu korrigieren, indem es automatische Konversionen des Typs (cast) vornimmt, was allerdings mit einem Zeitverlust verbunden ist. Der Fehler tritt auf, wenn diese Konversionen nicht möglich sind und insbesondere in folgenden Fällen:

- Anweisungen, die mit spezifischen Vorrichtungen arbeiten (SETTIMER, SETCOUNTER), die aber eine andere Vorrichtung erhalten.
- Anweisungen, die mit Bit arbeiten, aber eine Gleit-kommazahl erhalten (AND, OR, usw.)
- Anweisungen, die mit Matrix oder Array arbeiten, aber eine Variable (SORT, MOVEMAT, usw.) erhalten.
- Anweisungen, die mit Strings arbeiten, aber keine Strings erhalten.

Der Fehler tritt auf, wenn man auch versucht, auf einer Platine, die solche Anweisung nicht verwaltet, eine Anweisung auszuführen (Zum Beispiel, eine SENDPDO-Anweisung oder eine RECEIVEPDO-Anweisung auf einer Platine, die sich von TMSCan oder von TMSCan+ unterscheidet).

Lösung:
Den GPL-Code korrigieren.

4360 Fehler bei der Speicherzuordnung während der Ausführung (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Die GPL-Funktion hat versucht, einen Speicherplatz zum internen Gebrauch zuzuordnen, hat aber keinen verfügbaren Speicherplatz gefunden.
Diese Mitteilung könnte auf eine vorübergehende Situation hinweisen, z.B. aufgrund zu vieler gleichzeitig in Ausführung begriffener Tasks oder aufgrund von zu großen globalen Variablen.

Lösung:
Die Größe der globalen und lokalen Variablen überprüfen und versuchen, deren Größe zu vermindern. Prüfen, dass nicht zu viele Tasks gleichzeitig in Ausführung begriffen sind und sie evtl. vermindern.

4361 Zu viele aktive Aufgaben (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, mehr als 256 Tasks gleichzeitig auszuführen.

Lösung:
Die Anzahl gleichzeitig aktiver Tasks vermindern.

4362 Falsches Matrixformat (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine Anweisung, die mit Matrix arbeitet, hat ein ungültiges Format gefunden. Die Anweisungen, die einen solchen Systemfehler auslösen können, sind:
- MOVEMAT, wenn das Format der Ursprungsmatrix nicht mit dem Format der Zielmatrix übereinstimmt.
- CLEAR, wenn eine nicht vorhandene Matrixlinie gelöscht werden soll.
- GETAXIS, wenn das Format der Matrix, das als Parameter übertragen worden ist, nicht dem Format entspricht, das die Anweisung erwartet (die Dokumentation zur GPL-Sprache heranziehen).

Lösung:
Innerhalb des Tasks, die den Fehler ausgelöst hat, die links angegebenen Anweisungen über-prüfen. Insbesondere überprüfen, dass die an die Anweisung MOVEMAT übertragenen Matrizen über die gleiche Anzahl Spalten desselben Typs verfügen und dass die an GETAXIS übertragene Matrix das richtige Format besitzt.

4363 Zu viele aktive ONINPUT-Anweisungen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Mehr als 128 OnInput-Anweisungen wurden aktiviert.

Lösung:
Die Anzahl ONINPUTs vermindern.

4364 Achse dient bereits einem lokalem Bezug (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Der Fehler betrifft die Aktivierung von gedrehten und umgesetzten Dreiergruppen von Bezugsachsen, um auf mehreren kartesischen Achsen Interpolationen auszuführen.
Man hat versucht, eine SETRIFLOC auszuführen, wobei die der Anweisung angegebene Achse bereits als lokaler Bezug in einer Dreiergruppe dient. Der Fehler wird auch ausgelöst, wenn eine RESRIFLOC an einer Achse ausgeführt wird, die in keiner Dreiergruppe eingesetzt war. Der letzte Fall ist, daß keine Bezugsdreiergruppen mehr verfügbar sind. (maximal 32).

Lösung:
Überprüfen, dass die durch SETRIFLOC übertragenen Dreiergruppen keine gemeinsamen Achsen mehr haben.
Die RESRIFLOC überprüfen.
Ebenfalls überprüfen, ob vor der RESRIFLOC Anweisungen sind, auf einer bestimmten Höhe zu warten.
Beachten, dass solange die Interpolation nicht beendet ist, die RESRIFLOC nicht tat-sächlich ausgeführt wird.

4365 ONINPUT-Anweisung auf demselben Input aktiviert (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine ONINPUT-Anweisung hat mehrmals denselben Input erhalten.

Lösung:
Überprüfen, dass nicht zweierlei ONINPUT derselbe Parameter als Input übertragen wird.

4366 Zu viele aktive ONFLAG-Anweisungen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Es sind mehr als 128 OnFlag-Anweisungen aktiviert worden.

Lösung:
Die Anzahl der ONFLAGs vermindern.

4367 ONFLAG-Anweisung auf demselben FLAG aktiviert (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Eine ONFLAG-Anweisung hat mehrmals dasselbe Flag erhalten.

Lösung:
Überprüfen, dass nicht zweierlei ONFLAG dasselbe Flag als Parameter übertragen wird.

4368 Versuch, eine schreibgeschützte Variable zu schreiben (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Es wurde versucht, schriftlich auf eine schreibgeschützte Variable zuzugreifen,
Die schreibgeschüzten Variablen sind immer global und befinden sich im Steuerungsflash. Im Editor der globalen Variablen werden sie als "static" angegeben. Versucht man, auf eine dieser globalen Variablen zu schreiben, wird dieser System-Fehler ausgelöst.
Der Fehler wird außerdem ausgelöst, wenn Variablen aus dem PufferRAM ("nichtflüchtig") als Argument einiger Schreibanweisungen verwendet werden.
Zum Beispiel, muss in der Anweisung COORDIN die übergebene Variable, die die Zeile in Vorbereitung anzeigt, in RAM sein.

Lösung:
Überprüfen Sie alle static- und nichtflüchtigen Variablen.

4369 Zu viele aktive Master-Achsen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, mehr als vier Achsen gleichzeitig als Master zu aktivieren.
Dieser Fehler tritt nur bei der Ausführung der CHAIN-Anweisung auf.

Lösung:
Vermindern Sie die Anzahl der Master-Achsen.

4370 Zu viele aktive Slave-Achsen (Funktion:FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, mehr als acht Achsen als Slaves einer einzigen Master-Achse zu aktivieren.
Dieser Fehler tritt nur bei der Ausführung der CHAIN-Anweisung auf.

Lösung:
Vermindern sie die Anzahl der Slave-Achsen.

4372 Falscher Gebrauch einer Anweisung (Funktion:FunktionName Zeile: ZeilenNummer)

Ursache:
Dieser Fehler wird in einer der folgenden Situationen erzeugt:

1. Eine Anweisung zum Verwalten der Mailbox (sendmail, waitmail, endmail, ifmail) oder eine Anweisung zum IPC-Verwalten (sendipc, testipc, waitipc) innerhalb einer Funktion, die von einer Errsys, OnInput oder OnFlag-Anweisung aufgerufen ist.
2. Eine IfError- oder IfMessage-Anweisung wird verwendet, ohne die Verwaltung der Warnungen des Zustandes freigegeben zu haben.
3. Es wird die Watchdog-Anweisung verwendet ohne dass die TMSWD-Platine anwesend ist.
4. Die in einer Interpolationsanweisung definierten Parameter (linearinc, linearabs, circle, circinc, circabs, helicinc und helicabs) sind nicht konsistent. Zum Beispiel, ist die Anzahl der angegebenen Achsen ungleich der deklarierten Positionenzahl oder wurden mehr Achsen deklariert von denen, die die Anweisung verarbeiten kann.

Lösung:
Nachfolgend, sind die Lösungen für jede aufgeführte Ursache beschrieben:

1. Bewegen Sie die Anweisung, die den Fehler in einer anderen Funktion verursacht oder entfernen Sie die Anweisung.
2. Überprüfen Sie dass die Verwaltung der Zustandsalarme aktiviert worden ist. In der tpa.ini Datei, im Abschnitt [ALBATROS] unter AlarmsHaveState muss der Wert 1 immer zugewiesen sein.
3. Entfernen Sie WatchDog Bildung oder besorgen Sie eine TMSWD Karte.
4. Überprüfen Sie die Parameter der GPL-Bildung korrekt sind. Zu jeder Achse muss eine Position entsprechen. Die Anzahl der Achsen in der angegebenen Anweisung kann nicht größer sein als die Anzahl der den Systemfehler.

4373 Feed-Rate kann nicht abgelesen werden (Funktion:FunktionName Zeile: ZeilenNummer)

Ursache:
Die Instruktion GETFEED ist auf einer anderen Platine als TmsBus oder TmsCan verwendet worden.

Lösung:
In der Hardware-Konfiguration prüfen Sie dass die Platine eine Master-Platine ist.

4374 Zu viele IPC-Anweisungen in Ausführung begriffen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Die Höchstzahl von 16 gleichzeitig in Ausführung begriffenen IPC-Anweisungen ist überschritten worden.

Lösung:
Ändern Sie den GPL-Code.

4375 FASTREAD an Achsen verschiedener Platinen ausgeführt (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, eine FASTREAD-Anweisung auszuführen, an welche Parameter von Achsen übertragen worden sind, die nicht alle an derselben Platine stecken.

Lösung:
Ändern Sie den GPL-Code oder die Virtuell-physikalische Konfiguration entsprechend.

4378 Anweisung nicht freigegeben (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Der Benutzer hat versucht, eine Anweisung zu benutzen, deren Ausführung nicht freigegeben ist. Der Hardware-Schlüssel wurde nicht korrekt eingegeben oder fehlt.

Lösung:
Den Hardware-Schlüssel korrekt eingeben. Sollte das Problem weiterhin bestehen, mit dem Hersteller Kontakt aufnehmen.

4379 Anweisung nicht verwendbar in Funktionen, die von Interrupt gestartet werden (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, eine Anweisung auszuführen, die in eine von Interrupt gestarteten Funktion nicht zulässig ist. Die von Interrupt gestarteten Funktionen sind die als Parameter an die Anweisungen ONERRSYS, ONINPUT und ONFLAG übertragenen.

Lösung:
Den GPL-Code ändern. Die Liste der bei Interrupt nicht verwendbaren Anweisungen heranziehen.

4380 Zu viele Schreibversuche im Bereich des Pufferspeichers (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, gleichzeitig zu viele Schreibvorgänge im Pufferspeicher auszuführen (der Pufferspeicher hat eine relativ lange Aufrufzeit).

Lösung:
Prüfen Sie die Anweisungen, die Schreibvorgänge auf die im Pufferspeicher befindlichen Variablen ausführen: Zähler, Timer, Matrizen und als "nonvolatile" erklärte Variablen.

4381 Es ist nicht möglich, eine noch nicht geöffnete serielle Linie zu verwenden (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, eine Anweisung unter Verwendung eines seriellen Ports auszuführen, ohne den Port erst durch die Anweisung COMOPEN geöffnet zu haben.

Lösung:
Ändern Sie den GPL-Code.

4382 Es ist nicht möglich, eine bereits geöffnete serielle Linie zu öffnen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, an einem bereits durch die Anweisung COMOPEN geöffneten seriellen Port nochmals dieselbe Anweisung auszuführen.

Lösung:
Ändern Sie den GPL-Code.

4383 Versuch, zu viele Hilfsverfahren zu öffnen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, mehr als 4 Hilfsverfahren gleichzeitig zu öffnen.

Lösung:
Ändern Sie den GPL-Code.

4384 Das Hilfsverfahren läuft nicht (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, ein Hilfsverfahren aufzurufen, das nicht in Ausführung begriffen ist.

Lösung:
Ändern Sie den GPL-Code.

4385 Versuch, ein Hilfsverfahren aus einer anderen Aufgabe zu öffnen (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Man hat versucht, ein Hilfsverfahren aus einer Aufgabe zu öffnen, die nicht deren Ausführung gestartet hat. Ein Hilfsverfahren kann nur von jener Aufgabe verwendet werden, die deren Ausführung begonnen hat.

Lösung:
Ändern Sie den GPL-Code.

4391 Fehler bei der Aktivierung von SYSOK (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Die Aktivierung des Signals SYSOK hat nicht funktioniert. Normalerweise liegt es am nicht einwandfreien Betrieb des Senders des GreenBUS auf der Achsenplatine.

Lösung:
Fachleute können einen Hardware-Test des Dual-Port-Memory der Mikrosteuerung i296 ausführen. Bleibt das Problem bestehen, mit dem Hersteller Kontakt aufnehmen.

4394 Zu viele Zyklusfehler (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Mehr als 2000 Zyklusfehler sind aktiv.

Lösung:
Korrigieren Sie den GPL-Code, indem die Hinweise vermindert werden.

4395 Zu viele Nachrichten (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Mehr als 2000 Nachrichten sind aktiv.

Lösung:
Korrigieren Sie den GPL-Code, indem die Hinweise vermindert werden.

4397 Stapelüberlauf (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Der Stack einer GPL-Funktion hat die Höchstgrenze von 2Kbyte überschritten.

Lösung:
Kompilieren Sie erneut den GPL-Code und prüfen Sie im Bericht des Kompilierers die geschätzte Stack-Belegung der Funktion, die den Systemfehler hervorgerufen hat. Vermindern Sie die Anzahl an die Funktionen übertragene lokale Variablen und Parameter (z.B. indem man sie durch globale Variablen ersetzt). Vermindern Sie die CALL-Anzahl.

4398 Stapelunterlauf (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Kann nur bei einem schwerwiegenden Firmware-Fehler auftreten, z.B. im Fall einer falschen Verwaltung der Parameter einer Funktion oder der lokalen Variablen.

Lösung:
Nehmen Sie Kontakt mit dem Hersteller auf.

4399 Parameter außerhalb des Bereichs (Funktion: FunktionName Zeile: ZeilenNummer)

Ursache:
Einer GPL-Variable oder einer Vorrichtung ist ein Wert außerhalb des zulässigen Bereichs zugeordnet worden.

Lösung:
Korrigieren Sie und den GPL-Code erneut kompilieren.

4865 Die Definition der Maschine zur Interpolation (G216 oder G217) fehlt

Ursache:
Der Benutzer hat versucht, die Achsen mit einer ISO Interpolation zu bewegen oder sind die Konfiguration-Indices eingestellt worden, ohne zuerst die Matrices der Konfiguration und die Achsen, aus denen die Maschine besteht, festzulegen.

Lösung:
Kompilieren Sie erneut und durch die Anweisung ISOG216 korrigieren den GPL-Code.

4866 Die Definition der Indizes der ausgewählten Maschinenkonfiguration (M6) fehlt

Ursache:
Der Benutzer hat versucht, die Achsen mit einer ISO Interpolation zu bewegen, ohne zuerst die Indices der Maschinenkonfiguration festzulegen.

Lösung:
Durch die Anweisung ISOM6 korrigieren und den GPL-Code erneut kompilieren.

6.2.10 Von Kommunikationsdriver ausgelöste Fehler

16385 Modul nicht angeschlossen

Ursache:
Die Verbindung zwischen dem Supervisor-PC und einem Modul ist unterbrochen.
Mögliche Ursachen sind:

- fehlende Stromzufuhr zum Fernmodul
- zeitweilige Unterbrechung des Anschlusses mittels Ethernet-Kabel wegen eines defekten Kontakts in den Steckern oder wegen der Beschädigung der Kabel
- fehlende Stromzufuhr oder nicht einwandfreier Betrieb des Ethernet-Hubs (sofern vorhanden)
- Firmware des Fernmoduls wegen beschädigter Konfigurationsdateien blockiert
- Zurücksetzung der CPU des Fernmoduls wegen Überhitzung oder elektromagnetischer Störungen

Lösung:
Prüfen Sie, dass das Modul an ist. Ethernet-Kabel und -Stecker prüfen. Die Firmware am Fernmodul aktualisieren. Etwaige Überhitzung wegen mangelnder Ventilation oder elektromagnetische Störungen am Fernmodul ausschließen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16386 Modul angeschlossen

Ursache:
Ein Fernmodul hat sich nach Abschluss der Initialisierungsphase von Albatros an den Supervisor-PC angeschlossen. Während des Starts versucht Albatros alle von der Systemkonfiguration vorgesehenen Module anzuschließen. Diese Phase dauert circa 4 Sekunden; alle Module, die sich später anschließen verursachen einen Systemfehler.

16387 Modul wieder angeschlossen

Ursache:
Ein Fernmodul ist nach einer Unterbrechung des Anschlusses wieder an den Supervisor-PC angeschlossen. Dieser Fehler kann also nur infolge des Fehlers 16385 "Modul nicht angeschlossen" auftreten.

16388 Modul initialisiert

Ursache:
Ein Fernmodul ist während des normalen Betriebs reinitialisiert worden. Das bedeutet, dass der Anschluss des Moduls an den Supervisor-PC unterbrochen und wiederhergestellt worden ist. Dieser Fehler kann also nur infolge des Fehlers 16385 "Modul nicht angeschlossen" auftreten.
Dieser Fehler weist darauf hin, dass das Modul bereits rückgestellt worden ist, z.B. nach einer Unterbrechung der Stromzufuhr.

16389 Der Modul hat die Verbindung unterbrochen

Ursache:
Ein Fernmodul hat die Verbindung mit Albatros unterbrochen. Das passiert wenn der Modul lange keine Befehle oder Sendeabrufe vom Supervisor-PC bekommt. Dieser Fehler zeigt darum ein Problem (starke Verlangsamung oder Stopp) auf dem Supervisor-PC.

Lösung:
Uberprüfen Sie, dass auf dem Supervisor-PC keine Programme sind, die den Stopp oder die Systemsverlangsamung verursachen. Den Screen Saver auf dem Supervisor-PC entfähigen. Wenn das Problem anhält, setzen Sie sich mit dem Hersteller der Maschine in Verbindung.

16641 Die Steuerfirmware reagiert nicht auf Befehle

Ursache:
Bei der Initialisierung des Systems ist ein Fehler aufgetreten. Insbesondere funktioniert die Firmware nicht richtig. Das Problem kann von beschädigten Firmware-Dateien abhängen.

Lösung:
Versuchen Sie das System zurückzusetzen und eventuell Albatros erneut installieren. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16642 TpaSock reagiert nicht auf Befehle

Ursache:
Bei der Initialisierung des Systems ist ein Fehler aufgetreten. Insbesondere funktioniert die Software zur Kommunikation mit den Fernmodulen nicht richtig. Das Problem kann an beschädigten Dateien liegen.

Lösung:
Versuchen Sie das System zurückzusetzen und eventuell Albatros erneut installieren. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16643 Das Betriebssystem erlaubt es nicht, RTX zu verwenden

Ursache:
Das auf dem PC installierte Betriebssystem unterstützt RTX nicht; infolgedessen funktionieren die Versionen von Albatros nicht, auf denen RTX vorhanden ist.

Lösung:
Aktualisieren Sie das Betriebssystem des PCs. Schlagen Sie die Mindestausstattung des Systems im Installationshandbuch Albatros (InstallationGuide.pdf) nach.

16645 Fehler beim Senden des Firmware-Codes

Ursache:
Beim Initialisieren des Systems ist ein Fehler aufgetreten. Insbesondere ist die Übertragung einer Firmware-Datei an ein Modul gescheitert.

Lösung:
Versuchen Sie, die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16646 Firmware-Code konnte nicht erneut ausgeführt werden

Ursache:
Beim erneuten Initialisieren des Systems ist ein Fehler aufgetreten. Insbesondere ist der Start der Firmware gescheitert, nachdem diese bereits einmal blockiert war.

Lösung:
Versuchen Sie, die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16897 RTX wurde nicht installiert

Ursache:
Die installierte Version von Albatros erfordert das Vorhandensein von RTX auf dem PC, was aber nicht gefunden werden konnte.

Lösung:
Installieren Sie oder, sofern bereits vorhanden, installieren Sie RTX erneut. Schlagen Sie das Installationshandbuch Albatros (InstallationRTXGuide.pdf) nach.

16898 Der Benutzer hat keinen Zugang als Administrator

Ursache:
Albatros ist von einem Benutzer ohne Administrator-Rechte für den PC gestartet worden. Administrator-Rechte sind zum korrekten Betrieb von Albatros notwendig.

Lösung:
Beenden Sie die laufende Session und als "Administrator" oder anderer Benutzer mit Administrator-Rechten gewinnen Sie erneut den Zugang zum PC.

16899 Die Dimension der RAM des Moduls ist falsch

Ursache:
Die RAM-Menge am Fernmodul entspricht nicht der vorgesehenen Menge. Dieser Fehler weist meistens auf einen Hardware-Schaden hin.

Lösung:
Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16900 Die IP-Adresse des Moduls ist falsch

Ursache:
Ein Fernmodul ist ermittelt worden, dessen IP-Adresse nicht zum Unternetz des Supervisor-PCs gehört. Albatros kann also nicht korrekt mit dem Fernmoduls kommunizieren.

Lösung:
Prüfen Sie, ob die Einstellungen von AlbDHCP und der Netzwerkkarte des PCs korrekt sind. Ziehen Sie das "Installationshandbuch von Albatros" zu Rate. Schlagen Sie das Installationhandbuch Albatros (InstallationGUIDE.pdf) nach.

16901 Das Modul ist schon an eine andere Anlage angeschlossen

Ursache:
Ein Fernmodul ist ermittelt worden, das an einen anderen Supervisor-PC angeschlossen ist. Das kann daran liegen, dass im Netz ein anderer PC vorhanden ist, auf dem Albatros läuft und der dasselbe Modul verwendet. Es kann auch an einem nicht einwandfreien Betrieb der Kommunikationssoftware des Moduls liegen.

Lösung:
Prüfen Sie, dass das Modul nicht von einem anderen Supervisor-PC verwendet wird. Setzen Sie das Modul zurück. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

16902 Das Modul ist nicht konfiguriert

Ursache:
Ein Modul wurde ermittelt, das in der von Albatros "Systemkonfiguration" nicht konfiguriert ist.

Lösung:
Konfigurieren Sie das Modul.

16903 Die Einstellungen des Firewalls behindern die Kommunikation

Ursache:
Es wurde ein Firewall auf dem PC festgestellt, der die Kommunikation zwischen Albatros und den Fernmodulen behindert.

Bemerkung: Albatros ist fähig, das Vorhandensein des Windows-Firewalls zu identifizieren und nicht das der anderen Firewall, zum Beispiel der Firewall, die in einigen Antivirus-Software inbegriffen sind.

Lösung:
Entfähigen die Einstellungen des Firewalls ändern oder ihn.

16904 Netzwerkkarte nicht vorhanden oder aktiviert

Ursache:
Keine Netzwerkkarte, die für die Verbindung mit den Fernmodulen benutzbar ist, wurde festgestellt.

Anmerkung: die Tatsache, dass eine Netzwerkkarte festgestellt wird, versichert nicht, dass sie richtig konfiguriert und verbunden ist.

Lösung:
Überprüfen Sie das Vorhandensein und die richtige Konfiguration der Netzwerkkarte. Wenn das Problem anhält, sich mit dem Hersteller der Maschine in Verbindung setzen.

16905 Der Firmware-Code des Steuers fehlt

Ursache:
Albatros findet kein Firmware-File auf dem PC-Hard Disk. Das Problem kann ausbrechen, infolge des zufälligen Löschen der Firmware-Files oder infolge einer falschen Neubearbeitung.

Lösung:
Prüfen Sie, dass die Files in FW-Ordner der Installation von Albatros vorhanden und in der richtigen Version sind. Nehmen Sie Kontakt mit dem Hersteller auf.

16906 RTX-Version inkompatibel mit dem Firmware-Code des Steuers

Ursache:
Es wurde eine RTX-Version festgestellt, die nicht kompatibel ist mit der installierten Firmware.

Lösung:
Installieren Sie die richtige RTX-Version oder aktualisieren Sie die Firmware. Nehmen Sie Kontakt mit dem Hersteller auf.

16907 Version des Betriebssystems nicht kompatibel mit dem Firmware-Code des Steuers

Ursache:
Die Version des Betriebssystems des Fernmodules ist nicht kompatibel mit der installierten Firmware.

Lösung:
Installieren Sie auf dem Fernmodul die richtige Version des Betriebssystem oder aktualisieren Sie die Firmware.
Nehmen Sie Kontakt mit dem Hersteller auf.

17153 PlatinenTyp: Der Firmware-Code des GreenBUS-Senders fehlt

Ursache:
Im Ordner FW ist eine Firmware-Datei nicht gefunden worden. Das Problem liegt meistens am versehentlichen Löschen einer Datei oder an einer unvollständigen bzw. schadhaften Installation.

Lösung:
Installieren Sie nach einem Backup des Systems Albatros erneut. Wenden Sie sich an den Maschinenhersteller.

17154 PlatinenTyp: Auf dem Firmware-Code fehlt der Teil vom GreenBUS-Senders

Ursache:
Im Ordner FW ist die Datei mit dem Firmware-Code des Senders GreenBus vorhanden, aber sie ist schadhaft oder unvollständig.

Lösung:
Installieren Sie nach einem Backup des Systems Albatros erneut. Nehmen Sie Kontakt mit dem Hersteller auf.

17155 PlatinenTyp: Fehler beim Übertragen des Bootstrap-Codes des GreenBus-Senders

Ursache:
Beim Initialisieren des Systems ist ein Fehler aufgetreten. Insbesondere ist die Übertragung einer Firmware-Datei an ein Modul gescheitert.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17156 PlatinenTyp: Fehler beim Übertragen des Main-Codes des GreenBus-Senders

Ursache:
Bei der Initialisierung des Systems ist ein Fehler aufgetreten. Insbesondere ist die Übertragung einer Firmware-Datei an ein Modul gescheitert.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17157 PlatinenTyp: Der Bootstrap-Code fehlt

Ursache:
Im Ordner FW ist eine Firmware-Datei nicht gefunden worden. Das Problem liegt meistens am versehentlichen Löschen der Datei oder an einer unvollständigen bzw. schadhaften Installation.

Lösung
Installieren Sie nach einem Backup des Systems Albatros erneut. Nehmen Sie Kontakt mit dem Hersteller auf.

17158 PlatinenTyp: Der Main-Code fehlt

Ursache:
Im Ordner FW ist eine Firmware-Datei nicht gefunden worden. Das Problem liegt meistens am versehentlichen Löschen der Datei oder an einer unvollständigen bzw. schadhaften Installation.

Lösung:
Nach einem Backup des Systems installieren Sie Albatros erneut. Nehmen Sie Kontakt mit dem Hersteller auf.

17159 PlatinenTyp: Fehler beim Übertragen des Bootstrap-Codes

Ursache:
Bei der Initialisierung des Systems ist ein Fehler aufgetreten. Insbesondere ist die Übertragung einer Firmware-Datei an ein Modul gescheitert.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17160 PlatinenTyp: Fehler beim Übertragen des Main-Codes

Ursache:
Bei der Initialisierung des Systems ist ein Fehler aufgetreten. Insbesondere ist die Übertragung einer Firmware-Datei an ein Modul gescheitert.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17409 Die zusätzliche Programmdatei konnte nicht übertragen werden

Ursache:
Der Fehler kann während der Aktualisierung der Firmware eines Fernmoduls auftreten. Es kann an einem zeitweiligen Ausfall des Netzes aber auch an der Beschädigung der Firmware des Moduls liegen. Diese Fehlermitteilung kann einen Fehlercode enthalten.

Lösung:
Schalten Sie das Fernmodul aus- und ein. Wiederholen Sie das Verfahren zur Aktualisierung. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17410 Die zusätzliche Programmdatei konnte nicht in Ausführung gesetzt werden

Ursache:
Beim Initialisieren des Systems ist ein Fehler aufgetreten. Insbesondere konnte ein Hilfsprogramm nicht ausgeführt werden. Die Fehlermitteilung gibt außerdem den Namen des zusätzlichen Programms und eventuell einen Fehlercode an.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17667 DLLName: Der Firmware-Code konnte nicht in Ausführung gesetzt werden

Ursache:
Beim Initialisieren des Systems ist ein Fehler aufgetreten. Insbesondere konnte der Firmware-Code nicht ausgeführt werden. "DLLName" gibt an, welches Element den Fehler verursacht hat.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17668 DLLName: Man konnte den Zeiger auf das gemeinsame RAM nicht bekommen

Ursache:
Bei der Initialisierung des Systems ist ein Fehler aufgetreten. Insbesondere konnte der Kanal zur Kommunikation mit der Firmware nicht geöffnet werden. "DLLName" gibt an, welches Element den Fehler verursacht hat.

Lösung:
Versuchen Sie die Steuerung zurückzusetzen. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17921 NODETPA konnte nicht übertragen werden

Ursache:
Der Fehler kann während der Aktualisierung der Firmware eines Fernmoduls auftreten. Es kann an einem zeitweiligen Ausfall des Netzes aber auch an der Beschädigung der Firmware des Fernmoduls liegen. Diese Fehlermitteilung kann einen Fehlercode enthalten.

Lösung:
Schalten Sie das Fernmodul aus- und ein. Wiederholen Sie das Verfahren zur Aktualisierung. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17922 NODETPA ist nicht neu gestartet

Ursache:
Der Fehler kann während der Aktualisierung der Firmware eines Fernmoduls auftreten. Es kann an einem zeitweiligen Ausfall des Netzes aber auch an der Beschädigung der Firmware des Fernmoduls liegen. Diese Fehlermitteilung kann einen Fehlercode enthalten.

Lösung:
Schalten Sie das Fernmodul aus- und ein. Wiederholen Sie das Verfahren zur Aktualisierung. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

17923 NODETPA ist nicht in Betrieb

Ursache:
Im Netz ist ein Fernmodul ermittelt worden, dessen Software zur Kommunikation nicht in Ausführung begriffen ist. Es ist meistens ein Zeichen für den nicht einwandfreien Betrieb der Software zur Kommunikation. Diese Fehlermitteilung kann einen Fehlercode enthalten.

Lösung:
Schalten Sie das Fernmodul aus- und ein. Wiederholen Sie das Verfahren zur Aktualisierung. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

18177 NODETPA versuchte, eine ungültige Adresse abzurufen

Ursache:
Die Kommunikationsoftware eines Fernmoduls hat einen Fehler erzeugt. Diese Fehlermitteilung kann einen Fehlercode enthalten.

Lösung:
Schalten sie das Fernmodul aus- und ein. Wiederholen Sie das Verfahren zur Aktualisierung. Bleibt das Problem bestehen, nehmen Sie Kontakt mit dem Hersteller auf.

6.3 Allgemeine Signalisierungen

6.3.1 Albatros beginnt die Ausführung

Es signalisiert den Start von Albatros und zeigt einige nützliche Informationen über die Programmversion und die Ausführungsumgebung an.

6.3.2 Albatros beendet die Ausführung

Es signalisiert, dass Albatros dabei ist, die Ausrichtung zu beenden.

6.3.3 Der Computer geht in den Ruhezustand über

Es zeigt an, dass der Computer kurz vor dem Ruhezustand steht. Von diesem Moment an ist Albatros nicht mehr in der Lage, auf Anfragen und Meldungen des GPL-Zyklus zu reagieren.

6.3.4 Der Computer reaktiviert aus dem Ruhezustand

Es zeigt an, dass der Computer gerade aus dem Ruhezustand reaktiviert hat. Albatros läuft wieder, ohne neu gestartet zu werden.

6.3.5 Ausschalten des Computers

Es zeigt an, dass der Computer kurz vor dem Ausschalten steht, während Albatros noch läuft.

6.3.6 Aktuelle Zugriffsebene

Dieses Signal meldet eine Änderung der Zugriffsebene auf die Albatros-Funktionalität, in der Regel zur Durchführung von Wartungsarbeiten oder zur Änderung des Zyklus oder der Konfiguration.

6.3.7 Softwareupdate der Module

Es zeigt an, dass es aufgefordert wurde, die Software und Firmware in den Fernbedienungen zu aktualisieren.

6.3.8 Konfiguration wird den Modulen gesendet

Es zeigt an, dass es aufgefordert wurde, die Konfiguration und die Zyklen in den Fernbedienungen zu aktualisieren.

7 Systemkonfiguration

7.1 Einleitung

Im Kapitel über den Aufbau des Systems ist bereits dargestellt worden, dass ein Albatros-System aus einem oder mehreren Modulen besteht, die eine Anlage bilden und dass jede Anlage hierarchisch strukturiert ist.
Um eine Maschine aus dem Gesichtspunkt von Albatros zu konfigurieren, ist eine Aufeinanderfolge von Vorgängen zu befolgen, die der Konfiguration der verschiedenen logischen Ebenen und der zugehörigen Hardware dienen.
Zur Konfiguration eines Systems empfiehlt sich generell folgende Sequenz:

- Modulkonfiguration
- Definition von Gruppen und Untergruppen
- Konfiguration von Vorrichtungen
- Systemkonfiguration
- Hardwarekonfiguration
- Virtuell-physikalische Konfiguration

Grundsätzlich definieren die Modul-, Gruppen- und Maschinenkonfigurationen die logische Struktur der Maschine, während die System-, Hardware- und Virtuell-physikalische Konfiguration deren physikalische Struktur festlegen.
Alle angeführten Punkte werden in den folgenden Kapiteln detailliert dargestellt.

7.2 Konfiguration der Vorrichtungen

7.2.1 Einleitung

Im Kapitel Aufbau des Systems Albatros sind die Arten an Vorrichtungen dargestellt worden, die in einem Modul erscheinen können. Nachfolgend werden sie wieder aufgelistet, allerdings aus dem Gesichtspunkt ihrer Konfiguration.

Pro Art gibt es eine Höchstzahl konfigurierbarer Vorrichtungen, wie in folgender Liste erläutert:

Vorrichtungsart	**Höchstzahl**
Analoger Eingang	128
Analoger Ausgang	128
Digitaler Eingang	4096
Digitaler Ausgang	4096
Eingang-Port	512
Ausgang-Port	512
Achse	240
Timer	128
Zähler	128
Flag Bit	1024
Flag Switch	256
Flag Port	256

7.2.2 Allgemeine Vorrichtung

Für die meisten Vorrichtungen werden dieselben Konfigurationsparameter benötigt. Nachfolgend wird die Konfiguration eines digitalen Eingangs dargestellt, die aber genauso gilt für:

- Flag Bit
- Flag Switch
- Analoge Ausgänge
- Eingang-Port
- Ausgang-Port
- Flag-Port
- Timer
- Zähler

Um eine der oben aufgeführten Vorrichtungen zu konfigurieren, sind folgende Einstellungen notwendig:

- **Name:** Name der Vorrichtung mit einer maximalen Länge von 40 Zeichen.

- **Kommentar:** Kurze Beschreibung der Vorrichtung, kann in verschiedene Sprachen übersetzt werden, Leerzeichen sind nicht zulässig.
- **Zugänge zum Lesen:** Gibt an, welche Zugangsebene mindestens verlangt wird, um die Vorrichtung in den Fenstern Diagnostik oder synoptisches Schema anzuzeigen.
- **Zugänge zum Schreiben:** Gibt an, welche Zugangsebene mindestens verlangt wird, um den Zustand der Vorrichtung ändern zu können.
- **Öffentlich:** Gibt an, ob der Zustand der Vorrichtung von einem GPL-Code gelesen und geändert werden kann, der nicht zur Gruppe gehört, in der sich die Vorrichtung befindet.

7.2.3 Digitaler Ausgang

Digitale Ausgänge haben einen Parameter mehr als Standard-Vorrichtungen: *Monostabil*

Um einen digitalen Ausgang zu konfigurieren, sind folgende Einstellungen notwendig:

- **Name:** Name der Vorrichtung mit einer maximalen Länge von 40 Zeichen.
- **Kommentar:** kurze Beschreibung der Vorrichtung, kann in verschiedene Sprachen übersetzt werden, Leerzeichen sind nicht zulässig.
- **Monostabil:** wenn gewählt, konfiguriert es den Ausgang als monostabil, d.h.: wird der Ausgang auf ON gesetzt, kehrt er nach 200 ms automatisch zu OFF zurück.
- **Zugänge zum Lesen:** gibt an, welche Zugangsebene mindestens verlangt wird, um die Vorrichtung in den Fenstern Diagnostik oder synoptisches Schema anzuzeigen.
- **Zugänge zum Schreiben:** gibt an, welche Zugangsebene mindestens verlangt wird, um den Zustand der Vorrichtung ändern zu können.
- **Öffentlich:** gibt an, ob der Zustand der Vorrichtung von einem GPL-Code gelesen und geändert werden kann, der nicht zur Gruppe gehört, in der sich die Vorrichtung befindet.

7.2.4 Analoger Eingang

Analoge Eingänge haben einen Parameter mehr als Standard-Vorrichtungen: *Der Typ Eingangsspannung.*

Um einen analogen Eingang zu konfigurieren, sind folgende Einstellungen notwendig:

- **Name:** Name der Vorrichtung mit einer maximalen Länge von 40 Zeichen.
- **Kommentar:** kurze Beschreibung der Vorrichtung, kann in verschiedene Sprachen übersetzt werden, Leerzeichen sind nicht zulässig.
- **Typ:** dient der Auswahl des als Eingangsspannungen gelesenen Intervalls.
- **Zugänge zum Lesen:** gibt an, welche Zugangsebene mindestens verlangt wird, um die Vorrichtung in den Fenstern Diagnostik oder synoptisches Schema anzuzeigen.
- **Zugänge zum Schreiben:** gibt an, welche Zugangsebene mindestens verlangt wird, um den Zustand der Vorrichtung ändern zu können.
- **Öffentlich:** gibt an, ob der Zustand der Vorrichtung von einem GPL-Code gelesen und geändert werden kann, der nicht zur Gruppe gehört, in der sich die Vorrichtung befindet.

7.2.5 Achse

Basisdaten

Die anzugebenden Basisdaten sind:

- **Name:** Name der Vorrichtung mit einer maximalen Länge von 40 Zeichen.
- **Beschreibung:** Kurze Beschreibung der Vorrichtung, kann in verschiedene Sprachen übersetzt werden, Leerzeichen sind nicht zulässig.
- **Auflösung:** Auflösung des Encoders, hängt von den Eigenschaften des Encoders und von der angegebenen Maßeinheit ab. Zu beachten ist, daß die Platinen der Achsen von Albatros sowohl die Anstiegsflanken als auch die fallenden Flanken beider Encoder-Phasen als Impuls zählen (ein Encoder à 2500 Impulse pro Umdrehung wird also wie ein Encoder à 10000 Impulse pro Umdrehung gelesen).
- **AchsenTyp:** Vorgesehen ist ***Analog*** (analog gesteuert), ***Schrittschaltung***, ***Zählen*** (nur Encoder lesen), ***Digital***, ***Virtuell***.
- **Maßeinheit:** Zum Ausdruck der Achsenmaße verwendete Maßeinheit. Da von ihr alle abgeleiteten Größen abhängen, empfiehlt es sich, diesen Parameter vor allen anderen einzustellen.
- **Phasenumkehr:** Ermöglicht es, per Software eine etwaige Kabel-Inversion der Encoder-Phasen auszugleichen.
- **Bezugsumkehr:** Ermöglicht es, den Geschwindigkeitsbezug der Achse umzukehren. Zusammen mit der Phasenumkehr erlaubt es, die Richtung der Achse umzukehren (wenn die Verkabelung korrekt ist).
- **Kerbfreigabe:** Nur für Achsen des Typs Zählen verfügbar; stellt beim Lesen der Encoder-Kerbe das Maß automatisch auf Null.

Bewegungsparameter

Zur Punkt-Punkt - Bewegung der Achse verwendete Parameter.

- **Höchstgeschwindigkeit:** Höchstgeschwindigkeit der Achse.
- **Beschleunigung:** Dauer der Beschleunigungsrampe
- **Verzögerung:** Dauer der Verzögerungsrampe
- **Mindestgeschwindigkeit:** Nur bei Achsen mit Schrittschaltung einstellbar; vom Motor in einem Schritt erreichte Geschwindigkeit.
- **Rampentyp:** Beschleunigungs- und Verzögerungsrampentyp; bei Schrittmotoren nicht einstellbar.
- **Proportional:** Proportionaler Koeffizient der PID-Steuerung des Position Loop
- **Integral:** Integraler Koeffizient der PID-Steuerung des Position Loop
- **Ableitung:** Derivativer Koeffizient der PID-Steuerung des Position Loop
- **Vorschub vorwärts:** Prozentsatz des Vorschubs vorwärts. Ermöglicht den entsprechenden Ausgleich des Loop Fehlers bei gleicher Geschwindigkeit.
- **Vorschub vorwärts Beschleunigung:** Prozentsatz der Feed Forward Beschleunigung. Ermöglicht den Ausgleich des restlichen (nicht vom Vorschub vorwärts ausgeglichenen) Loop Fehlers während der Beschleunigungs- und Verzögerungsphasen der Achse.
- **Integral-Proben:** Bestimmt die Proben-Anzahl des Loop Fehlers, die zur Berechnung der integralen Komponenten herangezogen wird. Gültige Werte sind zwischen 1 und 200 inbegriffen. Der Default-Wert beträgt 50. Siehe SETINTEGTIME - GPL-Anweisung

Interpolationsparameter

Zur Interpolationsbewegung der Achse verwendete Parameter.

Diese Parameter haben dieselbe Bedeutung der Bewegungsparameter mit Ausnahme von der Mindestgeschwindigkeit, die erst für die Achsen mit Schrittschaltung in den Bewegungen Punkt-Punkt definiert wird. Sie werden jedoch für Interpolationsbewegungen verwendet.

Anm.: Die in den Interpolationsparametern eingestellte Beschleunigung- und Verzögerungswerte dürfen niedriger als die in den Bewegungsparametern entsprechenden Werte nicht sein.

Weitere Parameter

- **Manuelle Geschwindigkeit:** Gibt die maximale Höchstgeschwindigkeit an, die bei den manuellen Bewegungen verwendet werden kann. Sie wird nie höher als die vorgegebene Höchstgeschwindigkeit sein.
- **Dynamisches Servoerror:** Aktiviert oder deaktiviert den dynamischen Servoerror. Der Standardwert ist der deaktivierte und dynamische Servoerror, so bleibt der Servo-Fehler der Schwelle aktiviert. Siehe die GPL-Anweisung SETMAXERTYPE.
- **Referenzgeschwindigkeit** und **Loop Fehler**: Diese beiden Werte werden verwendet, um den realen Zusammenhang zwischen den beiden Größen in der Maschine zu berechnen. Damit die Werte berücksichtigt werden können, müssen diese positiv und ungleich Null sein und das Feld **Dynamischer Servoerror** muss aktiviert werden.
- **Wartezeit stillstehender Achse**: Aktiviert oder deaktiviert die Overshoot-Wiederherstellungsfunktion. Dieser Parameter sorgt für eine Wartezeit von 50 ms am Ende jeder Bewegung.
- **Timeout Achse Bewegung:** Gültige Werte liegen im Bereich von 0 bis 1024. Siehe GPL-Anweisung ENABLESTARTCONTROL.
- **Falsche Grenze der Encoder-Verbindung:** Die eingesellten Werte sind in der Maßeinheit ausgedrückt, in der die Achsenauflösung ausgedrückt ist. Die einstellbaren Werte liegen im Bereich von 128/Achsenauflösung und 16384/Achsenauflösung. Der standardmäßige Wert wird auf der Grundlage einer Schrittzahl von 1024, d.h. 1024/Achsenauflösung berechnet.
- **Positive Servoerror Grenze:** Höchstwert des Loop Fehlers für den Regel-Loop in positiver Richtung.
- **Negative Servoerror Grenze:** Höchstwert des Loop Fehlers für den Regel-Loop in negativer Richtung.
- **Positive Achsebegrenzung:** Höchstwert des Achsenhubs in positiver Richtung.
- **Negative Achsebegrenzung:** Höchstwert des Achsenhubs in negativer Richtung.
- **Positives Fenster Positionierungstoleranz:** Positionierungstoleranz in positiver Richtung.
- **Negatives Fenster Positionierungstoleranz:** Positionierungstoleranz in negativer Richtung.

Bezugsparameter

- **Bezugspannung:** Wert der Bezugsspannung, der die Höchstgeschwindigkeit entspricht.
- **Automatisches Adjust:** Aktiviert bzw. deaktiviert die Berechnung der Kompensation des automatischen Offsets. Ist normalerweise aktiviert.

- **Anfangsoffset:** Bezugswert für den Anfangsoffset. Der Wert hat zwischen –10 und 10 inbegriffen zu sein. Der Default-Wert lautet 0.
- **Notch Filter Frequenz:** Wert der zu filtrierenden Frequenz. Der Wert hat zwischen 0 und 500 inbegriffen zu sein.
- **Mindestspannung:** Dient der Einstellung der Mindestspannungsparameter für die angegebene Achse. Der negative Wert hat zwischen –10 und 0, der positive zwischen 0 und +10 inbegriffen zu sein. Siehe die SETDEADBAND-GPL-Anweisung.
- **Schwelle:** Dient der Einstellung der Schwellenwerte. Diese sind immer kleiner oder gleich den zugehörigen Werten der Mindestspannung, d.h. der negative Schwellenwert hat immer zwischen 0 und dem negativen Mindestspannungswert inbegriffen zu sein. Der maximale Schwellenwert hat immer zwischen 0 und dem positiven Mindestspannungswert inbegriffen zu sein.

Zugangsebenen

- **Zugänge zum Lesen:** Gibt an, welche Zugangsebene mindestens verlangt wird, um die Vorrichtung in den Fenstern Diagnostik oder synoptisches Schema anzuzeigen.
- **Zugänge zum Schreiben:** Gibt an, welche Zugangsebene mindestens verlangt wird, um den Zustand der Vorrichtung ändern zu können.
- **Öffentlich:** Gibt an, ob der Zustand der Vorrichtung von einem GPL-Code gelesen und geändert werden kann, der nicht zur Gruppe gehört, in der sich die Vorrichtung befindet.

Achsenverkettung

Parameter der Achsenverkettung. Es handelt sich um die Koeffizienten der PID-Steuerung, die die Unterschiede des Loop Fehlers zwischen Master-Achse und Slave-Achsen ausgleicht.

- **Proportional:** Proportionaler Koeffizient
- **Integral:** Integraler Koeffizient
- **Ableitung:** Derivativer Koeffizient

Linearitätskorrektoren

Einstellung der Linearitätskorrektoren der Achse. Die Korrektoren erlauben den Ausgleich von Positionierfehlern einer Achse, die auf der Ungenauigkeit der Achsenmechanik beruhen (Autokorrekturen) und von Fehlern, die durch die Wirkung der anderen Achsen der Maschine verursacht werden (gekreuzte Korrekturen), was typischerweise mit der Nachgiebigkeit der Struktur zusammenhängt. Die Korrektoren sind nicht automatisch aktiviert, sondern sind im Fenster zur Änderung der Korrekturwerte (Schaltfläche **[Bearbeiten...]**) freizugeben und im GPL-Code durch die Anweisung ENABLECORRECTION zu aktivieren.

- **Korrekturbereich:** Dient der Einstellung des Abstands zwischen zwei Korrekturen. Die Anzahl der Messungen ist durch die Länge der Achse geteilt durch die Länge des Korrekturintervalls gegeben.
- **Name Korrektur-Datei:** Dient der Vorgabe des Namens der Datei, in der die Korrekturwerte gespeichert werden. Es handelt sich um eine ASCII-Datei, in der die Werte durch das Zeichen ";" getrennt sind. Auf diese Weise können sie mit einem normalen Text-Editor bearbeitet werden. Der Zusatz der Datei braucht nicht angegeben zu werden, da automatisch der Zusatz ".csv" (comma separated values) zugewiesen wird.
- **Korrekturdaten:** Sie erlauben, die Liste der Achsen zu spezifizieren für die die laufende Achse eine Korrektur erzeugt. Die laufende Achse ist immer schon in der Liste vorgesehen, d.h. die Autokorrektur ist immer vorhanden. Es können bis zu 5 weitere Achsen angegeben werden. Um eine Achse hinzuzufügen, diese in der Liste links markieren und die Schaltfläche **[>>Hinzufügen]** drücken. Um eine Achse zu löschen, diese in der Liste rechts markieren und die Schaltfläche **[Entfernen<<]** drücken. Um die Korrekturwerte anzugeben, in der Liste rechts eine Achse markieren und die Schaltfläche **[Bearbeiten...]** drücken. Es erscheint ein Fenster mit einer Tabelle zum Einfügen der Korrekturwerte.

ANMERKUNG: Das System verwaltet eine Höchstzahl von ***235*** Linearitätskorrektoren pro Achse. Dementsprechend hat bei einer gegeben Achsenlänge das Meßintervall größer oder gleich dem zweihundertfünfunddreißigsten Teil der Achsenlänge zu sein. Bei einer 2500 mm langen Achse hat z.B. das Korrekturintervall größer oder gleich 10,63 mm zu sein. Es gibt außerdem eine Grenze für den Höchstwert einer einzelnen Korrektur, die kleiner als 1024 Encoderschritte zu sein hat. Bei einer Achse mit einer Auflösung von 256 Schritten/mm beträgt z.B. die maximale Korrektur ±4 mm.

7.3 Logische Konfiguration

7.3.1 Konfiguration einer Anlage

Um eine neue Maschine zu definieren oder eine bereits bestehende zu ändern, ist die Seite Modulkonfiguration aufzurufen. Die Modulkonfiguration ist die Konfiguration der die Anlage bildenden Module.
Das Öffnen der Konfigurationsumgebung ist auf der Zugangsebene Hersteller oder höher möglich.

Aufruf der Konfiguration

Im Menü **Datei** wählen Sie die Option ***Konfiguration öffnen***.

Sind in der Anlage noch keine Module konfiguriert, wird automatisch die Modulkonfiguration geöffnet; ansonsten die Maschinenkonfiguration. In diesem Fall folgendermaßen vorgehen, um die Modulkonfiguration aufzurufen:

Im Menü ***Bearbeiten*** wählen Sie die Option **Konfiguration des Moduls**.

Um ein Modul zur Anlage hinzuzufügen, einfach die Schaltfläche **[Neu]** drücken. Um Daten eines bereits existierenden Moduls zu ändern, die Schaltfläche **[Ändern]** drücken; um ein Modul zu löschen, die Schaltfläche **[Löschen]** drücken; um die Umgebung Konfiguration einer Anlage zu verlassen, die Schaltfläche **[Schließen]** drücken.

Bei den Daten, die eine Maschine kennzeichnen, und die also anzugeben sind, handelt es sich um:
- Die Nummer des Moduls: eine fortlaufende, ganze Zahl, die vom System vergeben wird, wenn nicht anders angegeben
- Eine kurze Beschreibung.

Es gibt einige Daten, die die zugehörige Hardware betreffen:
- **Frequenz Achsenkontrolle**: Zeigt die Frequenz, mit der die Daten zwischen der numerischen Steuerung und den mit ihr über den Feldbus angeschlossenen Geräten periodisch ausgetauscht werden.
- **Anzahl Interpolationskanäle**: Zeigt die maximale Anzahl von Interpolationskanälen (d.h. Gruppe von Achsen, die eine interpolierte Bewegung ausführen), die gleichzeitig verwaltet werden können.
- **Prozentsätze der Verwendung der CPU**: Zeigt den Prozentsatz der Zeit in Bezug auf die Periode der Achsensteuerung (d.h. die Umkehrung der „Frequenz der Achsensteuerung") die für die Ausführung der Firmware reserviert ist.

Dasselbe Fenster kann vom Zweig des Konfigurationsmoduls der Gruppen, vom Zweig des Maschinenkonfigurationsmoduls, und vom Zweig des Hardware-Konfigurationsmoduls geöffnet werden.

7.3.2 Konfiguration der Gruppen

Wenn man zum ersten Mal eine Maschine entwickelt, ist jedes Bauteil zu definieren und alle Kontrollzyklen müssen geschrieben werden. In vielen Fällen beginnt die Entwicklung auf der Basis einer bereits hergestellten Maschine, die der Funktionen der neuen Maschine zufolge geändert wird.

Erstellung einer Gruppe

Zum Erstellen einer neuen Gruppe ist die Seite der Gruppenkonfiguration aufzurufen. Der erste Zweig des Baumes ist das Modul, von dem alle Gruppen, Untergruppen und Vorrichtungen abgeleitet sind. Wird die **[Eingabetaste]** oder die Taste **[Ändern]** gedrückt, dann wird das Dialogfeld geöffnet, um die Daten des Moduls zu ändern.

Wählen Sie Im Menü ***Bearbeiten*** die Option ***Gruppen***.

Hier können neue Gruppen erstellt und bereits bestehende Gruppen geändert oder gelöscht werden.

Befehlsleiste zum Erstellen, Ändern, Löschen, Kopieren und Einfügen der Gruppen, Untergruppen und Vorrichtungen.

Befehl	Aktion
Erstellt eine neue Gruppe, Untergruppe, Vorrichtung	[Strg+Enter], Schaltfläche [Neu], Bearbeiten->Neu, Kontextmenü

Ändert eine Gruppe, Untergruppe, Vorrichtung	[Enter], Schaltfläche [Ändern], Bearbeiten->Ändern..., Kontextmenü
Entfernt eine Gruppe, Untergruppe, Vorrichtung	[Entf], Schaltfläche [Löschen], Bearbeiten->Löschen, Kontextmenü
Aktiviert oder deaktiviert die Verwendung einer Gruppe auf der Maschine	Kontextmenü, Schaltfläche [Aktivieren].
Kopiert eine Gruppe, Untergruppe, Vorrichtung	[Strg+C], Schaltfläche [Kopieren], Bearbeiten->Kopieren, Kontextmenü
Fügt eine Gruppe, Untergruppe, Vorrichtung ein	[Strg+V], Schaltfläche [Einfügen], Bearbeiten->Einfügen, Kontextmenü

Als eine neue Gruppe erstellt wird, dann erscheint ein Fenster, in dem
- der Name der Gruppe;
- ein Kommentar (übersetzbar in den von Albatros unterstützten Sprachen)

anzugeben sind.

Eine Gruppe kann außerdem als ***Gruppenübergreifend*** gekennzeichnet werden. Gruppe als gruppenübergreifende Gruppe zu kennzeichnen, weil diese Einstellung von Albatros zur Identifizierung der "Haupt"-Gruppe der Maschine verwendet wird. Dabei handelt es sich um die Gruppe, deren Hauptfunktion (die mit demselben Namen der Gruppe) beim Starten automatisch ausgeführt wird. Dieser Mechanismus wird verwendet, um die Maschine zu initialisieren und Aufgaben beim Start auszuführen, die sicherstellen, dass alles ordnungsgemäß funktioniert, bevor die Steuerung an den Benutzer übergeben wird.
Wenn Sie eine Gruppe deaktivieren, in der Geräte mit physischen Geräten verbunden sind, werden sie gefragt, ob Sie die physische-virtuelle Verbindung löschen möchten. Wenn Sie sich dafür entscheiden, die Verbindungen beizubehalten, werden die Pins der physikalischen Geräte, mit denen sie verbunden sind, in der graphischen Darstellung der physikalisch-virtuellen Verbindung grau dargestellt.

Hinzufügen einer Untergruppe zu einer Gruppe
Um die Untergruppe einer Gruppe zu erstellen muss man sich auf der Gruppe positioniert sein.

Wenn man der Untergruppe keinen spezifischen Namen anzugeben will, markieren Sie einfach das Feld *Liste der Vorrichtungen* und drücken Sie **[OK]**. Der Name der Untergruppe wird automatisch zugewiesen.

Nun können einzelne Vorrichtungen in die Untergruppegruppe eingefügt werden. Die Vorgehensweise ähnelt der zur Erstellung von Untergruppen. Ein Fenster erscheint mit der Liste der verfügbaren Vorrichtungen.

Nach der Auswahl der gewünschten Vorrichtung drücken Sie **[OK]** zum Bestätigen.

Ein Fenster erscheint, in dem der Name, ein Kommentar und sonstige Daten, die je nach Vorrichtung unterschiedlich sind, eingegeben werden können. Eine detaillierte Beschreibung der verschiedenen Arten von Vorrichtungen und der zugehörigen Einstellungen folgt im Kapitel Konfiguration der Vorrichtungen.

Kopieren einer Vorrichtung

Die Funktion Kopieren einer Vorrichtung dient der Erstellung einer Kopie einer Vorrichtung. Markieren Sie erst die Vorrichtung, dann drücken Sie die Schaltfläche **[Kopieren]**. Zum Einfügen der Vorrichtung in die Liste müssen Sie den Zweig auswählen, in dem die Vorrichtung einzufügen ist, und den Befehl **[Einfügen]** aktivieren. Der neue Vorrichtungsname muss in das Dialogfenster eingegeben werden.

Kopieren einer Gruppe
Die Funktion Kopieren einer Gruppe dient der Erstellung einer Kopie einer Gruppe einschließlich aller darin enthaltenen Untergruppen und Vorrichtungen. Außerdem wird das etwaige der Gruppe zugeordnete synoptische Schema kopiert (synoptisches Schema mit demselben Namen der Gruppe).
Das ermöglicht das rasche Erstellen von Gruppen, deren Struktur der von bereits vorhandenen Gruppen ähnelt, ohne sämtliche Vorrichtungen einzeln wieder erstellen zu müssen. Um eine Gruppe zu kopieren, markieren Sie erst die zu kopierende Gruppe, dann drücken Sie die Schaltfläche **[Kopieren]** und geben Sie im Dialogfenster den Namen der neuen Gruppe ein.

Die Kopie der Vorrichtungen, Untergruppen und Gruppen kann auch unter verschiedenen Modulen ausgeführt werden.

Auswahl der zur Maschine gehörenden Gruppen

Nach dem Erstellen des Gruppenarchivs müssen die tatsächlich vorhandenen Gruppen deaktiviert oder aktiviert werden.
Die Gruppen sind alle auf dem Computer vorhanden, sofern Sie entscheiden sich, sie durch die Schaltfläche **[Deaktivieren]** oder durch Auswählen des gleichen Befehls aus dem Kontextmenü zu deaktivieren. Wurde eine Gruppe deaktiviert, erscheint neben dem Gruppennamen das Wort **Nicht vorhanden**.

Um nur die auf der Maschine vorhandenen Gruppen anzuzeigen, wählen Sie aus dem Menü ***Bearbeiten*** den Eintrag ***Maschine.***
Um eine neue Gruppe einzufügen, müssen Sie **[Hinzufügen]** drücken. Ein Fenster mit der Liste der im Archiv vorhandenen und noch nicht in die Maschine eingefügten Gruppen erscheint.

Wählen Sie nun die gewünschte Gruppe aus und **ziehen Sie sie mit der Maus** ins Fenster der Maschinenkonfiguration oder wählen Sie **[Eingeben]**.
Sie können auch die Schaltfläche **[Entfernen]** verwenden, um eine vorhandene Gruppe zu entfernen oder nach einem Gruppennamen oder Gerätenamen im Kompositionsbaum der Maschine zu suchen.

In einer Maschine muss eine einzige Intergruppe vorhanden sein.

7.4 Physikalische Konfiguration

7.4.1 Systemkonfiguration

Die Systemkonfiguration erlaubt die Zuordnung der physikalischen Betriebsmittel (Steuereinheiten) zu den in der logischen Konfiguration definierten Modulen. Das wird in der Dialogbox der Systemkonfiguration ermöglicht. Die Modulliste der Anlage wird dargestellt, jeder von ihnen ein Netzknot zugeordnet werden kann.

- **Lokaler Knoten:** "Lokale" Systeme, in denen die HW, die die Steuerung implementiert, direkt auf dem PC installiert ist, der die Schnittstelle zum Benutzer des Systems darstellt.
- **Name eines Netzknotens:** Fernsteuerungssysteme, in denen die HW, die die Steuerung implementiert, per Netz an den PC angeschlossen ist.
- **Nicht konfiguriert**: Keine Konfiguration. Das ist die Standardeinstellung am Anfang. Sollte diese Auswahl bestätigt werden, dann wird im Dialogbox **Anschluss der Netzknoten** ein Fernmodul verbunden.

Bis zu 16 Module können konfiguriert werden und nur eines kann als lokaler Knoten konfiguriert werden.
Um ein Modul zuzuordnen, soll die Schaltfläche ausgewählt werden **[Bearbeiten]** oder soll man auf dem zu bearbeitenden Netzknoten mit der Maus doppelklicken. Nachdem sich ein Dropdown-Menü geöffnet hat, wird die Liste der verfügbaren Fernmodulen dargestellt, in denen der Benutzer entweder einen lokalen Knoten verwenden oder das Modul als nicht konfiguriert einstellen kann. Um die Auswahl zu bestätigen, soll die Schaltfläche ausgewählt werden
Führt man die Auswahl fort, werden nacheinander bis zur leeren Position alle verfügbaren Vorrichtungen aufgerufen  .

NB: Der Funktionsablauf in der Maschine von Albatros ist von einem USB - Hardwareschlüssel geschützt, der von TPA konfiguriert wird.

7.4.2 Hardware-Konfiguration

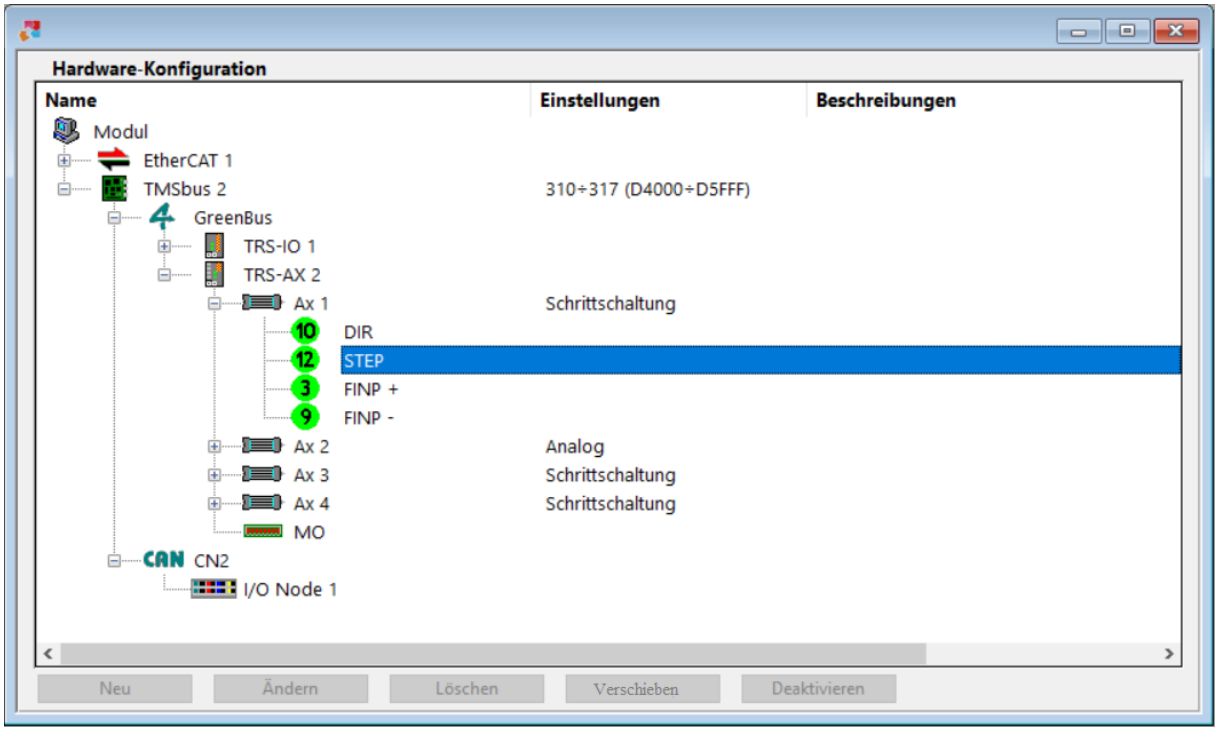
In der Hardwarekonfiguration werden die Platinen und die Knoten definiert, die das System zusammensetzen.
Die Platine in der ersten Position ist Master-Platine genannt.

Folgende Platinentypen können konfiguriert werden:

- TMSbus bis zu zwei
- TMSbus+ bis zu vier
- TMSCombo+ bis zu vier
- DualMech bis zu vier
- DualMech Mono bis zu vier
- TMSCan bis zu zwei
- TMSCan+ bis zu vier
- AlbMech bis zu zwei
- EtherCAT eins

Beschreibung des Hardware-Konfigurationsdialogfeldes

Die Öffnung des Hardware-Konfigurationsfeldes erfolgt durch die Auswahl der Option ***Bearbeiten->Hardware.***
Um eine Platine, ein E/A-Fernmodul, einen CAN-, oder einen EtherCAT-Knoten einzusetzen, muss man die Schaltfläche **[Neu]** drücken. So, erscheint ein Fenster, die die Auswahl der Platine oder des E/A Fernmoduls ermöglicht, sowie des CAN-Bus und GreenBUS Eingabeposition.
Pro Modul können bis zu 4 Platinen konfiguriert werden, und je nach dem Platine- oder Bustyp können bis zu einer veränderlichen Anzahl Module und Fern-E/A konfiguriert werden.



Hardwarekonfiguration

In der Spalte **Einstellungen** werden die Informationen über die Platine oder über den Knoten zugeordnet.
Mittels des Befehls **[Verschieben]** kann man eine Platine, einen Knoten, eine Erweiterung von einem TRS-IO oder einem TRS-CAT von einer Position zu einer anderen des Baums verlegen. Die Erweiterungen können nur innerhalb des selben Knotens verschoben werden. Dieser Vorgang erfolgt unter der Beibehaltung der Verbindungen in der Konfiguration Virtuell-Physisch.

Zudem kann ein Knoten oder eine Erweiterung von einem TRS-IO oder einem TRS-CAT deaktiviert werden.
Die Deaktivierung bewirkt die Beibehaltung der Verbindungen mit der virtuell-physischen Konfiguration. Wenn der Knoten zu einem GreenBUS-Bus gehört, werden der Knoten und die daran angeschlossenen Geräte vom System vollkommen ignoriert. Es wird also kein Fehler erzeugt, wenn das Modul während der Initialisierung nicht erkannt wird und es wird auch kein Fehler erzeugt, wenn an einem dem Modul zugeordneten Gerät eine GPL-Anweisung ausgeführt wird.
Wenn der Knoten oder die Erweiterung zu dem EtherCAT-Netz gehört oder wenn die Erweiterung zu einem GreenBUS-Bus gehört, dürfen sie nicht im Netz vorhanden sein. Wenn in GPL auf ein nicht angeschlossenes Gerät zugegriffen wird, tritt ein Systemfehler auf.
Beim Verwenden dieser Funktion ist demnach besondere Aufmerksamkeit geboten.

Um einen Knoten oder eine Erweiterung zu deaktivieren, muss man den Befehl **[Deaktivieren]** verwenden; um einen Knoten oder eine Erweiterung neu zu aktivieren, muss man den Befehl **[Aktivieren]** verwenden.

Vordefinierte Konfigurationen

Es sind mehrere Standardkonfigurationen verfügbar. Aus dem Menü **Bearbeiten->Steuerungstyp ändern** oder aus dem Kontextmenü auf dem Modulzweig können Sie die gewünschte Konfiguration auswählen. Wenn die Konfiguration neu ist, wird der Baum mit den definierten Platinen und Knoten

bestückt, andernfalls wird geprüft, ob die bereits vorhandene Hardware mit dem ausgewählten Modultyp kompatibel ist. Alles, was nicht kompatibel ist, wird gelöscht.

Den Knoten eines TPA-Busses konfigurieren

Folgende I/O-Fernmodule können auf GreenBUS (v3.0) konfiguriert werden:

- Albre8 8 digitale Eingänge und 8 Ausgänge
- Albre16 16 Kanäle, die per Software als digitale Ein- oder Ausgänge konfiguriert werden können
- Albre24 24 digitale Eingänge und 24 Ausgänge
- Albre48 48 digitale Eingänge und 48 Ausgänge
- Albrem 10 Eingang-Ports und 10 Ausgang-Ports
- AlbSTEP 8 digitale Eingänge und 6 Ausgänge, ein Schrittmotor
- AlbEV 20 oder 24 Elektroventile (D-sub 25 Pin-Stecker)
- Albrea 4 analoge Eingänge und 4 Ausgänge

Folgende Platinentypen können auf GreenBUS (v4.0) konfiguriert werden:

- TRS-AX 4 analoge Achsen oder Schrittachsen
- TRS-EV-24 24 Elektroventile (D-sub 25 Pin-Stecker)
- TRS-16 16 Kanäle, die per Software als digitale Ein- oder Ausgänge konfiguriert werden können
- TRS-IO 16 Kanäle, die via Software als digitaler Ein- oder Ausgang mit TRS-IO-E und TRS-AN-E Modulen bis zu max. 5 und TRS-AC-E Module konfiguriert werden können
- TRS-IO-E 16 Kanäle, die per Software als digitaler Ein- oder Ausgang konfiguriert werden können. Sie können nur als Erweiterungen eines TRS-IO Moduls gebraucht werden
- TRS-AN-E 1 analoger Eingang und 1 analoger Ausgang, die nur als Erweiterungen eines TRS-IO Moduls gebraucht werden kann.
- TRS-REM allgemeines Fernmodul zum Herstellen von speziellen Modulen. Man kann bis 12 Eingang- Ports und 12 Ausgang-Ports verbinden.
- TRS-AC-E 1 Zählachse und 2 Digitaleingänge, die als Nullmarke und Fast Input konfiguriert werden können. Die nachfolgende Tabelle beschreibt die Höchstzahl von TRS-AC-E, die in einem TRS-IO konfiguriert werden können. Wenn 3 totale Erweiterungen in TRS-IO-E und TRS-AN-E anwesend sind, kann nur eine Erweiterung TRS-AC-E konfiguriert werden. Wenn nur eine Erweiterung (TRS-IO-E oder TRS-AN-E) konfiguriert wird, erhält man bis zu 2 Erweiterungen TRS-AC-E.

Die TRS-AX, TRS-IO, TRS-REM und TRS-16 Fernmodule können ausschließlich an TMSbus, TMSbus+ e TMSCombo+ Platinen angeschlossen werden.

An jede TMSbus und TMSbus+ Platine kann man höchstens 4 TRS-AX-Fernmodule anschließen.
Die auf dem EtherCAT-Bus konfigurierbaren Fernmodule von Tpa sind:

- TRS-CAT 16 Kanäle, konfigurierbar via Software als digitaler Ein- oder Ausgang, erweiterbar durch TRS-IO-E, TRS-AN-E und TRS-AC-E Module;
- STAR-CAT mit Hilfe eines Eingangskanals und max. 3 verschiedener Ausgangskanäle wird eine EtherCAT-Netzwerk-Topologie in eine Sterntopologie umgewandelt

Die nachfolgende Tabelle beschreibt die Höchstzahl der in einem TRS-CAT konfigurierbaren

Zahl der TRS-IO-E und TRS-AN-E Erweiterungen	Zahl der TRS-AC-E Erweiterungen
7	0
5	1
3	2
1	3

Für die TRS-AX Fernmodule passiert das, dass indem die Anzahl der eingefügten TRS-AX erhöht, verringert die Anzahl von TRS-16 und TRS-IO, die gebraucht werden können. Um die Höchstzahl von TRS-16 und TRS-IO, die eingefügt werden können, zu berechnen, muss folgende Formel angewandt

werden: Anzahl von anderen Fernmodulen =32-(Anzahl von TRS-AX * 4). Z.B., sind mit einer TMBus Platine 3 TRS-AX verbunden und wenden wir die obengenannte Formel an, erhalten wir: Anzahl von anderen Fernmodulen 32-(3*4), dann kann eine Höchstzahl von 20 TRS-16 und/oder TRS-IO Fernmodulen eingefügt werden.
Die Position des Fernmoduls soll gemäß der eingestellten Adresse durch Switch auf Fernmodul gewählt werden. Bitte, beziehen Sie sich auf die Hardware-Dokumentation des einzelnen Fernmolduls.

Wird ein TRS-AX Fernmodul ausgewählt, kann es notwendig sein, den verwalteten Achsentyp einzustellen.

In der folgenden Übersicht beschreibt man welche Achsentype mit verschiedener Hardware assoziiert werden können.

- AlbMech-Platine — Digitale Achsen
- DualMech-Platine — Digitale Achsen
- DualMech-Mono-Platine — Digitale Achsen
- TRS-AX-Fernmodul — Analoge Achsen (wenn der spezifische Anschluss analog ist), Zählachsen (wenn der spezifische Anschluss analog ist), Achsen mit Schrittschaltung (wenn der spezifische Anschluss ein Schrittschaltungsanschluss ist)
- AlbStep-Fernmodul — Achsen mit Schrittschaltung
- TRS-AC-E-Erweiterung — Zahlachse

In den MECHATROLINK-II-Platinen kann jede Achse in Lageregelung oder Geschwindigkeitsregelung (Standard) verwaltet werden. Die Art der Steuerung kann für jede Achse im Fenster Hardware-Konfiguration in der Spalte **Einstellungen** geändert werden. Die Anzahl der konfigurierbaren Achsen variiert je nach eingestelltem Achsen-Steuerfrequenzwert:

Platine	Frequenz Achsenkontrolle (Hz)	Höchstzahl der Servoantriebe
AlbMech	1000	8
AlbMech	<=500	16
DualMech Mono	1000	8
DualMech Mono	500	20
DualMech Mono	250	30
DualMech	1000	16
DualMech	500	40
DualMech	250	60

Knoten eines CAN-Busses konfigurieren

Steuerplatine des Busses

Albatros kann Geräte auf CAN-Feldbussen durch TPA-Platinen verwalten, die mit einem CAN-Bus-Anschluss ausgestattet sind. Der Can-BUS ist auf den Platinen **TMSbus, TMSbus+, TMSCan+ und TMSCan** vorhanden.

Konfiguration der grundlegenden Daten und Dienste
Die Daten des CAN-Busses werden in der Hardwarekonfiguration definiert. Wählen Sie den Can-Bus aus, dessen Parameter definiert werden sollen, und klicken Sie auf die Schaltfläche **[Bearbeiten].**
Die Grunddaten sind:

- **Abtastzeit (TIME)**: Abtastzeit in ms. Sie kann nicht länger als 60000 (60 Sekunden) sein. Der Standardwert ist 2. Auf S-CAN-Bus wird nur der Wert 2 akzeptiert.
- **Zeit für PDO synchrone Kommunikation (TIMEPDO)**: Zeit in Millisekunden. Gibt die Zeit an, die für die synchrone Kommunikation von PDOs benötigt wird. Der eingestellte Wert kann nicht höher als der TIME-Wert sein (es ist kein Pflichtwert).
- **Wartezeit (TIMEAFTERRESET)**: Zeit in ms angegeben. Das zeigt die Wartezeit während der Anfangsphase infolge einer Software-Rückstellung der Knoten in der Netz, an. Sie kann nicht länger als 60000 (60 Sekunden) sein.

- **Anzahl der unbeantworteten CAN Zyklen (LIFETIMEFAKTOR)**: Das ist die Anzahl der CAN-Zyklen ohne Antwort der Node-Guarding-Anruf, ehe der Fehler des getrennten Knotens erzeugt wird. Der Wert kann nicht weder größer 100 noch kleiner als 1 sein (Standardwert ist 3).
- **Baud Rate:** Zahl Übertragungsgeschwindigkeit in Kilobits pro Sekunde (kann 1000, 500, 250, 125, 100 sein)

Dienste können aktiviert oder deaktiviert werden, indem die Option, die sich auf den Dienst bezieht, ausgewählt oder nicht wird. Dem Feld **Extra** können Werte zugewiesen werden, die eine vom Maschinenhersteller festgelegte Bedeutung haben. Dieser Wert wird nicht kontrolliert. Standardwert liegt bei 16.

Der Knoten CAN

Einen neuen Knoten einfügen

Ein Knoten wird eingefügt, indem Sie den Baumzweig CAN auswählen und auf **[Neu]** klicken. Der Knotentyp wird vom Bus-Typ abgeleitet. Wenn der Bus ein CAN-Bus ist, ist der Knoten ein I/O Knoten, wenn der Bus ein S-CAN ist, ist der eingefügte Bustyp ein Servo-Typ. Im Dialogfenster zur Eingabe des Knoten muss man folgendes auswählen:

- **Position:** ist die Anzahl des Knotens (ab 1).
- **AutoOp:** Wenn ausgewählt, ermöglicht das Gerät das automatische Ümschalten in den Betriebszustand nach einer erneuten Verbindung.

Einen Knoten konfigurieren

Für jeden gesteuerten Knoten können die Übertragung-PDOs und die Empfang-PDOs definiert werden. Wenn der Bus ein CanOpen-Bus ist, können maximal 8 PDOs für TMSBus und TMSBus+ Platinen und maximal 4 für TMSCan und TMSCan+ Platinen definiert werden. Wählen Sie den Knoten in der Baumstruktur aus und klicken Sie auf **[Neu].** Die im Dialogfeld einzustellenden Daten sind:

- **PDO-Typ:** Wählen Sie entweder **Übertragung** oder **Empfang,** je nachdem, ob Sie ein TPDO oder ein RPDO definieren.
- **Größe**: Größe des empfangenden oder übertragenden PDOs.
- **COB-ID:** Wert, der nur auf TMSBus- und TMSBus + Platinen definiert werden kann. Der Wert wird in Dezimalzahlen angezeigt und gespeichert. Um den Wert in hexadezimaler Notation anzuzeigen, aktivieren Sie das Kontrollkästchen **Hexadezimal** neben dem Wert selbst.
- **Asynchron**: Wenn aktiviert, werden asynchrone PDOs konfiguriert, d.h. werden sie nicht bei jedem Zyklus aktualisiert. Diese Option wird nur auf TMSCan- und TMSCan + Platinen verwaltet. Der Empfang der asynchronen PDOs ist im GPL-Code mithilfe der Anweisung RECEIVEPDO auszuführen.

Eigenschaften der Verwaltung EtherCAT in Albatros

Der Kommunikationsmodus ist immer DC-Synchronous. Der erste Knoten im Netzwerk besorgt das Taktsignal, so ist es unerlässlich sicher zu sein, dass dieser Knoten ein präzises und festes Taktsignal bietet, wie es, zum Beispiel, von TRS-CAT TRS-CAT gegeben wird. Es ist nicht möglich, diverse Modi, wie, z.B. Free-Run, anzuwenden.
Die verwalteten Protokolle sind: CoE (CAN application protocol over EtherCAT) und EoE (Ethernet over EtherCAT).
Die Geräte werden nach den Spezifikationen DS401 und DS402 und auf der Grundlage vom Standard-Betriebsmodus der Achsensteuerung '*cyclic synchronous velocity mode*' innerhalb von CoE, verwaltet.

Die Höchstzahl der Knoten EtherCAT ist 200.

Vorwort

Mit jedem physikalischen EtherCAT Gerät ist eine ESI - Datei (EtherCAT Slave Information) verknüpft, die die Eigenschaften und die Funktionalitäten des Geräts beschreibt. Die Datei hat XML-Format. Jedes Gerät muss eine einzige ESI-Datei haben. Normalerweise, kann man die ESI-Dateien aus der Internetseite des Herstellers herunterladen. Albatros sucht diese Dateien im in Tpa.ini definierten Ordner,

in der Einheit [tpa] unter DirESIFiles. Die standardmäßige Einstellung ist der Unterordner "\EtherCAT" vo System.

Aus den ESI Albatros Dateien, gewinnt sie die Informationen über das Gerät durch die Analyse aller Elemente "/Devices/Device/Type". Jedes Gerät wird von einem Vendor ID, einem Product ID und einem Revision Number identifiziert.

Immer aus den ESI-Dateien werden die Informationen über die Erweiterungen (auch Module genannt) der Geräte gewonnen. Indem Albatros in der ESI-Dateien des Geräts die "Modules/Module" - Elemente sucht, findet es die Informationen über die Erweiterungstypen.

PDO beschreiben

Man kann max. 16 vom (TxPDO) Knoten geschickte PDOs und max. 16 vom (RxPDO) erhaltene Knoten definieren. Jedes RxPDO beschreibt ein einziges PDO das der Knoten vom Master empfingt, so analoge und digitale Ausgänge für E/A Knoten oder Zielgeschwindigkeit und controlword für die Knoten der Achse. Jedes TxPDO beschreibt ein einziges PDO das der Knoten vom Master bekommt, daher analoge und digitale Ausgänge für E/A Knoten oder Zielgeschwindigkeit und für die Knoten der Achse.

Für die Liste und die Beschreibung der PDOs und der auf einem PDO abbildbaren Objekte muss man sich auf die Dokumentation der spezifischen EtherCAT - Vorrichtung und auf seine ESI-Datei beziehen.

Drei sind die Modi, um die PDOs eines Knotens zu beschreiben:

- Kein PDO angeben. Auf dieser Weise verwendet die numerische Steuerung die in der Vorrichtung standardmäßig konfigurierten PDOs.
- Nur die PDOs bestimmen ohne die Liste der Objekte angeben.
  Zu verwenden, wenn ein CN über verschiedene, alternative und nicht programmierbare PDOsDa verfügt.
- Das PDO umfassend beschreiben, indem Sie das Kommunikationsobjekt und die Liste der abzubildenden Objekte angeben.
  Das ist der beste Modus zum Prüfen der Informationen, die von CN gesandt und empfangen sind.

Jedes Objekt ist von seinem Index im Object Dictionary vom Knoten beschrieben, optional gefolgt von einem Sub-Index. Ist der Sub-Index nicht vorhanden, wird es als 0 betrachtet.

Das Object Dictionary ist der Kern jedes Gerätes. Es ermöglicht den Zugriff auf alle Typen von Gerätedaten, auf die Kommunikationsparameter, auf die Konfigurationsparameter und Datenverarbeitung.

Achtung: Nicht alle Objekte von Object Dictionary können in einem PDO abgebildet werden.

EtherCAT-Hardware-Konfiguration

Die Hardware wird durch die Beschreibung der Master-Platinen und, für jede Platine, der Liste der physikalischen Geräten konfiguriert, die auf dem Bus mit jener Platine verbunden sind. Die physikalischen Geräte sind auch "Knoten" des Feldbusses genannt. Für EtherCAT ist die Master-Platine keine spezifische Steuerplatine des Busses; es wird ein Netzanschluss des Moduls verwendet. Für die lokalen Module muss das Netzanschluss eins der Anschlüssen sein, die von RTX verwaltet sind, während wird für die Fernmodule ein spezifisches Netzanschluss des Moduls verwendet unter denen, die von Windows 6.0 verwalten sind. Für jedes lokale Modul oder Fernmodul kann nur ein Master konfiguriert werden.

Die Master-Platine muss in die Hardware-Konfiguration eingesetzt werden, indem Sie den Zweig Modul auswählen und auf die Schaltfläche **[Neu]** klicken**.**

Der EtherCAT Knoten (CN)

Um einen Knoten einzusetzten, wählen Sie den Tab-Zweig der Baumstruktur aus und klicken Sie auf die Schaltfläche **[Neu]**. Die Indizes der Knoten müssen fortlaufend sein, sodass für jeden Befehl **[Neu]** ein Knoten zu den anderen hinzugefügt wird. Für jeden Knoten müssen folgende Daten definiert werden:

- **Name des Geräts**: Wählen Sie den Namen des Geräts aus den aufgelisteten Namen aus. Es werden nur Geräte mit der neuesten Version angezeigt. Wenn Sie alle Revisionen anzeigen möchten, müssen Sie die Option **Revisionen anzeigen** auswählen.
- **Achsenmodus**: Wählen Sie den Betriebstyp für die Antriebsknoten gemäß dem Standard DS402, Objekt 6060_{16} „Betriebsarten", aus. Man wählt zwischen **Zyklische Synchronposition** und **Zyklische Synchrongeschwindigkeit** aus. Letzteres ist die Standardeinstellung.
- **Zwang als I/O Knoten ausüben**: Wenn aktiviert, wird die numerische Steuerung gezwungen, die Aktivierung als I/O-Knoten zu betrachten. Dieses Attribut gilt nur für Knoten, die DS402 (Servoantriebe) unterstützen.

Der Achsenmodus ist die einzige Möglichkeit, die Daten nach dem Betreten des gesteuerten Knoten zu ändern.

Ein EtherCAT-Knoten kann gelöscht, verschoben, deaktiviert und kopiert werden. Die Befehle sind im Kontextmenü jedes Zweigs, im Hauptmenü von Albatros und in einigen Schaltflächen des Hardware-Konfigurationsfensters verfügbar.

Der Befehl **Löschen** löscht den ausgewählten Knoten, alle seine Erweiterungen und seine virtuellen physischen Links. Alle nachfolgenden Knoten werden an die vorherige Adresse verschoben.
Der Befehl **Verlegen** verschiebt den ausgewählten Knoten an eine neue Adresse. Knoten, die der ausgewählten Adresse folgen, werden ebenfalls um eine Position verschoben.
Der Befehl **Kopieren** speichert die Daten des ausgewählten Knotens zur Verwendung in einem nachfolgenden Befehl Einfügen.
Der Befehl **Einfügen** fügt den zuvor kopierten Knoten in den Gerätebaum ein. Der Knoten wird an der vom Benutzer gewählten Position eingefügt. Alle folgenden Knoten werden verschoben.

EtherCAT-Erweiterung
Um eine Erweiterung einzufügen, wählen Sie den Knoten, dem Sie die Erweiterung hinzufügen möchten, und klicken Sie auf die Schaltfläche **[Neu]**. Die Indizes der Erweiterung müssen aufeinanderfolgend sein, so dass jeder Befehl **[Neu]** eine Erweiterung am Ende der anderen hinzufügt. Erweiterungs-PDO-Objekte sind nicht editierbar.

PDO-Antrieb ändern

Hinsichtlich der CN-Servoantriebe Knoten gibt es ein PDO für jeden Achse, so dass das n-te TxPDO und das n-te RxPDO von CN auf den n-ten Antrieb von CN sich beziehen. Die ersten zwei Objekte jedes RxPDO und TxPDO besitzen vorab zugewiesene Bedeutung und Größe. In der folgenden Tabelle wird beschrieben, wie Sie die PDOs konfigurieren, die sich auf die Achsen derselben NC beziehen und im Achsenmodus der **Zyklischen Synchrongeschwindigkeit** gesteuert werden.

Antrieb	RxPDO		TxPDO	
	1° Objekt 16 Bits Controlword	2° Objekt 32 Bits Target velocity	1° Objekt 16 Bits Statusword	2° Objekt 32 Bits Actual position
Achse	6040_{16}	$60FF_{16}$	6041_{16}	6064_{16}
Achse	6840_{16}	$68FF_{16}$	6841_{16}	6864_{16}
n-te Achse	Addieren Sie 800_{16} zu jedem Objekt der vorhergehenden Achse.			

In der folgenden Tabelle wird beschrieben, wie Sie die PDOs konfigurieren, die sich auf die Achsen derselben NC beziehen und im Achsenmodus der **Zyklischen Synchroposition** gesteuert werden.

Antrieb	RxPDO		TxPDO	
	1° Objekt	2° Objekt	1° Objekt	2° Objekt

	16 Bits Controlword	32 Bits Target position	16 Bits Statusword	32 Bits Actual position
Achse	6040_{16}	$607A_{16}$	6041_{16}	6064_{16}
Achse	6840_{16}	$687A_{16}$	6841_{16}	6864_{16}
n-te Achse	Addieren Sie 800_{16} zu jedem Objekt der vorhergehenden Achse.			

Um die Objekte einer Achse zu ändern, wählen Sie die Achse in der Baumstruktur aus und klicken Sie auf **[Bearbeiten]**. Der Dialog zeigt den Index der Übertragung - PDOs, den Index des Empfang-PDO und die Liste der konfigurierten Objekte.
Im Einzelnen sind die Daten in dem Dialogfeld:

- **PDO-Übertragungsindex**: Das ist der Wert des PDOs für die Übertragung der Achse. Es ist kein bearbeitbarer Wert.
- **Objekte**: Liste der Objekte des Übertragung - PDO. Die ersten beiden Objekte werden entsprechend dem gewählten Achsenmodus festgelegt.
- **Objektliste**: Liste der Übertragung-PDO-Objekte, die man verwenden kann. Für jedes Objekt werden zusätzlich zum Index der Subindex und die Beschreibung definiert.
- **PDO-Empfangsindex**: Das ist der Wert des PDOs für den Empfang der Achse. Es ist kein bearbeitbarer Wert.
- **Objekte**: Liste der Objekte des Empfang - PDO. Die ersten beiden Objekte werden entsprechend dem gewählten Achsenmodus festgelegt.
- **Objektliste**: Liste der Empfang-PDO-Objekte, die man verwenden kann. Für jedes Objekt werden zusätzlich zum Index der Subindex und die Beschreibung definiert.

Durch die **Bild auf** und **Bild ab** Taste ändern Sie die Reihenfolge der Objekte im Fenster der **Objekte**.
Die Schaltfläche **[Hinzufügen]** erlaubt ein Objekt einzufügen, das sich nicht unter den Anwesenden in der Objektliste befindet.
Um den Wert eines Objekts zu ändern, wählen Sie das Objekt aus und doppelklicken Sie mit der Maus.
Die Syntax zum Definieren eines neuen Objekts lautet:

- **Objektnummer**: ist die Objektnummer im Hexadezimalformat.
- **Subindex**: ist die Nummer des Subindex. Es muss durch das Zeichen '.' (Punkt) von der Objektnummer getrennt werden. Das ist ein optionaler Wert, der, falls nicht definiert, als Standardwert 0 angenommen wird. Es ist im Dezimalformat.
- **L**: Größe des Objekts in Bits. Muss ein Vielfaches von 8 sein.

Nicht vorhandene Werte werden aus den Daten im Object Dictionary übernommen, oder mit Standardwerten zugewiesen.

Beispiel:
Komplettes Objekt auch mit Subindex und Länge: 1600.1L16
Objekt ohne Subindex: 1601L24
Einfaches Objekt: 1603

Dann können diese Werte vom GPL durch die Anweisung GETAXIS gelesen werden, auf die verwiesen wird. Es ist auch möglich, zusätzliche Objekte sowohl vom Eichungsfenster als auch vom Oszilloskop zu verfolgen.

Allgemein ist es unter GPL möglich, im Lese-/Schreibmodus auf bestimmte Objekte innerhalb eines PDOs durch die GETPDO- und SETPDO-Anweisungen, auf die wir uns beziehen, zuzuweisen.

Zusätzliche PDOs

Um weitere Übertragung- und Empfang-PDOs zu definieren, wählen Sie den Knoten in der Baumstruktur aus und klicken Sie auf **[Ändern]**.

Um ein Übertragung-PDO hinzuzufügen, klicken Sie auf **[Zusätzliche TPDOs]**, und um ein Empfang-PDO hinzuzufügen, klicken Sie auf **[Zusätzliche RPDOs]**.

Das Dialogfeld zum Konfigurieren eines zusätzlichen PDO ähnelt dem Dialogfeld zum Einfügen von Objekten.

In diesem Fenster können Sie den PDO-Index und die zu verbindenden Objekte auswählen. Wenn das PDO über obligatorische und nicht bearbeitbare Objekte verfügt, werden diese angezeigt und alle Bearbeitungsschaltflächen sind deaktiviert. Das neue PDO wird der Baumstruktur am Ende der Antriebe hinzugefügt. Um die Daten zu ändern, wählen Sie und klicken Sie auf **[Ändern]**. Um ein zusätzliches PDO zu löschen, wählen Sie das PDO in der Baumstruktur und dann **[Löschen]**.

Automatische Erfassung von EtherCAT-Knoten

Wenn der EtherCAT-Bus in der Hardware-Konfiguration vorhanden ist, können die angeschlossenen Knoten über den Befehl **Automatische Knotenerfassung**, der über das Kontextmenü aktiviert werden kann, aus dem Netzwerk erfasst werden. Damit der Befehl ausgeführt werden kann, müssen die Hardware-Konfigurationsdaten und die auf der Steuerung vorhandenen Daten abgeglichen werden, und die ESI-Dateien, die den Knoten im Netzwerk beschreiben, müssen in dem Ordner vorhanden sein, der in der Datei Tpa.ini im Abschnitt [tpa] unter DirESIFiles

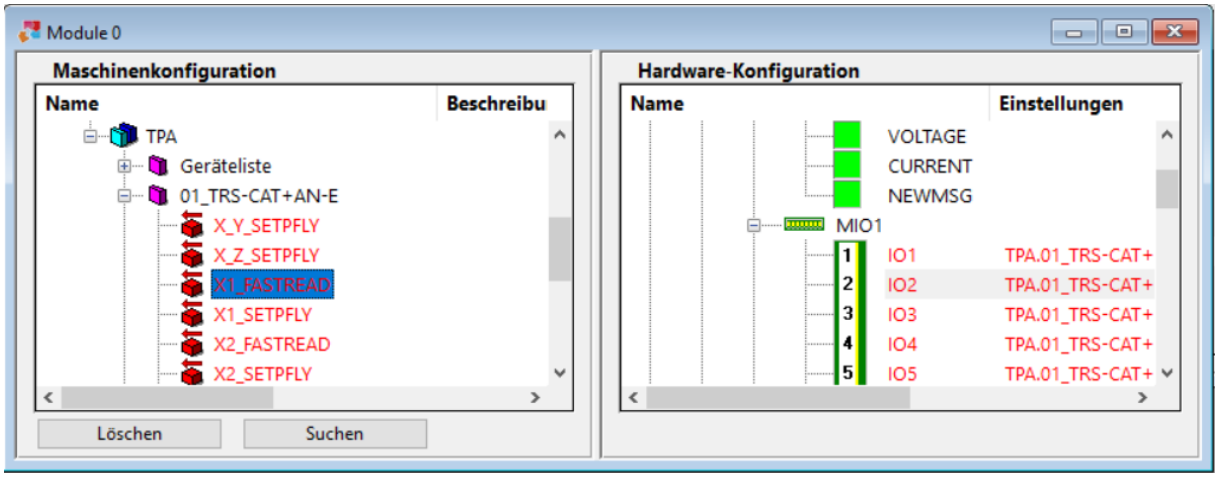
Die Erfassung der EtherCAT-Knoten aus dem Netzwerk folgt den aufgeführten Regeln:

- kein Knoten in der Hardware-Konfiguration konfiguriert ist: Daten, die sich auf Knoten beziehen, werden aus dem Netzwerk gelesen und in der Konfiguration angezeigt;
- EtherCAT-Knoten wurden bereits in die Hardware-Konfiguration eingefügt:
  - Wenn der vom Netzwerk gelesene Knoten die gleiche Adresse, die gleiche Hersteller-ID, die gleiche Produkt-ID und die gleiche Revisionnummer des konfigurierten Knotens hat, dann wird der konfigurierte Knoten mit allen seinen PDOs beibehalten;
  - Wenn der vom Netzwerk gelesene Knoten die gleiche Adresse wie der konfigurierte Knoten, aber eine andere Hersteller- oder Produktkennung hat, wird der Konfigurationsknoten zusammen mit seiner physischen virtuellen Adresse gelöscht und durch den Netzwerkknoten ersetzt. Wenn der Netzwerkknoten ein Laufwerk ist, werden die PDOs aus der ESI-Datei gelesen, andernfalls werden sie vom Netzwerkknoten gelesen.
  - Wenn der aus dem Netzwerk gelesene Knoten die gleiche Adresse, die gleiche Hersteller-ID, die gleiche Produkt-ID und eine andere Revisionsnummer des konfigurierten Knotens hat, wird nur die Revisionsnummer ersetzt und die PDOs bleiben erhalten.
  - Wenn der in der Konfiguration definierte Knoten keinen entsprechenden Knoten im EtherCAT-Netzwerk hat, der die gleiche Adresse belegt, wird er in der Konfiguration deaktiviert und der physikalische virtuelle Knoten bleibt erhalten.

7.4.3 Virtuell-physikalische Konfiguration

Die virtuell-physikalische Konfiguration ist die letzte Phase der Konfiguration und besteht in der Zuordnung der logischen Vorrichtungen zu Hardware-Komponenten.

Beim Öffnen der virtuell-physikalischen Konfiguration erscheinen zwei Fenster: links das der Maschinenkonfiguration (virtuell), rechts das der Hardwarekonfiguration (physikalisch). In beiden sind alle Elemente, aus denen das System besteht, in grafischer Form und als Baumstruktur dargestellt.



Virtuell-physikalische Konfiguration

Die bereits hergestellten virtuell-physikalischen Anschlüsse werden im Fenster "Maschinenkonfiguration" durch den Namen der Vorrichtung in rot und im Fenster "Hardwarekonfiguration" durch den Namen des Signaltyps, der der ebenfalls rot markierten Klemmennummer folgt, hervorgehoben.
Für jede Achse einer MECHATROLINK-II-Platine können 6 Eingänge und 1 digitalen Ausgang virtuell-physikalisch konfiguriert werden. Man wird auf die detaillierte Beschreibung im Kapitel **GPL Sprache->Instruktionen->MECHATROLINK-II->MECGETSTATUS** verwiesen.
Wenn in einem Modul EtherCAT-Bus vorhanden ist, können Sie immer noch Platinen für den MECHATROLINK-II-Bus, aber mit einigen Einschränkungen, konfigurieren: Liegt der Realtime-Wert bei 1 ms, können Sie nicht mehr als sechs MECHATROLINK-Achsen (je Bus) verbinden; liegt der Realtime-Wert bei 2 ms, steigt die Einschränkung auf 16 Achsen.

Die Farbe der noch zu verbindenden Vorrichtungen oder Klemmen ist schwarz. In Grau sind die Vorrichtungen dargestellt, die zu einer deaktivierten Gruppe gehören, deren Konfiguration Sie in physischer virtueller Form beibehalten wollten.

Vor den Achsen entsprechenden Signalen im Fenster "Hardwarekonfiguration" steht jeweils ein Rechteck in der gleichen Farbe des Kabelmantels innerhalb des Anschlusskabels.
Sie können einen Link hervorheben, indem Sie eine logische Vorrichtung (oder ein Hardware-Bestandteil) auswählen und die Leertaste betätigen; der Link erscheint als rote Linie, die das Gerät an Ihren Computerhardware anschließt. Außerdem können die Anschlüsse mit dem Befehl **"[Alt+Enter]"** andauernd sichtbar gemacht werden.
Um anzuzeigen, welche logische Vorrichtung an welche Hardware-Komponente angeschlossen ist, markieren Sie einen Hardware-Bestandteil und klicken Sie zweimal die Maus.
Zum Auswählen der logischen Vorrichtung und der anzuschließenden physikalischen Vorrichtung sind zwei verschiedene Verfahren möglich:

Erstes Verfahren

- Auf dem Bildschirm im Fenster "Hardwarekonfiguration" die physikalische Klemme sichtbar machen, an die die Vorrichtung anzuschließen ist.
- Die gewünschte virtuelle Vorrichtung im Fenster "Maschinenkonfiguration" markieren oder den Mauszeiger darauf richten

Zweites Verfahren

- Die gewünschte virtuelle Vorrichtung im Fenster "Maschinenkonfiguration" markieren oder den Mauszeiger darauf richten.
- Wählen Sie von Menü **Bearbeiten->Physikalisch geeignete Vorrichtung finden** oder die Tastenkombination **[Strg+Leertaste]**. Automatisch wird von Albatros im Fenster der "Hardware-Konfiguration" die erste freie physikalische Vorrichtung angezeigt, mit der die logische Vorrichtung verbunden werden kann.

Drittes mögliches Verfahren

- Eine virtuelle Vorrichtung im Fenster "Maschinenkonfiguration" markieren oder den Mauszeiger darauf richten
- Vom Menü **Bearbeiten->Nächste nicht verbundene Vorrichtung finden** oder die Tastenkombination **[Strg+NumPad+]** oder den Befehl **Bearbeiten->Die nächste nicht verbundene Vorrichtung finden** oder die Tastenkombination **[Strg+NumPad-]** auswählen.

Zum Anschließen der zwei ausgewählten Vorrichtungen:

- Auf der zu verbindenden logischen Vorrichtung die linke Taste der Maus drücken und bei gedrückter Taste die Maus in Richtung der Klemme ziehen. So wird eine rote Linie erscheinen, die auf die laufende Herstellung des Anschlusses weist. Nachdem die Zeile der Klemme errichtet worden ist, die Taste loslassen, um den Vorgang zu beenden oder
- Den Befehl **Verbinden!** vom Menü **Bearbeiten** auswählen oder die Schaltfläche **[Strg+L]** drücken.

Um einen Anschluss zu löschen, ist die jeweilige Vorrichtung oder Komponente zu markieren und die Schaltfläche **[Löschen]** oder die Drucktaste **[Entf]** zu drücken.

7.4.4 Verkabelungspläne

Nachdem die Anschlüsse zwischen virtuellen Vorrichtungen und zugehörigen physikalischen Vorrichtungen hergestellt worden sind, können Pläne oder Listen ausgedruckt werden, in denen die Zuordnung von physikalischen zu virtuellen Vorrichtungen dargestellt ist.

Zu diesem Zweck muss MS-Word (Version 6 oder höher) im System installiert sein, da Albatros dessen Funktionen zur Formatierung der Pläne ausnutzt.
Außerdem hat das System korrekt konfiguriert zu sein. Das bedeutet, dass im System die Modelldateien vorhanden sein müssen, die zur Erstellung der Pläne verwendet worden sind. Es handelt sich dabei um eine Serie von Dateien mit dem Zusatz ".doc", die sich normalerweise im Ordner System oder in einem anderen Ordner der Installation (normalerweise der Ordner "Pläne") befinden. Grundlegend ist, dass der Ordner, in dem sich diese Dateien befinden, mit dem in der Datei ***TPA.INI*** und dem Schlüssel "DirMaps" angegebenen Ordner übereinstimmt. Zum Beispiel:

```
[TPA]
DirMaps=C:\Albatros\Pläne
```

Um die Verkabelungspläne zu drucken, ist also eine beliebige Hardware-Komponente aus dem rechten Fenster der virtuell-physikalischen Konfiguration oder aus dem Fenster der Hardware-Konfiguration.

Drückt man auf die Schaltfläche "Druck" der Symbolleiste oder wählt man die Option ***Druck*** im Menü ***Datei*** erscheint das normale Auswahlfenster der Druckoptionen. Nach Einstellung der gewünschten Optionen und Bestätigung mittels der Schaltfläche **[OK]** erscheint ein Fenster mit der Liste der Hardware-Komponenten, die in der Konfiguration vorhanden sind.
In diesem Fenster können die Komponenten markiert werden, dessen Verkabelungsplan gedruckt werden soll.
Durch Druck auf die Schaltfläche **[OK]** werden die Verkabelungspläne schließlich gedruckt. An man die Option **Ausdruck auf Papier** werden die Pläne als Dokumente von MS-Word im Ordner Pläne+Name des laufenden Moduls (Mod.0, usw.) gespeichert.
Da u.U. eine große Anzahl an Seiten gedruckt werden können, empfiehlt es sich, erst einen Versuch anhand des Ausdrucks eines Plans einer einzigen Hardware-Komponente auszuführen, um sich zu vergewissern, dass alles nach Wunsch funktioniert. Wird anstelle der Pläne eine Liste von logischen Vorrichtungen gedruckt (z.B. eine Achsenplatine oder ein Fernmodul), ist im Fenster der Hardware eine Komponente nicht markiert worden.

7.5 Liste der Tastenkombinationen für die Navigation in einer Baumstruktur

Drucktaste	**Beschreibung**
Nach-oben-Taste Nach-unten-Taste	verschiebt die Auswahl in die vorherige oder in die nächste Zeile
Nach-rechts-Taste	erweitert um eine Ebene die ausgewählte Verzweigung und, falls sie bereits erweitert worden ist, verschiebt die Auswahl zur nachfolgenden Verzweigung
Nach-links-Taste	schließt die ausgewählte Verzweigung und falls sie bereits geschlossen worden ist, verschiebt die Auswahl in die vorherige Verzweigung
+	erweitert die ausgewählte Verzweigung um eine Ebene
-	schließt die ausgewählte Verzweigung
*	erweitern alle Niveaus der ausgewählten Verzweigung

8 Hilfsmittel zur Entwicklung

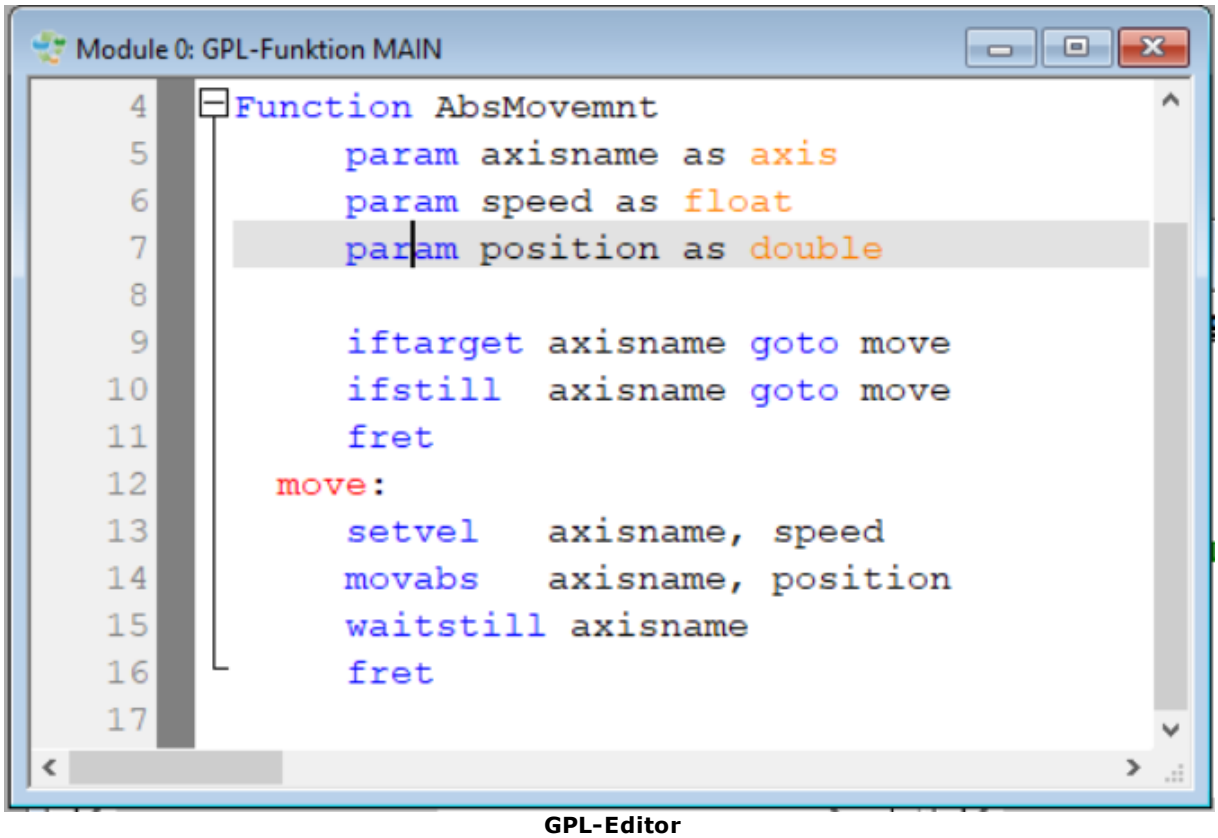
8.1 GPL-Editor

8.1.1 GPL-Editor Funktionen

Der GPL-Editor ist das Werkzeug zum Schaffen und Bearbeiten der Dateien mit dem GPL-Code von Albatros. Diese Funktion ist nur aktiv, wenn man über ein Passwort der Ebene "Hersteller" oder höher verfügt. Jeder Funktionsdatei sind Informationen zugeordnet, die mittels des Menüs ***Datei->Informationen*** angezeigt werden können.
Die Funktionsweise ist mit der eines typischen Editors vergleichbar, d.h. Befehle wie ***Kopieren, Einfügen, Suchen, Ersetzen***, usw. stehen zur Verfügung. All diese Befehle können von Menü ***Bearbeiten*** ausgewählt werden.

Rückgängig	Erlaubt, wenn möglich, den letzten Vorgang abzubrechen. Der vorhergehende Status wird wiederhergestellt.
Wiederholen	Wiederholt den letzten Status vor dem Befehl Abbrechen
Ausschneiden	Der Text oder die ausgewählten Daten werden gelöscht und in einen temporären Speicher kopiert, damit sie durch den Befehl *Einfügen* eventuell nochmals eingegeben werden können.
Kopieren	Der Text oder das ausgewählte Element wird in einen temporären Speicher kopiert. Diese können durch den Befehl *Einfügen* nochmals eingegeben werden.
Einfügen	Der Inhalt des temporären Speichers wird eingegeben mittels verschiedener Kriterien je nach der aktiven Funktion.
Löschen	Ein Text oder die Zeile oder das ausgewählte Element wird gelöscht. Was gelöscht worden ist kann wieder abgerufen werden, wenn sofort der Befehl *Rückgängig* ausgeführt wird.
Alles auswählen	Erlaubt den ganzen Text der aktiven Datei auszuwählen. Auf die ausgewählten Zeilen können die Befehle Kopieren, Ausschneiden, Einfügen angewandt werden.
Suchen...	Sucht einen Text im aktuellen Dokument. Die beim Suchen zu verwendenden Kriterien, wie die Suchrichtung und Unterschied zwischen Groß- und Kleinbuchstaben, die Richtung der Suche, die Unterscheidung zwischen Groß- und Kleinbuchstaben, die Suche nach ganzen Wörtern und Suche durch reguläre Ausdrücke können eingestellt werden.
Nächsten suchen	Erlaubt die vorhergehende Suche zu wiederholen und gibt die Möglichkeit, durch den Befehl Suchen die eingestellten Suchkriterien zu ändern.
Ersetzen	Erlaubt im aktuellen Dokument einen Text zu suchen und ihn durch einen anderen zu ersetzen.
Vorrichtung hinzufügen	Fügt eine Vorrichtung aus der Liste der Vorrichtungen ein. Diese Funktion ist besonders nützlich, wenn man mit zahlreichen Vorrichtungen arbeitet und sich an alle Namen erinnert. Es werden nur die Vorrichtungen des laufenden Moduls angezeigt, die aufgerufen werden können, sowie alle öffentlichen Vorrichtungen der anderen Module.
Funktion hinzufügen	Fügt eine Funktion oder einen Teil der Funktion ab der Cursorposition ein, die von der Prototypdatei, der vom Maschinenhersteller geschrieben ist, gelesen wird. Mehrere Prototypdateien können geschrieben werden. Die Prototypdatei ist eine Textdatei, dessen Name mit einem GPL-Präfix beginnen soll und dessen Erweiterung TXT sein muss. Diese Datei ist im Verzeichnis zu speichern, in dem die Bibliotheken (normalerweise system\lib) gespeichert sind. Werden mehrere Prototypdateien definiert, dann wird nach der Befehlauswahl ein Dialogbox geöffnet, in dem die Namen der Protypdateien ohne

	Präfix und ohne Erweiterung aufgelistet sind. Die Prototypdateien können zum Beispiel gebräuchliche Definitionen von const, allgemeine Funktionen, sowie Codes umfassen, die gebräuchliche Algorithmen implementieren, usw. Sie können auch Kommentare enthalten. Man kann eine Prototypdatei erstellen, wenn der ausgewählte Text in der Datei der GPL-Funktionen gespeichert wird. Der Befehl ist nur durch die Tastenkombination [Strg+Umschalt+C] erhältlich. Ein Dialogfenster öffnet, um den Namen einzugeben, zu dem der Codeauschnitt zuzuschreiben ist.
Nachricht hinzufügen	Fügt in den GPL-Text den mit der gewünschten Nachricht assoziierten numerischen Code ein. Erlaubt, in die Sprachdatei neue Nachrichten einzufügen.
Seitenwechsel aktivieren/deaktivieren	Fügt oder entfernt ein Seitenwechsel ein . Zum Springen zu definierten Positionen innerhalb der Funktionsdatei kann das Seitenwechsel als Textmarke gebraucht werden.
Zum nächsten Seitenwechsel gehen	Verschiebt den Cursor auf die Zeile des Seitenwechsels, das sich gleich vor seiner Stellung befindet
Zum vorherigen Seitenwechsel gehen	Verschiebt den Cursor auf die Zeile des Seitenwechsels, das sich gleich vor seiner Stellung befindet



GPL-Editor

Ob die Syntax korrekt ist, wird während der Archivierung geprüft, wenn der Text auch kompiliert wird. Der Programmierer kann trotzdem wenigstens eine Sichtprüfung ausführen, da der Text in verschiedenen Farben angezeigt wird, je nachdem, was er darstellt. Anweisungen sind z.B. blau, Kommentare grün, Etiketten rot, usw.
Die Werte der Tabulatoren für die Anfangsposition des GPL-Codes, die Anfangsposition des ersten Arguments der Anweisungen und die Anfangsposition des Kommentars können im Menü ***Optionen->Tabulator...*** geändert werden. Man kann zwei Arten von Registerkarten definieren:

- Absolute Tabulatorpositionen: Stellen die Anfangsposition für die Anweisungen des GPL-Codes, die Anfangsposition des ersten Argumentes der Instruktionen und die Anfangsposition für den Kommentar ein.
- Relative Tabulatorposition (Leerzeichen): Stellt ein, wie vielen Leerzeichen ein Tabstopp entspricht

Tabulatoren dienen dazu, den GPL-Code visuell begreiflicher zu machen.

Bei jeder Anweisung oder bei jedem Schlüsselwort steht eine Hilfe zur Verfügung, die das Abfassen der Funktion unterstützt. Um die Hilfe aufzurufen, einfach den Cursor auf die Anweisung setzen und die Taste **[F1]** drücken.

Pro Textzeile kann nur eine Anweisung geschrieben werden. Die Anweisung kann auf der folgenden Zeile weitergehen, indem man das Zeichen '_' (dem ein Leerzeichen vorangestellt ist) als das letzte in der Zeile eingegeben wird. Auf diese Weise können Kommentare in eine Anweisung eingefügt werden:

```
Message                                          _
  1000 ;Code der Mitteilung, die angezeigt wird  _
  3    ;Feld des synoptischen Schemas, in dem sie angezeigt wird [Enter].
```

Verwendung von regulären Ausdrücken

Reguläre Ausdrücke können im Fenster **Suchen** und im Fenster **Ersetzen** verwendet werden. Ein regulärer Ausdruck ist eine Folge von numerischen und alphanumerischen Zeichen, die zum Suchen und Ersetzen von Textteilen innerhalb eines größeren Textes verwendet wird. Albatros verwendet die Grammatik der regulären Ausdrücken von ECMAScript. Dieser Abschnitt beschreibt die Hauptverwendungen der regulären Ausdrücke in Albatros.

Der einfachste reguläre Ausdruck ist ein einzelnes Zeichen. Es gibt Ausnahmen, die durch die folgenden Zeichen dargestellt werden:

- **.(Punkt)**: der Punkt findet jedes beliebige Zeichen. Beispiel: "A..." findet alle Vorkommen des Großbuchstabens A, gefolgt von zwei beliebigen Zeichen.

- **[] (eckige Klammern)**: Mit eckigen Klammern können Sie eine Liste von Zeichen innerhalb dieser Klammern angeben. Der Text sucht nach Vorkommnissen für jedes der vorhandenen Zeichen. Wenn das erste Zeichen **^ (Caretzeichen)** ist, wird nach allen Zeichen gesucht, außer den in eckigen Klammern aufgeführten.

  Zum Beispiel:
  [<>]: Es wird nach allen Vorkommen des Zeichens < und des Zeichens > gesucht.
  [.]AX: Alle Vorkommen, die die Zeichenfolge "AX" enthalten, werden durchsucht.
  [a-d]: Alle Vorkommen der Zeichen a, b, c und d werden durchsucht. Der Bindestrich kennzeichnet eine Reihe von Zeichen.
  [\[\]]: Es wird nach allen Vorkommen des Zeichens [und Zeichen] gesucht.
  [^+]: Es wird nach allen Zeichen außer dem + Zeichen gesucht.

  ***(Sternchen)**: Alle Vorkommen eines Zeichens (oder von Gruppen von Zeichen), die vor dem Sternchen gefunden wurden, werden durchsucht. Das Sternchen betrifft nur das Zeichen davor: Damit es auf eine Gruppe von Zeichen wirkt, müssen Sie runde Klammern verwenden.
  Zum Beispiel:
  ;-*: Es wird nach allen Vorkommen der Zeichen gesucht **;** und **;-** und **;--------**.

- **+ (Pluszeichen)**: Sucht nach einem oder mehreren Vorkommen des davor stehenden Zeichens. Es unterscheidet sich von der Verwendung eines Sternchens, weil das Zeichen vor dem + Zeichen immer vorhanden sein muss. Das Plus-Zeichen wirkt sich nur auf das ihm vorangehende Zeichen aus: Damit er auf eine Gruppe von Zeichen wirkt, müssen runde Klammern verwendet werden. Nehmen wir das für das Sternchen verwendete Beispiel:

  ;-+:Es wird nach allen Vorkommen der Zeichen gesucht **;-** und **;---**. Das Zeichen allein wird nicht durchsucht ; **(Semikolon)**.

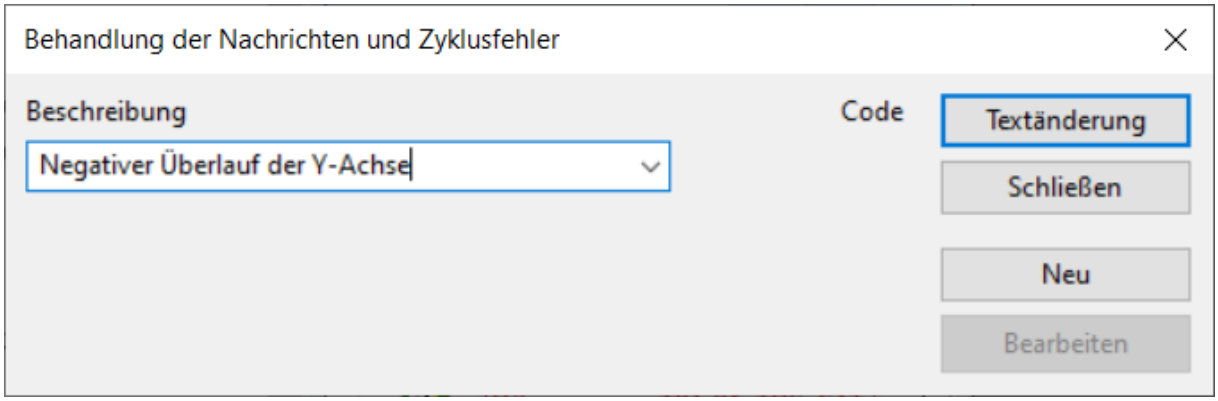
- **? (Fragezeichen)**: macht das Zeichen davor optional, das daher höchstens einmal vorhanden sein kann.
  Zum Beispiel:
  Setfeedi?: Alle Vorkommen des Wortes Setfeed und des Wortes Setfeedi werden gesucht.

- **{} (geschweifte Klammern)**: Geben an, wie oft ein Zeichen (oder eine Gruppe von Zeichen) innerhalb des Textes vorhanden sein muss.
- Zum Beispiel:

- ee{2}: Es wird nach allen Vorkommen von zwei aufeinanderfolgenden ee gesucht.
- **^. (Caretzeichen und Punkt)**: Findet das erste Zeichen jeder Zeile.
- **^ (Caretzeichen)**: Findet den gesuchten Begriff nur wenn er am Anfang einer Zeile steht.
- **| (Senkrechter Strich):** Findet die Begriffe, die sowohl vor als auch nach dem Zeichen '|' stehen.
  Zum Beispiel:
  Send|Const: Es wird nach allen Vorkommen des Wortes Send und des Wortes Const.
- **\ (Backslash)**: Der Backslash hat eine doppelte Funktion:
  1) macht das normale Zeichen zu einer Funktion
  \b findet den Anfangs- oder Endrand des Wortes
  \B findet das Wort außer dem Anfangrand
  \d findet nur die Zahlen
  \D findet alles außer den Zahlen
  \s findet den Raum
  \S findet alles außer dem Raum
  Beispiele:
  \bi findet alle Wörter im Text, die mit dem Buchstaben i beginnen .
  2) macht ein Sonderzeichen zu einem normalen Zeichen. Um beispielweise einen Text nach einem Sonderzeichen **^ (Caretzeichen)** zu durchsuchen, setzten Sie einfach den Backslash davor.

8.1.2 Nachricht hinzufügen

In Albatros gibt es zwei Arten von Nachrichten: die Modul- und die Gruppennachrichten. Der Befehl wird vom Menü ***Bearbeiten->Nachricht hinzufügen*** aktiviert.
Die Gruppennachrichten werden beim Abfassen des GPL-Codes mittels der Anweisung DEFMSG direkt in den Editor eingefügt. Sie sind nur innerhalb der Gruppe, in der sie definiert werden, sichtbar und verwendbar, so dass dieselbe Definition einer Nachricht in mehreren Gruppen vorkommen kann, ohne dass es zu Überschneidungen kommt.
Die Modulnachrichten können hingegen von allen Gruppen verwendet werden. Sie können mittels des Dialogfensters eingefügt werden, das den Aufruf einer in der Fremdsprachendatei bereits vorhandenen oder das Einfügen einer neuen Nachricht erlaubt.



Fenster zur Verwaltung von Nachrichten

Auf diese Weise braucht man nicht die Datei zu bearbeiten. Die Nachricht wird in der laufenden Sprache eingefügt, und man soll später in die anderen Sprachen (durch das Programm TpaLangs) sie übersetzen. Alle in der Fremdsprachendatei vorhandenen Nachrichten sind unter **Beschreibung** aufgelistet. Um eine Nachricht in die Funktion einzufügen, ist - nach Bestimmung des Texts - die Schaltfläche **[Textänderung]** zu wählen.
Um eine bestehende Nachricht zu ändern **[Bearbeiten]** oder eine neue zu schaffen **[Neu]**, ist zunächst die Änderung bzw. der neue Text einzugeben und dann die entsprechende Schaltfläche zu drücken.

8.1.3 Kryptographie

In Albatros ist es möglich, durch die Kryptographie, die Anzeige des Quell-Texts der Funktionen unmöglich zu machen.

Die Aktivierung der Kryptographie erfolgt mittels der Einführung in tpa.ini der Option Tele+=1. Der Standardwert liegt bei dem Wert 0, der die nicht aktivierte Kryptographie darstellt.

Wenn eine Datei von Funktionen gespeichert wird und die Kryptographie aktiviert ist, wird der folgende Nachricht angezeigt: "Möchten Sie die Datei chiffrieren?" Wenn Sie mit [Ja] antworten, wird die Datei verschlüsselt. Es ist möglich, eine Datei zu verschlüsseln, die vorher als unverschlüsselter Text gespeichert war, während eine verschlüsselte Datei immer so bleiben wird.

Wenn die Kryptographie aktiviert ist, und wenn das tägliche Passwort "Hersteller" ist, wird eine unverschlüsselte Datei nicht verschlüsselt werden.
Eine Datei von verschlüsselten Funktionen kann nur vom Benutzer, der vorher die Datei gespeichert hatte, in Albatros visualisiert oder geändert werden. Der Eigentümer einer Datei von verschlüsselten Funktionen kann nicht ändern!

Um die Kryptographie aus einem File zu entnehmen, muss ein externes Programm namens SBIANCA benutzt werden, das im Ordner Bin von Albatros ist.

Ein Verwendungsmodus des Programmes namens Sbianca zu verwenden, erfolgt durch die graphische Oberfläche. Wählen Sie die Dateien aus, die Sie entschlüsseln möchten. Für jede Datei erscheinen die Eigenschaften von **Zustand** und **Anmeldeinformationen**. **Zustand** kann **Nur-Text** sein, wenn die Datei unverschlüsselt ist oder **Verschlüsselt,** wenn die Datei verschlüsselt ist. Die Eigenschaft **Anmeldeinformationen** gibt Auskunft über die Sichtbarkeit der Datei: **Lesbar** bedeutet**,** dass die Datei ab der aktuellen Passwortebene angezeigt werden kann, **Gesperrt** bedeutet hingegen, dass die Datei nicht angezeigt werden kann.
Wenn Sie die Dateien ausgewählt haben, müssen Sie auf **[Entschlüsseln!]** klicken, um sie zu entschlüsseln oder auf **[Verschlüsseln!],** um sie zu verschlüsseln.

Eine zweite Verwendungsmöglichkeit erfolgt über die Befehlszeile. Befehle und Dateien werden als Argumente an das Programm übergeben. Wenn keine Argumente vorhanden sind, wird die Verwendungsart über die graphische Schnittstelle aktiviert. Die Syntax ist:
[-l|-e|-d] file1 file2 ...
Bei:

- -l, o keine Option, um den Zustand jeder Datei anzuzeigen
- -d um die Dateien zu entschlüsseln
- -e um die Dateien zu verschlüsseln

Am Ende der Ausführung gibt das Programm die Ergebnisse der angeforderten Operation aus.
Die Ausgabe des Programms erfolgt im Markdown-Format.

8.1.4 Liste der verfügbaren Tastaturkürzel

Text löschen

Drucktaste	**Beschreibung**
Rücktaste	löscht ein Zeichen, das sich links befindet, oder den markierten Text
Strg+Rücktaste	löscht das Wort, das sich links befindet,
Entf	löscht ein Zeichen, das sich rechts befindet, oder den markierten Text
Strg+T	löscht die Wörter oder die Leerzeichen, die sich rechts befinden
Strg+Entf	löscht das Wort, das sich rechts befindet und alle nachkommende Leerzeichen bis zum Anfang des neuen Wortes

Kommentar von mehreren Textzeilen

Drucktaste	**Beschreibung**
Strg+';'. In den Italienischen Tastaturen muss auch die Drucktaste [Shift] gedruckt werden.	entfernt oder fügt das Kommentarzeichen in die markierten Zeilen ein.

Cursorsteuerung

Drucktaste	**Beschreibung**
Nach-oben-Taste Nach-unten-Taste Nach-rechts-Taste Nach-links-Taste	Der Cursor wird in die gewünschte Richtung verschoben
Pos1	Der Cursor springt abwechselnd an den Anfangspunkt der Zeile und an das erste Kennzeichen der Zeile
Ende:	Der Cursor springt an das Ende der Zeile
Strg+Pos1	Der Cursor springt an den Anfang des Dokuments
Strg+Ende	Der Cursor springt an das Ende des Dokuments

Strg+Pfeil-nach-links	Der Cursor wird um ein Wort nach links verschoben
Strg+Pfeil-nach-rechts	Der Cursor wird um ein Wort nach rechts verschoben
Strg+Eingabe	Der Cursor wird an das erste Zeichen der nachkommenden Zeile verschoben

Auswählen

Drucktaste	**Beschreibung**
[Umschalt+Pos1]	erweitert die Markierung von der Position des Cursors bis zum Anfang der Zeile
Strg+Umschalt+Pos1	erweitert die Markierung von der Position des Cursors bis zum Anfang des Dokuments
Strg+Umschalt+Ende	erweitert die Markierung von der Position des Cursors bis zum Ende des Dokuments
Strg+Umschalt+Pfeil-nach-links	Die Wörter oder die Leerzeichen, die sich links des Cursors befinden, werden markiert
Strg+Umschalt+Pfeil-nach-rechts	Die Wörter oder die Leerzeichen, die sich rechts des Cursors befinden, werden markiert
Umschalt+Seite-nach-oben	Eine Seite wird nach oben von der derzeitigen Position des Cursors ausgewählt
Umschalt+Seite-nach-unten	Eine Seite wird nach unten von der derzeitigen Position des Cursors ausgewählt
Strg+W	Es wird das Wort markiert, auf dem der Cursor positioniert ist
Strg+A	Das ganze Dokument wird markiert

Rechteckige Auswahl

Drucktaste	**Beschreibung**
Alt+ Umschalt+Pfeil-nach-oben Umschalt+Pfeil-nach-unten Umschalt+Pfeil-nach-links Umschalt+Pfeil-nach-rechts	Ein rechteckiger Block des Codes wird markiert

Tabulierung

Drucktaste	**Beschreibung**
Tab	Wenn es kein markierter Test vorhanden ist, dann wird die Leerstelle eingesetzt, wie in ***Optionen->Tabulierung.*** Sind mehrere Zeile markiert, dann werden die für die dazugehörige Tabulierung eingestellten Leerzeichen eingefügt.
Umschalt+Tab	Wenn kein markierter Test vorhanden ist, dann wird der Cursor nach links der in ***Optionen->Tabulierung definierten Leerzeichen verschoben.*** Sind eine oder mehrere Zeilen markiert, dann werden sie nach links der für die relative Tabulierung eingestellen Leerstelle verschoben.

Kopieren und Einfügen

Drucktaste	**Beschreibung**
Strg+C Strg+Einfg	kopiert der markierten Text in die Zwischenablage
Strg+X [Shift+Entf	löscht den markierten Text, der in die Zwischenablage kopiert wird.
Strg+V Shift+Einfg	Von der Position des Cursors wird der Inhalt der Zwischenablage eingefügt
Strg+Y	Die Zeile, auf der der Cursor positioniert ist, wird gelöscht, und deren Inhalt in die Zwischenablage kopiert.
Drag'n'drop (mit der Maus)	verschiebt den markierten Text der, nachdem die Taste losgelassen worden ist, in die neue Position abgelegt wird.
Strg+Drag'n'drop (mit der Maus)	verschiebt den markierten Text der, nachdem die Taste losgelassen worden ist, in die neue Position kopiert wird.

Abbrechen/ Wiederherstellen

Drucktaste	Beschreibung
Strg+Z Alt+Rücktaste	annulliert die letzte Tastatureingabe
Strg+Umschalt+Z	stellt die letzte Tastatureingabe wieder

Suchen und ersetzen

Drucktaste	Beschreibung
Strg+F3	Im ganzen Dokument wird das Wort, auf dem der Cursor positioniert ist, nach unten gesucht
Strg+Umschalt+F3	Im ganzen Dokument wird das Wort, auf dem der Cursor positioniert ist, nach oben gesucht
F3	Sucht den nachfolgenden Treffer. Das Diagnostikfenster **Finden** muss geschlossen sein.
Umschalt+F3	Sucht den vorhergehenden Treffer. Das Diagnostikfenster **Finden** muss geschlossen sein.
Alt+F3	Das Diagnostikfenster **Finden** wird geöffnet und wird als zu suchende Text das Wort eingestellt, auf dem der Cursor positioniert ist.

Visualisierung der Kompilierungsfehler

Drucktaste	Beschreibung
Doppelklick auf dem Fehler	stellt den Cursor auf der Zeile der GPL- Funktion, in der der beschriebene Fehler aufgetreten ist.
F4	stellt den Cursor auf der Zeile der GPL- Funktion, in der der Fehler nach dem letztlich ausgewählten Fehler aufgetreten ist
Umschalttaste+F4	stellt den Cursor auf der Zeile der GPL- Funktion, in der der Fehler vor dem letztlich ausgewählten Fehler aufgetreten ist·

Eine Protypdatei erstellen

Drucktaste	Beschreibung
Strg+Umschalt+C	speichert den ausgewählten Text in der Funktionsdatei von GPL. Ein Dialogfenster öffnet, um den Namen einzugeben, zu dem der Codeauschnitt zuzuschreiben ist.

Faltung verwalten

Drucktaste	Beschreibung
Strg+M	offnet oder schließt die ausgewählte Faltung.

8.2 Bibliotheken

Eine Bibliothek besteht in einer Sammlung von GPL-Funktionen, die innerhalb des custom GPL-Codes aufgerufen werden können, ohne an eine besondere Konfiguration gebunden zu sein. Bibliotheken sind sehr nützlich, weil sie leicht von einer Maschine zur anderen zu kopieren sind. Auf diese Weise braucht man beim Implementieren neuer Maschinen gemeinsamen Code nicht erneut zu schreiben. Es kann z.B. eine Bibliothek geometrischer und mathematischer Funktionen geschaffen werden.

Die Dateien der Bibliothek werden im Ordner system\lib gespeichert. Sie werden kompiliert, wenn einer der folgenden Befehle gegeben wird: ***CNC->Initialisierung***, ***Datei->Alles kompilieren***, Bibliothek-Datei oder Datei der globalen Variablen ***Speichern*.**

Wird im GPL-Code einer Maschine der Name einer in einer Bibliothek bereits vorhandenen Funktion oder Variablen vergeben, so hat beim Kompilieren immer die Maschine den Vorrang. Kommt derselbe Name in zwei verschiedenen Bibliotheken vor, empfiehlt es sich zur einfacheren Erkennung, im GPL-Code folgende unabgekürzte Syntax zu verwenden: **BibliothekName.FunktionName**. Wenn z.B. die Funktion SegmentLänge bereits sowohl in der Bibliothek LIBGEO als auch in der Bibliothek LIBMAT vorhanden ist und man die Funktion aus der Bibliothek LIBGEO verwenden möchte, so ist folgendes zu schreiben: LIBGEO.SegmentLänge.

Alle Vorgänge, die eine Bibliothek betreffen, werden anhand eines Dialogfensters verwaltet. Mittels **[Neue]** kann eine neue Bibliothek geschaffen werden. Der zugewiesene Name erscheint im Verzeichnis der installierten Bibliotheken. Durch Aufruf im durch die Schaltfläche **[Importieren..]** geöffneten Dialogfenster können bereits vorhandene Bibliotheken importiert oder Gruppendateien in Bibliotheken umgewandelt werden. Die durch die Schaltfläche **[Löschen]** gelöschte Bibliotheken werden in den Papierkorb Windows verschoben.

Anhand der Schaltfläche **[Ändern]** kann der Code einer Bibliothek geändert werden. Auf diese Weise wird die Bibliothek vom GPL-Editor geöffnet. Beim Abfassen der Bibliotheksfunktionen sind folgende grundlegende Regeln zu beachten:

- es können keine Vorrichtungen, Funktionen, Variablen derjenigen Konfiguration aufgerufen werden, in der man die Funktion schreibt.
- es können öffentliche Funktionen und Variablen anderer Bibliotheken aufgerufen werden
- Bibliotheksfunktion werden per Default als privat definiert. Damit sie von anderen Funktionsdateien aufgerufen werden können, sind sie als PUBLIC zu erklären.

Die Möglichkeit, eine Bibliothek zu ändern, hängt von der Zugangsebene des Benutzers von Albatros ab. Die Zuweisung oder Änderung von Zugangsebenen erfolgt mittels der Schaltfläche **[Eigenschaften]**.

Die in einer Bibliothek enthaltenen globalen Variablen werden in der Diagnostik separat dargestellt. Die Anzeige der Elemente in einer Bibliothek hängt von der Zugangsebene des Benutzers von Albatros ab.

8.3 Debug

8.3.1 Debugger

Debugger ist eine Funktion von Albatros, die der schrittweisen Beobachtung der Anweisungen einer GPL-Task dient, wodurch etwaige Fehler in der Logik oder ungewünschtes Verhalten des Codes ermittelt und korrigiert werden können.

Diese Funktion ist nur bei Passwort Hersteller oder höher aktiv.
Mittels Debug kann man z.B.:

- Haltepunkte festlegen
- Die Ausführung einer Task unterbrechen und den Wert einer Variablen anzeigen
- Der Ausführungssequenz einer Funktion folgen
- Den von einer lokalen Variablen angenommenen Wert prüfen
- Im Fall einer If-Anweisung prüfen, dass der gewählte Zweig der richtige ist

Die im Debug-Modus verfügbaren Befehle können im Menü ***Debug*** gewählt werden. Die wichtigsten sind:

Weiter	nimmt die Ausführung eines blockierten Tasks wieder auf. Der Task wird bis zum Ende, oder bis zu einer neuen Blockierung oder bis zu einem Unterbrechungspunkt fort arbeiten.
Start	startet das Debug des ausgewählten Tasks.
Jetzt unterbrechen	stoppt die Ausführung des Tasks, der dem Debug unterzogen ist. Der Cursor wird auf der Zeile positioniert, in der die Instruktion unterbrochen wurde. Nach dem der Task gestoppt worden ist, kann seine Ausführung gesteuert und den Zustand der lokalen Variablen geprüft werden.
Einzelschritt	Führt eine einzelne GPL-Instruktion aus. Der Task muss vorher unterbrochen werden.
Ausführen bis Rücksprung	Führt jede Instruktion bis zur ersten Instruktion die nach der laufenden kommt aus.
Prozedurschritt	Führt eine einzelne GPL-Instruktion aus oder wenn die Instruktion ein Aufruf auf Funktion ist, führt die ganze Funktion aus.
Ausführen bis Cursor	Führt die Instruktionen bis zur Position des Cursors aus.
Debug beenden	Führt das Fehlersuchen zu Ende aus. Die Funktionsdatei der dem Debug unterzogen war wird im Editor-Modus geöffnet.

Zum Aufruf des Debugs ist die Liste der in Ausführung begriffenen Tasks (von Menü ***Debug->Task in Ausführung***) oder die Liste aller Tasks (von Menü ***Debug-> Alle Tasks***) anzuzeigen und der Task zu markieren, der dem Debug zu unterziehen ist.

Vor dem Debuggen ist es notwendig, dass keine Kompilierungsfehler der Funktionen (z.B. Syntax-Fehler, nicht angegebene Variablen) unterliegen und dass das das Modul, das Sie debuggen wollen korrekt initialisiert wurde.

Das Debug-Fenster ist dem des GPL-Editors ähnlich, allerdings ohne die Möglichkeit, den Code zu ändern. Der Hintergrund des Fensters ist grau und die in Ausführung begriffene Zeile ist gelb hervorgehoben.

Bemerkung: Sie können nicht mehrere Aufgaben, die zum selben Modul gehören, gleichzeitig debuggen.

8.3.2 Task in Ausführung begriffen

Der Befehl kann vom Menü ***Debug->Task in Ausführung*** ausgewählt werden. Zeigt die Liste der in Ausführung begriffenen Tasks, die einer Maschine oder einem Modul zugeordnet sind. Markiert man ein Task, kann er durch respektiven Druck auf die Schaltflächen **[Debug]** bzw. **[Ende]** dem Debug unterzogen oder beendet werden.

8.3.3 Alle Tasks

Es zeigt in einem Dialogfenster die Liste aller Tasks an, die im GPL-Code definiert worden sind. Sie sind in einer Baumstruktur grafisch dargestellt. Markiert man eine Funktion, wird die Datei geöffnet, in der sie definiert ist, und der Cursor befindet sich auf der ersten Anweisung. Auf diese Weise können noch Haltepunkte vor Beginn der Ausführung festgelegt werden.
Es ist möglich, der Debug beliebiger Tasks und Funktionen ohne Parameter auszuführen.

Nachfolgend wird die Bedeutung jedes Symbols im Ausführungsbaum der Tasks beschrieben. Ein besonderes Symbol steht für die rekursive Funktion, d.h. eine Funktion, die die Funktion aufruft, von der sie selbst aufgerufen wird.

Symbol	Beschreibung
	Task der Hauptfunktion der gruppenübergreifenden Gruppe
	Autorun - Task
	allgemeine Task
	Realtime - Task
	Gruppenfunktion
	Gruppenfunktion, die von Anweisungen wie ONINPUT, ONFLAG ausgeführt wird.
	Bibliotheksfunktion
	Bibliotheksfunktion, die von Anweisungen wie ONINPUT, ONFLAG ausgeführt wird.
	rekursive Funktion

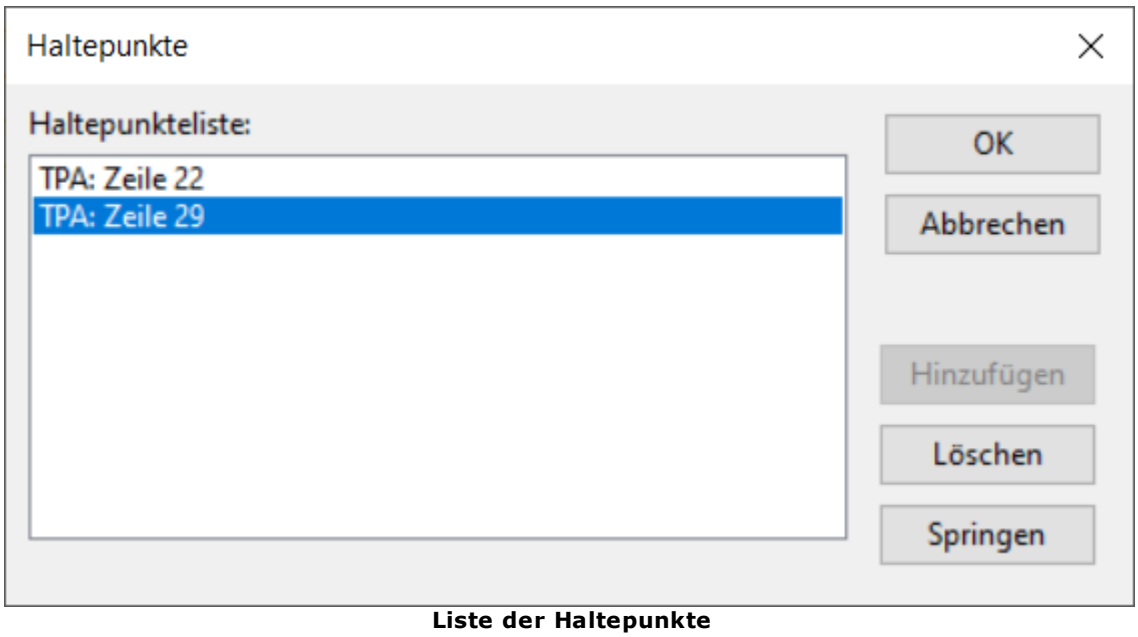
8.3.4 Aufrufeliste auf Funktionen

Während des Debugs kann die Liste der Funktionen angezeigt werden, die aufgerufen worden sind, aber bei denen die Anweisung FRET (Rückkehr) noch nicht ausgeführt worden ist. Es erscheint ein Dialogfenster mit den aufgerufenen Funktionen, die bis zur laufenden Anweisung geführt haben. An oberster Stelle befindet sich die zuletzt ausgeführte Funktion.

Um das Verhalten einer aufgerufenen Funktion zu beobachten:
- Setzen Sie den Cursor an die gewünschte Stelle innerhalb der Funktion.
- Wählen Sie ***Debug->Bis Cursor ausführen*** aus, um das Programm bis zur gewünschten Position auszuführen.
- Wählen Sie ***Debug->Aufrufeliste auf Funktionen*** oder die Tastenkombination Ctrl+K aus.
- Markieren Sie im Dialogfenster Aufrufe den Namen einer Funktion. Der Cursor springt zur ersten Anweisung der gewählten Funktion.

8.3.5 Haltepunkte

Ein Haltepunkt erlaubt eine detailliertere Beobachtung der Ausführungssequenz der Anweisungen, die Untersuchung und Änderung von Variablen und Vorrichtungen, die Untersuchung der Liste der aufgerufenen Funktionen, usw.
Die Ausführung eines Tasks hält beim Erreichen der Anweisung an, wo der Haltepunkt eingefügt worden ist.
Haltepunkte können sowohl vor als auch während der Ausführung eines bestimmten Tasks festgelegt werden (vom Menü ***Debug->Haltepunkte***). Haltepunkte, die überflüssig geworden sind, können gelöscht werden.



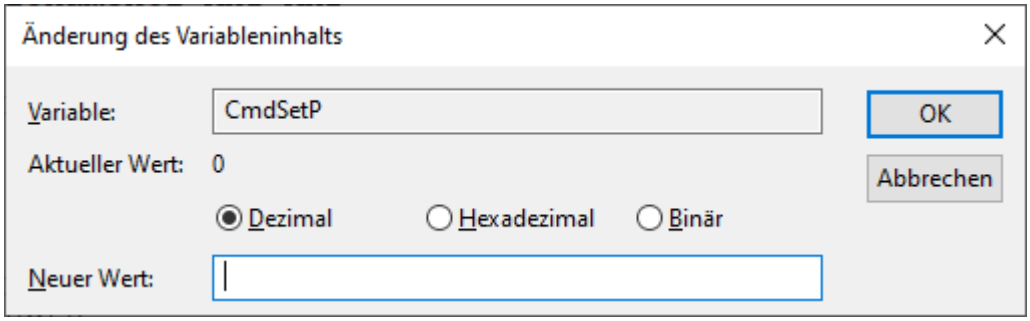
Liste der Haltepunkte

Es können sich Situationen ergeben, bei denen ein Task nicht anhält, weil die Ausführung den Haltepunkt nicht erreicht. In solchen Fällen kann der Task mittels des Befehls ***Debug->Jetzt unterbrechen*** angehalten werden. Der Cursor hält an der GPL-Anweisung, die bei der Unterbrechung im Begriff war, ausgeführt zu werden.

8.3.6 Variablenwert

Der Befehl kann vom Menü ***Debug->Variablenwert*** ausgewählt werden.
Nach der Unterbrechung einer Task ist es möglich, folgendes anzuzeigen:

- Wert der lokalen Variablen, die in der Funktion definiert sind, wo der Task unterbrochen worden ist
- Globale Variablen
- Den von einem Ausdruck angenommenen Wert
- Zustand der Vorrichtungen und der Parameter des Typs Vorrichtung.



Anzeige/Änderung des Inhalts einer Variable

Wenn die Variable (oder die Vorrichtung) nicht schreibgeschützt ist, kann deren Inhalt geändert werden: Die Änderung beeinflusst logischerweise die folgende Ausführung des Tasks.
Die Änderung des Werts einer Variablen oder einer Vorrichtung erlaubt das Ausprobieren der Ausführung unter anderen als den normalen Bedingungen, die Korrektur von eventuell auftretenden Fehlern sowie das Fortfahren mit der Ausführung der darauffolgenden Anweisungen.
Es ist möglich, den Inhalt einer Variablen, einer Vorrichtung oder einer Konstante ebenfalls durch die Verschiebung der Maus auf die Variable, auf den Namen der Vorrichtung oder auf die Konstante zu visualisieren. Ein Tooltip erscheint, in dem das Typ, den Namen und den Wert der Daten angezeigt werden. Wird ein Ausdruck ausgewählt, dann erscheint sein Ergebnis. Befindet sich der Cursor innerhalb einer Auswahl, dann wird die ganze Auswahl verwendet; anderenfalls nur das Wort in dem sich der Mauscursor befindet. Befindet sich der Cursor nicht innerhalb eines Wortes, dann wird das ganze Argument verwendet.
Z.B., um den Wert der Mx[3][Spalte]-Matrix zu veranschaulichen, wenn sich der Mauscursor auf "3" im Tooltip befindet, dann erscheint 3; befindet sich der Mauscursor auf "Spalte", dann erscheint der

Spaltenwert; befindet sich der Mauscursor auf "Matrix" dann erscheint nichts; befindet sich der Mauscursor auf einer eckigen Klammer, dann erscheint der Wert von Mx[3][Spalte].

8.3.7 Liste der verfügbaren Tastaturkürzel

Um die Befehle von **Debug zu** aktivieren, können die Optionen von Menü ***Debug*** oder direkt mittels der Tastatur ausgewählt werden.
Die zu verwendenden Tasten sind die folgenden:

Taste	**Bezeichnung**
Ctrl+F2	öffnet das Dialogfeld, in dem die Tasken in Ausführung aufgelistet sind
Ctrl+Shift+F5	öffnet das Dialogfeld, in dem alle Tasks aufgelistet sind
Ctrl+B	öffnet das Dialogfeld zum Einfügen oder zum Löschen der Haltepunkte
Ctrl+F9	fügt ein oder eliminiert den Anhaltspunkt in der Zeile, auf der der Cursor gesetzt ist
Ctrl+K	öffnet das Dialogfeld zur Anzeige der Liste jener Funktionen, die aufgerufen, aber noch nicht ausgeführt worden sind.
Shift+F9	öffnet ein Dialogfenster zur Anzeige des globalen Inhalts einer Variablen Wahl
F8	führt diese Anweisung aus. Ist diese eine Funktion, dann fügt sie sich in die Funktion
Shift+F7	führt jede Anweisung der Funtkion aus.
F10	führt diese Anweisung aus. Ist diese eine Funktion, dann führt sie diese Funktion aus, ohne sie darin einzufügen
F7	führt jede Anweisung bis zur Anweisung aus, auf der der Cursor gesetzt ist. Der Cursor soll auf einer Anweisung innerhalb der Funktion gesetzt sein.
Alt+Interr	bricht die Ausführung des Codes der letztlich ausgeführten Anweisung ab
F5	nimmt die Ausführung des Codes nach einer Unterbrechung wieder auf
Shift+F5	bringt den derzeitigen Task zum Schluss und führt er ihn nochmal aus
Alt+F5	bringt die Fehlersuche zum Schluss

8.4 Initialisierung der Steuerung

8.4.1 Netzanschluße

Der Funktionsablauf von Albatros in der Maschine ist von einem USB - Hardwareschlüssel geschützt, der von TPA konfiguriert wird.
Der Befehl kann vom Menü ***CNC>->Netzanschlüsse*** ausgewählt werden. Zeigt den Zustand der an das System angeschlossenen Fernmodule an. Wenn ein Modul nicht angeschlossen ist, erscheint das entsprechende Symbol mit einem roten X-Zeichen darüber.
Pro Modul gibt es drei Felder. Im ersten Feld steht der Name des Moduls, im zweiten Feld steht der Name des Netzplatzes. Normalerweise lauten bei letzterem die ersten Zeichen des Namens "TPA" und sind von der Seriennummer des Fernmodules gefolgt.

Zuordnung eines Netzknotens (Fernmodules) zu einem logischen Modul
Einem Modul kann durch Markierung der Angabe "Nicht konfiguriert " oder durch die Auswahl der Schaltfläche "**[Bearbeiten]**" mit dem Mauszeiger ein Netzknoten zugeordnet werden. Nach wenigen Sekunden erscheint ein Fenster mit einer Liste der im Netz verfügbaren Netzmodulen (sofern jedes Fernmodul eingeschaltet ist und über eine korrekte IP-Adresse verfügt).

Nun kann ausgewählt werden, welcher Netzknoten an das logische Modul angeschlossen werden soll. Die Wahl ist anhand der Schaltfläche  zu bestätigen.

Dieser Vorgang bedarf nur der Zugangsebene "Wartung", da es nicht notwendig ist, die Systemkonfiguration von Albatros aufzurufen, die hingegen der Zugangsebene "Hersteller" bedarf.

Software-Aktualisierung eines Fernmoduls

Sie können die im internen Speicher des Fernmoduls vorhandene Steuerungssoftware aktualisieren, wenn sie auf die Schaltfläche **[Aktualisieren]** klicken. Damit Albatros die Software aktualisiert, muss das zu aktualisierende Fernmodul ausgewählt werden, und mit Albatros verbunden und kommunizieren.

8.4.2 Hardware-Diagnostik

Der Befehl kann vom Menü ***Cnc->Hardware-Diagnostik*** ausgewählt werden. Die Hardware-Diagnostik zeigt die Liste und den Zustand der konfigurierten Module, der Platinen und der zugehörigen Knoten an, so wie sie in der Hardware-Konfiguration definiert worden sind. Befindet sich über dem Symbol einer Platine oder eines Knotens ein rotes X-Zeichen, bedeutet es, dass sie nicht unter der Hardware der Steuerung ermittelt werden konnten, oder dass es nicht möglich war, sie korrekt zu initialisieren. Ist sie mit einem gelben Fragezeichen versehen, bedeutet es, dass das System eine Platine oder einen Knoten ermittelt hat, aber dass der Typ nicht dem in der Konfiguration definierten Typ entspricht.

EtherCAT Netzwerktopologie

Im Hardware-Diagnosefenster wird bei Auswahl eines EtherCAT-Netzwerkknotens im Baum die Schaltfläche **[Details]** aktiviert, was wiederum die grafische Darstellung der EtherCAT-Netzwerktopologie ermöglicht.

Dieses Diagramm zeigt die Informationen der physisch im Netzwerk vorhandenen Knoten an: Den Status jedes Knotens, den Status der Achsen, wenn der Knoten ein Servoantrieb ist, und die Qualität der Kommunikation. Jeder Knoten und jede Achse wird als ein Rechteck dargestellt, dessen Farbe den Status definiert.

Wenn Sie den Mauszeiger über das Rechteck bewegen, erscheint ein Tooltip, der den Zustand des Knotens und Kommunikationsfehler oder den Zustand der Achse im Textformat beschreibt.

Anzeigen und Bearbeiten von Objekten in Knoten

Wenn im Hardware-Diagnosefenster ein EtherCAT-Netzwerkknoten im *Baum* ausgewählt ist, wird die Schaltfläche **[Object Dictionary]** zur Anzeige und Bearbeitung von Knotenobjekten aktiviert. Die Datenbearbeitung ist nur auf der **Hersteller**ebene möglich.

Die Objekte sind nach ihrer Adresse gruppiert, wie folgt:

Anfangsadresse	Endadresse	Bereichsname
0x0000	0x0FFF	Data type area
0x1000	0x1FFF	Communication area
0x2000	0x5FFF	Manufacture specific area
0x6000	0x6FFF	Input area
0x7000	0x7FFF	Output area
0x8000	0x8FFF	Configuration area
0x9000	0x9FFF	Information area
0xA000	0xAFFF	Diagnosis area

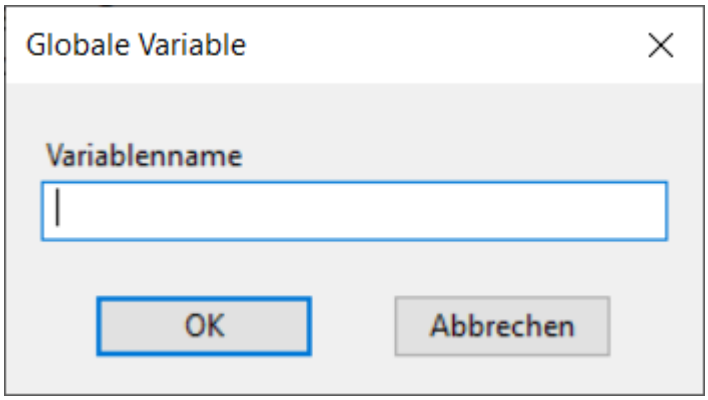
0xB000	0xBFFF	Service transfer area
0xC000	0xEFFF	Reserved area

Die ganzzahligen numerischen Werte werden sowohl in dezimaler als auch in hexadezimaler Basis angezeigt. Wenn ihr Wert geändert wird, kann die Zahl auch hexadezimal in der Schreibweise $...H und binär mit $...B eingegeben werden (wie bei GPL).

8.5 Test

8.5.1 Globale Variable speichern

Der Befehl kann vom Menü ***Test->Globale Variable speichern*** ausgewählt werden.
Speichert den Inhalt einer globalen Variablen auf Festplatte als formatierte Textdatei. Der Name der Datei lautet *Variablenname*.txt und die Speicherung erfolgt im Ordner *Report*. Der Vorgang ist nur möglich, wenn die Zugangsebene zum Lesen der globalen Variablen mit der laufenden Zugangsebene übereinstimmt.



Speichern einer globalen Variablen

8.5.2 Funktion ausführen

Der Befehl kann vom Menü ***Test-> Funktion ausführen*** ausgewählt werden.
Führt eine Funktion unabhängig vom Rest des Systems aus. Es wird also ein Task geschaffen, dessen Ausführung ab der gewählten Funktion beginnt und die auch deren Namen übernimmt. Es können nur diejenigen Funktionen ausgeführt werden, die keine Eingangsparameter haben und deren Zugangsebene zum Lesen mit der laufenden Zugangsebene übereinstimmt. Wenn die auszuführende Funktion die Hauptfunktion der gruppenübergreifenden Gruppe ist, werden anschließend auch alle Autorun-Tasks ausgeführt.

8.5.3 Nachrichten importieren

Die über die GPL-DEFMSG-Anweisung zugewiesenen Nachrichten der Gruppe werden in einer XMLNG-Datei gespeichert. Die Nachrichten können in der Datei bearbeitet worden. Um die Veränderungen im GPL-Code anzuzeigen, muss man die Anweisung ***Test->Nachrichten importieren*** benutzen.
Um einen ganzen GPL-Code importieren zu können, muss dieser fehlerfrei kompiliert sein. Sollte dies nicht der Fall sein, erscheint dem Benutzer die Nachricht "Der GPL-Code ist nicht vollständig kompiliert". Keine Nachrichten der Gruppe, die chiffrierten Dateien angehören, werden importiert (Siehe Kapitel ***Hilfsmittel zur Entwicklung->GPL-Editor->Kryptographie***), daher ist dem Benutzer keine Klartextveranschaulichung gestattet.

Es werden nur die Nachrichten importiert, die schon im GPL.Code definiert wurden. Die Änderung des GPL-Textes wird nicht durchgeführt, wenn mindestens eine nachfolgende DEFMSG mit einer IFDEF-Anweisung vorhanden ist.

Während des Importierens können Fehler auftreten, wenn
- unten den Texten einer speziellen Nachricht der Gruppe das Zeichen einer Sprache mehrmals vorhanden ist;
- irgendeiner Text leer ist (bzw.: "");
- der Name einer Gruppe oder einer Bibliothek mehrmals definiert ist.

Nach erfolgtem Importieren werden sämtliche Module, in denen Gruppen oder Bibliotheken geändert wurden, kompiliert.

Alle die unten aufgeführtene Fremdsprache können in den XMLNG-Dateien verwendet werden.
"AFK" Afrikaans
"ARA" Arabisch
"AZE" Aserbaidschanisch
"BAS" Baschkirisch
"BEL" Belarussisch
"BGR" Bulgarisch
"BSB" Bosnisch (lateinisches Alphabet)
"BSC" Bosnisch (kyrillisches Alphabet)
"BRE" Bretonisch
"CAT" Katalanisch
"CHS" Chinesisch (vereinfacht)
"CHT" Chinesisch (traditionell)
"COS" Korsisch
"CSY" Tschechisch
"CYM" Gällisch
"DAN" Dänisch
"DEA" Deutsch (Österreich)
"DEU" Deutsch (Deutschland)
"ELL" Griechisch
"ENG" Englisch
"ENU" Englisch (Vereinigte Staaten)
"ESP" Spanisch
"ETI" Estnisch
"EUQ" Baskisch
"FAR" Persisch
"FIN" Finnisch
"FRA" Französisch
"FPO" Filipino
"FRB" Französisch (Belgien)
"FYN" Friesisch
"GLC" Galizisch
"HAU" Hausa
"HEB" Hebräisch
"HRB" Kroatisch (Bosnien und Herzegowina)
"HRV" Kroatisch (Kroatien)
"HUN" Ungarisch
"IBO" Igbo
"IND" Indonesisch
"IRE" Irisch
"ISL" Isländisch
"ITA" Italienisch
"JPN" Japanisch
"KAL" Grönländisch
"KOR" Koreanisch
"SAH" Jakutisch
"KYR" Kirgisisch
"LVI" Lettisch
"LTH" Litauisch
"LBX" Luxemburghisch
"MNN" Mongolisch
"NON" Norwegisch Nynorsk
"NOR" Norwegisch Bokmål
"NLB" Niederländisch (Belgien)
"NLD" Niederländisch (Niederlande)
"OCI" Okzitanisch
"PLK" Polnisch
"PTB" Portugiesisch (Brasilien)
"PTG" Portugiesisch (Portugal)
"RMC" Rätoromanisch

"ROM" Rumänisch
"RUS" Russisch
"SKY" Slowakisch
"SLV" Slowenisch
"SQI" Albanisch
"SRM" Serbisch (Lateinisches Alphabet, Serbien)
"SRN" Serbien (kyrillisches Alphabet, Bosnien und Herzegowina)
"SRO" Serbien (kyrillisches Alphabet, Serbien)
"SRP" Serbien (lateinisches Alphabet, Montenegro)
"SRQ" Serbien (kyrillisches Alphabet, Montenegro)
"SRS" Serbien (lateinische Alphabet, Bosnien und Herzegowina)
"SVE" Swedisch
"TAJ" Tadschikisch
"TRK" Turkisch
"TTT" Tatarisch
"TUK" Turkmenisch
"UKR" Ukrainisch
"URD" Urdu
"UZB" Usbekisch
"VIT" Vietnamesisch
"WOL" Wolof
"XHO" Xhosa
"YOR" Yoruba
"ZUL" Zulu

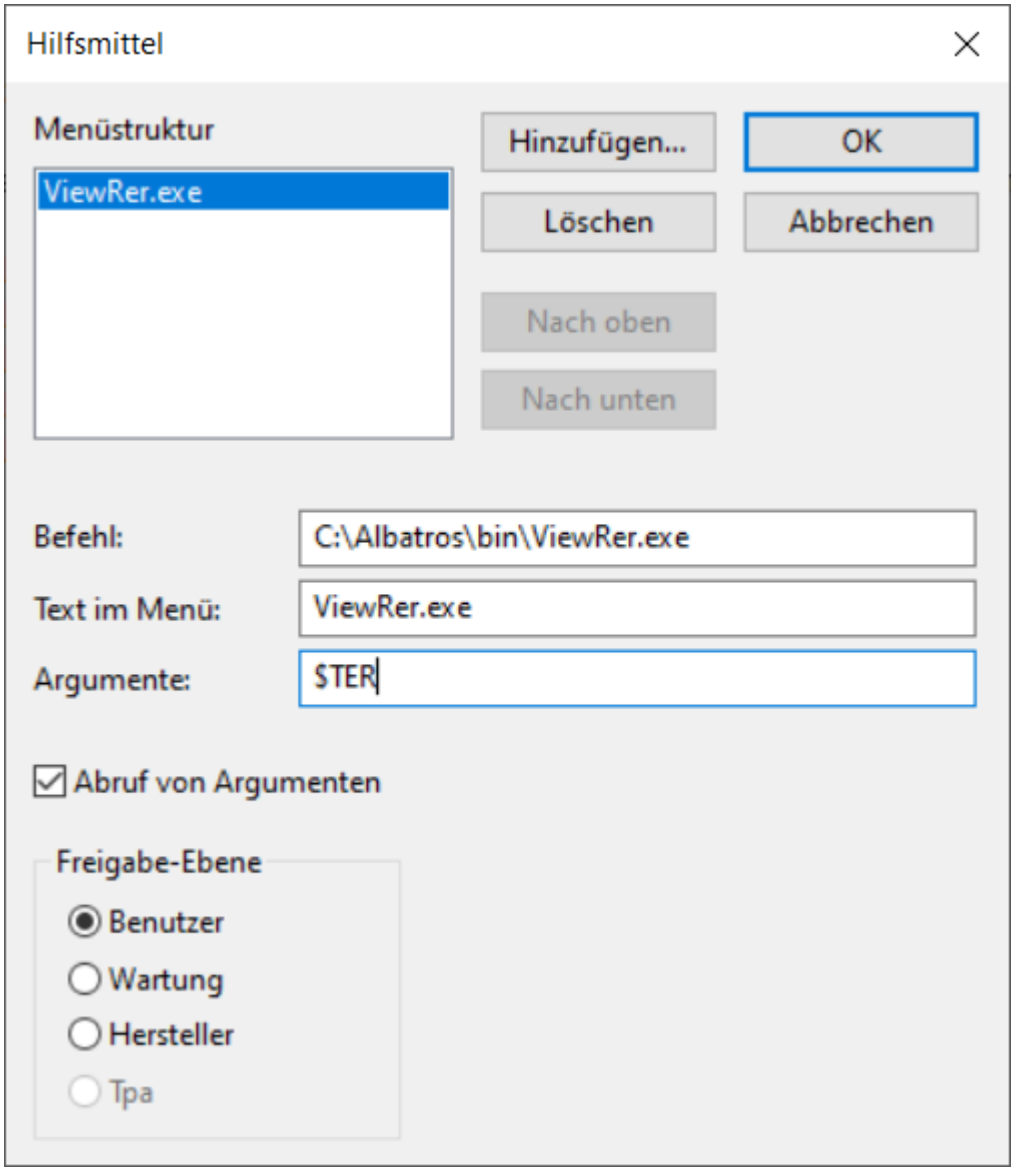
8.5.4 Benutzerhinweis in der Datei des Alarmberichtes

Mit dem Befehl **Bemerkung hinzufügen** aus dem Menü **Test** können Sie einen Bericht in die Alarmberichtsdatei für den aktuellen Monat (MONTHxx.TER) eintragen. Das Menüelement ist mit Passwortebene gleich oder höher als Assistenz freigegeben.

8.6 Hilfsmittel

8.6.1 Einstellungen

Der Befehl kann vom Menü ***Hilfsmittel->Einstellungen*** ausgewählt werden.
Erlaubt die Einstellung von höchstens 20 Programmen, deren Ausführung aus dem Menü ***Hilfsmittel*** von Albatros gestartet werden kann.



Konfiguration des Menüs Hilfsmittel

Menüstruktur: Die im Menü ***Hilfsmittel*** angezeigten Programme werden aufgelistet.

Befehl: Name des auszuführenden Programms. Als Option kann auch der Name des Ordners definiert werden, in dem das Programm gespeichert ist, sofern es sich weder um den Ordner handelt, aus dem Albatros ausgeführt wird, noch es sich um die Ordner handelt, in denen das Betriebssystem die ausführbaren Dateien (Variable von PFAD-Windows-Umgebung) sucht.

Text im Menü: Es handelt sich um den Namen, der im Menü ***Hilfsmittel*** zur Kennzeichnung des Ausführungsprogramms erscheinen soll.

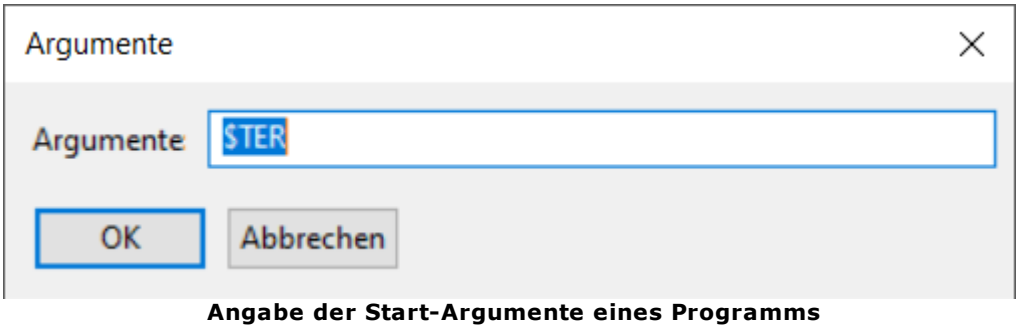
Argumente: Sämtliche Argumentkombinationen von Befehlszeilen, die zur korrekten Ausführung des Programms notwendig sind. Dynamische Argumente können eingesetzt werden. Zum Beispiel, wenn der String $TER bei der Ausführung von ViewRER verwendet wird, öffnet sich die Reportdatei des laufenden Monats.

Nachfolgend die Argumentliste:

$File	vollständiger Pfadname der laufenden Datei.
$FileName	Dateiname und Erweiterung der laufenden Datei.
$FileDir	Festplatte und Ordner der laufenden Datei.
$Ter	Vollständiger Pfadname der Fehlerberichtsdatei des laufenden Monats.
$DirModule	Festplatte mit Order enthaltend MODx der laufenden Datei.
$Module	Modulnummer der laufenden Datei.
$Bin	Festplatte mit Datei enthaltend die Programme von Albatros.

$TpaIni	Vollständiger Pfadname der Initialisierungsdatei TPA.INI.
$ReqDirModule	Pfad (Platte und Ordner) des Moduls Albatros. Wenn mehrere Module konfiguriert sind, dann erscheint das Dialogfeld des Moduls.
$ReqModule	Modulnummer Albatros. Wenn mehrere Module konfiguriert sind, dann erscheint das Auswahlfeld der Modulnummer.
$ReqFile	Name der Datei. Ein Fenster zur Anforderung des Dateinamens wird angezeigt. Die ausgewählte Datei wird unter den Argumenten des Programms das ausgeführt wird.

Abruf von Argumenten: Wenn aktiviert, erscheint bei jedem Befehl zur Ausführung des Programms ein Dialogfenster zur Eingabe von anderen Argumenten als diejenigen, die unter "Argumente" angegeben sind, und die je nach Ausführungsart des Programms unterschiedlich sein können.



Angabe der Start-Argumente eines Programms

Freigabe-Ebene: Legt die Anzeige-Ebene des Programms im Menü ***Hilfsmittel*** fest. Testprogrammen oder Programmen zur Änderung von Daten von Albatros wird normalerweise die Ebene "Hersteller" zugewiesen. Programmen zum Editieren von Bearbeitungen der Maschine wird die Ebene "Benutzer" zugeordnet.

Die Bearbeitung einiger Felder kann auch mittels der Schaltfläche **[Hinzufügen]** erfolgen. Diese öffnet das Dialogfenster **Hilfsmittel hinzufügen** zur Wahl des auszuführenden Programms. Zulässige Dateitypen sind: .EXE, .COM, .PIF, .BAT.
Schließt man das Dialogfenster durch Bestätigung der Daten, wird das Programm in das Fenster **Menüstruktur** eingefügt, der Name des Programms und der Ordner, der es enthält, in die Zeile **Befehl**.
Die anderen Schaltflächen lauten **[Löschen], [Nach oben], [Nach unten]** und dienen respektive dem Löschen eines Programms und dem Ordnen der Programmliste.

8.7 Browser

8.7.1 Browser

Der Browser ist eine Funktion von Albatros, die die vom Kompilierer hervorgebrachten Informationen verwendet, um eine Datenbank zur Schnellsuche der Symbole zu schaffen, die in den Funktionen definiert sind.
Diese Funktion ist nur aktiv, wenn man über ein Passwort der Ebene "Hersteller" oder höher verfügt. Die verfügbaren Befehle können aus dem Menü ***Debug*** gewählt werden.
Anhand des Browsers ist es möglich:

- den Cursor auf die Zeile zu setzen, wo eine Funktion, eine Variable oder eine Konstante eines Moduls, einer Gruppe oder einer Bibliothek zum ersten Mal definiert ist (vom Menü ***Debug->zur Definition springen***);
- den Cursor auf die Zeilen zu setzen, wo eine Funktion, eine Vorrichtung, eine Variable eines Moduls oder einer Gruppe, eine GPL-Anweisung (mit Ausnahme der Anweisungen FCALL und FRET) ihren Bezug haben (Vom Menü ***Debug->zur Referenz springen***, um die vorherige oder die nachkommende Referenz anzuzeigen, sind folgende Elemente beziehungsweise ***Debug->Nächster*** oder ***Debug->Voriger*** auszuwählen.)

Gruppenvariablen können nur im Editierfenster der Gruppe, der sie angehören, verwaltet werden.
Um beim Übergang zu einer neuen Version den Browser zu aktualisieren, empfiehlt es sich, zuerst die globalen Variablen zu speichern und dann den Befehl ***Datei->Alles kompilieren*** auszuführen.
Während des Editierens der Funktionen geht die Übereinstimmung zwischen Text und gesuchten Symbolen verloren, wird aber beim Speichern wiederhergestellt.

8.7.2 Identifizierer suchen...

Der Befehl kann vom Menü ***Debug->Identifizierer suchen...*** ausgewählt werden**.** Öffnet ein Dialogfenster zur Eingabe des Symbolnamens, der im GPL-Code gesucht werden soll. Je nach vorgegebener **Art der Suche** wird die Definition oder der erste Bezug des Symbols ermittelt.

Der eingegebene Name kann folgende Eigenschaften besitzen:

- Enthält kein Zeichen '.' (Punkt): wird in allen Funktionsdateien gesucht
- Enthält nur ein Zeichen '.' (Punkt): der Name vor dem Punkt wird als Gruppenname angesehen und das Symbol wird nur in der Gruppe gesucht. Sind z.B. zwei Funktionen mit dem Namen VisError definiert, davon eine in der Gruppe MAIN und die andere in der Gruppe ACHSEN, und schreibt man als Identifizierer ACHSEN.VisError, so wird der Cursor auf die erste Zeile der Funktion VisError in der Gruppe ACHSEN gesetzt.
- Enthält zwei Zeichen '.' (Punkt): der Name vor dem ersten Punkt wird als Gruppenname, der Name vor dem zweiten Punkt wird als Untergruppenname angesehen und das Symbol wird nur in der Untergruppe gesucht.
- Endet mit dem Zeichen '*' (Stern): Es werden alle Symbole gesucht, die mit den Zeichen vor dem Stern beginnen.

Kommt es bei der Suche nach dem Symbol zur Mehrdeutigkeit, erscheint ein Dialogfenster, in dem alle Symbole mit dem vorgegebenen Namen angezeigt werden, unter denen das gewünschte Symbol gewählt werden kann.

Nachfolgend wird die Bedeutung der besonderen Symbole beschrieben, die in der Liste zur Wahl des Identifizierers verwendet werden.

Symbol	**Beschreibung**
	GPL-Anweisung
	Konstante eines Moduls oder einer Gruppe oder einer Bibliothek
	Variable eines Moduls oder einer Gruppe
	Variable einer Bibliothek
	Vektor einer Bibliothek
	Matrix einer Bibliothek
	Funktion einer Bibliothek
	Gruppennachricht
	Etikett
	lokale Variable
	lokaler Vektor
	lokale Matrix
	einfacher Parameter
	Vektor-Parameter
	Matrix-Parameter

8.7.3 Liste der verfügbaren Tastaturkürzel

Um die Befehle des Browsers zu aktivieren, können die entsprechenden Optionen des Menüs ***Debug*** direkt oder mittels der Tastatur ausgewählt werden.
Die zu verwendenden Tasten sind:

Drucktaste	**Beschreibung**
F2	setzt den Cursor auf die Zeile, wo das markierte Symbol definiert ist. Sind in der Datenbank des Browsers mehrere Symbole mit dem vorgegebenen Namen definiert, erscheint ein Dialogfenster, in dem der Benutzer das gewünschte Symbol wählen kann.
Umschalt+F2	setzt den Cursor auf den ersten Bezug des ausgewählten Symbols. Tritt Mehrdeutigkeit auf, erscheint das Dialogfenster zur Wahl des gewünschten Symbols.
Strg+F2	öffnet ein Dialogfenster zur Wahl des gewünschten Symbols
Strgl+'+' oder Strg+Seite-nach-oben	setzt den Cursor auf den folgenden Bezug (die Taste '+' der Zehnertastatur verwenden)

Drucktaste	**Beschreibung**
Strg+'-' oder Strg+Seite-nach-unten	setzt den Cursor auf den vorherigen Bezug (die Taste '-' der Zehnertastatur verwenden)

9 Zubehörprogramme

9.1 XConfMerge: Programm zum Zusammenführen von Konfigurationsdatei

XConfMerge ist ein Tool zum Zusammenführen von Konfigurationsdateien. Es wird von der Befehlszeile aus dem Bin-Verzeichnis ausgeführt, da XConfMerge die Datei tpa.ini liest.

Die von XConfMerge gelesenen Dateien sind:

- hardware.xconf: enthält die (physischen) Daten des virtuellen-physischen und der Hardware-Konfiguration;
- devices.xconf: entählt die Konfigurationsdaten von Gruppen, Untergruppen und Geräten (logisch)
- devices.xmlng: enthählt die übersetzbaren Nachrichten. In jedem Fall werden alle Sprachdateien im Ordner betrachtet.
- addresses.xdb: enthält die logischen Adressen der Geräte.

Die zu übergebenden Argumente sind:

- der Folder in dem die neuen Dateien zu lesen sind;
- die Nummer des Moduls, auf das sich die benutzerdefinierten Daten beziehen. Wenn keine Zahl angegeben wird, gilt sie als 0 als Standard.

Der Pfad zum Lesen der zu aktualisierenden Datei, der dann mit dem Pfad zum Schreiben der mit der Zusammenführen-Prozedur erhaltenen Datei übereinstimmt, wird aus dem Datensatz in tpa.ini abgeleitet.

Warnung: Die zu aktualisierende Dateien werden überschrieben und es wird keine automatische Sicherung durchgeführt.

Zusammenführungsregeln für Gruppenkonfigurationsdateien:

1. Wenn dieselbe Gruppe, Untergruppe oder dasselbe Gerät in beiden Dateien vorhanden sind, behält man die Daten und aktiviert man die alte Datei.
2. Wenn die Gruppe, Untergruppe oder Gerät nur in der neuen Datei existiert, werden die Daten und die Aktivierungen kopiert.
3. Wenn die Gruppe, Untergruppe oder Gerät nur in der alten Datei existieren, werden sie gelöscht.

Zusammenführungsregeln der Hardware-Konfigurationsdatei und der physischen

1. Wenn in beiden Dateien die gleiche Hardware vorhanden ist, behalten sich die Hardware und die physische virtuelle Hardware der neuen Datei mit den in der alten Datei definierten Aktivierungen.
2. Wenn die Hardware nur in der neuen Datei vorhanden ist, behalten sich die neue Hardware mit ihrer Aktivierung und das neue Virtuell-Physisches.
3. Wenn die Hardware nur in der alten Datei vorhanden ist, wird sie zusammen mit der physischen virtuellen Datei gelöscht.

Zusammenführungsregel der Datei der logischen Adressen:

1. Wenn sich die Datei nicht im Ordner der zu importierenden Dateien befindet, werden die logischen Adressen der benutzerdefinierten Geräte beibehalten und den neuen Geräten, falls vorhanden, neue Adressen zugewiesen.
2. Wenn sich die Datei im Ordner der zu importierenden Dateien befindet, dann werden die darin enthaltenen logischen Adressen gelesen und verwendet.

Regeln für das Zusammenführen von Nachrichtendateien:

1. Es wird keine Zusammenführung durchgeführt, aber die neue Nachrichtendatei wird in den Modulordner kopiert.

Am Ende der Ausführung ergibt das Zusammenführungstool folgende Werte:

0	Alles OK
1	Die Dateizusammenführung ist fehlgeschlagen
2	Keine Argumente definiert
3	Die als Argument übergebene Modulnummer ist falsch
4	Der als erste Argument angezeigte Ordner existiert nicht

9.2 XParMerge: Programm zum Zusammenführen von zwei Parameterdateien

XParMerge ist ein Werkzeug zum Zusammenführen von zwei parametrischen Dateien.

Es wird von der Befehlszeile aus dem Ordner bin ausgeführt, da XParMerge die Datei tpa.ini liest. Die zu übergebenden Argumente sind: der Name der neuen Datei und die Nummer des Moduls, auf das sie sich bezieht.

Der Pfad zum Lesen der zu aktualisierenden Datei und der Pfad zum Schreiben der Datei, die mit dem Zusammenführen-Verfahren erhalten werden, werden aus dem Datensatz in tpa.ini abgeleitet.

Warnung: Es wird keine automatische Sicherung der alten parametrischen Datei durchgeführt, sondern wird sie überschrieben.

Zusammenführungsregeln der technologischen Parameterdateien:

1) wenn das gleiche Steuerelement in beiden Dateien vorhanden ist, wird der Wert der alten Datei behalten, aber werden die anderen Parameter, die sie definieren, aktualisiert, (deaktiviert, sichtbar, GPL-Variablenname...), indem Sie diese aus der neuen Datei übernehmen werden;

2) wenn eine Steuerung, die in der alten Datei nicht vorhanden ist, in der neuen Datei definiert wird, bleibt die Steuerung erhalten;

3) wenn in der alten Datei eine Steuerung definiert ist, die in der neuen Datei nicht vorhanden ist, wird die Steuerung gelöscht.

Zusammenführungsregeln der Werkzeugparameterdateien:

1) Wenn sich der Referenzdialog der neuen Datei vom Referenzdialog der alten Datei unterscheidet, werden alle Dialoge der neuen Datei, falls vorhanden, beibehalten und die der alten Datei aktualisiert (neue Steuerelemente entfernt oder hinzugefügt);

2) Wenn der Referenzdialog der neuen Datei mit dem Referenzdialog der alten Datei identisch ist, werden die Dialoge der neuen Datei zu den Dialogen der alten Datei hinzugefügt;

3) Wenn der Referenzdialog nur in der alten Datei vorhanden ist, werden er und seine Dialoge gelöscht.

10 GPL-Sprache

10.1 Grundlegende Begriffe

10.1.1 Einführung in die GPL-Sprache

Die GPL-Sprache (General Purpose Language) ist die zur Schaffung von Funktionen des Systems Albatros verwendete Sprache.
Ihre Struktur ähnelt teilweise BASIC, unterscheidet sich aber dadurch, daß zahlreiche Anweisungen zur Steuerung der Vorrichtungen vorhanden sind.
Die Sprache besteht aus über 200 *Anweisungen*, die der Bequemlichkeit halber in Gruppen von Anweisungen mit ähnlichen Funktionen geordnet werden.
Die Sprache ist außerdem multitasking, d.h. es können mehrere Tasks gleichzeitig ausgeführt werden.

Typische Syntax einer GPL-Anweisung
Alle GPL-Anweisungen folgen einer selben Struktur, die folgendes Schema widerspiegelt:

```
AnweisungName Parameter-1, Parameter-2, ..... Parameter-N
```

Die Anzahl tatsächlich vorhandener Parameter hängt von der Anweisung und dem Kontext ihrer Anwendung ab; die Höchstzahl für eine GPL-Anweisung oder -Funktion zulässiger Parameter beträgt generell 120. In manchen Fällen können auch keine Parameter vorhanden sein.

Der kleinste Block des GPL-Codes ist die Funktion.

Unterteilung des Codes in Gruppen
Der GPL-Code wird in Blöcke unterteilt, die die logische Unterteilung der Maschine in Gruppen widerspiegeln. Pro Gruppe gibt es also eine Datei mit dem ihr zugeordneten Code. Außer den Dateien mit dem Code der Gruppen, die an der Maschine vorhanden sind, gibt es eine Datei mit globalen Variablen und Konstanten, die vom GPL-Code aller Gruppen zu sehen sind, und die Bibliotheken. Darin ist von der Maschinenkonfiguration unabhängiger Code enthalten, der somit leicht von einer Maschine zur anderen übertragbar ist.

10.1.2 Variablen

Variablen sind Behälter für Informationen, die von der GPL-Sprache zur Ablage von Werten verwendet werden, die zum Ablauf des Programms notwendig sind.
Variablen sind durch einen "Typ" gekennzeichnet, der die Eigenschaften der darin abzulegenden Information widerspiegelt. Jeder Variablen ist außerdem eine bestimmte Sichtbarkeit zugeordnet, die festlegt, welche Code-Mengen oder -Untermengen auf sie einwirken (lesen und schreiben) dürfen.

Datentypen

EINFACHE oder SKALARE DATEN
GPL unterstützt einfache und zusammengesetzte Datentypen. Die einfachen Datentypen ähneln denen, die in den meisten Programmiersprachen zur Verfügung stehen.

Char
Es handelt sich um eine ganze Zahl, deren Vorzeichen im Intervall [-128 ; +127] eingeschlossen ist und deren Länge 1 Byte beträgt.
Die Deklaration einer Variablen des Typs Char erfolgt mittels folgender Syntax:

```
VariableName as char
```

Integer
Es handelt sich um eine ganze Zahl, deren Vorzeichen im Intervall [-2147483647 ; +2147483647] eingeschlossen ist und deren Länge 4 Byte beträgt (entspricht dem Typ "long" in C).
Die Deklaration einer Variablen des Typs Integer erfolgt mittels folgender Syntax:

```
VariableName as Integer
```

Float
Es handelt sich um eine Gleitkommazahl,die im Intervall [-3,402 E+38; +3,402 E+38] eingeschlossen ist und deren Länge 4 Byte beträgt (normalerweise zur Darstellung der Geschwindigkeit verwendet).
Die Deklaration einer Variablen des Typs Float erfolgt mittels folgender Syntax:

VariableName as Float

Double. Es handelt sich um eine Gleitkommazahl, die im Intervall [-1,797 E+308; 1,797 E+308] eingeschlossen ist und deren Länge 8 Byte beträgt (normalerweise zur Darstellung der Position verwendet).
Die Deklaration einer Variablen des Typs Double erfolgt mittels folgender Syntax:

VariableName as Double

Diese Datentypen können gleichzeitig in einem Ausdruck zum Einsatz kommen. GPL wandelt sie automatisch um, ohne durch eine Mitteilung darauf hinzuweisen. Es ist also auf einen möglichen Verlust von Informationen zu achten, der beim Gebrauch von verschiedenen Daten im selben Ausdruck vorkommen kann.

In manchen Situationen ist die Umwandlung nicht zulässig. In diesen Fällen erscheinen Hinweise vom Kompilierer oder Systemfehler.

ZUSAMMENGESETZTE DATEN

Array
Es handelt sich um eine Menge von einfachen Variablen, alle desselben Typs, die durch Zuordnung eines Index zum Namen der Variablen erhalten worden ist. Der Index ist in eckigen Klammern eingeschlossen. Heißt das Array z.B. "Parameter", wird das erste Element der Menge als "Parameter[1]", das zweite als "Parameter[2]" usw. angegeben.
Das Array verfügt über eine feste Anzahl Elemente, die bei der Deklaration festzulegen ist. Die typische Deklaration eines Array erfolgt mittels folgender Syntax:

Parameter[10] as Integer

Dabei gibt *Parameter[10]* an, dass der Name des Array "Parameter" lautet und dass er aus 10 Elementen besteht; *as integer* definiert den Typ der einfachen Daten der einzelnen Array-Elemente, nämlich in diesem Fall Elemente des Typs Integer.
Arrays können aus Daten einfachen Typs oder aus Strings bestehen.
Ein "Array" kann maximal aus 262144 Elementen bestehen.
Bei ihrer Deklaration kann man die Vektoren unmittelbar im GPL-Code initialisieren. Die GPL-Syntax kann sein wie folgt:
[READONLY] Vector[Zeilennummer] as integer = 1,2,3,4
[READONLY] Vector[Zeilennummer] as string = "eins", "zwei", "drei", "vier"

Matrizen
Matrizen sind zweidimensionale Arrays, d.h. Variablen, die mit zwei Indizes versehen sind. Eine Matrix ist mit einer Tabelle vergleichbar, deren Daten in Zeilen und Spalten geordnet sind. Um ein Feld der Tabelle anzugeben, definiert man in welcher Zeile und Spalte der Tabelle es sich befindet. Der erste Index der Matrix entspricht der Zeilennummer und der zweite der Spaltennummer.
Im Gegensatz zu den Arrays können die Matrizen unterschiedliche Datentypen enthalten, aber mit folgenden Einschränkungen: Wir können für jede Spalte einen einfachen Datentyp, unterschiedlich für jede Spalte, verwenden, aber wir können den Typ innerhalb der gleichen Spalte nicht ändern. Es kann z.B. eine Matrix definiert werden, bei der die erste Spalte des Typs Integer und die zweite des Typs Float ist. Hingegen ist eine Matrix mit einem Integer und einem Float in der ersten Zeile und einem Char und einem Double in der zweiten Zeile nicht zulässig. Die Zeilen haben in Bezug auf den Datentyp der sie zusammensetzenden Elemente gleich zu sein.
Die Deklaration einer Matrix kann nach einer der folgenden Syntax-Arten erfolgen:

offset[10] as double double double

Teil_Größe[50] as Float:Länge Float:Breite Float:Stärke

Bei der zweiten Deklarationsart wird jeder Spalte ein Etikett bzw. ein symbolischer Name zugeordnet. Die symbolischen Namen der Spalten erweisen sich als besonders nützlich, wenn man mit sehr großen Matrizen arbeitet. In solchen Fällen ist es schwer, sich die Bedeutung der Daten in allen Spalten der Matrix zu merken. Der symbolische Name dient der sofortigen Bestimmung des Datentyps, mit dem man arbeitet. "Offset[1][3]" ist z.B. weniger klar als "Offset[1].Achse_X".
Matrizen können nur Daten einfachen Typs enthalten. Es können z.B. keine Matrizen geschaffen werden, die Strings enthalten. Eine Matrix kann maximal aus 262144 Zeilen bestehen.
Bei ihrer Deklaration kann man die Matrizen unmittelbar im GPL-Code initialisieren. Die GPL-Syntax kann sein wie folgt:

```
[READONLY] nomematrice[numerorighe] as double double integer double = _
  1.1, 2.2, 3, 0.1 _
  1.2, 3.4, 5, 0.1 _
  2.1, 5.6, 6, 0.1
```

Strings
Strings sind Datenmengen bzw. Daten des Typs Char, die allerdings auf besondere Weise verwaltet werden, da sie lesbaren Text darstellen.
Ein String ist einem Char-Array sehr ähnlich. Der größte Unterschied besteht in einem Endzeichen, das automatisch zum Ende der Strings hinzugefügt wird. GPL verfügt außerdem über einige Anweisungen zur Bearbeitung der Strings.
Normalerweise werden Strings benutzt, um Nachrichten zu schreiben, die der Benutzer auf dem Bildschirm oder in einer Report-Datei lesen kann.
Die Deklaration einer Variablen des Typs String kann nach einer der folgenden Syntax-Arten erfolgen:

```
VariableName as String

VariableName as String[20]
```

Bei der ersten Deklarationsart wird per Default eine Größe von 256 Zeichen zugewiesen. Im zweiten Fall ist die maximale Größe des Strings definiert.
String-Werte sind Folgen von Zeichen, die durch doppelte Anführungszeichen begrenzt sind. Z.B.: "Drücken Sie die Schaltfläche".
Um das Zeichen '"' (Anführungszeichen) einzugeben, müssen Sie es zweimal eingeben. Z.B.: "Drücken Sie die Schaltfläche ""Start""".
Um Zeichen durch den numerischen Code einzufügen, schreiben Sie innerhalb der Zeichenfolge die Zeichen \u, gefolgt vom numerischen Wert des hexadezimalen Zeichens. Beispiel: \u20ac ist das Euro-Symbol. Wenn Sie "\u20ac 15,6" schreiben, erhalten Sie € 15,6.

Konventionen und Begriffe

Häufig verwendete Begriffe

ARGUMENT	Es handelt sich um ein Argument der Anweisung; je nach Anweisung kann es als *Konstante*, *Variable* oder als *Parameter* definiert werden; steht es in eckigen Klammern (**[ ]**), kann es ausgelassen werden, was natürlich eine andere Ausführungsart der Anweisung nach sich zieht
SCHLÜSSELWORT	Es handelt sich um ein Argument, das unter vordefinierten Argumenten zu wählen ist, die generell in Großbuchstaben geschrieben sind; für die Liste der Schlüsselwörter steht eine eigene Hilfe-Seite zur Verfügung
PARAMETER	Es handelt sich um das Argument einer Anweisung, das nicht in der Anweisung selbst definiert ist, sondern bei Ausführung der Funktion eben als Argument an diese übergeben wird; wird manchmal auch *Parameter-Argument* genannt
KONSTANTE	Es handelt sich um ein durch den Metabefehl CONST fest definiertes Argument oder um ein fest in der Anweisung verankertes Argument
VARIABLE	Es handelt sich um ein Argument, das als globale Variable eines Moduls entweder einer Gruppe oder mittels der Anweisung LOCAL definiert worden ist und das als einfache Variable, als Vektor oder als Matrix organisiert werden kann. Siehe Variablen
KONFIGURATIONSPARAMETER	Es handelt sich um ein Argument, das in der Konfiguration definiert worden ist, z.B. die Parameter einer Achse

Häufig wiederkehrende Argumente in der Beschreibung von Anweisungen

Nachfolgend sind Argumente betreffende, häufig wiederkehrende Begriffe aus der Syntax von GPL-Anweisungen aufgelistet, jeweils gefolgt von der zugehörigen Beschreibung. In den Fällen, in denen ein Argument andere Werte als die nachfolgend beschriebenen annehmen kann, wird dessen Beschreibung im Abschnitt *Argumente* der Hilfe-Seite der Anweisung wiederaufgenommen.

EingangName	Vorrichtungsname eines digitalen Eingangs
AusgangName	Vorrichtungsname eines digitalen Ausgangs
FlagName	Vorrichtungsname eines Flag Switch oder Flag Bits
PortName	Vorrichtungsname eines Eingangs-, Ausgangs- oder Flag-Ports
TimerName	Vorrichtungsname eines Timers
ZählerName	Vorrichtungsname eines Zählers
FunktionsName	Funktionsname (gilt auch als Parameter des Typs Vorrichtung im Fall von ERRSYS.)
UnterprogrammName	Name eines Unterprogramms, entspricht einem *Etikett*, an dessen Erklärung verwiesen wird; der Aufruf eines Unterprogramms erfolgt mittels der Anweisung "CALL UnterprogrammName".
Achse	Name einer Achse
Konstante	ein Zeichen oder eine ganze Zahl oder ein Double oder ein Schlüsselwort
Wert	Konstante oder Variable (der *Typ* hängt von der Anweisung ab)
Variable	Name: einer Variablen, eines Vektor- oder eines Matrix-Elements
VorrichtungVariable	Name eines *Parameters des Typs Vorrichtung*
Matrix	Name einer Matrix
Vektor	Name eines Vektors
Etikett	Name des Sprung-Etiketts oder Name eines Unterprogramms.
Zustand	logischer Zustand, kann ON oder OFF bzw. 1 oder 0 sein
Timeout	Zeitdauer, innerhalb welcher ein bestimmtes Ereignis stattzufinden hat, oder eine Verzögerung (Konstante oder Variable)
Maß	Wert des Maßes (Double-Konstante oder Double-Variable)
Radius	Wert des Radius (Double-Konstante oder Double-Variable)
Winkel	Wert des Winkels (Double-Konstante oder Double-Variable)
UmdrZahl	Anzahl Umdrehungen (Double-Konstante oder Double-Variable)
Geschwindigkeit	Geschwindigkeitswert (Float-Konstante oder Float-Variable)
Richtung	Drehung im oder entgegen dem Uhrzeigersinn (Variable oder Konstante: CW oder CCW)
Operand	(Konstante oder Variable oder VorrichtungName)
Ergebnis	Ergebnis des Vorgangs (Variable oder VorrichtungName)
VorrichtungName	Name einer beliebigen Vorrichtungsart (oder Parameter des Typs Vorrichtung)
StrKonstante	Sequenz von Zeichen zwischen Anführungszeichen (z.B. "String")
StrVariable	Name eines Zeichenvektors oder String
Operator	vergleichende Operatoren, auch untereinander kombiniert: > (größer) = (gleich) < (kleiner) sie können durch Annäherung kombiniert werden, z.B. >= (bedeutet größer oder gleich)
Typ	Konstanten- oder Variablentyp: "Char" (8 Bit), "Integer" (32 Bit), "Float" (32 Bit), "Double" (64 Bit), "String"
VorrichtungsParameter	eine Variable, die eine Vorrichtung darstellt. Die Vorrichtungen werden in Konfiguration definiert.

Häufig bei Achsen verwendete Begriffe

theoretisches Maß (oder Target)	Laufende "theoretische" Position, die jeden Augenblick von der Numeriksteuerung gemäß dem Algorithmus zur Schaffung des Geschwindigkeitsprofils vorgegeben wird.
echtes Maß	Tatsächliche Achsenposition wie sie vom Stellungsgeber ermittelt worden ist. Unterscheidet sich vom theoretischen Maß um eine "Verfolgungsfehler" oder "Loop-Fehler" genannte Abweichung.

End-Maß Programmiertes Endmaß einer Bewegung. Der Algorithmus zur Berechnung des Geschwindigkeitsprofils sorgt dafür, dass das theoretische Maß genau das Endmaß erreicht.

Fenster Positionierungstoleranz Programmierbares Intervall, dessen Mittelpunkt dem theoretischen Endmaß entspricht; wenn das tatsächliche Maß in das Intervall fällt, wird die Bewegung als abgeschlossen angesehen.

Loop-Fehler Unterschied, der sich jeden Augenblick zwischen theoretischem und tatsächlichem Maß einer Achse ergibt; ist normalerweise proportional zur Bewegungsgeschwindigkeit und umgekehrt proportional zum "proportionalen Loop-Gain".

großes Fenster Erreichen Höhe Fenster für Ankunft in Höhe multipliziert mit einem mit der Anleitung SETBIGWINFACTOR programmierbaren Faktor.

proportionaler [Loop] Gain Programmierbarer Parameter zur Einstellung der Achse; legt das Verhältnis zwischen aktueller Geschwindigkeit und relativem Loop-Fehler fest.

Feed forward Programmierbarer Parameter zur Einstellung der Achse; bestimmt einen direkten (zur programmierten Geschwindigkeit proportionalen) Beitrag, der auf die Geschwindigkeit des Antriebs einwirkt. Dient der Verminderung des Loop-Fehlers bei gleicher Geschwindigkeit und gleichem proportionalen Gain.

Feed rate override Prozentsatz der programmierten Geschwindigkeit. Dieser Parameter dient der Verminderung der Ausführungsgeschwindigkeit um einen Prozentsatz zwischen 0 und 100% im Verhältnis zur programmierten Geschwindigkeit.

Toleranz Verschiebungswert, um den die Achse von der ursprünglichen Linie in einer Mehrachsen-Interpolation zwischen zwei auf einander folgenden Verschiebungsblöcken abweicht.

mechanisches Spiel Platz zwischen Hohlraum und Zahn eines Getriebepaares.

Daten konvertieren

In jedem mathematischen Ausdruck werden mit Ausnahme der EXPR-Anweisung die Datentypen der Operanden in den Datentyp der Variablen Ergebnis konvertiert und dann wird der Vorgang ausgeführt.

Es ist sehr wichtig, die Deklaration der Datentypen zu beachten, denn diese können das Ergebnis beeinflussen. Die Nachfolgende Tabelle dient als Beispiel, wie die Ergebnisse gemäß dem Datentyp ändern können:

DIV	Operand 1(Integer)	Operand 2 (Double)	Ergebnis (char)
	3	5.0	0
	5	1.9	5
	1200	107.2	Undefiniert
	1200	250.0	Undefiniert

DIV	Operand 1(Double)	Operand 2 (Double)	Ergebnis (Double)
	3	5.0	0.6
	5	1.9	2.631
	1200	107.2	11.194
	1200	250.0	4.8

Wenn in der Instruktion EXPR sind die Operanden nicht alle vom gleichen Typ sind, dann wird eine automatische Konvertierung durchgeführt so dass das Vorgangsergebnis gleich ist dem, das der beiden größer ist, nach folgender Regel:

- char <integer
- float < double

- char oder integer < float oder double.

Nachdem der Ausdruck gelöst worden ist, dann wird das Ergebnis in den Typ Variable konvertiert.

EXPR	Operand 1 (Double)	+	Operand 2 (Integer)	/	Operand 3 (Float)	Ergebnis (Integer)
	900.0	+	100	/	400.0	900

EXPR	Operand 1 (Double)	+	Operand 2 (Integer)	/	Operand 3 (Float)	Ergebnis (Double)
	900.0	+	100	/	400.0	900.25

Deklaration und Sichtbarkeit der Variablen

Die Deklarationen der Variablen und Konstanten kann nur an bestimmten Punkten des GPL-Codes erfolgen.
Folgende Variablen können definiert werden:

- Globale Modul-Variablen
- Globale Gruppen-Variablen
- Lokale Variablen (nur Variablen)
- Globale Bibliothek-Variablen

Es können höchstens 2048 globale (Modul- und Gruppen-) Variablen deklariert werden.

Es können *Modifizierfaktoren* definiert werden, die den Variablen zusätzliche Eigenschaften verleihen.

Globale Modul-Variablen
Die globalen Modul-Variablen sind in einer besonderen Datei enthalten, die durch die Menüoption **Datei->Globale Variable öffnen** aufgerufen wird.
Die Deklaration erfolgt - wie in den vorherigen Abschnitten erklärt - durch Spezifizierung des Namens der Variablen, gefolgt vom Schlüsselwort "AS", gefolgt vom Datentyp (oder von den Datentypen im Fall der Matrizen).
Diese Variablen sind direkt vom Code aller Gruppen sichtbar.

Globale Gruppen-Variablen
Die globalen Gruppen-Variablen werden am Anfang des die Gruppe betreffenden Codes definiert. Sie sind vor den GPL-Funktionen zu deklarieren.
Diese Variablen sind direkt vom gesamten Code innerhalb der Gruppe sichtbar. Außerdem kann die Sichtbarkeit dieser Variablen außerhalb der Gruppe erweitert werden, indem man sie zu "öffentlichen" Variablen deklariert.
Öffentliche Variablen sind von außerhalb der Gruppe nicht direkt zugänglich. Um sie aufzurufen, ist ihr Name nach dem zugehörigen Gruppennamen zu verwenden. Möchte man z.B. die öffentliche Variable "Offset" der Gruppe "Achsen" aus dem Code der Gruppe "Main" ändern, ist "SETVAL 10 Achsen.Offset" zu schreiben.
Zur Deklaration einer globalen Gruppen-Variable wird die gleiche Syntax wie für globale Modul-Variablen verwendet. Der Hauptunterschied besteht in der Definition der öffentlichen Variablen.
Zur Definition einer oder mehrerer öffentlicher und privater Variablen werden die Etiketten "Public:" und "Private:" verwendet. Zum Beispiel:

```
Public:
  Offset as Double
  Geschwindigkeit as Float
Private:
  Werkzeug as Integer
```

Lokale Variablen
Lokale Variablen werden innerhalb des Funktionskörpers deklariert. Sie sind vor sämtlichen anderen Anweisungen zu deklarieren mit Ausnahme der Deklaration der Funktionsparameter.
Lokale Variablen sind nur innerhalb der Funktion aufrufbar.
Diese Variablen werden mit Wert 0 erst bei Beginn der Ausführung der jeweiligen Funktion geschaffen (der notwendige Speicherplatz wird zugewiesen) und am Ende der Ausführung gelöscht (die Zuweisung von Speicherplatz wird aufgehoben). Globale Variablen werden hingegen bei der Initialisierung des Moduls geschaffen und sind immer in der "Diagnostik" sichtbar.
Die Deklaration einer lokalen Variablen folgt der bereits dargestellten Syntax, allerdings ist das Schlüsselwort "LOCAL" vorauszuschicken.
Zum Beispiel:

```
Function Bearbeitung
local Maß_Zentrum as Double
movabs X,Maß_Zentrum
```

fret

Globale Bibliothek-Variablen
Die globalen Bibliothek-Variablen sind in den GPL-Code - Bibliotheken enthalten. Sie verhalten sich analog zu den globalen Gruppen-Variablen.

Modifizierfaktoren

Modifizierfaktoren: READONLY
Die globalen Modul- und Gruppen-Variablen können als READONLY deklariert werden.
Eine schreibgeschützte Variable ist eine Variable, deren Wert nicht vom GPL-Code aus, sondern nur "von außen", d.h. vom Archiv der technologischen Parameter von Albatros geändert werden kann.
Das Archiv der technologischen Parameter ist eine Datenbank, in der die Maschine kennzeichnenden Werte gespeichert sind, die sich aber langfristig ändern können, sei es aufgrund von Änderungen an der Maschine, sei es aufgrund von außerordentlichen Wartungseingriffen. Diese Informationen werden normalerweise während der Initialisierung der Steuerung in eine GPL-Matrix eingefügt.
Beispiele für diese Art Informationen sind die Offsets der Bearbeitungsbereiche oder die Abmessungen und die technologischen Parameter der Werkzeuge.
Deklariert man diese Variablen als schreibgeschützt, verhindert man versehentliche Änderungen von Informationen, die generell beim normalen Betrieb der Maschine nicht zu ändern sind.
Die maximale Größe einer readonly Variable beträgt 128 Kbyte.
Die Deklaration einer readonly Variablen erfolgt mittels folgender Syntax:

readonly VariableName as Typ

Modifizierfaktoren: NONVOLATILE
Als **NONVOLATILE** deklarierte Variablen werden auf nichtflüchtigem (mit Batterie versehenem) RAM statt auf normalem RAM gespeichert. Daher gehen die in diesen Variablen enthaltenen Werte beim Ausschalten der Numeriksteuerung nicht verloren.
Die Deklaration einer nichtflüchtigen Variablen erfolgt mittels folgender Syntax:

nonvolatile VariableName as Typ

Zum Beispiel:
nonvolatile OffsetBereiche[2] as Double:OffsetX Double:OffsetY Double:OffsetZ

Nur die globalen Variablen einer Gruppe und eines Moduls können als "nonvolatile" definiert werden.
Die maximale Gesamtgröße von Variablen, die auf nichtflüchtigem RAM gespeichert werden, beträgt 65536 Byte. Die maximale Größe einer einzelnen nichtflüchtigen Matrix beträgt 1024 Byte.

Zuweisung eines RANGE

Bei der Deklaration einer Variablen kann ihr auch ein Wertebereich zugewiesen werden. Zur Zeit, wird jedoch die Beachtung der Bereichsgrenzen während der Ausführung noch nicht geprüft, ausschließlich der Kompilierer kontrolliert, ob konstante Werte zugewiesen worden sind (z.B. zum Initialisieren der Variablen).
Der Nutzen liegt also vorwiegend in einer Art Eigendokumentation des Codes.
Die Definition von Bereichen erfolgt mittels folgender Syntax:

VariableName Range: minWert..maxWert AS Typ

Zum Beispiel:
WerkzeugNummer Range: 1..100 as Integer

Zugangsebenen zum Lesen / Schreiben

Die Zugangsebenen zum Lesen und Schreiben dienen der Angabe der Mindest-Zugangsebene zum System, die zur Anzeige (Zugang zum Lesen) und Änderung (Zugang zum Schreiben) eines Werts notwendig ist.
Es ist folgende Syntax zu beachten:

VariableName Read=S Write=M AS Typ

Die Schlüsselwörter zur Angabe der Ebenen lauten:

- READ — Lesen
- WRITE — Schreiben

Folgende Werte können zugewiesen werden:

• U oder USER	Benutzer
• S oder SERVICE	Wartung
• M oder MANUFACTURER	Hersteller
• T oder TPA	tpa

Bei Default lauten die Werte wie folgt:

• READ	Lesen für Wartung (S oder SERVICE)
• WRITE	Schreiben für Hersteller (M oder MANUFACTURER) und tpa (T oder TPA)

10.1.3 Konstanten

GPL sieht vier Konstantentypen vor:
- Integer
- Double
- Char
- String

Die Konstanten des Typs Char werden durch Einsatz der einfachen Anführungszeichen wie folgt deklariert:

```
Const COD = 'A'
```

Die Konstanten des Typs String werden durch Einsatz der doppelten Anführungszeichen wie folgt deklariert:

```
Const MSG = "Bearbeitungsstart"
```

Konstanten des Typs Integer und Double werden mittels folgender Syntax deklariert:

```
Const PI = 3.14
Const MSGBOX = 12
```

Für Konstanten des Typs Integer ist auch eine binäre und hexadezimale Schreibweise vorgesehen:

```
Const MASK = $11001001b    ; binär
Const MASK = $F5h          ; hexadezimal
```

Auch die Gruppe-Konstanten und die Bibliothek können öffentlich oder privat sein.
Die Syntax ist dieselbe der Variablen.

Beispiel:

```
Public:
  Const PI = 3.14
  Const MSGBOX = 12
Private:
  Const MASK = $11001001b
```

BEMERKUNG: Es gibt keine Konstanten des Typs Float. Dezimale Werte sind notwendigerweise als Double zu deklarieren. Dies kann manchmal Hinweisnachrichten seitens des Kompilierers hervorrufen (wenn GPL-Anweisungen verwendet werden, die zum Gebrauch von Floats optimiert sind).

Es ist möglich die Konstanten als Resultat von Berechnungs-Ausdrücke zu bestimmen mit der folgenden Syntax:

```
Const a = 10
Const b = 20
Const c = a + b
```

Die zulässigen Operatoren sind dieselben, die in der Anweisung EXPR verwendet werden.

Vordefinierte Konstanten mit vorbelegtem Wert

In der GPL-Sprache sind einige vordefinierte Konstanten vorgesehen. Diese können direkt verwendet werden, ohne vorher definiert worden zu sein.
Die vordefinierten Konstanten und die zugehörigen Werte lauten:

ON	**1**
OFF	**0**
UP	**+1**
DOWN	**-1**
POSITIVE	**+1**
NEGATIVE	**-1**
CW	**1**
CCW	**0**
TRUE	**1**
FALSE	**0**
NOWAIT	**0**
WAIT	**1**
WAITACK	**2**
STORE	**1**
NOSTORE	**0**
NOPLACE	**0**
COM1	**0**
COM2	**1**
COM3	**2**
COM4	**3**
COM5	**4**
COM6	**5**
COM7	**6**
COM8	**7**
NOPARITY	**0**
ODDPARITY	**1**
EVENPARITY	**2**

Vordefinierte Konstanten mit vorbelegtem Wert in der Albatros-Inbetriebnahme

Die GPL-Sprache bietet einige vordefinierte Konstanten, deren Wert beim Start von Albatros festgelegt wird. Sie können in der Anweisung IFDEF verwendet werden.

_ID_MODULE	aktuelle Modulnummer. Die Modulnummer liegt zwischen 0 und 15.
_REMOTE_MODULE	Modultyp. Bei Fernmodul liegt der Wert der Konstante bei 1, bei Lokalmodul liegt der Wert bei 0; wenn das Modul in Systemkonfiguration nicht konfiguriert ist, ist die Konstante nicht definiert.
_VER_MAJOR	Albatros Hauptversionsnummer. Wenn die Version Albatros 3.2.1 ist, liegt der Wert der Hauptversion bei 3.
_VER_MINOR	zweite Versionsnummer Albatros. Wenn die Version Albatros 3.2.1 ist, liegt der zweite Wert der Version bei 2.
_VER_REVISION	Revisionsnummer Albatros. Wenn die Version Albatros 3.2.1 ist, liegt der Revisionswert bei 1.
_VER_SP	Zeichenkette, die das Service Pack beschreibt, wenn es installiert ist, z. B. "Service Pack 1f", sonst undefiniert.
_VER_FULL	vollständige Versionsnummer. Im Fall der Version 3.2.1 ist sie $00030201H.

10.1.4 Schlüsselwörter

Schlüsselwörter sind Identifizierer reservierten Gebrauchs und können nicht anderweitig eingesetzt werden.

Die verfügbaren Schlüsselwörter lauten:

Alle Namen der GPL-Anweisungen	Für die Beschreibung aller GPL-Anweisungen wird auf den Abschnitt "Anweisungen" des Handbuchs verwiesen
Alle Datentypen	Siehe Variablen
Die Parameter des Typs Vorrichtung	Siehe Parameter des Typs Vorrichtung

EXIST	Wird in der Anweisung IFDEF benutzt, um die Anwesenheit einer Gruppe zu prüfen. Siehe Anweisung IFDEF.
NOTEXIST	Wird in der Anweisung IFDEF benutzt, um die Abwesenheit einer Gruppe zu prüfen. Siehe Anweisung IFDEF.
LINKED	ist in der Anweisung IFDEF verwendet, um die Kompilierung der Codeblöcke zu aktivieren, wenn das Gerät im Virtuell-Physisch angeschlossen ist. Siehe die Anweisung IFDEF.
UNLINKED	ist in der Anweisung IFDEF verwendet, um die Kompilierung der Codeblöcke zu aktivieren, wenn das Gerät im Virtuell-Physisch nicht angeschlossen ist. Siehe die Anweisung IFDEF.
FUNCTION	Deklaration einer Funktion. Siehe Funktionen
AS	Wird zur Deklaration der Variablen verwendet. Siehe Variablen
PUBLIC	Ist eine Eigenschaft einer Funktion. Siehe Funktionen
AUTORUN	Ist eine Eigenschaft einer Funktion. Gibt an, daß der Start der Funktion automatisch erfolgt. Siehe Funktionen
R= o READ	Ist eine Eigenschaft einer Funktion oder einer Variablen. Gibt die Zugangsebene zum Lesen an. Siehe Funktionen und Variablen und Zugangsebenen
W=o WRITE	Ist eine Eigenschaft einer Funktion oder einer Variablen. Gibt die Zugangsebene zum Schreiben an. Siehe Funktionen und Variablen und Zugangsebenen
CONST	Dient der Zuweisung eines symbolische Konstante genannten Namens mit einer Bedeutung anstelle einer Zahl, eines Zeichens oder eines Strings. Siehe Variablen
READONLY	Ist eine Eigenschaft einer globalen Variablen. Siehe Variablen
NONVOLATILE	Ist eine Eigenschaft einer globalen Variablen. Siehe Variablen
PRIVATE	Ist eine Eigenschaft einer Funktion. Siehe Funktionen
RANGE	Wird zur Definition eines Werte-Intervalls einer Variablen verwendet. Siehe Variablen
USER	Ist eine Eigenschaft einer Funktion oder einer Variablen. Gibt die Zugangsebene an. In diesem Fall Benutzer. Siehe Funktionen oder Variablen.
SERVICE	Ist eine Eigenschaft einer Funktion oder einer Variablen. Gibt die Zugangsebene an. In diesem Fall Wartung. Siehe Funktionen oder Variablen.
MANUFACTURER	Ist eine Eigenschaft einer Funktion oder einer Variablen. Gibt die Zugangsebene an. In diesem Fall Hersteller. Siehe Funktionen oder Variablen.
TPA	Ist eine Eigenschaft einer Funktion oder einer Variablen. Gibt die Zugangsebene an. In diesem Fall TPA. Siehe Funktionen oder Variablen.

10.1.5 Funktionen

Funktionen sind der kleinste Block im GPL-Code. GPL-Anweisungen können nicht nacheinander in eine Datei eingefügt werden, sondern sind in Funktionen zu gruppieren.

Funktionen sind - vom Kompilierer aus gesehen - all diejenigen Blöcke im GPL-Code, die mit einer Zeile beginnen, deren erstes Wort FUNCTION lautet. Es gibt jedoch kein Schlüsselwort, das auf das Ende des Textes einer Funktion hinweist. Die Funktion endet mit der Zeile vor Beginn der nächsten Funktion oder am Ende der Datei, die die Funktionen enthält.

Die Syntax zur Definition einer Funktion lautet:

FUNCTION *FunktionsName Attribute*
Parameter
lokale Variablen
Liste der GPL-Anweisungen

Eine Funktion ist außerdem eine besondere Vorrichtungsart von Albatros. Als Vorrichtung ist sie durch eine Serie von Eigenschaften gekennzeichnet, die alle Vorrichtungsarten gemeinsam haben: ein eindeutiger Name (nicht übersetzbar), ein beschreibender Name (in andere Sprachen übersetzbar, allerdings nicht innerhalb des GPL-Texts), ein Zeichen der Sichtbarkeit (je nachdem, ob die Vorrichtung öffentlich ist oder nicht), eine Zugangsebene zum Lesen und eine zum Schreiben (siehe folgenden Abschnitt).

Zugangsebenen
Da Funktionen besondere Arten von Vorrichtungen sind, unterstehen sie wie die anderen Vorrichtungen den Zugangsebenen. Mittels der Zugangsebenen wird angegeben, welcher der minimale Zugang zum System ist, mit dem etwas angezeigt (Zugangsebene zum Lesen) und ausgeführt (Zugangsebene zum Schreiben) werden kann.
Die Syntax lautet:

Function FunktionsName READ=S WRITE=M

Die Zugangsebenen sind durch die Schlüsselwörter READ (Lesen) und WRITE (Ausführung) bestimmt.
Folgende, den Zugangsebenen entsprechenden Werte können zugewiesen werden:

U oder USER	Benutzer
S oder SERVICE	Wartung
M oder MANUFACTURER	Hersteller
T oder TPA	tpa

Standardmäßig lauten die Werte wie folgt:

READ	Lesen für Wartung (S oder SERVICE) und Benutzer (U oder USER)
WRITE	Schreiben für Hersteller (M oder MANUFACTURER) und tpa (T oder TPA)

Autorun-Funktionen
Eine Autorun-Funktion wird automatisch bei Initialisierung der Maschine ausgeführt.
Autorun-Funktionen haben eine Eigenschaft, dass sie automatisch vom System nochmals gefahren werden, wenn sie aufgrund eines Systemfehlers beendet werden.
Die Syntax lautet:

Function FunktionsName autorun

Es reicht also, den Modifizierfaktor "autorun" zur Deklaration der Funktion hinzuzufügen.

Öffentliche Funktionen
Eine Funktion kann normalerweise nur vom Code, der sich innerhalb der Datei der Gruppe befindet, ausgeführt (aufgerufen) werden. Damit eine Funktion vom Code einer anderen GPL-Gruppe ausgeführt werden kann, ist sie als Funktion des Typs **public (öffentlich)** zu definieren. Die Syntax zur Definition einer öffentlichen Funktion lautet wie folgt:

Function FunktionsName public

Es reicht also, den Modifizierfaktor "public" zur Deklaration der Funktion hinzuzufügen. Der gruppenübergreifenden Gruppe angehörende Funktionen bilden eine Ausnahme, da sie immer **öffentlichen** Typs sind.

Untergruppenfunktionen
Eine Funktion kann einer Untergruppe zugewiesen werden, indem einfach der Name der Untergruppe dem der Funktion vorausgeschickt wird. Der Name der Untergruppe ist durch einen Punkt "." von dem der Funktion zu trennen. Folgende Funktion gehört z.B. zur Untergruppe "X" der Gruppe "Achsen".

```
Function    X.Nullstellung
  local     vel as float
  movabs    X,100
  waitstill X
Fret
```

Asynchrone Funktionen
Eine asynchrone Funktion wird automatisch von der numerischen Steuerung aufgerufen, wenn das Ereignis stattfindet, mit dem die Funktion verbunden ist.
Es gibt drei Arten von Ereignissen:

- Wechsel des Zustands eines digitalen Eingangs: Anweisung ONINPUT
- Wechsel des Zustands eines Flag Bit oder Flag Switch: Anweisung ONFLAG
- Auftreten eines Systemfehlers: Anweisung ONERRSYS

Wenn das Ereignis stattfindet, wird die Funktion - sofort nach Beendigung der laufenden Anweisung - implizit als FCALL aufgerufen (nicht als eigenständiger Task, sondern im Zusammenhang des Tasks, in dem die entsprechende Anweisung ON... ausgeführt worden ist).
Asynchrone Funktionen dienen normalerweise zur Verwaltung von Notfällen und haben daher äußerst rasch zu sein. Aus diesem Grund verwenden diese Funktionen nicht eine beliebige GPL-Anweisung, sondern eine Untermenge, die rasche Ausführungszeiten gewährleistet.

Funktionen mit Eingangsparametern (Parameterfunktionen)
Eine Funktion kann über deklarierte Eingangsparameter verfügen, aber in keinem Fall einen Wert zurückgeben.

Die Parameter können als besondere lokale Variablen angesehen werden, deren Wert von außen in dem Augenblick initialisiert wird, in dem die Funktion ausgeführt wird. Parameter werden durch das Schlüsselwort PARAM angegeben und folgen derselben Syntax wie die lokalen Variablen. Die Parameter sind in den ersten Zeilen des Funktionskörpers zu deklarieren, vor jeglicher Anweisung und vor den lokalen Variablen.

Parameter werden auf zwei Arten übergeben:

- ***als Wert:*** als Wert werden alle einfachen Datentypen übergeben, d.h. CHAR, INTEGER, FLOAT und DOUBLE. Die Übergabe als Wert beinhaltet das Schaffen einer Kopie des ursprünglichen Werts. Änderungen des Parameters kommen nur im Zusammenhang der Funktion zur Geltung.
- ***als Bezug:*** als Bezug werden die strukturierten Typen übergeben, d.h. ARRAYS, MATRIZEN und STRINGS. Die Übergabe als Bezug beinhaltet die Verwendung der ursprünglichen Variablen. Dementsprechend kommen Änderungen des Parameters im Zusammenhang der aufrufenden Funktion zur Geltung. Diese Eigenschaft kann dazu verwendet werden, der aufrufenden Funktion Rückkehrwerte zu liefern.

Die Ausführung einer Funktion startet normalerweise durch die Anweisung FCALL. Handelt es sich um eine Parameterfunktion, so sind nach dem Namen der Funktion die den Parametern zuzuordnenden Werte zu deklarieren.
Im folgenden Beispiel ist eine Parameterfunktion dargestellt, die eine Bohrung ausführt. Die Maße des Mittelpunkts der Bohrung und die Bewegungsgeschwindigkeit der Achse Z werden als Parameter an die Funktion übergeben.

```
Function Bohrung
  Param Qx as Double      ; Maß X des Mittelpunkts der Bohrung
  Param Qy as Double
  Param vel as Float      ; Bewegungsgeschwindigkeit

  Movabs      X, Qx, Y, Qy
  Waitstill   X,Y
  ....
  Fret
```

Der Aufruf dieser Funktion, um eine Bohrung z.B. an den Maßen (12.5 , 25.7) und bei einer Bewegungsgeschwindigkeit von 3 m pro Minute auszuführen, könnte folgendermaßen lauten:

```
Fcall   Bohrung 12.5, 25.7, 3.0
```

Die an die Funktion übergebenen Parameter haben der Anzahl und dem Typ nach mit den Parametern der aufgerufenen Funktion übereinzustimmen. Die Ausführung der aufrufenden Funktion beginnt am Ende der Ausführung der aufgerufenen Funktion wieder.
Als Parameter einer Funktion kann auch eine Vorrichtung angegeben werden. Auf diese Weise können Funktionen allgemeiner Verwendung geschrieben werden. Z.B. eine Nullstellung-Anweisung, die an allen Achsen der Maschine verwendet werden kann:

```
Function NULLSTELLUNG PUBLIC
        param       Achse as Axis
        movabs      Achse,100
Fret

Function MAIN
        .....
        Achsen.Nullstellung x
Fret
```

Die Funktion Nullstellung gehört zur Gruppe Achsen und wird als PUBLIC angegeben, so dass sie auch von Funktion anderer Gruppen erfasst werden kann. Die Main-Funktion ruft die Nullstellung-Funktion der Gruppe Achsen auf und gibt als Eingangsparameter an, welche Achse bewegt werden soll.

10.1.6 Parameter des Typs Vorrichtung

Die Parameter des Typs Vorrichtung sind besondere Variablen, die dem Bezug zu einer Vorrichtung der Maschine dienen.
Diese Datentypen können **ausschließlich** in der Deklaration der Parameter einer Funktion verwendet werden. Es können also keine Variablen dieses Typs angegeben werden. Die Definition der Namen und der anderen Eigenschaften der Vorrichtungen erfolgt in der Systemkonfiguration.
In der folgenden Tabelle sind die Parameter des Typs Vorrichtung und die entsprechenden Schlüsselwörter zur Deklaration der zugehörigen Parameter aufgelistet.

Typ	**Schlüsselwort**
Digitaler Eingang	INPUTDIG
Digitaler Ausgang	OUTPUTDIG
Analoger Eingang	INPUTANALOG
Analoger Ausgang	OUTPUTANALOG
Achse	AXIS
Timer	TIMER
Zähler	COUNTER
Flag Bit	FLAGBIT
Flag Switch	FLAGSWITCH
Flag Port	FLAGPORT
Eingangs-Port	INPUTPORT
Ausgangs-Port	OUTPUTPORT
Funktion	FUNCTION
Allgemeine Vorrichtung	DEVICE
Task	TASK

Im folgenden Beispiel wird ein Parameter des Typs Achse angegeben und verwendet:

```
Function Probe
   Param Achse as axis

   MovAbs       Achse,100
   WaitStill    Achse

Fret
```

10.1.7 Multitasking

Das System Albatros ist ein Multitasking-System, d.h. es können mehrere GPL-Tasks in Ausführung begriffen sein, wobei unter Task der Prozess zur Verwaltung einer logischen Einheit (im typischen Fall eine Gruppe) zu verstehen ist.

Zwei Task-Typen stehen zur Verfügung: normale Tasks und die "Realtime - Tasks".

Normale Tasks
Multitasking gründet sich auf einem kooperativen Algorithmus, der auf Prioritäten basiert. Dies gewährleistet, daß alle Tasks zyklisch ausgeführt werden und dass deren Priorität geändert werden kann. Der Planungsalgorithmus sieht die Ausführung einer Anweisung pro aktivem Task ("running"-Zustand) vor. Jedem Task ist eine Priorität zugeordnet, die mittels der Anweisung SETPRIORITYLEVEL zugewiesen wird. Die Priorität wird durch eine ganze Zahl zwischen 0 (größte Priorität) und 255 (kleinste Priorität) bestimmt. Von den Tasks mit Priorität 0 (Null) wird eine Anweisung pro Planungszyklus ausgeführt; von den Tasks mit Priorität 1 wird eine Anweisung alle zwei Planungszyklen ausgeführt, usw. bis zu den Tasks mit Priorität 255, von denen eine Anweisung alle 256 Planungszyklen ausgeführt wird.

Die Ausführung normaler Tasks ist asynchron im Verhältnis zur Aktualisierungsfrequenz der Achsen. Dies bedeutet, dass nicht gewährleistet werden kann, daß eine GPL-Funktion in der Zeitdauer zwischen zwei Aktualisierungen des Achsenzustands vollendet wird.

Ein Task wird vom Namen der GPL-Funktion bestimmt, bei dem seine Ausführung beginnt.
Die Ausführung eines Tasks beginnt in folgenden Fällen:
- automatisch bei der Initialisierung des Systems: Hauptfunktion der gruppenübergreifenden Gruppe und Autorun-Funktionen.
- infolge der Ausführung einer STARTTASK-Anweisung.
- infolge des manuellen Starts in der grafischen Schnittstelle von Albatros.

Jeder Task ist durch einen internen Zustand gekennzeichnet:

AUSFÜHRUNG	der Task ist in Ausführung begriffen
STILLSTAND	der Task steht still
UNTERBROCHEN	der Task ist vom Debug unterbrochen

Die Hierarchie der Tasks ist in einer Baumstruktur organisiert. Jeder Task wird von einem anderen geschaffen, dessen Tochter er ist, was bedeutet, dass am Ende des Mutter-Tasks auch alle Tochter-Tasks beendet werden.
Die Höchstzahl gleichzeitig in Ausführung begriffener Tasks beträgt 500.
Es ist zu betrachten dass eine relativ hohe Anzahl von Task in Ausführung die Senkung der Geschwindigkeit einbringt, mit der jeder einzelner Task ausgeführt wird.
Sollte es vorausgesetzt werden, dass der realisierende Anwendungsbereich den Gebrauch einer höhere als 200 Anzahl von Task einschließt, wird es nötig sein, eine passende Hardware wie die **Cn2128-Module** anzuwenden.

Realtime - Tasks
Realtime - Tasks unterscheiden sich von den vorhergehenden dadurch, dass sie weder einem Planungsmechanismus unterliegen noch eine zugeordnete Priorität haben, sondern bei jeder Aktualisierung des Achsenzustands (Achsen - Realtime) vollständig ausgeführt werden.
Es ist absolut notwendig, daß die Ausführung dieser Tasks innerhalb einer bestimmten Zeit beendet wird, da die Ausführung der vorher beschriebenen GPL-Tasks während der Ausführung der Realtime - Tasks stillsteht.
Das System prüft die Ausführungsdauer der Realtime - Tasks und ruft bei Überschreitung der zulässigen Höchstzeit einen Systemfehler hervor.
Es ist also von der Schaffung unendlicher Zyklen (z.B. mittels GOTO-Anweisungen) innerhalb dieser Tasks abzuraten, zumal solche Zyklen überflüssig sind, weil die Ausführung des Codes bei jeder Achsen - Realtime von vorne beginnt.
Um überlange Ausführungszeiten zu vermeiden, unterstehen die Realtime - Tasks einigen Einschränkungen bei der Verwendung von GPL-Anweisungen. Die Anweisungen, deren Gebrauch nicht zulässig ist, sind die bei Interrupt nicht verwendbaren Anweisungen.

Der Einsatz von Realtime - Tasks empfiehlt sich nur für diejenigen Tätigkeiten, die notwendigerweise synchron zur Aktualisierung der Achsenmaße auszuführen sind. Für die meisten Steuertätigkeiten ist der Gebrauch von normalen Tasks angebrachter.

Realtime - Tasks werden mittels der Anweisung STARTREALTIMETASK gestartet und können mittels der Anweisung ENDREALTIMETASK unterbrochen werden. Es können höchstens 256 Realtime - Tasks gleichzeitig aktiviert werden.
Die Vererbung-Funktionierung gilt nicht mehr: wenn der Task, der ein Realtime-Task gestartet hat, endet, wird der letzte in Ausführung bleiben.
Die im Realtime - Task erklärten lokalen Variablen werden nur bei dem Taskanlauf gebraucht und danach behalten sie den Wert der letzten Ausführung.

Realtime - Tasks sind nicht von den Zuständen normaler Tasks gekennzeichnet. Ein Realtime - Task kann dem Debug unterzogen werden, aber das System deklassiert den Task in diesem Fall über die gesamte Debug-Dauer automatisch zu einem "normalen Task".

Tritt in einem Realtime - Task ein Systemfehler auf, wird er zu einem normalen Task deklassiert und zum Stillstand gebracht, um ihn einer Debug-Analyse zu unterziehen.

10.1.8 Kommunikationen

Die Kommunikation zwischen GPL und der Außenwelt erfolgt auf drei verschiedene Weisen:
- SEND / RECEIVE
- Serielle Kommunikation
- IPC

Send / Receive
Die Anweisungen SEND und RECEIVE implementieren einen an der Nachricht orientierten Kommunikationsmechanismus.
Die Kommunikation kann entweder innerhalb eines Moduls erfolgen (kaum vorteilhaft) oder zwischen mehreren Modulen einer Linie oder zwischen den Modulen und dem Supervisor von Albatros oder mittels OLE-Anwendungen.
Die Funktionsweise ähnelt der der Post; jeder Nachricht sind ein Empfänger, ein Identifizierer der gesendeten Information (oder Anfrage), die Information selbst und Nebeninformationen zugeordnet. Albatros sorgt für die Sammlung und Verteilung der Informationen und gibt in manchen Fällen direkt die erfragten Informationen.
Diese Kommunikationsweise wird normalerweise zum Senden von Bearbeitungsprogrammen zwischen dem Supervisor und den Kontrollmodulen verwendet sowie zur Synchronisierung der Tätigkeit der Maschinen in einer Linie und als Schnittstelle zu externen Anwendungen (OLE Server).

Serielle Kommunikation
Die GPL-Sprache enthält einige Anweisungen, z.B. COMREAD und COMWRITE, die dem Senden und Empfangen von Daten mittels der seriellen Ports der Steuerung dienen. Es ist also möglich, die Steuerung an externe Vorrichtungen wie Inverter, Terminals oder PLCs anzuschließen. Angemessen verwendet, erlauben diese Anweisungen Protokolle zur seriellen Kommunikation wie MODBUS-RTU usw. zu implementieren.

IPC
IPC oder Inter Process Communication stellt eine Kommunikationsweise zwischen Prozessen dar. Insbesondere erlaubt diese Methode die Bestimmung eines zwischen zwei oder mehr Prozessen gemeinsam benutzten Speicherbereichs, der zum Austausch von Informationen verwendet werden kann.
IPC kommt normalerweise zum Einsatz, wenn die Leistungen der Schnittstelle OLE von Albatros nicht angebracht sind.
Auf GPL-Seite wird die Kommunikation mittels IPC durch die Anweisungen SENDIPC, WAITIPC und TESTIPC implementiert. Bei dem lokalem Modul können hingegen die externen Prozesse auf die API von RTX zurückgreifen oder auf die Komponente COM **gplipc2.dll** von TPA, die deren Gebrauch vereinfacht.

Bei dem Fernmodul verwenden die im Supervisor laufenden Prozesse, die COM **gplipc2net.dll** Komponente verwenden.

Für weitere Informationen nehmen Sie bitte mit TPA Kontakt auf.

10.1.9 Bei der Programmierung zu verwendende Variablen

Die meisten Anweisungen sind so beschaffen, dass sie zu verschiedenen Variablentypen (CHAR, INTEGER, FLOAT, DOUBLE) passen. Jede Anweisung hat jedoch einen bevorzugten Variablentyp, auf den sie optimiert ist. Um während der Ausführung des GPL-Codes bessere Leistungen zu erhalten, ist es ratsam, die Variablentypen zu verwenden, die bei der Beschreibung jeder Anweisung empfohlen werden. Allgemein empfiehlt es sich, folgende Tabelle zu beachten, in der den am meisten gebrauchten Größen bei der Programmierung die jeweils optimalen Typen zugeordnet sind:

Größe	***Typ***
Maß	*double*
Geschwindigkeit	*float*
Zeit	*double*
Zähler	*integer*
Port-Wert / Port-Flag	*integer*
Timeout	*double*
analoger Ein- / Ausgang	*float*

Richtungskosinusse	*double*
String-Kontrollzeichen	*char*
Beschleunigungen / Verzögerungen	*integer*

10.1.10 Achsen

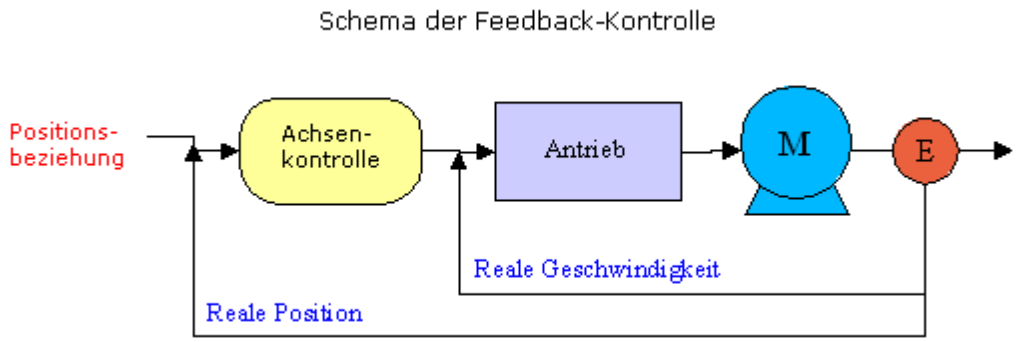
Als "Achse" wird normalerweise ein elektromechanisches System bezeichnet, dessen Zweck in der gesteuerten Bewegung eines Elements einer Werkzeugmaschine liegt.
Das System kann anhand der Bauteile, aus denen es zusammengesetzt ist, beschrieben werden. Letztere wiederum können aufgrund ihrer technologischen Eigenschaften unterteilt werden.
Es sind also mechanische Bauteile vorhanden, darunter:

- Das Gestell
- Führungen
- Lager
- Schraube + Endlosschraube

deren Aufgabe es ist, die ins Spiel tretenden Kräfte auszugleichen, die Reibung zu vermindern, die Bewegung von rotierend in umsetzend zu verwandeln, usw.
Es sind auch elektrische und elektronische Bauteile vorhanden, darunter:

- Der Motor
- Endschalter
- Der Encoder
- Tacho-Dynamo

deren Aufgabe es ist, die zur Bewegung notwendige Leistung zuzuführen und den Zustand des Systems abzulesen.
Diese Elemente sind so untereinander verbunden, dass Bewegungen auf gesteuerte Weise ausgeführt werden können.



Schema einer Achse mit Feedback-Kontrolle

Aufgabe der Numerischen Steuerung ist die Kontrolle der Position und der Bewegung einer Achse.

Die Bewegung einer Achse kann in 5 Phasen unterteilt werden:

Beschleunigung Anfangsphase, während der die Geschwindigkeit der Achse langsam zunimmt bis zum Erreichen der gewünschten Geschwindigkeit

Konstant Zwischenphase, während der die Geschwindigkeit der Achse konstant bleibt (diese Phase kann auch ausbleiben, wenn die zurückzulegende Strecke kleiner ist als die während der Beschleunigung und Verzögerung zurückgelegte Strecke)

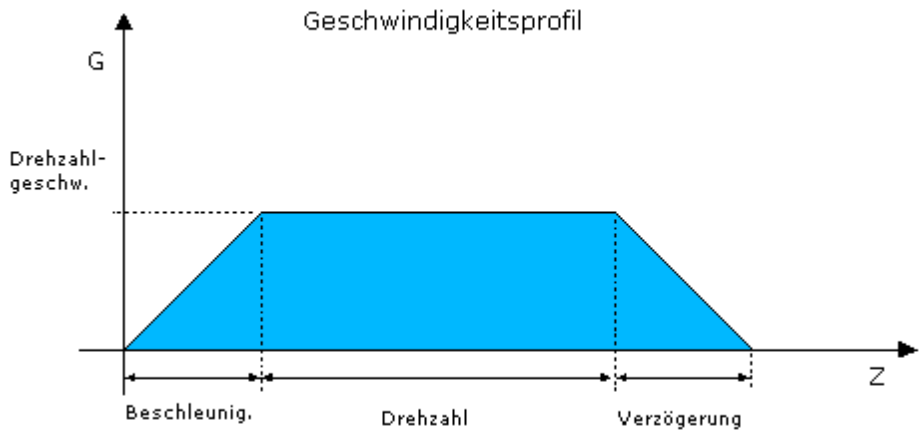
Verzögerung Phase, während der die Geschwindigkeit der Achse langsam bis 0 reduziert wird

Toleranz Phase, in der abgewartet wird, dass der Loop-Fehler bis zum Wert sinkt, der in der Konfiguration als "Positionierungstoleranz" angegeben ist

Maß Ende der Bewegung

Am Ende der Bewegung hat sich die Achse innerhalb eines "Positionierungstoleranz" genannten Intervalls zu positionieren (definiert die zulässige Toleranz bei der Positionierung der Achse). Erfolgt dies

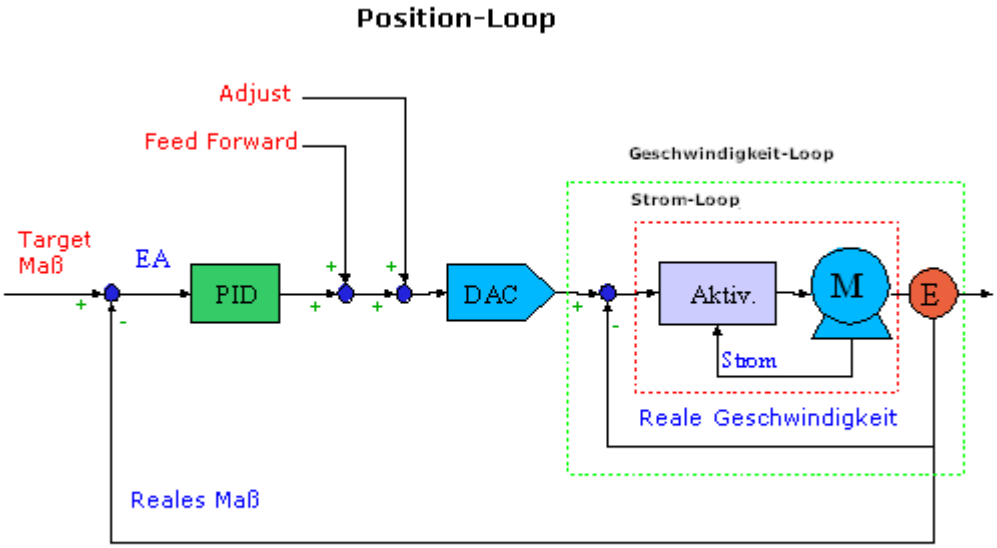
nicht innerhalb 5 Sekunden ab dem vorgesehenen Ende der Bewegung, tritt ein Systemfehler "Bewegung nicht beendet" auf.



Geschwindigkeitsprofil der Achsenbewegung

Die numerische Steuerung berechnet für jede Bewegung ein Geschwindigkeitsprofil wie das in Abbildung 9.2, dann berechnet sie die Zielmaße durch Unterteilung des Geschwindigkeitsprofils in Zeitintervalle, die der Aktualisierungszeit der Achse entsprechen, und durch Berechnung der Fläche jedes Intervalls. Die Fläche stellt die Maßzunahme dar, die die Achse im jeweiligen Zeitintervall zu erreichen hat, um das oben dargestellte Geschwindigkeitsprofil zu beachten.

Die Steuerung der Achse wird durch eine PID-Kontrolle implementiert, die für die "Schließung des Loop-Rings" sorgt, d.h. dem Antrieb einen Geschwindigkeitsbezug bietet, der aufgrund des zu erreichenden Maßes (Target-Maß) und des tatsächlichen, vom Encoder abgelesenen Maßes berechnet ist. Der Unterschied zwischen tatsächlichem Maß und Zielmaß heißt ***Loop-Fehler***.



Schema der Achsenkontrolle von Albatros

10.1.11 Linearitätskorrektoren

Die Tabelle der Linearitätskorrektoren für eine Achse wird als eine Matrix angesehen, deren Name *Gruppe.Name.Untergruppe.Name.AchseName#*correctors oder *GruppeName.AchseName#*correctors lautet und die in allen Anweisungen verwendet werden kann, in denen auf Matrizen, Matrixelemente und Matrixzeilen zugegriffen wird.

Die Anzahl der Matrixspalten entspricht der Anzahl der im Fenster Linearitätskorrektoren eingestellten Achsen in Achsenkonfigurationen. Die Autokorrektoren werden in der ersten Spalte eingefügt. Alle Korrekturwerte sind vom Typ Float. Die Gesamtzahl der Zeilen der Matrix kann mit Hilfe der Anweisung GPL-LASTELEM ermittelt werden.

Auf diese Matrizen kann sowohl im Lese- als auch im Schreibmodus zugegriffen werden, und wenn sie geändert werden, werden ihre Werte sofort für die Dimensionskorrektur verwendet, nur wenn die Korrektur aktiviert wurde.

Beispiel:

```
Function ReadCorr
local i as integer
local j as integer
local row as integer
local column as integer
local firstvalue as float

; Ablesen des ersten selbstkorrigierenden Wertes der AX-Achse
firstvalue = X.Ax#correctors[1][1]
; Anzahl der im Fenster des Linearitätskorrektors eingestellten Achsen
setval  3  column
lastelem X.AX#correctors  row
; Erhöhen aller Korrekturen um einen konstanten Wert
for i 1 row
     for j 1 column
        X.Ax#correctors[i][j] = X.Ax#correctors[i][j] + 0.025
      next
next

fret
```

10.1.12 Verwaltung von Nachrichten in fremder Sprache

Wie im Kapitel Aufbau des Systems beschrieben, unterstützt Albatros die Anzeige von Text-Nachrichten in verschiedenen Sprachen.

Diese Unterstützung erfolgt durch TpaLangs, das ein von Albatros unabhängiges Programm zur Verwaltung von Nachrichtenarchiven ist. Dieses Programm unterstützt die Übersetzung der Nachrichten in den verschiedenen Sprachen.

Text im Zusammenhang mit Zyklusfehlern und Nachrichten
Ein besonderer Texttyp, der normalerweise von Albatros angezeigt wird, sind die vom GPL-Code hervorgerufenen Nachrichten und Zyklusfehler.
Diese werden normalerweise vom Entwickler des GPL-Codes beim Abfassen desselben definiert. Um die Arbeit des Programmierers zu vereinfachen, erlaubt der GPL-Editor den Text einer Nachricht direkt aus Albatros einzufügen, ohne also TpaLangs benutzen zu müssen.
Eine weitere Möglichkeit zur Verwaltung von Nachrichten in fremder Sprache ist durch die GPL-Anweisung DEFMSG gegeben.

10.1.13 Verwaltung der Systemfehler

Wenn ein Systemfehler auftritt (siehe Kapitel ***Systemfehler->Einführung in die Systemfehler)*** beendet die Steuereinheit normalerweise sämtliche Tasks: mit der Verwaltung der Systemfehler kann die Beendigung der Tasks, für die sie aktiviert wurde, vermeidet werden.
Die von fault, stack underflow und stack overflow generierten Systemfehler werden direkt von der Steuereinheit verwaltet, ohne die Systemverwaltungs-Funktion aufzurufen: die Task wird auf HOLD gestellt.

Fehlerwaltungsfunktion

Im GPL-Code muss eine oder mehrere Funktionen für die Systemfehleranalyse und für die folgende Definition von entsprechenden Handlungen definiert sein, um die Maschine in Sicherheitszustand zu bringen. Die aufzurufende Funktion wird als Parameter der ONERRSYS-GPL-Anweisung weitergegeben (siehe Kapitel ***GPL-Sprache->Anweisungen->Verwaltung des Ablaufs->ONERRSYS).***

Bei Auftreten eines Systemfehlers, wird die Task, die den Fehler generiert hat, auf HOLD gestellt. Wenn die autorun-Tasks Systemfehler generieren, werden sie nur dann wieder gestartet, wenn der Systemfehler kein FAULT ist.
Wenn der Systemfehler ohne Task-Nummer generiert wird, wird die laufende Task auf HOLD gestellt.

10.2 Sondernfunktionen

10.2.1 Achsenbewegung anpassen

Die graphische Schnittstelle des Systems Albatros sieht die Möglichkeit vor, die Achsen manuell zu bewegen. Sie gibt der Einstellung der Achsen selbst einen graphischen Träger.
Die manuelle Bewegung wird durch das Fenster der Achsenbewegung durchgeführt. Die Einstellung kann durch die Einstellungsfenster durchgeführt werden. Sie können beide von der Diagnostik-Funktion und von den synoptischen Fenstern wieder eingespeichert worden.
In beiden Fällen wird die Achsenbewegung von einer Gesamtheit GPL-Funktionen geführt, deren Ausführung für den Benutzer unsichtbar bleibt.
Das System hat eine vorherbestimmte Menge dieser Funktionen, die für die meisten Fällen passend sind. Doch ist es nötig, dass sie dem Kunden angepasst werden, zum Beispiel um Beschränkungen in Achsenbewegungen einzuführen, die zum Maschinen-Zustand verbunden sind oder um Hilfs-Vorrichtungen zu führen, wie die Achsen-Bremsen.

Die Anpassung am Kunden erfolgt durch die Schaffung von zwei GPL-Funktionen für jede Achse: einer für die manuelle Bewegung und einer für die Einstellung. Diese Funktionen sind optional, darum werden sie verwendet, nur wenn sie vom System gefunden werden. Andernfalls werden die Standard-Funktionen verwendet. Außerdem ist auch eine partielle Anpassung am Kunden der Bewegungsfunktionen vorgesehen.

Manuelle Bewegung

Die am Kunden angepassten Funktionen von *manuelle Bewegung* müssen die folgenden Eigenschaften erfolgen:

- Die Funktion muss der selben Gruppe oder Untergruppe gehören, an der die Achse der Funktion gehört.
- Der Name der Funktion muss **MoveAx#Achse_*Name*** sein. Hier muss Name_Achse mit dem Namen ersetzt worden, der in der Konfiguration bestimmt ist. Zum Beispiel wird für die X-Achse der Name der Funktion: MoveAx#X sein.
- Die Funktion muss die folgenden Parameter vorsehen:

1. ***Gebrauchter Befehl***: Es kann eine Bewegung an einem absoluten Maß, einer inkrementalen Bewegung, einem Stop usw. sein. Die Befehle werden von einer ganzen Zahl bestimmt. Der GPL-Compiler sieht die vorherbestimmten Konstanten vor, die mit den folgenden Befehlen verbunden sind:

_MOVAXABS	Bewegung mit absolutem Maß
_MOVAXINC	Bewegung mit inkrementalem Maß
_MOVAXSET	Einführung des Maßes
_MOVAXFREE	Einführung des Free-Zustands
_MOVAXNORMAL	Einführung des Normal-Zustands
_MOVAXEND	Wiederherstellung des Achsen-Zustands am Ende der Bewegung (es ist nicht nötig die Achse in Stillstand zu setzen)

2. **Ergebnis**: Der resultierte Parameter dient dem System um zu wissen, ob der gefragte Befehl von der am Kunden angepassten Funktion geführt wird. Wenn die Aktion nicht geführt wird, wird die entsprechende Standard-Funktion verwendet. Dieser Parameter ist also ein Rückkehrwert, den die am Kunden angepasste Funktion einsetzen muss und zu diesem Zweck ist das Resultat als ein Parameter, der als Bezug gegeben wird (Array eines einzigen Elements), bestimmt.
3. ***Geschwindigkeit***: Sie ist bedeutend nur wenn der gefragte Befehl eine Bewegung ist und die Geschwindigkeit einführt.
4. ***Maß***: Ist bedeutend nur für die Bewegungsbefehle und die Maß-Einstellung.

Beispiel einer Funktion einer am Kunden angepassten Bewegung:

```
Function MoveAx#X
   param action as integer
   param result[1] as integer
   param speed as float
   param position as double

   setval      1,result[1]

   select action
   case _MOVAXEND
         fcall EndMovement
   case _MOVAXABS
         fcall AbsMovement X, speed, position
   case _MOVAXINC
         fcall IncMovement X, speed, position
   case _MOVAXSET
         fcall PositionSet X, position
   case _MOVAXFREE
         fcall FreeAxis
   case _MOVAXNORMAL
         fcall NormalAxis
   case else
         call  Unknown
   endselect

   fret

Unknown:
   setval      0, result[1]
   ret
```

Die Funktionen EndMovement, AbsMovement, usw. (Namen sind nicht verbindlich) müssen die am Kunden angepasste Verwaltung der erfragten Befehle implementieren. Um die Arbeit des Programmierers zu erleichtern, werden hier die Standard Bewegungsfunktionen angegeben, die als Vorbild für die Entwicklung der am Kunden angepassten Funktionen dienen können.

Einstellung

Die am Kunden angepassten *Einstellungsfunktionen* müssen folgende Eigenschaften folgen:

- Die Funktion muss derselben Gruppe oder Untergruppe gehören, an der die Bezugsachse gehört.
- Der Name der Funktion muss **CalibAx*#Achse_Name*** sein. Name_Achse muss mit dem Namen der Achse, die in der Konfiguration bestimmt wird, ersetzt werden. Zum Beispiel wird für die X-Achse der Name: CalibAx#X sein.
- Die Funktion muss die folgenden Parameter vorsehen:

1. ***Gebrauchter Befehl***: kann eine Schnell-Bewegung (Punkt-Punkt-Bewegung) oder eine interpolierte Bewegung sein
2. **Ergebnis**: der resultierte Parameter dient dem System um zu wissen, ob der gefragte Befehl von der am Kunden angepassten Funktion geführt wird. Wenn die Aktion nicht geführt wird, wird die entsprechende Standard-Funktion verwendet
3. ***Geschwindigkeit***: Bewegungsgeschwindigkeit während der Einstellung
4. ***Positives Maß***: Positives Maß der alternierenden Einstellung-Bewegung
5. ***Negatives Maß:*** Negatives Maß der alternierenden Einstellung-Bewegung
6. ***Wartungszeit:*** Wartungszeit zwischen einer Bewegung und der nächsten

ANMERKUNG: In einigen Fällen verursachen die Befehle, die in der Einstellungsfenster durchgeführt werden, die Durchführung der Funktion der Achsenbewegung. Zum Beispiel wird am Ende der

Einstellungsbewegung (wenn der Stop-Knopf gedrückt wird) ein Vorgang für die Wiederherstellung der Achse durchgeführt, für die die am Kunden angepasste Funktion der Achsenbewegung mit dem Befehlparameter am_MOVAXEND gerufen wird. In derselben Weise wird die Funktion der Achsenbewegung mit dem am_MOVAXSET eingeführten Befehlparameter angerufen, wenn das Achsenmaß vom Einstellungsfenster geändert wird.

Beispiel der am Kunden angepasste Einstellungsfunktion:

```
Function CalibAx#X
   param action as integer
   param result[1] as integer
   param speed as float
   param PosPosition as double
   param NegPosition as double
   param WaitTime as float

   setval      1,result[1]

   select action
   case _CALAXPP
         fcall PPCalibration X, speed, PosPosition, NegPosition,
WaitTime
   case _CALAXINT
         fcall IntCalibration X, speed, PosPosition, NegPosition,
WaitTime
   case else
         call  Unknown
   endselect

   fret

Unknown:
   setval      0, result[1]
   ret
```

Die PPCalibration- und IntCalibration-Funktionen (Namen sind nicht verbindlich) müssen die am Kunden angepasste Verwaltung der erfragten Befehle implementieren. Um die Arbeit des Programmierers zu erleichtern, werden hier die Standard-Einstellung_Funktionen angegeben, die als Vorbild für die Entwicklung der am Kunden angepassten Funktionen dienen können.

Interaktion mit dem Fenster der Achsenbewegung im Handbetrieb

Die Funktionen für die Interaktion mit dem Fenster der manuellen Bewegung müssen die folgenden Spezifikationen einhalten:

- Die Funktion muss derselben Gruppe oder Untergruppe angehören, der die Achse, auf die sich bezieht, angehört.
- Der Name der Funktion muss **MoveAx*#Achse_Name#Aktion*** sein, wobei Name_Achse durch den bei der Konfiguration festgelegten Namen der Achse ersetzt wird und Tätigkeit eine der folgenden Definitionen annehmen kann:

OPEN	Zeigt an, dass der Benutzer soeben das Fenster der Achsenbewegung geöffnet hat
CLOSE	Zeigt an, dass der Benutzer das Fenster der Achsenbewegung schließt
ACTIVE	Signalisiert, dass das Fenster der Achsenbewegung eingeschaltet ist
INACTIVE	Signalisiert, dass das Fenster der Achsenbewegung nicht eingeschaltet ist
JOG	Zeigt an, dass eine Bewegung für die Verschiebung der Laufzeit Verwaltungen vom Benutzer eingestellt wurde
STEP	Zeigt an, dass eine Bewegung mit vorbestimmter Steigungsverschiebung eingestellt ist
ABSOLUTE	Zeigt an, dass eine Bewegung mit Verschiebung in festgelegter Höhe eingestellt ist

Wenn, zum Beispiel, das Fenster der Achsenbewegung für die X-Achse geöffnet worden ist, dann wird die Funktion mit dem Namen MoveAx#X#Open genannt.

Änderungen am Fenster der Achsenbewegung im Handbetrieb
Es können bis zu 4 Tasten zum Fenster der Achsenbewegung hinzugefügt werden. Die GPL-Funktionen mit unveränderlichem Namen MoveAx#AchseName#BUTTONText müssen in derselben Gruppe oder Untergruppe, in der die betroffene Achse definiert wurde, definiert werden. NameAchse bezeichnet den Namen der betroffenen Achse und Text bezeichnet den Text, der auf der Taste erscheint. Der Text kann das Zeichen '&' zur Einführung eines Tastenbeschleunigers enthalten. Wenn Text mit einer Zahl zwischen 1 und 4 beginnt, wird die Zahl als die Position, in der die Taste im Fenster der Achsenbewegung eingefügt wird, angesehen. Der Text der Taste kann übersetzt werden, indem eine DEFMSG in die Gruppe, in der sich die Achse befindet, eingefügt wird, die MOVEAX#BUTTONtesto als Identifizierer besitzt. Das Drücken der persönlich gestalteten Taste führt zur Durchführung der verbundenen GPL-Funktion. Es wird weder ein Abwarten des Durchführungsendes noch irgendeine Überprüfung beim Durchführungsbeginn der Funktion ausgeführt.

10.2.2 Standardmäßige Bewegung und Einstellungsfunktionen

Die nachfolgenden Funktionen sind die standardmäßigen Funktionen, die vom Fenster der Achsenbewegung und vom Einstellung-Fenster verwendet werden.
Die Funktionen ändern nach dem Achsentyp, der verwendet wird: Berechnung-, mit Schrittmotor**,** usw.
Diese Funktionen können angepasst werden.

Standardmäßige manuelle Funktionen der Bewegung

Bewegung mit absolutem Maß

```
; für Achsen mit Schrittmotor
Function AbsMovement
   param axisname as axis
   param speed as float
   param position as double

   ifstill     axisname goto move
   fret
move:
   setvel      axisname, speed
   movabs      axisname, position
   waitstill   axisname
   fret

; für alle andere Achsentypen
Function AbsMovement
   param axisname as axis
   param speed as float
   param position as double

   iftarget    axisname goto move
   ifstill     axisname goto move
   fret
move:
   setvel      axisname, speed
   movabs      axisname, position
   waitstill   axisname
   fret
```

Inkrementale Bewegung

```
; für Achsen mit Schrittmotor
Function IncMovement
   param axisname as axis
   param speed as float
   param position as double

   ifstill     axisname goto move
   fret
move:
   setvel      axisname, speed
   movinc      axisname, position
   waitstill   axisname
   fret
```

```
; für alle andere Achsentype
Function IncMovement
  param axisname as axis
  param speed as float
  param position as double

  iftarget    axisname goto move
  ifstill     axisname goto move
  fret
move:
  setvel      axisname, speed
  movinc      axisname, position
  waitstill   axisname
  fret
```

Maßeinstellung

```
; für Berechnungsachsen
Function PositionSet
  param axisname as axis
  param position as double

  setquote    axisname, position
  fret

; für Achsen mit Schrittmotor
Function PositionSet
  param axisname as axis
  param position as double

  ifstill     axisname goto set
  fret
set:
  setquote    axisname, position
  fret

; für alle andere Achsentype
Function PositionSet
  param axisname as axis
  param position as double

  iftarget    axisname goto set
  ifstill     axisname goto set
  fret
set:
  setquote    axisname, position
  fret
```

Einstellung des Free-Zustandes

```
Function FreeAxis
  param axisname as axis

  free        axisname
  fret
```

Einstellung des normalen Zustandes

```
Function NormalAxis
  param axisname as axis

  normal      axisname
  fret
```

Standardmäßige Einstellungsfunktionen

Schnelleinstellung

```
; für Achsen mit Schrittmotor or stepper motor axes
Function PPCalibration
  param axisname as axis
  param speed as float
  param PosPosition as double
  param NegPosition as double
  param WaitTime as float

  setvel      axisname, speed
loop:
  movabs      axisname, PosPosition
  waitstill   axisname
  delay       WaitTime
  movabs      axisname, NegPosition
  waitstill   axisname
  delay       WaitTime
  goto        loop
  fret

; für alle andere Achsentype
Function PPCalibration
  param axisname as axis
  param speed as float
  param PosPosition as double
  param NegPosition as double
  param WaitTime as float

  setvel      axisname, speed
loop:
  movabs      axisname, PosPosition
  waitstill   axisname
  ifquotet    axisname,<>,PosPosition goto exit
  delay       WaitTime
  movabs      axisname, NegPosition
  waitstill   axisname
  ifquotet    axisname,<>,NegPosition goto exit
  delay       WaitTime
  goto        loop
exit:
  fret
```

Einstellung in Interpolation

```
Function IntCalibration
  param axisname as axis
  param speed as float
  param PosPosition as double
  param NegPosition as double
  param WaitTime as float

  setveli     axisname, speed
loop:
  linearabs   axisname, PosPosition
  waitstill   axisname
  ifquotet    axisname,<>,PosPosition goto exit
  delay       WaitTime
  linearabs   axisname, NegPosition
  waitstill   axisname
  ifquotet    axisname,<>,NegPosition goto exit
  delay       WaitTime
  goto        loop
exit:
  fret
```

10.2.3 Funktion OnUIEnd#

Die Funktion "OnUIEnd#" wird, wenn vorhanden, von Albatros vor dem Schluss jedes Tasks eines Moduls ausgeführt. Sie muss in der Funktionsdatei der gruppenübergreifenden Gruppe definiert werden. Die Höchstausführungszeit der Funktion "OnUIEnd#" ist auf 2 Sekunden festgelegt; danach wird Albatros alle Tasks beenden.

Die maximale Wartezeit, die notwendig ist, damit diese Funktion die Ausführung beendet, ist im Abschnitt [Albatros], unter Timeout.OnUIEnd=*Wert* konfigurierbar sein, in dem *Wert* in Millisekunden ist und darf größer als 60000 nicht sein.

10.2.4 Funktion OnUIPlugged#

Die Funktion "OnUInplugged#" ist ausgeführt, wenn es notwendig ist, zu wissen, zum Beispiel während der Zündung einer Anlage, wenn Albatros mit dem Fernmodul mitgeteilt hat.
Diese Funktion muss innerhalb der Intergruppe definiert werden.

10.2.5 Funktion OnUIUnPlugged#

Die Funktion "OnUIUnPlugged#" ist ausgeführt vor dem Schluss von Albatros (dann bevor Albatros trennt die Verbindung mit einem Modul). Diese Funktion muss innerhalb der Intergruppe definiert werden. Albatros führt diese Funktion innerhalb max. 2 Sekunden aus. Während dieser Zeit, werden folgende Eintragungen gelesen:

- Zyklusfehler
- Systemfehler
- Nachrichten

Am Ende dieser Ausführung kommt Albatros zum Schluss.

Die maximale Wartezeit, die notwendig ist, damit diese Funktion die Ausführung beendet, ist im Abschnitt [Albatros], unter Timeout.OnUIEnd=*Wert* konfigurierbar sein, in dem *Wert* in Millisekunden ist und darf größer als 60000 nicht sein.

10.3 Anweisungen

10.3.1 Konventionen

Die folgenden Seiten sind in der Form von Datenblättern organisiert und beschreiben von jeder Anweisung:

- die Syntax
- eine Beschreibung der Argumente: Datentyp und zulässige Werte
- eine Beschreibung der Funktionsweise
- eventuelle Hinweise
- eventuelle Beispiele

Anweisungen desselben Typs sind gruppiert worden, um das Lernen und Nachschlagen zu vereinfachen.

10.3.2 Anweisungstypen der GPL-Sprache

Die Sprache besteht aus Anweisungen, die folgendermaßen gruppiert werden können:

Anweisungen zur Verwaltung der Ein-/Ausgänge

GETFEED	liest den Feed Rate Override
INPANALOG	liest einen analogen Eingang ab
INPFLAGPORT	liest einen Flag-Port ab
INPPORT	liest einen digitalen Port ab
MULTIINPPORT	liest bis 4 Ausgang-Ports ab
MULTIOUTPORT	legt bis 4 Ausgang-Ports fest
MULTIRESETFLAG	legt mehrere Flags auf 0 fest
MULTIRESETOUT	legt mehrere Ausgänge auf 0 fest
MULTISETFLAG	legt mehrere Flags auf 1 fest
MULTISETOUT	legt mehrere Ausgänge fest

MULTIWAITFLAG	wartet den Zustand eines Flag-Bits oder Flag-Switch ab
MULTIWAITINPUT	wartet den Zustand von mehreren Eingänge ab
OUTANALOG	ändert einen analogen Ausgang
OUTFLAGPORT	ändert einen Flag-Port
OUTPORT	ändert einen digitalen Port
RESETFLAG	legt einen Flag auf 0 fest
RESETOUT	legt einen Ausgang auf 0 fest
SETFLAG	legt einen Flag auf 1 fest
SETOUT	legt einen Ausgang auf 1 fest
WAITFLAG	wartet den Zustand eines Flag-Bits oder Flag-Switch ab
WAITINPUT	wartet den Zustand eines Eingangs ab
WAITPERSISTINPUT	wartet den fortdauernden Zustand eines Eingangs ab

Anweisungen zur Verwaltung der Achsen

CHAIN	verkettet eine Achse mit einer anderen
CIRCABS	absolute kreisförmige Interpolation
CIRCINC	inkrementale kreisförmige Interpolation
CIRCLE	führt einen Kreis aus
COORDIN	koordinierte Achsenbewegung
DISABLECORRECTION	deaktiviert die Linearitätskorrektur für die angegebene Achse
EMERGENCYSTOP	forciert den Not-Halt
ENABLECORRECTION	deaktiviert die Linearitätskorrektur für die angegebene Achse
ENDMOV	beendet die Bewegung einer Achse
FASTREAD	schnelles Ablesen der Achsenmaße
FREE	setzt die Achse in den Free-Zustand
HELICABS	absolute heilkoidale Interpolation
HELICINC	inkrementale helikoidale Interpolation
JERKCONTROL	aktiviert oder deaktiviert die Kontrolle bei den Interpolationsbewegungen
JERKSMOOTH	verbindet mit stetigen Beschleunigung und Geschwindigkeit die Geschwindigkeitsprofile der Achsen bei der Konturierungsbewegungen
LINEARABS	absolute lineare und helikoidale Interpolation
LINEARINC	inkrementale lineare Interpolation
MOVABS	Achsenbewegung im absoluten Modus
MOVINC	Achsenbewegung im inkrementalen Modus
MULTIABS	lineare mehrachsige Interpolation im Absolut-Modus
MULTIINC	lineare mehrachsige Interpolation in inkrementalem Modus
NORMAL	entfernt den Free-Zustand von einer Achse
RESRIFLOC	stellt die anfänglichen Bezüge wieder her
SETINDEXINTERP	verbindet eine Achse mit einer Variable zum Zählen der ausgeführten Interpolationsblöcke
SETLABELINTERP	verbindet eine Achse mit einer Variable zum Bezeichnen eines Verzetzungsblöcks
SETPFLY	Eil-Nulleinstellung
SETPFLYCHAINSTRAT	kontrolliert wie die Slave-Achse im Vergleich zu einer Setfly-Anweisung auf dem Master funktioniert
SETPZERO	Nulleinstellung auf Nullkerbe
SETPZEROCHAINSTRAT	kontrolliert wie die Slave-Achse im Vergleich zu einer Setpzero-Anweisung auf dem Master funktioniert
SETQUOTE	legt das Maß fest
SETQUOTECHAINSTRAT	steuert die Slave-Achse bei einer Setquote-Instruktion auf dem Master
SETRIFLOC	legt räumliche Bezüge fest
SETTOLERANCE	legt die Toleranzwerte für die lineare Interpolation fest
START	nimmt die Bewegung einer Achse wieder auf
STARTINTERP	erzwingt den Beginn einer Interpolation
STOP	unterbricht die Bewegung einer Achse
SWITCHENC	ermöglicht den Encoder einer Achse mit dem einer anderen Achse auszutauschen
WAITACC	Beschleunigung der Achse abwarten
WAITCOLL	Wartezeit bis die Achse den Ausgangspunkt erreicht hat, von welchem eine eventl. Kollisionsgefahr festgestellt werden könnte
WAITDEC	Verzögerung der Achse abwarten
WAITREG	Konstante Geschwindigkeit der Achse abwarten
WAITSTILL	Übereinstimmung zwischen Endmaß und Ziel-Maß abwarten
WAITTARGET	wartet dass die Achse am Ziel ist
WAITWIN	wartet dass die Achse in der Toleranz ist

Anweisungen zur Verwaltung der Achsenparameter Schreiben/Lesen

DEVICEID	schreibt die logische mit einer Vorrichtung verbundene Adresse
GETAXIS	liest eine oder mehrere Größen einer Achse ab

Punkt-Punkt Bewegung

SETACC	legt die Beschleunigung fest
SETDEC	legt die Verzögerung fest
SETDERIV	legt den Koeffizienten der Differential-Komponente fest
SETFEED	legt den Punkt-Punkt Feed Rate fest
SETFEEDF	legt das Feed - Forward fest
SETFEEDFA	setzt das Feed - Forward der Beschleunigung fest
SETINTEG	legt den Koeffizienten der Integral-Komponente fest
SETMULTIFEED	legt den Prozentwert von Feed Rate override der angegebenen Achsen fest
SETPROP	legt den Koeffizienten der proportionalen Aktion fest
SETSLOPE	legt den Rampentyp in den Schnellbewegungen fest
SETVEL	legt die Geschwindigkeit fest

Interpolierte Bewegung

LOOKAHEAD	legt den Interpolation-Lookahead fest
SETACCI	legt die Beschleunigung bei der Interpolation fest
SETACCLIMIT	aktiviert und deaktiviert die automatische Berechnung der konstanten Geschwindigkeit bei der Interpolation
SETACCSTRATEGY	sortiert Beschleunigungstyp aus
SETAXPARTYPE	wechselt den verwendeten Satz der Achsenparameter
SETCONTORNATURE	legt den Profilierwinkel fest
SETDECI	legt die Verzögerung bei der Interpolation fest
SETDERIVI	legt setzt den Koeffizienten der Differential-Komponente bei der Interpolation fest
SETFEEDFAI	legt das Feed Forward der Beschleunigung in der Interpolation fest
SETFEEDFI	legt das Feed Forward in der Interpolation fest
SETFEEDI	legt das Feed Rate in der Interpolation fest
SETINTEGI	legt den Koeffizienten der Integral-Komponente bei der Interpolation fest
SETPROPI	legt den Koeffizienten der proportionalen Aktion der Interpolation fest
SETSLOPEI	legt den Rampentyp in den Interpolationsbewegungen fest
SETSLOWPARAM	Bearbeitet die Parameter, die zum Berechnen der Geschwindigkeitsabnahme notwendig sind, im Falle die funktionale Geschwindigkeitsabnahme bei der Konturierung aktiv ist
SETVELI	legt die Interpolationsgeschwindigkeit fest
SETVELILIMIT	legt die einzelnen Geschwindigkeitskomponenten der spezifizierten Achse fest

Koordinierte Bewegung

SETFEEDCOORD	legt den Prozentwert der maximal augenblicklichen Änderung des Feedrates der Achse.
SETOFFSET	legt ein Maß-Offset fest

Verkettete Bewegung

RATIO	dient der Einstellung des Verkettungsverhältnisses zwischen einer Slave- und der zugehörigen Master-Achse
SETDYNRATIO	ändert bei der Bewegung der Master-Achse das Verkettungsverhältnis auf dynamische Weise.

Allgemeine Parameter

DYNLIMIT	aktiviert oder deaktiviert die dynamische Prüfung der Überschreitung der Achsenbegrenzung.
ENABLESTARTCONTROL	aktiviert und stellt das Timeoutfest, das den fehlenden Anfang oder den plötzlichen Halt der Achse untersucht
NOTCHFILTER	dient der Einstellung des Frequenz-Grenzwerts für den Notchfilter der angegebenen Achse
RESLIMNEG	deaktiviert die negative Grenze der Achse
RESLIMPOS	deaktiviert die positive Grenze der Achse
SETADJUST	legt den Adjust-Wert einer Achse fest
SETBACKLASH	verringert oder entfernt die Effekte der mechanischen Spielen auf der Trajektorie der Achse
SETBIGWINFACTOR	ändert den Multiplikationsfaktor zur Berechnung des großen Fensters auf der gewünschten Achse
SETDEADBAND	legt die Parameter der minimale Spannung für die angegebene Achse fest
SETENCLIMIT	legt die Anschlussgrenzen des falschen Encoders
SETINDEXEN	aktiviert oder deaktiviert die Annullierung der Quote an der Null-Kerbe auf der angegebenen Achse
SETINTEGTIME	führt die Anzahl der Muster der Ring-Fehler ein, die für die Berechnung der Integral-Komponente verwendet werden
SETIRMPP	legt die Rampenstartgeschwindigkeit fest
SETLIMNEG	legt die negative Grenze der Achse fest
SETLIMPOS	legt die positive Grenze der Achse fest
SETMAXER	legt den höchst zulässigen Gewinn-Wert fest
SETMAXERNEG	legt den höchst zulässigen Gewinn-Wert (negative Richtung) fest
SETMAXERPOS	setzt den höchst zulässigen Gewinn-Wert (positive Richtung)
SETMAXERTYPE	legt den Testtyp auf dem Servofehler fest
SETPHASESINV	aktiviert oder deaktiviert auf der angegebenen Achse die Inversion der Phasen
SETREFINV	aktiviert oder deaktiviert auf der angegebenen Achse die Inversion der Geschwindigkeitsreferenz
SETRESOLUTION	wechselt die Auflösung einer Achse

Anweisungen zur Verwaltung der Zähler

DECOUNTER	dekrementiert einen Zähler
INCOUNTER	inkrementiert einen Zähler
SETCOUNTER	legt einen Zähler fest

Anweisungen zur Verwaltung der Timer

HOLDTIMER	hält einen Timer
SETTIMER	stellt einen Timer ein
STARTTIMER	startet den Timer

Anweisungen zur Verwaltung der Variablen, Vektoren und Matrizen

CLEAR	Löschen einer Variablen, eines Vektors, einer Matrix
FIND	Suche nach einem Element
FINDB	Suche nach einem Element in auf zunehmender Weise geordneten Vektor oder Matrix
LASTELEM	Suche nach dem letzten Element eines Vektors oder einer Matrix, zunehmend angeordnet
LOCAL	Deklaration einer lokalen Variablen, eines lokalen Vektors, einer lokalen Matrix
MOVEMAT	kopiert eine Matrix-Zeile in eine andere Matrix
PARAM	Deklaration eines Parameters einer Funktion
SETVAL	ändert eine Variable
SORT	Ordnung eines Vektors oder einer Matrix

Anweisungen zur Verwaltung der Zeichenfolge

ADDSTRING	verkettet zwei Zeichenfolge
CONTROLCHAR	legt ein Kontrollzeichen in einer Zeichenfolgevariablen
LEFT	entnimmt die ersten Zeichen
LEN	liest die Länge einer Zeichenfolge ab
MID	entnimmt einige Zeichen
RIGHT	entnimmt die letzten Zeichen
SEARCH	sucht in einer Zeichenfolge
SETSTRING	ändert eine Zeichenfolgevariablen
STR	konvertiert von Nummer in Zeichenfolge
VAL	konvertiert von Zeichenfolge in Nummer

Anweisungen zur Verwaltung der Kommunikationen

CLEARRECEIVE	entleert die zu erfüllende RECEIVE-Liste
COMCLEARRXBUFFER	leert den Empfang-Pufferspeicher eines seriellen Ports
COMCLOSE	schließt einen seriellen Port
COMGETERROR	liest den Fehlercode ab
COMGETRXCOUNT	liest die Anzahl im Empfangs-Pufferspeicher vorhandener Byte
COMOPEN	öffnet einen seriellen Port
COMREAD	liest vom seriellen Port
COMREADSTRING	liest einen String vom seriellen Port
COMWRITE	schreibt auf den seriellen Port
COMWRITESTRING	schreibt einen String auf den seriellen Port
RECEIVE	Datenempfang von außen
SEND	Datenversand nach außen
SENDIPC	sendet eine IPC-Information
WAITIPC	wartet eine IPC-Information ab
WAITRECEIVE	Datenempfang von außen mit Wartezeit

Anweisungen zur Verwaltung der Mathematik

ABS	absoluter Wert
ADD	Summe
AND	binäres AND
ARCCOS	Arkuskosinus
ARCSIN	Arkussinus
ARCTAN	Arkustangent
COS	Kosinus
DIV	Division
EXP	exponentiell
EXPR	löscht mathematische Ausdrücke
LOG	natürlicher Logarithmus
LOGDEC	Logarithmus zur Basis 10
MOD	Modul
MUL	Multiplikation
NOT	binäres NOT
OR	binäres OR
RANDOM	erstellt eine zufällige Zahl
RESETBIT	legt ein Bit auf 0 fest
ROUND	Rundung
SETBIT	legt ein Bit auf 1 fest
SHIFTL	zyklische Verschiebung der Bits nach links
SHIFTR	zyklische Verschiebung der Bits nach rechts
SIN	Sinus
SQR	Quadratwurzel
SUB	Subtraktion
TAN	Tangente
TRUNC	Abschneiden
XOR	binäres XOR

Anweisungen zur Multitask-Verwaltung

ENDMAIL	weist auf das Ende der Ausführung eines Befehls hin
ENDREALTIMETASK	beendet einen Realtime-Task
ENDTASK	beendet einen Task
GETPRIORITYLEVEL	liest das Prioritätsniveau des laufenden Tasks ab
GETREALTIME	gibt die seit Beginn des Achsen-Realtime vergangene Zeit zurück
GETREALTIMECOUNT	gibt die Anzahl vergangener Realtime an
HOLDTASK	unterbricht die Ausführung eines Tasks
RESUMETASK	gibt die Anzahl vergangener Realtime an
SENDMAIL	sendet einen Befehl an die Mailbox 'Mail'
SETPRIORITYLEVEL	setzt die Priorität für den laufenden Task
STARTREALTIMETASK	startet einen Realtime-Task
STARTTASK	startet die Ausführung eines Tasks
STOPTASK	hält die Ausführung eines Tasks und die Bewegung der zugeordneten Achsen an
WAITMAIL	erhält einen Befehl von der Mailbox 'mail'
WAITTASK	wartet das Ende der Ausführung eines Tasks ab

Anweisungen zur Verwaltung des Ablaufs

CALL	Aufruf eines Unterprogramms
DELONFLAG	deaktiviert die Verwaltung von Notfällen an Flag Bit oder Flag Switch
DELONINPUT	deaktiviert die Verwaltung von Notfällen am digitalen Eingang
ENDREP	Ende der Block-Wiederholung mittels REPEAT
FCALL	Aufruf einer Funktion
FOR	Erweiterung von REPEAT
FRET	Rückkehr vom Aufruf einer Funktion
GOTO	Sprung zum Etikett
IF	prüft eine Variable
IFACC	prüft, ob sich die Achse beschleunigt
IFAND	prüft einen AND-Vorgang
IFBIT	prüft ein Bit
IFBLACKBOX	prüft, ob die Aufnahme der Aktivität der logischen Vorrichtungen aktiv ist.
IFCHANGEVEL	prüft, ob die Achse die Geschwindigkeit wechselt
IFCOUNTER	prüft einen Zähler
IFDEC	prüft, ob die Achse verzögert wird.
IFDIR	prüft die Achsenrichtung
IFERRAN	prüft den Loop-Fehler
IFERROR	prüft den aktiven Zyklusfehler.
IFFLAG	prüft ein Flag
IFINPUT	prüft einen Eingang
IFMESSAGE	die aktive Nachricht
IFOR	prüft einen OR-Vorgang
IFOUTPUT	prüft einen Ausgang
IFQUOTER	prüft ein reelles Maß
IFQUOTET	prüft ein Target-Maß
IFRECEIVED	prüft den Datenempfang
IFREG	prüft, ob die Geschwindigkeit der Achse konstant ist
IFSAME	prüft, daß sich zwei Argumente auf dieselbe Größe beziehen
IFSTILL	prüft, ob die Achse stillsteht
IFSTR	prüft einen String
IFTARGET	prüft, ob die Achse am Ziel ist
IFTASKHOLD	prüft, ob die parallele Funktion stillsteht
IFTASKRUN	prüft, ob die parallele Funktion gerade ausgeführt wird
IFTIMER	prüft einen Timer
IFVALUE	prüft eine Variable
IFVEL	prüft die Achsengeschwindigkeit
IFWIN	prüft, ob die Achse innerhalb der Toleranz liegt
IFXOR	prüft einen XOR-Vorgang
NEXT	Ende der Block-Wiederholung mittels FOR
ONERRSYS	setzt den Aufruf einer Funktion bei Systemfehler
ONFLAG	Notfall an Flag Bit oder Flag Switch
ONINPUT	Notfall an digitalem Eingang
REPEAT	Wiederholung eines Anweisungsblocks
RET	Rückkehr vom Unterprogramm
SELECT	Mehrfach-Auswahl mit Sprung
TESTIPC	prüft, ob eine IPC-Information vorhanden ist

TESTMAIL	Test und Empfang eines Befehls

Verschiedene Anweisungen

CLEARERRORS	löscht alle Zyklusfehler des Moduls
CLEARMESSAGES	löscht alle Nachrichten des Moduls
DEFMSG	definiert eine Gruppennachricht
DELAY	hält die laufende Funktion eine bestimmte Zeit an
DELERROR	löscht einen vorherigen Zyklusfehler
DELMESSAGE	löscht eine vorherige Nachricht
ERROR	sendet dem PC einen Zyklusfehler
IFDEF/ELSEDEF/ENDDEF	Test zur Kompilierung mit Bedingungen
MESSAGE	sendet dem PC eine Nachricht
SYSFAULT	deaktiviert das SYSOK-Signal
SYSOK	aktiviert das SYSOK-Signal
TYPEOF	Argumenttyp
WATCHDOG	aktiviert, aktualisiert, deaktiviert den Watchdog von GPL auf dem TMSWD-Hardware-Modul.

Anweisungen zur MECHATROLINK-II-Verwaltung

MECCOMMAND	sendet dem Achsenbetrieb einen Befehl
MECGETPARAM	liest einen Parameter der angegebenen Achse
MECGETSTATUS	liest die STATUS, ALARAM und IO_MON-Werte
MECSETPARAM	schreibt einen Parameter in der angegebenen Achse

Anweisungen zur Verwaltung der standardmäßigen Feldbusse

AXCONTROL	legt einen Wert für ControlWord fest
AXSTATUS	ergibt den in StatusWord enthaltenen Wert
CNBYDEVICE	ergibt die Zahl einer Platine und von CN einer Vorrichtung
READDICTIONARY	liest den Inhalt eines Objekts des Wörterbuchs
WRITEDICTIONARY	schreibt den Inhalt eines Objektes des Wörterbuchs

Anweisung zur EtherCAT-Verwaltung

ACTIVATEMODE	legt einen Betriebsmodus fest
ECATGETREGISTER	ergibt den Inhalt eines ESC-Registers (EtherCAT Slave Controller)
ECATSETREGISTER	schreibt den Inhalt eines ESC-Registers (EtherCAT Slave Controller)
GETPDO	ergibt ein Objekt innerhalb eines EtherCAT-PDOs
SETEOE	aktiviert oder deaktiviert den Sniffer
SETPDO	legt das Objekt innerhalb eines EtherCAT-PDOs fest

Anweisungen zur Verwaltung des CAN-Busses

GETCNSTATE	ergibt den Zustand des NMT-Protokolls für einen Knoten einer Platine.
GETSDOERROR	ergibt den letzten aufgetretenen Fehler
GETMNSTATE	ergibt den Zustand des Protokolls NMT für den Master-Knoten der Platine.
RECEIVEPDO	schreibt den Inhalt eines asynchronen PDOs
SENDPDO	schreibt den Inhalt eines asynchronen PDOs
SETNMTSTATE	legt den Zustand des NMT-Protokolls für den Master-Knoten der Platine fest.

Anweisungen zur Simulation

DISABLE	deaktiviert einen oder mehrere Achsen
DISABLEFORCEDINPUT	deaktiviert die Forcierung der Eingänge

ENABLE	aktiviert eine oder mehrern Achsen
ENABLEFORCEDINPUT	aktiviert die Forcierung der Eingänge
RESETFORCEDINPUT	forciert eine Eingang auf OFF
SETFORCEDANALOG	forciert einen analogen Eingang
SETFORCEDINPUT	forciert eine Eingang auf ON
SETFORCEDPORT	forciert einen Eingang-Port

Anweisungen zur "BlackBox"-Funktion

ENDBLACKBOX	beendet die Aufnahmefunktion
PAUSEBLACKBOX	unterbricht die Aufnahmefunktion
STARTBLACKBOX	startet die Aufnahmefunktion

Anweisungen zur ISO-Verwaltung

ISOG0	legt die Schnellbewegung fest
ISOG1	legt die interpolierte Bewegung fest
ISOG9	legt den Zwangshalt der Bewegung fest
ISOG90	legt die Interpretation der Maße als absolute Maße fest
ISOG91	legt die Interpretation der Maße als relative Maße fest
ISOG93	legt die Interpretation der Geschwindigkeit als Inverse der Zeit fest
ISOG94	stellt die Interpretation der Geschwindigkeit als Messeinheit pro Minute fest
ISOG216	definiert die Matrizen für die Parametrisierung der Maschine
ISOG217	beschreibt die physikalischen Achsen und die virtuellen Achsen, die die Maschinen gestalten
ISOM2	befreit die Achsen von der ISO-Bewegung
ISOM6	wählt die Indices der Matrizen der Parametrisierung aus
ISOSETPARAM	legt einige Parameter fest, die das Fließvermögen der interpolierten ISO-Bewegung bezeichnen
KINEMATICEXPR	legt die einzelnen Ausdrücke der inversen und direkten Kinematik fest

10.3.3 Ein-/Ausgänge

GETFEED

Syntax

GETFEED **Variable**

Argumente

Variable feed rate

Beschreibung

Diese Anweisung kopiert den von der Platine des Fern-E/As abgelesenen Feed rate-Wert in die angegebene **Variable**. Der Feedrate-Wert liegt zwischen 0 und 100 und ist ein Prozentwert.
Wirkt auf einen analogen Eingang ein, der in der Konfiguration nicht sichtbar ist.
Alle Feed Rate steuernende Tpa-Platinen verfügen über einen dedizierten Stecker.

INPANALOG

Syntax

INPANALOG **NameAnalogEin, Variable**

Argumente

NameAnalogEin Name der Vorrichtung des Typs analoger Eingang
Variable Variable

Beschreibung

Diese Anweisung kopiert den Wert des analogen Eingangs **NameAnalogEin** in die angegebene **Variable**.

INPFLAGPORT

Syntax

INPFLAGPORT **FlagPortName, Variable**

Argumente

FlagPortName Name der Vorrichtung des Typs Flag-Port
Variable Variable

Beschreibung

Diese Anweisung kopiert den Zustand des Flag-Ports **FlagPortName** in die angegebene **Variable**. Der Flag-Port wird als Bit-Maske angesehen. Jedem Flag-Port ist ein Bit zugeordnet. Befindet sich ein Flag im Zustand "ON", wird das entsprechende Bit auf 1 gesetzt.

INPPORT

Syntax

INPPORT **PortName, Variable**

Argumente

PortName Name der Vorrichtung des Typs Eingangs-Port
Variable Integer oder char-Variable

Beschreibung

Diese Anweisung kopiert den Zustand des Eingangs-Ports **PortName** in die angegebene **Variable**. Der Eingangs-Port wird als Bit-Maske angesehen. Befindet sich ein Eingang des Ports im Zustand "ON", wird das entsprechende Bit auf 1 gesetzt.

MULTIINPORT

Syntax

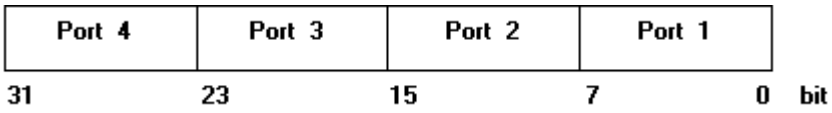
MULTIINPPORT **Port1[,...,Port4], Variable**

Argumente

Port1 versorgt die Bits von 0 zu 7
Port2 versorgt die Bits von 8 a 15
Port3 versorgt die Bits von 16 a 23
Port4 versorgt die Bits von 24 a 31
Variable Diese Integer-Variable empfängt die Eingang-Ports

Beschreibung

Vier Eingangsports werden gleichzeitig gelesen und in einer **Variablen** geschrieben**.** Der **Wert** wird in vier Ausgangsports gleichzeitig geschrieben. Die angegebenen Ports werden automar geschrieben: Das garantiert, dass die Ports innerhalb des selben Realtime geschrieben werden. Port1 entspricht dem kleinerem Byte, Port4 entspricht dem größerem Byte.



MULTIOUTPORT

Syntax

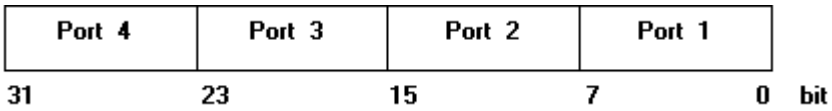
MULTIOUTPORT **Wert, Portname1[,...,Portname4]**

Argumente

Wert Zahl oder Integer-Variable zu schreiben in den Ausgangsports
Portname1 empfängt die Bits von 0 zu 7
Portname2 empfängt die Bits von 8 zu 15
Portname3 empfängt die Bits von 16 zu 23
Portname4 empfängt die Bits von 24 zu 31

Beschreibung

Der **Wert** wird in vier Ausgangsports gleichzeitig geschrieben. Die angegebenen Ports werden automar geschrieben, und garantieren, dass die Ports innerhalb des selben Realtime geschrieben werden. Falls **portname2**, **portname3**, **portname4** nicht spezifiziert sind, liegt der Wert ihren Bytes bei 0.



MULTIRESETFLAG

Syntax

MULTIRESETFLAG **Maske, FlagName1[, ..., FlagName32]**

Argumente

Maske Maske der betroffenen Flags - Konstante oder Variable
FlagName1 Name der Vorrichtung des Typs flag

Beschreibung

Diese Anweisung deaktiviert, d.h. setzt auf "OFF" all die Flags **FlagName** (1÷32), deren Bit im Argument **Maske** auf 1 ist.
Das Bit 0 der **Maske** (das mit dem niedrigsten Stellenwert) entspricht **FlagName1**.

MULTIRESETOUT

Syntax

MULTIRESETOUT **Maske, AusgangName1[, ..., AusgangName32]**

Argumente

Maske Maske der betroffenen Ausgans - Konstante oder Variable
AusgangName1 Name der Vorrichtung des Typs Ausgang

Beschreibung

Diese Anweisung deaktiviert all die Ausgänge **AusgangName** (1÷32), deren Bit im Argument **Maske** auf 1 ist. Das Bit 0 der **Maske** (das mit dem niedrigsten Stellenwert) entspricht **AusgangName1**.

MULTISETFLAG

Syntax

MULTISETFLAG **Maske, FlagName1[, ..., FlagName32]**

Argumente

Maske Maske der betroffenen Flags - Konstante oder Variable
FlagName1 Name der Vorrichtung des Typs Flag

Beschreibung

Diese Anweisung aktiviert, d.h. setzt auf "ON" all die Flags **FlagName** (1÷32), deren Bit im Argument **Maske** auf 1 ist. Das Bit 0 der **Maske** (das mit dem niedrigsten Stellenwert) entspricht **FlagName1**.

MULTISETOUT

Syntax

MULTISETOUT **Maske, AusgangName1[, ..., AusgangName32]**

Argumente

Maske Maske der betroffenen Ausgans - Konstante oder Variable
AusgangName1 Name der Vorrichtung des Typs Ausgang

Beschreibung

Diese Anweisung aktiviert all die Ausgänge **AusgangName** (1÷32), deren Bit im Argument **Maske** auf 1 ist.
Das Bit 0 der **Maske** (das mit dem niedrigsten Stellenwert) entspricht **AusgangName1**. Handelt es sich um einen monostabilen Ausgang, wird er automatisch nach einem festen Timeout von 200 Millisekunden deaktiviert.

MULTIWAITFLAG

Syntax

MULTIWAITFLAG	**Maske, Flag1[, ..., Flag32], Zustand [, Timeout [, GOTO Etikett]]**
MULTIWAITFLAG	**Maske, Flag1[, ..., Flag32], Zustand [, Timeout [, CALL UnterprogrammName]]**
MULTIWAITFLAG	**Maske, Flag1[, ..., Flag32], Zustand [, Timeout [, FunktionsName]]**

Argumente

Maske	Konstante oder Variable. Maske der betroffenen Flags
Flag1[,...Flag32]	Name der Vorrichtung des Typs Flag
Zustand	vordefinierte Konstante. Die zulässigen Werte lauten: **ON** Flag-Zustand aktiviert **OFF** Flag-Zustand deaktiviert
Timeout	Konstante oder Variable. Es ist die maximale Wartezeit
Etikett	Sprung-Etikett (GOTO)
UnterprogrammName	Etikett eines Unterprogramms (CALL)
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung wartet ab bis sich die angegebenen Flags von **Flag1...Flag32** in dem Zustand befinden, der vom Parameter **Zustand** (ON/OFF) angegeben wird.
Von allen Flags werden diejenigen geprüft, deren Bit im Argument **Maske** auf ON ist. Das Bit 0 des Arguments **Maske** (das mit dem kleinsten Stellenwert) entspricht dem von **Flag1** definiertem Bit, Bit 1 entspricht dem von **Flag2** definierten Bit und so weiter bis zum Bit, das von **Flag32** definiert wird.
Der Parameter **Timeout** dient der Einstellung eines anderen Timeouts als des Default-Werts von einer Sekunde.
Ist ein **Etikett** oder **UnterprogrammName** oder **FunktionsName** vorhanden, springt das Programm bei Ablauf des Timeouts zum **Etikett** oder ruft **UnterprogrammName** oder **FunktionsName** auf.

MULTIWAITINPUT

Syntax

MULTIWAITINPUT	**Maske, Eingang1[, ..., Eingang32], Zustand [, Timeout [, GOTO Etikett]]**
MULTIWAITINPUT	**Maske, Eingang1[, ..., Eingang32], Zustand [, Timeout [, CALL UnterprogrammName]]**
MULTIWAITINPUT	**Maske, Eingang1[, ..., Eingang32], Zustand [, Timeout [, FunktionsName]]**

Argumente

Maske	Konstante oder Variable. Maske der betroffenen Eingangs
Eingang1[,...Eingang32	Eingang Name
Zustand	vordefinierte Konstante. Die zulässigen Werte lauten: **ON** Eingang-Zustand aktiviert **OFF** Eingang-Zustand deaktiviert
Timeout	Konstante oder Variable. Es ist die maximale Wartezeit
Etikett	Sprung-Etikett (GOTO)
UnterprogrammName	Etikett eines Unterprogramms (CALL)
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung wartet ab bis sich die angegebenen Eingänge von **Eingang1...Eingang32** im Zustand befinden, der vom Parameter **Zustand** (ON/OFF) angegeben wird.
Von allen Eingängen werden diejenigen geprüft, deren Bit im Argument **Maske** auf ON ist. Das Bit 0 des Arguments **Maske** (das mit dem niedrigsten Stellenwert) entspricht dem von **Eingang1** definiertem Bit, Bit 1 entspricht dem von **Eingang2** definierten Bit und so weiter bis zum Bit, das von **Eingang32** definiert wird.
Sind keine optionalen Argumente angegeben, erscheint eine Sekunde nach Ausführungsbeginn der Anweisung (Default-Timeout) automatisch folgende Parameternachricht: "Wait input ON/OFF". Der Name des Eingangs, auf den hingewiesen wird, ist der Name des ersten aktivierten Eingangs, der den Zustand noch nicht erfüllt hat. Wenn der Parameter **Timeout** vorhanden ist, erscheint oben beschriebene Anzeige nach Ablauf des eingestellten Timeouts. Ist nach Ablauf des Timeouts die gewünschte Bedingung erfüllt, erscheint automatisch eine Parameternachricht zum Löschen der vorher gesendeten Nachricht.

Ist ein **Etikett** oder **UnterprogrammName** oder **FunktionsName** vorhanden, springt das Programm bei Ablauf des Timeouts zum **Etikett** oder ruft **UnterprogrammName** oder **FunktionsName** auf.

OUTANALOG

Syntax

OUTANALOG **NameAnalogAus, Wert**

Argumente

NameAnalogAus Name einer Vorrichtung des Typs analoger Ausgang oder Achse
Wert Konstante oder Variable

Beschreibung

Diese Anweisung setzt den analogen Ausgang oder die durch NameAnalogAus ausgedrückte Achse auf die vom Wert angegebene Spannung.

OUTFLAGPORT

Syntax

OUTFLAGPORT **FlagPortName, Wert**

Argumente

FlagPortName Name einer Vorrichtung des Typs Flag-Port
Wert Konstante oder Variable

Beschreibung

Diese Anweisung kopiert den **Wert** in den Flag-Port **FlagPortName**.
Der Parameter **Wert** wird als Bit-Maske angesehen. Jedem Bit ist ein Flag-Port zugeordnet. Ist der Wert eines Bits 1, wird das entsprechende Flag in den Zustand "ON" gesetzt.

OUTPORT

Syntax

OUTPORT **PortName, Wert**

Argumente

PortName Name einer Vorrichtung des Typs Ausgangs-Port
Wert Konstante oder Variable, Integer oder Char

Beschreibung

Diese Anweisung kopiert den **Wert** in den Ausgangs-Port **PortName**.
Der Ausgangs-Port wird als Bit-Maske angesehen. Ist der Wert eines Bits 1, wird der entsprechende Ausgang in den Zustand "ON" gesetzt.

RESETFLAG

Syntax

RESETFLAG **FlagName**

Argumente

FlagName Name der Vorrichtung des Typs Flag

Beschreibung

Diese Anweisung deaktiviert (setzt auf OFF) das Flag **FlagName**.

RESETOUT

Syntax

RESETOUT **AusgangName**

Argumente

AusgangName Name einer Vorrichtung des Typs digitaler Ausgang

Beschreibung

Diese Anweisung deaktiviert (setzt auf OFF) den Ausgang **AusgangName**.

SETFLAG

Syntax

SETFLAG	**FlagName**

Argumente

FlagName	Name einer Vorrichtung des Typs Flag

Beschreibung

Diese Anweisung aktiviert (setzt auf ON) das Flag **FlagName**.

SETOUT

Syntax

SETOUT	**AusgangName**

Argumente

AusgangName	Name der Vorrichtung des Typs digitaler Ausgang

Beschreibung

Diese Anweisung aktiviert (setzt auf ON) den Ausgang **AusgangName**.
Handelt es sich um einen als monostabil konfigurierten Ausgang, wird er automatisch nach einem festen Timeout von 200 Millisekunden deaktiviert.

WAITFLAG

Syntax

WAITFLAG	**FlagName, Zustand [, Timeout [, GOTO Etikett]]**
WAITFLAG	**FlagName, Zustand [, Timeout [, CALL UnterprogrammName]]**
WAITFLAG	**FlagName, Zustand [, Timeout [, FunktionsName]]**

Argumente

FlagName	Name einer Flag-Vorrichtung
Zustand	vordefinierte Konstante. Die zulässigen Werte lauten: **ON** Zustand Flag aktiviert **OFF** Zustand Flag deaktiviert
Timeout	Konstante oder Variable. Es ist die maximale Wartezeit
Etikett	Sprung-Etikett (GOTO)
UnterprogrammName	Etikett eines Unterprogramms (CALL)
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung wartet bis das Flag **FlagName** den vom Parameter **Zustand** (ON/OFF) angegebenen Zustand annimmt.
Ist von den optionalen Argumenten nur **Timeout** vorhanden, erscheint bei Ablauf des Timeouts der Zyklusfehler "Flag **FlagName** wartet **Zustand** ab".
Wird die Bedingung nach Ablauf des Timeouts erfüllt, wird der vorher für den Task gesendete Zyklusfehler automatisch gelöscht.
Ist ein **Etikett** oder **UnterprogrammName** oder **FunktionsName** vorhanden, springt das Programm bei Ablauf des Timeouts zum **Etikett** oder ruft **UnterprogrammName** oder **FunktionsName** auf, ohne dass automatisch eine Anzeige erscheint.

Anmerkung

Um das Abwarten eines Flags während eines Arbeitszyklus zu vermeiden, wird die Einstellung eines Timeouts nahegelegt.

WAITINPUT

Syntax

WAITINPUT	**EingangName, Zustand [, Timeout [, GOTO Etikett]]**
WAITINPUT	**EingangName, Zustand [, Timeout [, CALL UnterprogrammName]]**
WAITINPUT	**EingangName, Zustand [, Timeout [, FunktionsName]]**

Argumente

EingangName	Eingang Name
Zustand	vordefinierte Konstante. Die zulässigen Werte lauten: **- ON** Zustand des Eingangs: aktiv

	- OFF Zustand des Eingangs: nicht aktiv
Timeout	Konstante oder Variable. Es ist die maximale Wartezeit
Etikett	Sprung-Etikett (GOTO)
UnterprogrammName	Etikett eines Unterprogramms (CALL)
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung wartet bis der Eingang **EingangName** den vom Parameter **Zustand** (ON/OFF) angegebenen Zustand annimmt.
Sind keine optionalen Argumenten angegeben, erscheint 20 Sekunden nach Ausführungsbeginn der Anweisung automatisch der Zyklusfehler "Digitaler Eingang **EingangName** wartet **Zustand** ab".
Ist von den optionalen Argumenten nur **Timeout** vorhanden, erscheint oben genannter Zyklusfehler bei Ablauf desselben.
Wird die Bedingung nach Ablauf des **Timeouts** erfüllt, wird automatisch der vorher für den Task gesendete Zyklusfehler gelöscht.
Ist ein **Etikett** oder **UnterprogrammName** oder **FunktionsName** vorhanden, springt das Programm bei Ablauf des Timeouts zum **Etikett** oder ruft **UnterprogrammName** oder **FunktionsName** auf, ohne daß automatisch eine Anzeige erscheint.

Anmerkung

Um das Abwarten eines Eingangssignals während eines Arbeitszyklus zu vermeiden, wird die Einstellung eines kürzeren Timeouts als der Default-Wert (20 Sekunden) nahegelegt.

Beispiel

Achsen-Nullstellung-Routine

WAITPERSISTINPUT

Syntax

WAITPERSISTINPUT	**EingangName, Zustand, timepersist [, Timeout [, GOTO Etikett]]**
WAITPERSISTINPUT	**EingangName, Zustand, timepersist [, Timeout [, CALL UnterprogrammName]]**
WAITPERSISTINPUT	**EingangName, Zustand, timepersist [, Timeout [, FunktionsName]]**

Argumente

EingangName	Name der Vorrichtung vom Typ digitaler Eingang
Zustand	vordefinierte Konstante. Die zulässigen Werte lauten: **- ON** Zustand Eingang aktif **- OFF** Zustand Eingang nicht aktif
timepersist	Konstante oder Variable
Timeout	Konstante oder Variable. Es ist die maximale Wartezeit
Etikett	Sprung-Etikett (GOTO)
UnterprogrammName	Etikett eines Unterprogramms (CALL)
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung wartet bis der Eingang **EingangName** den vom Parameter **Zustand** (ON/OFF) angegebenen Zustand annimmt und die in **timepersist** angegebene Dauer (Maßeinheit: Sekunden) in dem Zustand bleibt.
Sind keine optionale Argumente angegeben, erscheint 20 Sekunden nach Ausführungsbeginn der Anweisung automatisch der Zyklusfehler: "Digitaler Eingang **EingangName** wartet **Zustand** ab".
Ist von den optionalen Argumenten nur **Timeout** vorhanden, erscheint oben genannter Zyklusfehler bei Ablauf desselben.
Wird die Bedingung nach Ablauf des **Timeouts** erfüllt, wird automatisch der vorher für den Task gesendete Zyklusfehler gelöscht.
Ist ein **Etikett** oder **UnterprogrammName** oder **FunktionsName** vorhanden, springt das Programm bei Ablauf des Timeouts zum **Etikett** oder ruft **UnterprogrammName** oder **FunktionsName** auf, ohne daß automatisch eine Anzeige erscheint.

Anmerkung

Um das Abwarten eines Eingangssignals während eines Arbeitszyklus zu vermeiden, wird die Einstellung eines kürzeren Timeouts als der Default-Wert (20 Sekunden) nahegelegt.

10.3.4 Achsen

CHAIN

Syntax

CHAIN	**Achse_Master, Achse_Slave1 [, ...Achse_Slave5]**

Argumente

Achse_Master	Name der Vorrichtung des Typs Achse, die als Master funktionieren wird
Achse_Slave1...Achse_Slave5	Name der Vorrichtung des Typs Achse, die als Slave funktionieren wird

Beschreibung

Nach Ausführung dieser Anweisung führen die **Achsen_Slave** (1÷5) Bewegungen aus, die durch das Verkettungsverhältnis mittels der Anweisung RATIO mit den Bewegungen der **Achse_Master** verbunden sind. Sowohl Punkt-Punkt- wie auch Interpolationsbewegungen werden verkettet sein.
Die **Achse_Slave1** ist kein optionaler Parameter, sondern ist immer zu definieren.
Um verkettet werden zu können, darf die Slave-Achse weder in einer Interpolation begriffen sein noch darf sie ihrerseits Master anderer Slaves sein.
Die Master-Achse darf ihrerseits nicht Slave anderer Achsen sein.
Die Verkettung kann sowohl erfolgen, wenn die Achsen am Maß stehen als auch während der Bewegung der Achsen.
Um die Verkettung der Achsen zu deaktivieren, ist es ausreichend, die Anweisung NORMAL an der Master-Achse auszuführen. Dieser Vorgang kann sowohl erfolgen, wenn die Achsen am Maß stehen als auch während der Bewegung der Achsen. Wenn die Verkettung während einer Bewegung der Achsen deaktiviert wird, führt die Slave-Achse eine Verzögerungsrampe aus und hält an.
Nicht mehr als 8 Master-Achsen können gleichzeitig definiert werden.
In dieser Anweisung können Achsen im Schrittbetrieb (Stepper) verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind. Jede Achse im Schrittbetrieb muss über einen reellen und nicht simulierten Encoder verfügen. Ansonsten wird der Systemfehler „4101 – Unpassende Verwaltung der Achse AchseName" erzeugt.
Siehe auch RATIO.

Beispiel

```
; Y-Achse mit X-Achse verketten
CHAIN           X, Y
; X-Achse bewegen.
; Y wiederholt die Bewegung von X
MOVINC          X, 100
```

CIRCABS

Syntax

CIRCABS	**[Etikett],Achse1, Maß1, Achse2, Maß2, Richtung, ±Radius [, Winkel]**

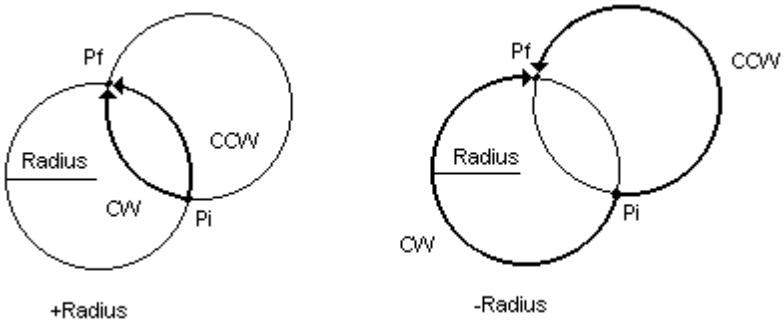
Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1, Achse2	Namen von Vorrichtungen des Typs Achse
Maß1,Maß2	Konstante oder Variable. Stellt das Maß der absoluten Versetzung dar
Richtung	Integer-Variable. Zur Angabe der Drehrichtung sind folgende Werte zulässig: **CW** im Uhrezeigersinn **CCW** gegen den Uhrseigersinn
Radius	Konstante oder Variable. Stellt den Radiuswert des Kreises dar
Winkel	Konstante oder Variable. Stellt den Startwinkel dar

Beschreibung

Zweiachsige, kreisförmige Interpolation mit *absoluter Versetzung*, die auf den programmierten Maßen **Maß1** und **Maß2** gründet.
Der Bogen ist vom Anfangspunkt (laufender Punkt), vom Endpunkt, vom Wert des **Radius** und von der **Richtung** der Bewegung bestimmt.
Der Zusatz des Vorzeichens zum **Radius** erlaubt den kleineren (+Radius) oder größeren Bogen (-Radius) zu wählen.

Im Sonderfall des Anfangsmaßes der Achse 1, das mit dem Endmaß **Maß1** übereinstimmt, und des Anfangsmaßes der Achse 2, das mit dem Endmaß **Maß2** übereinstimmt, hat man einen vollen Kreis: in diesem Fall ist das Argument **Winkel** mit derselben Bedeutung wie in der Anweisung CIRCLE zu deklarieren (unter der nachzusehen ist).
Der Parameter Winkel dient der eindeutigen Bestimmung der Kreismitte und hat dieselbe Bedeutung wie in der Anweisung CIRCLE. Er wird nur berücksichtigt, wenn vor Ausführung der Anweisung das **Maß1** und **Maß2** den laufenden Achsenmaßen übereinstimmen. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren.
In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind. In diesem Fall ist zu bedenken, dass sich der Begriff Interpolation auf die koordinierte Bewegung mehrerer Achsen bezieht, die aufgrund der Antriebsmethode der Achse einen diskreten Fehler aufweisen.



CIRCINC

Syntax

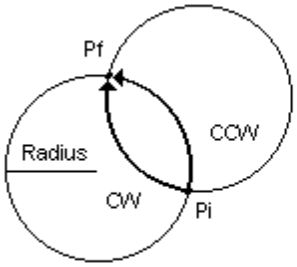
CIRCINC **[Etikett],Achse1, Maß1, Achse2, Maß2, Richtung, ±Radius [, Winkel]**

Argumente

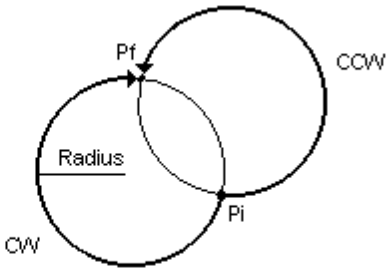
Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1, Achse2	Namen von Vorrichtungen des Typs Achse
Maß1,Maß2	Konstante oder Variable. Stellt das Maß der absoluten Versetzung dar
Richtung	Variable integer. Zur Angabe der Drehrichtung sind folgende Werte zulässig: **CW** im Uhrezeigersinn **CCW** gegen den Uhrseigersinn
Radius	Konstante oder Variable. Stellt den Radiuswert des Kreises dar
Winkel	Konstante oder Variable. Stellt den Startwinkel dar

Beschreibung

Zweiachsige, kreisförmige Interpolation mit *inkrementaler Versetzung*, die auf den programmierten Maßen **Maß1** und **Maß2** gründet.
Der Bogen ist vom Anfangspunkt (laufender Punkt), vom Endpunkt, vom Wert des **Radius** und von der **Richtung** der Bewegung bestimmt.
Der Zusatz des Vorzeichens zum **Radius** erlaubt den kleineren (+Radius) oder größeren Bogen (-Radius) zu wählen.
Im Sonderfall von Maß1 = Maß2 = 0, hat man einen vollen Kreis: in diesem Fall ist das Argument **Winkel** mit derselben Bedeutung wie in der Anweisung CIRCLE zu deklarieren (unter der nachzusehen ist).
Der Parameter Winkel dient der eindeutigen Bestimmung der Kreismitte und hat dieselbe Bedeutung wie in der Anweisung CIRCLE. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren.
In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind. In diesem Fall ist zu bedenken, dass sich der Begriff Interpolation auf die koordinierte Bewegung mehrerer Achsen bezieht, die aufgrund der Antriebsmethode der Achse einen diskreten Fehler aufweisen.



+Radius



-Radius

CIRCLE

Syntax

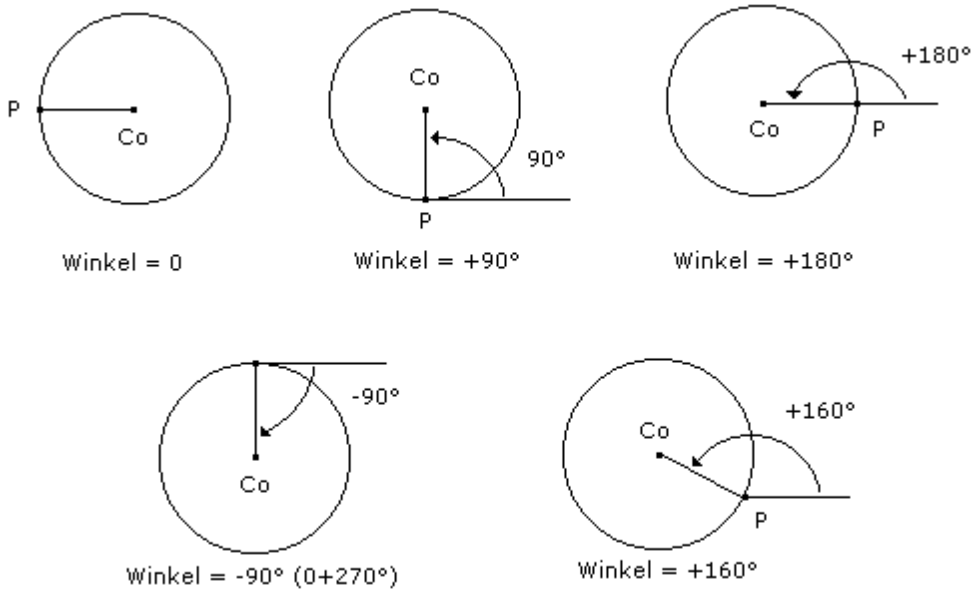
CIRCLE **[Etikett], Achse1, Achse2, Richtung, Radius, Winkel**

Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1, Achse2	Namen von Vorrichtungen des Typs Achse
Richtung	Variable integer. Zur Angabe der Drehrichtung sind folgende Werte zulässig: **CW** im Uhrezeigersinn **CCW** gegen den Uhrseigersinn
Radius	Konstante oder Variable. Stellt den Radiuswert des Kreises dar
Winkel	Konstante oder Variable. Stellt den Startwinkel dar.

Beschreibung

Vollständige, kreisförmige Interpolation.
Schafft einen Kreis mit einer **Achse1** und einer **Achse2**, in der angegebenen **Richtung**, mit dem zugeordneten Wert des **Radius** und gemäß dem vorgegebenen Start**winkel**.
Radius kann nur positive Werte annehmen.
Der **Winkel** ist gemäß trigonometrischer Konvention von der X-Achse ausgehend entgegen dem Uhrzeigersinn positiv anzugeben. Die Position des Mittelpunkts C0 des Kreises ist nämlich vom Winkel bestimmt, der vom Radius durch den programmierten Anfangspunkt P (laufender Punkt) und die waagrechte Richtung X+ gebildet wird. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren.
In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind. In diesem Fall ist zu bedenken, dass sich der Begriff Interpolation auf die koordinierte Bewegung mehrerer Achsen bezieht, die aufgrund der Antriebsmethode der Achse einen diskreten Fehler aufweisen.



COORDIN

Syntax

COORDIN **Matrix, Wert DeltaT, Richtung, ini, fin, Maske, Achse1, Nr_Spalte_Achse1 [, (Achse2, Nr_Spalte_Achse2) ÷ (Achse32, Nr_Spalte_Achse32)]**

Argumente

Matrix	Datenmatrix
Wert DeltaT	Konstante oder Variable. Zeitbasis
Richtung	vordefinierte Konstante. Datenleserichtung in der Matrix. **UP** von der obersten zur untersten Zeile **DOWN** von der untersten zur obersten Zeile.
ini	globale Integer-Variable. Ist die Nummer der Anfangszeile
fin	globale Integer-Variable. Ist die Nummer der Endzeile
Maske	Maske der zu aktivierenden Achsen
Achse1 [....Achse 32]	Namen von Vorrichtungen des Typs Achse
Nr_Spalte_Achse1[...Nr_Spalte_Achse32]	Nummer der auf die Achse bezogenen Matrixspalte

Beschreibung

Diese Anweisung erlaubt die Ausführung von synchronisierten Bewegungen der Achsen **Achse1**, **Achse2**, usw. mittels inkrementaler Bewegungen (Mikrovektoren), die von einer Daten-**Matrix** definiert sind.
Die Parameter **Achse1** und **Nr_Spalte_Achse1** sind immer zu definieren.
Die in der **Matrix** enthaltenen Werte geben die nach und nach von den Achsen erreichten, absoluten Maße an.
Die jeweiligen inkrementalen Bewegungen (Maßunterschied zwischen Zeile (n) und Zeile (n-1)) werden in einem Zeitintervall ausgeführt, der einem **Mehrfachen** der Zeitbasis entspricht (1 ms = Realtime der Achsen-Aktualisierung) und vom Argument **Wert** **Δt** definiert wird, der also von einer ganzen Zahl ausgedrückt wird.
Nach Festlegung dieses Zeitwerts bedingt die Größe der einzelnen Achsenbewegungen deren Geschwindigkeit. Diese Anweisung dient der koordinierten Bewegung von höchstens 32 Achsen entlang beliebiger kurvenförmiger Strecken im Raum, wie sie von SPLINE-Techniken geschaffen werden. Diese Anweisung, deren vollständige Ausführung nicht notwendig ist, benötigt zum Starten die Anweisung STARTINTERP nicht. Dennoch soll die Anweisung WAITSTILL zum Schluss gebracht werden, damit die Achsen das Maß korrekt erreichen. Etwaige Änderungen von Feedrate Ovverride

sollen durch die Anweisung SETFEEDI ausgeführt und durch die Anweisung SETFEEDCOORD bearbeitet werden.
Der Parameter **Richtung** dient der Bestimmung der Ablaufrichtung der Matrix, so daß die Strecke in beide Richtungen ausgeführt werden kann.
Die Spalten der abzulesenden Matrix können des Typs Float oder Double sein, aber nicht beides gleichzeitig.

Außer der Bewegung der Achsen eine beschränkte Strecke entlang (die von Zeilen-Anzahl der Matrix bestimmt wird) ist es möglich eine unendliche Bewegung zu erzielen, wie folgt:

- Bei einer einzeiligen Matrix liest die Steuerung in dieser Funktionierungsmodalität immer die einzige Zeile der Matrix und verwendet den Achsen die Maße, die in der Matrix selbst angegeben sind. Es ist nötig die Zeile der Matrix passend zu ändern, um die Achsen laufen zu lassen. Es ist besser einen Realtime-Task zu verwenden, der die Synchronisierung der Aktualisierung der Achsenmaße mit der Frequenz des Wiederlesen der Achsenmaße versichert.
  Auf dieser Weise können elektronische Nocken oder Verkettungen mit einem Verhältnis verschieden als 1:1 verwirklicht werden.
  Um diese Funktionierungsmodalität zu aktivieren **ini** = 1**, fin** = 0 und **Richtung** = UP (Richtung = UP) müssen eingesetzt werden.
  Wenn diese Funktionierungsmodalität verwendet wird, muß die Anweisung "STOP" NICHT verwendet werden.
- Bei einer mehrzeiligen Matrix kann die Matrix, wenn die Werte **ini** = 1, **fin** = 0 e **Richtung** =UP eingestellt sind, durch Zyklen von der ersten bis zur letzten Zeile auf unbestimmte Zeit skandiert werden. Falls von einer mehrzeiligen Matrix eine einzige Zeile zu skandieren ist, müssen die Parameter **ini**, **fin** und **Richtung** wie folgt eingestellt werden: **ini**= Nummer der auszuführenden Zeile, **fin**= Nummer der Zeile, die die auszuführende Zeile vorhergeht, **Richtung**=UP. In anderem Fall wird eine Systemfehler erzeugt. In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

DISABLECORRECTION

Syntax

DISABLECORRECTION **Achse [, Achse1, ..., Achse6]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Achse1, ..., Achse6	Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung deaktiviert die Linearitätskorrektur für die angegebene **Achse**.
Der erste Parameter ist die Achse, deren Korrektur man deaktivieren möchte. Ist es der einzige angegebene Parameter, werden alle Korrekturen in der Konfiguration deaktiviert. Die folgenden Parameter dienen der Angabe, welche Korrekturen deaktiviert werden sollen. Stimmt einer dieser Parameter mit dem ersten überein, wird die Autokorrektur deaktiviert.
Eine detailliertere Beschreibung ist unter ENABLECORRECTION zu finden.

Beispiel

```
; deaktiviert allein die Autokorrektur der X-Achse
DISABLECORRECTION     X, X

; deaktiviert die gekreuzte Korrektur der Z-Achse (zu X und Y), aber
; nicht die Autokorrektur
DISABLECORRECTION     Z, X, Y
```

EMERGENCYSTOP

Syntax

EMERGENCYSTOP **Achse, Zeit**

Argumente

Achse	Name der Achsenvorrichtungen
Zeit	Variable-Integer. Zeit der Rampe (ms)

Beschreibung

Hält die spezifizierte Achse an und damit alle die von der interpolierten Bewegung eventuell betroffen sind. Die Bewegung wird durch die Verzögerungsrampe innerhalb der von der **[Zeit] Variable aufgezeichneten Zeitspanne zum Stillstand gebracht.**

Ist in den Punkt-Punkt-Bewegungen die festgelegte Zeit größer als die aufgezeichnete Verzögerungszeit, dann wird diese letzte verwendet.
Wenn in den interpolierten Bewegungen die festgelegte Zeit bei dem Höchstwert der Verzögerungszeiten jeder betroffenen Achse liegt, dann wird die in der Konfiguration festgelegte Höchstzeit verwendet.
Die Bewegung wird von einer START Anweisung wiederaufgeführt.
Diese Anweisung kann nicht verwendet werden, wenn **[Achse]** eine Slave-Achse ist.
Diese Anweisung kann folgende Systemfehler erstellen:

- „4101 – Unpassende Verwaltung der Achse", wenn **[Achse]** eine synchronisierte Bewegung, eine multilineare Interpolation oder eine ISO Bewegung gerade ausführt.
- „4105 – Auf der Achse nicht ausführbare Anweisung", wenn **[Achse]** eine Zählachse ist.
- „4399 – Parameter außerhalb des Bereichs" wenn die aufgezeichnete **[Zeit]** gleich oder kleiner als 0 ist.

ENABLECORRECTION

Syntax

ENABLECORRECTION **Achse [, Achse1, ..., Achse6]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Achse1, ..., Achse6	Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung aktiviert die Linearitätskorrektur für die angegebene **Achse**. Die Korrektur setzt sich aus Autokorrektur und gekreuzter Korrektur zusammen. Die Autokorrektur ist eine Korrektur des tatsächlichen Maßes einer Achse in Bezug auf die eigene Position, während die gekreuzte Korrektur eine Korrektur des tatsächlichen Maßes einer Achse in Bezug auf die Position anderer Achsen ist. Es können bis zu fünf gekreuzte Korrekturen definiert werden.
Der erste Parameter ist die Achse, deren Korrektur man aktivieren möchte. Ist es der einzige angegebene Parameter, werden alle Korrekturen in der Konfiguration aktiviert.
Die folgenden Parameter dienen der Angabe, welche Korrekturen aktiviert werden sollen. Stimmt einer dieser Parameter mit dem ersten überein, wird die Autokorrektur aktiviert.
Siehe auch DISABLECORRECTION.

ANMERKUNG: Damit die Anweisung wirksam ist, muss die Korrektur auch in der Konfiguration aktiviert werden.

Beispiel

```
; aktiviert alle in der Konfiguration vorgesehenen Korrekturen
; der X-Achse
ENABLECORRECTION      X

; aktiviert allein die Autokorrektur der X-Achse
ENABLECORRECTION      X, X

; aktiviert die Autokorrektur und die gekreuzte Korrektur
; der Z-Achse (zu X und Y)
ENABLECORRECTION      Z, X, Y, Z
```

ENDMOV

Syntax

ENDMOV **Achse [, Maß]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Maß	Konstante oder Variable.

Beschreibung

Diese Anweisung beendet die Bewegung der angegebenen Achse. Unterscheidet sich von einer STOP-Anweisung darin, dass die Bewegung beendet und durch eine etwaige START-Anweisung nicht fortgeführt wird. Wenn angegeben, bestimmt der Parameter **Maß** die Position, an der die Achse ihre Bewegung beendet; andernfalls hängt der Punkt, an dem die Achse anhält, von der laufenden Geschwindigkeit und der zuletzt programmierten Verzögerung ab. Falls notwendig, sorgt die Steuerung für eine Umkehr der Achsenbewegung, um den Bewegungsendpunkt zu erreichen.

Anmerkung

Dieser Parameter wird nur bei Punkt-Punkt Bewegung gebraucht. Bei interpolierter Bewegung, wird die Achsenbewegung beendet, ohne den Wert dieses **Maßes** zu betrachten.

Beispiel

```
; hält die laufende Bewegung der Achse an und führt sie zum Maß 0.0
ENDMOV    X, 0.0
```

FASTREAD

Syntax

FASTREAD	**Achse1, Zustand, Variable1 [,Achse2, Variable2],[..., Achse8, Variable8]**

Argumente

Achse1...[...Achse8]	Namen von Vorrichtungen des Typs Achse. Achse1 ist die Master-Achse.
Zustand	vordefinierte Konstante. Nimmt folgende Werte an: **ON** Anstiegsflanke **OFF** fallende Flanke
Variable1...[...Variable8]	Variable oder Element einer Matrix/eines Vektors des Typs Double. Gespeichertes Maß

Beschreibung

Die Maße der angegebenen **Achsen** werden in dem Augenblick abgelesen und in den **Variablen** gespeichert, in dem der Fastread-Eingang der **Achse1** (Master-Achse) in den vorgegebenen Zustand wechselt.

Sind die angegebenen Achsen analog, dann müssen sie der selben Platine (4 für TRS-AX) angehören. Sind die angegebenen Achsen digital, dann befindet sich das Schnellsignal unmittelbar auf dem Antrieb; demzufolge, ist bei mehrfachen FASTREAD das Signal auf den verschiedenen Vorrichtungen parallel anzuschließen.
Bei EtherCAT-Busse wird die erlaubte Höchstzahl von Achsen für jede Achse, die ihren Encoder vertauschte, um Eins sinken (Siehe Anweisung SWITCHENC). Wann man solche Grenze überschreitet, wird den Systemfehler „4400 Zu viele aktive Achsen in FASTREAD" generiert. Außerdem muss der zusätzliche Encoder an einer auf-Trs-CAT TRS-AC-E-Erweiterung anschließen werden. Andernfalls wird der Systemfehler „4375 FASTREAD an Achsen verschiedener Platinen ausgeführt" generiert.
Die Anweisung endet, wenn der Eingang in den vorgegebenen **Zustand (ON/OFF)** wechselt.
Wird vor dem Wechsel des Fastread-Eingangs eine STOP-Anweisung gegeben, bleiben diese Anweisungen aktiv und werden nach der START-Anweisung wiederaufgenommen.
Es können gleichzeitig mehrere Fastread-Anweisungen auf derselben Platine aktiviert werden.

Während der Ausführung der Anweisung können die Anweisungen SETPZERO und SETPFLY gleichzeitig auf der selben Achse nicht ausgeführt werden, wenn diese an Platinen mit MECHATROLINK-II-Bus angeschlossen ist.

Bemerkung

Der fastread-Eingang für Achsen digitales Typs auf Platine mit MECHATROLINK-II-Bus ist auf dem Eingang EXTI2 anwesend und bedarf keine virtuell-physikalische Konfiguration.
Die fastread-Eingänge der digitalen Achsen MECHATROLINK-II müssen kurzgeschlossen werden, denn die Speicherung des Maßes einer Achse wird nur in Bezug auf ihrem eigenen Fastread-Eingang ausgeführt.

FREE

Syntax

FREE	**Achse [, Spannung]**

Argumente

Achse	Name einer Vorrichtung des Typs Achse
Spannung	Float-Konstante oder Float-Variable. Bezugsspannung

Beschreibung

Diese Anweisung setzt die **Achse** in den Zustand "offener Loop" (Free), wodurch die *Positionskontrolle* deaktiviert wird. Wenn in einer Verkettung mit anderen Achsen die **Achse** Slave ist, wird die Einschränkung gebrochen und die Bewegung der **Achse** gestoppt. Ist der Parameter **Spannung** angegeben, wird die Bezugsspannung der Achse auf den Wert gesetzt.

Diese Anweisung findet im Fall von Messachsen zur Ermittlung von Maßen Anwendung oder bei Achsen, deren Bewegung von externen mechanischen Elementen modifiziert werden kann, die also deren Position fälschen würden.
Während des Betriebs wird das Maß der Achse abgelesen und aktualisiert, d.h. nach der Wiederherstellung der Positionskontrolle (Anweisung NORMAL) kann die Achse wieder im Absolut-Modus positioniert werden.

HELICABS

Syntax

HELICABS **[Etikett], Achse1, Maß1, Achse2, Maß2, Achse3, Maß3, Richtung, ±Radius [,Winkel [, UmdrZahl [, Achse4, Maß4 [, ...,Achse6, Maß6]]]]**

Argumente

Etikett	Konstante oder Variable Integer. Etikett das den Versetzungsblock identifiziert
Achse1...Achse3[...Achse6]	Name von Vorrichtungen des Typs Achse
Maß1...Maß3[...Maß6]	Konstante oder Variable. Maß der absoluten Versetzung
Richtung	Integer-Variable. Drehrichtung im/entgegen dem Uhrzeigersinn (CW/CCW)
Radius	Konstante oder Variable. Radius der Schraubenlinie
Winkel	Konstante oder Variable. Startwinkel
UmdrZahl	Konstante oder Variable. Anzahl Umdrehungen

Beschreibung

Helikoidale Interpolation mit absoluter Versetzung gemäß den programmierten Maßen **Maß1**, **Maß2** und **Maß3**. Die Bewegung besteht aus einer kreisförmigen Interpolation, die die Achsen **Achse1** und **Achse2** betrifft (mit denselben Syntax-Regeln wie CIRCABS /CIRCINC in bezug auf die Argumente **Richtung**, **±Radius** und **Winkel**), und aus einer linearen Bewegung der **Achse3** (eventuell **Achse4**, **Achse5** und **Achse6**). Die helikoidale Bewegung kann mehrere Umdrehungen beinhalten, die vom Argument **UmdrZahl** angegeben werden. Das Maß der Achse, die die lineare Bewegung ausführt (sowie evtl. die Maße **Achse4**, **Achse5**, **Achse6**), bezieht sich auf die gesamte Versetzung (und nicht auf die Versetzung/Umdrehung). Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren.
In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

Anmerkung

1. Es wird nur die Konturierungsbedingung der ersten drei Achsen bewertet, aus denen das Bezugssystem zusammengesetzt ist. Wird eine andere hinzugefügt und sollte sie geändert werden, dann wird das Geschwindigkeitsprofil falsch verwaltet. Um eine korrekte Bewegung zu erzielen, ist es notwendig, eine Anweisung WAITSTILL zwischen einer HELICABS - Anweisung und der anderen einzulegen.
2. Ist ein örtliches Bezugssystem anhand der Instruktion SETRIFLOC eingestellt, dann müssen die drei Achsen, die das neue Bezugssystem bestimmen, unter den Parametern der Anweisung HELICABS angezeigt werden, obwohl sie keine Versetzung durchführen.

HELICINC

Syntax

HELICINC **[Etikett],Achse1, Maß1, Achse2, Maß2, Achse3, Maß3, Richtung, ± Radius [,Winkel [, UmdrZahl [, Achse4, Maß4 [, ..., Achse6, Maß6]]]]**

Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1...Achse3[...Achse6]	Namen von Vorrichtungen des Typs Achse
Maß1...Maß3[...Maß6]	Konstante oder Variable. Maß der inkrementalen Versetzung
Richtung	Integer-Variable. Drehrichtung im/entgegen dem Uhrzeigersinn (CW/CCW)
Radius	Konstante oder Variable. Radius der Schraubenlinie
Winkel	Konstante oder Variable. Startwinkel
UmdrZahl	Konstante oder Variable. Anzahl Umdrehungen

Beschreibung

Helikoidale Interpolation mit inkrementaler Versetzung gemäß den programmierten Maßen **Maß1**, **Maß2** und **Maß3**.
Die Bewegung besteht aus einer kreisförmigen Interpolation, die die Achsen **Achse1** und **Achse2** betrifft (mit denselben Syntax-Regeln wie CIRCABS /CIRCINC in bezug auf die Argumente **Richtung**, **±Radius** und **Winkel**), und aus einer linearen Bewegung der **Achse3** (eventuell **Achse4**, **Achse5** und **Achse6**). Die helikoidale Bewegung kann mehrere Umdrehungen beinhalten, die vom Argument **UmdrZahl** angegeben werden.
Das Maß der Achse, die die lineare Bewegung ausführt (sowie evtl. die Maße **Achse4**, **Achse5**, **Achse6**), bezieht sich auf die gesamte Versetzung (und nicht auf die Versetzung/Umdrehung). Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren. In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

Anmerkung

1. Es wird nur die Konturierungsbedingung der ersten drei Achsen bewertet, aus denen das Bezugssystem zusammengesetzt ist. Wird eine andere hinzugefügt und sollte sie geändert werden, dann wird das Geschwindigkeitsprofil falsch verwaltet. Um eine korrekte Bewegung zu erzielen, ist es notwendig, eine Anweisung WAITSTILL zwischen einer HELICINC - Anweisung und der anderen einzulegen.
2. Ist ein örtliches Bezugssystem anhand der Instruktion SETRIFLOC eingestellt, dann müssen die drei Achsen, die das neue Bezugssystem bestimmen, unter den Parametern der Anweisung HELICINC angezeigt werden, obwohl sie keine Versetzung durchführen.

JERKCONTROL

Syntax

JERKCONTROL **Achse, Zustand**

Argumente

Achse	Namen von Vorrichtungen des Typs Achse
Zustand	vordefinierte Konstante. Die zulässigen Werte lauten: **ON** Zustand Flag aktiviert **OFF** Zustand Flag deaktiviert

Beschreibung

Diese Anweisung aktiviert, wenn dem Parameter **Zustand** der Wert ON zugewiesen wird, oder deaktiviert, wenn dem Parameter **Zustand** der Wert OFF zugewiesen wird, die Jerk-Kontrolle der Interpolations- und Punkt-Punkt-Bewegungen der **Achse**.
Die Jerk-Kontrolle wird nur mit den Achsen aktiviert, die *S*-förmige und *doppel S*-förmige Beschleunigung- und Verzögerungsrampe konfiguriert haben. Wenn die Achse eine lineare Rampe konfiguriert hat, wird die Jerk-Kontrolle nicht vorgenommen.

JERKSMOOTH

Syntax

JERKSMOOTH **Achse, Wert**

Argumente

Achse	Name von Vorrichtungen des Typs Achse
Wert	Konstante oder Variable des Typs Float

Beschreibung

Bei der klassischen interpolierten Bewegungen können die Achsen in einem Kornturierungsverfahren arbeiten, d.h. bewegen sie sich ohne Halt zwischen zwei nacheindanderfolgenden Versetzungsblöcken, wenn der Winkel zwischen den Tangenten der Trajektorie als der Parameter "Höchstwert der Konturierung" kleiner ist (Standardwert liegt bei 15°, er kann durch die Anweisung SETCONTORNATURE geändert werden). Im gegenteiligen Fall werden die Achsen auf dem Punkt der Kanteder zwei Blöcken mit kontrollierten Verzögerung gehalten und dem neuen Block entlang mit kontrollierten Beschleunigung und Geschwindigkeit nochmal starten lassen. Aufhalten und Neustarten vermindern jedoch die Maschinenleistung. Wenn der Winkel der Konturierung erhebliche Werte, wie z.B. unstetigen Tangenswert höher als 5 Grad, aufnimmt, werden bei der Bewegung nicht unerhebliche Geschwindigkeitssprünge der Achsen bestimmt, mit darauffolgenden unendlichen Beschleunigungswerten, Jerk und beträchtliche mechanischen Belastungen, die sich auch auf die Bearbeitungsqualität rückwirken können.
Die Anweisung JERKSMOOTH ermöglicht, gegenüber einem von dem Benutzer bestimmten Wert, die Geschwindigkeitsprofile der Achsen bei der Konturierungsbewegungen auf glatter Weise, d.h. mit

stetigen Beschleunigung und Geschwindigkeit, zu verbinden. Es ist zu beachten, dass diese unendliche Verbindung setzt, in Vergleich zu der theoretischen Trajektorie, einige kleine Änderungen in die ausgeführte Trajektorie ein. Rund um den Konturierungspunkt zeigen die Achsen ein Geschwindigkeitsprofil ungleich dem theoretischen.
Die Variable **Wert**, die durch einen Prozentwert zwischen 0 und 100 ausgedrückt ist, bestimmt den Wert, um die Geschwindigkeitsprofile auf glatter Weise miteinander zu verbinden. Ein Wert von 0 beinhaltet das theoretische Profil und erstellt unstetige Beschleunigungen und Geschwindigkeitsprofile. Durch einen Wert von 100 werden andauernde verbundene Profile, bessere Leistungen erreicht, sondern auch die höchste Abweichung von der theorischen Trajektorie, die der Geschwindigkeit der Trajektorie entlang proportional ist.

Anmerkung:
Die Anweisung wird nur bei Bewegungen mit klassischen Interpolation (Anweisungen LINEARABS, LINEARINC, CIRCABS, CIRCINC, HELICABS, HELICINC). verwendet, und wird bei den Bewegungen der mehrachsigen Interpolation (Anweisungen MULTIABS und MULTIINC) nicht verwendet.

LINEARABS

Syntax

LINEARABS	**[Etikett],Achse1, Maß1, [Achse2, Maß2 [, Achse3, Maß3 [, ..., Achse6, Maß6]]]**

Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1[...Achse2[...Achse6]]	Namen von Vorrichtungen des Typs Achse
Maß1[...Maß2[...Maß6]]	Konstante oder Variable. Maß der absoluten Versetzung

Beschreibung
Lineare Interpolation *mit absoluter Versetzung* gemäß den von **Maß1**, **Maß2**, usw. angegebenen Maßen. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren.
In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

Anmerkung
1. Es wird nur die Konturierungsbedingung der ersten drei Achsen bewertet, aus denen das Bezugssystem zusammengesetzt ist. Wird eine andere hinzugefügt und sollte sie geändert werden, dann wird das Geschwindigkeitsprofil falsch verwaltet. Um eine korrekte Bewegung zu erzielen, ist es notwendig, eine Anweisung WAITSTILL Instruktion zwischen einer LINEARABS - Anweisung und der anderen einzulegen
2. Wird ein örtliches Bezugssystem anhand der Instruktion SETRIFLOC eingestellt, dann müssen die drei Achsen, die das neue Bezugssystem bestimmen, unter den ersten drei Parametern der Instruktion LINEARABS angezeigt werden, obwohl sie keine Versetzung durchführen.

LINEARINC

Syntax

LINEARINC	**[Etikett],Achse1, Maß1, [Achse2, Maß2 [, Achse3, Maß3 [, ..., Achse6, Maß6]]]**

Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1[..Achse2[...Achse6]]	Namen von Vorrichtungen des Typs Achse
Maß1[...Maß2[...Maß6]]	Konstante oder Variable. Maß der inkrementalen Versetzung

Beschreibung
Lineare Interpolation *mit inkrementaler Versetzung* gemäß den von **Maß1**, **Maß2**, usw. angegebenen Maßen. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren.
In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.
In diesem Fall beachten Sie, dass man das Wort "Interpolation" die koordinierte Bewegung von mehreren Achsen bedeutet. Diese Bewegung bekennt einen diskreten Fehler, der von der Steuermethode der Achsen verursacht ist.

Anmerkung

1. Es wird nur die Konturierungsbedingung der ersten drei Achsen bewertet, aus denen das Bezugssystem zusammengesetzt ist. Wird eine andere hinzugefügt und sollte sie geändert werden, dann wird das Geschwindigkeitsprofil falsch verwaltet. Um eine korrekte Bewegung zu erzielen, ist es notwendig, eine Instruktion WAITSTILL zwischen einer LINEARINC - Anweisung und der anderen einzulegen.
2. Ist ein örtliches Bezugssystem anhand der Instruktion SETRIFLOC eingestellt, müssen die drei Achsen, die das neue Bezugssystem bestimmen, unter den Parametern der Anweisung LINEARINC angezeigt werden, obwohl sie keine Versetzung durchführen.

MOVABS

Syntax

MOVABS **Achse1, Wert1 [, Achse2, Wert2 [, ..., Achse6, Wert6]]**

Argumente

Achse1...[...Achse6] Namen von Vorrichtungen des Typs Achse
Wert1...[...Wert6] Konstante oder Variable. Wert der absoluten Versetzung

Beschreibung

Diese Anweisung sorgt dafür, dass die angegebenen Achsen eine *absolute Bewegung* gemäß den vom **Wert1 [,...Wert6]** angegebenen Maßen ausführen.
Um die Bewegung ausführen zu können, darf die Achse nicht in einer Interpolation begriffen sein und sie hat sich am Maß oder in der Toleranz zu befinden. Die Bewegung der Achse beginnt, sobald die Anweisung ausgeführt wird. Werden innerhalb desselben Tasks mehrere Anweisungen von Punkt-Punkt - Bewegungen ausgeführt, so sind diese untereinander verkettet. Versucht ein zweiter Task an einer bereits in Bewegung begriffenen Achse weitere Punkt-Punkt - Bewegungen auszuführen, wartet dieser Task das Ende der vom ersten Task gesteuerten Bewegung ab.
Außerdem kann mittels der Anweisung SETVEL die Geschwindigkeit zwischen einer Punkt-Punkt - Bewegung und der folgenden geändert werden. Die zwei Bewegungen werden durch eine Geschwindigkeitsrampe verbunden, ohne daß die Achsen zum Stehen kommen.
Wird keine Instruktion SETVEL verwendet, dann ist die möglich Höchstgeschwindigkeit vom manuellen Geschwindigkeitswert dargestellt, der in der Konfiguration bestimmt ist.
Eine Punkt-Punkt - Bewegung kann mittels der Anweisung STOP unterbrochen und durch die Anweisung START wiederaufgenommen werden. Während der Unterbrechung der Bewegung bleibt die Achse im Zustand "konstante Geschwindigkeit", obwohl sie sich nicht bewegt.
Eine Bewegung kann durch die Anweisung ENDMOV beendet werden. In diesem Fall kann die Bewegung nicht wiederaufgenommen werden.

HINWEIS:

1) Bisher galt bei Punkt-Punkt - Bewegungen:
- Die Geschwindigkeit konnte nur bei stehender Achse geändert werden. Das jetzige Verhalten ist mit dem der Interpolationsbewegungen vergleichbar.
- Bei Unterbrechung durch die Anweisung STOP nahm die Achse den Zustand "am Maß" an.

2) Man wird beraten, die Anweisungen über die Linearinterpolation statt die Anweisungen über die Punkt-Punkt Bewegung zu gebrauchen, wenn die Zahl der Bewegungsblöcke größer ist als 32 und die Blöcke aus Mikrosegmenten bestehen. Für ausführlichere Informationen ist bei TPA das Dokument "Limiti Firmware Movimento Punto Punto.doc" erhältlich.

Beispiel 1

Setpoint-Nullstellung bei Interrupt

Beispiel 2

```
; Geschwindigkeitswechsel
Function Geschw.Wechs
   setvel      X, 20
   setvel      X, 20
    movabs     X, 100, Y, 200
    movabs     X, 150, Y, 180
    setvel     X, 5
    movabs     X, 80, Y, 100
    waitstill  X, Y
fret
```

MOVINC

Syntax

MOVINC **Achse1, Wert1 [, Achse2, Wert2 [, ..., Achse6, Wert6]]**

Argumente

Achse1...[...Achse6] Namen von Vorrichtungen des Typs Achse
Wert1...[...Wert6] Konstante oder Variable. Wert der inkrementalen Versetzung

Beschreibung

Diese Anweisung sorgt dafür, dass jede Achse eine *inkrementale Bewegung* gemäß dem entsprechenden **Wert** ausführt.
Um die Bewegung ausführen zu können, darf die Achse nicht in einer Interpolation begriffen sein und sie hat sich am Maß oder in der Toleranz zu befinden. Die Bewegung der Achse beginnt, sobald die Anweisung ausgeführt wird. Werden innerhalb desselben Tasks mehrere Anweisungen von Punkt-Punkt - Bewegungen ausgeführt, so sind diese untereinander verkettet. Versucht ein zweiter Task an einer bereits in Bewegung begriffenen Achse weitere Punkt-Punkt - Bewegungen auszuführen, wartet dieser Task das Ende der vom ersten Task gesteuerten Bewegung ab.
Außerdem kann mittels der Anweisung SETVEL die Geschwindigkeit zwischen einer Punkt-Punkt - Bewegung und der folgenden geändert werden. Die zwei Bewegungen werden durch eine Geschwindigkeitsrampe verbunden, ohne dass die Achsen zum Stehen kommen.
Wird keine Instruktion SETVEL verwendet, dann ist die mögliche Höchstgeschwindigkeit vom manuellen Geschwindigkeitswert dargestellt, der in der Konfiguration bestimmt ist.
Eine Punkt-Punkt - Bewegung kann mittels der Anweisung STOP unterbrochen und durch die Anweisung START wiederaufgenommen werden. Während der Unterbrechung der Bewegung bleibt die Achse im Zustand "konstante Geschwindigkeit", obwohl sie sich nicht bewegt.
Eine Bewegung kann durch die Anweisung ENDMOV beendet werden. In diesem Fall kann die Bewegung nicht wiederaufgenommen werden.

HINWEIS:

1) Bisher galt bei Punkt-Punkt - Bewegungen:
- Die Geschwindigkeit konnte nur bei stehender Achse geändert werden. Das jetzige Verhalten ist mit dem der Interpolationsbewegungen vergleichbar.
- Bei Unterbrechung durch die Anweisung STOP nahm die Achse den Zustand "am Maß" an.

2) Man wird beraten, die Anweisungen über die Linearinterpolation statt die Anweisungen über die Punkt-Punkt Bewegung zu gebrauchen, wenn die Zahl der Bewegungsblöcke größer ist als 32 und die Blöcke aus Mikrosegmenten bestehen. Für ausführlichere Informationen ist bei TPA das Dokument "Limiti Firmware Movimento Punto Punto.doc" erhältlich.

Beispiel 1

Achsen-Nullstellung-Routine

Beispiel 2

```
; Geschwindigkeitswechsel
Function Geschw.Wechs
  setvel      X, 20
  setvel      X, 20
  movinc      X, 100, Y, 200
  movinc      X, 150, Y, 180
  setvel      X, 5
  movinc      X, 80, Y, 100
  waitstill   X, Y
fret
```

MULTIABS

Syntax

MULTIABS **[Etikett],Achse1, Wert1, [Achse2, Wert2 [, Achse3, Wert3 [, ..., Achse16, Wert 16]]**

Argumente

Etikett **Namen von Vorrichtungen des Typs Achse**
Achse **Namen von Vorrichtungen des Typs Achse**
Achse1...Achse16 **Namen von Vorrichtungen des Typs ACHSE**
Wert1...[...Wert16] **Konstante oder Variable. Wert der theoretischen Höhe für Ende des Versetzungsblock**

Beschreibung

Multilineare absolut Interpolation bis zu 16 Achsen. Diese Interpolationsbewegung ermöglicht die Verstellung der Geschwindigkeitsprofile, indem mit der SETTOLERANCE-Anweisung die entsprechenden Toleranzen auf den Achsen eingestellt werden (unter Toleranz versteht man den Abschnitt des Verlaufs, in dem kein konstantes Interpolationsverhältnis vorhanden sein kann). Einfügungsbefehl der Achsen in der MULTIABS-Anweisung **muss** stets gleich sein und **alle** von der Bewegung betroffenen Achsen müssen vorhanden sein. Die Versetzungsblöcke werden im normalen Lookahead hinter einander gereiht und die Bewegung beginnt bei der Durchführung einer Anweisung WAITSTILL, STARTINTERP bzw. beim Füllen dieses Lookahead. Unter den von der Bewegung betroffenen Achsen muss eine mit der WAITCOLL-Anweisung als Collider benutzt werden. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu identifizieren. In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

Anmerkung

Mit diesem Interpolationstyp können keine virtuellen Bezugssysteme benutzt werden (Anweisung SETRIFLOC und RESRIFLOC). Es können Bewegungen mit verketteten Achsen (in CHAIN) durchgeführt werden. Die in der interpolierten Mehrachsenbewegung betroffenen Achsen sollen Master anderer nicht in der Bewegung betroffenen Achsen deklariert werden, während der FeedRate Override angewendet werden kann. Die Anweisung SETINDEXINTERP kann nicht verwendet werden.

Beispiel

```
setquote        x, 0
setquote        y, 0
setquote        z, 0
; erster Block
setveli         x, velx1
setveli         y, vely1
setveli         z, velz1
multiabs        x, quotax1, y,quotay1, z,quotaz1
; zweiter Block
settolerance    x,tollx2,  y,tolly2,  z,tollz2
setveli         x, velx2
setveli         y, vely2
setveli         z, velz2
multiabs        x,quotax2,  y,quotay2,  z,quotaz2
; dritter Block
settolerance    x,tollx3,  y,tolly3,  z,tollz3
setveli         x, velx3
setveli         y, vely3
setveli         z, velz3
multiabs        x,quotax3, y,quotay3, z,quotaz3
; vierter Block
settolerance    x,tollx4,  y,tolly4,  z,tollz4
setveli         x, velx4
setveli         y, vely4
setveli         z, velz4
multiabs        x,quotax4, y,quotay4, z,quotaz4
waitstill       x,  y,  z
```

MULTIINC

Syntax

MULTIINC **[Etikett],Achse1, Wert1, [Achse2, Wert2 [, Achse3, Wert3 [, ..., Achse16, Wert 16]]]**

Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert
Achse1...Achse	Namen von Vorrichtungen des Typs Achse
Wert1...	Konstante oder Variable. Wert der theoretischen Höhe für Ende des Versetzungsblocks

Bezeichnung

Multilineare inkremental Interpolation bis zu 16 Achsen. Diese Interpolationsbewegung ermöglicht die Verstellung der Geschwindigkeitsprofile, indem mit der SETTOLERANCE-Anweisung die entsprechenden Toleranzen auf den Achsen entsprechend eingestellt werden (unter Toleranz versteht man den Abschnitt des Verlaufs, in dem kein konstantes Interpolationsverhältis vorhanden sein kann). Der Einfügungsbefehl der Achsen in die MULTIINC-Anweisung **muss** stets gleich sein und **alle** von der Bewegung betroffenen Achsen müssen vorhanden sein. Die Versetzungsblöcke werden im normalen Lookahead aufgereiht und die Bewegung beginnt bei der Durchführung einer Anweisung WAITSTILL, STARTINTERP bzw. beim Füllen dieses Lookahead. Unter den von der Bewegung betroffenen Achsen muss eine mit der WAITCOLL-Anweisung als Collider benutzt werden. Der optionale Parameter **Etikett** wird mit der Instruktion SETLABELINTERP verwendet, um den Versetzungsblocks eindeutig zu indentifizieren. In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden, nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

Anmerkung

Mit diesem Interpolationstyp können keine virtuellen Bezugssysteme benutzt werden (Anweisung SETRIFLOC und RESRIFLOC). Es können Bewegungen mit verketteten Achsen (in CHAIN) durchgeführt werden. Die in der interpolierten Mehrachsenbewegung betroffenen Achsen sollen Master anderer nicht in der Bewegung betroffenen Achsen deklariert werden, während der FeedRate Override angewendet werden kann. Die Anweisung SETINDEXINTERP kann nicht verwendet werden.

NORMAL

Syntax

NORMAL **Achse**

Argumente

Achse Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung aktiviert die Positionskontrolle der **Achse** und deaktiviert die Achsenverkettung.
Beim Einschalten des Systems werden alle konfigurierten Achsen in "Free" gestellt. Die Achsen gehen nach dem Stand "normal", wenn diese Anweisung beendet ist oder wenn die erste Bewegung ausgeführt wird.
Es empfiehlt sich, diese Anweisung auf jeden Fall vor dem Verfahren zum Nullstellen der Achsen auszuführen, um etwaige Notfall-Zustände auszuschalten.

RESRIFLOC

Syntax

RESRIFLOC **Achse1, Achse2, Achse3**

Argumente

Achse1...Achse3 Name von Vorrichtungen des Typs Achse

Beschreibung

Diese Anweisung stellt den absoluten Bezug für die Achsen X Y Z (**Achse1**, **Achse2**, **Achse3**) wieder her.
Diese Anweisung wird normalerweise nach der Einstellung eines drehversetzten Bezugssystems mittels der Anweisung SETRIFLOC verwendet.

SETINDEXINTERP

Syntax

SETINDEXINTERP **Achse, VarName**

Argumente

Achse Name der Vorrichtung des Typs Achse
VarName Name der globale Variable des typs Integer

Beschreibung

Diese Anweisung definiert einen Index, der die von einer Achse ausgeführten Interpolationsblöcke zählt.
Während einer Interpolationsbewegung wird bei jedem Blockwechsel die Variable **VarName** um 1 erhöht.

Anmerkung
Die als Index verwendete Variable muss eine globale Gruppen- oder globale Maschinenvariable sein.
Die Instruktion kann nur bei den Bewegungen mit klassischer Interpolation (Instruktionen LINEARABS, LINEARINC, CIRCABS, CIRCINC, HELICABS, HELICINC) verwendet werden.
Bei den Bewegungen von mehrachsigen Interpolation (Instruktionen MULTIABS und MULTIINC) kann sie nicht verwendet werden.

SETLABELINTERP

Syntax

SETLABELINTERP	**Achse, Wert**

Argumente

Achse	Name einer Vorrichtung des Typs Achse
Wert	der globalen Variable des Typs Integer

Beschreibung
In der Variablen **Wert,** während einer Interpolationsbewegung wird bei jedem Blockwechsel den Etikettwert den neuen Blocks gegeben. Das Etikett ist in den Anweisungen der interpolierten Bewegung definiert.

Anmerkung
Die Variable **Wert** soll eine globale Gruppenvariable oder eine Modulglobale sein

SETPFLY

Syntax

SETPFLY	**Achse, Zustand, Geschwindigkeit, Maß,[Fehler]**

Argumente

Achse	Name einer Vorrichtung des Typs Achse
Zustand	vordefinierte Konstante. Gibt den Zustand des zu prüfenden Mikroschalters an. Die einstellbaren Werte lauten: **ON** **OFF**
Geschwindigkeit	Float-Konstante oder -Variable.
Maß	Konstante oder Variable.
Fehler	Integer-Variable. Fehlercode

Beschreibung
Diese Anweisung dient dem Eil-Nullstellen des Achsenmaßes. Das Nullstellen wird von einem Mikroschalter gesteuert, der an den Fastread-Eingang des Steckers der Achse (für die Platinen mit MECHATROLINK-II-Bus beziehen Sie sich auf EXTI1) angeschlossen ist.
Während der Bewegung der **Achse**, die durch eine MOVABS-Anweisung ausgeführt wird, wartet die Anweisung darauf, dass der zugehörige Nullstellung-Mikroschalter in den angegebenen **Zustand** wechselt. Sobald dieser Wechsel erfasst wird, wird das reale Achsenmaß ohne Unterbrechung der Bewegung auf Null gesetzt und es werden automatisch das Target-**Maß** und die **Geschwindigkeit** dynamisch neu definiert. Es wird daher die korrekte Bewegung nochmals konstruiert, um das Zielmaß zu erreichen und, falls notwendig, wird auch die Bewegung umgekehrt. Wird das angegebene Maß erreicht, ohne dass ein Zustandswechsel des Eingangs erfasst wird und ist der Parameter **Fehler** vorgegeben worden, erscheint ein Systemfehler. Ist der Parameter **Fehler** eingestellt worden, enthält er den numerischen Code des jeweiligen Systemfehlers.
In diesem Fall ist keine Nullstellung ausgeführt worden und die Anweisung SETQUOTE ist auszuführen, damit erneut nach dem Mikroschalter gesucht wird.
Um die Ausführung der Eil-Nullstellung zu unterbrechen, reicht es, an der Achse eine NORMAL-Anweisung auszuführen oder den Task zu beenden, der zur Ausführung der Nullstellung geführt hat.

Während der Ausführung der Anweisung können die Anweisungen SETPZERO und FASTREAD gleichzeitig auf der selben Achse nicht ausgeführt werden, wenn diese an Platinen mit MECHATROLINK-II-Bus angeschlossen ist.

Beispiel

Nullstellung-Routine bei Interrupt

SETPFLYCHAINSTRAT

Syntax

SETPFLYCHAINSTRAT Achse, [Typ]

Argumente

Achse Name der Vorrichtung Achsentyp

Typ Konstante Integer-Typ. Die zulässigen Werte sind die folgenden:
0 = nur die Master-Achse annulliert das Maß; die Slave-Achse behaltet das vorherige Maß
1 = die Master-Achse und die Slave-Achse annullieren beide das Maß in synchroner Weise

Beschreibung

Diese Anweisung erlaubt es, die Weise einzustellen, mit der die angegebene Slave-Achse im Vergleich zu einer Setpfly-Anweisung auf dem Master funktionieren wird. Die Anweisung muss auf der Slave-Achse durchgeführt werden. Wird die Variable **Typ** ausgelassen, dann wird ein 0 - Standardwert genommen.

SETPZERO

Syntax

SETPZERO Achse, Maß [,Fehler]

Argumente

Achse	Name der Vorrichtung des Typs Achse
Maß	Konstante oder Variable. Das ist ein inkrementales Maß
Fehler	Integer-Variable. Fehlercode

Beschreibung

Diese Anweisung startet eine inkrementale Bewegung der **Achse** zum angegebenen **Maß** und wartet das Ablesen der Nullkerbe des Encoders ab (vor Erreichen des angegebenen Maßes).
Sobald die Kerbe erfasst wird, wird das reale Maß auf Null gesetzt und die Achse angehalten.
Wird das angegebene Maß erreicht, ohne dass die Nullkerbe erfasst wird und ist der Parameter **Fehler** vorgegeben worden, erscheint ein Systemfehler. Ist der Parameter **Fehler** eingestellt worden, enthält er den numerischen Code des jeweiligen Systemfehlers. In diesem Fall ist keine Nullstellung ausgeführt worden und die Anweisung SETQUOTE ist auszuführen, damit erneut nach der Kerbe gesucht wird.
Die von dieser Anweisung erzeugte Bewegung der Achsen kann durch eine STOP-Anweisung unterbrochen und durch eine START-Anweisung wiederaufgenommen werden.
Falls die Anweisung mit S-CAN Achsen und mit EtherCAT Achsen ausgeführt wird, dann muss man zuerst eine FREE-Anweisung ausführen.

Während der Ausführung der Anweisung können die Anweisungen SETPZERO und FASTREAD gleichzeitig auf der selben Achse nicht ausgeführt werden, wenn diese an Platinen mit MECHATROLINK-II-Bus angeschlossen ist.

Beispiel

```
FREE   X
SETPZERO X, 100
```

SETPZEROCHAINSTRAT

Syntax

SETPZEROCHAINSTRAT Achse, [Wert]

Argumente

Achse	Name der Vorrichtung des Typs Achse

Wert	Variable des Typs Integer. Die zulässigen Werte sind: 0 = nur die Master-Achse stellt die Höhe null, die Slave-Achse behält die vorherige Höhe bei anders als 0 = die Masterachse und die Slave-Achse stellen die Höhe im Synchronisiermodus null

Beschreibung

Diese Anweisung stellt die Kontrolle des Modus ein, in dem sich die angegebene Slave-Achse gegenüber einer Anweisung SETPZERO auf dem Master verhält.
Die Anweisung ist auf der Slave-Achse auszuführen.

Wenn die Variable **Wert** ausgelassen wurde, wird ein Default-Wert gleich 0 angenommen.

SETQUOTE

Syntax

SETQUOTE **Achse, Maß**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Maß	Konstante oder Variable

Beschreibung

Diese Anweisung modifiziert gleichzeitig das theoretische und das tatsächliche Maß einer Achse auf den von **Maß** angegebenen Wert. Ist die Achse in Bewegung begriffen, führt dies zum sofortigen Halt der Achse, das sich die Achse unter dem Gesichtspunkt der Steuerung plötzlich am Maß befindet (tatsächliches Maß stimmt mit Target-Maß überein). Daher wird vom Gebrauch dieser Anweisung an einer in Bewegung begriffenen Achse abgeraten, es sei denn bei äußerst niedriger Geschwindigkeit.

Beispiel

Achsen-Nullstellung - Routine

SETQUOTECHAINSTRAT

Syntax

SETQUOTECHAINSTRAT **Achse, [Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Variable des Typs Integer. Die zulässigen Werte sind: 0 = nur die Master-Achse initialisiert auf der neuen Höhe, die Slave-Achse behält die vorherige Höhe bei anders als 0 = die Höhen der Slave-Achse werden im Synchronisiermodus mit den Höhen der Master-Achse initialisiert

Beschreibung

Diese Anweisung stellt die Kontrolle des Modus ein, in dem sich die angegebene Slave-Achse gegenüber einer Anweisung SETQUOTE auf dem Master verhält.
Die Anweisung ist auf der Slave-Achse auszuführen.
Wenn die Variable **Wert** ausgelassen wurde, wird ein Default-Wert gleich 0 eingenommen.

SETRIFLOC

Syntax

SETRIFLOC **Maß1_ax1, Maß2_ax1, Maß3_ax1,**
Maß1_ax2, Maß2_ax2, Maß3_ax2,
Maß1_ax3, Maß2_ax3, Maß3_ax3,
Achse1, Achse2, Achse3

Argumente

Maß1_ax1...Maß3_ax3	Richtungskosinusse der drei Achsen
Achse1...Achse3	Namen von Vorrichtungen des Typs Achse

Beschreibung

Diese Anweisung dient der Aktivierung eines kartesischen Bezugssystems X' Y' Z', das im Verhältnis zum absoluten Bezugssystem der Maschine X Y Z, dargestellt von den physikalischen Achsen **Achse1**, **Achse2** und **Achse3**, drehversetzt ist.
Die neun Argumente geben die Richtungskosinusse der drei lokalen Achsen im Verhältnis zu den absoluten an,

$\cos\alpha_1$	$\cos\beta_1$	$\cos\gamma_1$
$\cos\alpha_2$	$\cos\beta_2$	$\cos\gamma_2$
$\cos\alpha_3$	$\cos\beta_3$	$\cos\gamma_3$

die die Matrix zur Umwandlung der Koordinaten darstellen.
Der Ursprung für das neue Bezugssystem wird auf den laufenden Punkt gesetzt.
Alle die X-, Y- und Z-Achse betreffenden Anweisungen zu Interpolationsbewegungen werden bis zur Ausführung der Anweisung RESRIFLOC auf dieses Bezugssystem bezogen.

SETTOLERANCE

Syntax

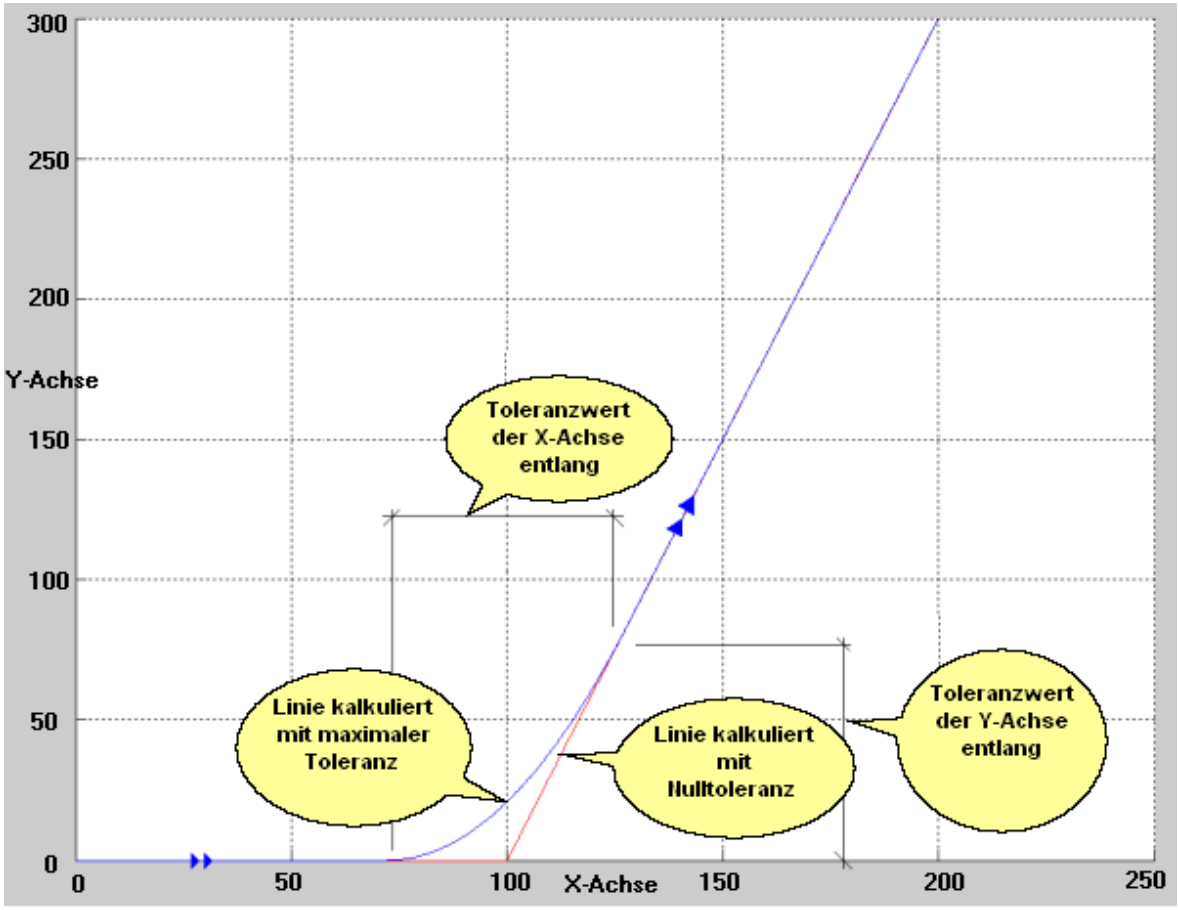
SETTOLERANCE **Achse1, Wert1, [Achse2, Wert2 [, Achse3, Wert3 [, ..., Achse16, Wert 16]]]**

Argumente

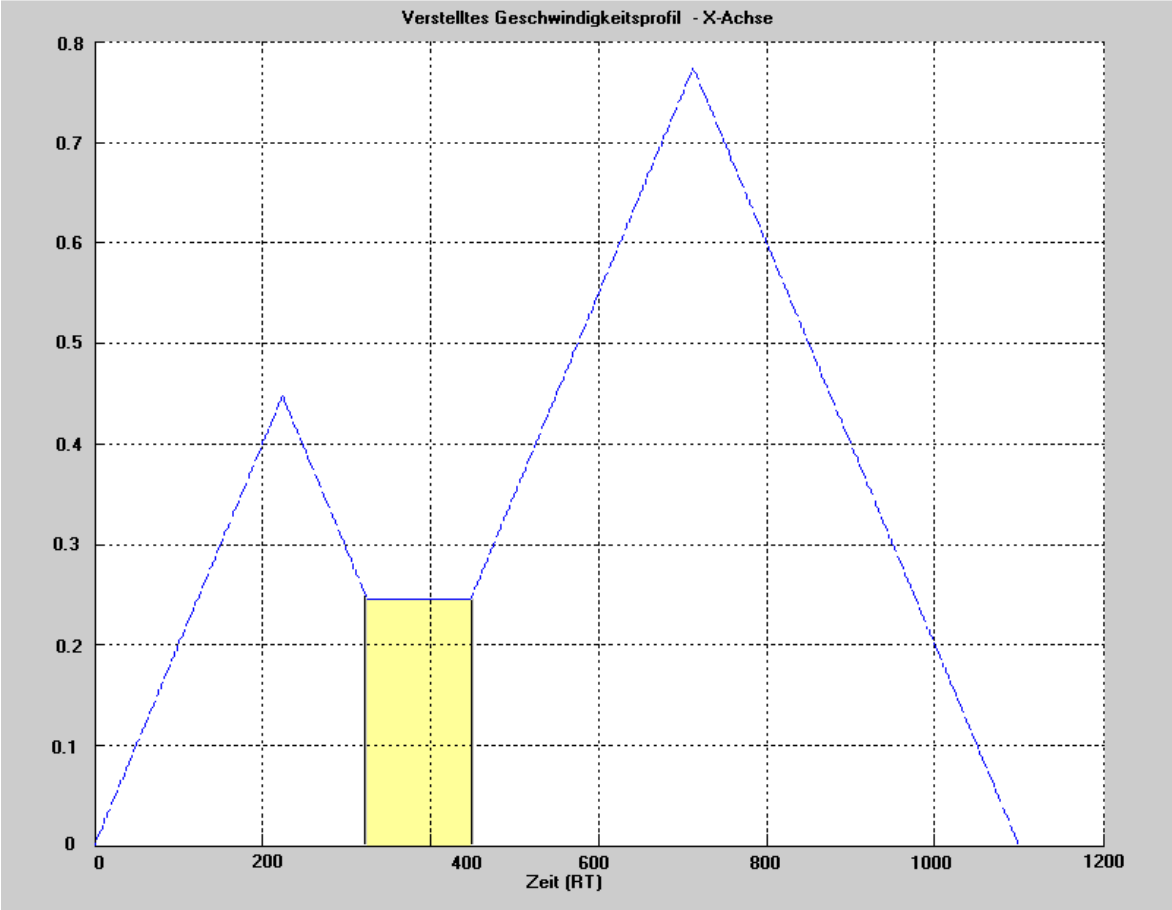
Achse1...Achse16	Namen der Vorrichtungen des Typs Achse
Wert1...[...Wert16]	Konstante oder Variable. Wert der maximalen an der Achse anwendbaren Toleranz

Beschreibung

Diese Anweisung stellt für jede definierte **Achse** den bei den Bewegungen der Mehrachsen-Interpolation anwendbaren Toleranz**wert** ein. Der Toleranzwert ist der Versetzungswert, um den sich die Achse von der ursprünglichen Linie bei einer Multiachsen-Interpolation verschiebt.
Die Toleranz wird für jede von der Interpolation betroffene **Achse** eingestellt und das System nimmt die Verstellung der Geschwindigkeitsprofile vor, wobei die Toleranzen auf allen Achsen eingehalten werden und der Rampenraum, der den maximalen Grenzwert für die Verstellung der Profile darstellt, nicht überschritten wird. Die fehlende Zuweisung der Toleranz vor einer Mehrachsen-Zuweisung führt zur Anwendung der letzten auf dieser Achse eingestellten Toleranz. Falls nie ein Toleranzwert einer bestimmten Achse zugewiesen wurde, wird dieser mit Nulltoleranz berücksichtigt. In diesem Fall nimmt die diese Achse betreffende Mehrachsenbewegung keine Rampenverstellung vor. Es empfiehlt sich also vor einer Anweisung einer Mehrachsen-Interpolationsbewegung der systematische Gebrauch dieser Anweisung für jede betroffene Achse.



Auf der obigen Abbildung ist eine klassische Mehrachsenlinie, die aus zwei Bewegungsblöcken besteht, dargestellt, von denen der erste aus einer Versetzung von 100 der X-Achse besteht, während der zweite aus einer Bewegung der Y-Achse um 300 und der X-Achse immer um 100 besteht. Die rote Linie hebt die Linie bei Nulltoleranz hervor, während die blaue die Linie bei maximaler Toleranz ist.
Die Toleranz kann auch als vom Geschwindigkeitsprofil während des Verstellungsintervalls untergespannte Fläche angesehen werden, wie die nachfolgende Abbildung zeigt.



START

Syntax

START **Achse**

Argumente

Achse Name der Vorrichtung des Typs Achse

Beschreibung

Kompensation der **Achsen**bewegung nach einem Stop.

STARTINTERP

Syntax

STARTINTERP **Achse**

Argumente

Achse Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung startet eine Interpolation, deren Kanal durch **Achse** bestimmt ist. Normalerweise beginnt die Bewegung der einem Interpolationskanal zugeordneten Achsen, wenn der Interpolations-Pufferspeicher komplett voll ist (512 Anweisungen) oder wenn eine WAITSTILL-Anweisung erreicht wird, wodurch die Bewegung beendet wird. Dies erlaubt dem Interpolations-Algorithmus, optimale Geschwindigkeitsprofile festzulegen, da zahlreiche (wenn nicht alle) Strecken der Interpolationsbewegung zur Verfügung stehen. Die Anweisung STARTINTERP erlaubt die Modifizierung des Starts der Achsenbewegung, auch wenn die eben beschriebenen Bedingungen dafür nicht vorliegen.

STOP

Syntax

STOP **Achse**

Argumente

Achse Name der Vorrichtung des Typs Achse

Beschreibung

Unterbrechung der **Achsen**bewegung. Die Achse führt eine Verzögerungsrampe aus, deren Dauer von der laufenden Geschwindigkeit und von den Konfigurationsparametern abhängt.

Beispiel

Achsen-Nullstellung-Routine

SWITCHENC

Syntax

SWITCHENC **Achse1, [Achse2, [Richtung, Maß]]**

Argumente

Achse1 Name der Vorrichtung des Typs Achse

Achse2 Vorrichtungsname des Achsentyps zeigt eine Ausgangsvorrichtung an

Richtung vorbestimmteKonstante.
UP=Encoderaustausch bei dem Überschreiten des Maßes in der positiven Richtung
DOWN=Encoderaustausch bei dem Überschreiten des Maßes in der negativen Richtung

Maß Konstante Double oder Variable Double

Beschreibung

Diese Anweisung ermöglicht, den Encoder der **Achse1** durch den Encoder der **Achse2** auszutauschen. Dieser Austausch wird bei dem Überschreiten des festgelegten Maßes in der positiven (UP) oder negativen (DOWN) **Richtung** ausgeführt.
Wenn die Parameter **Richtung** und **Maß** ausgelassen werden, dann wird der Encoder sofort ausgetauscht, unabhängig von der Position der Achsen.

Wenn nur **Achse1** deklariert wird**,** dann wird den Betrieb mit einem einzelnen Encoder wiederherstellt.

Achse1 kann nicht des Typs Schrittschaltung, Berechnung und virtuell sein, **Achse2** kann nur eine Berechnungsachse sein. Außerdem, sowohl **Achse1** als auch **Achse2** können als Slave-Achse mit verketteten Bewegungen nicht begriffen sein.

Diese Anweisung generiert den Systemfehler „4101 – Unpassende Verwaltung der Achse AchseName", wenn **Achse1** oder **Achse2** in einer verketteten Bewegung Slave deklariert ist, oder wenn **Achse1** eine FASTREAD- oder SETPFLY-Anweisung aus führt. Der Systemfehler „4105 – Auf der Achse AchseName nicht ausführbare Anweisung" kann außerdem generiert werden, wenn der deklarierte Achsentyp nicht zu den Möglichen gehört.

WAITACC

Syntax

WAITACC **Achse1 [, ..., Achse6]**

Argumente

Achse1 [...,Achse6] Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung wartet den Beschleunigungszustand oder einen der nachfolgenden Zustände an allen angegebenen **Achsen** (1÷6) ab.
Der Task in dem die Anweisung durchgeführt wird wird in Wartezeit gesetzt bis wann die Achse sich in Beschleunigungszustand oder in einem der nachfolgenden Zuständen befindet.

Die Zustände der Achse werden von einer ganzen Zahl erkannt:
- Beschleunigung = 1
- Drehzahl = 2
- Verminderung der Geschwindigkeit = 3
- Maß = 4
- Wartezeit großes Fensters = 5
- Wartezeit Stillstand Achse = 6
- Wartezeit kleines Fensters = 7

WAITCOLL

Syntax

WAITCOLL **Achse, Wert, Timeout, Delta**

Argumente

Achse	Name der Vorrichtung von Achsentyp
Wert	Konstante oder Variable. Wert absoluter Koordinate
Timeout	Konstante oder Variable. Das ist die Wartezeit bei stillstehender Achse
Delta	Konstante oder Variable. Das ist das Fensterwert um stillstehende Achse zu erzielen.

Beschreibung

Bei Achsenbewegung kann eine programmierte Koordinate aufgrund eines mechanischen Hindernisses, das manchmal das Werkstück selbst ist, nicht erreicht werden. In diesem Fall generiert das System einen Systemfehler von "Servoerror" oder "Bewegung nicht beendet". Mit solcher Anweisung wird ein Koordinaten**wert** definiert, von dem das System beginnt, folgende Parameter zu prüfen:
- Dasein einer Kollision;
- Wartezeit (**timeout**), bevor die **Achse** nach der ereigneten Kollision auf einen Zustand "Koordinate" gesetzt wird;
- **Delta** der Toleranz auf der Achsenpositionierung.

Wenn die Achse höher als die im **Wert** definierte Koordinate ist, wird geprüft ob die Achse immer noch in Bewegung ist. Nachdem das Hindernis gefunden worden ist, wird die kritische Situation bestimmt und, obwohl der Motorantrieb garantiert ist, wird die Grenze des Loop-Fehler nicht mehr überwunden. Die Bewegungsrichtung, auf der die ereignete Kollision geprüft wurde, hat dieselbe Richtung der letzten Bewegung hinterher.
Timeout ist in Sekunden ausgedrückt, der Wert **Delta** muss größer als 0.001 mm und kleiner als die Differenz zwischen dem programmierten Endmaß und dem Maß des **Wertes** sein. Die Anweisung kann mit dem mehrachsigen Interpolator verwendet werden, da in solchem Interpolator der zeitweilige Ausfall des Interpolationsverhältnisses akzeptiert ist.
Die Anweisung kann auch auf Master-Achsen einer verketteten Bewegung angewandt werden.
Ein Systemfehler wird generiert, wenn:
- Dabei ist, eine klassisch interpolierte oder eine koordinierte Bewegung oder ISO interpolierte Bewegungen auszuführen (siehe Anweisungen LINEARBS, LINEARINC,. CIRCABS, CIRCINC, HELICABS, HELICINC);
- Die Achse eine Slave-Achse ist;
- Die Achse zum Zählen oder eine Schrittachse oder eine virtuelle Achse ist;
- Der eingestellte Wert höher als die Koordinate des Bewegungsendes ist.

Beispiel

```
; setzt die Position der X-Achse fest
SETQUOTE                X, 0.0
; bewegt die X-Achse zum absoluten Maß 1000
MOVABS                  X, 1000.0
; wartet das Kollisionsmaß, wartet 2 Sekunden bis die Achse auf
; den Zustand "Maß" gesetzt ist
; nachdem eine Kollision mit Präzision 0.01 mm gefunden worden ist.
WAITCOLL                X, 980.0,2.0,0.01
```

WAITDEC

Syntax

WAITDEC **Achse1 [, ..., Achse6]**

Argumente

Achse1 [...,Achse6] Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung wartet den Verzögerungszustand oder einen der nachfolgenden Zustände an allen angegebenen **Achsen** (1÷6) ab.
Der Task in dem die Anweisung durchgeführt wird, wird in Wartezeit gesetzt bis wann die Achse sich in Verminderung der Geschwindigkeit, Maß, Wartezeit größes Fensters, Wartezeit kleines Fensters und Wartezeit Stillstand Achse Stand befindet.

Die Zustände der Achse werden von einer ganzen Zahl erkannt:
- Beschleunigung = 1
- Drehzahl = 2
- Verminderung der Geschwindigkeit = 3
- Maß = 4
- Wartezeit großes Fensters = 5
- Wartezeit Stillstand Achse = 6
- Wartezeit kleines Fensters = 7

WAITREG

Syntax

WAITREG **Achse1 [, ..., Achse6]**

Argumente

Achse1 [...,Achse6] Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung wartet die konstante Geschwindigkeit oder einen der nachfolgenden Zustände an allen angegebenen **Achsen** (1÷6) ab.
Der Task in dem die Anweisung durchgeführt wird wird in Wartezeit gesetzt bis wann die Achse sich in Drehzahl, Verminderung der Geschwindigkeit, Maß, Wartezeit größes Fensters, Wartezeit kleines Fensters und Wartezeit Stillstand Achse Stand befindet.

Die Zustände der Achse werden von einer ganzen Zahl erkannt:
- Beschleunigung = 1
- Drehzahl = 2
- Verminderung der Geschwindigkeit = 3
- Maß = 4
- Wartezeit großes Fensters = 5
- Wartezeit Stillstand Achse = 6
- Wartezeit kleines Fensters = 7

WAITSTILL

Syntax

WAITSTILL **Achse1 [, ..., Achse6]**

Argumente

Achse1 [...,Achse6] Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung wartet das Ende der Bewegung (Zustand Maß) an allen angegebenen **Achsen** (1÷6) ab.

Beispiel

Achsen-Nullstellung - Routine

WAITTARGET

Syntax

WAITTARGET **Achse1 [, ..., Achse6]**

Argumente

Achse1 [...,Achse6]	Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung wartet ab, dass an allen angegebenen **Achsen** (1÷6) das theoretische Maß mit dem Target-Maß übereinstimmt. Das tatsächliche Maß wird sich so lange vom theoretischen Maß unterscheiden bis der Loop-Fehler ausgeglichen wird.

WAITWIN

Syntax

WAITWIN	**Achse1 [, ..., Achse6]**

Argumente

Achse1 [...,Achse6]	Name der Vorrichtung des Typs Achse gr

Beschreibung

Diese Anweisung wartet den Zustand Positionierungstoleranz oder einen der nachfolgenden Zustände an allen angegebenen **Achsen** (1÷6) an.
Der Task in dem die Anweisung durchgeführt wird wird in Wartezeit gesetzt bis wann die Achse sich in Wartezeit größes Fensters, Wartezeit kleines Fensters und Wartezeit Stillstand Achse Stand befindet.

Die Zustände der Achse werden von einer ganzen Zahl erkannt:
- Beschleunigung = 1
- Drehzahl = 2
- Verminderung der Geschwindigkeit = 3
- Maß = 4
- Wartezeit großes Fensters = 5
- Wartezeit Stillstand Achse = 6
- Wartezeit kleines Fensters = 7

Achsenparameter

Schreiben/Lesen

DEVICED

Syntax

DEVICEID	**Vorrichtung, Variable**

Argumente

Vorrichtung	Name der Vorrichtung oder Parametervorrichtung
Variable	Integer-Variable, die die logische Adresse empfängt

Beschreibung

In **Variabler** wird die logische mit jeder beliebigen Vorrichtung assoziierten **Adresse** geschrieben.
Diese Anweisung gibt die Möglichkeit, einen mit der Vorrichtung eindeutig assoziierten "Schlüssel", als Index oder Suchschlüssel in Datenstrukturen zu haben.

GETAXIS

Syntax

GETAXIS	**Achse, GrößenName, VarName**
GETAXIS	**Achse, GrößenName1, GrößenName2, [...,GrößenName20,] Matrix[Zeile]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
GrößenName	vordefinierte Konstante (siehe Liste im folgenden Teil). Achsenparameter (1÷20)
VarName	Variable oder Name der Vorrichtung
Zeile	Integer-Konstante oder -Variable. Nummer der Zeile der Matrix
Matrix	Name der Matrix

Beschreibung

In der ersten Form liest die Anweisung eine der Größen (**GrößenName**) einer Achse und speichert sie in einer Variablen.
In der zweiten Form liest diese Anweisung gleichzeitig mehrere Größen einer Achse (von 1 bis 20) und speichert sie in der Reihenfolge, in der sie in den Elementen der angegebenen Matrixzeile angefordert worden sind.
In diesem Fall hat die Anzahl Spalten der Matrix mit der Anzahl angeforderter Daten übereinzustimmen.
Folgende Liste umfasst alle vordefinierten Konstanten, die dem Parameter **GrößenName** zugewiesen werden können.
Die erste Spalte ist der Name der Konstante.
Die zweite Spalte ist die Beschreibung der Größe der Achse, die von der Anweisung gelesen wird.
Die dritte ist das Format der Größe, die in die Variable **VarName** oder **Matrix[Zeile]** zurückgegeben wird, wobei:

- **d** steht für **Double**,
- **f** steht für **Float**,
- **i** steht für **Integer**,
- **b** steht für **Char**.

Unterscheidet sich die Deklaration der Variablen, in der die Größe gespeichert wird, vom Wert, der von der Anweisung zurückgegeben wird, führt der Kompilierer eine Umwandlung (cast) der Größe in den vom Benutzer angeforderten Typ aus. Dies beinhaltet manchmal einen bedeutenden Verlust an Daten. Ein Double-Wert von 12,345 wird z.B. 12 durch die Umwandlung in eine ganze Zahl. Es ist also ratsam, sich bei der Deklaration der Variablen **VarName** und **Matrix[Zeile]** an den angeforderten Typ zu halten. Die letzte Spalte beschreibt entweder den Rückkehrwert oder die Maßeinheit des jeweiligen Parameters.
Die per "_CFG" beginnenden Konstanten dienen dem Lesen der Konfigurationswerte, d.h. der Werte, die beim Einschalten der Maschine gesetzt werden.

Konstante	Beschreibung	Typ	Rückkehrwert
_CFGTYPE	Achsentyp	i	1=Analog,3=Schrittschaltung,4=digital,5=zum Zählen, 6=Frequenz/Richtung, 7=Virtuell
_CFGUM	Maßeinheit	i	0=Millimeter,1=Zoll,2=Grad,3=Umdrehungen
_CFGRIS	Auflösung	d	Impulse pro _ME
_CFGVMAX	maximale Geschwindigkeit	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_CFGVMAXD	maximale Geschwindigkeit im Manuell-Modus	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_CFGVMAXI	maximale Interpolationsgeschwindigkeit	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_CFGPHINV	Umkehr Encoder-Phasen	b	0=keine Umkehr, 1=Umkehr
_CFGRFINV	Bezugsumkehr	b	0=keine Umkehr, 1=Umkehr
_CFGZIND	Aktivierung Maß-Zurücksetzung Nullkerbe	b	0=deaktiviert, 1=aktiviert
_CFGSLOPE	Rampentyp für Beschleunigungen/Verzögerungen in Punkt-Punkt	b	0=linear, 1=a 'S', 2= Doppel S-förmig
_CFGKFFA	Beschleunigung feed forward	f	
_CFGKFFAI	Beschleunigung bei Interpolation feed forward	f	
_CFGSRPP	Anfangsgeschwindigkeit Rampe Schrittschaltung	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_CFGACC	Beschleunigungsdauer von 0 bis _CFGVMAX	i	ms
_CFGDEC	Verzögerungsdauer von _CFGVMAX bis 0	i	ms
_CFGACCI	Beschleunigungsdauer von 0 bis _CFGVMAXI	i	ms
_CFGDECI	Verzögerungsdauer von _CFGVMAXI bis 0	i	ms
_CFGQLP	positive Achsengrenze	d	Maß
_CFGQLN	negative Achsengrenze	d	Maß

Konstante	Beschreibung	Typ	Rückkehrwert
_CFGKP	proportionaler Koeffizient	f	
_CFGKI	integrativer Koeffizient	f	
_CFGKD	derivativer Koeffizient	f	
_CFGKFF	feed forward	f	Prozentsatz
_CFGKPS	proportionaler Koeffizient Slave-Achse	f	
_CFGKIS	integrativer Koeffizient Slave-Achse	f	
_CFGKDS	derivativer Koeffizient Slave-Achse	f	
_CFGQEAP	positiver Loop-Fehler	d	Maß
_CFGQEAN	negativer Loop-Fehler	d	Maß
_CFGKPI	proportionaler Interpolation Koeffizient	f	
_CFGKII	integrativer Interpolation Koeffizient	f	
_CFGKDI	derivativer Interpolation Koeffizient	f	
_CFGTMINP	minimale positive Spannung	f	Volt
_CFGTMINN	minimale negative Spannung	f	Volt
_CFGSTMINP	positive Schwelle - Spannung	f	Volt
_CFGSTMINN	negative Schwelle - Spannung	f	Volt
_CFGESC	Timeout Achse Bewegung	i	ms
_CFGDSE	Aktivierung Dynamisches Servofehler	b	0=deaktiviert, 1=aktiviert
_CFGAEN	Aktivierung Automatisches Adjust	b	0=deaktiviert, 1=aktiviert
_CFGOFFSET	Spannung von Adjust - anfänglicher Offset	f	Volt
_CFGCEE	Maß von falscher Encoder-Verbindung	d	Maß
_CFGNOTCH	Notch Filter Frequenz	i	Hz
_CFGBUFI	integrative berechnung Buffer-Dimension	i	[1, 200]
_CFGQAP	Fenster positive Positionierungstoleranz	d	
_CFGQAN	Fenster negative Positionierungstoleranz	d	
_CFGSLOPEI	Rampentyp für Beschleunigungen/Verzögerungen in Interpolation	f	0=linear, 1=a 'S', 2= Doppel S-förmig
_CFGKFFI	feed forward Interpolation	f	Prozentsatz
_CFGAAF	Wartezeit stillstehender Achsen	b	0=deaktiviert, 1=aktiviert
_CFGENCTYPE	Encodertyp	i	0= simuliert oder abwesend, 1="reell"
_SRPP	Anfangsgeschwindigkeit Rampe Schrittschaltung	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_ACC	Beschleunigungsdauer von 0 bis _VMAX	i	ms
_DEC	Verzögerungsdauer von _VMAX bis 0	i	ms
_ACCI	Beschleunigungsdauer von 0 bis _VMAXI in Interpolation	i	ms
_DECI	Verzögerungsdauer von _VMAXI bis 0 in	i	ms

Konstante	Beschreibung	Typ	Rückkehrwert
	Interpolation		
_QLP	positiver Achsengrenze	d	Maß
_QLN	negative Achsengrenze	d	Maß
_KP	proportionaler Koeffizient	f	
_KI	integrativer Koeffizient	f	
_KD	derivativer Koeffizient	f	
_KFF	feed forward	f	Prozentsatz
_KPS	proportionaler Koeffizient Slave-Achse	f	
_KIS	integrativer Koeffizient Slave-Achse	f	
_KDS	derivativer Koeffizient Slave-Achse	f	
_QEAP	positiver Loop-Fehler	d	Maß
_QEAN	negativer Loop-Fehler	d	Maß
_VEL	Punkt-Punkt Geschwindigkeit	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_VELI	Interpolationsgeschwindigkeit	f	m/1' oder inch/1" oder Grad/1" oder Umdrehungen/1'
_MODE	Betriebsart der Achse	b	1=normal, 2=frei, 8=Interpol., 10=Koord.
_PHINV	Umkehr Encoder-Phasen	b	0=keine Umkehr, 1=Umkehr
_RFINV	Bezugsumkehr	b	0=keine Umkehr, 1=Umkehr
_ZIND	Aktivierung Maß-Zurücksetzung Nullkerbe	b	0=deaktiviert, 1=aktiviert
_KPI	proportionaler Koeffizient Interpolation	f	
_KII	integrativer Koeffizient Interpolation	f	
_KDI	derivativer Koeffizient Interpolation	f	
_KFFI	feed forward Interpolation	f	Prozentsatz
_KFFA	feed forward Beschleunigung	f	Prozentsatz
_KFFAI	feed forward Beshcleunigung bei Interpolation	f	Prozentsatz
_ESC	Timeout Achse Bewegung	i	ms
_CEE	Maß von falscher Encoder-Verbindung	d	Maß
_NOTCH	Notch Filter Frequenz	i	Hz
_BUFI	integrativer Berechnug Buffer-Dimension	i	[1,200]
_QAP	Fenster positive Positionierungstoleranz	d	
_QAN	Fenster negative Positionierungstoleranz	d	
_QEAPINV	Grenze des positivien Loop-Fehlers bei der Inversion	d	
_QEANINV	Grenze des negativen Loop-Fehlers bei der Inversion	d	
_OFSCOORD	Offset-Maß der koordinierten Bewegung	d	
_MS	Master oder Slave - Achsentyp	b	0=nicht verkettert,4=Master,5=Slave
_QENC	Encoder-Maß	d	Maß
_QR	tatsächliches Maß	d	Maß

Konstante	Beschreibung	Typ	Rückkehrwert
_RIS	von der Achse verwendete Auflösung	d	
_ST	Zustand der Achse	b	1=Beschl.,2=konstant,3=Verzög.,4=Maß,5=größes Wartefenster,6=Wartezeit stillstehender Achse,7=kleines Wartefenster,8=Start
_QT	teoretisches Maß	d	Maß
_EA	Loop-Fehler	d	Maß
_FF	feed forward	i	
_VC	laufende Geschwindigkeit	f	
_P	proportionaler Korrektor	i	
_I	integrativer Korrektor	i	
_D	derivativer Korrektor	i	
_FLGS	Achsen-Flags	b	
_VCR	tatsächliche laufende Geschwindigkeit	f	
_ADJUST	Achsen-Offset-Kompensation	i	Dieser ganzzahlige Wert stellt die Spannung dar, die dem Betrieb, von der Seite der DAU-Platinenachse zu liefern ist. Zum Antrieb beträgt der maßstäbliche Wert 10 Volt und zur DAU 32767.
_DAC	DAC-Wert (digital-to-analog-converter/DAU-Wert)	i	Dieser ganzzahlige Wert stellt die Spannung dar, die dem Betrieb, von der Seite der DAU-Platinenachse zu liefern ist. Zum Antrieb beträgt der maßstäbliche Wert 10 Volt und zur DAU 32767.
_ACCINST	Wert der augenblicklichen Beschleunigung	f	
_FFA	Feed Forward Beschleunigung	i	
_GONETIME	Vom Bewegungsanfang vergangene Zeit	f	Sek. (0 für Slave-Achsen und Schrittachsen)
_RESTIME	Restliche Zeit am Bewegungsende. Die Werte betreffen bei deren Anforderung die im Zwischenspeicher befindliche Bewegung	f	Sek. (0 für Slave-Achsen, Achsen mit koordinierter Bewegung und Schrittachsen)
_TARGETTIME	gebrauchter Zeitraum um die Zielposition zu erstellen	f	Mikrosekunden
_GONESPACE	Vom Bewegungsbeginn zurückgelegener Raum. Die Werte betreffen bei deren Anforderung die im Zwischenspeicher befindliche Bewegung.	f	Prozentsatz (100 für Slave-Achsen mit koordinierter Bewegung und Schrittachsen)
_RESSPACE	Restlicher Raum am Bewegungsende. Die Werte betreffen bei deren Anforderung die im Zwischenspeicher befindliche Bewegung.	f	Prozentsatz (0 für Slave-Achsen mit koordinierter Bewegung und Schrittachsen)
_AXESJERK	Aktivierung auf der Kontrollachse des Rucks.	b	1=freigegebene Kontrolle, 0=nicht freigegebene Kontrolle
_MOVEJERK	Aktivierung der Kontrollachse des Rucks bei der Achsenbewegung.	b	1=freigegebene Kontrolle, 0=nicht freigegebene Kontrolle
_MOVETYPE	Bewegungstyp der Achse	b	1=klassische interpolierte Bewegung,2=interpolierte Bewegung der

Konstante	Beschreibung	Typ	Rückkehrwert
			Multiachse, 3=koordinierte Bewegung,4= Punkt-Punkt Bewegung, 5=verkettete Bewegung (nur Slave-Achsen)
_PARTYPESET	Während der Bewegung gebrauchter Achsenparametertyp	i	1=Interpolation, 0=Punkt Punkt
_AXINRIFLOC	laufende Achse in einem örtlichen Bezugssystem	i	1=ja, 0=nein
_QTARGETTOOL	zielgerichtetes Maß der Achse. Bei ISO-Interpolation: Zielgerichtetes Maß der Koordinate der Werkzeugspritze der Achse	d	
_QREALTOOL	zielgerichtetes Maß der Achse. Bei ISO-Interpolation: Reelles Maß der Koordinate der Werkzeugspritze der Achse	d	
_BACKLASH	Wert des Spieles bestimmt für die Achse.	d	
_DISABLED	Deaktivierung einer Achse	b	1=Achse deaktiviert,0=Achse aktiviert
_DYNLIMIT	Aktivierung der dynamischen Kontrolle der Achsengrenzwerte	b	1=Kontrolle aktiviert, 0=Kontrolle nicht aktiviert
_AXESFEED	Wert für Feedrate Override korrekt auf die Achse angewandt	f	
_CORRLIN	Korrektortyp der verwendeten Linearität	i	0=kein verwendeter Korrektor, 1=Selbstkorrektor, 2=gekreuzter Korrektor 3=Selbstkorrektor mit gekreuztem Korrektor
_VELISO	Geschwindigkeit der Werkzeugspitze während der ISO-Bewegung	f	
_ISOSTOPS	Anzahl der Zwangsunterbrechungen der interpolierten Bewegung aufgrund von Grenzsituationen bei der Lookahead-Verwaltung.	f	
_CURRATIO	Wert des zurzeit eingesetzten Verkettungsverhältnisses	d	
_DYNRATIO	wird ergeben, wenn das Verkettungsverhältnis einen dynamischen Wechsel durchläuft	i	0=nein, 1 = ja
_RESBLOCK	Anzahl der noch durchzuführenden Bewegungsblöcke	i	
_EXECBLOCK	Anzahl der durchgeführten Bewegungsblöcke	i	
_TOTALBLOCK	Anzahl der in der Bewegung gesamten gereihten Bewegungsblöcke (aktueller Wert)	i	
_SWITCHENC	überwacht, wenn der Encoder-Austausch im	i	-1=die Achse verwendet die Anweisung SWITCHENC nicht, 0=eine SWITCHENC ist

Konstante	Beschreibung	Typ	Rückkehrwert
	Gang ist		ausgeführt worden, aber die Achse verwendet ihren Encoder, 1= eine Anweisung SWITCHENC ist ausgeführt worden und die Achse verwendet den Encoder der Berechnungsachse
_QOFSENC	Wert des Offset-Encoders	d	
_LENSETPZERO	Wegstrecke um die Nullmarke zu erreichen	d	Position
_TORQUEINST	unverzögerte Drehmomentkennzahl	i	
_CURRSLOPE	gibt den derzeitig verwendeten Rampentyp in den Schnellbewegungen zurück	i	0=linear, 1= 'S` - förmig, 2= Doppel-S-förmig
_CURRSLOPEI	gibt den derzeitig verwendeten Rampentyp in den interpolierten Bewegungen zurück	i	0=linear, 1= 'S` - förmig, 2= Doppel-S-förmig
_REALSLOPEI	gibt den verwendeten Rampentyp in den interpolierten Bewegungen zurück	i	0=linear, 1= 'S` - förmig, 2= Doppel-S-förmig
_AXISPAR1...AXISPAR8	zusätzliche Optionsparameter für die EtherCAT-Achse	i	Nummer
_ISOMOVETYPE	Typ der laufenden ISO-Bewegung	i	0=schnelle ISO-Bewegung, 1 = interpolierte ISO-Bewegung, -1 = mehr
_QMAINENC	Reales Maß des Haupt-Encoders, wenn der sekundäre Encoder in Gebrauch ist	d	

Punkt-Punkt Bewegung

SETACC

Syntax

SETACC **Achse, [Wert]**

Argumente

Achse Name der Vorrichtung des Typs Achse
Wert Konstante oder Variable. Beschleunigungsdauer

Beschreibung

Diese Anweisung ordnet der **Achse** die von **Wert** bestimmte Beschleunigungsdauer zu. Die Beschleunigungsdauer wird in Millisekunden ausgedrückt.
Wird **Wert** ausgelassen, wird der Konfigurationsparameter angenommen. Wenn die Anweisung zwischen zwei Anweisungen MOVABS oder MOVINC gesetzt wird, dann wird die erste Bewegungsanweisung (die Bewegung wird gestoppt) mittels der vorher eingestellten Beschleunigung und Verzögerungsparameter, ausgeführt. Die zweite Anweisung wird mittels der neuen Beschleunigugnsparameter ausgeführt. Die Anweisung SETACC wirkt nur auf die Bewegungen, die deren Ausführung nachkommen.
Ist der angegebene **Wert** kleiner als der Konfigurationswert, wird der Konfigurationswert verwendet.

Siehe auch SETDEC, SETACCI und SETDECI.

SETDEC

Syntax

SETDEC **Achse, [Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Verzögerungsdauer

Beschreibung

Diese Anweisung ordnet der **Achse** die von **Wert** bestimmte Verzögerungsdauer zu. Die Verzögerungsdauer wird in Millisekunden ausgedrückt. Wird **Wert** ausgelassen, wird der Konfigurationsparameter angenommen. Wenn die Anweisung zwischen den zwei Anweisungen MOVABS oder MOVINC gesetzt wird, dann wird die erste Bewegungsanweisung (die Bewegung wird gestoppt) mittels der vorher eingestellten Beschleunigung und Verzögerungsparameter, ausgeführt. Die zweite Anweisung wird mittels der neuen Verzögerungsparameter ausgeführt. Die Anweisung SETACC wirkt nur auf die Bewegungen, die deren Ausführung nachkommen.
Ist der angegebene **Wert** kleiner als der Konfigurationswert, wird der Konfigurationswert verwendet.

Siehe auch SETACC, SETACCI und SETDECI.

SETDERIV

Syntax

SETDERIV **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Koeffizient der Differential-Komponente. Die Typen Char und Integer sind nicht zulässig

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Koeffizient der Differential-Komponente* zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient der Differential-Komponente verwendet.
Die Anweisung findet bei Schrittmotoren keine Anwendung.
Siehe auch die Anweisung SETDERIVI.

SETFEED

Syntax

SETFEED **Achse, Wert**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Stellt den Prozentsatz Feed Rate Override dar

Beschreibung

Diese Anweisung ändert den prozentualen Feed Rate Override - **Wert** der **Achse** in Bezug auf *Punkt-Punkt - Bewegungen*. Siehe auch SETFEEDI.

SETFEEDF

Syntax

SETFEEDF **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Vorschub vorwärts - Prozentsatz

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Vorschub vorwärts - Prozentsatz* zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient für den Vorschub vorwärts verwendet.

Wird die Anweisung auf einen Schrittmotor angewendet, tritt ein Systemfehler auf. Ein Systemfehler wird auch erzeugt, wenn der Variablen **Wert** ein Wert zugewiesen wird, der nicht zwischen 0 und 100 begriffen ist.
Siehe auch die Anweisungen SETFEEDFI, SETFEEDFA, SETFEEDFAI.

SETFEEDFA

Syntax

SETFEEDFA	**Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Vorschub vorwärts - Prozentsatz

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Feed Forward - Prozentsatz* zur Beschleunigung bei Punkt-Punkt - Bewegungen zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient für Feed Forward verwendet.
Wird die Anweisung auf einen Schrittmotor angewendet, tritt ein Systemfehler auf. Ein Systemfehler wird auch erzeugt, wenn der Variablen **Wert** ein Wert zugewiesen wird, der nicht zwischen 0 und 100 begriffen ist.
Siehe auch die Anweisungen SETFEEDF,SETFEEDFI,SETFEEDFAI.

SETINTEG

Syntax

SETINTEG	**Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Koeffizient der Integral-Komponente. Die Typen Char und Integer sind nicht zulässig

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Koeffizient der Integral-Komponente* zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient der Integral-Komponente verwendet.
Die Anweisung findet bei Schrittmotoren keine Anwendung.
Siehe auch die Anweisung SETINTEGI.

SETMULTIFEED

Syntax

SETMULTIFEED	**Achse1, Wert2 Achse2, Wert2 [, Achse3, Wert3 [, ..., Achse16, Wert 16]]]**

Argumente

Achse1...Achse16	Name von Vorrichtungen des Typs Achse
Wert1...[...Wert16]	Konstante oder Variable. Stellt den Prozentsatz von Feed-Rate-Override dar

Beschreibung

Diese Anweisung bearbeitet den **Prozentwert** von Feed-Rate-Override der **angegebenen Achsen**, bezüglich der *Bewegungen Punkt-Punkt*. Für jede Achse kann ein anderer Wert eingestellt werden.

SETPROP

Syntax

SETPROP	**Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Koeffizient der Proportional-Komponente. Die Typen Char und Integer sind nicht zulässig

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Koeffizient der Proportional-Komponente* zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient der Proportional-Komponente verwendet.
Die Anweisung findet bei Schrittmotoren keine Anwendung.
Siehe auch die Anweisung SETPROPI.

SETSLOPE

Syntax

SETSLOPE **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Integer Variable. Rampentyp.

Beschreibung

legt den für die Schnellbewegungen Rampentyp fest:

- 0 lineare Rampe
- 1 ‚S' förmige Rampe
- 2 ‚doppel S' förmige Rampe

Wird der **Wert** ausgelassen, so wird die Konfigurationsrampe wiederherstellt.
Der Typ der Rampe kann geändert werden nur wenn die Achse nicht im Stillstand und beim Zustand *MASS*. Wenn dies nicht der Fall ist, dann wird der Systemfehler „4101 – Unpassende Verwaltung der Achse AchseName" generiert.
In Verbindung mit dieser Anweisung und durch die Anweisung GETAXIS, anhand des Parameters _CURRSLOPE können Sie den zurzeit verwendeten Rampentyp überprüfen.

Siehe auch die Anweisung SETSLOPEI.

SETVEL

Syntax

SETVEL **Achse [,Geschwindigkeit]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Geschwindigkeit	Float-Konstante oder float-Variable

Beschreibung

Diese Anweisung legt die **Höchstgeschwindigkeit** der Achse für die Punkt-Punkt Bewegungen fest.
Die Geschwindigkeit ist in der Maßeinheit der Achse ausgedrückt, wie sie in der Konfiguration angegeben worden ist.
Wenn die programmierte **Geschwindigkeit** größer als die Höchstgeschwindigkeit der Konfiguration ist, wird diese letzte verwendet.
Wird das Argument **Geschwindigkeit** ausgelassen, wird die Konfigurationsgeschwindigkeit angenommen. Es sind nur positive **Geschwindigkeit**swerte zulässig.
Siehe die Anweisung SETVELI.

Beispiel

Achsen-Nullstellung - Routine

Interpolierte Bewegung

LOOKAHEAD

Syntax

LOOKAHEAD **[Wert]**

Argumente

Wert	Konstante oder Variable. Lookahead-Wert

Beschreibung

Diese Anweisung setzt den Lookahead-Wert für die Interpolation. Lookahead ist die Anzahl Interpolationsblöcke, die vor Beginn der Achsenbewegung verarbeitet werden. Lookahead dient der Erzeugung von optimalen Bewegungsprofilen, insbesondere wenn "S"-förmige Rampen verwendet werden.
Wird der Parameter **Wert** nicht angegeben, verwendet das System ein Lookahead von 512 Blöcken (Default).
Der höchst zulässige Wert beträgt **4096/KanalAnz**, wobei **KanalAnz** die Anzahl in der Modulkonfiguration definierter Interpolationskanäle ist. Der mindestzulässige Wert beträgt 256.

Bemerkung

Unter Interpolationsblock versteht man alle einer beliebigen Anweisung zur Interpolationsbewegung (z.B. LINEARABS) zugeordneten Informationen.

Beispiel

```
LOOKAHEAD       1024
```

SETACCI

Syntax

SETACCI **Achse1 [, ..., Achse6] [, Wert]**

Argumente

Achse1,[...Achse6]	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Beschleunigungsdauer

Beschreibung

Diese Anweisung ordnet den Achsen **Achse1**, **Achse2** die von **Wert** bestimmte Beschleunigungsdauer bei Interpolationsbewegungen zu. Die Dauer ist in Millisekunden auszudrücken. Wird **Wert** ausgelassen, wird der Konfigurationsparameter angenommen.
Siehe auch SETACC, SETDEC und SETDECI.

SETACCLIMIT

Syntax

SETACCLIMIT **Achse,[Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Zeit-Konstante des Antriebs

Beschreibung

Diese Anweisung aktiviert und deaktiviert die automatische Berechnung der konstanten Interpolationsgeschwindigkeit je nachdem welche Beschleunigungen von den Achsen toleriert werden. Der Parameter **Wert** ist eine Zeit-Konstante, die zur Bestimmung der von der **Achse** tolerierten Geschwindigkeitserhöhung (in Millisekunden) verwendet wird. Dieser Parameter ist optional. Wird der Parameter weggelassen, sorgt die Anweisung für die Deaktivierung der automatischen Berechnung. Ein Standard-Wert für diesen Parameter ist 30 Millisekunden. Verringert man diese Zeit, wird das Profil langsamer und die Bewegung entsprechend sanfter. Erhöht man sie, erhält man die entgegengesetzte Wirkung. Diese Anweisung kann auf den schraubenförmigen Interpolationen nicht angewandt.

SETACCSTRATEGY

Syntax

SETACCSTRATEGY **Achse, [Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
[Wert]	Variable oder Konstante Integer

Beschreibung

Diese Anweisung erlaubt die Auswahl des gewünschten Beschleunigungstyps für die folgenden Interpolationsbewegungen. Die Anweisung ist für alle von der Interpolation betroffenen Achsen auszuführen.

Die für den Parameter **Wert** sind 0, 1 und 2 die zulässigen Werte. Wird der Wert 0 übergeben, wird die übliche Beschleunigungsstrategie angewandt (als Profilbeschleunigung wird die geringste unter allen von der Interpolation betroffenen Achsen gewählt). Bei Wert gleich 2 und Linearinterpolation wird die größte Beschleunigung gewählt, die die verschiedenen Achsen aushalten können (unter Berücksichtigung der Einzelkomponenten aller Achsen, Linear-und/oder Drehachsen), und bleibt das Beschleunigungsmanagement bei der Kreisinterpolation unverändert. Der Fall des Wertes 1 ruft ein veraltetes Management auf, das aus Komptibilitätsgründen beibehalten wird. In letzterem Fall werden nur die linearen Interpolationsstrecken berücksichtigt, und der Algorithmus funktioniert nur wenn die Beschleunigungs- und Verzögerungsrampen in derselben Interpolationsstrecke enthalten sind.

SETAXPARTYPE

Syntax

SETAXPARTYPE Achse, [Wert]

Argumente

Achse	Name der Vorrichtung Achsentyp
[Wert]	Variable oder Konstante Integer

Beschreibung

Bei der Ausführung einer multilinearen Interpolation, erlaubt diese Anweisung durch die Übergang von den typischen Interpolationsparametern (**Wert**=1) zu denen, die für die Punkt-Punkt-Bewegung (**Wert**=0) verwendet sind, den gebrauchten Parametersatz zu wechseln. Wird die Variable **Wert** ausgelassen, dann wird als Parameter der Auflösungsparameter gebraucht.
Man kann den Parametersatz wechseln nur wenn der Achse bei dem Zustand MASS still steht, ansonsten generiert die Anweisung den Systemfehler „4101 – Unpassende Verwaltung der Achse *AchseName*".

SETCONTORNATURE

Syntax

SETCONTORNATURE [Wert1,[Wert2]]

Argumente

Wert1	Konstante oder Variable. Maximaler Winkel der Konturierung
Wert2	Konstante oder Variable. Maximaler Winkel der Geschwindigkeitsabnahme

Beschreibung

Zwischen zwei Interpolationstangenten wird der Mindestwinkel eingesetzt, bei dessen Überschreitung die Maschine keine Profilierung ausführt, d.h. die Achsen werden am Ende der ersten Strecke angehalten und längs der zweiten Strecke neu gestartet.
Aus diesem Grund wird ein *maximaler Winkel der Konturierung* als **Wert 1** definiert, der zwischen zwei Bewegungsstrecken den maximalen Winkel darstellt, unterhalb dessen die Bewegung hält nicht an.
Wenn der Winkel zwischen den zwei Bewegungsblöcken höher als der maximale Winkel der Konturierung ist, hält die Bewegung an.
Um den Stillstand zu vermeiden, wird der maximale Winkelwert der Geschwindigkeitsabnahme als **Wert 2** eingestellt.
Falls der Winkel zwischen zwei Bewegungsblöcken zwischen dem *maximalen Winkel der Konturierung und dem maximalen Winkel der Geschwindigkeitsabnahme* liegt, dann hält die Bewegung nicht an, sondern wird langsamer.
Der maximale Winkel der Geschwindigkeitsabnahme stellt also den Winkel dar, jenseits dessen die Bewegung anhalten muss.
Für Winkel die kleiner als der maximale Winkel der Konturierung sind, **nimmt die Bewegung nicht ab;** für Winkel zwischen dem maximalen Winkel der Konturierung und dem maximalen Winkel der Geschwindigkeitsabnahme nimmt die Bewegung ab; für Winkel die größer als der maximale Winkel der Geschwindigkeitsabnahme **hält die Bewegung ab**.
Wert1 und **Wert2** sind optionale Parameter: Werden beide nicht eingestellt, dann werden als Standardwert 15 Grad übernommen.
Wird nur der erste Parameter eingestellt, dann nimmt man an, dass *der maximale Winkel der Geschwindigkeitsabnahme* gleich dem *maximalen Winkel der Konturierung* ist.
Die Funktion von Geschwindigkeitsabnahme wird deaktiviert, wenn der maximale Winkel der Geschwindigkeitsabnahme kleiner oder gleich dem maximalen Winkel der Konturierung ist.
Der maximale Winkelwert von Geschwindigkeitsabnahme liegt bei 180°.
Wird ein größerer Wert eingestellt, dann wird der folgende Systemfehler generiert: „4399: Parameter außerhalb des Bereichs".

Die Funktion von Geschwindigkeitsabnahme wird freigegeben, nur wenn die Instruktion JERKSMOOTH aktiv ist, während die Konturierung immer aktiv ist.

Bemerkung

Der Gebrauch dieser Instruktion ist mit dem Gebrauch der Anweisungen JERKSMOOTH und SETSLOWPARAM und übt seine Wirkung nur auf die Bewegungen mit klassischer Interpolation (Anweisungen LINEARABS, LINEARINC, CIRCABS, CIRCINC, HELICABS, HELICINC aus.

SETDECI

Syntax

SETDECI **Achse1 [, ..., Achse6] [, Wert]**

Argumente

Achse1,[...Achse6]	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Verzögerungsdauer

Beschreibung

Diese Anweisung ordnet den Achsen **Achse1**, **Achse2** die von **Wert** bestimmte Verzögerungsdauer bei Interpolationsbewegungen zu. Die Dauer ist in Millisekunden auszudrücken. Wird **Wert** ausgelassen, wird der Konfigurationsparameter angenommen.
Siehe auch SETACC, SETDEC und SETACCI.

SETDERIVI

Syntax

SETDERIVI **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Koeffizient der Differential-Komponente. Die Typen Char und Integer sind nicht zulässig

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Koeffizient der Differential-Komponente* für Interpolationsbewegungen der Achsen zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient der Differential-Komponente verwendet.
Die Anweisung findet bei Schrittmotoren keine Anwendung.
Siehe auch die Anweisung SETDERIV.

SETFEEDFAI

Syntax

SETFEEDFAI **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Vorschub vorwärts - Prozentsatz

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Feed Forward - Prozentsatz* zur Beschleunigung bei Interpolationsbewegungen zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient für Feed-Forward verwendet.
Wird die Anweisung auf einen Schrittmotor angewendet, tritt ein Systemfehler auf. Ein Systemfehler wird auch erzeugt, wenn der Variablen **Wert** ein Wert zugewiesen wird, der nicht zwischen 0 und 100 begriffen ist.
Siehe auch die Anweisungen SETFEEDF,SETFEEDFI,SETFEEDFA.

SETFEEDFI

Syntax

SETFEEDFI **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Vorschub vorwärts - Prozentsatz

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Feed Forward - Prozentsatz* bei Interpolationsbewegungen zu.
Ist kein **Wert** angegeben, nimmt das System den Vorschub vorwärts - Prozentsatz an, der in den Konfigurationsparametern der betroffenen Achse definiert worden ist.
Die Anweisung ist nicht auf Schrittmotoren anwendbar.
Für die Variable **Wert** sind Werte zwischen 0 und 100 zulässig.
Siehe auch die Anweisung SETFEEDF,SETFEEDFA,SETFEEDFAI.

SETFEEDI

Syntax

SETFEEDI	**Achse, Wert**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Stellt den Prozentsatz Feed-Rate-Override dar

Beschreibung

Diese Anweisung ändert den **Wert** Prozentsatz des Feed-Rate-Override der **Achse** in Bezug auf *Interpolationsbewegungen*. Siehe auch die Anweisung SETFEED.

SETINTEGI

Syntax

SETINTEGI	**Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Koeffizient der Integral-Komponente. Die Typen Char und Integer sind nicht zulässig

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Koeffizient der Integral-Komponente* für Interpolationsbewegungen der Achsen zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient der Integral-Komponente verwendet.
Die Anweisung findet bei Schrittmotoren keine Anwendung.
Siehe auch die Anweisung SETINTEG.

SETPROPI

Syntax

SETPROPI	**Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Koeffizient der Proportional-Komponente. Die Typen Char und Integer sind nicht zulässig

Beschreibung

Diese Anweisung weist der **Achse** den **Wert** *Koeffizient der Proportional-Komponente* für Interpolationsbewegungen der Achsen zu.
Ist kein **Wert** angegeben, wird der Konfigurationskoeffizient der Proportional-Komponente verwendet.
Die Anweisung findet bei Schrittmotoren keine Anwendung.
Siehe auch die Anweisung SETPROP.

SETSLOPEI

Sintassi

SETSLOPEI **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung des Typs einer Achse
Wert	Konstante oder Integer Variable. Rampentyp.

Descrizione

legt den für die Schnellbewegungen Rampentyp fest:

- 0 lineare Rampe
- 1 ‚S' förmige Rampe
- 2 ‚doppel S' förmige Rampe

Wird der **Wert** ausgelassen, so wird die Konfigurationsrampe wiederherstellt.

Der Typ der Rampe kann geändert werden nur wenn die Achse noch nicht in einen Interpolationskanal eingebunden ist.
Wenn dies nicht der Fall ist, dann wird der Systemfehler 4101 „Unpassende Verwaltung der Achse AchseName" generiert.

In Verbindung mit dieser Anweisung und durch die Anweisung GETAXIS anhand des Parameters _CURRSLOPEI können Sie den zurzeit verwendeten Rampentyp überprüfen. Durch den Parameter _REALSLOPEI können Sie den von der Achse zurzeit verwendeten Rampentyp (die Rampe des Kanals in dem die Achse eingebunden ist) überprüfen.

Siehe auch die Anweisung SETSLOPE.

SETSLOWPARAM

Syntax

SETSLOWPARAM **[Achse, [,Wert1, Wert2]**

Argumente

Achse	Name einer Vorrichtung von Achsentyp
Wert1	Konstante oder Variable double. Allgemeiner Verzögerungsfaktor
Wert2	Konstante oder Variable double. Allgemeiner Verzögerungsfaktor

Beschreibung

Diese Anweisung bearbeitet die Parameterwerte, die zum Berechnen der Geschwindigkeitsabnahme notwendig sind, im Falle die funktionale Geschwindigkeitsabnahme bei der Konturierung in Betrieb ist (s. Instruktion SETCONTORNATURE).
Am Anfang wird die Verzögerungsgeschwindigkeit auf theoretischer Weise Achse für Achse berechnet. Bei Umkehr der Bewegung kann die Verzögerungsgeschwindigkeit reduziert werden, wenn bei der Kalkulation der **Wert2** verwendet ist. Nachher wird die Mindestgeschwindigkeit unter allen berechneten Geschwindigkeiten betrachtet; auf dieser Weise wird die Dynamik der Achse erfüllt, die die meisten Grenzen setzt. Schließlich kann die Verzögerungsgeschwindigkeit eines Faktors, der vom **Wert1** ausgelassen wird, abhängt, weiter reduziert werden.

Falls **Wert1** oder **Wert2** fehlen, dann werden Standardwerte übernommen, die zu keiner Wirkung führen. Der Parameter **Wert1** stellt den Prozentwert der Reduzierung der theoretischen Verzögerungsgeschwindigkeit dar. Die angewandte Verzögerungsgeschwindigkeit liegt bei (100 - **Wert1**) / 100mal die theoretische Verzögerungsgeschwindigkeit. Der Höchstwert der Reduzierung liegt bei 100. In diesem Fall ist die daraus resultierende Geschwindigkeit gleich 1% der theoretischen Geschwindigkeit. Umgekehrt, ist der Wert Null oder fehlt, betrachtet man den Standardwert oder betrachtet man die vollständige theoretische Geschwindigkeit.
Der Parameter **Wert2** stellt den Prozentwert der Reduzierung, zwischen 1 und 10 mal, der theoretischen Verzögerungsgeschwindigkeit dar, bei Umkehr der Achsenbewegung. Im Einzelnen, wenn der **Wert2** bei 100 liegt, sinkt die Geschwindigkeit um 10 Mal. Umgekehrt, wenn der Wert liegt bei Null oder fehlt, dann sinkt die Geschwindigkeit nicht.
Die Instruktion erzeugt einen Systemfehler 4399: „Parameter außerhalb des Bereichs", wenn der eingestellte Wert kleiner als Null oder größer als 100 ist. Es ist jedoch wichtig daran zu erinnern, dass, falls der Parameter **Wert1** ausgelassen wird, dann auch der Parameter **Wert2** ausgelassen werden muss.

Bemerkung

Der Gebrauch dieser Instruktion ist mit dem Gebrauch der Anweisungen JERKSMOOTH und SETCONTORNATURE und übt seine Wirkung nur auf die Bewegungen mit klassischer Interpolation (Anweisungen LINEARABS, LINEARINC, CIRCABS, CIRCINC, HELICABS, HELICINC)

SETVELI

Syntax

SETVELI **Achse1 [, ..., Achse6] [, Geschwindigkeit]**

Argumente

Achse1 [...Achse6]	Name der zu interpolierenden Vorrichtung des Typs Achse
Geschwindigkeit	Float-Konstante oder -Variable

Beschreibung

Diese Anweisung setzt die **Geschwindigkeit** der Achsen **Achse1**, **Achse2** für Interpolationsbewegungen.
Die Geschwindigkeit ist in der Messeinheit der Achse ausgedrückt, wie sie vom Konfigurationsparameter angegeben worden ist. Wird das Argument **Geschwindigkeit** ausgelassen, wird die maximale Konfigurationsgeschwindigkeit angenommen.
Schrittachsen können in dieser Anweisung verwendet werden, nur wenn sie von einem TRS-AX-Fernmodul gesteuert sind.
Siehe die Anweisung SETVEL.

SETVELILIMIT

Syntax

SETVELILIMIT **Achse, Geschwindigkeit**

Argumente

Achse	Name der Vorrichtung Achsentyp
Geschwindgkeit	Float-Konstante oder variable Float-Variable

Beschreibung

Diese Anweisung führt die einzelnen **Geschwindigkeitskomponenten** der **spezifizierten Achse** ein, für die interpolierten Bewegungen.
Die Geschwindigkeit wird in der Achsen-Maßeinheiten angegeben.

Koordinierte Bewegung

SETFEEDCOORD

Syntax

SETFEEDCOORD **Achse, Wert1, Wert2**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert1	Konstante oder Variable double. Bezeichnet den maximalen Prozentsatz der augenblicklichen Veränderung des Feedrate
Wert2	Konstante oder Variable Integer. Bezeichnet die Nummer von Realtime, in der keine Veränderung der Feedrate angewandt wird

Beschreibung

Diese Anweisung ändert den **Wert1**, maximaler augenblicklicher Änderungsprozentsatz des Feedrate der **Achse**. Der Feed Rate wird nicht mehr für das in Realtime ausgedrückte, von der Variablen **Wert2** definierte Zeitintervall geändert. Mit anderen Worten: Nachdem die Änderung des Feedrate Override, höchstens mit **Wert1** so oft wie eine Zahl von Realtimes, derer **Wert2** beträgt, angewandt wurde, kann keine neue Feedrate-Änderung angewandt werden. Die Kombination dieser beiden Parameter definiert eine Art Beschleunigung/Geschwindigkeitsabnahme, welche die Achse tragen kann. Durch Modulieren dieser beiden Parameter können "Stufen"-Rampen mit gewünschter Neigung erzielt werden.

Bemerkung

Für jede von der koordinierten Bewegung betroffene Achse müssen der Feedrate-Wert und das Zeitintervall eingestellt werden, andernfalls werden die Default-Werte **Wert1**=100 und **Wert2**=1 angenommen. Während der Ausführung der koordinierten Bewegung (Anweisung COORDIN) berechnet das System die auf die Bewegung anzuwendenden Parameter **Wert1** und **Wert2** auf der Grundlage der Parameter aller betroffenen Achsen neu. Die stillstehenden Achsen werden von der Kontrolle ausgeschlossen. Die beiden Parameter werden wie folgt berechnet:
Wert1: an den sich bewegenden Achsen eingestellter Mindestwert
Wert2: durch Dividieren des **Werts1** durch das Mindestverhältnis **Wert1/Wert2** erzielter Wert.

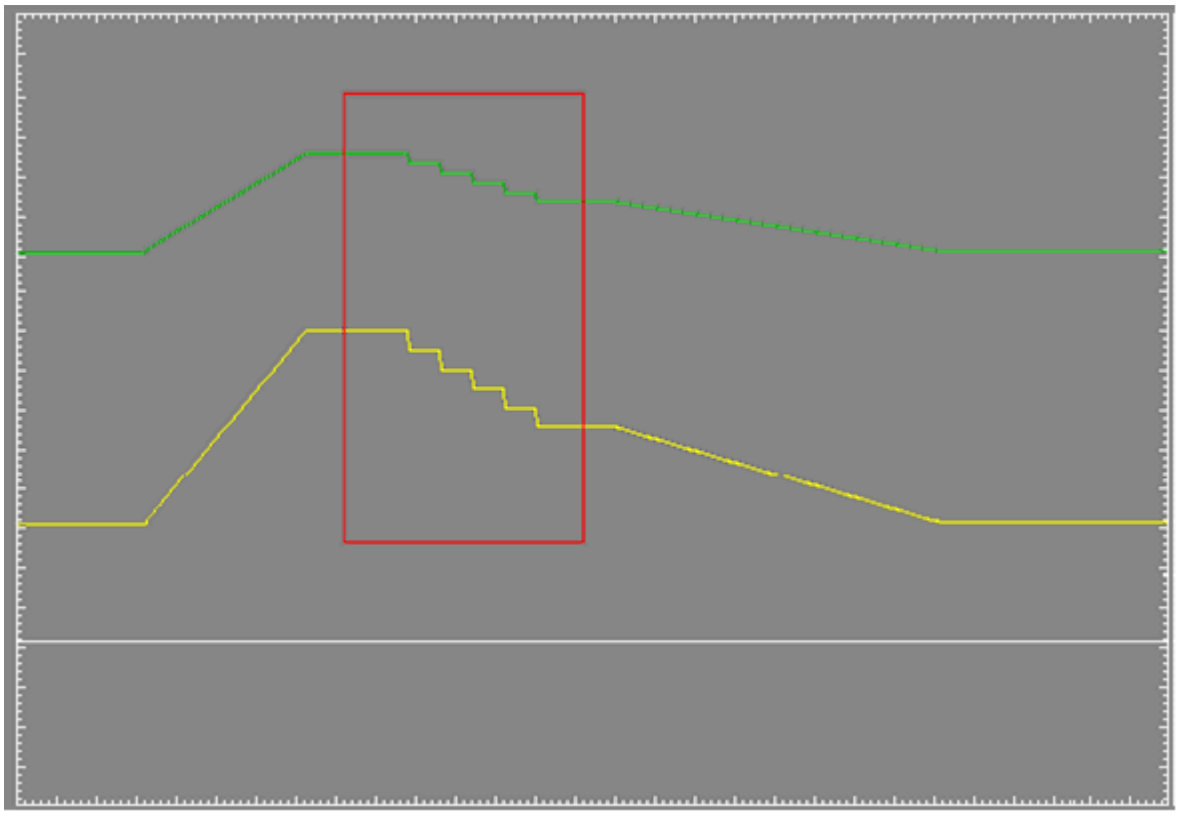
Beispiel

```
;
Function KoordinierteBewegung

      Setquote              X,0
      Setquote              Y,0
      Setquote              Z,0

      setFeedCoord          X, 20, 80
      setFeedCoord          Y, 10, 1
      setFeedCoord          Z, 3, 3

      coordin               matrix, deltaT, UP, rigaInit, rigaEnd,mask,_
                            X,columnX, Y,columnY, Z,columnZ
      waitstill             x,y,z

fret
```

Nehmen wir an, dass in einem bestimmten Durchgang der koordinierten Bewegung die Z-Achse sich nicht bewegt. Die eingestellten Parameter sind

Max_Änderung = **10**
Delta_T = 10 / 0.25 = **40**

Wir erhalten also die folgende Kurve des Oszilloskops mit dem Geschwindigkeitsprofil der X-Achse in Grün und dem der Y-Achse in Gelb.



SETOFFSET

Syntax

SETOFFSET **Achse, Maß**

Argumente

Achse Name der Vorrichtung des Typs Achse
Maß Konstante oder Variable. Offset bei koordinierter Bewegung

Beschreibung

Diese Anweisung stellt einen Offset für die Maße der koordinierten Bewegung ein.
Der vom Parameter **Maß** angegebene Offset wird auf die folgenden koordinierten Bewegungen angewandt, wobei das angegebene Maß zu allen in der Tabelle vorhandenen Maßen summiert wird.
Siehe auch die Anweisung COORDIN.

Verkettete Bewegung

RATIO

Syntax

RATIO **Achse, [Wert]**

Argumente

Achse Name der Vorrichtung des Typs Achse.
Wert Konstante oder Variable. Untersetzungsverhältnis.

Beschreibung

Diese Anweisung dient der Einstellung des Verkettungsverhältnisses zwischen einer Slave- und der zugehörigen Master-Achse. Die Bewegungen der Slave-Achse erfolgen entsprechend dem eingestellten Verkettungsverhältnis zur Master-Achse untersetzt. Läßt man den Parameter **Wert** aus, wird das Verhältnis wieder auf 1.0 gesetzt (identische Bewegungen). Die Anweisung verursacht einen Systemfehler, wenn sie ausgeführt wird, während die Achse nicht im Slave-Zustand und die zugehörige Master-Achse nicht am Maß ist. Siehe Anweisung CHAIN.

Beispiel

```
CHAIN       X, Y
RATIO       Y, 0,5      ; Untersttzungverhältnis 1/2

MOVABS      X, 100      ; Y-Achse bewegt sich zum Maß 50
WAITSTILL   X
```

SETDYNRATIO

Syntax

SETDYNRATIO **Achse, Wert**

Argumente

Achse Name der Vorrichtung Achsentyps
Wert Konstante oder Variable Double

Beschreibung

Diese Instruktion erlaubt, während der Bewegung der Master-Axis das Verkettungsverhältnis auf dynamische Weise zu ändern. Ein neuer Wert des Verkettungsverhältnisses kann angewandt werden, obwohl die vorige Änderung nicht zu Ende gekommen ist. Die deklarierte **Achse** muss eine Slave-Achse sein.
Wenn die Instruktion mit Master-Achse im Zustand MASS ausgeführt wird, dann wird der neue Wert des Verkettungsverhältnisses **Wert** sofort angewandt.
Die Änderung des Verkettungsverhältnisses wird mittels einer linearen Beschleunigung- oder Verzögerungsrampe ausgeführt und der verwandte Beschleunigungswert wird von der Beschleunigung der Master-Axis gegeben, die zur Zeit für die Punkt-Punkt-Bewegung gebraucht ist. Dies bedeutet dass es auch möglich ist, diese Rampe durch die Festlegung eines neuen Beschleunigungswertes mittels der Instruktion SETACC zu ändern.
Diese Instruktion kann folgenden Systemfehler erstellen:

- "4101: Unpassende Verwaltung der Achse", wenn die deklarierte **Achse** nicht eine Slave-Achse ist.

Allgemeine Parameter

DYNLIMIT

Syntax

DYNLIMIT **Achse, Zustand**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Zustand	Vorbestimmte Konstante. Die zulässigen Werte sind: **ON** Einschaltung der dynamischen Kontrolle der Achsengrenzwerte **OFF** Ausschaltung der dynamischen Kontrolle der Achsengrenzwerte

Beschreibung

Diese Anweisung schaltet den dynamischen Test für das Überschreiten der Achsengrenzwerte ein oder aus.
Der dynamische Test für das Überschreiten der Achsengrenzwerte unterscheidet sich vom statischen Test für das Überschreiten der Achsengrenzwerte, da er bei jedem Realtime prüft, dass die Achse auf der Grundlage ihrer aktuellen Geschwindigkeit und ihrer maximalen Geschwindigkeitsabnahme ihre Grenzwerte überschreitet. Der Test des statischen Typs prüft hingegen Augenblick für Augenblick, dass die aktuelle Ankunftshöhe jeder Achse in den positiv oder negativ eingestellten Grenzwerten liegt. Darüber hinaus prüft der Test des statischen Typs vor dem Bewegungsstart, ob die mit den Anweisungen der Bewegung weitergeleiteten Höhen die eingestellten Grenzwerte überschreiten.
Vor einer DYNLIMIT Anweisung sind die SETLIMPOS-und SETLIMNEG Anweisungen einzustellen, um die neuen Grenzen zu bestimmen.

Beispiel

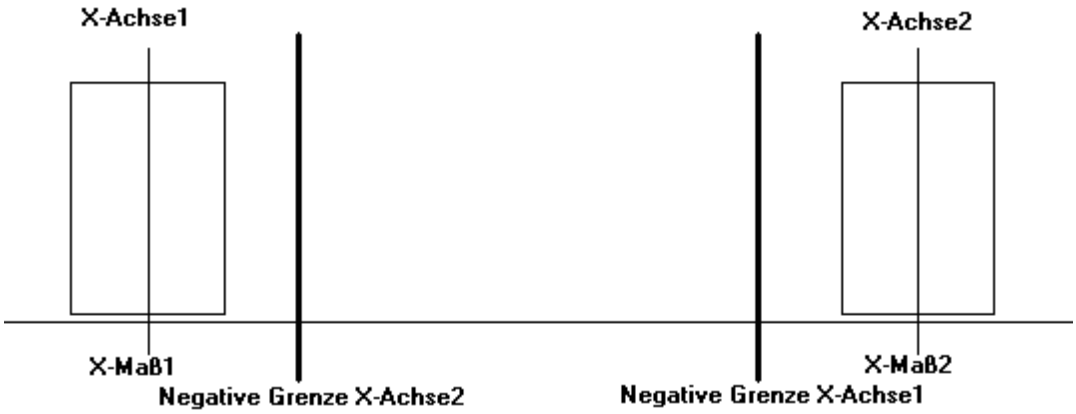
Prüfung der Achsengrenzwerte gemäß den beiden Testtypologien, statisch und dynamisch, mit Achsen auf derselben Bewegungslinie.

Statischer Test

In einer allgemeinen Bewegung kann die **X-Achse1** nicht den von der Höhe der **X-Achse2** gegebenen positiven Anfangsgrenzwert überschreiten. Die Prüfung der Achsengrenzwerte generiert einen Systemfehler n. 4108"Achse X1: Endmaß außerhalb der Softwaregrenze".

Dynamischer Test

In einer allgemeinen Bewegung wird geprüft, dass die augenblickliche **Höhe X1** in den reduzierten Achsengrenzwerten mit dem entsprechenden Zeichen gemäß der Bewegungsrichtung der Achse und des Mindestplatzes für das Anhalten der Achse selbst liegt. Der Mindestplatz für das Anhalten wird je nach augenblicklicher Geschwindigkeit und der bei der Konfiguration eingestellten Geschwindigkeitsabnahme für die Bewegung Spitze-Spitze berechnet. Darüber hinaus, wird die Anfangskontrolle für das Überschreiten der durch die mit den Anweisungen der Bewegung übermittelten Höhen eingestellten Grenzwerte ausgeschaltet.



ENABLESTARTCONTROL

Syntax

ENABLESTARTCONTROL **Achse, [timeout]**

Argumente

Achse	Name der Vorrichtung Achsentyp
timeout	Variable oder Konstante Integer. Es ist die Grenz-Wartezeit in Realtime

Beschreibung

Diese Anweisung erlaubt es, der **timeout** für den Kontroll des fehlenden Anfangs oder des plötzlichen Stops der Achse zu befähigen und einzuführen.
Wenn die Achse sich nicht in mindestens 2 Schritten in 200 Realtime in Anbetracht einer Bewegungs-Anfrage bewegt, wird der Systemsfehler Nr.3 "Servoerror" eintreten.
Wenn das **Timeout-**Parameter auf Null eingeführt ist, wird die Steuerung außer Dienst gesetzt. Die Anweisung hat keine Folge, wenn die teorische Geschwindigkeit kleiner ist als zwei Schritte in 200 Realtime oder wenn die Bewegung in weniger als 200 Realtime beendet.

Beispiel

```
; Achsen-Start-Timeout: 10 Realtime
ENABLESTARTCONTROL X, 10
```

NOTCHFILTER

Syntax

NOTCHFILTER	**Achse, [Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Frequenzwert [Hz] Zulässige Werte zwischen **0** und **500** inbegriffen

Beschreibung

Diese Anweisung dient der Einstellung des Frequenz-Grenzwerts für den Notchfilter der angegebenen Achse. Ist der **Wert** gleich 0, wird der Filter deaktiviert. Bleibt der Parameter **Wert** unbestimmt, wird der in der Konfiguration eingestellte Wert verwendet.

Beispiel

```
; Frequenz-Grenzwert 97 Hz
NOTCHFILTER          X, 97
```

RESLIMNEG

Syntax

RESLIMNEG	**Achse**

Argumente

Achse	Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung deaktiviert den Test der negativen Grenze an der angegebenen **Achse**.
Diese Anweisungen werden gewöhnlich in Nullstellung-Routines zur Suche der Nullstellung-Switch verwendet, um den Achsen das Übertreten der vorgegebenen Konfigurationswerte zu ermöglichen.
Siehe auch die Anweisungen SETLIMNEG, SETLIMPOS, RESLIMPOS.

Beispiel

Achsen-Nullstellung - Routine

RESLIMPOS

Syntax

RESLIMPOS	**Achse**

Argumente

Achse	Name der Vorrichtung des Typs Achse

Beschreibung

Diese Anweisung deaktiviert den Test der positiven Grenze an der angegebenen **Achse**.
Diese Anweisungen werden gewöhnlich in Nullstellung-Routines zur Suche der Nullstellung-Switch verwendet, um den Achsen das Übertreten der vorgegebenen Konfigurationswerte zu ermöglichen.

Siehe auch die Anweisungen RESLIMNEG, SETLIMPOS, SETLIMNEG.

Beispiel

Achsen-Nullstellung - Routine

SETADJUST

Syntax

SETADJUST **Achse, Zustand, [Wert]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Zustand	vordefinierte Konstante. Kann folgende Werte annehmen: • **ON** aktiviert • **OFF** deaktiviert
[Wert]	Float – Variable oder Konstante. Spannung [Volt]

Beschreibung

Diese Anweisung aktiviert oder deaktiviert an der angegebenen **Achse** die Berechnung des automatischen Offset-Ausgleichs, d.h. Adjust.
Die Adjust-Einstellung dient der Kompensation von leichten Positionsoffsets am Ende der Bewegung der Achse. Normalerweise ist die Adjust-Einstellung aktiviert.
Die Deaktivierung dieser Einstellung kann vorteilhaft sein bei Achsen, die von Motoren mit hoher Positionhysterese bewegt werden, die also keinen Nutzen aus dieser Funktion der Steuerung ziehen würden.
Wird Adjust nach einer Deaktivierung erneut aktiviert, berücksichtigt die Steuerung den vorher berechneten Wert nicht. Die Anweisung kann also auch zum Nullstellen des an einer Achse angehäuften Adjust verwendet werden, ohne die Steuerung neu initialisieren zu müssen.
Wenn der dritte Parameter da ist, wird der Offset am angegebenen **Wert** des Programms unabhängig von der Aktivierung oder Entaktivierung des automatischen Adjusts eingestellt. Diese Benutzung der Anweisung erlaubt es, einen eventuellen Offset des Bezugs der Geschwindigkeit durch den Software auszugleichen, statt der Offset auf dem Motorantrieb ausgegleicht wird; die Ausgleichung auf dem Driver, aber, ist der des Softwares vorzuziehen.
Die Anweisung findet nur Anwendung an analog gesteuerten Achsen.

SETBACKLASH

Syntax

SETBACKLASH **Achse, Wert**

Argumente

Achse	Vorrichtungsname Typ Achse
Wert	Variable oder Float-Konstante. Spielwert.

Beschreibung

Diese Anweisung ermöglicht, die Nachwirkungen der mechanischen Spielen auf dem Achsenpfad zu versinken oder zu beseitigen. Der Spiel**wert,** der eingestellt werden kann, muss zwischen 0.0. und 3.0 liegen. Dieser Wert hängt von der ausgewählten Maßeinheit nicht ab.
Besondere Situationen ergeben sich in folgenden Fällen:

- Bei deaktivierter Achse wird die Funktion vom Spielausgleich, trotz ihrer Anforderung, nicht angewandt.
- Bei senkrechter Achse ergibt sich kein Spiel auf Grund der besonderen Konfiguration.
- Bei Achse mit großer Trägheit kann die Last teilweise oder manchmal völlig ausgeglichen werden. Gegeben die Lastmasse, ist es nämlich möglich, dass die Achsenbewegung, dem Motorstillstand gegenüber spät zum Stillstand gebracht wird. Aufgrund der daraus resultierenden Positionierung der Zähne des Untersetzungszahnrads gegenüber der Positionierung der Zähne des Motorgetriebes kann der Spiel vermindert oder sogar annulliert werden.

Die Visualisierung der reellen Quoten und des **Achsencodierers**, die vom Oszilloskop bemustert sind, zeigt in den Punkten in denen die Wiederhaufnahme der Spiele aktiviert ist, den selben Höhewert des Spieles selbst.

Die Anweisung erzeugt einen Systemfehler, wenn sie

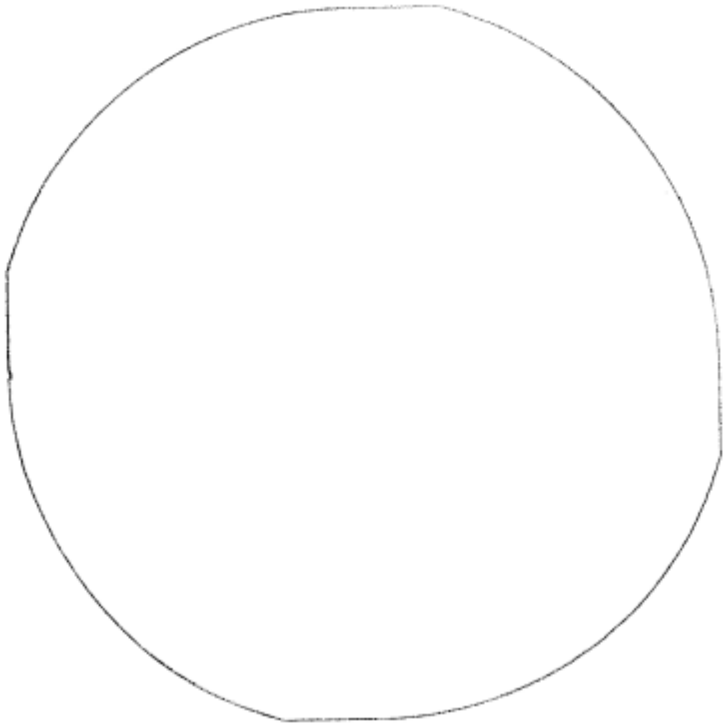
- auf Schrittachsen, die nicht vom Ferngeräten TRS-AX verwaltet sind, Zählachsen, virtuelle Achsen
- auf Schrittachsen, die vom Ferngeräten TRS-AX mit simulierten Encoder verwaltet sind

verwendet ist.

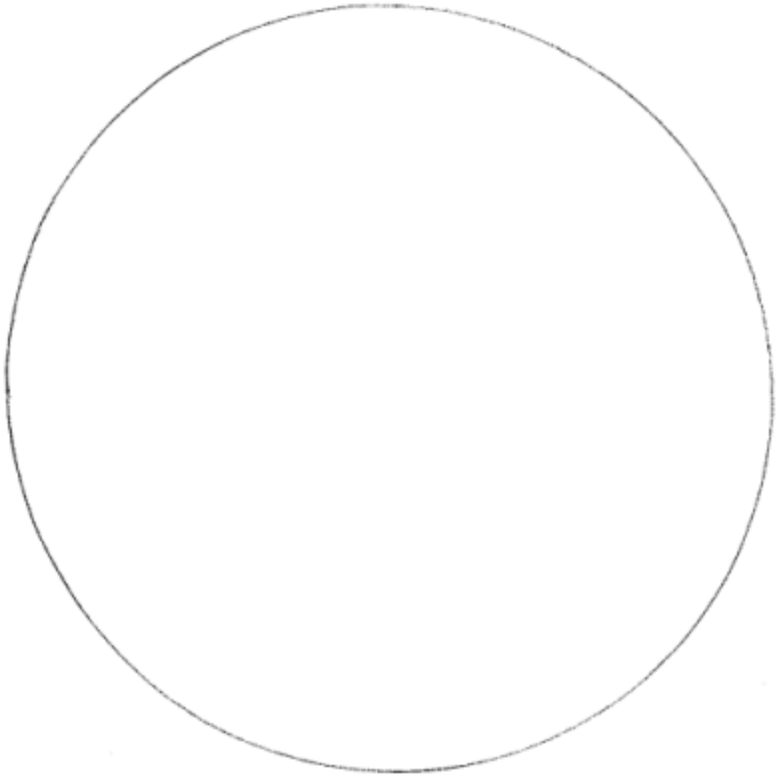
Beispiel

```
; Funktion mit deaktiviertem Spielausgleich (rotes Zeichen abgebildet)
SETQUOTE        X, 0
SETQUOTE        y, 0
SETVELI         X, 1.0
CIRCLE          X,Y,cw,100,90
WAITSTILL       X,Y

; Funktion mit freigegebenem Spielausgleich (schwarzes Zeichen
                abgebildet)
SETQUOTE        X, 0
SETQUOTE        y, 0
SETVELI         X, 1.0
SETBACKLASH     X, 1.9
SETBACKLASH     y, 1.8
CIRCLE          X,Y,cw,100,90
WAITSTILL       X,Y
```

Die Ausführung der zwei Funktionen generiert zwei verschiedene Spuren.
Im ersten Bild wird die Interpolation von zwei Achsen dargestellt, die einen Spiel in der Kombination Motor-Getriebe zeigen.



In der zweiten Abbildung wird die selbe Interpolation dargestellt, in der die Anweisung zum Spielausgleich gebraucht ist.



SETBIGWINFACTOR

Syntax

SETBIGWINFACTOR **Achse, Wert**

Argumente

Achse	Vorrichtungsname Typ Achse
Wert	Variable oder Gleitkommazahl-Konstante. Multiplikationsfaktor zur Berechnung des großen Fensters.

Beschreibung

Diese Instruktion ermöglicht, den Multiplikationsfaktor zur Berechnung des großen Fensters auf der gewünschten **Achse** zu ändern. Um das große Fenster zu kalkulieren, ist die Variable **Wert** mit dem in der Konfiguration der Achsen von Fenster Positionierungstoleranz zu berechnen. Der **Wert**, der eingestellt werden kann, soll sich zwischen 1 und 257 befinden mit Ausnahme der Extremwerte. Defaultwert ist 4.0.

SETDEADBAND

Syntax

SETDEADBAND **Asse,VMinPos,VMinNeg,VSogliaPos,VSogliaNeg**

Argumente

Achse	Name der Vorrichtung Achsen-Typ
VMinPos	Variable oder Float-Konstante. Minimale positive Spannung [Volt]
VMinNeg	Variable oder Float-Konstante. Minimale negative Spannung [Volt]
VSogliaPos	Variable oder Float-Konstante. Positive Schwelle [Volt]
VSogliaNeg	Variable oder Float-Konstante. Negative Schwelle [Volt]

Beschreibung

Diese Anweisung führt die Parameter der minimale Spannung für die angegebene Achse ein. Die Werte der minimalen (positiven/negativen) Spannung werden der theoretischen (positiven/negativen) Bezugsspannung summiert, wenn der Wert dieser Spannung den eingeführten (positiven/negativen) Schwelle-Wert übertrifft. Wenn die theoretische Bezugsspannung innerhalb der Schwelle-Werte ist, wird die effektive Bezugsspannung zum Null forciert. Es ist möglich, die Kontrolle

der minimalen Spannung außer Betrieb zu setzen, mittels der Einstellung aller Werte zum Null. Die Schwelle-Werte müssen immer den entsprechenden Werten der minimalen Spannung kleiner oder gleich sein.
Bei der Initialisierung des Systems wird die Kontrolle der minimalen Spannung außer Betrieb gesetzt.

SETENCLIMIT

Syntax

SETENCLIMIT **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung Achsentyp
Wert	Double-Konstante oder Variable

Beschreibung

Diese Anweisung ändert die falsche Encoder-Verbindungs-Grenze. Der Parameter ist in der Maßeinheit der Achse ausgedrückt. Die zulässigen Werte müssen im Intervall 128 – 16384 - Encoders-Schritte eingeschlosen sein. Wenn der Parameter ausgelassen wird, wird der Default-Wert mit 1024 Schritte wieder eingegeben.
Zum Beispiel, für eine Achse mit Auflösung 1000 "Impulse/mm" werden die zulässigen Werte im Intervall 0,128 - 16,384 mm sein.
Wenn das **Wert-Parameter** auf Null eingestellt ist, wird die Steuerung der falschen Encoder-Verbindungsgrenze außer Dienst gesetzt.

Beispiel

```
; eingesetzter Encoder-Verbindungs-Grenzwert gleich 3.5
SETENCLIMIT     X, 3.5
```

SETINDEXEN

Syntax

SETINDEXEN **Achse, Zustand**

Argumente

Achse	Name der Vorrichtung Achsentyp
Zustand	vorbestimmte Konstante. Die zulässigen Werte sind: **ON** aktiver Null-Kerbe-Zustand **OFF** entaktivierter Null-Kerbe-Zustand

Beschreibung

Diese Anweisung aktiviert oder entaktiviert die Annullierung der Quote an der Null-Kerbe auf der angegebenen **Achse**.
Die Achse muß eine Berechnungsachse-Typ sein, um diese Anweisung auszuführen.

SETINTEGTIME

Syntax

SETINTEGTIME **Achse [, Wert]**

Argumente

Achse	Name der Vorrichtung Achsentyp
Wert	Konstante Integer oder Variable

Beschreibung

Diese Anweisung führt die Anzahl der Muster der Ring-Fehler ein, die für die Berechnung der Integral-Komponente verwendet werden. Die gültigen Werte sind im Intervall 1 - 200 eingeschlossen. Die plötzliche Variation dieses Parameters ist möglich, sie kann, aber, Stufen auf dem Geschwindigkeitsreferenz der Achse verursachen. Es wird darum empfohlen, auf diesem Parameter mit in Stillstand und deaktivierten Achsen oder vorzugsweise in Free hinzugreifen.

SETIRMPP

Syntax

SETIRMPP **Achse, Geschwindigkeit**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Geschwindigkeit	Float-Konstante oder -Variable. Anfangsgeschwindigkeit der Rampe

Beschreibung

Diese Anweisung weist der **Achse** den Wert der **Geschwindigkeit** *am Anfang der Rampe* zu. Es handelt sich um die Mindest-Geschwindigkeit des Schrittmotors.
Diese Anweisung wird für Achsen verwendet, die von Schrittmotoren bewegt werden.

SETLIMNEG

Syntax

SETLIMNEG **Achse [, Maß]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Maß	Konstante oder Variable. Negative Grenze

Beschreibung

Diese Anweisung weist der **Achse** das negative Grenz**maß** zu.
Wird der **Maß** Parameter ausgelassen, wird die negative Grenze gemäß Konfiguration aktiviert.
Diese Anweisungen werden gewöhnlich in Nullstellung-Routines zur Suche der Nullstellung-Switch verwendet, um den Achsen das Überschreiten der vorgegebenen Konfigurationswerte zu ermöglichen.
Siehe auch die Anweisungen RESLIMNEG, SETLIMPOS, RESLIMPOS.

Beispiel

Achsen-Nullstellung - Routine

SETLIMPOS

Syntax

SETLIMPOS **Achse [, Maß]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Maß	Konstante oder Variable. Positive Grenze

Beschreibung

Diese Anweisung weist der **Achse** das positive Grenz**maß** zu.
Wird der **Maß** Parameter ausgelassen, wird die positive Grenze gemäß Konfiguration aktiviert.
Diese Anweisungen werden gewöhnlich in Nullstellung-Routines zur Suche der Nullstellung-Switch verwendet, um den Achsen das Übertreten der vorgegebenen Konfigurationswerte zu ermöglichen.
Siehe auch die Anweisungen RESLIMNEG, RESLIMPOS, SETLIMNEG.

Beispiel

Achsen-Nullstellung - Routine

SETMAXER

Syntax

SETMAXER **Achse, Wert [, Richtung]**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Wert	Konstante oder Variable. Maximaler Loop-Fehler
Richtung	vordefinierte Konstante. Achsenrichtung. Kann folgende Werte annehmen: **POSITIVE**

NEGATIVE

Beschreibung

Diese Anweisung weist der **Achse** den *höchst zulässigen Loop-Fehler* - **Wert** in der angegebenen Richtung zu, bevor die Steuerung auf den "Servofehler" hinweist.
Wird die **Richtung** ausgelassen, wird der Loop-Fehler - Höchstwert für beide Richtungen festgelegt.

SETMAXERNEG

Syntax

SETMAXERNEG **Achse, Verzug, Vorschuß**

Argumente

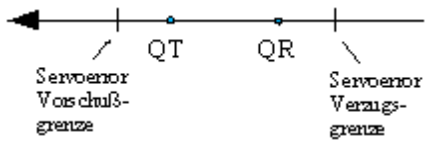
Achse	Name der Vorrichtung des Typs Achse
Verzug	Konstante oder Variable. Maximaler Loop-Fehler
Vorschuß	Konstante oder Variable. Maximaler Vorschuß-Fehler

Beschreibung

Diese Anweisung weist der **Achse** die höchst zulässigen Loop- und Vorschuß-Fehler in negativer Richtung zu, bevor die Steuerung auf den "Servoerror" hinweist.

- **Verzug**: Das ist der Loop-Höchstwert im Fall eines statischen Servoerror-Tests, oder im Fall eines dynamischen Servoerror-Tests ist es der Wert, der bei Zusammenrechnen theoretischen Proportionalfehler und Geschwindigkeit den Loop-Höchstwert bestimmt.
- **Vorschuß**: Das ist der Loop-Höchstwert während der Umdrehung von negativer zu positiver Bewegungsrichtung.

Der Loop-Fehler ist der Unterschied zwischen theoretischem Maß (wo sich die Achse befinden sollte) und tatsächlichem Maß. Wenn sich die Achse in negativer Richtung bewegt, weist ein Loop-Fehler mit negativem Vorzeichen auf einen Verzug der Achse hin, während ein Loop-Fehler mit positivem Vorzeichen einen Vorschuß der Achse signalisiert. Verwendet man diese Anweisung nicht, geht die Steuerung von den Höchstwerten des Loop-Fehlers gemäß der Achsenkonfiguration aus; in diesem Fall entspricht die Vorschußgrenze 1/4 des Verzugs.



Beispiel

```
;Der maximale Verzug der Achse beträgt 10mm , der maximale Vorschuß 5mm
SETMAXERNEG     Assi.X, 10, 5
```

SETMAXERPOS

Syntax

SETMAXERPOS **Achse, Verzug, Vorschuß**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Verzug	Konstante oder Variable. Maximaler Loop-Fehler
Vorschuß	Konstante oder Variable. Maximaler Vorschuß-Fehler

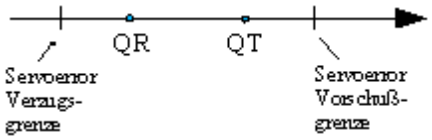
Beschreibung

Diese Anweisung weist der **Achse** die höchst zulässigen Loop- und Vorschuß-Fehler in positiver Richtung zu, bevor die Steuerung auf den "Servoerror" hinweist.

- **Verzug**: Das ist der Loop-Höchstwert im Fall eines statischen Servoerror-Tests, oder im Fall eines dynamischen Servoerror-Tests ist es der Wert, der bei Zusammenrechnen theoretischen Proportionalfehler und Geschwindigkeit den Loop-Höchstwert bestimmt.
- **Vorschuß**: Das ist der Loop-Höchstwert während der Umdrehung von negativer zu positiver Bewegungsrichtung.

Der Loop-Fehler ist der Unterschied zwischen theoretischem Maß (wo sich die Achse befinden sollte) und tatsächlichem Maß. Wenn sich die Achse in positiver Richtung bewegt, weist ein Loop-Fehler mit positivem Vorzeichen auf einen Verzug der Achse hin, während ein Loop-Fehler mit negativem Vorzeichen einen Vorschuß der Achse signalisiert. Verwendet man diese Anweisung nicht, geht die

Steuerung von den Höchstwerten des Loop-Fehlers gemäß der Achsenkonfiguration aus; in diesem Fall entspricht die Vorschußgrenze 1/4 des Verzugs.



Beispiel

```
;Der maximale Verzug der Achse beträgt 10mm , der maximale Vorschuß 5mm
SETMAXERPOS     Assi.X, 10, 5
```

SETMAXERTYPE

Syntax

SETMAXERTYPE **Achse, Typ**

Argumente

Achse Name der Vorrichtung Achsentyp
Typ Konstante Integer-Typ. Die zulässigen Werte sind:
0 = führt Schwelle-Servoerror (Default-Wert) ein
1 = führt dynamischen Servoerror ein

Beschreibung

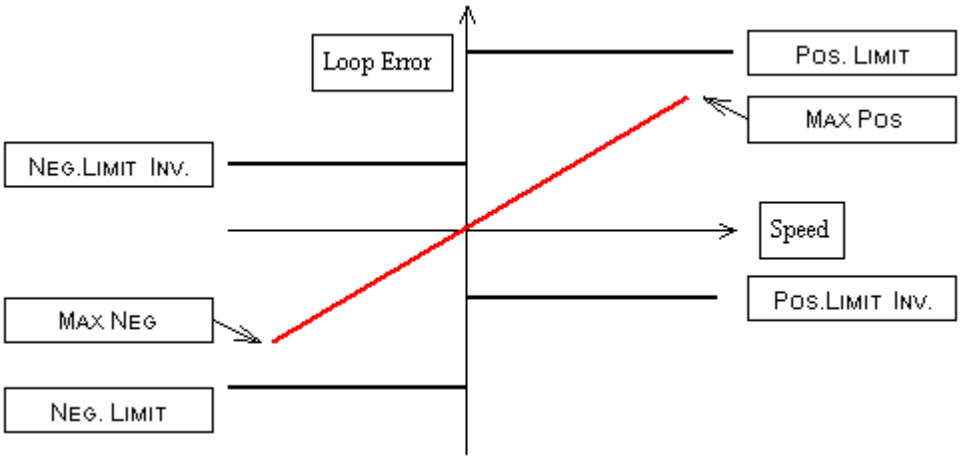
Diese Anweisung erlaubt es, der **Teststyp** auf dem Servoerror einzuführen. Die übliche Verwaltung des Servoerrors sieht ein Paar Grenzwerte (positiver und negativer) vor, die in Beziehung mit der Geschwindgkeitsänderung der Achse konstant sind. Dieser Verwaltungstyp schränkt die Grenzwerte abhängig von der höchsten Geschwindigkeit der Achse ein, d.h. der Grenzwert wird so eingeführt, dass es nicht möglich ist, dass ein Fehler mit normaler Funktionierung eintritt. Doch hat der Ring-Fehler mit geringen Geschwindigkeiten Werte, die im generell viel kleiner sind als den eingeführten Grenzwert. Darum bringt es eine Verspätung in der Identifizierung einer Fehler-Kondition mit sich.

Die Verwaltung des Fenster-Servoerrors basiert auf der Berechnung des theoretischen Ring-Fehlers. Die Grenzwerte des positiven oder negativen Servoerrors werden abhängig von diesem Servoerrors berechnet, indem ein Grenzwert ihnen summiert und abgezogen wird. Wenn der reelle Ring-Fehler diese Limite überschreit, ereignet sich der Servofehler-Fehler.

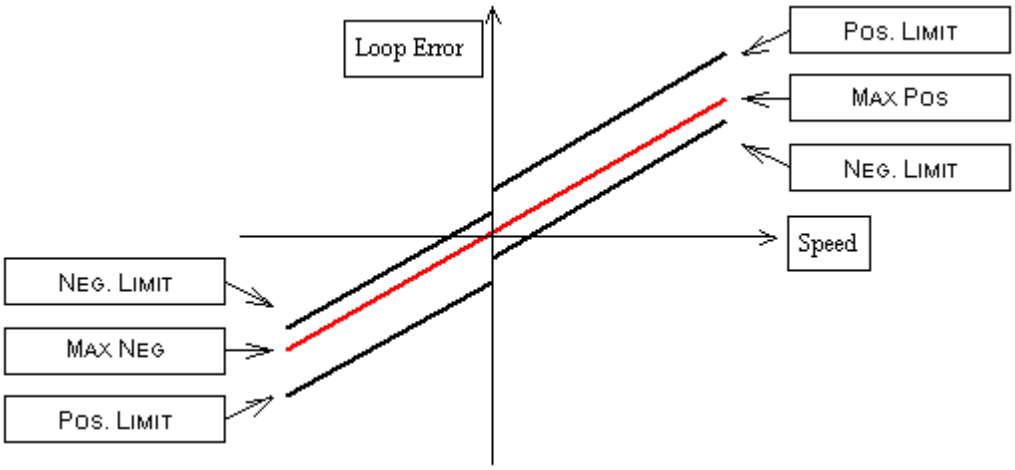
Bemerkung

Ist der Text auf dem dynamischen Servoerror eingestellt, dann sind die Grenzwerte des positiven Servoerrors und des negativen Servoerrors allgemein zu ändern, die in der Achsenkonfiguration des Schwelle-Servoerrors eingestellt sind. Die obengenannten Werte werden nämlich als Anfangswerte um den Loop-Fehler zu berechnen.

Üblicher Servoerror-Grenzwert:



"Fenster"-Servoerror-Grenzwert:



SETPHASESINV

Syntax

SETPHASESINV **Achse, Zustand**

Argumente

Achse	Name der Vorrichtung Achsentyp
Zustand	vorherbestimmte Konstante. Die zulässigen Werte sind: **ON** aktiver Phasen-Inversion-Zustand **OFF** entaktivierter Phasen-Inversion-Zustand

Beschreibung

Diese Anweisung aktiviert oder deaktiviert die Phasen-Inversion auf der angegebenen **Achse**. Eine eventuelle Verkabelung-Inversion der Encoder-Phasen kann mittels Softwares ausgeglichen werden. Die Achsenrichtung kann umgeschalten werden zusammen mit der Verwendung der Referenz-Inversion (wenn die Verkabelung richtig ist).
Um diese Anweisung auszuführen, muß die Achse in FREE-Zustand sein.

SETREFINV

Syntax

SETREFINV **Achse, Zustand**

Argumente

Achse	Name der Vorrichtung Achsentyp
Zustand	vorherbestimmte Konstante. Die zulässigen Werte sind: **ON** aktiver Referenz-Inversion-Zustand **OFF** entaktivierter Referenz-Inversion-Zustand **ON** aktiviert die Umkehrung der Bezugsgeschwindigkeit **OFF** deaktiviert die Umkehrung der Bezugsgeschwindigkeit

Beschreibung

Diese Anweisung aktiviert oder deaktiviert auf der angegebenen **Achse** die Umkehrung der Bezugsgeschwindigkeit. Die Achsenrichtung kann umgeschalten werden zusammen mit der Verwendung der Phasen-Inversion (wenn die Verkabelung richtig ist).
Um diese Anweisung auszuführen, muß die Achse in FREE-Zustand sein.
Siehe auch SETPHASESINV.

SETRESOLUTION

Syntax

SETRESOLUTION **Achse [, Wert]**

Argumente

Achse	Name einer Vorrichtung von Achsentyp
Wert	Konstante oder Variable Double

Beschreibung

Diese Anweisung wechselt die Auflösung der spezifizierten Achse. Wird der **Wert** ausgelassen, dann wird der in der Konfiguration eingestellte Auflösungswert verwendet. Der Auflösungswert kann geändert werden nur wenn die Achse stillstehend ist (Achsenzustand=Koordinate); ansonsten wird der Systemfehler Nr. 4101 "Unpassende Verwaltung der Achse" generiert.

10.3.5 Zähler

DECOUNTER

Syntax

DECOUNTER **ZählerName [, Wert]**

Argumente

ZählerName	Name der Vorrichtung des Typs Zähler
Wert	Konstante oder Variable oder Vorrichtung des Typs Zähler

Beschreibung

Diese Anweisung reduziert den Zähler **ZählerName** um den angegebenen **Wert**. Fehlt die Angabe des **Werts**, wird der Wert 1 angenommen. Siehe auch die Anweisungen SETCOUNTER und INCOUNTER.

INCOUNTER

Syntax

INCOUNTER **ZählerName [, Wert]**

Argumente

ZählerName	Name der Vorrichtung des Typs Zähler
Wert	Konstante oder Variable oder Vorrichtung des Typs Zähler

Beschreibung

Diese Anweisung erhöht den Zähler **ZählerName** um den angegebenen **Wert**. Fehlt die Angabe des **Werts**, wird der Wert 1 angenommen. Siehe auch die Anweisungen SETCOUNTER und INCOUNTER.

SETCOUNTER

Syntax

SETCOUNTER **ZählerName, Wert**

Argumente

ZählerName	Name der Vorrichtung des Typs Zähler
Wert	Konstante oder Variable oder Vorrichtung des Typs Zähler

Beschreibung

Diese Anweisung setzt den Zähler **ZählerName** auf den angegebenen **Wert**.
Siehe auch INCOUNTER und DECOUNTER.

10.3.6 Timer

HOLDTIMER

Syntax

HOLDTIMER **TimerName**

Argumente

TimerName	Name der Vorrichtung des Typs Timer

Beschreibung

Diese Anweisung hält die Aktualisierung des Timers **TimerName** an.
Siehe auch STARTTIMER und SETTIMER

SETTIMER

Syntax

SETTIMER **TimerName, Zeit**

Argumente

TimerName	Name der Vorrichtung des Typs Timer
Zeit	Konstante oder Variable oder Vorrichtung des Typs Timer

Beschreibung

Diese Anweisung setzt **TimerName** auf die angegebene **Zeit** (in Sekunden).
Es sind nur Werte größer als Null zulässig. Die höchste Genauigkeit der Timer beträgt 4 ms.
Siehe auch STARTTIMER und HOLDTIMER.

Beispiel

```
;Die Funktion setzt einen Timer
;der Timeout-Timer auf dem
;20-Sekunden-Wert eingestellt
SETTIMER        Timeout,20
; startet der Timer im absteigenden Modus. Wenn er 0 erreicht, hält er
an.

STARTTIMER      Timeout,DOWN
```

STARTTIMER

Syntax

STARTTIMER **TimerName [, Richtung]**

Argumente

TimerName	Name der Vorrichtung des Typs Timer
Richtung	vordefinierte Konstante. Die zulässigen Werte lauten: **UP** zunehmend **DOWN** abnehmend

Beschreibung

Diese Anweisung startet den Timer **TimerName** gemäß der eventuell in **Richtung** angegebenen Betriebsart.
Wird der **Richtungsp**arameter weggelassen, wird die Betriebsart **DOWN** angenommen.
Wenn ein (in abnehmender Betriebsart gestarteter) Timer die Null erreicht, hält er automatisch an.
Siehe auch HOLDTIMER und SETTIMER.

10.3.7 Variablen, Vektoren und Matrizen

CLEAR

Syntax

CLEAR **VarName oder Vektor oder Matrix[ZeileMatrix]**

Argumente

VarName	Name der Variablen
Vektor	Name des Vektors
Matrix	Name der Matrix
MatrixZeile	Konstante oder Variable oder Zähler. Zeile der Matrix

Beschreibung

Diese Anweisung initialisiert den für Variablen (**VarName**), Vektoren (**Vektor**), Matrizen (**Matrix**) oder für die Elemente einer Matrixzeile reservierten Speicherplatz mit Wert 0.

FIND

Syntax

FIND	**Matrix, Spalte, min_Grenze, max_Grenze, Wert, Variable**
FIND	**Vektor, min_Grenze, max_Grenze, Wert, Variable**

Argumente

Matrix	Name der Matrix. Matrix, in der gesucht werden soll
Vektor	Name des Vektors. Vektor, in dem gesucht werden soll
Spalte	Integer-Konstante oder -Variable oder ZählerName. Nummer der Matrixspalte, in der gesucht werden soll
min_Grenze	Konstante oder Variable. Minimaler Index des Vektors oder der Matrix, ab dem gesucht werden soll
max_Grenze	Konstante oder Variable. Maximaler Index des Vektors oder der Matrix, bis zu dem gesucht werden soll
Wert	Konstante oder Variable. Zu suchender Wert
Variable	Variable. Ergebnis der Suche

Beschreibung

Diese Anweisung führt eine sequentielle Suche eines Werts in einem **Vektor** oder in einer **Matrixspalte** aus und setzt den Index des Elements in **Variable**.
Wird der Wert nicht gefunden, wird **Variable** den Wert -1 enthalten.

FINDB

Syntax

FINDB	**Matrix, Spalte, min_Grenze, max_Grenze, Wert, Variable**
FINDB	**Vektor, min_Grenze, max_Grenze, Wert, Variable**

Argumente

Matrix	Name der Matrix. Matrix, in der gesucht werden soll
Vektor	Name des Vektors. Vektor, in dem gesucht werden soll
Spalte	Integer-Konstante oder -Variable oder ZählerName. Nummer der Matrixspalte, in der gesucht werden soll soll
min_Grenze	Konstante oder Variable. Minimaler Index des Vektors oder der Matrix, ab dem gesucht werden soll
max_Grenze	Konstante oder Variable. Maximaler Index des Vektors oder der Matrix, bis zu dem gesucht werden soll
Wert	Konstante oder Variable. Zu suchender Wert
Variable	Variable. Ergebnis der Suche

Beschreibung

Diese Anweisung führt eine schnelle Suche eines Werts in einem **Vektor** oder in einer **Matrixspalte** aus und setzt den Index des Elements in **Variable**. Damit die Suche erfolgreich verläuft, ist es notwendig, daß der **Vektor** oder die **Matrixspalte** vorher mit der Anweisung SORT nach einer zunehmenden Ordnung geordnet worden sind.
Wird der Wert nicht gefunden, wird **Variable** den Wert -1 enthalten.

LASTELEM

Syntax

LASTELEM	**Vektor, ElementeVek**
LASTELEM	**Matrix, ZeilenMat**

Argumente

Matrix	Name der Matrix
Vektor	Name des Vektors
ElementeVek	Variable. Anzahl Elemente des Vektors
ZeilenMat	Variable. Anzahl Zeilen der Matrix

Beschreibung

Diese Anweisung schreibt in die Variable **ElementeVek** die Anzahl Elemente des **Vektors** oder in die Variable **ZeilenMat** die Anzahl Zeilen der **Matrix**.

LOCAL

Syntax

LOCAL	**VarName AS Typ**
LOCAL	**Vektor[Anz Elemente] AS Typ**
LOCAL	**Matrix[Anz Zeilen] AS Typ, Typ, Typ, etc.**
LOCAL	**Matrix[Anz Zeilen] AS Typ:NameSpal1, Typ: NameSpal2, Typ:NameSpal3, etc.**

Argumente

VarName	Name der Variablen
[Anz Elemente]	Variable oder Konstante (obligatorisches Argument). Anzahl Elemente des Vektors
[Anz Zeilen]	Konstante oder Variable (obligatorisches Argument). Anzahl Zeilen der Matrix
Typ	Char, Integer (32 Bit), Float (32 Bit), Double (64 Bit), String, Timer
NameSpal1...NameSpalN	Name der Spalte. Etikett

Beschreibung

Deklaration einer lokalen Variablen. Vor dieser Anweisung kann nur die Anweisung PARAM erscheinen, die die Parameter der Funktion definiert.
Für weitere Informationen zu lokalen Variablen wird auf lokale Variablen verwiesen.

MOVEMAT

Syntax

MOVEMAT	**NameQuellMat, NameZielMat**
MOVEMAT	**NameQuellMat[QuellZeile], NameZielMat[ZielZeile],**
MOVEMAT	**NameQuellMat[QuellZeile], NameZielMat[ZielZeile],AnzZeile**

Argumente

NameQuellMat	Name der Quellmatrix
QuellZeile	Nummer der Startzeile zur Kopie der Quellematrix (obligatorisches Argument)
NameZielMat	Name der Ziel-Matrix
ZielZeile	Nummer der Startzeile zur Kopie auf die Ziel-Matrix (obligatorisches Argument)
AnzZeile	Anzahl der Zeilen zu kopieren

Beschreibung

Diese Anweisung kopiert den Inhalt einer ganzen Matrix **NameQuellMat** in eine andere Matrix **NameZielMat** oder in eine oder mehrere Zeile **AnzZeile** der Matrixzeile **NameQuellMat[QuellZeile]** in die Matrixzeile **NameZielMat[ZielZeile].** Wenn der Paramter **AnzZeile** ist nicht festgelegt, dann wird nur eine Zeile kopiert. Die zwei Matrizen müssen den gleichen Strukturtyp (Anzahl der Spalten und Datentyp in jeder Spalte) und im Falle vollständiger Matrizen auch die gleiche Anzahl von Zeilen haben. Es können auch Datenzeilen innerhalb derselben Matrix versetzt werden.

Beispiel

```
Movemat    Mx1, Mx2   ; kopiert die Matrix Mx1 in die Mx2

; kopiert die Zeile 10 der Matrix Mx1 in die Zeile 3 von Mx2
Movemat    Mx1[10], Mx2[3]

; kopiert die Zeile 1 der Matrix Mx1 in die Zeile 7 von Mx1
Movemat    Mx1[1], Mx1[7]
; kopiert 6 Zeilen ab der Zeile 2 der Matrix Mx1 in die Matrix
; Mx2 ab der Zeile 8
Movemat    Mx1[2], Mx2[8],6

; kopiert 4 Zeilen ab der Zeile 2
; der Matrix Mx1 in dieselbe Matrix
; Mx1 ab der Zeile 10
Movemat  Mx1[2], Mx1[10],4
```

PARAM

Syntax

[PARAM]	**VarName AS Typ**
[PARAM]	**Vektor[Anz Elemente] AS Typ**
[PARAM]	**Matrix[Anz Zeilen] AS Typ, Typ, Typ, etc.**
[PARAM]	**Matrix[Anz Zeilen] AS Typ:alias, Typ:alias, Typ:alias, etc.**

Argumente

VarName	Name der Variablen
[Anz Elemente]	Konstante (obligatorisches Argument)
[Anz Zeilen]	Konstante (obligatorisches Argument)
Typ	Char, Integer (32 Bit), Float (32 Bit), Double (64 Bit), String

Beschreibung

Die Parameter verhalten sich wie lokale Variablen (siehe LOCAL), werden aber von dem Code initialisiert, der die Funktion aufruft. Die Syntax zur Deklaration der Parameter ist dieselbe wie für lokale Variablen. Die Parameter werden je nach Typ als Wert oder als Bezug übergeben.
Siehe "Funktionen".
Sie sind vor sämtlichen anderen Anweisungen zu deklarieren.
Für weitere Informationen wird auf lokale Variablen verwiesen.

SETVAL

Syntax

SETVAL	**Wert, VarName**

Argumente

Wert	Konstante oder Variable oder VorrichtungName
VarName	Variable oder VorrichtungName

Beschreibung

Diese Anweisung ordnet den agegebenen **Wert** der Variablen **VarName** oder dem i-ten Element eines Vektors oder einer Matrix zu.

SORT

Syntax

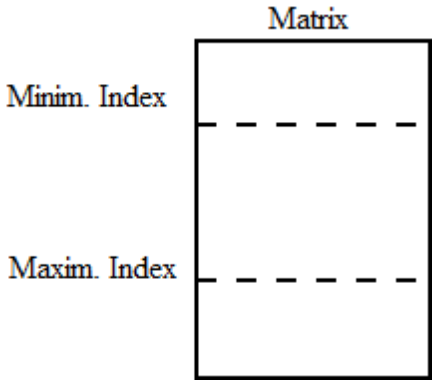
SORT	**Matrix, Spalte [, Richtung], min_Grenze, max_Grenze**
SORT	**Vektor [,Richtung], min_Grenze, max_Grenze**

Argumente

Matrix	Name der Matrix.
Vektor	Name des Vektors.
Spalte	Konstante oder Variable. Spalte Zeilen der Matrix
Richtung	vordefinierte Konstante. Gibt die Beschaffenheit von Richtung Die zulässigen Werte lauten: **UP** zunehmender Anordnung **DOWN** abnehmender Anordnung
min_Grenze	Konstante oder Variable. Minimaler Index des Vektors oder der Matrix, ab dem anordnet werden soll
max_Grenze	Konstante oder Variable. Maximaler Index des Vektors oder der Matrix, bis zu dem anordnet werden soll

Beschreibung

Diese Anweisung ordnet die Werte in einem **Vektor** oder in einer **Matrix**, wobei die Richtung von der Konstanten **Richtung** definiert wird.
Im Fall einer Matrix ist das Ordnen der Zeilen von der zu- (UP) oder abnehmenden Anordnung (DOWN) der Werte in der gewählten **Spalte** abhängig.
Wird das Argument **Richtung** ausgelassen, wird automatisch die Art UP angenommen.



10.3.8 Zeichenfolgen

ADDSTRING

Syntax

ADDSTRING	**String1Name, String2Name, String3Name**

Argumente

String1Name	String-Konstante oder Variable. Quell-String
String2Name	String-Konstante oder Variable. Hinzuzufügender String
String3Name	String-Variable. Ergebnis-String

Beschreibung

Verkettung von zwei Strings.
Diese Anweisung fügt zum von **String1Name** bestimmten String den von **String2Name** bestimmten String hinzu und setzt das Ergebnis in den von **String3Name** bestimmten String.
Die maximale Größe eines Strings beträgt 255 Zeichen+ Endmarke, d.h. das Ergebnis der Verkettung der ersten zwei Strings hat diese Grenze nicht zu überschreiten.

Beispiel

Vorgänge an Strings

CONTROLCHAR

Syntax

CONTROLCHAR	**Wert, StringName**

Argumente

Wert	Char- oder Integer-Konstante oder Char- oder Integer-Variable. Umzuwandelnder Wert
StringName	String-Variable. Ergebnis String

Beschreibung

Diese Anweisung wandelt den von **Wert** bestimmten Wert in ein ASCII-Zeichen um und setzt das Ergebnis in den String **StringName** (praktisch in das erste Byte).
Der vorherige Inhalt des Strings geht verloren. Diese Anweisung ist nützlich, wenn Kontrollzeichen oder nicht abdruckbare Zeichen (z.B. das Zeichen NULL = 0x00) in einen String einzufügen sind.
Akzeptiert Strings von mindestens 2 Zeichen: 1 Zeichen + Endmarke. Wenn der String nur aus einem Zeichen besteht array[1] as char, wird ein Systemfehler "Maktro-Argument ist fehlerhaft" gemeldet.

Beispiel

Vorgänge an Strings

LEFT

Syntax

LEFT	**QuellStringName, ZeichenAnz, ZielStringName**

Argumente

QuellStringName	String-Konstante oder -Variable. Quell-String
ZeichenAnz	Konstante oder Variable. Anzahl der zu kopierenden Zeichen

ZielStringName	String-Variable. Ziel-String

Beschreibung

Diese Anweisung kopiert die ersten **ZeichenAnz** des Strings **QuellStringName** in den String **ZielStringName**.
Praktisch wird der Teil links vom Quell-String entnommen. Siehe auch die Anweisungen MID und RIGHT.

Beispiel

Vorgänge an Strings

LEN

Syntax

LEN	**StringName, Variable**

Argumente

StringName	String-Variable. String
Variable	Variable

Beschreibung

Diese Anweisung berechnet die Anzahl im String **StringName** enthaltener Zeichen (unter Ausschluß des Schlußzeichens) und setzt sie in **Variable**.

Beispiel

Vorgänge an Strings

MID

Syntax

MID	**QuellStringName, primocar [, ZeichenAnz], ZielStringName**

Argumente

QuellStringName	String-Konstante oder -Variable. Quell-String
ZeichenAnz	Konstante oder Variable. Anzahl der zu kopierenden Zeichen
ZielStringName	String-Variable. Ziel-String
ErstZeich	Konstante oder Variable. Position des Zeichens zum Beginn der Kopie

Beschreibung

Diese Anweisung entnimmt aus dem von **QuellStringName** bestimmten String beginnend an der Position **ErstZeich** eine von **ZeichenAnz** bestimmte Anzahl Zeichen.
Der entnommene Unterstring wird in den von **QuellStringName** bestimmten String gesetzt.
Wird **ZeichenAnz** ausgelassen, wird **QuellString** beginnend bei der Position **ErstZeich** bis zum Ende desselben kopiert. Praktisch wird der mittlere Teil vom Quell-String entnommen. Siehe auch die Anweisungen LEFT und RIGHT.

Beispiel

Vorgänge an Strings

RIGHT

Syntax

RIGHT	**QuellStringName, ZeichenAnz, ZielStringName**

Argumente

QuellStringName	String-Konstante oder String-Variable. String Quelle
ZeichenAnz	Konstante oder Variable. Anzahl der zu kopierenden Zeichen
ZielStringName	Variable String. Ziel-String

Beschreibung

Diese Anweisung kopiert die letzten **ZeichenAnz** des Strings **QuellStringName** in den String **ZielStringName**.
Praktisch wird der Teil rechts vom Quell-String entnommen. Siehe auch die Anweisungen LEFT und MID.

Beispiel

Vorgänge an Strings

SEARCH

Syntax

SEARCH	**StringName, Zeichen, Variable**

Argumente

StringName	Variable String.
Zeichen	costante char o String-Konstante oder String-Variable. Zeichen oder String zu suchen
Variable	Variable

Beschreibung

Diese Anweisung sucht die Position des von **Zeichen** (das auch ein String sein kann) bestimmten ASCII-Zeichens innerhalb des Strings **StringName** und setzt den Index des Ergebnisses in die **Variable**.
Wird **Zeichen** nicht gefunden, wird **Variable** den Wert -1 enthalten.

Beispiel

Vorgänge an Strings

SETSTRING

Syntax

SETSTRING	**"Wert", StringName**

Argumente

Wert	String-Konstante oder -Variable (zwischen Anführungszeichen)
StringName	Ziel-String

Beschreibung

Kopie eines Strings.
Diese Anweisung kopiert die im von "**Wert**" bestimmten String enthaltenen Zeichen in den von **StringName** bestimmten String.
Um nicht abdruckbare Zeichen in einen String einzufügen, wird auf die Anweisung CONTROLCHAR verwiesen.

Beispiel

Vorgänge an Strings

STR

Syntax

STR	**Wert, StringName**

Argumente

Wert	Konstante oder Variable. Umzuwandelnder Quellwert
StringName	String-Variable. Ziel-String

Beschreibung

Diese Anweisung wandelt den **Wert** in ASCII-Zeichen um und setzt das Ergebnis in den String **StringName**. Kann zur Umwandlung einer Variablen des Typs Integer in einen String verwendet werden. Die Ziffer 10 wird z.B. in den String "10" umgewandelt.

Beispiel

Vorgänge an Strings

VAL

Syntax

VAL	**StringName, Ergebnis**

Argumente

StringName	String-Variable. Umzuwandelnder String
Ergebnis	Variable. Umgewandelter String

Beschreibung

Diese Anweisung wandelt den Inhalt des Strings **StringName** in eine Dezimalzahl um und setzt das Ergebnis in die **Variable**.
Der String "123" wird z.B. in den Wert 123 umgewandelt.

Beispiel
Vorgänge an Strings

10.3.9 Kommunikationen

CLEARRECEIVE

Syntax
CLEARRECEIVE

Argumente
Kein Argument

Beschreibung
Diese Anweisung leert die Liste der empfangenen, aber noch nicht erfüllten RECEIVE.

COMCLEARRXBUFFER

Syntax
COMCLEARRXBUFFER **COMNummer**

Argumente
COMNummer vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**

Beschreibung
Diese Anweisung leert den Empfangs-Pufferspeicher des seriellen Ports **COMNummer**. Alle eventuell vorhandenen Daten werden gelöscht.

COMCLOSE

Syntax
COMCLOSE **COMNummer**

Argumente
COMNummer vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**

Beschreibung
Diese Anweisung schließt die von einer **COMOPEN** geöffnete serielle Linie **COMNummer**. Die Schließung der seriellen Linie ist auch dann notwendig, wenn der Task, der die serielle Linie geöffnet hat, aus einem beliebigen Grund beendet wird.

COMGETERROR

Syntax
COMGETERROR **COMNummer, Variable**

Argumente
COMNummer vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**
Variable Integer-Variable. Ergebnis der letzten am seriellen Port ausgeführten Vorgang

Beschreibung
Diese Anweisung liest den Rückkehrcode der letzten Anweisung zur seriellen Kommunikation, die vom Port **COMNummer** aufgerufen wurde. Anhand dieser Anweisung erfährt man, ob ein Les- oder Schreibvorgang erfolgreich gewesen ist. Wenn nicht, erfährt man, welcher Fehlercode zurückgegeben worden ist.
Nachfolgend sind die Fehlercodes aufgelistet:

Normaler Rückkehrwert	0
Sende-Pufferspeicher voll	2
Vorrichtung bereits offen	3
Ungültiger oder nicht konfigurierter Port	6
Aktivierung des I/0-Ports gescheitert	7
Anschluß an den Interrupt nicht möglich	8
Serieller Port (com) noch nicht offen	9
Serielle Vorrichtung (com) belegt	12
Anschluß an RTX nicht möglich	14

COMGETRXCOUNT

Syntax

COMGETRXCOUNT **COMNummer, CharAnz**

Argumente

COMNummer vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**

CharAnz Anzahl im Pufferspeicher vorhandener Zeichen

Beschreibung

Diese Anweisung gibt die Anzahl der im Empfangs-Pufferspeicher vorhandener Zeichen zurück. Anhand der Anweisung erfährt man, ob der serielle Port Zeichen empfangen hat.

COMOPEN

Syntax

COMOPEN **COMNummer, BaudRate, WordSize,StopBits,Parität**

Argumente

COMNummer vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**

BaudRate Baud rate der Kommunikation. Die zulässigen Werte lauten: 2400, 4800, 9600, 19200, 38400, 57600, 115200

WordSize Größe des Datenworts. Die zulässigen Werte lauten: 5,6,7,8

StopBits Stop-Bit. Die zulässigen Werte lauten: 1, 2

Parität vordefinierte Konstante. Parität. Die zulässigen Werte lauten: **NOPARITY**, **ODDPARITY** und **EVENPARITY**

Beschreibung

Diese Anweisung öffnet eine serielle Linie. Ist vor sämtlichen anderen Anweisungen zur Verwaltung einer seriellen Linie auszuführen. Wird eine andere, sich auf dieselbe serielle Linie beziehende Anweisung vor COMOPEN ausgeführt, erscheint ein Systemfehler. Die übergebenen Parameter haben den oben angegebenen Werten zu entsprechen.
Siehe auch COMCLOSE, COMREAD, COMWRITE, COMREADSTRING, COMWRITESTRING.

Bemerkung

Die Zahl der verfügbaren seriellen Linien ist mit der Hardware-Umgebung der Numerischen Steuerung verbunden (s. Dokumentation). Im RTW - Umgebung sind nur COM1 und COM2 verfügbar.

COMREAD

Syntax,

COMREAD **COMNummer, Pufferspeicher, ZuLesCharAnz,AnzGelesenerChar [,Timeout]**

Argumente

COMNummer vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**

Pufferspeicher Char-Vektor. Es ist der Vektor, in dem die Daten abgelegt werden.

ZuLesCharAnz Anzahl Zeichen, die man von der seriellen Linie ablesen möchte

AnzGelesenerChar Anzahl tatsächlich abgelesener Zeichen

Timeout Wartedauer (in Sekunden)

Beschreibung

Diese Anweisung liest Zeichen von der seriellen Linie **COMNummer** ab. Die abgelesenen Zeichen werden in der Variablen **Pufferspeicher** gespeichert. Das Feld **ZuLesCharAnz** gibt die Anzahl Zeichen an, die die Anweisung abzulesen hat. Sind im Empfangs-Pufferspeicher weniger Zeichen und ist der Parameter **Timeout** nicht angegeben, endet die Anweisung sofort unter Angabe der tatsächlich abgelesenen Zeichenanzahl im Parameter **AnzGelesenerChar**. Ist der Parameter **Timeout** angegeben, hat die Anweisung maximal die Anzahl in der Variablen angegebener Sekunden abzuwarten, daß weitere Zeichen kommen. Läuft das **Timeout** ab, endet die Anweisung unter Angabe der tatsächlich in den **Pufferspeicher** kopierten Zeichenanzahl im Parameter **AnzGelesenerChar**.

COMREADSTRING

Syntax

COMREADSTRING	**COMNummer, Pufferspeicher, AnzGelesenerChar [,Schlußzeichen [,Timeout]]**

Argumente

COMNummer	vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**
Pufferspeicher	Char-Vektor. Es ist der Vektor, in dem die Daten abgelegt werden.
AnzGelesenerChar	Anzahl tatsächlich abgelesener Zeichen
Schlußzeichen	Schlußzeichen der Sendung
Timeout	Wartedauer (in Sekunden)

Beschreibung

Diese Anweisung liest Zeichen von der seriellen Linie **COMNummer** ab. Sie unterscheidet sich von COMREAD dadurch, daß bis zum Schlußzeichen von der seriellen Linie abgelesen wird. Die abgelesenen Zeichen werden in der Variablen **Pufferspeicher** gespeichert. Diese Variable hat des Typs Char-Vektor zu sein. Das Feld **AnzGelesenerChar** gibt die Anzahl Zeichen an, die die Anweisung tatsächlich von der seriellen Linie abgelesen und in den **Pufferspeicher** kopiert hat. Der Parameter **Schlußzeichen** gibt das Zeichen an, das als Schlußzeichen der Sendung wirkt. Die Anweisung wird praktisch so lange Zeichen von der seriellen Linie ablesen bis sie auf dasselbe Zeichen stößt wie in diesem Parameter angegeben. Dieser Parameter ist optionell. Ist kein Zeichen angegeben, versteht sich Null als Schlußzeichen. Das Zeichen Null wird nicht in den als Parameter übergebenen Pufferspeicher kopiert, während das eventuell in der Anweisung angegebene andere Schlußzeichen in den Pufferspeicher kopiert wird. **Timeout** ist ein weiterer Parameter, der angibt, wieviel Sekunden die Anweisung weitere Zeichen abzuwarten hat, nachdem sie den Pufferspeicher geleert hat, ohne das Schlußzeichen gefunden zu haben. Ist der Parameter **Timeout** nicht angegeben, endet die Anweisung, sobald der Empfangs-Pufferspeicher geleert worden ist.

COMWRITE

Syntax

COMWRITE	**COMNummer, Pufferspeicher, ZuSchreiben**

Argumente

COMNummer	vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**
Pufferspeicher	Char-Vektor. Es ist der Vektor, der die zu schreibenden Daten enthält.
ZuSchreiben	Anzahl zu schreibender Zeichen

Beschreibung

Diese Anweisung schreibt die in der Variablen **Pufferspeicher** vorhandenen Zeichen in die serielle Linie **COMNummer**. Der Parameter **ZuSchreiben** enthält die Anzahl tatsächlich zu schreibender Zeichen.

COMWRITESTRING

Syntax

COMWRITESTRING	**COMNummer, Pufferspeicher [,Schlußzeichen]**

Argumente

COMNummer	vordefinierte Konstante. Nummer des seriellen Ports. Die zulässigen Werte lauten: von **COM1** bis **COM8.**
Pufferspeicher	Char-Vektor. Es ist der Vektor, der die zu schreibenden Daten enthält
Schlußzeichen	Schlußzeichen der Sendung

Beschreibung

Diese Anweisung schreibt die in der Variablen Pufferspeicher vorhandenen Zeichen in die serielle Linie **COMNummer**. Sie unterscheidet sich von COMWRITE dadurch, daß bis zum **Schluß** zeichen in die serielle Linie geschrieben wird. Das Zeichen **Schlußzeichen** ist optionell. Ist kein Zeichen angegeben, versteht sich Null als Schlußzeichen. Das Zeichen Null wird nicht übertragen, während das eventuell angegebene Schlußzeichen übertragen wird.

RECEIVE

Syntax

RECEIVE **[Quelle,] Identifizierer, Flags [, Behälter]**

Argumente

Quelle	Konstante des Typs String
Identifizierer	Konstante des Typs String
Flags	Konstante des Typs Integer
Behälter	Name einer Vorrichtung oder Variablen (numerisch oder String)

Beschreibung

Zusammen mit der Anweisung SEND dient diese Anweisung dem Austausch von Informationen zwischen den Modulen der Anlage und dem Supervisor-PC. SEND wird zum Senden von Informationen verwendet; RECEIVE hingegen, um Informationen anzufordern. Die Informationen können von Albatros oder einem externen Programm (OLE Automation Server) abgefragt werden. Im letzteren Fall wird die Anforderung auf jeden Fall von Albatros empfangen und an das externe Programm weitergeleitet.

Der Parameter **Quelle** ist ein String zur Angabe, an wen die Anforderung nach Informationen adressiert ist. Es gibt drei Klassen von Quellen:

- Quellen, die mit dem Zeichen "@" beginnen (siehe Liste weiter vorne). Die Quelle ist genau genommen Albatros bzw. eine seiner Funktionen.
- Quellen, die nicht mit dem Zeichen "@" beginnen. Sie werden wie OLE Server angesehen. Bei Erhalt der ersten, an sie adressierten Anforderung von Informationen versucht Albatros, sie auszuführen und die vom Modul erhaltene Informationsanforderung an sie zu übertragen.
- nicht angegebene Quelle (es handelt sich um einen optionellen Parameter). In diesem Fall wird die Information in einer von Albatros aufbewahrten Tabelle abgelesen. Ist die Information nicht in der Tabelle vorhanden, bleibt die Anforderung so lange unerfüllt, bis die Information verfügbar ist (von einem anderen Modul oder von einem externen Programm geliefert).

Der Parameter **Identifizierer** ist der Name der angeforderten Information und kann nicht ausgelassen werden. Er nimmt je nach Quelle verschiedene Bedeutungen an:

- ist die Quelle Albatros, handelt es sich um einen Befehl, der mit der aufgerufenen Funktion zusammenhängt
- ist die Quelle ein OLE Server, handelt es sich um eine Eigenschaft des angefragten OLE-Objekts.
- ist die Quelle nicht angegeben, handelt es sich um das Etikett, das die Information innerhalb der von Albatros aufbewahrten Tabelle identifiziert.

Der Parameter **Flags** dient der Angabe, wie die angeforderte Information von Albatros behandelt werden soll. Die zulässigen Werte und deren Wirkung sind folgende:

Wert	***Befehl***	***Beschreibung***
$0008H	CancelAfter	Nach dem Lesen wird die Information gelöscht
$0800H	UpdateFlags	Ändert den Zustand der Information (gelesen / zu lesen), ohne die Daten zu ändern
$8000H	Delete	Löscht die Information

Der Parameter **Behälter** ist die Variable (oder Vorrichtung), in der die angeforderte Information gespeichert wird. Wird diese Angabe ausgelassen, fordert man die Bekanntgabe eines Ereignisses (kann zur Synchronisierung der Ausführung des GPL-Codes an verschiedenen Modulen verwendet werden).

Liste der von Albatros verwalteten **Quellen** und der zugehörigen Befehle:

"@List"

Ermöglicht die Kontrolle der Befehle Simulation und Setpoint.
Folgende Befehle (Parameter **Bezeichner**) sind zugelassen:

- Sim,0,Behälter: Erfordert den Zustand der Schaltfläche Simulation, die im FlagSwitch Simulation geschrieben wird. Die Rückgabevariable **Behälter** gilt 1, wenn keine Fehler aufgetreten sind, ansonsten gilt sie 0.

- Setp,0,Behälter:Erfordert den Zustand der Schaltfläche Setpoint, der im FlagSwitch CmdSetP geschrieben wird. Die Rückgabevariable **Behälter** gilt 1, wenn keine Fehler aufgetreten sind, ansonsten gilt sie 0.
- Esc, o Behälter: Erfordert den Zustand der Schaltfläche Setpoint, der im FlagSwitch Escluso geschrieben wird. Esc, o Behälter: Erfordert den Zustand der Schaltfläche Setpoint, der im FlagSwitch Escluso geschrieben wird.

"@Environ"

Ermöglicht es, Informationen über den Zustand des Systems zu erhalten: Zugangsebene des Benutzers, an den Supervisor angeschlossene Module, usw. Die angeforderte Information wird im Parameter **Behälter** gespeichert. Die zulässigen Werte für den Parameter **Identifizierer** und die zugehörigen Rückkehrwerte lauten:

- "AccessLevel" Zugangsebene zum System 0=Benutzer, 1=Wartung, 2=Hersteller, 3=tpa
- "MaskConfModules" Maske der konfigurierten Module
- "MaskActiveModules" Maske der angeschlossenen Module
- "CurrentModule" Modul, von dem die Anforderung stammt
- "*mod*:NamePC" Name des PCs, der dem Modul "*mod*". entspricht (*mod* zwischen 0 und 15 inbegriffen)
- "LocalDateTime" Datum und Zeit des PCs im JJJJ/MM/TT SS:MM:SS-Format

Der Parameter Behälter wird das Datum und die Uhrzeit der PC entsprechend einem mit seinem Typ verbundenen Typ erhalten:

- char: Zahl des Tages der Woche
- integer: Anzahl von Sekunden seit dem 1/1/1970
- float: Anzahl von Tagen und Fraktionen eines Tages seit dem 1/1/1900
- double: Anzahl von Tagen und Fraktionen eines Tages seit dem 1/1/1900
- string: Text "JJJJ/MM/TT hh:mm:ss"

Die Masken der angeschlossenen und konfigurierten Module sind Bit-Masken. Das Bit mit dem kleinsten Stellenwert entspricht dem Modul 0. Das Bit jedes angeschlossenen oder konfigurierten Moduls ist 1. Im Fall von "NamePC" kann von der Modulnummer abgesehen werden. Läßt man sie aus, wird die Nummer des Moduls angenommen, von dem die Anforderung stammt.

"@Syn"

Kommunikation zwischen GPL und der Anzeige der synoptischen Darstellungen. Dient dem Öffnen und Schließen von synoptischen Darstellungen mittels GPL und der Anforderung von Informationen aus Feldern von synoptischen Darstellungen. Die zulässigen Befehle lauten (Parameter **Identifizierer**):

- "Open:*Dateiname*" öffnet die synoptische Datei *Dateiname.xsyn*
- "Close:*Dateiname*" schließt die synoptische Datei *Dateiname.xsyn*
- "*Feldname*" Feld, aus dem die angeforderte Information gelesen wird

Es können Informationen über das Fenster der Achsenbewegung gemäß den Spezifikationen auch für den unten wiedergegebenen Parameter **Quelle** "@Devices" erhalten werden.

"@FileName"

Speichert eine Bindung zwischen der Konstante einer Zeichenfolge und einer Datei, deren Namen mit Stringvariablen zusammengestellt werden kann. Nachdem Albatros den Bindungsbericht erhalten hat, wird jeder nachkommende Dateiname mit dem durch diese Anweisung empfangenen Namen ersetzt. Der Parameter **Identifizierer** ist der Name der Datei. Der Name der Datei ist eine Zeichenfolgevariable. Ist im Parameter Identifizierer der vollständige Pfad nicht angegeben in dem die Datei zu archivieren ist, dann wird als Pfad betrachtet, der der in tpa.ini unter dirreport definiert ist. Der Wert des Parameters Identifizierer wird in tpa.ini in dem Abschnitt [GPLFileName] unter *Log* gespeichert, um ihn in den nachkommenden Ausführungen von Albatros gebrauchen zu können. Um die Bindung zu entfernen, muss man als Parameter Identifizierer eine leere Zeichenfolge einstellen. Die in dieser Weise definierte Bindung gilt für alle Module.

"@FileDelete"

Löschen einer Datei. Der Parameter **Identifizierer** ist der Name der Datei, die gelöscht wird (vollständiger Pfad). Falls im Identifikationsparameter der vollständige Pfad in dem die Datei zu archivieren ist nicht angegeben ist, dann wird als Pfad betrachtet, der der in tpa.ini unter *dirreport* definiert ist.
Der Dateiname kann gemäß den in Übereinstimmung mit dem Parameter **Quelle** @FileRead beschriebenen Regeln definiert werden. Der Parameter **Behälter** enthält den Wert:

- 1, wenn die Datei gelöscht worden ist
- 0 im gegenteiligen Fall

"@FileRead"

Liest den Inhalt einer Datei. Der Parameter **Identifizierer** ist der Name der Datei, die gelesen wird (vollständiger Pfad). Falls im Identifikationsparameter der vollständige Pfad, in dem die Datei zu archivieren ist, nicht angegeben ist, dann wird als Pfad betrachtet, der der in tpa.ini unter *dirreport* definiert ist. Wenn der Identifizierer mit einem Zeichen % beginnt und endet, wird der String darin in tpa.ini im Abschnitt [tpa] gesucht und als Dateiname verwendet. Im Namen können Zeichen, die während der Ausführung der Anweisung ersetzt werden, eingegeben werden:

- %n Nummer des Moduls, das die Anweisung RECEIVE ausführt
- %h aktuelle Uhrzeit (Format 00-23)
- %d aktueller Tag (Format 01-31)
- %m aktueller Monat (Format 01-12)
- %y aktuelles Jahr (im vierstelligen Format)

Wenn der Parameter **Behälter** als Variable Char definiert wurde, enthält er ein von der Datei gelesenes Byte, wenn er als String definiert wurde, enthält er eine ganze Zeile der Textdatei wenn er als Integre definiert wurde, enthält er die Byte-Nummer, die am Ende der Datei fehlen (0 = Dateiende).
Zur Positionierung des Cursors auf der Datei am Dateianfang muss der Parameter **Behälter** ausgelassen werden.

"@FileExist"

Prüft das Bestehen einer Datei. Der Parameter **Identifizierer** ist der Name der Datei, die gelesen wird (vollständiger Pfad). Falls im Identifikationsparameter der vollständige Pfad in dem die Datei zu archivieren ist nicht angegeben ist, dann wird als Pfad betrachtet, der der in tpa.ini unter *dirreport* definiert ist. Der Dateiname kann gemäß den in Übereinstimmung mit dem Parameter **Quelle** @FileRead beschriebenen Regeln definiert werden. Der Parameter **Behälter** enthält den Wert:

- anders als 0, wenn es die Datei gibt;
- 0, wenn es die Datei nicht gibt.

"@FileLastWrite"

Ruft das Datum der letzten Änderung der Datei ab. Der Parameter **Identifizierer** ist der Name der Datei (vollständiger Pfad). Ist im Parameter Identifizierer der vollständige Pfad nicht angegeben, in dem die Datei zu archivieren ist, dann wird als Pfad betrachtet der, der in dem Abschnitt [tpa] unter *dirreport* definiert ist. Der Dateiname kann gemäß den in Übereinstimmung mit dem Source-Parameter @FileRead beschriebenen Regeln definiert werden. Der Parameter **Behälter** wird das Datum der letzten Änderung der Datei gemäß einem Format enthalten, der mit dem Parametertyp verbunden ist:

- char: Nummer des Tages der Woche
- integer: Anzahl von Sekunden ab dem 1. Januar 1970
- float: Anzahl von Tagen und Bruchteile eines Tages ab dem 1. Januar 1900
- double: Anzahl von Tagen und Bruchteile eines Tages ab dem 1. Januar 1900
- string: Text im Format "AAAA/MM/GG hh:mm:ss"

"@FileInfo"

Es liest einige Informationen aus einer Datei. Der Parameter **Identifizierer** muss in der Form "Eigenschaft:Dateiname" ausgedrückt werden, wobei **Eigenschaft** den Namen der zu lesenden Eigenschaft angibt und **Dateiname** der Name der Datei ist. Der Dateiname kann durch "Namen" eingestellt werden. Der Parameter **Behälter** enthält die aus der Datei gelesenen Daten:

- "version:": gibt einen ganzzahligen Datentyp in einem Behälter zurück. Die vier Zahlen, die die Version identifizieren, befinden sich in den 4 Bytes der Behältervariablen. Wenn ein Fehler auftritt, ist der Wert der Behältervariablen 0.
- "size:": gibt einen ganzzahligen oder Float- oder Double-Datentyp in einem Behälter zurück. Die Daten sind die Dateigröße. Wenn ein Fehler auftritt, ist der Wert der Behätervariablen -1.

"@Devices"

Anforderung, das Diagnostik-Fenster des Moduls, das die Anforderung nach Information sendet, zu öffnen oder zu schließen. Der Parameter Identifizierer kann folgende Werte annehmen:

- "Open" Öffnen der Diagnostik
- "Close" Schließen der Diagnostik

Der Parameter **Identifizierer** kann die folgenden Werte annehmen, wenn mit dem Fenster der Achsenbewegung interagiert werden soll:

"MoveAX#Name_Achse#HasFocus"	der Parameter **Behälter** enthält 1, wenn das Fenster der spezifizierten Achsenbewegung eingeschaltet ist, andernfalls enthält er 0.

"MoveAX#Name_Achse#Jog"	der Parameter **Behälter** enthält 1, wenn die Bewegung für die Versetzungen der Verwaltungen der Laufzeit vom Benutzer eingestellt wird, andernfalls enthält er 0.
"MoveAX#Name_Achse#Step"	der Parameter **Behälter** enthält 1, wenn die Bewegung für die Versetzungen der vorbestimmten Steigung eingestellt wird, andernfalls enthält er 0.
"MoveAX#Name_Achse#Absolute"	der Parameter **Behälter** enthält 1, wenn die Bewegung mit Versetzung mit festgelegter Höhe eingestellt wird, andernfalls enthält er 0.

wobei Name_Achse den Namen der im Fenster angezeigten Achse bezeichnet. Wenn zum Beispiel geprüft werden soll, ob das Fenster der Bewegung der X-Achse eingeschaltet ist, ist der Parameter **Identifizierer** "@MoveAX#X#HasFocus". Der Achsenname kann in einer der folgenden Formen sein:

1. Name_Gruppe.Name_Untergruppe.Name_Achse oder Name_Gruppe.Name_Achse: Es wird der vollständige Verlauf der Achse geliefert.
2. Name_Achse: Zur Identifizierung der richtigen Achse werden die folgenden Prüfungen in Folge gemacht:
   - Wenn die Task, aus der der Befehl stammt, eine Funktion der Untergruppe ist, wird die Achse in dieser Untergruppe gesucht.
   - Wenn die Task, aus der der Befehl stammt, eine Funktion der wichtigsten Untergruppe ist, wird die Achse in der gesamten Gruppe gesucht. Wenn mehr als eine Achse mit diesem Namen vorhanden ist, schlägt die Suche fehl.
   - Wenn die vorherigen Prüfungen fehlgeschlagen sind, wird die Achse in allen Gruppen des Moduls gesucht. Wenn mehr als eine Achse mit dem Namen Name_Achse vorhanden ist, hat die Suche kein positives Ergebnis.

"@Vars"

Fordert die Aktualisierung einer globalen GPL-Variablen. Dient der Auffrischung der Daten der technologischen Parameter und Werkzeuge. Die Parameter-Daten werden normalerweise während der Initialisierung der Maschine an GPL gesendet. Der Parameter **Identifizierer** nimmt die Bedeutung des Namens der globalen (Maschinen- oder Gruppen-) Variablen an, deren Aktualisierung angefordert wird. Der Parameter **Behälter** enthält den Wert:

- 1, wenn die Variable korrekt aktualisiert worden ist
- 0 im gegenteiligen Fall

"@Application"

Interaktion mit Albatros· Dient der Anzeige von Nachrichtenfenstern (message boxes) auf dem Bildschirm und dem Schließen von Albatros· Die zulässigen Werte für den Parameter **Identifizierer** lauten:

"Quit"	schließt Albatros
"IsLocked"	überprüft, wenn das Beenden von Albatros gesperrt ist. Der Parameter **Behälter** wird 1 enthalten, wenn die Schnittstelle gesperrt ist, 0, wenn es möglich ist, Albatros zu beenden.
"MsgBox*Flags*"	liest die Antwort einer zuvor durch SEND-Meldung geöffneten MessageBox

Anhand des Parameters **Behälter** läßt sich ermitteln, welche Schaltfläche der Benutzer im Fall eines Nachrichtenfensters gedrückt hat:

- 1 Schaltfläche "OK"
- 2 Schaltfläche "Abbrechen"
- 4 Schaltfläche "Retry"
- 6 Schaltfläche "Ja"
- 7 Schaltfläche "Nein"

Im Fall des Befehls "Quit" enthält der Parameter **Behälter** den Wert:

- 1, wenn Albatros korrekt geschlossen worden ist
- 0 im entgegengesetzten Fall

"@Param"

Ermöglicht die fortlaufende Eintragungsnummer der parametrischen Dateien Partec.xpar und Partool.xpar zu kennen. Die angeforderte Information wird im Parameter **Behälter** gespeichert. Die zulässigen Werte für den Parameter **Identifizierer** lauten:

- "partec" fordert die fortlaufende Eintragsnumerierung von partec.xpar an
- "partool" fordert die fortlaufende Eintragsnumerierung von partool.xpar an

"@Ini"

Schreibt eine Schlüsselkombination=Wert in der tpa.ini-Datei. Der Parameter **Identifizierer** ist der in den [Tpa] - Abschnitteinzufügende Schlüsselname. Um in einem bestimmten Abschnitt zu

schreiben, muss dem Schlüsselnamen der Abschnittsname in eckige Klammern eingefügt werden ("[Abschnitt]Schlüssel").
Der Parameter **Information** kann eine eine String- oder Numerische Variable, eine Numerische- oder Stringkonstante sein.

"@ShellExecute"
erfordert dem Betriebssystem, die Datei durch das mit der Dateierweiterung assoziierte Programm zu öffnen. Es ist auch möglich, eine Exe-Datei auszuführen. Der Parameter **Identifizierer** ist der Name der zu öffnenden Datei oder der Name des zu startenden Programmes. Der Name der Datei kann mit einem vollständigen Pfad deklariert werden, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name der Datei wird auch unter denen gesucht, die durch "@FileName". definiert sind. Der Parameter **Behälter** wird den Wert 0 enthalten, wenn beim Öffnen der Datei keine Fehler aufgetreten sind, ansonsten wird er den Fehlercode enthalten.

"@StartProg"
führt das Programm aus, dass im Parameter **Identifizierer** definiert ist. Es ist nicht möglich, dem zu startenden Programm Argumente zu übergeben. Der Name des Programmes muss den vollständigen Pfad enthalten, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name des Programmes wird auch unter denen gesucht, die durch "@FileName". definiert sind. Der Parameter **Behälter** wird den Wert 0 enthalten, wenn das Programm erfolgreich gestartet worden ist, ansonsten wird er den Fehlercode enthalten. Als das Programm bereits gestartet wurde, lautet der Fehlercode 1056.

"@ProgRunning"
überprüft, wenn das mit "@StartProg" gestartete Programm immer noch ausgeführt wird. Der Name des Programmes ist im Parameter **Identifizierer** definiert. Der Name des Programmes muss den vollständigen Pfad enthalten, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name des Programmes wird auch unter denen gesucht, die durch "@FileName" definiert sind. Der Parameter **Behälter** wird den Wert 1 enthalten, wenn das Programm immer noch ausgeführt wird, ansonsten wird er den Wert 0 enthalten.

"@TermProg"
beendet das Programm, das im Parameter **Identifizierer** definiert ist und das durch "@StartProg" gestartet ist. Der Name des Programmes muss den vollständigen Pfad enthalten, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name des Programmes wird auch unter denen gesucht, die durch "@FileName" definiert sind. Der Parameter **Behälter** wird den Wert 0 enthalten, wenn das Programm erfolgreich gestartet worden ist, ansonsten wird er den Fehlercode enthalten. Als das Programm bereits gestartet wurde, lautet der Fehlercode 1056.

"@DialogFile"
öffnet das Dialogfeld von Datei Öffnen oder von Datei Speichern, um die Auswahl der Namen der Datei zu ermöglichen. Um das Fenster von Datei Öffnen zu öffnen, muss der Parameter **Identifizierer** = "Open" festgelegt werden. Um das Fenster von Datei Speichern zu öffnen, muss der Parameter **Identifizierer** = "Save" festgelegt werden. Der Name der ausgewählten Datei wird im Parameter **Behälter** gespeichert.

"@AxisCorrectors"
ersetzt die Tabelle der Linearitätskorrektoren für eine Achse durch eine neue, aus Datei geladene Tabelle, die immer noch die gleiche Anzahl von Korrektoren und die gleichen Achsen für Kreuzkorrektoren haben muss.

Der Parameter **Identifizierer** ist der Name der Datei, deren Erweiterung typischerweise .csv ist und die sich im Ordner …\Mod.*n*\Config (der Name der Datei wird 'normalisiert', wie z.B. für "@FileExist") befindet. Der Parameter **Behälter** ist als Ganzzahlvariable definiert und enthält 1, wenn neue Korrektoren gesendet wurden, ansonsten 0.

"@Language"
empfängt den übersetzbaren Text, der einer Gruppen- oder Bibliotheks- oder Modulnachricht entspricht, die mit einer MESSAGE- oder ERROR-Anweisung verknüpft ist. Die zulässigen Werte für den Parameter **Identifizierer** sind:

- DEFMSG:Nummer", wobei Nummer eine Ziffernfolge ist. Albatros schreibt in den **Behälter** den Text des Nachrichtenformulars mit der Nummer "Nummer".
- "DEFMSG:Name", wobei "Name" der Name, einschließlich Gruppe und Bibliothek, einer DEFMSG ist. Albatros schreibt in den **Behälter** den Text des angegebenen

Nachrichtenformulars. Sollte der Name der Gruppe oder der Bibliothek fehlen, dann wird der Name des Tasks verwendet, der die RECEIVE-Anweisung gesandt hat.

- "DEFMSG:*", schreibt Albatros in den **Behälter** den Text der Gruppen- oder Modulnachricht, die zuvor durch die SEND-Anweisung angegeben wurde.

Beispiel

```
; in GPL
     RECEIVE "@Param", "partec", 0, prog
     RECEIVE "@Param", "partool", 0, prog

; in GPL
; liest den Wert des Radix-Schlüssels in der Abteilung [Albatros] von
                          der tpa.ini - Datei.
   RECEIVE "@INI", "[Albatros]Radix", 0, Wert

; öffnet das Fenster von Datei Öffnen und speichert den Namen in der
                          Variablen NameDatei
RECEIVE "@DialogFile", "Open", 0, NameDatei


; vollständiges Dateilesen
Function ReadProperties
 PARAM file       AS STRING
 LOCAL version AS INTEGER
 LOCAL size      AS DOUBLE

 SEND "@FileName" "theFile" 0 file
 WAITRECEIVE "@FileInfo", "version:theFile", 0,version
 WAITRECEIVE "@FileInfo","size:theFile", 0, size
```

SEND

Syntax

SEND **[Empfänger,] Identifizierer, Flags [, Information]**

Argumente

Empfänger	Konstante des Typs String
Identifizierer	Konstante des Typs String
Flags	Konstante des Typs Integer
Information	Name einer Vorrichtung oder Variablen (numerisch oder String)

Beschreibung

Zusammen mit der Anweisung RECEIVE dient diese Anweisung dem Austausch von Informationen zwischen den Modulen der Anlage und dem Supervisor-PC. SEND wird zum Senden von Informationen verwendet; RECEIVE hingegen, um Informationen anzufordern. Die Informationen können an Albatros oder an ein externes Programm (OLE Automation Server) gesendet werden. Im letzteren Fall wird die Information auf jeden Fall von Albatros empfangen und an das externe Programm weitergeleitet.

Der Parameter **Empfänger** ist ein String zur Angabe, an wen die Information adressiert ist. Es gibt drei KlAchsen von Empfängern:

- Empfänger, die mit dem Zeichen "@" beginnen (siehe Liste weiter vorne). Der Empfänger ist genau genommen Albatros bzw. eine seiner Funktionen.
- Empfänger, die nicht mit dem Zeichen "@" beginnen. Sie werden wie OLE Server angesehen. Bei Erhalt der ersten, an sie adressierten Information, versucht Albatros, sie auszuführen und die vom Modul erhaltene Information an sie zu übergeben.
- nicht angegebener Empfänger (es handelt sich um einen optionellen Parameter). In diesem Fall wird die Information von Albatros in einer Tabelle aufbewahrt und steht zur Anforderung (seitens eines anderen Moduls oder eines externen Programms) zur Verfügung.

Der Parameter **Identifizierer** ist der Name der Information und kann nicht ausgelassen werden. Er nimmt je nach Empfänger verschiedene Bedeutungen an:

- ist der Empfänger Albatros, handelt es sich um einen Befehl, der mit der aufgerufenen Funktion zusammenhängt
- ist der Empfänger ein OLE Server, handelt es sich um eine Eigenschaft des angefragten OLE-Objekts.

- ist der Empfänger nicht angegeben, handelt es sich um das Etikett, das die Information innerhalb der von Albatros aufbewahrten Tabelle identifiziert.

Der Parameter **Flags** dient der Angabe, wie die angeforderte Information von Albatros behandelt werden soll. Die zulässigen Werte und deren Wirkung sind folgende:

Wert	***Befehl***	***Beschreibung***
$0001H	Broadcast	Normales Senden der Information
$0008H	CancelAfter	Nach dem Lesen wird die Information gelöscht
$0020H	ReadOnly	Die Information kann nur vom Sender gelöscht werden
$1000H	UpdateFlags	Ändert den Zustand der Information (gelesen / zu lesen), ohne die Daten zu ändern
$8000H	Delete	Löscht die Information

Der Parameter **Information** ist die Information, die man sendet. Läßt man diese Angabe aus, sendet man eine leere Information, die die Bedeutung der Bekanntgabe eines Ereignisses annimmt (kann zur Synchronisierung der Ausführung des GPL-Codes an verschiedenen Modulen verwendet werden).

Liste der von Albatros verwalteten **Empfänger** und der zugehörigen Befehle:

"@List"

Ermöglicht die Kontrolle der Befehle Simulation und Setpoint.
Folgende Befehle sind zugelassen (Parameter **Identifizierer**):

- Sim: Teilt die Zustandsänderung des Switch-Flags Simulation mit. Je nach dem Flagszustand wird die Schaltfläche gedrückt oder losgelassen, womit er in der Werkzeugsleiste identifiziert (1=gewählt, 0=deaktiviert) wird.
- Setp: Teilt die Zustandsänderung des Switch-Flags CmdSetp mit. Je nach dem Flagszustand wird die Schaltfläche gedrückt oder losgelassen, womit er in der Werkzeugsleiste identifiziert (1=gewählt, 0=deaktiviert) wird.
- Esc: Teilt die Zustandsänderung des Switch-Flags Escluso mit. Je nach dem Flagszustand wird die Schaltfläche gedrückt oder losgelassen (derselbe von CmdSetp Flag), womit er in der Werkzeugsleiste identifiziert (1=gewählt, 0=deaktiviert) wird.
- End: Beendet die Ausführung der Liste. Dieser Befehl drückt die Schaltflächen von Start und Stop und deaktiviert die Optionen der Menüs Start und Stop.
- Hold: Drückt die Schaltfäche Stop nieder und aktiviert die Option aus der Menü Stop.

"@Syn"

Kommunikation zwischen GPL und der Anzeige der synoptischen Darstellungen. Dient dem Öffnen und Schließen von synoptischen Darstellungen mittels GPL und das Senden von Informationen in Felder von synoptischen Darstellungen. Die zulässigen Befehle lauten (Parameter **Identifizierer**):

- "Open:*Dateiname*" öffnet die synoptische Datei *Dateiname.xsyn*
- "Close:*Dateiname*" schließt die synoptische Datei *Dateiname.xsyn*
- *"*Open*"* öffnet eine synoptische Darstellung. Der Dateiname wird von der Variablen **Information** abgelesen
- *"*Close*"* schließt eine synoptische Darstellung. Der Dateiname wird von der Variablen **Information** abgelesen
- "*Feldname*" Feld, in dem die angeforderte Information angezeigt wird

Mit dem Fenster der Achsenbewegung kann gemäß den definierten Spezifikationen auch für den unten wiedergegebenen Parameter **Empfänger** "@Devices" interagiert werden.

"@File"

Schreibt in eine Datei. Dient der Schaffung von individuellen Log-Dateien zur Speicherung der von einer Maschine ausgeführten Vorgänge. Es handelt sich um Text-Dateien (ASCII). Der Parameter **Identifizierer** ist der Name der Datei, in die sie geschrieben wird.
Falls im Identifikationsparameter der vollständige Pfad, in dem die Datei zu archivieren ist, nicht angegeben ist, dann wird als Pfad betrachtet, der der in tpa.ini unter *dirreport* definiert ist.
Wenn der Identifizierer mit einem Zeichen von % anfängt und beendet, der String wird im seinem Inneren in tpa.ini in der Sektion [tpa] gesucht und wie Name des Files verwendet. Im Inneren des Namens können einige Zeichen eingeführt werden, die während der Ausführung der Anweisung ausgetauscht werden:

- %n Nummer des Moduls, das die Anweisung SEND ausführt
- %h laufende Zeit (Format 00-23)
- %d laufender Tag (Format 01-31)
- %m laufender Monat (Format 01-12)
- %y laufendes Jahr (vierstellig-Format)

Das Beispiel sehen.
Die Schreibvorgänge erfolgen im append-Modus (d.h. die Daten werden am Ende zur Datei hinzugefügt). Zu einer Datei können numerische Daten (die automatisch in ASCII umgewandelt werden) oder Strings gesendet werden. Die Strings können im Format Datum und Uhrzeit geschrieben werden, indem man die Formatzeichen %d für das Datum und %t für die Uhrzeit verwendet.
Zur Einstellung der Uhrzeit wird das Format "HH:mm:ss" verwendet (d.h. Stunden, Minuten und Sekunden, von ":" getrennt); zur Einstellung des Datums wird ein Format gemäß dem regionalen Einstellungen verwendet. Es ist möglich, ein anderes Format zu verwenden, indem die Option "LogNoLocale=1" in tpa.ini, im Abschnitt [Albatros] festgelegt wird. Die standardmäßige Einstellung ist è LogNoLocale=0, d.h. Verwendung des derzeitigen Formats. Es ist auch möglich, das zu verwendende Format von Datum und Uhrzeit, unabhängig von dem bereits eingestellten Format Windows, festzulegen, wie folgt:
- In tpa.ini, im Abschnitt [Albatros] die Optionen "LogDateFormat=" und "LogTimeFormat=" festlegen;
- Eine Zeichenkette gemäß der nachfolgende Tabelle zuweisen.
Falls diese Optionen nicht vorhanden oder leer sind, dann werden die von Windows festgelegten Formaten verwendet.

Datum und Zeitformat

h	Stunde im 12-Stunden-Format ohne vorangestellte Nullen
hh	Stunde im 12-Stunden-Format mit vorangestellten Nullen
H	Stunde im 24-Stunden-Format ohne vorangestellte Nullen
hh	Stunde im 24-Stunden-Format mit vorangestellten Nullen
m	Minuten ohne vorangestellte Nullen
mm	Minuten mit vorangestellten Nullen
s	Sekunden ohne vorangestellte Nullen
ss	Sekunden mit vorangestellten Nullen
t	ein einzelnes Zeichen zum Anzeigen des AM/PM-Kennzeichners, z.B. A oder P
tt	mehrere Zeichen zum Anzeigen des AM/PM-Kennzeichners, z.B. AM oder PM

Anmerkung: Die Formate "t" und "tt" verwenden den AM/PM-Kennzeichner, der in der Systemsteuerung des derzeitigen Bedieners aufgewiesen ist.
Nicht unbedingt müssen sie "AM" und "PM" sein.
Beispiel: Wenn es 11:29 Uhr abends ist und der String "hh':'mm':'ss tt" ist, erscheint "11:29:40 PM".

Tagesformat

d	Tag des Monats ohne vorangestellte Nullen, mit Ziffern dargestellt
dd	Tag des Monats mit vorangestellten Nullen, mit Ziffern dargestellt
ddd	Wochentag, mit Zeichen dargestellt und auf drei Buchstaben verkürzt
dddd	Wochentag, mit Zeichen mit vollständigem Namen dargestellt
M	Monat ohne vorangestellte Nullen, mit Ziffern dargestellt
MM	Monat mit vorangestellten Nullen, mit Ziffern dargestellt
MMM	Monat, mit Zeichen dargestellt und auf drei Buchstaben verkürzt
MMMM	Monat, mit Zeichen mit vollständigem Namen dargestellt
y	Jahr mit zwei Ziffern ohne vorangestellte Nullen für Jahre unter 10
yy	Jahr mit zwei Ziffern mit vorangestellten Nullen für Jahre unter 10
yyyy	Das Jahr ist mit vier oder fünf Ziffern (abhängig vom verwendeten Kalender) dargestellt
yyyyy	Das Jahr ist mit vier oder fünf Ziffern (abhängig vom verwendeten Kalender) dargestellt

Beispiel: Wenn es Mittwoch 31. August 1994 und der String "ddd',' MMM dd yy" ist, dann erscheint "Mi, August 31. 94".
Wird die Information ausgelassen, wird zur Datei ein Wagenrücklauf hinzugefügt.

"@FileName"

Speichert eine Bindung zwischen einer String-Konstante und einer Datei, deren Namen mit Stringvariablen zusammengestellt werden kann. Nachdem Albatros den Bindungsbericht erhalten hat, wird jeder nachkommende Dateiname mit dem durch diese Anweisung empfangenen Namen ersetzt. Der Parameter **Identifizierer** ist der Name der Datei, in die man geschrieben wird. Der Name der Datei ist eine Zeichenfolgevariable. Ist im Parameter Identifizierer der vollständige Pfad nicht angegeben, in dem die Datei zu archivieren ist, dann wird als Pfad betrachtet der, der in dem Abschitt [tpa] unter dirreport definiert ist. Der Wert des Parameters Identifizierer wird in tpa.ini in der Abschnitt [GPLFileName] unter Log gespeichert, um ihn in den nachkommenden Ausführungen von Albatros gebrauchen zu können. Um die

Bindung zu entfernen, soll man als Parameter Identifizierer eine leere Zeichenfolge einstellen. Die in dieser Weise definierte Bindung gilt für alle Module.

"@FileDelete"
Löschen einer Datei. Der Parameter **Identifizierer** ist der Name der Datei, die gelöscht wird (vollständiger Pfad). Falls im Identifikationsparameter der vollständige Pfad, in dem die Datei zu archivieren ist, nicht angegeben ist, dann wird als Pfad betrachtet, der der in tpa.ini unter *dirreport* definiert ist. Der Dateiname kann gemäß den in Übereinstimmung mit dem Parameter **Empfänger** @File beschriebenen Regeln definiert werden.

"@FileRead"
Positioniert den Cursor auf die Datei am Dateianfang. Der Parameter **Identifiziere** ist der Name der Datei (vollständiger Pfad). Falls im Identifikationsparameter der vollständige Pfad, in dem die Datei zu archivieren ist, nicht angegeben ist, dann wird als Pfad betrachtet, der der in tpa.ini unter *dirreport* definiert ist. Der Dateiname kann gemäß den in Übereinstimmung mit dem Parameter **Empfänger** @FileRead beschriebenen Regeln definiert werden.

"@Axis"
Interagiert mit dem Fenster der Achsenbewegung gemäß den Spezifikationen definiert auch für den unten wiedergegebenen Empfängerparameter "@Devices". Ist ein Fenster für die Verwaltung der angegebenen Achse bereits offen, wirkt der Befehl auf dieses Fenster, sowohl es in einer synoptischen Darstellung, als auch in der Diagnostik geöffnet ist. Ist das Fenster geschlossen, versucht der Befehl dieses Fenster in einer der bereits offenen synoptischen Darstellungen zu öffnen, die diese Achse enthält.

"@Devices"
Anforderung, das Diagnostik-Fenster des Moduls, das die Information sendet, zu öffnen oder zu schließen. Ausführung der Befehle innerhalb des Diagnostik-Fensters der Achsenbewegung. Der Parameter Identifizierer kann folgende Werte annehmen:
- "Open" Öffnen der Diagnostik
- "Close" Schließen der Diagnostik

Der Parameter **Identifizierer** kann die folgenden Werte annehmen, wenn mit dem Fenster der Achsenbewegung interagiert werden soll:
"MoveAX#nome_asse#Open"Öffnung des Fensters der Achsenbewegung.
"MoveAX#nome_asse#Close Schließung des Fensters der Achsenbewegung.
"
"MoveAX#nome_asse#Plus" Druck der Taste der Achsenbewegung in positive Richtung.
"MoveAX#nome_asse#Minus Druck der Taste der Achsenbewegung in negative Richtung.
"
"MoveAX#nome_asse#Stop" Druck der Taste für Bewegungsstopp.
"MoveAX#nome_asse#Jog" stellt die Bewegungsweise für vom Benutzer verwaltete Versetzungen der Laufzeit ein.
"MoveAX#nome_asse#Step" stellt die Bewegungsweise für die Versetzungen der vorbestimmten Steigung ein.
"MoveAX#nome_asse#Absolute" stellt die Bewegungsweise mit Versetzung mit festgelegter Höhe der Achse ein.

wobei Name_Achse den Namen der im Fenster angezeigten Achse bezeichnet. Wenn zum Beispiel das Fenster der Bewegung der X-Achse geöffnet werden soll, ist der Parameter **Identifizierer** "@MoveAX#X#Open". Der Name der Achse kann in einer folgenden Formen sein:
1. Name_Gruppe.Name_Untergruppe.Name_Achse oder Name_Gruppe.Name_Achse: Es wird der vollständige Verlauf der Achse geliefert.
2. Name_Achse: Zur Identifizierung der richtigen Achse werden die folgenden Prüfungen in Folge gemacht:
   - Wenn die Task, aus der der Befehl stammt, eine Funktion der Untergruppe, wird die Achse in dieser Untergruppe gesucht.
   - Wenn die Task, aus der der Befehl stammt, eine Funktion der wichtigsten Untergruppe ist, wird die Achse in der gesamten Gruppe gesucht. Wenn mehr als eine Achse mit diesem Namen vorhanden ist, schlägt die Suche fehl.
   - Wenn die vorherigen Prüfungen fehlgeschlagen sind, wird die Achse in allen Gruppen des Moduls gesucht. Wenn mehr als eine Achse mit dem Namen Name_Achse vorhanden ist, hat die Suche kein positives Ergebnis.

Es kann verhindert werden, dass der Benutzer alle Tasten der Achsenbewegung aller Diagnostik-Fenster der Achsenbewegung des Moduls betätigt, indem der Parameter **Identifizierer** wie folgt eingestellt wird:
- "MoveAX##UIENABLE" wenn der Parameter **Information** auf 0 eingestellt wird, wird die Achsenbewegung von Albatros ausgeschaltet, wenn er auf 1 eingestellt wird, wird die Achsenbewegung von Albatros eingeschaltet.

Das Ausschalten der Achsenbewegung von Albatros empfiehlt sich, wenn die Achsen von der Schalttafel der Maschine bewegt werden.

"@Vars"
Fordert die Speicherung des Inhalts einer globalen GPL-Variable im Archiv der technologischen Parameter oder Werkzeugen an. Der Parameter **Identifizierer** ist der Name der globalen Variable (Maschine, Gruppe oder Bibliothek) von der die Aktualisierung angefordert wird.

"@Application"
Interaktion mit Albatros. Diese Anweisung ermöglicht das Schließen von Albatros oder die Anzeige von "Dialogfeldern" auf dem Bildschirm, um den Benutzer zu informieren oder um die Zustimmung zu nachfolgenden Aktivitäten zu erbitten. Die zulässigen Werte für den Parameter **Identifizierer** lauten:

"Quit"	schließt Albatros
"Lock"	behindert den Schluss von Albatros aus ***Datei->Beenden*** oder aus der Tastenkombination [ALT+F4] oder aus einer Schaltfläche zum Schließen.
"UnLock"	stellt wieder her die Möglichkeit, Albatros zu schließen
"MsgBox:*Flags*"	öffnet ein Nachrichtenfenster

Das Verhalten der Nachrichtenfenster wird auf der "*Flags*"-Seite durch den **Identifizier**-String gesteuert. Dieser kann aus einer Folge der folgenden Zeichen (sie können entweder groß oder klein geschrieben sein) bestehen:

"O"	Schaltfläche "OK"
"C"	Schaltfläche "Abbrechen"
"Y"	Schaltfläche "Ja"
"N"	Schaltfläche "Nein"
"R"	Schaltfläche "Retry"
"S"	Symbol des Stop-Plakats
"?"	Informationssymbol, bestehend aus einem Kleinbuchstaben "i" innerhalb eines Kreises
"!"	Symbol mit Ausrufezeichen
"*"	Informationssymbol
"1"	die Default-Schaltfläche ist die erste
"2"	die Default-Schaltfläche ist die zweite
"3"	die Default-Schaltfläche ist die dritte
"4"	die Default-Schaltfläche ist die vierte

Falls nicht angegeben, ist die Default-Schaltfläche die erste.
"MsgBox:?YN2" definiert z.B. ein Nachrichtenfenster mit dem Informationssymbol , zwei Schaltflächen "Ja" und "Nein", wobei die zweite die Default-Schaltfläche ist.
Der Parameter **Information** kann ein String sein, der den anzuzeigenden Text enthält oder eine ganze Zahl, die als Code einer von TpaLangs.exe verwalteten Modulnachricht interpretiert wird, oder ein Etikett einer durch die Anweisung DEFMSG definierten Gruppennachricht.
Was für den Text anbelangt und wenn sich im Text ein Zeilenumbruchzeichen "\u000A" befindet, dann wird der Text in zwei Teile geteilt und der erste Teil als Text des Nachrichtenfeldes angezeigt, während der zweite Teil als Erklärung oder Detail des Textes angezeigt wird.
Die Sprache, in der die Schaltflächen angezeigt werden, ist die Windows-Sprache.

"@Help"
Öffnet eine Hilfe-Datei. Dient der Anzeige einer Hilfe-Datei unter Angabe des anzuzeigenden Inhalts. Die zulässigen Werte für den Parameter **Identifizierer** lauten:

- "Open:*Dateiname*" öffnet eine Hilfe-Datei
- "Close:*Dateiname*" schließt eine Hilfe-Datei

Der Teil "*Dateiname*" des Strings gibt den Namen der zu öffnenden Hilfe-Datei an.
Der Parameter **Information** kann ein String oder eine Ziffer sein und nimmt respektive die Bedeutung eines Kontext-Schlüssels oder einer Kontext-Ziffer an (dient der Bestimmung der Seite oder des Themas der Hilfe, die man anzeigen möchte).

"@Report"
Fügt dem Albatros-Report-File (MONTH(Monat Nr.).TER) Anweisungen an. Der Parameter **Identifizierer** ist:

- "Add"
- Der Parameter **Information** kann:
- eine Variable des Typs String oder eine Konstante des Typs String sein: in diesem Fall wird der Text gespeichert, der im String beinhalten ist.
- eine Variable des Typs Integer oder ein numerischer Wert des Typs Integer sein: in diesem Fall wird der Text gespeichert, der in der Anweisung DEFMSG bestimmt ist.

"@Ini"
Schreibt eine Schlüsselkombination=Wert in der tpa.ini-Datei. Der Parameter **Identifizierer** ist der in den [Tpa] - Abschnitteinzufügende Schlüsselname. Um in einem bestimmten Abschnitt zu schreiben, muss dem Schlüsselnamen der Abschnittsname in eckige Klammern eingefügt werden ("[Abschnitt]Schlüssel").
Der Parameter **Information** kann eine eine String- oder Numerische Variable, eine Numerische- oder Stringkonstante sein.

"@ShellExecute"
Erfordert dem Betriebssystem, die Datei durch das mit der Dateierweiterung assoziierte Programm zu öffnen. Es ist auch möglich, eine Exe-Datei zu starten. Der Parameter **Identifizierer** ist der Name der zu öffnenden Datei oder der Name des zu startenden Programmes. Der Name der Datei kann mit einem vollständigen Pfad deklariert werden, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name der Datei wird auch unter denen gesucht, die durch "@FileName". definiert sind.

"@StartProg"
Führt das Programm aus, dass im Parameter **Identifizierer** definiert ist**.**· Es ist nicht möglich, dem zu startenden Programm Argumente zu übergeben. Der Name des Programmes muss den vollständigen Pfad beinhalten, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name des Programmes wird auch unter denen gesucht, die durch "@FileName". definiert sind.

"@TermProg"
Bringt das Programm, das im Parameter **Identifizierer** definiert ist, zu Ende und wird er durch "@StartProg" gestartet · Der Name des Programmes muss den vollständigen Pfad beinhalten, ansonsten wird er im laufenden Ordner von Albatros gesucht. Der Name des Programmes wird auch unter denen gesucht, die durch "@FileName". definiert sind.

"@DialogFile"
Erlaubt, einige Parameter des Dialogfensters der Datei zu öffnen oder der Datei *Speichern* festzulegen.
Die zulässigen Werte für den Parameter **Identifizierer** lauten wie folgt:

"Extension"	setzt der Benutzer keine Erweiterung ein, dann wird die im Parameter **Information** (Variable oder Stringkonstante) definierte Erweiterung verwendet.
"Filter"	legt den Filter auf die zu verwendenden Dateitypen fest. Der Parameter **Information** kann eine Zeichenfolgevariable oder eine Zeichenfolgekonstante sein und in diesem Fall wird der in der Zeichenfolge enthaltene Text oder eine Integer - Variable oder ein numerischer Integer-Wert als Filter verwendet und in diesem Fall wird der in der Anweisung DEFMSG enthaltene Text als Filter verwendet.
"Flags"	legt die Initialisierungsflags fest. Für die Liste der im Feld **Information** (Variable oder Integerkonstante) anzuwendenden Werte, sehen Sie die offizielle Dokumentation von Microsoft hinsichtlich des Flags-Elements der OPENFILENAME - Struktur.
"InitalDir"	legt den anfänglichen Folder fest, der im Feld **Information** (Variable oder Zeichenfolgekonstante) definiert ist.
"Title"	legt den Titel des Fensters fest. Der Parameter **Information** kann eine Zeichenfolgevariable oder eine Zeichenfolgekonstante sein und in diesem Fall wird der in der Zeichenfolge enthaltene Text, eine Integervariable oder ein numerischer Integerwert als Titel verwendet und in diesem Fall wird der in der Anweisung DEFMSG verwendete Text als Titel definiert.

"@Language"
Stellt die Gruppen- oder Modul- oder Bibliotheksmeldungsnummer, die beim nächsten RECEIVE mit der gleichen Identifizierer verwendet wird. Der zulässige Wert für den Parameter **Identifizierer** ist "DEFMSG:*". Der Parameter Information Variable-Integer oder eine Konstante-Integer sein und definiert in diesem Fall die Nummer der anzuzeigenden Gruppennachricht. Sie kann vom Typ String oder String-Konstante sein und definiert in diesem Fall den Namen der DEFMSG.

Beispiel

```
;Beispiel der Anweisung "Send File" mit in Ausführung aufgebauten Name.
:Nehmen wir an, daß das Datum in dem die Ausführung ausgeführt wird
;der 31-01-2000 ist
```

```
; in GPL
     SEND "@File", "%Log%", 0, "Anfangen der Ausführung"
     SEND "@File", "%Log%", 0      ; fügt ein "Neue Zeile Beginnen" hinzu
; in der Sektion [TPA] wird in der Datei tpa.ini
     Log=c:\Albatros\report\%y\Rep%m%d.txt  ; hinzugefügt

; Der Name des resultierenden Files ist:
c:\Albatros \report\2000\Rep0131.txt

; Anweisungsbeispiel send  Vars
; Eine Variable Var_SendVars as double wird in der Datei der globalen
                        Variablen definiert.
; Im technologischen Parameter wird Var_SendVars in das Feld Matrix Name
                        eingegeben
; in GPL
     SETVAL  100.0,Var_SendVars
; sendet den Wert 100.0 dem Parameter der technologischen Parameter
; assoziiert mit der Variable Var_SendVars
     SEND "@Vars", "Var_SendVars", 0

; Beispiel einer send INI-Anweisung
; man schreibt in tpa.ini den Radix-Schlüssel in dem Abschnitt
                        [Albatros] um eine numerische Grundlage einer
                        Dezimalanzeige festzulegen
 SEND "@INI", "[Albatros]Radix", 0;1

; Beispiel: Bindung einer GPL-String-Konstante mit
; dem Name einer Datei einstellen.

;Deklaration einer String-Variable
Dateiname as String
; Zusammenstellung des Namen der Datei
Setstring "C:\albatros\report\LogFile.txt",Dateiname
; Bindung
 SEND "@FileName", "LOG",0,Dateiname
;von jetzt an ist jeder Schriftvorgang
;in der von der Variablen Dateiname definierten Datei ausgeführt
 SEND "@Datei", "LOG",0,"Schrift in der Datei LOG"
```

SENDIPC

Syntax

SENDIPC	**IPCName, Warten [, VarName1 [, VarNameN, ...]]**
SENDIPC	**IPCName, Warten, Matrix[Zeile]**
SENDIPC	**IPCName, Warten, Vektor**
SENDIPC	**IPCName, Warten, Matrix**

Argumente

IPCName	String-Konstante.Name der IPC
Warten	vordefinierte Konstante. Warte-Modus des Lesens des Befehls Die zulässigen Werte lauten: **WAIT** wartet das Lesen des Befehls ab **NOWAIT** wartet das Lesen des Befehls nicht ab
VarName1[...VarNameN]	Konstante oder Variable. Namen der Variablen 1÷N
Matrix[Zeile]	Konstante oder Integer-Variable. Nummer der Zeile der Matrix
Vektor	Name des Vektors
Matrix	Name der Matrix

Beschreibung

Diese Anweisung sendet einen IPC-Befehl an den gemeinsam benutzten Speicher **"IPCName"**.
Bei der ersten Ausführung einer SENDIPC-Anweisung wird der gemeinsam benutzte Speicherplatz zugewiesen, dessen Größe aufgrund des Umfangs der gesendeten Daten berechnet wird. Die maximale Größe des gemeinsam benutzten Speicherplatzes beträgt 64 Kb. 48 geteilte Speicher können höchstens bestimmt werden und sind mit 48 eindeutigen Namen identifiziert.
Dem gemeinsam benutzten Speicher ist ein Statusanzeigebit zur Synchronisierung der Ausführung der darauf zugreifenden Tasks zugeordnet. Der Task, der die Daten schreibt, aktiviert das

Statusanzeigebit nach dem Schreiben und der Task, der die Daten liest, deaktiviert das Statusanzeigebit nach dem Lesen.
Ist als **Warte**-Parameter WAIT angegeben worden, wartet der Task, der die Daten geschrieben hat, bis sie gelesen werden (Statusanzeigebit deaktiviert), bevor er die Ausführung fortsetzt.
Eine SENDIPC-Anweisung ohne Daten ist nichts weiter als eine Synchronisierung von Tasks. In diesem Fall wird der gemeinsam benutzte Speicherplatz nicht zugewiesen.

IPC Intermodul
Durch IPCs können zwei Fernmodule Daten austauschen. Diese IPCs sind Intermodul-IPCs genannt.
Um Ein IPC-Intermodul zu definieren muss man den IPCName gemäß dem folgenden Formalismus schreiben:
Zahl des Quellmoduls, "->", Zahl des Empfängermoduls, ":", danach die andere Kennzeichen des Namen vom IPC.
Zum Beispiel "0->1:Grundparameter".

Siehe auch WAITIPC und TESTIPC.

WAITIPC

Syntax

WAITIPC	**IPCName [, VarName1 [, VarNameN, ...]]**
WAITIPC	**IPCName, Matrix[Zeile]**
WAITIPC	**IPCName, Vektor**
WAITIPC	**IPCName, Matrix**

Argumente

IPCName	String-Konstante.Name der IPC
VarName1[...VarNameN]	Konstante oder Variable. Namen der Variablen 1÷N
Matrix[Zeile]	Konstante oder Integer-Variable. Nummer der Zeile der Matrix
Vektor	Name des Vektors
Matrix	Name der Matrix

Beschreibung

Diese Anweisung erhält einen IPC-Befehl vom gemeinsam benutzten Speicher **"IPCName"**.
Bei der ersten Ausführung einer WAITIPC-Anweisung wird der gemeinsam benutzte Speicherplatz zugewiesen, dessen Größe aufgrund des Umfangs der gesendeten Daten berechnet wird. Die maximale Größe des gemeinsam benutzten Speicherplatzes beträgt 64 Kb. 48 geteilte Speicher können höchstens bestimmt werden und sind mit 48 eindeutigen Namen identifiziert.
Dem gemeinsam benutzten Speicher ist ein Statusanzeigebit zur Synchronisierung der Ausführung der darauf zugreifenden Tasks zugeordnet. Der Task, der die Daten liest, wartet ab, daß das Statusanzeigebit von dem Task, der die Daten schreibt, aktiviert wird, liest die Daten und deaktiviert das Statusanzeigebit.
Eine WAITIPC-Anweisung ohne Daten ist nichts weiter als eine Synchronisierung von Tasks. In diesem Fall wird der gemeinsam benutzte Speicherplatz nicht zugewiesen.
Siehe auch SENDIPC und TESTIPC.

WAITRECEIVE

Syntax

WAITRECEIVE	**[Quelle,] Identifizierer, Flags [, Behälter]**

Argumente

Quelle	Konstante des Typs String
Identifizierer	Konstante des Typs String
Flags	Konstante des Typs Integer
Behälter	Name einer Vorrichtung oder Variablen (numerisch oder String)

Beschreibung

Diese Anweisung wartet ab, daß die angeforderte (vom **Identifizierer** angegebene) Information angekommen ist, bevor die Ausführung des GPL-Programms fortgesetzt wird. In bezug auf die Verwendung wird auf die Erklärung der Anweisung RECEIVE verwiesen.

10.3.10 Mathematik

ABS

Syntax

ABS	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den absoluten Wert des **Operands** und setzt ihn ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SET -10,op ; weist der Variablen op -10 zu
VAL
ABS op,var ; der in die Variable var gesetzte Wert wird 10 sein
```

ADD

Syntax

ADD	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung summiert **Operand1** und **Operand2** und setzt das Ergebnis ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SET 5,op1 ; weist der Variablen op1 5 zu
VAL
ADD op1,3
    ,var

;Der in die Variable var gesetzte Wert wird 8 sein
```

AND

Syntax

AND	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt ein binäres AND (*zwischen zwei Bits, ergibt 1 nur wenn der Wert von beiden 1 ist*) zwischen **Operand1** und **Operand2** aus und setzt das Ergebnis ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
; Der in die Variable var gesetzte Wert wird 1 sein
```

```
; (Binäre Schreibweise: 5 = 0101, 3 = 0011, 1 = 0001)

AND 5,3,var
```

ARCCOS

Syntax

ARCCOS	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den Arkuskosinus des **Operands** und setzt den Wert in Grad ins **Ergebnis**. Der Wert des Ergebnisses ist im Intervall 0°÷180° inbegriffen. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

ARCSIN

Syntax

ARCSIN	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den Arkussinus des **Operands** und setzt den Wert in Grad ins **Ergebnis**. Der Wert des Ergebnisses ist im Intervall -90°÷+90° inbegriffen. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

ARCTAN

Syntax

ARCTAN	**Operand1 [, Operand2], Ergebnis**

Argumente

Operand1...[Operand2]	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Ist **Operand2** ausgelassen, wird der Arkustangens des **Operands1** berechnet und der Wert in Grad ins **Ergebnis** gesetzt.
Ist **Operand2** vorhanden, wird der Winkel berücksichtigt, dessen Sinus vom **Operand1** und dessen Kosinus vom **Operand2** gegeben ist. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

COS

Syntax

COS	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den Kosinus des **Operands** und setzt den Wert ins **Ergebnis**.
Das Argument **Operand** ist in Grad und etwaige Bruchwerte sind in Hundertsteln ausgedrückt (Bsp.: 30° 15" = 30,25.). Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL    60,op ; weist der Variablen op 60 zu
COS       op,va
          r

;Der in die Variable var gesetzte Wert wird 0,5 sein
```

DIV

Syntax

DIV	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung dividiert **Operand1** durch **Operand2** und setzt das Ergebnis ins **Ergebnis**. Die Anweisung kann einen Systemfehler hervorrufen, wenn **Operand2** gleich 0 ist. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETV  10,op1  ; weist der Variablen op1 10 zu

SETV  5,op2   ; weist der Variablen op2 5 zu

DIV   op1,op2
      ,var

;Der in die Variable var gesetzte Wert wird 2 sein
```

EXP

Syntax

EXP	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet die Exponentialziffer des **Operands** und setzt den Wert ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVA  2.30258509 ;weist der Variablen op
       L3,op
                  ;2,302585093 zu
EXP    op,var

;Der in die Variable var gesetzte Wert wird 10 sein
```

EXPR

Syntax

EXPR **Variable = Ausdruck**

Argumente

Variable Name der Vorrichtung oder Variable
Ausdruck Menge von Operatoren

Beschreibung

Die Faktoren können Konstante, Vorrichtungen-Name oder Variablen sein. Die Syntax der Anweisung bestimmt, dass zwischen jedem Operator und jedem Operand ein Zwischenraum bestehen muss. Wenn die Operanden des Ausdruckes nicht alle vom gleichen Typ sind, dann wird eine automatische Konvertierung durchgeführt so dass das Vorgangsergebnis gleich ist dem, das der beiden größer ist, nach folgender Regel:

- char <integer
- float < double
- char oder integer < float oder double.

Nachdem der Ausdruck gelöst worden ist, dann wird das Ergebnis in den Typ **Variable** konvertiert. Die erlaubten Operatoren sind:

()	Klammer
-	Zeichen-Wechsel-Operator
ABS	Absoluter Wert des Operands
ROUND	auf/abrunden auf der Einheit
TRUNC	Abkürzung bis zum gesamten Wert
LOG	natürliches Logaritmus
LOGDEC	Logarithmus von 10
EXP	exponential
SRQ	Quadratwurzel-Vorgang
SIN	Sinus-Vorgang. Der Operand ist in Graden ausgedrückt mit eventuellem Hunderstelbruch-Teil (Beispiel: 30° 15" = 30,25.)
COS	Kosinus-Vorgang. Der Operand ist in Graden mit eventuellem Hunderstelbruch-Teil ausgedrückt (Beispiel: 30° 15" = 30,25.)
TAN	Tangente-Vorgang. Das Argument ist/wird in Graden ausgedrückt
ARCSIN	Bogensinus-Vorgang. Das Resultat ist in Graden ausgedrückt und sein Wert wird im Intervall.-90°÷+90° eingeschlossen
ARCCOS	Bogensinus-Vorgang. Das Resultat ist in Graden ausgedrückt und sein Wert wird im Intervall.0°÷180° eingeschlossen
ARCTAN	führt eine Bogen-Tangente-Vorgang aus. Siehe ARCTAN
^	Potenz-Operator

*	Produkt-Operatoren
/	Teilung
%	Teilungsrest (Modul)
+	Summe-Operatoren
-	Subtraktion

Anhand dieser Anweisung lässt sich die Darstellung des GPL-Codes in den Fällen vereinfachen, in denen mathematische Berechnungen durchgeführt werden müssen, indem die einzelnen GPL-Anweisungen, die den in der Tabelle aufgeführten Operatoren entsprechen, ersetzt werden. Diese Anweisungen bleiben aus Kompatibilitätsgründen verfügbar.

Beispiel

```
; Berechnung des Abstands zwischen zwei Punkten
```

```
EXPR dist = SQR ( ( Xb - Xa ) ^ 2 + ( Yb - Ya ) ^ 2 )
```

LOG

Syntax

LOG	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den natürlichen Logarithmus des **Operands** und setzt den Wert ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETV 10,op ; weist der Variablen op 10 zu
  AL
LOG  op,va
     r

;Der in die Variable var gesetzte Wert wird 2.302585093 sein
```

LOGDEC

Syntax

LOGDEC	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den Logarithmus zur Basis 10 des **Operands** und setzt den Wert ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL    10,op       ; weist der Variablen op 10 zu
LOGDEC    op,var

;Der in die Variable var gesetzte Wert wird 1 sein
```

MOD

Syntax

MOD	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Integer-Variable oder Name einer Vorrichtung
Operand2	Konstante oder Integer-Variable oder Name einer Vorrichtung
Ergebnis	Integer-Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt einen Modulo zwischen **Operand1** und **Operand2** aus und setzt das Ergebnis ins **Ergebnis**. Der Modulo ist der Rest aus der Division des Operands 1 durch den Operanden 2. Die Anweisung kann einen Systemfehler hervorrufen, wenn **Operand2** gleich 0 ist. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL   20,op1       ; weist der Variablen op1 20 zu
SETVAL   3,op2        ; weist der Variablen op2 3 zu
```

```
MOD     op1,op2,var

;Der in die Variable var gesetzte Wert wird 2 sein
```

MUL

Syntax

MUL	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt einen Modulo zwischen **Operand1** und **Operand2** aus und setzt das Ergebnis ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL  5,op1       ; weist der Variablen op1 5 zu

SETVAL  2,op2       ; weist der Variablen op2 2 zu
MUL     op1,op2,var

;Der in die Variable var gesetzte Wert wird 10 sein
```

NOT

Syntax

NOT	**Operand**

Argumente

Operand	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt ein binäres NOT (*Umkehrung der einzelnen Bits*) des vom **Operanden** ausgedrückten Werts aus. Das Ergebnis wird im **Operanden** gespeichert.

Beispiel

```
SETVAL      5,var   ; weist "var" den Wert 5 zu
NOT         var

; Das Ergebnis wird var = -6
; binäre Schreibweise: 5 = 0000 0101,
; binäre Schreibweise:10 = 0000 1010
; hexadezimale Schreibweise 5 = 0000 0000 0000  0005
; hexadecimale Schreibweise 10 = 0000 0000 0000 000A
; die Ausführung von einem NOT auf Wert 5 ergibt 0xFFFF FFFF FFFF FFFA =
; -6
```

OR

Syntax

OR	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt ein binäres OR (zwischen zwei Bits, ergibt 1 wenn der Wert von mindestens einem von beiden 1 ist) zwischen **Operand1** und **Operand2** aus und setzt das Ergebnis ins

Ergebnis. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
;Der Wert der Variablen var wird 7 sein
;(binäre Schreibweise: 5 = 0101, 3 = 0011, 7 = 0111 )

OR  5,3,var
```

RANDOM

Syntax

RANDOM	**min, max, Ergebnis**

Argumente

min	Konstante oder Variable
max	Konstante oder Variable
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung gibt im **Ergebnis** eine pseudozufällige Ziffer zwischen **min** und **max** (Grenzwerte inbegriffen) zurück.
Durch wiederholtes Ausführen der Anweisung erhält man eine Sequenz von pseudozufälligen Ziffern.
Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL    2,op1           ; weist der Variablen op1 2 zu
SETVAL    100,op2         ; weist der Variablen op2 100 zu
RANDOM    op1,op2,var

;Der in die Variable var gesetzte Wert wird
;eine zufällige Ziffer zwischen 2 und 100 sein
```

RESETBIT

Syntax

RESETBIT	**Maske, BitNr**

Argumente

Maske	Integer-Konstante oder -Variable oder ZählerName oder Port-Name. Stellt den zu ändernden Wert dar (max. 32 Bit)
BitNr	Integer-Konstante oder -Variable oder ZählerName. Nummer des zu ändernden Bits

Beschreibung

Diese Anweisung setzt ein einzelnes, von **BitNr** angegebenes Bit der übergebenen Bit-**Maske** auf 0. Das Argument **Maske** muß einem ganzen Wert mit höchstens 32 Bit entsprechen können. Die Bit-Nummer, **BitNr**, geht von 1 bis 32.

Beispiel

Zustand des Ports vor Ausführung des Codes

Zustand des Ports nach Ausführung des Codes


```
;-------------------------------------------------------
; Beispiel zur Deaktivierung einer Linie eines Flag-Ports:
;-------------------------------------------------------
```

```
SETVAL          2,BitNr
RESETBIT        FlagPort,BitNr

; deaktiviert Linie 2 des Flag-Ports
```

ROUND

Syntax

ROUND	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung rundet den **Operanden** zur Einheit ab und setzt das Ergebnis ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL    5.7,op            ;weist der Variablen op 5,7 zu
ROUND     op,var

;Der in die Variable var gesetzte Wert wird 6 sein

SETVAL    5.2,op            ;weist der Variablen op 5,2 zu
ROUND     op,var

;Der in die Variable var gesetzte Wert wird 5 sein
```

SETBIT

Syntax

SETBIT	**Maske, BitNr**

Argumente

Maske	Integer-Konstante oder -Variable oder ZählerName oder Port-Name. Stellt den zu ändernden Wert dar (max. 32 Bit)
BitNr	Integer-Konstante oder -Variable oder ZählerName oder Port-Name. Nummer des zu ändernden Bits

Beschreibung

Diese Anweisung setzt ein einzelnes, von **BitNr** angegebenes Bit der übergebenen Bit-**Maske** auf 1. Das Argument **Maske** muß einem ganzen Wert mit höchstens 32 Bit entsprechen können. Die Bit-Nummer, **BitNr**, geht von 1 bis 32.

Beispiel

Zustand des Ports vor Ausführung des Codes

Zustand des Ports nach Ausführung des Codes


```
;-------------------------------------------------------
; Beispiel zur Aktivierung einer Linie eines Flag-Ports:
;-------------------------------------------------------

SETVAL          2,BitNr
SETBIT          FlagPort,BitNr
```

```
; aktiviert Linie 2 des Flag-Ports
```

SHIFTL

Syntax

SHIFTL **Operand1 [, Operand2]**

Argumente

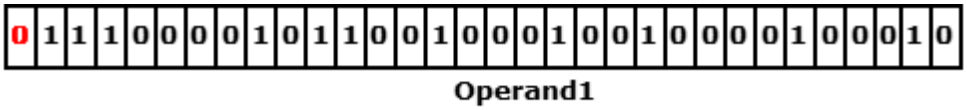
Operand1 Variable (integer oder char) oder Name einer Vorrichtung
Operand2 Variable (integer oder char) oder Name einer Vorrichtung

Beschreibung

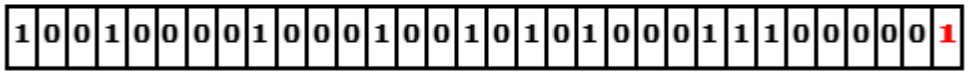
Ist **Operand2** nicht bestimmt, dann wird eine Verschiebungsvorgänge nach links der Bits ausgeführt, die den **Operanden1** zusammenstellen.
Ist auch der zweite Operand bestimmt, dann wird eine Rotation zwischen den Bits des **Operanden2**, und den Bits des **Operanden1** ausgeführt. Am Ende dieses Vorgangs wird der **Operand2** den Übertrag enthalten, d.h. den hohen Bit des **Operanden1.**

Beispiel

Rotation von Integer-Operanden (Verschiebung nach links mit Übertrag)
Vor der Rotation



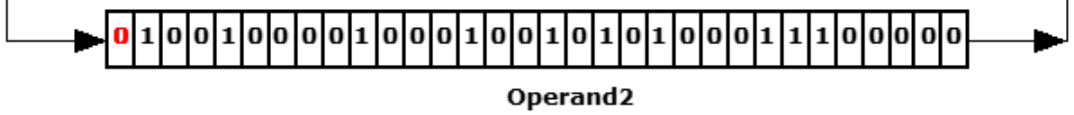
Operand1



Operand2

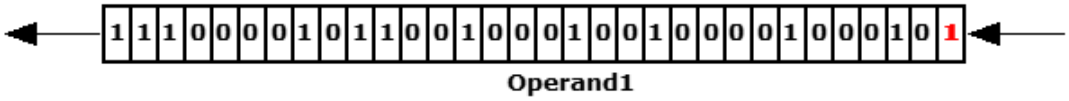
Nach der Rotation

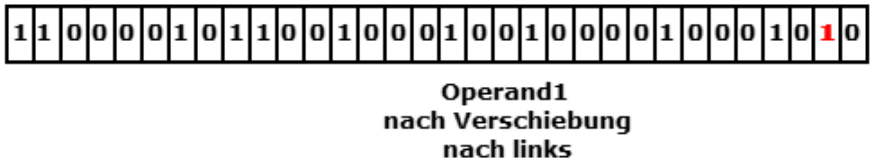
Operand1



Operand2

Verschiebung (Verschiebung nach links ohne Übertrag





SHIFTR

Syntax

SHIFTR **Operand [, Ergebnis]**

Argumente

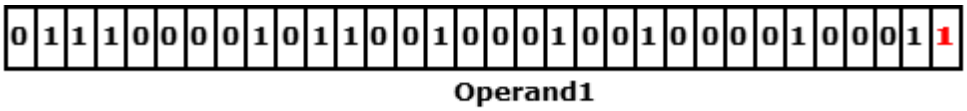
Operand1 Variable (integer oder char) oder Name einer Vorrichtung
Operand2 Variable (integer oder char) oder Name einer Vorrichtung

Beschreibung

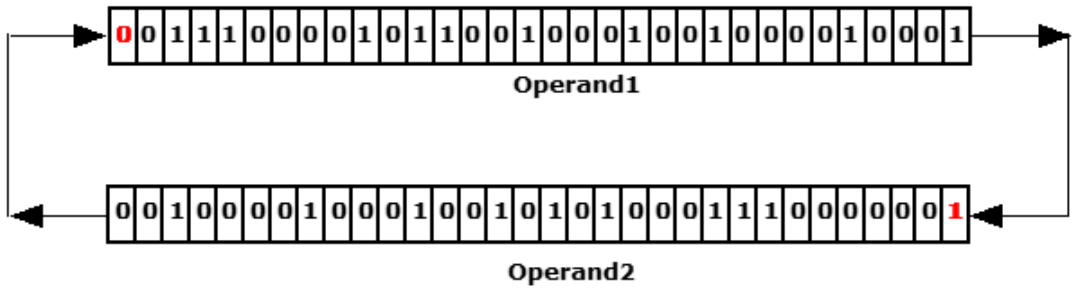
Ist **Operand2** nicht bestimmt, dann wird einen VerschiebungsVorgang nach rechts der Bits ausgeführt, die den **Operanden1** zusammenstellen.
Wenn **Operand1** als Char-Typ deklariert wird, ist das High Input Bit immer 0. Wenn **Operand1** als Integer deklariert ist, ist Bit 32 das Vorzeichenbit.
Ist auch der zweite Operand bestimmt, dann wird eine Rotation zwischen **dem Operanden2**, als Wert 0 oder ungleich 0 gemeint, und den Bits des **Operanden1** ausgeführt. Am Ende dieses Vorgangs wird der **Operand2** den Übertrag des Vorgangs enthalten; das Bit von **Operanden1** mit höherem Gewicht wird 0 oder 1 in Abhängigkeit vom Anfangswert des **Operanden2** (0 oder 1) sein.

Beispiel

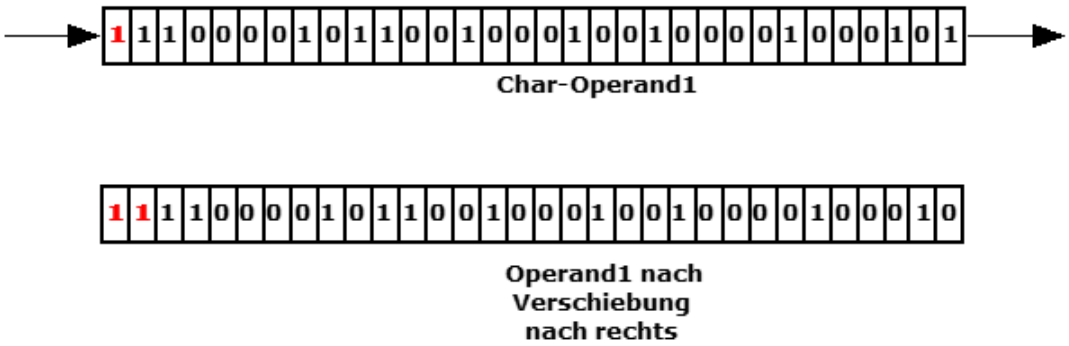
Rotation von Integer-Operanden (Verschiebung nach rechts mit Übertrag)
Vor der Rotation

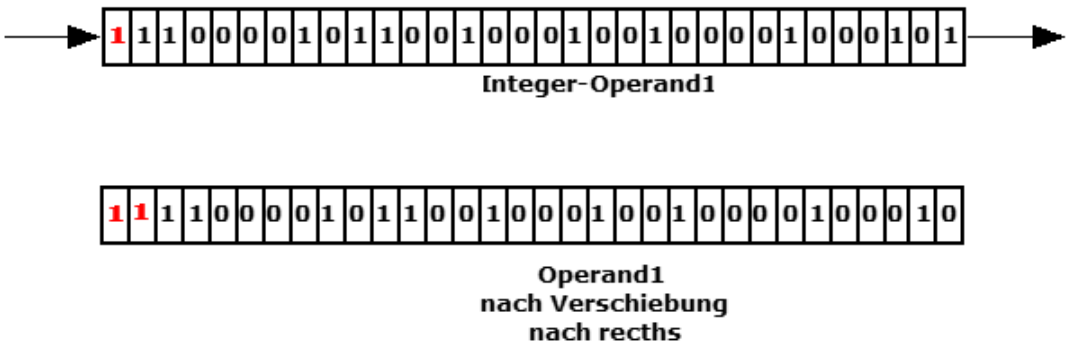


Nach der Rotation



Verschiebung nach rechts eines Chars (Verschiebung nach rechts ohne Übertrag)





SIN

Syntax

SIN	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den Sinus des **Operands** und setzt den Wert ins **Ergebnis**.
Das Argument **Operand** ist in Grad und etwaige Bruchwerte sind in Hundertsteln ausgedrückt (Bsp.: 30° 15" = 30,25.). Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SET 30,op ;weist der Variablen op 30 zu
VAL
 Sin op,va
     r

;Der in die Variable var gesetzte Wert wird 0,5 sein
```

SQR

Syntax

SQR	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet die Quadratwurzel des **Operands** und setzt den Wert ins **Ergebnis**.
Für den Parameter **Operand** sind nur positive Werte zulässig. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL  81,op       ;weist der Variablen op 81 zu
SQR     op,var

;Der in die Variable var gesetzte Wert wird 9 sein
```

SUB

Syntax

SUB	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt eine Subtraktion zwischen **Operand1** und **Operand2** aus und setzt das Ergebnis ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL      10,op1      ; weist der Variablen op1 10 zu
SETVAL      4,op2       ; weist der Variablen op2 4 zu
SUB         op1,op2,var

;Der in die Variable var gesetzte Wert wird 6 sein
```

TAN

Syntax

TAN	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung

Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung berechnet den Tangens des **Operands** und setzt den Wert ins **Ergebnis**. Das Argument **Operand** ist in Grad ausgedrückt. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL    45,op        ;weist der Variablen op 45 zu
TAN       op,var

;Der in die Variable var gesetzte Wert wird 1 sein
```

TRUNC

Syntax

TRUNC	**Operand, Ergebnis**

Argumente

Operand	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung schneidet den vom **Operanden** ausgedrückten Wert zur ganzen Zahl ab und setzt den Wert ins **Ergebnis**. (Die Dezimalwerte gehen verloren). Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
SETVAL  5.7,op       ;weist der Variablen op 5,7 zu
TRUNC   op,var

;Der in die Variable var gesetzte Wert wird 5 sein
```

XOR

Syntax

XOR	**Operand1, Operand2, Ergebnis**

Argumente

Operand1	Konstante oder Variable oder Name einer Vorrichtung
Operand2	Konstante oder Variable oder Name einer Vorrichtung
Ergebnis	Variable oder Name einer Vorrichtung

Beschreibung

Diese Anweisung führt ein binäres XOR (zwischen zwei Bits, ergibt 1, wenn nur eines von beiden 1 ist) zwischen **Operand1** und **Operand2** aus und setzt das Ergebnis ins **Ergebnis**. Zum Konvertieren von Daten gemäß dem angegebenen Datentyp, verweisen wir Sie auf das Kapitel Daten konvertieren.

Beispiel

```
XOR              5,3,var

;Der Wert der Variablen var wird 6 sein
;(binäre Schreibweise: 5 = 0101, 3 = 0011, 6 = 0110)
```

10.3.11 Multitasking

ENDMAIL

Syntax

ENDMAIL **mail**

Argumente

mail Konstante oder Variable des Typs Integer. Nummer der Mailbox (1÷256)

Beschreibung

Diese Anweisung weist auf das Ende der Ausführung eines Befehls hin, der mit einer aus der Mailbox **mail** entnommenen Nachricht zusammenhängt.
Der Task, der die Nachricht gesendet hatte (mittels der Anweisung SENDMAIL) und die Ausführung des Befehls abwartete (Warte-Argument WAITACK), kann somit die eigene Ausführung fortsetzen. Diese Anweisung ist **nur wirksam, wenn sie von dem Task ausgeführt wird**, der zuvor die Nachricht erhalten hatte (mittels der Anweisung WAITMAIL oder TESTMAIL).

Siehe auch die Anweisungen SENDMAIL, WAITMAIL und TESTMAIL.

Beispiel

Server der Achsenbewegung

ENDREALTIMETASK

Syntax

ENDREALTIMETASK **FunktionsName**

Argumente

FunktionsName Name der Funktion

Beschreibung

Diese Anweisung beendet die Ausführung von einem Real time. Siehe auch STARTREALTIMETASK.

ENDTASK

Syntax

ENDTASK **[TaskName]**

Argumente

TaskName TaskName

Beschreibung

Diese Anweisung beendet die Ausführung eines Tasks und aller von ihm aktivierten Tasks (Tochter-Tasks).
Diese Anweisung hält außerdem die Bewegung der Achsen an, löscht unerfüllte RECEIVE-Anweisungen und schließt etwaige Verbindungen mit seriellen Ports.
Wird die Variable **TaskName** ausgelassen, wird die Ausführung des laufenden Tasks beendet.

GETPRIORITYLEVEL

Syntax

GETPRIORITYLEVEL **Niveau [,FunktionsName]**

Argumente

Niveau Variable. Prioritätsniveau der Ausführung.
FunktionsName Name der Funktion

Beschreibung

Diese Anweisung gibt in die Variable **Niveau** den Prioritätswert des von **FunktionsName** definierten Tasks zurück. Dieser Wert entspricht einer Zahl zwischen 1 und 255, wobei 1 die höchste und 255 die niedrigste Priorität darstellt. Ist **FunktionsName** nicht definiert, wird der Prioritätswert des

laufenden Tasks zurückgegeben, d.h. der Funktion, in der die Anweisung GETPRIORITYLEVEL ausgeführt wird.
Siehe auch SETPRIORITYLEVEL.

GETREALTIME

Syntax

GETREALTIME **VarName**

Argumente

VarName Integer-Variable

Beschreibung

Diese Anweisung gibt in die Variable **VarName** die Zeit zurück, die seit der letzten real time zur Verwaltung der Achsen vergangen ist. Die zurückgegebene Zeit ist in Mikrosekunden ausgedrückt. Siehe auch GETREALTIMECOUNT.

GETREALTIMECOUNT

Syntax

GETREALTIMECOUNT **VarName**

Argumente

VarName Integer-Variable

Beschreibung

Diese Anweisung gibt in die Variable **VarName** die Anzahl an real time - Tasks zur Verwaltung der Achsen zurück, die seit der letzten Initialisierung der Numeriksteuerung ausgeführt worden ist. Siehe auch GETREALTIME.

HOLDTASK

Syntax

HOLDTASK **[TaskName]**

Argumente

TaskName TaskName

Beschreibung

Diese Anweisung unterbricht die Ausführung des in **TaskName** definierten Tasks. Diese Anweisung unterbricht nicht die Bewegung der Achsen, die mit STOP-Anweisungen anzuhalten sind.
Wird **TaskName** ausgelassen, wird die Ausführung des laufenden Tasks unterbrochen.

RESUMETASK

Syntax

RESUMETASK **[TaskName]**

Argumente

TaskName TaskName

Beschreibung

Diese Anweisung nimmt die Ausführung des in **TaskName** definierten Tasks wieder auf. Wird **TaskName** ausgelassen, wird die Ausführung des laufenden Tasks wiederaufgenommen. War der Task mittels der Anweisung STOPTASK angehalten worden, werden auch etwaige Bewegungen der Achsen wiederaufgenommen.

SENDMAIL

Syntax

SENDMAIL **mail, Warten [, VarName1 [,..VarNameN]]**
SENDMAIL **mail, Warten, Matrix[Zeile]**

Argumente

mail	Konstante oder Integer-Variable. Nummer der Mailbox (1÷256)
Warten	Vordefinierte Konstante. Warte- oder Ausführung-Modus des Lesens des Befehls. Die zulässigen Werte lauten: **WAIT** wartet das Lesen des Befehls ab **NOWAIT** wartet das Lesen des Befehls nicht ab **WAITACK** wartet die Ausführung des Befehls ab
VarName1[...VarNameN]	Konstante oder Integer-Variable. Namen der Variablen 1÷20
Matrix[Zeile]	Konstante oder Integer-Variable. Nummer der Zeile der Matrix

Beschreibung

Diese Anweisung sendet der Mailbox **mail** eine Nachricht (oder einen Befehl). Die Nachrichten können zur Synchronisierung und zum Austausch von Informationen zwischen zwei oder mehreren Tasks verwendet werden.

Existiert die Mailbox **mail** nicht oder ist noch keine WAITMAIL oder TESTMAIL-Anweisung ausgeführt worden, wird die Anweisung ignoriert.

Erwartet der empfangende Task keine Nachricht (Anweisung WAITMAIL) oder ist er belegt, werden die von der Anweisung angegebenen Daten (**VarName** (1÷20) oder die durch **Matrix[Zeile]** angegebene **Matrix)** in einer Warteschlange gespeichert. In diesem Fall:

1. fährt die Ausführung mit der nächsten Anweisung fort, wenn das Warte-Argument **NOWAIT** ist;
2. wartet die Ausführung ab, daß der empfangende Task die Nachricht liest, wenn das Warte-Argument **WAIT** ist;
3. wartet die Ausführung ab, daß die Nachricht gelesen und die Ausführung des Befehls seitens des empfangenden Tasks bestätigt wird (mittels der Anweisung ENDMAIL oder einer neuen WAITMAIL-Anweisung), wenn das Warte-Argument **WAITACK** ist.

Es ist sehr wichtig, daß die Anzahl und der Typ der übertragenen Variablen mit denen übereinstimmt, die mittels der Anweisung WAITMAIL zur Schaffung der Mailbox verwendet worden sind. Die Steuerung läßt keinen anderen Typ zu und führt auch keine automatischen Umwandlungen (cast) aus, was normalerweise der Fall ist.
Eine SENDMAIL-Anweisung ohne optionelle Parameter (Daten) ist nichts weiter als ein einfacher Mechanismus zur Synchronisierung von Tasks.

Beispiel

Server der Achsenbewegung

SETPRIORITYLEVEL

Syntax

SETPRIORITYLEVEL **Niveau [, FunktionsName]**

Argumente

Niveau	Konstante oder Variable. Prioritätsniveau der Ausführung
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung weist in die Variable **Niveau** den Prioritätswert des von **FunktionsName** definierten Tasks zu. Dieser Wert entspricht einer Zahl zwischen 0 und 255, wobei 0 die höchste und 255 die niedrigste Priorität darstellt. Ist der Name des Tasks in der Variablen **FunktionsName** nicht definiert, wird der Prioritätswert des laufenden Tasks geändert, d.h. das Ausführungsniveau der Funktion, in der die Anweisung ausgeführt wird.
Siehe auch GETPRIORITYLEVEL.

STARTREALTIMETASK

Syntax

STARTREALTIMETASK **FunktionsName**

Argumente

FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung aktiviert die Ausführung von einem Realtime-Task. Dieser Task wird mit derselben Frequenz der Realtime - Kontrolle der Achsen ausgeführt. Er unterscheidet sich von normalen GPL-

Tasks dadurch, daß er bei jeder Realtime vollständig von der ersten Anweisung der Funktion bis zur ersten FRET-Anweisung ausgeführt wird. Siehe auch ENDREALTIMETASK.

Bemerkung:

Die im Realtime-Taske erklärten lokalen Variablen werden nur bei dem Taskanlauf gebraucht und danach behalten sie den Wert der letzten Ausführung.

STARTTASK

Syntax

STARTTASK **TaskName [, Parameter]**

Argumente

TaskName	Name des Tasks
Parameter	eventuell während der Ausführung des Tasks notwendige Parameter

Beschreibung

Diese Anweisung aktiviert die Ausführung des in der Variablen **TaskName** definierten Task.
Dem Task können u.U. während der Ausführung verwendete **Parameter** übergeben werden. Die Anzahl und der Typ der übergebenen Parameter hat mit denen von der Funktion angegebenen übereinzustimmen, die den Task implementiert. Ist der Task bereits in Ausführung begriffen, bleibt die Anweisung wirkungslos.

Beispiel

Sequentielle / Parallele Ausführung

STOPTASK

Syntax

STOPTASK **TaskName**

Argumente

TaskName	Name des Tasks

Beschreibung

Diese Anweisung unterbricht die Ausführung eines Task und aller von ihm aktivierten Tasks (Tochter-Tasks) sowie die Bewegung der Achsen (sofern in Bewegung begriffen).
Wird **TaskName** ausgelassen, wird die Ausführung des laufenden Tasks unterbrochen. Die Ausführung des Tasks und die Bewegung der Achsen kann mittels der Anweisung RESUMETASK wiederaufgenommen werden.

WAITMAIL

Syntax

WAITMAIL **mail [, VarName1 [,..VarNameN]]**
WAITMAIL **mail, Matrix[Zeile]**

Argumente

mail	Integer-Variable oder -Konstante. Nummer der Mailbox (1÷256)
VarName1[...VarNameN]	Integer-Variable oder -Konstante. Namen der Variablen 1÷20
Matrix[Zeile]	Integer-Variable oder -Konstante. Nummer der Zeile der Matrix

Beschreibung

Diese Anweisung erhält eine Nachricht von der Mailbox **mail**. Der Nachricht können Daten zugeordnet sein.
Die gemeinsam mit der Nachricht erhaltenen Daten werden in den angegebenen Variablen **VarName** (1÷20) oder in der durch **Matrix[Zeile]** angegebenen Zeile der Matrix gespeichert.
Sind bei Ausführung der Anweisung WAITMAIL keine noch zu lesenden Nachrichten vorhanden, nimmt der Task den Zustand HOLD an, den er erst verläßt, wenn ein anderer Task der Mailbox eine Nachricht mit der Anweisung SENDMAIL sendet.
Die Übereinstimmung zwischen übertragenen und von der Anweisung erwarteten Daten wird während der Ausführung der Anweisung kontrolliert.
Eine WAITMAIL-Anweisung ohne optionelle Parameter ist nichts weiter als ein einfacher Mechanismus zur Synchronisierung von Tasks.
Siehe auch die Anweisungen SENDMAIL, ENDMAIL und TESTMAIL.

Beispiel
Server der Achsenbewegung

WAITTASK

Syntax
WAITTASK **TaskName**

Argumente
TaskName Task Name

Beschreibung
Diese Anweisung wartet ab, daß der Task **TaskName** die Ausführung beendet hat.

Beispiel
Sequentielle / Parallele Ausführung

10.3.12 Verwaltung des Ablaufs

CALL

Syntax
CALL **UnterprogrammName**

Argumente
UnterprogrammName Name des Unterprogramms, Etikett

Beschreibung
Diese Anweisung führt das vom Etikett definierte Unterprogramm **UnterprogrammName** aus. Jedes Unterprogramm hat am Ausgangspunkt mit der Anweisung RET zu enden, um zur Anweisung nach der CALL-Anweisung zurückkehren zu können.

Anmerkung
Zusammen mit der Anweisung RET stellt diese Anweisung eine typische Fehlerquelle bei der Programmierung dar. Daher empfiehlt sich bei deren Verwendung besondere Aufmerksamkeit, insbesondere ist es ratsam, Unterprogramme ans Ende eines Funktionskörpers zu setzen (nach der Anweisung FRET), um eine versehentliche Ausführung des Unterprogramm-Codes als Bestandteil des Hauptcodes zu vermeiden. Infolge dieser Situation erscheint bestenfalls ein Systemfehler, in anderen Fällen kommt es jedoch zu Betriebsstörungen der Maschine, deren Ursache schwer festzustellen ist.

DELONFLAG

Syntax
DELONFLAG **FlagName**

Argumente
FlagName Name einer Flag-Vorrichtung

Beschreibung
Diese Anweisung deaktiviert die zuvor mittels ONFLAG aktivierte Verwaltung der Software-Unterbrechung, die vom Zustand eines Flag Bits oder Flag Switch abhängt.

DELONINPUT

Syntax
DELONINPUT **EingangName**

Argumente
EingangName Name des Eingangs

Beschreibung
Diese Anweisung deaktiviert die zuvor mittels ONINPUT aktivierte Verwaltung der Software-Unterbrechung, die vom Zustand eines Eingangs abhängt.

FCALL

Syntax

[FCALL] FunktionsName [, Parameter]
FunktionsName [Parameter]

Argumente

FunktionsName	Name der aufzurufenden Funktion
Parameter	eventuell an die Funktion zu übergebende Parameter

Beschreibung

Diese Anweisung führt den Aufruf einer Funktion aus, d.h. die Funktion **FunktionsName** wird ausgeführt. Der Funktion können eventuell **Parameter** übergeben werden. Diese haben der Anzahl und dem Typ nach mit denen übereinzustimmen, die in der aufgerufenen Funktion angegeben worden sind.
Die Ausführung der aufrufenden Funktion (diejenige, in der die Anweisung FCALL ausgeführt wird) wird am Ende der Ausführung der aufgerufenen Funktion (die vom Parameter **FunktionsName** angegebene) fortgesetzt.

Der Unterschied zur STARTTASK- Anweisung besteht darin, daß letztere eine weitere Funktion parallel zur aufrufenden Funktion in Ausführung setzt (dient der gleichzeitigen Ausführung mehrerer Tasks).

Beispiel

Sequentielle / Parallele Ausführung

FOR/NEXT

Syntax

FOR Index, Anfang, Ende [, step]
Anweisung
Anweisung
...
NEXT

Argumente

Index	Variable oder ZählerName
Anfang	Konstante oder Variable oder ZählerName. Anfangswert
Ende	Konstante oder Variable oder ZählerName. Endwert
Step	Konstante oder Variable oder ZählerName. Zu- oder Abnahmewert

Beschreibung

Diese Anweisung sorgt für die zyklische Wiederholung der zwischen der Anweisung FOR und der Anweisung NEXT enthaltenen Anweisungen.
Während des ersten Zyklus wird die Variable **Index** mit dem Wert der Variablen **Anfang** initialisiert. Beim zweiten Zyklus wird der Wert der Variablen **Index** gleich (**Anfang**+**Step)** sein, und so fort bis die Variable **Index** nicht größer wird als die Variable **Ende** (oder kleiner im Fall eines negativen Werts der Variablen **Step**). Wird die Variable **Step** ausgelassen, wird der Wert per Default auf +1 gesetzt.
Die zwischen FOR und NEXT enthaltenen Anweisungen können die Anzahl Wiederholungen ändern, indem sie **Index** ändern.
Wenn die Wiederholungen beendet sind, wird die der Anweisung NEXT folgende Anweisung ausgeführt.

Beispiel

```
FUNCTION Loop
  LOCAL i As integer
  LOCAL Vektor[10] as integer

  FOR          i,1,10
    SETVAL     i, Vektor[i]      ; füllt die Elemente des Vektors
                                 ; mit den Ziffern 1,2, .... 10
```

```
  NEXT
FRET



FUNCTION loop2
  LOCAL j As integer
  LOCAL Vektor[10] as integer

  FOR           j,1,10,2

     SETVAL     27, Vektor[j]      ; fügt den Wert 27 in folgende
                                   ; Elemente des Vektors ein: 1,3,5,7,9
  NEXT
FRET
```

FRET

Syntax

FRET

Argumente

Kein Argument

Beschreibung

Rückkehr von einer Funktion. Beendet die Ausführung der Funktion und die Freigabe des Speicherplatzes, das den lokalen Variablen zugewiesen war. War die Funktion mittels einer FCALL-Anweisung in Ausführung gesetzt worden, fährt die Ausführung der aufrufenden Funktion ab der darauffolgenden Anweisung fort.
Waren vorher WAITTASK-Anweisungen mit der laufenden Funktion (die, in der die Anweisung FRET ausgeführt wird) als Argument ausgeführt worden, werden die wartenden Tasks freigegeben.

GOTO

Syntax

GOTO **Etikett**

Argumente

Etikett Etikett

Beschreibung

Diese Anweisung führt einen bedingungslosen Sprung zum Etikett aus, das vom Parameter **Etikett** angegeben wird.
Ein Etikett wird durch ein Schlüsselwort direkt gefolgt vom Zeichen ":" definiert.
Das Etikett hat sich im Körper der Funktion zu befinden, in der die Anweisung GOTO ausgeführt wird.

Anmerkung

Der Körper einer Funktion ist zwischen der Anweisung FUNCTION enthalten, die den Namen der Funktion angibt, und einer vergleichbaren Anweisung, die die darauffolgende Funktion definiert (oder dem Ende der Datei). Daraus folgt die Möglichkeit, Sprünge vom Hauptkörper der Funktion zu etwaigen Unterprogrammen auszuführen (siehe Anweisungen CALL und RET). Diese Programmierungsart ist streng nicht zu empfehlen, da sie eine Quelle zahlreicher, schwer festzustellender Fehler ist.

Beispiel

```
; Funktion, die ein Flag blinken läßt
; (z.B. ein Alarm-Led in einer synoptischen Darstellung)

FUNCTION Loop

loop:
    SETFLAG     Alarm       ; aktiviert das Flag
```

```
    DELAY       1
    RESETFLAG   Alarm       ; deaktiviert das Flag
    DELAY       1
    GOTO        loop

FRET
```

IF/IFVALUE/IF-THEN-ELSE

Syntax

IF **VarName, Vergleichsoperator, Wert, GOTO Etikett**
IF **VarName, Vergleichsoperator, Wert, CALL UnterprogrammName**
IF **VarName,Vergleichsoperator, Wert, FunktionsName**

IF **VarName, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ENDIF

IF **VarName, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

VarName	Konstante oder Variable oder NameDevice
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)
Wert	Konstante oder Variable oder NameDevice
Etikett	Name des Etiketts zu dem man springen soll
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Die Anweisungen IF und IFVALUE sind Synonyme. Es wird die Verwendung der Kurzform empfohlen.
Die Anweisung ermöglicht es, einen Vergleich zwischen **VarName** und **Wert** auszuführen und je nach Ergebnis des Vergleichs einen Vorgang auszuführen.
Ist der Vergleich positiv, wird in den ersten drei Formen ein Sprung zu einem Etikett (GOTO) oder der Aufruf eines Unterprogramms (CALL) oder der Aufruf einer Funktion (FunktionsName) ausgeführt. Am Ende der Ausführung der aufgerufenen Funktion oder des aufgerufenen Unterprogramms fährt die Ausführung ab der darauffolgenden Zeile fort. Ist der Vergleich hingegen negativ, wird die Ausführung des Programms fortgesetzt.
Die Konstruktion IF...THEN erlaubt die Ausführung einer oder mehrerer Anweisungen, die mit Bedingungen verbunden sind. Die zwischen den Schlüsselwörtern THEN und ENDIF enthaltenen Anweisungen werden ausgeführt, wenn der Vergleich zwischen **VarName** und **Wert** positiv ist.
Die Konstruktion IF...THEN...ELSE dient der Definition von zwei Blöcken von Anweisungen, von denen nur einer ausgeführt wird. Ist der Vergleich zwischen **VarName** und **Wert** positiv, werden die zwischen den Schlüsselwörtern THEN und ELSE enthaltenen Anweisungen ausgeführt; ist er negativ, werden die zwischen den Schlüsselwörtern ELSE und ENDIF enthaltenen Anweisungen ausgeführt. In beiden Fällen fährt die Ausführung ab der darauffolgenden Anweisung bis zu ENDIF fort.

Anmerkung

IFVALUE wird aus Gründen der Kompatibilität mit vorherigen GPL-Versionen beibehalten.

IFACC

Syntax

IFACC **Achse, GOTO Etikett**
IFACC **Achse, CALL UnterprogrammName**
IFACC **Achse, FunktionsName**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob die von der Variablen **Achse** definierte Achse beschleunigt wird. Ergibt die Bedingung wahr, erfolgt ein Sprung zu **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

IFAND

Syntax

IFAND **Operand1, Operand2, PrüfWert, GOTO Etikett**
IFAND **Operand1, Operand2, PrüfWert, CALL UnterprogrammName**
IFAND **Operand1, Operand2, PrüfWert, FunktionsName**

```
IFAND                Operand1, Operand2, PrüfWert THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFAND                Operand1, Operand2, PrüfWert THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

Operand1	Konstante oder Variable oder VorrichtungName
Operand2	Konstante oder Variable oder VorrichtungName
PrüfWert	Konstante. Zum Prüfen des Ergebnisses des Vorgangs verwendeter Wert. Kann folgende Werte annehmen: **TRUE** 1 **FALSE** 0
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Es werden zwei Vergleiche ausgeführt, der erste zwischen **Operand1** und **Operand2** und der zweite zwischen dem Ergebnis des ersten Vergleichs und dem **PrüfWert**.
Der erste Vergleich besteht in einem binären AND zwischen **Operand1** und **Operand2**. Die zwei Operanden werden als Bit-Masken angesehen. Hat das Ergebnis des binären ANDs wenigstens ein Bit ungleich 0, ist das Ergebnis des ersten Vergleichs TRUE. Dieses Ergebnis wird daraufhin mit dem **PrüfWert** verglichen. Stimmen die zwei Werte überein, erfolgt der Sprung zum Etikett oder der Aufruf einer Funktion oder eines Unterprogramms.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFBIT

Syntax

IFBIT **Maske, BitNr, Zustand, GOTO Etikett**
IFBIT **Maske, BitNr, Zustand, CALL UnterprogrammName**
IFBIT **Maske, BitNr, Zustand, FunktionsName**

IFBIT **Maske, BitNr, Zustand THEN**
Anweisung
Anweisung
...
ENDIF

IFBIT **Maske, BitNr, Zustand THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

Maske	Integer-Konstante oder -Variable oder ZählerName oder PortName.Wert zu prüfenden
BitNr	Integer-Konstante oder -Variable oder ZählerName. Nummer des Bits (1÷32)
Zustand	vordefinierte Konstante. Zustand zu prüfen über Maske Die zulässigen Werte lauten: **ON** bit gewählt zu 1 **OFF** bit gewählt zu 0
Etikett	Sprung-Etikett (GOTO)
UnterprogrammName	Aufruf des Unterprogramms (CALL)
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft ein einzelnes Bit der übergebenen Bit-**Maske**. Das Argument **Maske** muß einem ganzen Wert mit höchstens 32 Bit entsprechen können. Die der Variablen **BitNr** zuzuweisende Ziffer zur Bestimmung des zu prüfenden Bits geht von 1 bis 32. Ergibt die in **Zustand** angegebene Bedingung wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFBLACKBOX

Syntax

IFBLACKBOX **GOTO Etikett**
IFBLACKBOX **CALL UnterprogrammName**
IFBLACKBOX **FunktionsName**

Argumente

Etikett	Name der Etikett zu der man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Falls die Aufnahme aktiv ist, springt diese Ausweisung zur **Etikette** oder ruft sie **UnterprogrammName** oder **FunktionsName** auf. Siehe auch STARTBLACKBOX, PAUSEBLACKBOX und ENDBLACKBOX.

IFCHANGEVEL

Syntax

IFCHANGEVEL **Achse [, Zustand], GOTO Etikett**
IFCHANGEVEL **Achse [, Zustand], CALL UnterprogrammName**
IFCHANGEVEL **Achse [, Zustand], FunktionsName**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Zustand	Änderung Typ. Die zulässigen Werte lauten: **POSITIVE** **NEGATIVE**
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob eine Änderung der Achsengeschwindigkeit eingetreten ist.
Wenn die von der Variablen **Achse** definierte Achse dabei ist, während einer Bewegung ihre Geschwindigkeit zu wechseln, erfolgt ein Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Der Parameter **Zustand** gibt an, ob die Geschwindigkeit zugenommen (POSITIVE) oder abgenommen hat (NEGATIVE).

IFCOUNTER

Syntax

IFCOUNTER	**ZählerName, Vergleichsoperator, Wert, GOTO Etikett**
IFCOUNTER	**ZählerName, Vergleichsoperator, Wert, CALL UnterprogrammName**
IFCOUNTER	**ZählerName, Vergleichsoperator, Wert, FunktionsName**

Argumente

ZählerName	Name del Zähler
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)
Wert	Konstante oder Variable oder ZählerName
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Die Anweisung prüft einen Zähler.
Wenn der Inhalt des in der Variablen **ZählerName** definierten Zählers die vom **Vergleichsoperator** ausgedrückte Bedingung mit dem in der Variablen **Wert** ausgedrückten Wert erfüllt, erfolgt ein Sprung zu dem von **Etikett** angegebenen Etikett oder ein Aufruf eines von **UnterprogrammName** definierten Unterprogramms oder einer von **FunktionsName** definierten Funktion.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFDEC

Syntax

IFDEC	**Achse, GOTO Etikett**
IFDEC	**Achse, CALL UnterprogrammName**
IFDEC	**Achse, FunktionsName**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob die von der Variablen **Achse** definierte Achse verzögert wird.
Ergibt die Bedingung wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

IFDIR

Syntax

```
IFDIR                    Achse, Richtung, GOTO Etikett
IFDIR                    Achse, Richtung, CALL UnterprogrammName
IFDIR                    Achse, Richtung, FunktionsName

IFDIR                    Achse, Richtung THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFDIR                    Achse, Richtung THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

Achse	Name der Vorrichtung des Typs Achse
Richtung	Richtung Achse. Die zulässigen Werte lauten: **POSITIVE** Richtung Achse positive **NEGATIVE** Richtung Achse negative
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft die laufende Richtung einer Achse.
Bewegt sich die **Achse** in der von der Variablen **Richtung** angegebenen Richtung, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFERRAN

Syntax

```
IFERRAN                  Achse, Vergleichsoperator, Wert, GOTO Etikett
IFERRAN                  Achse, Vergleichsoperator, Wert, CALL UnterprogrammName
IFERRAN                  Achse, Vergleichsoperator, Wert, FunktionsName

IFERRAN                  Achse, Vergleichsoperator, Wert THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFERRAN                  Achse, Vergleichsoperator, Wert THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

Achse	Name del Vorrichtung des Typs Achse
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)
Wert	Konstante oder Variable o ZählerName

Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft den Loop-Fehler der in der Variablen **Achse** definierten Achse.
Wenn der Loop-Fehler der **Achse** die vom **Vergleichsoperator** ausgedrückte Bedingung mit dem von **Wert** ausgedrücktem Wert erfüllt, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFERROR

Syntax

IFERROR **Zahl, PosID, GOTO Etikett**
IFERROR **Zahl, PosID, CALL Etikett**
IFERROR **Zahl, PosID, FunktionsName**
IFERROR **DeviceName, Zustand,PosID, GOTO Etikett**
IFERROR **DeviceName, Zustand,PosID, CALL Etikett**
IFERROR **DeviceName, Zustand,PosID, FunktionsName**

Argumente

Zahl	DEFMSG oder Konstante oder Variable
DeviceName	Device Name
Zustand	Kann folgende Werte annehmen: **ON, OFF**
PosID	Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert
Etikett	Name des Etiketts zu dem man springt
FunktionsName	Name der Funktion

Beschreibung

Untersuchung auf den aktiven Zyklusfehler.
Wenn der Zyklusfehler, der durch **Zahl** und **PosID** oder **DeviceName**, **Zustand** und **PosID** identifiziert wird, aktiv ist, wird ein Sprung zu **Etikett** oder ein Anruf nach der Funktion **FunktionsName** ausgeführt.
Der Parameter **Zahl** kann einen Zyklusfehler eines Moduls (naemlich eine ganze Zahlenwert) oder einer Gruppe (in diesem Fall wird eine DEFMSG gebraucht) identifizieren.
Der Parameter **DeviceName** ist der Name einer Vorrichtung und der Parameter **Zustand** stellt den ON/OFF-Zustand in dem sich bei dem Fehlererzeugnis die Vorrichtung befinden muss.
Der Parameter **PosID** ist ein wahlfreier Parameter. Er spezifiziert den in den synoptischen Schemen gebrauchten Zahlwert zur Verteilung der Zyklusfehler in verschiedene Zellen.
Er muss dem vom Bildner der synoptischen Schemen angegebenen Wert zu diesem Sonderdisplay entsprechen. Sollte es nicht nötig sein eine bestimmte Zelle hinzuweisen, muss die vorausbestimmte Konstante NOPLACE zugeteilt werden. Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen.
Wenn die Instruktion gebraucht wird, ohne dass die Alarmbehandlung zu Zustaenden ermoeglicht ist, tritt ein Systemfehler ein.
Siehe auch die Anweisung ERROR.

IFFLAG

Syntax

IFFLAG **FlagName, Zustand, GOTO Etikett**
IFFLAG **FlagName, Zustand, CALL UnterprogrammName**
IFFLAG **FlagName, Zustand, FunktionsName**

IFFLAG **FlagName, Zustand THEN**
Anweisung
Anweisung
...
ENDIF

IFFLAG **FlagName, Zustand THEN**
Anweisung
Anweisung
...

ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

FlagName	Name der Flag-Vorrichtung
Zustand	vordefinierte Konstante. Zustand zu prüfen. Die zulässigen Werte lauten: **ON** aktiviert **OFF** deaktiviert
Etikett	Name des Etiketts, zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft den logischen Zustand eines Flags.
Befindet sich das von der Variablen FlagName definierte Flag im angegebenen **Zustand**, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFINPUT

Syntax

IFINPUT **EingangName, Zustand, GOTO Etikett**
IFINPUT **EingangName, Zustand, CALL UnterprogrammName**
IFINPUT **EingangName, Zustand, FunktionsName**

IFINPUT **EingangName, Zustand THEN**
Anweisung
Anweisung
...
ENDIF

IFINPUT **EingangName, Zustand THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

EingangName	Name Eingang
Zustand	vordefinierte Konstante. Zustand zu prüfen Die zulässigen Werte lauten: **ON** aktiviert **OFF** deaktiviert
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft den logischen Zustand eines Eingangs.
Befindet sich der in der Variablen **EingangsName** definierte Eingang im angegebenen **Zustand**, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFMESSAGE

Syntax

IFMESSAGE **Zahl, PosID, GOTO Etikett**
IFMESSAGE **Zahl, PosID, CALL Etikett**
IFMESSAGE **Zahl, PosID, FunktionsName**

Argumente

Zahl	DEFMSG Konstante oder Variable
PosID	Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert
Etikett	Name des Etiketts zu dem man springt
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft die aktive Nachricht.
Wenn die Nachricht, die durch **Zahl** und **PosID** identifiziert wird, aktiv ist, wird ein Sprung nach **Etikett** oder ein Anruf nach der Funktion **FunktionsName** ausgeführt.
Der Parameter **PosID** ist ein wahlfreier Parameter. Er spezifiziert den in den synoptischen Schemen gebrauchten Zahlwert zur Verteilung der Zyklusfehler in verschiedene Zellen.
Er muss dem vom Bildner der synoptischen Schemen angegebenen Wert zu diesem Sonderdisplay entsprechen. Sollte es nicht nötig sein, eine bestimmte Zelle hinzuweisen, muss die vorausbestimmte Konstante NOPLACE zugeteilt werden. Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen.
Wenn die Anweisung verwendet wird ohne dass die Alarmbehandlung zu Zuständen ermöglicht wird, tritt ein Systemfehler ein.
Siehe auch die Anweisung MESSAGE.

IFOR

Syntax

```
IFOR                    Operand1, Operand2, PrüfWert, GOTO Etikett
IFOR                    Operand1, Operand2, PrüfWert, CALL UnterprogrammName
IFOR                    Operand1, Operand2, PrüfWert, FunktionsName

IFOR                    Operand1, Operand2, PrüfWert THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFOR                    Operand1, Operand2, PrüfWert THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

Operand1	Konstante oder Variable oder nomedevice
Operand2	Konstante oder Variable oder nomedevice
PrüfWert	Konstante. Wert verwendet zu die Ergebnis des Vorgangs prüfen Kann folgende Werte annehmen: **TRUE** 1 **FALSE** 0
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Es werden zwei Vergleiche ausgeführt, der erste zwischen **Operand1** und **Operand2** und der zweite zwischen dem Ergebnis des ersten Vergleichs und dem **PrüfWert**.
Der erste Vergleich besteht in einem binären OR zwischen **Operand1** und **Operand2**. Die zwei Operanden werden als Bit-Masken angesehen. Hat das Ergebnis des binären ORs wenigstens ein Bit ungleich 0, ist das Ergebnis des ersten Vergleichs TRUE. Dieses Ergebnis wird daraufhin mit dem **PrüfWert** verglichen. Stimmen die zwei Werte überein, erfolgt der Sprung zum Etikett oder der Aufruf einer Funktion oder eines Unterprogramms.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFOUTPUT

Syntax

IFOUTPUT **AusgangName, Zustand, GOTO Etikett**
IFOUTPUT **AusgangName, Zustand, CALL UnterprogrammName**
IFOUTPUT **AusgangName, Zustand, FunktionsName**

IFOUTPUT **AusgangName, Zustand THEN**
Anweisung
Anweisung
...
ENDIF

IFOUTPUT **AusgangName, Zustand THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

AusgangName	Name Ausgang
Zustand	vordefinierte Konstante. Stellt den an der Ausgang zu prüfenden Zustand dar. Die zulässigen Werte lauten: **ON** aktiviert **OFF** deaktiviert
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft den logischen Zustand eines Ausgangs.
Befindet sich der in der Variablen **AusgangName** definierte Ausgang im angegebenen **Zustand**, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFQUOTER

Syntax

IFQUOTER **Achse, Vergleichsoperator, Wert, GOTO Etikett**
IFQUOTER **Achse, Vergleichsoperator, Wert, CALL UnterprogrammName**
IFQUOTER **Achse, Vergleichsoperator, Wert, FunktionsName**

IFQUOTER **Achse, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ENDIF

IFQUOTER **Achse, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

Achse	Name der Vorrichtung des Typs Achse
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich)

	> (größer als) **=<** (kleiner gleich)
	>= (größer gleich) **<>** (nicht gleich)
Wert	Konstante oder Variable oder ZählerName
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft das tatsächliche, von der Variablen **Achse** ausgedrückte Maß.
Wenn der Wert der Variablen **Achse**, die vom **Vergleichsoperator** ausgedrückte Bedingung mit dem von **Wert** ausgedrücktem Wert erfüllt, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFQUOTET

Syntax

```
IFQUOTET                    Achse, Vergleichsoperator, Wert, GOTO Etikett
IFQUOTET                    Achse, Vergleichsoperator, Wert, CALL UnterprogrammName
IFQUOTET                    Achse, Vergleichsoperator, Wert, FunktionsName

IFQUOTET                    Achse, Vergleichsoperator, Wert THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFQUOTET                    Achse, Vergleichsoperator, Wert THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

Achse	Name der Vorrichtung des Typs Achse
Vergleichsoperator	die Symbole für den Vergleich sind:
	< (kleiner als) **=** (gleich)
	> (größer als) **=<** (kleiner gleich)
	>= (größer gleich) **<>** (nicht gleich)
Wert	Konstante oder Variable oder ZählerName
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft das theoretische, von der Variablen **Achse** ausgedrückte Maß.
Wenn der Wert der Variablen **Achse,** die vom **Vergleichsoperator** ausgedrückte Bedingung mit dem von **Wert** ausgedrücktem Wert erfüllt, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFRECEIVED

Syntax

```
IFRECEIVED                  [Quelle, ] Identifizierer, GOTO Etikett
IFRECEIVED                  [Quelle, ] Identifizierer, CALL UnterprogrammName
IFRECEIVED                  [Quelle, ] Identifizierer, FunktionsName
```

Argumente

Quelle	String-Konstante
Identifizierer	String-Konstante
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms

FunktionsName Name der Funktion

Beschreibung

Diese Anweisung prüft, ob eine RECEIVE-Anweisung erfüllt worden ist.
Ist eine bestimmte, vorhergehende RECEIVE-Anweisung erfüllt worden, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Siehe auch die Anweisungen RECEIVE, WAITRECEIVE, SEND.

IFREG

Syntax

IFREG	**Achse, GOTO Etikett**
IFREG	**Achse, CALL UnterprogrammName**
IFREG	**Achse, FunktionsName**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob sich die von der Variablen **Achse** definierte Achse bei konstanter Geschwindigkeit bewegt.
Ergibt die Bedingung wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

IFSAME

Syntax

IFSAME	**Operand1, Operand2, GOTO Etikett**
IFSAME	**Operand1, Operand2, CALL UnterprogrammName**
IFSAME	**Operand1, Operand2, FunktionsName**

Argumente

Operand1	Variable oder NameDevice
Operand2	Variable oder NameDevice
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Test zwischen zwei Operanden.
Diese Anweisung prüft, daß der in **Operand1** und **Operand2** definierte Wert entweder auf dieselbe Vorrichtung oder auf denselben Speicherbereich bezogen sind.
Ergibt der Test zwischen den zwei Operanden wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

IFSTILL

Syntax

IFSTILL	**Achse, GOTO Etikett**
IFSTILL	**Achse, CALL UnterprogrammName**
IFSTILL	**Achse, FunktionsName**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob die in der Variablen **Achse** definierte Achse stillsteht bzw. "am Maß" ist.
Ergibt die Bedingung wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

Siehe auch IFTARGET und IFWIN.

IFSTR

Syntax

```
IFSTR                   String1, Vergleichsoperator, String2, GOTO Etikett
IFSTR                   String1, Vergleichsoperator, String2, CALL
                        UnterprogrammName
IFSTR                   String1, Vergleichsoperator, String2, FunktionsName

IFSTR                   String1, Vergleichsoperator, String2 THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFSTR                   String1, Vergleichsoperator, String2 THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

String1	String-Variable. Si ist die erste ASCII-Zeichenfolge
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)
String2	String-Variable. Sie ist die zweiste ASCII-Zeichenfolge
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft ASCII-Zeichenfolge.
Wenn die in **String1** definierte Zeichenfolge die vom **Vergleichsoperator** ausgedrückte Bedingung mit der in **String2** enthaltenen Zeichenfolge erfüllt, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFTARGET

Syntax

```
IFTARGET                Achse, GOTO Etikett
IFTARGET                Achse, CALL UnterprogrammName
IFTARGET                Achse, FunktionsName
```

Argumente

Achse	Name der Vorrichtung des Typs Achse
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob die von der Variablen **Achse** definierte Achse das programmierte Endmaß erreicht hat. Auch wenn sie das Target-Maß erreicht hat, steht die Achse nicht notwendigerweise still, da sie normalerweise noch den Loop-Fehler auszugleichen hat. Ergibt die Bedingung wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Siehe auch IFSTILL und IFWIN.

IFTASKHOLD

Syntax

IFTASKHOLD **TaskName, GOTO Etikett**
IFTASKHOLD **TaskName, CALL Etikett**
IFTASKHOLD **TaskName, FunktionsName**

Argumente

TaskName	TaskName parallel
Etikett	Name des Etikettszu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob der Task stillsteht (Hold-Zustand).
Steht der Task **TaskName** still, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

IFTASKRUN

Syntax

IFTASKRUN **TaskName, GOTO Etikett**
IFTASKRUN **TaskName, CALL UnterprogrammName**
IFTASKRUN **TaskName, FunktionsName**

Argumente

TaskName	TaskName parallel
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob der Task in Ausführung begriffen ist.
Ist der in **TaskName** definierte Task in Ausführung begriffen, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.

IFTIMER

Syntax

IFTIMER **TimerName, Vergleichsoperator, Wert, GOTO Etikett**
IFTIMER **TimerName, Vergleichsoperator, Wert, CALL UnterprogrammName**
IFTIMER **TimerName, Vergleichsoperator, Wert, FunktionsName**

IFTIMER **TimerName, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ENDIF

IFTIMER **TimerName, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

TimerName	Name der Vorrichtung des Typs Timer
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)

Wert	Konstante oder Variable oder TimerName. Wert in der vergleichen wird
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft einen Timer.
Wenn der Inhalt des Timers **TimerName** die vom **Vergleichsoperator** ausgedrückte Bedingung mit dem von **Wert** ausgedrückten Wert erfüllt, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFVEL

Syntax

IFVEL **Achse, Vergleichsoperator, Wert, GOTO Etikett**
IFVEL **Achse, Vergleichsoperator, Wert, CALL UnterprogrammName**
IFVEL **Achse, Vergleichsoperator, Wert, FunktionsName**

IFVEL **Achse, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ENDIF

IFVEL **Achse, Vergleichsoperator, Wert THEN**
Anweisung
Anweisung
...
ELSE
Anweisung
Anweisung
...
ENDIF

Argumente

Achse	Name der Vorrichtung des Typs Achse
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft die laufende Geschwindigkeit einer Achse.
Wenn die Geschwindigkeit der Achse die vom **Vergleichsoperator** ausgedrückte Bedingung mit dem von **Wert** ausgedrückten Wert erfüllt, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Für weitere Details siehe den Satz IF-THEN-ELSE.

IFWIN

Syntax

IFWIN **Achse, GOTO Etikett**
IFWIN **Achse, CALL UnterprogrammName**
IFWIN **Achse, FunktionsName**

Argumente

Achse	Name der Vorrichtung des Typs Achse
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung prüft, ob sich die von der Variablen **Achse** definierte Achse innerhalb der Positionierungstoleranz befindet (siehe Konventionen und Begriffe).
Ergibt die Bedingung wahr, erfolgt der Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Siehe auch IFTARGET und IFSTILL.

IFXOR

Syntax

```
IFXOR                   Operand1, Operand2, PrüfWert, GOTO Etikett
IFXOR                   Operand1, Operand2, PrüfWert, CALL UnterprogrammName
IFXOR                   Operand1, Operand2, PrüfWert, FunktionsName

IFXOR                   Operand1, Operand2, PrüfWert THEN
        Anweisung
        Anweisung
        ...
ENDIF

IFXOR                   Operand1, Operand2, PrüfWert THEN
        Anweisung
        Anweisung
        ...
ELSE
        Anweisung
        Anweisung
        ...
ENDIF
```

Argumente

Operand1	Konstante oder Variable oder NameDevice
Operand2	Konstante oder Variable oder NameDevice
PrüfWert	Konstante. Zum Prüfen des Ergebnisses des Vorgangs verwendeter Wert. Kann folgende Werte annehmen: **TRUE** 1 **FALSE** 0
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Es werden zwei Vergleiche ausgeführt, der erste zwischen **Operand1** und **Operand2** und der zweite zwischen dem Ergebnis des ersten Vergleichs und dem **PrüfWert**.
Der erste Vergleich besteht in einem binären XOR zwischen **Operand1** und **Operand2**. Die zwei Operanden werden als Bit-Masken angesehen. Hat das Ergebnis des binären XORs wenigstens ein Bit ungleich 0, ist das Ergebnis des ersten Vergleichs TRUE. Dieses Ergebnis wird daraufhin mit dem **PrüfWert** verglichen. Stimmen die zwei Werte überein, erfolgt der Sprung zum Etikett oder der Aufruf einer Funktion oder eines Unterprogramms.
Für weitere Details siehe den Satz IF-THEN-ELSE.

ONERRSYS

Syntax

```
ONERRSYS                FunktionsName
```

Argumente

FunktionsName	Name der Funktion

Beschreibung

Diese Anweisung aktiviert die Verwaltung der Systemfehler. Beim Auftreten eines Systemfehlers besteht das normale Verhalten der Steuerung im Beenden aller Tasks. Die Verwaltung der Systemfehler erlaubt das Beenden der Tasks zu vermeiden, für die sie aktiviert worden ist.

Wenn ein Systemfehler auftritt, wird die Ausführung der Funktion **FunktionsName** gestartet. Aufgabe dieser Funktion ist es, den Systemfehler zu analysieren und angemessene Maßnahmen für die Sicherheit der Maschine zu treffen.

Die Funktion **FunktionsName** untersteht zwei Einschränkungen.
Erstens hat sie folgende Parameter zu akzeptieren:
- die Nummer des Systemfehlers, als Integer
- der Task, in dem der Fehler aufgetreten ist, als Funktion
- die Vorrichtung, die den Fehler hervorgerufen hat, als Vorrichtung

Zweitens kann sie eine bestimmte Anzahl von GPL-Anweisungen nicht enthalten. Siehe Liste der bei Interrupt nicht verwendbaren Anweisungen.

Im Fall von mehreren Systemfehlern wird die Funktion sequentiell ein Mal pro Fehler aufgerufen. Ruft die Funktion selbst einen Systemfehler hervor, werden alle Tasks beendet.

Während der Ausführung der Funktion wird der Task, für den die Fehlerverwaltung aktiviert worden ist, angehalten, und nur am Ende der ersten von ONERRSYS aufgerufenen Funktion wieder fortgesetzt. Insbesondere wird der Task beim Ausführen der vom Systemfehler unterbrochenen Anweisung wieder starten.

Beispiel

Main-Zyklus mit Fehlerverwaltung

ONFLAG

Syntax

ONFLAG	**FlagName, [Zustand,] FunktionsName[,Argumente]**

Argumente

FlagName	Name der Vorrichtung des Typs Flag
Zustand	vordefinierte Konstante. Zustand zu prüfen. Die zulässigen Werte lauten: **ON** aktiviert **OFF** deaktiviert
FunktionsName	Name der Funktion
Argumente	optionale Argumente der Funktion

Beschreibung

Diese Anweisung aktiviert die Software-Unterbrechung des in Ausführung begriffenen Tasks, wenn das Flag während der Ausführung den angegebenen Zustand annimmt. Wenn das Flag den angegebenen Zustand annimmt (Interrupt), wird die Ausführung des Tasks unterbrochen und die Ausführung der von **FunktionsName** angegebenen Funktion gestartet. Danach fährt die Ausführung des Tasks dort fort, wo sie unterbrochen worden war. Die bei Auftreten des Interrupts ausgeführte Funktion unterliegt Einschränkungen. Insbesondere sind nicht alle GPL-Anweisungen im Funktionskörper zulässig. Diese Einschränkung hat den Zweck, Situationen wie kritisches Blockieren des GPL-Codes oder überlange Wartezeiten zu vermeiden. Siehe Liste der bei Interrupt nicht verwendbaren Anweisungen.
Wird das Argument **Zustand** ausgelassen, wird die Funktion bei jedem Zustandswechsel des Flags aufgerufen.
Der Test des Flag-Zustands wird alle 5 ms ausgeführt, d.h. zwischen dem Wechsel des Flags und der Ausführung der Funktion liegt eine maximale Latenzzeit von 5 ms.
Es können nicht mehrere ONFLAG am selben Flag definiert werden.
Als Argumente der in **FunktionsName** definierten Funktion können keine Vektoren oder lokalen Matrizen verwendet werden.
Siehe auch die Anweisungen DELONFLAG, ONINPUT, DELONINPUT.

ONINPUT

Syntax

ONINPUT	**EingangName, [Zustand,] FunktionsName [,Argumente]**

Argumente

EingangName	Eingang Name
Zustand	vordefinierte Konstante. Zustand zu prüfen. Die zulässigen Werte lauten: **ON** aktiviert

	OFF deaktiviert
FunktionsName	Name der Funktion
Argumente	optionale Argumente der Funktion

Beschreibung

Diese Anweisung aktiviert die Software-Unterbrechung des in Ausführung begriffenen Tasks, wenn das Flag während der Ausführung den angegebenen Zustand annimmt. Wenn das Flag den angegebenen Zustand annimmt (Interrupt), wird die Ausführung des Tasks unterbrochen und die Ausführung der von **FunktionsName** angegebenen Funktion gestartet. Danach fährt die Ausführung des Tasks dort fort, wo sie unterbrochen worden war. Die bei Auftreten des Interrupts ausgeführte Funktion unterliegt Einschränkungen. Insbesondere sind nicht alle GPL-Anweisungen im Funktionskörper zulässig. Diese Einschränkung hat den Zweck, Situationen wie kritisches Blockieren des GPL-Codes oder überlange Wartezeiten zu vermeiden. Siehe Liste der bei Interrupt nicht verwendbaren Anweisungen.
Wird das Argument **Zustand** ausgelassen, wird die Funktion bei jedem Zustandswechsel des Eingangs aufgerufen.
Der Test des Zustands des Eingangs wird alle 5 ms ausgeführt; dazu kommen 4 ms Antiprell-Filter der Eingangsverwaltung, d.h. vor Ausführung der Funktion kann es zu einer Latenzzeit von 9 ms kommen.
Es können nicht mehrere ONINPUT am selben Eingang definiert werden.
Siehe auch die Anweisungen DELONINPUT, ONFLAG und DELONFLAG.

REPEAT/ENDREP

Syntax

REP Wert

Anweisung
Anweisung
...
END REP

Argumente

Wert	Konstante oder Variable oder ZählerName. Anzahl Wiederholungen

Beschreibung

Diese Anweisung wiederholt die zwischen der Anweisung REPEAT und der Anweisung ENDREPEAT enthaltenen Anweisungen so oft wie von der Variablen **Wert** angegeben.
Wenn das Programm auf die Anweisung ENDREP stößt, zählt der Zähler der Wiederholungen rückwärts; solange die Zahl nicht kleiner als oder Null ist, wird der Anweisungsblock ab der Anweisung auf der Zeile nach REPEAT erneut ausgeführt. Die Anweisungen werden also wenigstens einmal ausgeführt (selbst wenn der Parameter Wert von Anfang Null oder negativ ist).
Wenn die Wiederholungen beendet sind, wird die Anweisung nach ENDREP ausgeführt.
Siehe auch die Anweisung FOR/NEXT.

Beispiel

```
; Beispiel eines Zyklus zur Bewegung einer Achse
; 10 Mal zwischen zwei Maßen

FUNCTION Zyklus
  REPEAT      10
  MOVABS      Achse,100
  WAITINPUT   Switch,ON
  MOVABS      Achse,-100
  WAITINPUT   switch, OFF
  ENDREP
FRET
```

RET

Syntax

RET

Argumente

kein Argument

Beschreibung

Diese Anweisung beendet die Ausführung des Unterprogramms und kehrt zur Anweisung direkt nach der aufrufenden CALL-Anweisung zurück.
Siehe auch die Anweisung CALL.

Bemerkung

Zusammen mit der Anweisung CALL stellt diese Anweisung eine typische Fehlerquelle bei der Programmierung dar. Daher empfiehlt sich bei deren Verwendung besondere Aufmerksamkeit, insbesondere ist es ratsam, Unterprogramme ans Ende eines Funktionskörpers zu setzen (nach der Anweisung FRET), um eine versehentliche Ausführung des Unterprogramm-Codes als Bestandteil des Hauptcodes zu vermeiden. Infolge dieser Situation erscheint bestenfalls ein Systemfehler, in anderen Fällen kommt es jedoch zu Betriebsstörungen der Maschine, deren Ursache schwer festzustellen ist.

SELECT

Syntax

SELECT VarName
CASE Wert
Anweisung
CASE Wert1 TO Wert2
Anweisung
CASE IS < => Wert
Anweisung
CASE ELSE
Anweisung
ENDSELECT

Anweisung muss von einem der folgenden Werten ersetzt werden:
GOTO Etikett
CALL UnterprogrammName
[FCALL] FunktionsName [Parameter1,...ParameterN]
[EXPR] Variable = Ausdruck
[EXPR] Vorrichtung = Ausdruck

Argumente

VarName	Konstante oder Integer-Variable oder ZählerName
Wert, Wert1, Wert2	Integer-Konstanten
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion
Parameter1...ParameterN	An die aufgerufene Funktion übergebener Parameter
Variable	Name der Variable
Vorrichtung	Name der Vorrichtung
Ausdruck	Operatorensatz

Beschreibung

Mehrfachauswahl basierend auf dem **Wert** der Variablen **VarName.** Der Code, der sich im CASE der nachgeprüften Bedingung befindet, wird ausgeführt. Der Zweig CASE - ELSE wird ausgeführt, wenn kein vorheriger CASE erfüllt ist. Für jeder CASE (optional) kann es nur eine GOTO, CALL, FCALL oder EXPR - Anweisung geben.
Es muss mindestens ein CASE zwischen SELECT und ENDSELECT geben. Letzteres signalisiert das Ende der SELECT-Anweisung.
Nach jedem CALL, FCALL oder EXPR wird die Funkionsausführung bis zur Anweisung nach ENDSELECT fortgesetzt.

Mehrfach-Auswahl mit Sprung zu **Etikett**, Aufruf von **UnterprogrammName** oder **FunktionsName** aufgrund des **Werts** der Variablen **VarName**.

Pro CASE (Option) ist nur eine GOTO-, CALL- oder FCALL-Anweisung zulässig.
Wenigstens ein CASE muß zwischen SELECT und ENDSELECT enthalten sein. Letztere weist auf das Ende der SELECT-Anweisung hin.
Nach jeder CALL oder FCALL-Anweisung fährt die Ausführung der Funktion ab der Anweisung nach ENDSELECT fort.
Der Zweig CASE - ELSE wird ausgeführt, wenn kein CASE von den vorhergehenden erfüllt wird.

Beispiel

Server der Achsenbewegung

TESTIPC

Syntax

TESTIPC	**IPCName, [, VarName1 [, VarNameN, ...]], GOTO Etikett**
TESTIPC	**IPCName, [, VarName1 [, VarNameN, ...]], CALL UnterprogrammName**
TESTIPC	**IPCName, [, VarName1 [, VarNameN, ...]], FunktionsName**
TESTIPC	**IPCName, Matrix[Zeile], GOTO Etikett**
TESTIPC	**IPCName, Matrix[Zeile], CALL UnterprogrammName**
TESTIPC	**IPCName, Matrix[Zeile], FunktionsName**
TESTIPC	**IPCName, Vektor, GOTO Etikett**
TESTIPC	**IPCName, Vektor, CALL UnterprogrammName**
TESTIPC	**IPCName, Vektor, FunktionsName**

Argumente

IPCName	String-Konstante.Name der IPC
VarName1[...VarNameN]	Konstante oder Variable. Namen der Variablen 1÷N
Matrix[Zeile]	Konstante oder Integer-Variable. Nummer der Zeile der Matrix
Vektor	Name des Vektors
Matrix	Name der Matrix
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Test und Empfang eines IPC-Befehls.
Bei der ersten Ausführung einer TESTIPC-Anweisung wird der gemeinsam benutzte Speicherplatz zugewiesen, dessen Größe aufgrund des Umfangs der gesendeten Daten berechnet wird. Die maximale Größe des gemeinsam benutzten Speicherplatzes beträgt 64 Kb.
Dem gemeinsam benutzten Speicher ist ein Statusanzeigebit zur Synchronisierung der Ausführung der darauf zugreifenden Tasks zugeordnet. Der darauf zugreifende Task prüft das Vorhandensein eines aktiven Statusanzeigebits, liest die Daten vom gemeinsam benutzten Speicherplatz und deaktiviert das Statusanzeigebit.
Anschließend erfolgt die Ausführung der Anweisung des Sprungs zu einem Etikett oder zu der als letztem Parameter der Anweisung TESTIPC angegebenen Funktion oder Unterprogramm.
Siehe auch SENDIPC und WAITIPC.

TESTMAIL

Syntax

TESTMAIL	**mail, [VarName1 [,..VarNameN]], GOTO Etikett**
TESTMAIL	**mail, [VarName1 [,..VarNameN]], CALL Etikett**
TESTMAIL	**mail, [VarName1 [,..VarNameN]], FunktionsName**
TESTMAIL	**mail, Matrix[Zeile], GOTO Etikett**
TESTMAIL	**mail, Matrix[Zeile], CALL UnterprogrammName**
TESTMAIL	**mail, Matrix[Zeile], FunktionsName**

Argumente

mail	Konstante oder Integer-Variable (1÷256). Nummer der Mailbox
[VarName1[,..VarNameN]]	Integer-Variable. Namen der Variablen 1÷20
Matrix[Zeile]	Konstante oder Integer-Variable. Nummer der Zeile der Matrix
Etikett	Name des Etiketts zu dem man springt
UnterprogrammName	Name des Unterprogramms
FunktionsName	Name der Funktion

Beschreibung

Test und Empfang einer Nachricht.
Die erste TESTMAIL an die Mailbox **mail** schafft die Mailbox.
Ist die Nachricht in der Mailbox **mail** vorhanden, werden die mit der Nachricht gesendeten Daten in den Variablen **VarName** (1÷20) gespeichert, sofern sie angegeben sind, oder in der von **Matrix[Zeile]** angegebenen Zeile der Matrix. Außerdem erfolgt ein Sprung zum **Etikett** oder der Aufruf von **UnterprogrammName** oder **FunktionsName**.
Während der Ausführung wird die Kongruenz zwischen den übertragenen und den von der Anweisung erwarteten Daten überprüft.
Siehe auch die Anweisungen SENDMAIL, WAITMAIL und ENDMAIL.

10.3.13 Verschiedene Anweisungen

CLEARERRORS

Syntax

CLEARERRORS **[PosID]**

Argumente

PosID Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert Verwendeten numerischen Wert

Beschreibung

Diese Anweisung sorgt dafür, dass der Supervisor-PC alle vorher mittels der Anweisung ERROR gesendeten Zyklusfehler des Moduls löscht, das die Anweisung ausführt. Der Parameter **PosID** ist ein optionaler Parameter, der den numerischen Wert angibt, der von den synoptischen Schemen verwendet wird, um Zyklusfehler in verschiedene Felder zu ordnen. Er muss dem im Bildner der synoptischen Schemen angegebenen Wert für das spezifische Anzeigefeld entsprechen. Albatros verwendet diesen Identifizierer, um Zyklusfehler in getrennten Warteschlangen zu verwalten. Pro PosID wird eine Schlange geschaffen. Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen. Wenn der Parameter **PosID** nicht angegeben ist, dann alle Zyklusfehler sowohl in der Default-Schlange als auch in den eventuellen anderen Schlangen werden gelöscht.
Siehe auch die Anweisungen ERROR und DELERROR.

CLEARMESSAGES

Syntax

CLEARMESSAGES **[PosID]**

Argumente

PosID Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert verwendeten numerischen Wert

Beschreibung

Diese Anweisung sorgt dafür, dass der Supervisor-PC alle vorher mittels der Anweisung MESSAGE gesendeten Nachrichten des Moduls löscht, das die Anweisung ausführt. Der Parameter **PosID** ist ein optionaler Parameter, der den numerischen Wert angibt, der von den synoptischen Schemen verwendet wird, um Nachrichten in verschiedene Felder zu ordnen. Er muss dem im Bildner der synoptischen Schemen angegebenen Wert für das spezifische Anzeigefeld entsprechen. Albatros verwendet diesen Identifizierer, um Nachrichten in getrennten Warteschlangen zu verwalten. Pro PosID wird eine Schlange geschaffen. Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen. Wenn der Parameter **PosID** nicht angegeben ist, dann werden alle Nachricht sowohl in der Default-Schlange als auch in den eventuellen anderen Schlangen gelöscht.
Siehe auch die Anweisungen MESSAGE und DELMESSAGE.

DEFMSG

Syntax

DEFMSG **Etikett [, SprachPräfix1], "NachrichtString" , ..., [, SprachPräfixN, "NachrichtString"]**

Argumente

Etikett	mnemonischer Name der anzuzeigenden Nachricht
SprachPräfix	vordefinierte Konstante. Sprache, in der die Nachricht geschrieben ist
NachrichtString	anzuzeigende Nachricht. Hat zwischen doppelten Anführungszeichen zu stehen ("")

Beschreibung

Diese Anweisung weist einer Nachricht ein **Etikett** zu. Die Anweisung DEFMSG ist vor der Implementierung der Funktionen zu deklarieren. Die Definition der Nachricht kann nur innerhalb der Datei (oder Gruppe) verwendet werden, in der sie deklariert ist. Durch Gebrauch der vordefinierten Konstante **SprachPräfix** können Nachrichten in verschiedenen Sprachen eingefügt werden (siehe das Kapitel "Nachrichten importieren" für die Sprachpräfixe Liste). In diesem Fall sorgt eine MESSAGE-Anweisung für die Anzeige der Nachricht in der laufend in Albatros verwendeten Sprache. Eine Nachricht, der kein Präfix zugeordnet ist, wird dann verwendet, wenn die laufend gebrauchte Sprache keinem der bestehenden Präfixe entspricht.
Die Etiketten der verschiedenen Sprachen werden entweder alle auf derselben Zeile oder auf mehreren Zeilen geschrieben, wobei die Zeile mit der Tastenkombination **[SHIFT]+[ENTER]** gewechselt wird.
Einer Funktion kann die Anweisung DEFMSG als Parameter durchgegeben werden; somit kann sie von der aufnehmenden Funktion als ein der drei Argumente von ERROR e MESSAGE gebraucht werden (siehe Beispiel 2). Siehe auch die Anweisungen MESSAGE, DELMESSAGE, ERROR, DELERROR.

Beispiel 1:

```
;Zuweisung eines Nachricht-Strings an ein Etikett ohne Auswahl
;der Sprache
DEFMSG    MSG_GRU_1   "Nachricht Gruppe 1"

;Gebrauch der Definition
MESSAGE   MSG_GRU_1   ;zeigt an: "Nachricht Gruppe 1"


;Zuweisung eines Nachricht-Strings an ein Etikett mit Auswahl
;der
  Sprac
  he
DEFMSG   MSG_GRU DEU "Nachricht Gruppe 1"
         _1
                ENG "Message group 1"

;Gebrauch der Definition, wenn die Sprache von Albatros ENG ist
MESSAGE   MSG_GRU_1   ;zeigt an: "Message group 1"
```

Beispiel 2:

```
  ;In einer Gruppe:
DEFMSG  MSG_TEST  "Ausführungsfehler"

  FUNCTION TestAufruf
         Test MSG_TEST
  FRET

  ;In einer Bibliothek:
    DEFMSG  MSG_BASE   "Fehlermeldung: $1"
    ...
  FUNCTION Test Public
    PARAM  Code AS integer
    ERROR  MSG_BASE NOPLACE NOSTORE Code
  FRET
  Der angezeigte Zyklusfehler ist: Fehlermeldung: Ausführungsfehler
  ; Der angezeigte Zyklusfehler ist: Fehlermeldung: Ausführungsfehler
```

DELAY

Syntax

DELAY	**Wert**

Argumente

Wert	Konstante oder Variable. In Sekunden ausgedrückte Verzugszeit

Beschreibung

Diese Anweisung wartet bis die in **Wert** angegebene Zeit vergeht. Bei Ablauf des Zeitintervalls wird die folgende Anweisung ausgeführt. Der kleinste einführbare Wert ist 4 ms. (0,004 Sekunden).

DELERROR

Syntax

DELERROR	**DeviceName [,Zustand [, PosID [log]]**
DELERROR	**Zahl [, PosID [log]]**

Argumente

DeviceName	Device Name
Zahl	DEFMSG oder Konstante oder Variable
PosID	Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert.
Zustand	vordefinierte Konstante. Kann folgende Werte annehmen: **ON** **OFF**
log	vordefinierte numerische Konstante oder Variable Integer. Sie kann die folgenden Werte annehmen : **STORE-**Fehler in Datei gespeichert, **NOSTORE**-Fehler nicht in Datei gespeichert.

Beschreibung

Diese Anweisung sorgt dafür, daß der Supervisor-PC einen vorher mittels der Anweisung ERROR gesendeten Zyklusfehler löscht.
Wird statt der Nummer der Name einer Vorrichtung angegeben, werden dem PC der Typ und die logische Adresse der Vorrichtung übergeben. Damit das Löschen stattfinden kann, haben alle vorgegebenen Parameter mit denen übereinzustimmen, die zur Erzeugung des Fehlers verwendet worden sind. Die Einstellung des Protokollparameters **log** auf **STORE** bewirkt, dass der Zyklusfehler in der Fehlerberichtsdatei für den aktuellen Monat aufgezeichnet wird. Ein in der Report-Datei gespeicherter Fehler wird nämlich nicht in der Datei gelöscht, sondern nur im Fehlerfenster. In der Datei kommt die neue Speicherung der Fehlerlöschung hinzu.
Der Parameter **PosID** ist ein wahlfreier Parameter. Er spezifiziert den in den Synoptikern gebrauchten Zahlwert zur Verteilung der Zyklusfehler in verschiedene Zellen.
Er muss dem vom Bildner der synoptischen Schemen angegebenen Wert zu diesem Sonderdisplay entsprechen. Sollte es nicht nötig sein, eine bestimmte Zelle hinzuweisen, muss die vorausbestimmte Konstante NOPLACE zugeteilt werden. Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen.
Wenn die Zyklusfehler als Hinweise verwaltet sind, werden alle Löschen-Anfragen gesendet. Sind die Alarme als Zustände verwaltet, wird das Löschen der Zyklusfehler gesendet, nur wenn der Zyklusfehler aktiv war. Sonst wird die Anweisung DELERROR ignoriert.
Siehe auch die Anweisungen ERROR, CLEARERRORS.

DELMESSAGE

Syntax

DELMESSAGE	**Zahl [, PosID]**

Argumente

Zahl	DEFMSG oder Konstante oder Variable
PosID	Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert

Beschreibung

Diese Anweisung sendet dem PC eine Anforderung, die vorher mittels der Anweisung MESSAGE gesendete Nachricht zu löschen.
Wenn die Messages als Hinweise verwaltet sind, werden alle dazu entsprechenden Message gelöscht. Sind die Messages als Zustände verwaltet, wird das Löschen der Message gesendet, nur wenn ihr Zustand aktiv war. Sonst wird die Anweisung DELMESSAGGE ignoriert.
Der Parameter **PosID** ist ein wahlfreier Parameter, der den in den synoptischen Schemen gebrauchten Zahlwert spezifiziert, um den Zyklusfehler in verschiedene Zelle zu verteilen.
Er muss dem vom Bildner der synoptischen Schemen angegebenen Wert zu diesem Sonderdisplay entsprechen. Sollte es nicht noetig sein, eine bestimmte Zelle hinzuweisen, muss die vorausbestimmte Konstante NOPLACE zugeteilt werden. Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen.

Siehe auch die Anweisung MESSAGE.

ERROR

Syntax

ERROR **DeviceName [,Zustand [, PosID [, log]]]**
ERROR **Zahl [, PosID [, log [, arg1, ..., arg3]]]**

Argumente

DeviceName	Device Name
Zahl	DEFMSG oder Konstante oder Variable
PosID	Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert.
Zustand	vordefinierte Konstante. Kann folgende Werte annehmen: **ON** **OFF**
log	vorbestimmte numerische Konstante oder Integer-Variable. Kann folgende Werte annehmen: **STORE** Fehler auf Datei gespeichert **NOSTORE** Fehler nicht auf Datei gespeichert
arg1, ..., arg3	Konstante oder Vorrichtung oder Variable.

Beschreibung

Diese Anweisung srzeugt einen Zyklusfehler. Der Fehler wird vom Parameter **Zahl** oder vom Namen der Vorrichtung bestimmt.
Der Parameter **Zahl** kann ein Modul-Zyklusfehler identifizieren (d.h. einen numerischen Wert) oder einen Aggregat-Fehler (in diesem Fall wird eine DEFMSG verwendet). Wird statt der Nummer der Name einer Vorrichtung angegeben, werden dem PC der Typ und die logische Adresse der Vorrichtung übergeben. Der Zyklusfehler wird an den Supervisor-PC gesendet und auf der Fehlerleiste von Albatros angezeigt.
Der Parameter **PosID** wird in synoptischen Schemen verwendet, um Zyklusfehler in verschiedene Felder zu ordnen. Er muss dem im Bildner der synoptischen Schemen angegebenen Wert für das spezifische Anzeigefeld entsprechen. Albatros verwendet diesen Identifizierer, um Zyklusfehler in getrennten Warteschlangen zu verwalten. Pro PosID wird eine Schlange geschaffen. Ist IDposiz nicht spezifiziert, oder wird die vordefinierte Konstante NOPLACE verwendet, geht der Zyklusfehler in die Default-Schlange (IDposiz=0). Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen.
Die Einstellung des Parameters **log** auf **STORE** sorgt für die Speicherung des Zyklusfehlers in der Fehlerreport-Datei des laufenden Monats. Eine hohe Zahl von Fehler zu generieren (oder zu löschen) kann das Leistungsniveau der Fernmodule beeinträchtigen. Der PC-Supervisor muss nämlich jeden gesandten Fehler (und ihre eventuelle Löschung) handeln. Demzufolge kann die Übermittlung von wichtigen Daten an die Steuerung langsamer werden Die optionalen Parameter **arg1, ..., arg3** erlauben die Definition von parameterabhängigen Fehlernachrichten. In den String zur Definition der Fehlernachricht sind Marker einzusetzen, die beim Auftreten des Fehlers durch den Wert oder den Namen der Vorrichtung oder des als Variable übertragenen Parameters ersetzt werden. Die in den String einzusetzenden Marker lauten:

- $1, ... $2 ersetzt durch den **Namen** der Vorrichtung oder der Variablen ($1 entspricht arg1, usw.)
- $(1), ..., $(3) ersetzt durch den **Wert** der Vorrichtung oder der Variablen

Die zulässigen Datentypen für die Parameter arg1, ..., arg3 lauten:

- CHAR
- INTEGER
- FLOAT
- DOUBLE (wird jedoch automatisch in FLOAT umgewandelt)
- Nachrichtennummer (oder DEFMSG-Etikett)
- Vorrichtung
- globale oder lokale Variable
- Funktionsparameter. Es ist möglich, auch das Etikett als Funktionsparameter zu benutzen, das mit einer Anweisung DEFMSG bestimmt ist

Strings, Matrizen und Vektoren können nicht als Parameter eingesetzt werden (einzelne Vektoren- oder Matrizenelemente sind jedoch zulässig). Bei lokalen Variablen kann nur der Wert, nicht aber der Name decodiert werden.
Zum Zweck der Löschung einer Nachricht durch die Anweisung DELERROR werden die Parameter arg1, ...arg3 ignoriert.

Zwei Modalitäten der Zyklusfehler werden bestimmt, die vom Maschinenhersteller festgesetzt sind:

Alarme, die als Hinweise verwaltet sind: alle Zyklusfehler werden gesendet. Albatros behält eine Schlange der letzten 100 Fehler der angegebenen Schlange und der letzten 100 Fehler der Default-Schlange..
Alarme, die als Zustände verwaltet sind: der Fehler wird als aktiv oder nicht aktiv betrachtet. Wenn der Fehler aktiv ist, wird jede weitere Eingabe desselben Zyklusfehler (durch Anweisung ERROR) ignoriert

Siehe auch die Anweisungen DELERROR, CLEARERRORS.

Beispiel 1

```
DEFMSG    ERR_TOOL    "Werkzeug nicht vorhanden"
DEFMSG    ERR_TOOL_P  "Werkzeug $(1) in Position $(2) laden"

; Tag für die Synoptischen Fenster
CONST     TOOLCHANGE = 5

; Fehler nicht auf der Fehlerleiste oder in den nicht markierten
; Synoptischen Ordnern visualisiert.
ERROR     ERR_TOOL

; Fehler in Report gespeichert und in den Synoptischen Ordnern
; visualisiert, die mit dem Code 5 markiert sind
ERROR     ERR_TOOL, TOOLCHANGE, STORE

; Fehler in Report gespeichert, aber nicht mit den Synoptischen
; Ordnern verbunden
ERROR     ERR_TOOL, NOPLACE, STORE

; Fehler mit Parametern
ERROR     ERR_TOOL_P, NOPLACE, NOSTORE, MxWerkzeuge[3].Cod, 5
```

Beispiel 2

```
       ; in einem Aggregat bestimmt
DEFMSG    MSG_ERR_CARICO    "Laden Werkzeugs nicht ausgeführt"

Funktion MostraMessaggio
    MsgTool  MSG_ERR_CARICO MxUtensili[3].Cod
fret


Function MostraErrore
  ErrTool STORE MSG_ERR_CARICO MxUtensili[3].Cod
Fret


; in einer Bibliothek bestimmt
DEFMSG    MSG_ERR_TOOL      "Fehler Werkzeugs: $1 $(2)"


Funktion MsgTool public
PARAM parameter1 as integer
PARAM parameter2 as integer

       MESSAGE MSG_ERR_TOOL NOPLACE parameter1 parameter2
fret

Function ErrTool public
PARAM log as integer
PARAM argument1 as integer
PARAM argument2 as integer

  ERROR MSG_ERR_TOOL NOPLACE log argument1 argument2
```

```
fret
```

IFDEF/ELSEDEF/ENDDEF

Syntax

```
IFDEF                   Konstante
        Anweisung
        ...
ENDDEF

IFDEF                   Konstante, Vergleichsoperator, Wert
        Anweisung
        ...
ENDDEF

IFDEF                   EXIST, GruppeName
        Anweisung
        ...
ENDDEF


 IFDEF                  LINKED, GeräteName
        Anweisung
        ...
 ENDDEF

 IFDEF                  UNLINKED, GeräteName
        Anweisung
        ...
 ENDDEF

IFDEF                   Konstante, Vergleichsoperator, Wert
        Anweisung
        ...
ELSEDEF
        Anweisung
        ...
ENDDEF
```

Argumente

Konstante	Integer, Char, Double, Zeichenfolge Konstant
VarName	Konstante des Typs Integer, char, double, String
Vergleichsoperator	die Symbole für den Vergleich sind: **<** (kleiner als) **=** (gleich) **>** (größer als) **=<** (kleiner gleich) **>=** (größer gleich) **<>** (nicht gleich)
Wert	Konstante oder Name einer Vorrichtung
GruppeName	String-Konstante oder Gruppe Name
GeräteName	Zeichenfolge oder Name eines Geräts

Beschreibung

Die Kompilierung unter Angabe von Bedingungen dient der Auswahl der Teile einer GPL-Funktionsdatei, die zu kompilieren und somit auszuführen sind. Der Kompilierer prüft, daß die als Argument der IFDEF-Anweisung geforderte Bedingung wahr ergibt. In diesem Fall wird der Code zwischen der Anweisung IFDEF und der Anweisung ENDDEF oder ELSEDEF kompiliert. Ist die Anweisung ELSEDEF vorhanden und ergibt die Bedingung nicht wahr, wird der Code zwischen der Anweisung ELSEDEF und der Anweisung ENDDEF kompiliert.
Die Bedingung zur Kompilierung kann auf drei verschiedene Arten ausgedrückt werden:

- nach der IFDEF-Anweisung wird der Name einer Konstante angegeben. In diesem Fall ergibt die Bedingung dann wahr, wenn eine globale Konstante oder Gruppenkonstante mit dem angegebenen Namen existiert.
- Nach der IFDEF-Anweisung wird ein Verhältnis zwischen zwei Operatoren und einem Operand angegeben. Der erste Operand hat eine Konstante zu sein. In diesem Fall ergibt die Bedingung wahr, wenn das Verhältnis wahr ist (z.B. MAX_TOOLS = 100).
- Nach der IFDEF-Anweisung wird das Schlüsselwort EXIST oder NOTEXIST gefolgt vom Namen einer Gruppe der Maschine oder von einem String mit dem Namen einer Gruppe der Maschine

oder mit dem Namen einer Bibliothek angegeben. In diesem Fall ergibt die Bedingung wahr, wenn in der Maschinenkonfiguration eine Gruppe mit demselben Namen existiert bzw. nicht existiert.

- Nach der IFDEF-Anweisung wird das LINKED oder UNLINKED - Schlüsselwort spezifiziert, gefolgt vom Namen eines Geräts. In diesem Fall ist die Bedingung überprüft, wenn das Gerät im Virtuell-Physisch angeschlossen (LINKED) oder nicht (UNLINKED) angeschlossen ist. Der Name des Geräts kann unter folgender Form ausgedrückt werden:
  Gruppe_Name.Untergruppe_Name.Geräte_Name, oder
  Gruppe_Name.Geräte_Name, oder Untergruppe_Name_GeräteName, oder
  Geräte_Name. Wenn in der Konfiguration das Gerät nicht existiert, wird es als nicht verbunden betrachtet.

Es können mehrere IFDEF-Anweisungen genestet werden, sofern man berücksichtigt, daß jeder IFDEF-Anweisung eine ENDDEF-Anweisung zu entsprechen hat.

Beispiel 1

```
; die Ausführung des GPL-Codes ändert sich je nachdem ob die
Fräsruppe
; in der Maschine vorhanden ist
Const FräsGruppe = "Fräse"
IFDEF Exist GruppoFresa
 Anweisung
 Anweisung
ELSEDEF
 Anweisung
 Anweisung
ENDDEF
```

Beispiel 2

```
; die Ausführung des GPL-Codes ändert sich
; je nach Modul
IFDEF _ID_MODULE = 1     ; kompiliert die Anweisungen für das
Modul 1
 Anweisung
 Anweisung
ELSEDEF       ; kompiliert die Anweisungen für die anderen
Module
 Anweisung
 Anweisung
ENDDEF

; den Code für die Version 3.2.0 von Albatros ausfüllen

IFDEF _VER_MAJOR = 3
 IFDEF _VER_MINOR = 2
       IFDEF _VER_REVISION = 0
             Anweisung
             Anweisung
       ENDDEF
 ENDDEF
ENDDEF

; den Code für die Version
; Service Pack 10 von Albatros ausfüllen
IFDEF _VER_SP  = "Service Pack 10"
 Anweisung
ENDDEF

; den Code nur ausfüllen, wenn das System
; für ein Fernmodul konfiguriert ist
IFDEF _REMOTE_MODULE  = 1  ; 1 = Fernmodul, andernfalls 0 =
Lokalmodul
```

```
  Anweisung
ENDDEF

; den Code für die Version
; 2.4 service pack 10 Albatros ausfüllen
IFDEF _VER_FULL  = $0002040AH
  Anweisung
ENDDEF
```

Beispiel 3

```
; die Ausführung des GPL-Codes ändert,
; wenn das Gerät im Virtuell-Physisch angeschlossen ist
IFDEF LINKED  out1   ; wenn Out1 angeschlossen ist, wird der
Code ausgeführt
  Anweisung
  Anweisung
  Anweisung
ENDIF
```

MESSAGE

Syntax

MESSAGE	**Zahl [, PosID [, arg1, ..., arg3]]**

Argumente

Zahl	Konstante oder Variable
PosID	Konstante oder Variable. In den synoptischen Schemen gebrauchter Zahlenwert
arg1, ..., arg3	Konstante oder Vorrichtung oder Variable.

Beschreibung

Diese Anweisung erzeugt eine Nachricht für den Bediener. Der Parameter **Zahl** kann eine Modul-Message identifizieren (d.h. eine ganzen numerischen Wert) oder eine Aggregat-Message (in diesem Fall wird eine DEFMSG verwendet). Als Option kann auch ein Argument, dargestellt von **PosID**, übergeben werden, das angibt, in welchem synoptischen Fenster die Nachricht angezeigt werden soll. Es muss dem Bildner der synopischen Schemen angegebenen Wert für das spezifische Anzeigefeld entsprechen. Albatros verwendet diesen Identifizierer, um Nachrichten in getrennten Warteschlangen zu verwalten. Pro PosID wird eine Schlange geschaffen. Ist PosID nicht angegeben, wird die Nachricht in die Default-Schlange gesetzt (PosID=0).
Der Bereich der einzurichtenden Werten ist zwischen 0 (NOPLACE) und 1023 eingeschlossen.
Die optionalen Parameter **arg1, ..., arg3** erlauben die Definition von parameterabhängigen Fehlernachrichten. In den String zur Definition der Fehlernachricht sind Marker einzusetzen, die beim Auftreten des Fehlers durch den Wert oder den Namen der Vorrichtung oder der als Parameter übertragenen Variablen ersetzt werden. Die in den String einzusetzenden Marker lauten:

- $1, ... $2 ersetzt durch den **Namen** der Vorrichtung oder der Variablen ($1 entspricht arg1, usw.)
- $(1), ..., $(3) ersetzt durch den **Wert** der Vorrichtung oder der Variablen

Die zulässigen Datentypen für die Parameter arg1, ..., arg3 lauten:

- CHAR
- INTEGER
- FLOAT
- DOUBLE (wird jedoch automatisch in FLOAT umgewandelt)
- Nachrichtennummer (oder DEFMSG-Etikett)
- Vorrichtung
- globale oder lokale Variable
- Funktionsparameter. Es ist möglich, auch das Etikett als Funktionsparameter zu benutzen, das mit einer Anweisung DEFMSG bestimmt ist.

Zwei Verwaltungsmodalitäten der Messages werden bestimmt, die vom Maschinenhersteller festgesetzt sind:
Messages, die als Hinweise verwaltet sind: alle Messages werden gesendet. Albatros behält eine Schlange der letzten 100 Nachrichten der angegebenen Schlange und der letzten 100 Nachrichten der Default-Schlange. Wenn die Nachrichtenschlange voll ist, wird die älteste Nachricht überschrieben. Ist die vorhergehende Nachricht in der Schlange identisch mit der Nachricht, die dabei ist, gesendet zu werden, wird die Nachricht nicht gesendet (gleicher Task, gleiche Nummer, gleiches Argument).

Messages, die als Zustände verwalten werden: die Message wird aktiv oder nicht aktiv betrachtet. Wenn aktiv, wird jede weitere Eingabe derselben Message (durch MESSAGE-Anweisung) ignoriert.
Strings, Matrizen und Vektoren können nicht als Parameter eingesetzt werden (einzelne Vektoren- oder Matrizenelemente sind jedoch zulässig). Bei lokalen Variablen kann nur der Wert, nicht aber der Name decodiert werden.
Zum Zweck der Löschung einer Nachricht durch die Anweisung DELMESSAGE werden die Parameter arg1, ...arg3 ignoriert.
Siehe auch die Anweisungen DELMESSAGE und CLEARMESSAGES.

Beispiel 1

```
DEFMSG    MSG_TOOL    "Werkzeug-Wechsel ausführen"
DEFMSG    MSG_TOOL_P  "Werkzeug Nummer $(1) geladen"

;Tag für die Synoptischen Fenster
CONST     TOOLCHANGE = 7

;Message auf der Fehlerleiste oder in den nicht markierten
;Synoptischen Ordnern visualisiert.
MESSAGE   MSG_TOOL

;Message  auf  der  Fehlerleiste  und  in  den  Synoptischen  Ordnern
visualisiert,
;die mit dem Code 7 markiert sind
MESSAGE   MSG_TOOL, TOOLCHANGE

; Nachricht mit Parametern
MESSAGE   MSG_TOOL_P, NOPLACE, MxWerkzeuge[3].Cod
```

Beispiel 2

```
;in einem Aggregat bestimmt
 DEFMSG      MSG_CARICO  "Laden ausgeführt"


 Function MostraMessaggio
     MsgTool  MSG_CARICO MxUtensili[3].Cod
 fret

 ; in einer Bibliothek bestimmt
 DEFMSG      MSG_TOOL    "Werkzeug: $(1) $2"


 Function MsgTool public
 PARAM parameter1 as integer
 PARAM parameter2 as integer

        MESSAGE MSG_TOOL NOPLACE parameter1 parameter2
 fret
```

SYSFAULT

Syntax

SYSFAULT

Argumente

kein Argument

Beschreibung

Diese Anweisung deaktiviert das Signal SYSOK.
Dieses Signal wird deaktiviert, um darauf hinzuweisen, daß die Maschine nicht mehr sicher ist (z.B. sind die GPL-Tasks zur Verwaltung von Notfällen nicht mehr in Ausführung begriffen).
Siehe auch die Anweisung SYSOK.

SYSOK

Syntax

SYSOK **[AusgangName1 [, … AusgangName8]]**

Argumente

AusgangName1 [...AusgangName8] Device Name des Typs Digitaler Ausgang

Beschreibung

Es wir der Numerischen Steuerung gezeigt, welche Ausgänge mit dem Sicherheitskreis der Maschinen verbunden sind (es kann sich handeln um eine mit einem Sicherheitsrelais verbundene Ausgang. Das Relais steuert die Leistungsversorgung der Maschine). Die Ausgänge werden aktiviert, nachdem die Numerische Steuerung die Initialisierung der Maschine vervollständigt hat und alle Tasks der Notfallverwaltungen aktiviert hat. Die Maschine kann somit als sicher angesehen werden. Nicht mehr als 8 Digitalausgänge insgesamt können definiert werden. Auf jedem Fernmodul kann man eine einzige Ausgang aktiviert werden. Auf CN 2004 können nur die darauf vorhandenen Ausgänge aktiviert werden und nicht auf den Fernmodulen, die damit angeschlossen sind. Die Reihe der bei dem ersten Gebrauch erklärten Ausgänge der SYSOK-Anweisung darf in den eventuell nachfolgenden Sysok- Aufrufen nicht geändert werden, ohne die Steuerung initialisiert zu haben. Wird die Anweisung ohne Parameter ausgeführt, dann wird das Signal von SYSOK wiederhergestellt.

Siehe auch die Anweisung SYSFAULT.

Anmerkung:

Die SYSOK Anweisung kann nur

- auf allen v3.0 GreenBUS Fernmodulen mit digitalen Ausgängen,
- auf TRS-IO v.4.0 GreenBUS Fernmodulen,
- auf TRS-CAT Fernmodulen, nur auf der Basis und nicht auf den Erweiterungen, die eine 1.2 oder höhere Firmware-Version (Rev. 1.00 des Fernmoduls) haben, aktiviert werden.

TYPEOF

Syntax

TYPEOF **Name, Ergebnis**

Argumente

Name Name einer Vorrichtung, Konstante, FunktionsName, Variable, Vektor, Matrix oder Matrix-Zeile

Ergebnis Integer-Variable. Typ des ersten Arguments

Beschreibung

Diese Anweisung gibt in die Variable **Ergebnis** den Typ des Arguments **Name** zurück.

WATCHDOG

Syntax

WATCHDOG **Zustand**

Argumente

Zustand vorbestimmte Konstante. Kann folgende Werte annehmen: **ON und OFF.**

Beschreibung

Die Anweisung aktiviert die Verwendung des mit dem TMSWD angeschlossenen Watchdogs. Sie ermöglicht Fehlersituationen zu identifizieren, die bei der Ausführung des GPL-Codes auftreten.
Das erste Mal dass diese Anweisung durch den Parameter **Zustand** bei ON ausgeführt wird, dann ermöglicht die Anweisung den Gebrauch des Watchdogs. Alle nachfolgenden Zeiten muss man ON dem **Zustandparameter** zuordnen, um den Kartenzähler zu aktualisieren.
Wird die Aktualisierung nicht ausgeführt, dann tritt der Watchdog in Aktion und das TMSWD Modul deaktiviert die Notausgang der Maschine.
Um die Verwendung des Watchdogs zu beenden, muss man dem **Zustand** Parameter OFF zuweisen.

Diese Anweisung kann nur mit TMSbus+, TMSCan+ und TMSCombo+ Platinen mit FPGA Version 2.0 (oder höher) und mit montiertem TMSWDHardware-Modul verwendet werden.

Beispiel:

```
Funktion TestWatchDog autorun

   Watchdog ON ; aktiviert die Steuerung des Watchdogs

loop:

   Watchdog ON ; aktualisiert den Zähler der Platine

   goto loop

fret
```

10.3.14 MECHATROLINK-II

MECCOMMAND

Syntax

MECCOMMAND **Achse, Register,Befehl, Parameter, Antwort, Fehler**

Argumente

Achse	Name einer Vorrichtung des Typs digitale Achse
Befehl	Konstante Integer.
Parameter	Integer Vektor
Antwort	Integer Vektor
Fehler	Variable integer. Fehlercode

Beschreibung

Diese Anweisung sendet der hingewiesenen **Achsen**ausführung einen **Befehl** und wartet auf dessen Antwort. Die zur Befehlsausführung nötigen Daten werden in den Vektor **Parameter** eingesetzt, während die aus der Anweisungsausführung zurückgegebenen Daten im Vektor **Antwort** gespeichert werden. Die Vektoren **Parameter** und **Antwort** müssen die selbe Dimension und Höchstzahlelemente 14 enthalten. Der angesehene Wert ist das niedrigste Byte unter der einzelnen Ganzzahlen. Der Parameter **Fehler** enthält die Codes der bei des Vorgangs möglich generierten Fehler.
Die Fehler-Code müssen vom GPL als Zyklusfehler verwaltet werden.
Die ergebenen Zuklus**fehler** sind die folgenden:

Fehlercode	**Meldung**
-40	Befehl bei dem gängigen Betriebszustand nicht erstattet.
-41	Timeout-Fehler bei der Ausführung eines MECHATROLINK-II-Befehls
-44	Timeout-Fehler bei der Ausführung eines MECHATROLINK-II-Unterbefehls
-45	Fehler bei der Antriebsschaltung

Für die den Parametern **Befehl**, **Parameter**, **Antwort** und **Fehler** zuzuordnenden Werte nehme man Bezug auf die offizielle Yaskawa MECHATROLINK-II-Dokumentation, in der die zu dem Befehl zuzuordnenden Werte vom Index 2 bis zum Index 15 beschrieben sind. Die für die Unterbefehlen einzustellenden Werte sind vom Index 18 bis zum Index 32 zu beschrieben.
Im Vergleich zu der offiziellen Dokumentation können die **Befehl**swerte ändern.
Die Befehle können sich unterscheiden, wie folgt:

- Befehl. Sie haben einen Code zwischen 0x00 und 0xFF. Aus Sicherheitsgründen werden sie nur mit deaktivierter Servo-Achse ausgeführt.
- Unterbefehl. Die als Unterbefehle verstandenen Befehle müssen zu dem dokumentierten Wert den Code 0x100 addieren. Zum Beispiel, hat der Befehl NOP einen dokumentierten Code 0x00, ist, als verstandene Unterbefehl, 0x100.
- Prozedur. Die als Prozeduren verstandenen Befehle haben einen Code mit Befehl ab 0x200. Derzeitig sind folgende Prozeduren vorgesehen:
    - $201H Aktivierungsprozedur Offline-Parameter (bei deaktivierter Achse zu verwenden)

Diese Anweisung ist nur mit Platinen AlbMech, DualMech und DualMech Mono zu verwenden. Für weitere Informationen hinsichtlich dieser Anweisung TPA ansprechen.

Bemerkung

Diese Anweisung wirkt auf dem Verhalten der digitalen Achsen, dann ist sie in einem gesteuerten Kontext zu gebrauchen.

MECGETPARAM

Syntax

MECGETPARAM **Achse, Parameter,Dimension, Daten, Fehler**

Argumente

Achse	Name einer Vorrichtung des Typs digitale Achse
Parameter	Konstante oder Variable Integer
Dimension	Konstante oder Variable Integer
Daten	Variable Integer
Feher	Variable Integer. Fehlercode

Beschreibung

Diese Anweisung liest einen Parameter des Achsenantiriebs, der in der Variablen **Daten** gespeichert wird. Im Parameter **Fehler** sind die Codes der während des Verweisens möglich generierten Fehlern enthalten. Die Fehlercodes sind vom GPL als Zyklusfehler zu verwalten..
Die ergebenen **Ziklusfehler** sind die folgenden

Fehlercode	**Meldung**
-40	Befehl bei dem gängigen Betriebszustand nicht erstattet.
-41	Timeout-Fehler bei der Ausführung eines MECHATROLINK-II-Befehls
-44	Timeout-Fehler bei der Ausführung eines MECHATROLINK--II-Unterbefehls
-45	Fehler bei der Antriebsschaltung

Für die den Variablen **Parameter**, und **Dimension** zuzuordnenden Werte nehme man Bezug auf die offizielle Yaskawa MECHATROLINK-II-Dokumentation.
Diese Anweisung ist nur mit Platinen AlbMech, DualMech und DualMech Mono zu verwenden. Für weitere Informationen hinsichtlich dieser Anweisung TPA ansprechen.

MECGETSTATUS

Syntax

MECGETSTATUS **Achse, Status, Inout, Fehler**

Argumente

Achse	Name einer Vorrichtung des Typs digitale Achse
Zustand	Konstante oder Variable Integer
Inout	Konstante oder Variable Integer
Fehler	Variable Integer. Fehlercode

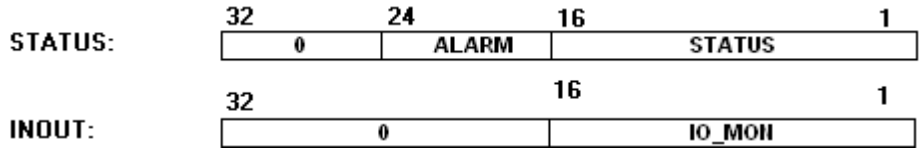
Beschreibung

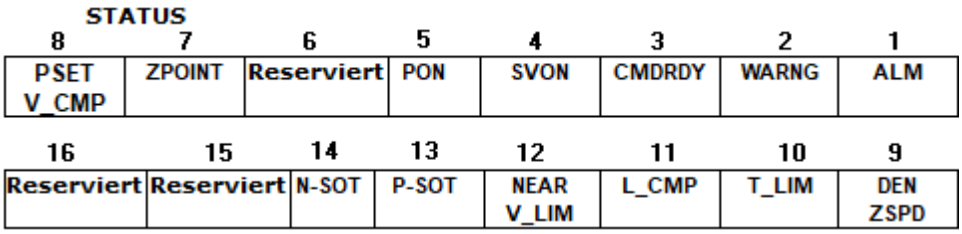
Diese Anweisung liest und speichert in der Variablen **Status** den Wert von STATUS und von ALARM und in der Variablen **inout** den Wert von IO_MON, die die angegebene **Achse** betreffen. Für die Werte von STATUS, ALARM, IO-MON nehme man Bezug auf die offizielle Yaskawa MECHATROLINK-II-Dokumentation.
Im Parameter **Fehler** sind die Codes der während des Verweisens möglich generierten Fehlern enthalten. Die Fehlercodes sind vom GPL als Zyklusfehler zu verwalten.
Die ergebenen Zuklus**fehler** sind die folgenden:

Fehlercode	**Meldung**
-40	Befehl bei dem gängigen Betriebszustand nicht erstattet.
-41	Timeout-Fehler bei der Ausführung eines MECHATROLINK-II-Befehls
-44	Timeout-Fehler bei der Ausführung eines MECHATROLINK-II-Unterbefehls
-45	Fehler bei der Antriebsschaltung

Es wird eine Fehlerkategorie vorbestimmt. Die Kategorien stellen den von ALARM höheren Nibble-Wert dar.
Um Alarme, die in einer der folgenden Kategorien 0x30,0x70,0xD0,0xF0 eingeschlossen sind, zu löschen muss ein CLEAR(0x06)-Befehl gesandt werden. Alarme, die in einer der folgenden Kategorien 0x00,0x10,0x40,0xB0 eingeschlossen sind, können durch keinem Befehl entfernt werden.
Es ist nötig, das das Alarm verursachende Problem zu löschen und den Servo-Antrieb aus- und einzuschalten.

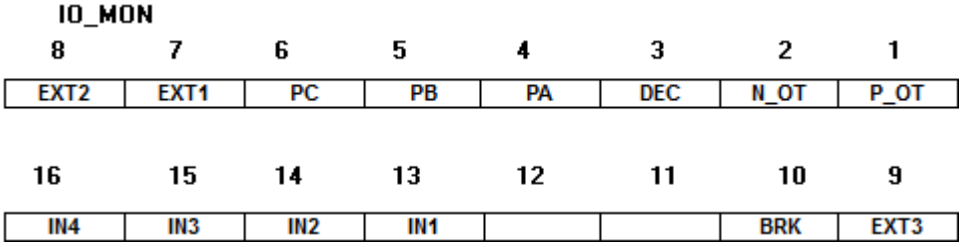
Die Struktur der Variablen **Zustand** und **Inout** ist eine strukturierte bit-Maske, wie unten dargestellt:





Was die Status-Bits bedeuten

Bit	Befehl	Physikalische Pins, die im virtuellen-physikalischen Bereich verbunden werden können
1	ALM (Alarm)	Digitaler Eingang
2	WARNG (Warning)	Digitaler Eingang
3	CMDRDY (Command Ready)	
4	SVON (Servo ON)	Digitaler Ausgang
5	PON (Main Power ON)	Digitaler Eingang
6	Reserviert	
7	ZPOINT (Zero Point)	
8	PSET (Position Complete) V_CMP (Velocity Agreement)	
9	DEN (Command Distribution Completed Flag) ZSPD (Zero Velocity)	
10	T_LIM (Torque Limit)	
11	L_CMP (Latch Completed)	
12	NEAR (Position Proximity) V_LIM (Velocity Limit)	
13	P-SOT (Forward-direction Software Limit)	
14	N-SOT (Reverse-direction Software Limit)	
15	Reserviert	
16	Reserviert	



Was die IO_MON-Bits bedeuten

Bit	Befehl	Physikalische Pins, die im virtuellen-physikalischen Bereich verbunden werden können
1	P_OT (Forward Over Travel)	
2	N_OT (Reverse Over Travel)	
3	DEC (Deceleration Limit Switch)	
4	PA (Phase A)	
5	PB (Phase B)	
6	PC (Phase C)	Digitaler Eingang
7	EXT1 (First external latch input)	Digitaler Eingang
8	EXT2 (Second external latch input)	Digitaler Eingang
9	EXT3 (Third external latch input)	
10	BRK (Brake output)	
11		
12		
13	IN1 (General-purpose input 1)	
14	IN2 (General-purpose input 2)	
15	IN3 (General-purpose input 3)	
16	IN4 (General-purpose input 4)	

Diese Anweisung ist nur mit Platinen AlbMech, DualMech und DualMech Mono zu verwenden. Für weitere Informationen hinsichtlich dieser Anweisung TPA ansprechen.

MECSETPARAM

Syntax

MECSETPARAM **Achse, Parameter, Dimension, Daten, Fehler**

Argumente

Achse Name einer Vorrichtung des Typs digitale Achse
Parameter Konstante oder Variable Integer
Dimension Konstante oder Variable Integer
Daten Variable Integer.
Fehler Variabile Integer. Fehlercode

Beschreibung

Diese Anweisung schreibt **Daten** im **Parameter** der angegebenen **Achse.**
Für die den Variablen **Parameter** und **Dimension** zuzuordnenden Werte nehme man Bezug auf die offizielle Yaskawa MECHATROLINK-II-Dokumentation. Im Parameter **Fehler** sind die Codes der während des Verweisens möglich generierten Fehlern enthalten. Die Fehlercodes sind vom GPL als Zyklusfehler zu verwalten. Die Fehlercodes sind vom GPL als Zyklusfehler zu verwalten.
Die ergebenen Zuklus**fehler** sind die folgenden:

Fehlercode	Meldung
-40	Befehl bei dem gängigen Betriebszustand nicht erstattet
-41	Timeout-Fehler bei der Ausführung eines MECHATROLINK- II-Befehls
-44	Timeout-Fehler bei der Ausführung eines MECHATROLINK- II-Unterbefehls
-45	Fehler bei der Antriebsschaltung

Diese Anweisung ist nur mit Platinen AlbMech, DualMech und DualMech Mono zu verwenden. Für weitere Informationen hinsichtlich dieser Anweisung TPA ansprechen.

Bemerkung

Diese Anweisung wirkt auf dem Verhalten der digitalen Achsen, dann ist sie in einem gesteuerten Kontext zu gebrauchen.
Zum Schreiben von Daten in dem nichtflüchtigen Speicher ist die Anweisung MECCOMMAND zu verwenden.

10.3.15 Standardmäßige Feldbusse

AXCONTROL

Syntax

AXCONTROL **Achse, Daten**

Argumente

Achse Name einer Vorrichtung des Typs Achse
Daten Integer-Variable oder Konstante, die ControlWord einstellt.

Beschreibung

Diese Anweisung stellt die **Daten** von ControlWord gemäß der aktiven Betriebsfähigkeit, "CiA 402 CANopen device profile" entsprechend.

Tabelle zur Bestimmung der Werte für EtherCAT:

Bit	Bedeutung	Name in Virtuell-Physisch
1	so=Switch ON	CW1
2	ev=Enable voltage	EV
3	qs=Quick stop	STOP
4	eo=Enable Operation	SVON
5	oms=Operation mode specific	CW5
6	oms=Operation mode specific	CW6
7	oms=Operation mode specific	CW7
8	fr=Fault reset	RESALM
9	h=Halt	CW9
10	oms=Operation mode specific	CW10
11	r=Reserved	CW11
12	ms=Manufacturer specific	CW12
13	ms=Manufacturer specific	CW13
14	ms=Manufacturer specific	CW14
15	ms=Manufacturer specific	CW15
16	ms=Manufacturer specific	CW16

Tabelle zur Bestimmung der Werte für S-CAN

Bit	Bedeutung	Name in Virtuell-Physisch
1	Ten_cmd=Befehl zur Aktivierung des Drehmoments. 1:Achse Drehmoment 0:freie Achse	SVON

2	Ten_cmd=Befehl zur Aktivierung der Bewegungen 1:Bewegungen aktiviert 0:Achsen im Stillstand	ENMOVE
3	Stp_cmd=Stop Befehl 1: Stop Befehl aktiv 0: Stop Befehl nicht aktiv	STOP
4	Alm_rst=Alarmzustand 1:Befehl zum Alarm-	RESALM
5	Ltc_rst:Zurücksetzung bit 5 von StatusWord	CW5
6	oms=Spezifisch vom gewählten Modus	CW6
7	oms=Spezifisch vom gewählten Modus	CW7
8	oms=Spezifisch vom gewählten Modus	CW8

AXSTATUS

Syntax

AXSTATUS **Achse, Wert**

Argumente

Achse Name einer Vorrichtung des Typs Achse
Wert Integer-Variable.

Beschreibung

Diese Anweisung ergibt den in StatusWort enthaltenen Wert, "CiA 402 CANopen device profile" entsprechend.

Tabelle zur Bestimmung der Werte für EtherCAT:

Bit	Bedeutung	Name im Virtuell-Physisch
1	rtso=Ready to switch on	RTSO
2	so=Switched on	SW2
3	oe=Operation enabled	OE
4	f=Fault	ALM
5	ve=Voltage enable	VE
6	qs=Quick stop	QS
7	sod=Switch on disabled	SOD
8	w=Warning	WARNG
9	ms=Manufacturer specific	SW9
10	rm=Remote	SW10
11	tr=Target reached or reserved	SW11
12	ila=Internal limit active	SW12
13	oms=Operation mode specific	SW13
14	oms=Operation mode specific	SW14
15	ms=Manufacturer specific	SW15
16	ms=Manufacturer specific	SW16

Tabelle zur Bestimmung der Werte für S-CAN

Bit	Bedeutung	Name im Virtuell-Physisch
1	Ten_st=Zustand des Drehmoments 1:Achse Drehmoment 0:freie Achse	SW1
2	Ien_st=Zustand der Bewegungsaktivierung 1:Bewegungen aktiviert 0:Achse im Stillstand	SW2
3	Stp_st=Halt Zustand 1: laufende Stop-Rampe 0: Stop nicht aktiviert oder Rampe beendet	SW3
4	Alm_st=Alarmzustand 1:Maschien im Alarmzustand 0:Kein Alarmzustand gefunden	ALM
5	Ltc_st=Zustand der Latch-Position 1:Latch-Position ausgeführt, Register bereit zum Lesen 0:Keine Latch-Position gefunden	SW5
6	oms=Spezifisch vom gewählten Modus	SW6
7	oms=Spezifisch vom gewählten Modus	SW7
8	oms=Spezifisch vom gewählten Modus	SW8

CNBYDEVICE

Syntax

CNBYDEVICE **Vorrichtung, Platine,cn**

Argumente

Vorrichtung Vorrichtungsname.
Platine Integer-Variable. Ergebene Platinennummer
cn Integer-Variable. Ergebene Knotenzahl

Beschreibung

Diese Anweisung ergibt Knotenzahl und die Zahl der Platine der im Parameter **Vorrichtung** festgelegten Vorrichtung. Diese Instruktion ist bei dem Gebrauch der Instruktionen angestellt, die über keine direkte Verbindung mit Vorrichtungen verfügen, wie, zum Beispiel, die Instruktionen READDICTIONARY und WRITEDICTIONARY.
Die Anweisung kann für an CAN-, S-CAN-Busse, Greenbusse oder EtherCAT konfigurierte Geräte verwendet werden.
Wenn die zurückgegebene Knotennummer ein negativer Wert ist, bedeutet es, dass der Knoten deaktiviert ist.

READDICTIONARY

Syntax

READDICTIONARY **Platine,Cn, Index, Unterindex, Dimdaten, Daten, Fehler**

Argumente

Platine Konstante oder Integer-Variable. Platinenzahl
Cn Konstante oder Integer-Variable. Knotenzahl
Index Konstante oder Integer-Variable. Index des Objekts im Diktionär
Unterindex Konstante oder Integer-Variable. Unterindex im Diktionär
Dimdaten Integer-Variable. Größe der gelesenen Daten
Daten Char-Variable, Integer, Float, Double, Char-Vektor, String. Variable, die die Daten erhält
Fehler Integer-Variable. Vom Knoten ergebener Fehlercode

Beschreibung

Diese Anweisung liest den Inhalt eines Objekts vom Diktionär der im Knoten enthaltenen Objekte. Die Instruktion ermöglicht, durch das SDO-Protokoll alle gem. "CiA 402 CANopen device profile"

festgesetzte Objekte, neben den vom Hersteller des Knotens zur Verfügung gestellten Objekten, zu lesen. Für die Bedeutung der Parameter **Index**, **Unterindex** und **Dimdaten** sich an "CiA 402 CANopen device profile" oder an die Spezifikation des Herstellers des Knotens anwenden.
Für die S-CAN Geräte ist der Parameter Unterindex immer auf Null festzulegen.

WRITEDICTIONARY

Syntax

WRITEDICTIONARY **Platine,Cn, Index, Unterindex, Dimdaten,Daten, Fehler**

Argumente

Platine	Konstante oder Integer-Variable. Platinenzahl
Cn	Konstante oder Integer-Variable. Knotenzahl
Index	Konstante oder Integer-Variable. Index des Objekts im Diktionär
Unterindex	Konstante oder Integer-Variable. Unterindex im Diktionär
Dimdaten	Konstante oder Integer-Variable. Dimension der zu schreibenden Daten gelesenen
Daten	Char-Variable, Integer, Float, Double, Char-Vektor, String. Variable, die die Daten erhält
Fehler	Integer-Variable. Vom Knoten ergebener Fehlercode

Beschreibung

Diese Anweisung schreibt den Inhalt eines Objekts vom Diktionär der im Knoten enthaltenen Objekte. Die Instruktion ermöglicht, durch das SDO-Protokoll alle gem. "CiA 402 CANopen device profile" festgesetzte Objekte, neben den vom Hersteller des Knotens zur Verfügung gestellten Objekten, zu lesen. Für die Bedeutung der zu schreibenden Parameter **Index**, **Unterindex** und **Dimdaten** sich an "CiA 402 CANopen device profile" oder an die Spezifikation des Herstellers des Knotens anwenden. Für die S-CAN Geräte ist der Parameter Unterindex immer auf Null festzulegen.

10.3.16 EtherCAT

ACTIVATEMODE

Syntax

ACTIVATEMODE **Achse, Daten, Fehler**

Argumente

Achse	Name einer Vorrichtung des Typs Achse
Daten	Integer-Variable oder Konstante. Betriebsmodus
Fehler	Integer-Variable. Vom Servokontroller ergebener Fehlercode

Beschreibung

Diese Anweisung stellt den Betriebsmodus ein, der in der Variablen **Daten,** CiA 402 CANopen device profile entsprechend, festgelegt ist. Der Betriebmodus der startenden Achse entspricht dem **Datenwert** = 9, d.h. "Einstellung der Synchrongeschwindigkeit". Die Instruktion ergibt den Wert **err=** 0, wenn der Befehl Erfolg gehabt hat, ansonsten ergibt sie einen Fehlercode.
Nachfolgend, die Tabelle der den **Daten** zuzuordnenden Werte, nach dem gewählten Betriebsmodus.

Wert	Definition
+6	Homing modus
+9	Synchronischer zyklischer Geschwindigkeitmodus

ECATGETREGISTER

Syntax

ECATGETREGISTER **Knoten,Adresse,Dim,Daten,Fehler**

Argumente

Knoten	Konstante oder Variable Integer. In der EtherCAT-Kette (ab 1 an) vom Slave belegte Position
Adresse	Konstante oder Variable Integer. Adresse des ESC-Registers (Ether
Dim	Konstante oder Variable Integer. Anzahl der zu lesenden Bytes (1, 2 oder 4)

Daten	Variable Integer. Behälter der gelesenen Daten
Fehler	Variable Integer. Fehlercode

Beschreibung

Es gibt **[Daten]** den Inhalt eines ESC-Registers (EtherCAT (EtherCAT Slave Controller) des angegebenen EtherCAT-Knotens zurück. Der **Fehlerparameter** enthält den numerischen Code des Fehlers, 0 wenn keine Fehler aufgetreten sind.

ECATSETREGISTER

Syntax

ECATSETREGISTER **Knoten,Adresse,Dim,Daten,Fehler**

Argumente

Knoten	Konstante oder Variable Integer. In der EtherCAT-Kette (ab 1 an) vom Slave belegte Position
Adresse	Konstante oder Variable Integer. Adresse des ESC-Registers (EtherCAT Slave Controller) zum Schreiben (ab 0 an)
Dim	Konstante oder Variable Integer. Anzahl der zu schreibenden Bytes (1, 2 oder 4)
Daten	Konstante oder Variable Integer. Daten zu schreiben
Fehler	Variable Integer. Fehlercode

Beschreibung

Die Anweisung schreibt den Inhalt **[Daten]** eines ESC-Registers (EtherCAT (EtherCAT Slave Controller) des angegebenen EtherCAT-Knotens. Der **Fehlerparameter** enthält den numerischen Code des Fehlers, 0 wenn keine Fehler aufgetreten sind.

GETPDO

Syntax

GETPDO **Platine,Knoten,nPDO,nObj,Daten,[Fehler]**

Argumente

Platine	Konstante oder Variable Integer. Platinennummer
Knoten	Konstante oder Variable Integer. Position der Sklave in der EtherCAT - Kette (von 1 an)
nPDO	Konstante oder Variable Integer. Identifizierer von PDO (Beispiel $1600h) oder seiner Position in der Liste der in der Hardware-Konfiguration definierten PDOs für den in Rede stehenden Knoten (von 1 bis 16).
nObj	Konstante oder Variable Integer. Identifizierer des Objektes (Beispiel. $6040h) oder seine Position innerhalb der Liste der im PDO (von 1 bis 32) konfigurierten Objekte
Daten	Variable Integer Empfängt den Wert
Fehler	Variable Integer Fehlercode

Beschreibung

Es ergibt in **[Daten]** den Inhalt eines Objekts, das durch die für den EtherCAT - Knoten konfigurierten PDOs ausgetauscht wurde. Sind die übermittelten Argumente falsch und wurde der **Fehlerparameter** nicht festgelegt, dann ereignet sich ein Systemfehler. Wurde ein **Fehlerparameter** festgelegt, dann wird er den numerischen Code des entsprechenden Systemfehler enthalten.

SETEOE

Syntax

SETEOE **Zustand,[Fehler]**

Argumente

Zustand	Vorbestimmte Konstante. Die zulässigen Werte sind die folgenden: - ON aktiviert die Behandlung des Protokolls EoE; - OFF deaktiviert die Verwaltung des Protokolls EoE.
Fehler	Variable Integer Fehlercode

Beschreibung

Diese Anweisung erlaubt, die Verwaltung des Protokolls (Ehernet over EtherCAT) zu aktivieren und deaktivieren. Diese Funktion verwendet die SnifferCE oder SnifferRTX EoE - Hilfssoftware, die die Kommunikation zwischen der numerischen Steuerung und Pc Supervisor verwaltet. Die Fehlervariable kann die folgenden Werte annehmen:
0, wenn Albatros korrekt geschlossen worden ist
-1, wenn der Befehl innerhalb der maximalen Zeit von 5 Sekunden nicht abgeschlossen worden ist
-2 wenn ein Kommunikationsfehler aufgetreten ist.

SETPDO

Syntax

SETPDO	**Platine,Knoten,nPDO,nObj,Daten,[Fehler]**

Argumente

Platine	Konstante oder Variable Integer. Platinennummer
Knoten	Konstante oder Variable Integer. Position der Sklave in der EtherCAT - Kette (von 1 an)
nPDO	Konstante oder Variable Integer. Identifizierer von PDO (Beispiel $1600h) oder seiner Position in der Liste der in der Hardware-Konfiguration definierten PDOs für den in Rede stehenden Knoten (von 1 bis 16).
nObj	Konstante oder Variable Integer. Identifizierer des Objektes (Beispiel. $6040h) oder seine Position innerhalb der Liste der im PDO (von 1 bis 32) konfigurierten Objekte
Daten	Variable Integer festgelegter Wert
Fehler	Variable Integer Fehlercode

Beschreibung

Es legt den Inhalt **[Daten]** eines Objekts fest, das durch die für den EtherCAT - Knoten konfigurierten PDOs ausgetauscht wurde. Sind die übermittelten Argumente falsch und wurde der **Fehlerparameter** nicht festgelegt, dann ereignet sich ein Systemfehler. Wurde ein **Fehlerparameter** festgelegt, dann wird er den numerischen Code des entsprechenden Systemfehler enthalten.

10.3.17 TMBUS Platinen mit CAN-Stererung

GETCNSTATE

Syntax

GETCNSTATE	**Platine, Knoten, Zustand**

Argumente

Platine	Konstante oder Variable Integer Zahl der Platine
Knoten	Konstante oder Variable Integer Knotennummer
Zustand	Konstante oder Variable Integer

Beschreibung

Diese Anweisung ergibt den Zustand des NMT-Protokolls für den **Knoten** der angegebenen **Platine**. Weitere Informationen zur Bedeutung dieser Parameter können direkt der Dokumentation der jeweiligen Vorrichtung entnommen werden.

GETSDOERROR

Syntax

GETSDOSERROR	**Platine, Fehler**

Argumente

Platine	Konstante oder Variable Integer Zahl der Platine (von 1 bis 4)
Fehler	Variable Integer Fehlercode

Beschreibung

Diese Anweisung ergibt den letzten aufgetretenen **Fehler**, hinsichtlich der SDO-Kommunikation für die angegebene **Platine**. Weitere Informationen zur Bedeutung dieses ergebenen Fehlercodes können direkt der Dokumentation der jeweiligen Vorrichtung entnommen werden.

GETMNSTATE

Syntax

GETMNSTATE **Platine, Zustand**

Argumente

Platine	Konstante oder Variable Integer Zahl der Platine (von 1 bis 4)
Zustand	Konstante oder Variable Integer

Beschreibung

Diese Anweisung ergibt den Zustand des NMT-Protokolls für den Master-Knoten der angegebenen **Platine**. Weitere Informationen zur Bedeutung dieser Parameter können direkt der Dokumentation der jeweiligen Vorrichtung entnommen werden.

RECEIVEPDO

Syntax

RECEIVEPDO **Platine, Knoten, PDOZahl**

Argumente

Platine	Konstante oder Variable Integer Zahl der Platine (von 1 bis 4)
Knoten	Konstante oder Variable Integer Knotenzahl
PDOZahl	Konstante oder Variable Integer PDOZahl

Beschreibung

Diese Anweisung liest den Inhalt des von der **PDOZahl** spezifizierten PDOs für den bezeichneten Knoten. Diese Anweisung wird zum Lesen von asynchronen PDOs verwendet (d.h. die PDOs, deren die **Asynchron**-Option in der Hardware-Konfiguration aktiviert ist).
Die gelesenen Daten werden in den virtuellen-physikalisch verbundenen Geräten kopiert.
Diese Anweisung kann nur mit Platinen TMSCan und TMSCan+ verwendet werden.

SENDPDO

Syntax

SENDPDO **Platine, Knoten, PDOZahl**

Argumente

Platine	Konstante oder Variable Integer Platinennummer
Knoten	Konstante oder Variable Integer Knotenzahl
PDOZahl	Konstante oder Variable Integer PDOZahl

Beschreibung

Diese Anweisung schreibt den Inhalt des von der **PDOZahl** spezifizierten PDOs für den bestimmten Knoten. Diese Anweisung wird zum Lesen von asynchronen PDOs verwendet (d.h. die PDOs, deren die **Asynchron**-Option in der Hardware-Konfiguration aktiviert ist).
Die gelesenen Daten werden in den virtuellen-physikalisch verbundenen Geräten kopiert.
Diese Anweisung kann nur mit Platinen TMSCan und TMSCan+ verwendet werden.

SETNMSTATE

Syntax

SETNMNSTATE **Platine, Knoten, Zustand**

Argumente

Platine	Konstante oder Variable Integer Zahl der Platine (von 1 bis 4)
Knoten	Konstante oder Variable Integer Knotennummer
Zustand	Konstante oder Variable Integer

Beschreibung

Diese Anweisung stellt den Zustand des NMT-Protokolls für den **Knoten** der angegebenen **Platine**.

Ist der Wert des Knotens gleich 0 (Null) oder größer als 126, dann wird die Einstellung auf jeden konfigurierten Knoten angewandt, der im Kanal anwesend ist. Weitere Informationen zur Bedeutung dieser Parameter können direkt der Dokumentation der jeweiligen CANopen-Vorrichtung entnommen werden.

Wert	Protokollzustand
1	Operational
128	Pre-Operational

10.3.18 Simulation

DISABLE

Syntax

DISABLE **Achse1,[...Achse6]**

Argumente

Achse1...[...Achse6] Namen von Vorrichtungen des Typs Achse

Beschreibung

Diese Anweisung deaktiviert die angegebenen Achsen. Dank der Deaktivierung können Simulationen der Maschinenzyklen ausgeführt werden, bei denen die Achsen nicht wirklich bewegt werden. Eine deaktivierte Achse liest die vom Encoder kommenden Informationen nicht, simuliert aber einen zur laufenden Geschwindigkeit proportionalen Loop-Fehler. Die Deaktivierung der Achse hat allerdings keine Wirkung auf den Geschwindigkeitsbezug, d.h. auch während der simulierten Bewegungen ist am Achsenanschluß Spannung vorhanden. Daher ist es notwendig, bei simulierten Bewegungen bzw. bei deaktivierten Achsen die Antriebe von der Stromzufuhr oder von der Achsenplatine zu trennen. Siehe auch ENABLE.

Anmerkung

In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

DISABLEFORCEDINPUT

Syntax

DISABLEFORCEDINPUT

Argumente

kein Argument

Beschreibung

Diese Anweisung deaktiviert die Funktionen zur Modifizierung der Eingänge. Waren Eingänge bereits modifiziert worden, bringt diese Anweisung sie wieder in den tatsächlichen Zustand. Siehe auch ENABLEFORCEDINPUT, DISABLEFORCEDINPUT, SETFORCEDINPUT, RESETFORCEDINPUT, SETFORCEDPORT, SETFORCEDANALOG.

ENABLE

Syntax

ENABLE **Achse1,[...Achse6]**

Argumente

Achse1...[...Achse6] Namen von Vorrichtungen des Typs Achse

Beschreibung

Diese Anweisung aktiviert die angegebenen Achsen. Die Achsen werden bei der Initialisierung immer aktiviert. Diese Anweisung ist nur aufzurufen, wenn die Achsen vorher durch eine DISABLE-Anweisung deaktiviert worden sind.

Anmerkung

In dieser Anweisung können Achsen im Schrittbetrieb verwendet werden nur wenn sie von einem Ferngerät TRS-AX verwaltet sind.

ENABLEFORCEDINPUT

Syntax

ENABLEFORCEDINPUT

Argumente

kein Argument

Beschreibung

Diese Anweisung aktiviert die Modifizierung der Eingänge. Vor Einsatz der Anweisungen zur modifizierten Aktivierung oder Deaktivierung der Eingänge ist diese Anweisung auszuführen. Andernfalls bleiben die Anweisungen zur Modifizierung der Eingänge wirkungslos.
Siehe auch DISABLEFORCEDINPUT, SETFORCEDINPUT,RESETFORCEDINPUT, SETFORCEDPORT, SETFORCEDANALOG.

RESETFORCEDINPUT

Syntax

RESETFORCEDINPUT **EingangName**

Argumente

EingangName Digitale Eingang Name

Beschreibung

Diese Anweisung modifiziert den von **EingangName** angegebenen Eingang auf OFF.
Um diese Anweisung verwenden zu können, muß man die Modifizierung der Eingänge mittels der Anweisung ENABLEFORCEEINPUT bereits aktiviert haben.
Siehe auch DISABLEFORCEDINPUT, SETFORCEDINPUT, SETFORCEDPORT, SETFORCEDANALOG.

SETFORCEDANALOG

Syntax

SETFORCEDANALOG **AnalogEingang, Wert**

Argumente

AnalogEingang Name der Vorrichtung analoger Eingang
Variable Integer- oder Float- oder Double-Konstante oder -Variable

Beschreibung

Diese Anweisung modifiziert den **Wert** des von **AnalogEingang** angegebenen analogen Eingangs.
Um diese Anweisung verwenden zu können, muß man die Modifizierung der Eingänge mittels der Anweisung ENABLEFORCEEINPUT bereits aktiviert haben.
Siehe auch DISABLEFORCEDINPUT, SETFORCEDINPUT,RESETFORCEDEINPUT, SETFORCEDPORT.

SETFORCEDINPUT

Syntax

SETFORCEDINPUT **EingangName**

Argumente

EingangName Digitale Eingang Name

Beschreibung

Diese Anweisung modifiziert den von **EingangName** angegebenen Eingang auf ON.
Um diese Anweisung verwenden zu können, muß man die Modifizierung der Eingänge mittels der Anweisung ENABLEFORCEDINPUT bereits aktiviert haben.
Siehe auch DISABLEFORCEDINPUT, RESETFORCEDINPUT, SETFORCEDPORT, SETFORCEDANALOG.

SETFORCEDPORT

Syntax

SETFORCEDPORT **NamePort, Wert**

Argumente

NamePort	Name der Vorrichtung des Typs Eingang Port
Variable	Konstante oder Interger oder char-Variable

Beschreibung

Diese Anweisung modifiziert den **Wert** des von **PortName** bestimmten Eingangs-Ports. Der Eingangs-Port wird als Bit-Maske angesehen. Ist der Wert eines Bits 1, wird der entsprechende Eingang auf "ON" modifiziert.
Um diese Anweisung verwenden zu können, muß man die Modifizierung der Eingänge mittels der Anweisung ENABLEFORCEDINPUT bereits aktiviert haben.
Siehe auch DISABLEFORCEDINPUT, SETFORCEDINPUT,RESETFORCEDINPUT, SETFORCEDANALOG.

10.3.19 BlackBox

Die Funktion "BlackBox" soll in einer Datenbank alle Aktivitäten einer Maschine aufnehmen, d.h. ein lokales Modul oder ein Fernmodul. Als "Tätigkeit einer Maschine" gilt die Veränderung im Laufe der Zeit einer Teilmenge aller logischen Geräte, die in GPL verwendet werden können. Dies macht es möglich, das Verhalten der Maschine nachträglich zu analysieren in Zusammenhang mit den gespeicherten Gerätezuständen. Die Datenbank hat eine Tabelle, die eine Zeitangabe und den Status von allen Geräten in jedem Augenblick, einen zu jede Spalte, enthält. In die GPL-Sprache wurden neue Anweisungen zum Starten, zum Beenden und zum Abfragen der Aufnahmeaktivität eingetragen, die nachfolgend beschrieben sind.

Jede BlackBox-Datei ist ein SQLite-Datenbank der Informationen über ein einziges Modell enthält. Der Dateiname schließt die Zahl des Moduls, das Datum und die Uhrzeit hinsichtlich der Aufnahmeaktivität ein.

Aufnahmen werden in den Datenbank transaktional eingetragen. Jede Transaktion enthält höchstens die in 1 Sekunde generierten Aufnahmen. Im Falle von Blackout wird die Kohärenz der Datei garantiert und die letzte Transaktion kann verloren gehen. Die Höchstdauer der Transaktion kann mit einem Eintrag in Tpa.ini geändert werden (für weitere Informationen, bitte nehmen Sie Kontakt mit TPA auf).

Der Dauer der Aufnahme wurde eine Grenze von 12 Stunden eingegeben. Dies bedeutet dass jeder Datenbank wird immer nur die letzten 12 Stunden Aufnahme enthalten. Während der Aufnahme werden vom Datenbank die ältesten Aufnahmen entfernt. Die Höchstdauer der im Datenbank aufgenommenen Geschichte kann durch einen Eintrag in Tpa. Ini geändert werden (für weitere Informationen, bitte nehmen Sie Kontakt mit TPA auf).

Diese Funktion ist verfügbar für physikalische Geräte auf GreenBUS, EtherCAT, CAN, S-CAN, MECHATROLINK-II-Busse, die durch TMSbus, TMSbus+, TMScan, TMScan+, DualMech, DualMech Mono, AlbMech angeschlossen sind.

ENDBLACKBOX

Syntax

ENDBLACKBOX

Beschreibung

Diese Anweisung beendet die Aufnahmefunktion in der Datei jeder Aktivität eines lokalen Moduls oder eines Fernmoduls. Siehe auch STARTBLACKBOX und PAUSEBLACKBOX

PAUSEBLACKBOX

Syntax

PAUSEBLACKBOX

Beschreibung

Diese Anweisung hält die Aufnahmefunktion in der Datei jeder Aktivität eines lokalen Moduls oder eines Fernmoduls an. Um die Aufnahme fortzusetzen, muss man die Anweisung STARTBLACKBOX ohne Argumente ausführen. Siehe auch ENDBLACKBOX.

STARTBLACKBOX

Syntax

STARTBLACKBOX **[Wert][,Fehler]**

Argumente

Wert Konstante oder Variable Integer Aufnahmezeit
Fehler Variable Integer. Fehlercode

Beschreibung

Diese Anweisung aktiviert die Aufnahmefunktion in der Datei jeder Aktivität eines Lokalen Moduls oder eines Fernmoduls. Als Aktivität eines Moduls gilt die Änderung im Laufe der Zeit des Status der logischen Geräten, unter Ausschluss der Flag-Switch-Geräte.
Die Aufnahmezeit (**Wert**) ist in Millisekunden angegeben. Sie darf nicht niedriger als 10 und muss ein Vielfaches der Realtime-Zeit sein. Wäre es nicht der Fall, würde der System Fehler Nr. 4399 (Parameter außerhalb des Bereichs) auftreten.
Wenn die Anweisung eine Aufnahme startet und der **Wert** weggelassen ist, gilt 20 als Standardwert.
Wenn die Anweisung nach einer Unterbrechung die Aufnahme fortsetzt, wird kein festgesetzter **Wert** betrachtet.
Falls nicht möglich wurde, die Aufnahme starten zu lassen, **Fehler** enthält einen Wert ungleich 0, sonst enthält er 0.
Siehe auch PAUSEBLACKBOX und ENDBLACKBOX.

Fehler - Code	Beschreibung
0	Kein Fehler
1	Es gibt Unterschiede in der Konfiguration der Geräte in der Numerischen Steuerung und in Albatros
2	Die Zahl der aufzunehmenden Geräte ist größer als die vorgesehene Höchstzahl des Systems.
3	In der Konfiguration sind keine Geräte verfügbar
4	Im Ferngerät unterstützt die Kommunikationssoftware die BlackBox - Funktion nicht (nur Fernmodule)
5	Die Numerische Steuerung versperrt den Beginn der Registrierung
6	Fehler beim Hochladen der Bibliothek der Datenbankverwaltung
7	Die Zahl der Spalten für die Tabelle ist größer als die Zahl der von der Datenbank bewältigbaren Spalten
8	Man konnte die Datenbank auf der Platte nicht öffnen
9	Man konnte in der Datenbank die Tabelle der Aufnahmen nicht erstellen
10	Fehler bei der IP-Adresse für die Kommunikation mit dem Fernmodul (nur Fernmodule)
11	Man konnte den Kommunikationssocket zum Erhalten der Data nicht erstellen (nur Fernmodule)
12	Man konnte keine lokale Adresse mit dem Kommunikationssocket verbinden (nur Fernmodule)
13	Man konnte sich mit dem Fernsocket nicht verbinden (nur Fernmodule)
14	Man konnte auf den an der numerischen Steuerung beteiligten Arbeitsspeicherbereich nicht zugreifen
15	Die Hardware-Konfiguration versperrt die Verwendung der "BlackBox"- Funktion
16	Die Funktion wurde in tpa.ini deaktiviert

Siehe auch PAUSEBLACKBOX und ENDBLACKBOX.

10.3.20 ISO

ISOG0

Syntax

ISOG0 **Etikett, Achse1 Maß1, Achse2, Maß2, Achse3, Maß3, Achse4, Maß4,Achse5, Maß5, [Wert]**

Argumente

Etikett Konstante oder Variable Integer Etikett, das den Versetzungsblock identifiziert. N im Standard-ISO
Achse1 Name der Vorrichtung des Typs Achse. (X im Standard-ISO)
Maß1 Konstante oder Variable. Maß des Betriebsraumes der Achse1
Achse1 Name der Vorrichtung des Typs Achse. (Y im Standard-ISO)
Maß2 Konstante oder Variable. Maß des Betriebsraumes der Achse2

Achse3	Name der Vorrichtung des Typs Achse. (Z im Standard-ISO)
Maß3	Konstante oder Variable. Maß des Betriebsraumes der Achse3
Achse4	Name der Vorrichtung des Typs Achse. (C im Standard-ISO)
Maß4	Konstante oder Variable. Maß des Gelenkraums der Achse4
Achse5	Name der Vorrichtung des Typs Achse. (B im Standard-ISO)
Maß5	Konstante oder Variable. Maß des Gelenkraums der Achse5
Wert	Konstante oder Variable double. Stellt den Prozentsatz von feed rate dar (F im Standard-ISO).

Beschreibung

Durch diese Anweisung wird die Schnellbewegung eingestellt. Die Strecken der Schnellbewegungen sind in Synchronisation verwaltet. Die vom Benutzer definierten Punkte sind die Extrema der einzelnen Versetzungsstrecke, die so belaufen wird, dass die Achsen miteinander synchronisiert sind. Dies bedeutet dass die physikalischen Achsen sich unabhängig voneinander bewegen, während sie jedoch gleichzeitigen Start- und Abschlusspunkt aufbewahren, wie in den MULTIABS und MULTIINC Anweisungen. Die Werkzeugspitze beläuft keine Strecke im Betriebsraum und seine Trajektorie erweist sich als nicht geprüft. Der Parameter **Etikett** wird mit der SETLABELINTERP-Instruktionverwendet, um den Versetzungsblock eindeutig zu identifizieren. Die ersten drei **Maße** identifizieren die Position der Spitze im Betriebsraum, während die nachfolgenden zwei den Drehachsenwert im Gelenkraum bestimmen. Der **Wert** von FeedRate definiert die Reduzierungsquote hinsichtlich der möglichen Höchstgeschwindigkeit (In ISO: F0 Höchstgeschwindigkeit, F100 FeedRate null, dann stillstehende Achsen). Die Anweisung erzeugt einen Systemfehler (4105 – Auf der Achse AchseName nicht ausführbare Anweisung), falls sie auf Schrittachsen verwendet ist.
Es ist nicht möglich die Anweisung WAITCOLL zu verwenden; nämlich würde das Interpolationsverhältnis mit den anderen Achsen, die an der Bewegung mitwirken, von der Kollision an verloren gehen, und demzufolge würde ein abweichendes Profil von den erwarteten generiert werden. Falls diese Anweisung verwendet ist, wird der Systemfehler „4101 – Unpassende Verwaltung der Achse AchseName" generiert.

ISOG1

Syntax

ISOG1 **Etikett, Achse1, Maß1, Achse2, Maß2, Achse3, Maß3, Achse4, Maß4, Achse5, Maß5, [Wert]**

Argumente

Etikett	Konstante oder Variable Integer. Etikett, das den Versetzungsblock identifiziert. (N im Standard-ISO
Achse1	Name der Vorrichtung des Typs Achse. (X im Standard-ISO)
Maß1	Konstante oder Variable. Maß des Betriebsraumes der Achse1
Achse2	Name der Vorrichtung des Typs Achse. (Y im Standard-ISO)
Maß2	Konstante oder Variable. Maß des Betriebsraumes der Achse2
Achse3	Name der Vorrichtung des Typs Achse. (Z im Standard-ISO)
Maß3	Konstante oder Variable. Maß des Betriebsraumes der Achse3
Achse4	Name der Vorrichtung des Typs Achse. (C im Standard-ISO)
Maß4	Konstante oder Variable. Maß des Gelenkraums der Achse4
Achse5	Name der Vorrichtung des Typs Achse. (B im Standard-ISO)
Maß5	Konstante oder Variable. Maß des Gelenkraums der Achse5
Wert	Konstante oder Variable double. Sie stellt den Wert von Feed Rate dar (F im Standard-ISO).

Beschreibung

Durch diese Anweisung wird im Betriebsraum der Punkt definiert, der von der Werkzeugspitze am Schluss der Interpolation des ausgewählten Blocks erreicht werden soll. Der Parameter **Etikett** wird mit der SETLABELINTERP - Anweisung verwendet, um den Versetzungsblock eindeutig zu identifizieren. Die ersten drei **Maße** identifizieren die Position der Spitze im Betriebsraum, während die nachfolgenden zwei den Drehachsenwert im Gelenkraum bestimmen. Der **Wert** Feed definiert die Geschwindigkeit der Spitze als Messeinheit (Millimeter oder Grad) pro Minute (bei der ISOG94 - Instruktion) als Inverse der Ausführungszeit
(im Beisein von der ISOG93 - Instruktion). Der Parameter **Wert** ist obligatorisch für die erste ISOG1-Anweisung der Interpolationsbewegung. Die Anweisung erzeugt einen Systemfehler (4105 – Auf der Achse AchseName nicht ausführbare Anweisung), falls sie auf Schrittachsen verwendet ist.
Es ist nicht möglich die Anweisung WAITCOLL zu verwenden; nämlich würde das Interpolationsverhältnis mit den anderen Achsen, die an der Bewegung mitwirken, von der Kollision an verloren gehen, und demzufolge würde ein abweichendes Profil von den erwarteten generiert werden. Falls diese Anweisung verwendet ist, wird der Systemfehler „4101 – Unpassende Verwaltung der Achse AchseName" generiert.

ISOG9

Syntax

ISOG9 **Achse**

Argumente

Achse Name einer Vorrichtung des Typs Achse

Beschreibung

Durch diese Anweisung wird das Zwangsausschalten der Bewegung ermöglicht. Im Beisein von dieser Instruktion werden die Interpolation oder die Schnellbewegung vor der Übergang in den nachkommenden Block aufgehalten. Es handelt sich nicht jedoch um eine sperrende Anweisung, wie die WAITSTILL - Anweisung ist. Informationen über einen Zwangshalt werden der Steuerung übermittelt, dann geht der Aufnahmeprozess der bewegenden Blöcke bis zur Lookahead-Erfüllung weiter. Der zu haltende 5-Achsen-Interpolationskanal wird vom Parameter **Achse** am Ende des vorher kalkulierten Blocks ermittelt. In diesem Fall ist die Ausführung einer ISOG1 - Anweisung oder einer ISOG0 - Anweisung gleichgültig.

ISOG90

Syntax

ISOG90 **Achse**

Argumente

Achse

Name einer Vorrichtung des Typs Achse

Beschreibung

Durch diese Anweisung wird die Interpretation der Maße als absolute Maße eingestellt. Der Parameter **Achse** ermittelt den 5-Achsen-Interpolationskanal, der von dieser Anweisung an die Achsenmaße als absolute Maße (Default-Zustand) interpretiert. In diesem Fall ist die Ausführung einer ISOG1 - Anweisung oder einer ISOG0 - Anweisung gleichgültig.

ISOG91

Syntax

ISOG91 **Achse**

Argumente

Achse Name einer Vorrichtung des Typs Achse

Beschreibung

Durch diese Anweisung wird die Interpretation der Maße als relative Maße eingestellt. Der Parameter **Achse** ermittelt den 5-Achsen-Interpolationskanal, der von dieser Anweisung an die Achsenmaße als relative Maße (Default-Zustand) interpretiert. In diesem Fall ist die Ausführung einer ISOG1 - Anweisung oder einer ISOG0 - Anweisung gleichgültig.

ISOG93

Syntax

ISOG93 **Achse**

Argumente

Achse Name einer Vorrichtung des Typs Achse

Beschreibung

Durch diese Anweisung wird die Interpretation der Geschwindigkeiten als Inverse der Ausführungszeit eingestellt. Der Parameter **Achse** ermittelt den 5-Achsen-Interpolationskanal, der von dieser Anweisung an den aus den F-Parametern entstehenden Wert der ISOG1 - Anweisung als Inverse der Ausführungszeit (in Minuten) interpretiert. Aus diesem Grund ist die Steuerung in der Lage, die Geschwindigkeit zu bestimmen, die die Werkzeugspitze in den Interpolationsblöcken halten soll.

ISOG94

Syntax

ISOG94 **Achse**

Argumente

Achse Name einer Vorrichtung des Typs Achse

Beschreibung

Durch diese Anweisung wird die Interpretation der Geschwindigkeiten als Messeinheit pro Minute eingestellt. Der Parameter **Achse** ermittelt den 5-Achsen-Interpolationskanal, der ab dieser Anweisung an die Geschwindigkeiten als Messeinheit pro Minute (Default-Zustand) interpretiert.

ISOG216

Syntax

DrehachsenMatrixName, WerkzeughalterMatrixName,

Argumente

DrehachsenMatrixName	Name der Matrix. Sie enthält die Daten über die Drehachsen
WerkzeughalterMatrixName	Name der Matrix. Sie enthält die Daten über den Werkzeughalter Name der Matrix. Sie enthält die Daten über die Werkzeuge
Freigabemaske	Variable oder Konstante Integer. Freigabemaske der Achsen C und B
Achse1	Name der Vorrichtung des Typs Achse. (X im Standard-ISO)
Achse2	Name der Vorrichtung des Typs Achse. (Y im Standard-ISO)
Achse3	Name der Vorrichtung des Typs Achse. (Z im Standard-ISO)
Achse4	Name der Vorrichtung des Typs Achse. (C im Standard-ISO)
Achse5	Name der Vorrichtung des Typs Achse. (B im Standard-ISO)

Beschreibung

Identifikation der drei Matrizen für der Maschinenparametrisierung und der fünf Vorrichtungen von Achsentyp, voraus dieselbe besteht.Diese Anweisung **muss** vor jeder anderen ISO - Anweisung durchgeführt werden. Der Parameter **FreigabeMaske** definiert welche Drehachsen (C und/oder B) freizugeben. In der nachfolgenden Tabelle sind die festzulegenden Werte gelistet:

FreigabeMaske	Description
31	Aktivierung der Achsen C und B
23	Aktivierung nur der Achse B
15	Aktivierung nur der Achse C
7	Deaktivierung der Achsen C und B

Bemerkung

Die Maßeinheit der in der Konfiguration auszudrückenden Werte der Drehachsen muss Grad sein.
Die durch diese Anweisung eingestellte Verbindung zwischen den physikalischen Achsen und den virtuellen ISO- Achsen wird entweder durch die ISOM2 - Anweisung gelöst oder wenn der Task, in dem die Anweisung definiert ist, zu Ende gebracht. Demzufolge können die Achsen für klassische Bewegungen benutzt werden.

ISOG217

Syntax

ISOG217

Achse1,Achse2,Achse3,Achse4,Achse5,virtuelleAchse1,virtuelleAchse2,virtuelleAchse3,virtuelleAchse4,virtuelleAchse5

Argumente

Achse1	Name einer Achsenvorrichtung
Achse2	Name einer Achsenvorrichtung
Achse3	Name einer Achsenvorrichtung
Achse4	Name einer Achsenvorrichtung
Achse5	Name einer Achsenvorrichtung
virtuelleAchse1	Name einer Vorrichtung virtueller Achse (Y im ISO-Standard)
virtuelleAchse2	Name einer Vorrichtung virtueller Achse (Y im ISO-Standard)
virtuelleAchse3	Name einer Vorrichtung virtueller Achse (Z im ISO-Standard)
virtuelleAchse4	Name einer Vorrichtung virtueller Achse (C im ISO-Standard)
virtuelleAchse5	Name einer Vorrichtung virtueller Achse (B im ISO-Standard)

Beschreibung

Diese Anweisung beschreibt die physischen Achsen und die virtuellen Achsen, die die Maschinen zusammenstellen. Die virtuellen Achsen, die die Position und die Orientierung des Werkzeugs beschreiben, müssen in der Konfiguration von Albatros als virtuelle Achsen deklariert werden. Die ersten fünf spezifizierten Achsen müssen physische Achsen sein und sind vom Interpolator gesteuert werden. Die nachkommenden fünf müssen virtuelle Achsen sein; sie sind die Achsen die in den Anweisungen ISOG0 und ISOG1 verwendet werden.
Diese Anweisung **muss** vor jeder anderen ISO - Anweisung durchgeführt werden.
Die Formel von direkter und inverser Kinematik, um von einer Position im Gelenkraum (physische Achsen) zum Betriebsraum (virtuelle Achsen) zu springen, müssen durch die Anweisung KINEMATICEXPR für jeden der zehn definierten Achsen der ISOG217- Anweisung spezifiziert werden.
Die Anweisung erzeugt einen Systemfehler (4105 – Auf der Achse AchseName nicht ausführbare Anweisung), falls sie auf Achsen im Schrittbetrieb verwendet ist.

Bemerkung

Die durch diese Anweisung eingestellte Bindung zwischen den physikalischen Achsen und den virtuellen ISO- Achsen wird gelöst, wenn der Task, in dem die Anweisung definiert ist, zu Ende gebracht wird oder wenn eine ISOM2 durchgeführt wird. Demzufolge können die Achsen für klassische Bewegungen verwendet werden.

ISOM2

Syntax

ISOM2	**Achse**

Argumente

Achse	Name einer Vorrichtung des Typs Achse

Beschreibung

Durch diese Anweisung werden die Achsen von der ISO-Bewegung, die durch die ISOG216- oder die ISOG217 - Anweisung eingestellt worden ist, erlöst.

ISOM6

Syntax

ISOM6	**Achse, DrehachsenMatrixZeilenIndex, WerkzeughalterMatrixZeilenIndex, WerkzeugeMatrixZeilenIndex**

Argumente

Achse	Name der Vorrichtung des Typs Achse
DrehachsenMatrixZeilenIndex	Konstante oder Variable Integer. Zeilenindex der Matrix der Drehachsen
WerkzeughalterMatrixZeilenIndex	Konstante oder Variable Integer. Zeilenindex der Matrix des Werkzueghalters
WerkzeugeMatrixZeilenIndex	Konstante oder Variable Integer. Zeilenindex der Matrix der Werkzuege

Beschreibung

Durch diese Anweisung wird eine Gruppe von Parametern eingestellt, die die Kinematik der Maschine beschreiben. Die **Indices** beziehen sich auf drei Matrizen, deren Name vom ’Benutzer festgelegt ist; sie sind in der Datei der globalen Variablen von Albatros angegeben. Der Parameter **Achse**

identifiziert den entsprechenden Interpolationskanal. In den nachfolgenden Tabellen wird aufgezeigt, wie die drei Matrizen in der Datei der globalen Variablen anzugeben sind:

Matrix-Felder	Matrix der Drehachsen
X - Offset	Offset entlang X zwischen dem Drehpunkt und der Kopfsteuerung
Y - Offset	Offset entlang Y zwischen dem Drehpunkt und der Kopfsteuerung
Z - Offset	Offset entlang Z zwischen dem Drehpunkt und der Kopfsteuerung
X - Achsenversatz	X - Abweichung zwischen der Rotationsachse und der Schwenkungsachse (wenn die Position der C-Achse = 0)
Y- Achsenversatz	Y - Abweichung zwischen der Rotationsachse und der Schwenkungsachse (wenn die Position der C-Achse = 0)
Z- Achsenversatz	Abstand Nase-Drehpunkt
δ - Winkel	Winkel um Z zur korrrekten Positionierung des Kopfes in Beziehung zum Maschinennullpunkt
γ - Winkel	Winkel zwischen der Drehebene und der Schwenkungsebene

Matrix-Felder	Werkzeugshalter-Matrix
PU X-Offset	Offset in X zwischen dem Kupplungspunkt des Werkzeughalters und dem Kupplungspunkt des Werzeuges mit dem Werkzeugshalter (wenn die Position der C-Achse = 0 und Vertikalmotor)
PU Y-Offset	Offset in Y zwischen dem Kupplungspunkt des Werkzeughalters und dem Kupplungspunkt des Werzeuges mit dem Werkzeugshalter (wenn die Position der C-Achse = 0 und Vertikalmotor)
PU Z - Offset	Offset in Z zwischen dem Kupplungspunkt des Werkzeughalters und dem Kupplungspunkt des Werzeuges mit dem Werkzeugshalter (wenn die Position der C-Achse = 0 und Vertikalmotor)
α - Winkel	Winkelversatz zwischen der Motorachse und der Werkzeugsachse (in Bezug auf Z)
β - Winkel	Winkelversatz zwischen der Motorachse und der Werkzeugsachse (in Bezug auf Y)

Matrix-Felder	Werkzeugmatrix
Werkzeuglänge	Länge des Werkzeuges

ISOSETPARAM

Syntax

ISOSETPARAM **ParameterKennzeichen, Wert**

Argumente

ParameterKennzeichen Konstante oder Variable Integer. Das ist die Zahl, die einen Parameter ermittelt

Wert Konstante oder- Float-Variable. Einzustellender Wert.

Beschreibung

Diese Anweisung stellt einige Parameter ein, die das Fließvermögen der interpolierten ISO-Bewegung bezeichnen. In der nachfolgenden Tabelle sind die Bedeutung von jedem **ParameterKennzeichen,** die Werte in denen die Variable **Wert** und die Defaultwerte eingeschlossen sein sollten, erklärt.

ParameterKennzeichen	Bereich	Default	Bedeutung
0	0.0-100.0	50.0	Prozentsatz der Geschwindigkeitsabnahme auf den linearen Achsen im Falle eines Eckpunktes.

			(0=Keine Geschwindigkeitsabnahme, 100=Höchstwert der vom Interpolator zugelassenen Geschwindigkeitsabnahme)
1	0.0-100.0	50.0	Prozentsatz der Geschwindigkeitsabnahme auf den rotierenden Achsen im Falle eines Eckpunktes. (0=Keine Geschwindigkeitsabnahme, 100=Höchstwert der vom Interpolator zugelassenen Geschwindigkeitsabnahme)
2	0.5-1.0	0.9	Faktor von Geschwindigkeitsverminderug auf krummliniger Abszisse im Falle eines Eckpunktes. (1=keine Geschwindigkeitsabnahme, 0.5=Höchstwert der zugelassenen Abnahme)
3	0.0-100.0	60.0	Prozentsatz der Geschwindigkeitsabnahme im Falle von nahen Diskontinuitäten. (0=keine Abnahme, 100= Höchstwert der vom Interpolator zugelassenen Abnahme)
4	0.0-100.0	10.0	Smooth-Prozentsatz der Trajektorie
5	(Größer als 0.0)-1.0	0.2	Mindestgröße der zurückzulegende Strecke nur mit linearen Achsen. Der Wert ist in Millimetern ausgedrückt.
6	(Größer als 0.0)-1.0	0.1	Mindestrgröße der zurückzulegende Strecke nur mit Drehachsen. Der Wert ist in Grad ausgedrückt
7	0.0-100.0	100.0	Smooth Filteruntergrenze
8	1.0.100.0	1.0	Multiplikationsfaktor angewendet auf die

			in der Konfiguration definierten Beschleunigungen und Verzögerungen. Er inkrementiert die maximalen Beschleunigung und Verzögerung nur der linearen Achsen. Werte außerhalb des Intervalls generieren den Systemfehler 4399 – Parameter außerhalb des Bereichs.
9	1.0-100.0	1.0	Multiplikationsfaktor angewendet auf die in der Konfiguration definierten Beschleunigungen und Verzögerungen. Er inkrementiert die maximalen Beschleunigung und Verzögerung nur der rotierenden Achsen. Werte außerhalb des Intervalls generieren den Systemfehler 4399 – Parameter außerhalb des Bereichs.
10	0.0-1.0	0.0	Flag um im Falle eines Eckpunktes die Geschwindigkeitsverminderung zwischen aufeinander folgenden Blöcken zu aktivieren (0.0) oder deaktivieren (1.0). Die Deaktivierung kann auch durch die Verwendung der Anweisung mit den folgenden Parametern ISOSETPARAM 0 0.0 ISOSETPARAM 1 0.0 ISOSETPARAM 2 1.0 erreicht werden.

KINEMATICEXPR

Syntax

KINEMATICEXPR **Achse = Ausdruck**

Argumente

Achse Name einer Vorrichtung physischer oder virtueller Achse
Ausdruck Menge von Operatoren

Beschreibung

Diese Anweisung ermöglicht, die einzelnen Ausdrücken von direkter und inverser Kinematik zu definieren. Vor der Ausführung dieser Anweisung, muss die Anweisung ISOG217 abgerufen werden. Diese Anweisung beschreibt die physischen Achsen und die virtuellen Achsen, die die Maschinen zusammenstellen. Für jede in der ISOG217 definierte Achse muss die Anweisung KINEMATICECPR

abgerufen werden. Der Ausdruck von Kinematik einer Achse im Gelenkraum (inverse Kinematik) kann eine Funktion von
Variablen;
Konstanten;
Koordinaten der Achsen im Betriebsraum
sein.
Die Syntax **des Ausdrucks** ist gleich wie der in der Anweisung EXPR beschriebene Ausdruck. Der Unterschied besteht darin, dass keine lokale Variable verwendet werden kann.

Außerdem, kann man die Achsen nicht verwenden, die des gleichen Typs wie der in der **Achse** sind und die nicht in der Anweisung ISOG217 deklariert sind. Zum Beispiel, wird gerade die Kinematik einer virtuellen Achse definiert, die in der Anweisung ISOG217 bereits deklariert ist, dann können in dem Ausdruck nur die ersten fünf physischen Achsen verwendet werden, die in der ISOG217 deklariert sind.

Beispiel

```
ut as double              ; Werkzeugsnummer
offsety as double         ; offset Y Nase Drehpunkt
offsetz as double         ; offset Z Nase Drehpunkt

Funktion ISO5Ax


    setval 100,ut
    setval 120.0,offsety
    setval 60.0,offsetz
    ; EXPLIZITE KINEMATIK
    ISOG217 Rx Ry Rz Rc Rb X Y Z C B

    ; DEFINITION DER AUSDRÜCKE DER KINEMATIK
    ; EXPLIZITE INVERSE KINEMATIK physische Rx-Achse
    KinematicExpr Rx = X - 135 + ut * sin ( B ) * cos ( C )

    ; EXPLIZITE INVERSE KINEMATIK physische Ry-Achse
    KinematicExpr Ry = Y + offsety + ut * sin ( B ) * sin ( C )

    ; EXPLIZITE INVERSE KINEMATIK Rz-ACHSE
    KinematicExpr Rz = Z + offsetz + ut * cos ( B )

    ; EXPLIZITE INVERSE KINEMATIK Rc-ACHSE
    KinematicExpr Rc = C

    ; EXPLIZITE INVERSE KINEMATIK Rb-ACHSE
    KinematicExpr Rb = B

    ; EXPLIZITE DIREKTE KINEMATIK X-ACHSE
    KinematicExpr X = Rx + 135 - ut * sin ( Rb ) * cos ( C )

    ; EXPLIZITE INVERSE KINEMATIK Y-ACHSE
    KinematicExpr Y = Ry - offsety - ut * sin ( Rb ) * sin ( C )

    ; EXPLIZITE DIREKTE KINEMATIK Z-ACHSE
    KinematicExpr Z = Rz - offsetz - ut * cos ( Rb )

    ; EXPLIZITE DIREKTE KINEMATIK C-ACHSE
    KinematicExpr C = Rc

    ; EXPLIZITE DIREKTE KINEMATIK B-ACHSE
    KinematicExpr B = Rb


    ; BEWEGUNG
    ISOG0 1001,X 998.0,Y 600.0,Z 0.0,C 90.0,B 45.0,0.0
    ISOG1 1001,X 998.0,Y 600.0,Z 0.0,C 90.0,B 45.0,1000.0
    ISOG1 1001,X 998.0,Y 600.0,Z 0.0,C 90.0,B 45.0,1000.0
```

```
    ISOG1 1003,X 996.0,Y 600.0,Z 0.0,C 90.0,B 45.0,1000.0
    WAITSTILL  X
    ISOM2 X
FRet
```

10.3.21 Nicht mehr verfügbare Anweisungen

GETVF	liest den Wert des Spannung/Frequenz-Koverters
INPBCD	liest eine Serie von digitalen Nibbles im Format BCD ab
OUTBCD	ändert eine Reihe von digitalen Nibbles im BCD-Format
SETFORCEDBCD	forciert einen Satz von Nibble im Format BCD
CANOPENDRIVER	öffnet einen CANopen-Kommunikationskanal
CANCLOSEDRIVER	schließt einen CANopen-Kommunikationskanal
CANRESETBOARD	führt ein Reset der CANopen-Platine aus
CANSETOBJECT	schreibt ein CANopen-Objekt
CANGETOBJECT	liest ein CANopen-Objekt ab
SLMCOMMAND	führt einen SLM-Befehl aus
SLMEEPROMDISABLE	führt einen Befehl zum Sperren des Schreibens auf EEPROM aus
SLMEEPROMENABLE	führt einen Befehl zur Freigabe des Schreibens auf EEPROM aus
SLMGETEEPROM	liest einen EEPROM-Speicherplatz ab
SLMGETPARAM	liest einen SLM-Parameter ab
SLMGETREGISTER	liest ein SLM-Register ab
SLMGETSTATUS	liest eine Größe des Antriebs ab
SLMSETEEPROM	schreibt einen EEPROM-Speicherplatz
SLMSETPARAM	legt einen SLM-Parameter fest
SLMSETREGISTER	legt ein SLM-Register fest
HOMING	sucht die "Nullstellung"
SYNCROOPEN	öffnet einen Kanal zur synchronisierten Bewegung
SYNCROCLOSE	schließt einen Kanal der synchronisierten Bewegung
SYNCROMOVE	weist einen Punkt der synchronisierten Bewegung zu
SYNCROSETACC	legt die Beschleunigung für die synchronisierten Bewegungen fest
SYNCROSETDEC	legt die Verzögerung für die synchronisierten Bewegungen fest
SYNCROSETVEL	legt die Achsengeschwindigkeit für die synchronisierten Bewegungen fest
SYNCROSETFEED	schaltet in einer synchronisierten Bewegung die Achsengeschwindigkeiten zurück
SYNCROSTARTMOVE	startet die Bearbeitung einer synchronisierten Bewegung

10.3.22 Bei Interrupt nicht verwendbare Anweisungen

Folgende Anweisungen können bei Funktionen, die von den Anweisungen ONFLAG, ONINPUT und ONERRSYS aufgerufen werden, nicht verwendet werden. Das Verbot der Verwendung gilt auch in Realtime - Tasks.

Anweisungen, die ihrerseits eine Funktion bei Interrupt aufrufen:

- ONFLAG
- ONINPUT
- ONERRSYS

Anweisungen, die mit einer Wartezeit verbunden sind:

- WAITINPUT
- WAITFLAG
- WAITACC
- WAITCOLL
- WAITDEC
- WAITREG
- WAITTARGET
- WAITWIN
- WAITSTILL
- WAITTASK
- WAITRECEIVE
- WAITPERSISTINPUT
- MULTIWAITFLAG
- MULTIWAITINPUT

Kommunikationsanweisungen:
- SEND
- RECEIVE
- CLEARRECEIVE
- COMOPEN
- COMCLOSE
- COMREAD
- COMREADSTRING
- COMWRITE
- COMWRITESTRING
- COMGETERROR
- COMCLEARRXPufferspeicher
- COMGETRXCOUNT

Folgende, mit der Achsenbewegung verbundene Anweisungen:
- MOVINC
- MOVABS
- LINEARINC
- LINEARABS
- CIRCLE
- CIRCINC
- CIRCABS
- HELICINC
- HELICABS
- COORDIN
- SETRIFLOC
- RESRIFLOC
- SETPFLY
- SETPZERO
- SETINDEXINTERP
- STARTINTERP
- FASTREAD
- ENABLE
- DISABLE
- ENDMOV

ISO-Anweisungen
- ISOG0
- ISOG1
- ISOG9
- ISOG90
- ISOG91
- ISOG93
- ISOG94
- ISOG216
- ISOG217
- ISOM2
- ISOM6
- ISOSETPARAM
- KINEMATICEXPR

BlackBox-Anweisungen:
- ENDBLACKBOX
- PAUSEBLACKBOX
- STARTBLACKBOX

EtherCAT II betreffende Anweisungen:
- READDICTIONARY
- WRITEDICTIONARY

Multitasking betreffende Anweisungen:
- SENDMAIL
- WAITMAIL
- ENDMAIL
- SENDIPC
- WAITIPC

- TESTMAIL
- TESTIPC

Anweisungen, die eine lange Verarbeitungszeit beinhalten:

- SORT
- FIND
- FINDB
- MOVEMAT

10.4 Beispiele

10.4.1 Nullstellung bei Interrupt

```
;---------------------------------------------------------------------
;  Beispiel für Eil-Nullstellung - Routine
;
;  Die Funktion führt folgende Vorgänge aus:
;
;     1) Stellt die Achse durch Deaktivierung der Software-Grenzen
;        und Nullsetzen des Maßes ein
;     2) Prüft, daß der Richtungr nicht bereits im Zustand ON ist.
;        Ist er im Zustand ON, wird die Achse bewegt und abgewartet
;        bis sie wieder im Zustand OFF ist. Erfolgt dies nicht innerhalb
von 30 Sekunden,
;        wird eine Fehlermeldung erzeugt.
;     3) Stellt die Geschwindigkeit zur Suche des Richtungrs ein
;     4) Startet eine Bewegung der Achse und aktiviert die Eil-Nullstellung
- Vorgang
;        für dieselbe Achse. Beim Ansprechen des Interrupts
;        wird das Maß der Achse auf Null gesetzt und automatisch
;        eine Bewegung zu einem Freigabe-Maß gestartet.
;     5) Wartet ab bis die Achse am Freigabe-Maß ist
;     6) Aktiviert erneut die Achsengrenzen
;
;
;   © TPA.
;---------------------------------------------------------------------

Function Fast_Nullstellung

  ResLimPos     Achse                          ; Achsen-Initialisierung
  ResLimNeg     Achse
  SetQuote      Achse,0

  IfInput       FastInput,OFF,Goto Continue    ; Prüft, ob Richtung
                                               ; belegt ist
  SetVel        Achse,5                        ; Setzt
                                               ; Freigabe-Geschwindigkeit
  MovAbs        Achse,30                       ; Bewegt die Achse
  WaitInput     FastInput,OFF,30,Call Error    ; Steuert Freigabe
                                               ; Mikro,
                                               ; Fehler nach TimeOut=30

  EndMov        Achse                          ; Achsen-Stop
  WaitStill     Achse                          ; Achsen-Wartezeit

  Continue:
  SetVel        Achse,10                       ; Suchgeschwindigkeit
                                              Nullstellung-Richtung
  MovAbs        Achse,-1000                    ; Negative Bewegung zur
                                              Suche der Richtung
  SetPFly       Achse,ON,10,0                  ; Interrupt-Anschluss
                                               ; und Einstellung des
                                              Freigabe-Maßes
                                               ; und der Geschwindigkeit
  WaitStill     Achse                          ; Achsen-Wartezeit
```

```
    SetLimPos      Achse                          ; Erneutes Aktivieren der
                                                  Achsengrenzen
    SetLimNeg      Achse

    Fret

; Unterprogramm zum Senden einer Fehlermeldung
Error:
    Error          ERR_SETP                       ;Fehlermeldung Fortführung
                                                  unmöglich
    Ret
```

10.4.2 Server der Achsenbewegung

```
;-------------------------------------------------------------
; Beispiel für einen Server der Achsenbewegung:
;
;  Der Server bewegt die Achsen der Maschine
;  im Auftrag anderer Tasks.
;
;  Die Client-Tasks senden Befehle in der Form von
;  Nachrichten (mail) an eine Mailbox.
;
;  Der Server entnimmt die Befehle aus der Mailbox und führt sie aus.
;
;  Die Anforderungen bilden eine Schlange in der Mailbox, d.h.
;  kommt eine Anforderung während der Server belegt ist,
;  geht sie nicht verloren, sondern wird sobald wie möglich ausgeführt.
;
;  Der Server ist der einzige Task zur Achsenbewegung. Auf
;  diese Weise werden Konflikte vermieden.
;
;  Der Server wird von der Funktion Achse_Master implementiert.
;
;  Ein Beispiel für einen Client wird von der Funktion Check_Flag
;  implementiert.
;  Diese Funktion prüft regelmäßig den Zustand eines
;  Flags und, wenn es im Zustand ON angefunden wird, sendet
;  sie dem Server den Befehl zur Ausführung des Achsen-Nullstellungs.
;  Das Flag wird voraussichtlich manuell vom Bediener auf ON
;  gesetzt, z.B. mittels einer synoptischen Darstellung.
;
;-------------------------------------------------------------

;-----------------------------------
; -- GLOBALE MASCHINENKONSTANTEN --
;-----------------------------------
Const MBOX = 101  ; bestimmt die Mailbox
                  ; Befehls
Const SETP = 10   ; Achsen-Nullstellung
Const CHG =  11   ; Wechsel des Werkzeugs
Const FORO = 12   ; Ausführung einer Bohrung



;---------------------
; --- ACHSENGRUPPE ---
;---------------------

; Definition der Fehlermeldungen
Defmsg ERR_CMD "Befehl unbekannte Achsengruppe"

; --- Server ---
Function Master_Achsen autorun


  local cmd as integer             ; Befehl
```

```
  local cota_X as double            ; X-Maß Bohrung
  local cota_Y as double            ; Y-Maß Bohrung

loop:
  waitmail MBOX,cmd,cota_X,cota_Y   ; Befehl abwarten

  ; Bei Erhalt des Befels wird er bestimmt
  ; und die pAchsenende Maßnahme unternommen
  Select cmd


  case SETP
   fcall Achsen_Nullstellung        ; Nullstellung der Achsen
  case CHG
   fcall Werkzeug_Wechsel           ; Führt einen Wechsel des Werkzeugs aus
  case FORO
   fcall Bohrung X_Maß,Y_Maß        ; Bohrung an den
                                    ; angegebenen Maßen

  case else
   call error
  endselect

  endmail MBOX                      ; weist auf die Ausführung des Befehls
                                    hin
  goto loop                         ; kehrt zum Abwarten eines neuen
                                    ; Befehls zurück
  fret

; Unterprogramm zum Senden einer Fehlermeldung
error:
  error ERR_CMD
  ret


;-------------------------
; --- GRUPPE ALLGEMEIN ---
;-------------------------

; --- Client ---
Function Check_flag

loop:

   ifflag Setp_assi,OFF, goto loop  ; prüft den Flag-Zustand

                                    ; OK das Flag steht auf ON, Befehl wird
gesendet
   sendmail MBOX,WAITTACK,SETP,0.0,0.0

   resetflag Setp_Achsen            ; Reset des Flags

   goto loop                        ; zurück zum Abwarten

   fret



loop:

  ifflag Setp_ejes,OFF, goto loop   ; prüft den Flag-Zustand

  ; OK das Flag steht auf ON, Befehl wird gesendet
  sendmail MBOX,WAITTACK,SETP,0.0,0.0
```

```
    resetflag Setp_ejes              ; Reset des Flags
    goto loop                        ; zurück zum Abwarten
    fret

     ; BITTE BEACHTEN SIE:
     ; - nach dem Befehl "SETP", sind die zwei Parameter
     ;   "X_Maß" und "Y_Maß" anzugeben, auch wenn dies
     ;   keinen Sinn für den Nullstellung-Vorgang hat.
     ;   Der Server kann allerdings nicht im vorhinein wissen,
     ;   welcher Befehl ihm gesendet wird; deswegen sind zwei
     ;   Werte desjenigen Typs anzugeben, die der Server erwartet,
     ;   in diesem Fall zwei DOUBLE. Die übergebenen Werte lauten
     ;   "0.0" und "0.0".
     ; - der Parameter "WAITACK" sorgt dafür, daß der Client abwartet
     ;   bis der Server den Befehl ausgeführt hat.
     ;   Der Client fährt mit der eigenen Ausführung erst fort,
     ;   wenn der Server eine ENDMAIL-Anweisung ausgeführt oder
     ;   die Verarbeitung eines neuen Befehls (WAITMAIL) begonnen hat.
```

10.4.3 Main-Zyklus mit Fehlerverwaltung

```
;-------------------------------------------------------------
; Hypothetische Hauptfunktion
; initialisiert die Maschine und führt einen Kontrollzyklus aus
;-------------------------------------------------------------
Function Main

     OnErrSysGestErrS ; aktiviert die Fehlerverwaltung
              ys

     StartTasNotfälle ; initialisiert
     k
     StartTasVerarbei
     k       ter
     AchsenAk
     tivierun
     g

loop:
      IfFlag Flag,OFF, NotWiederherst
      .........
      Goto loop
Fret


;-----------------------------------
; Fehlerverwaltungs-Funktion
;-----------------------------------
Function GestErrSys
      Param nFehler as integer
      Param task as function
      Param Typdevice as device

      EndTask Task;                          ; Beendet Verarbeiter-Task
      If nFehler, >, 5, goto noerraxis       ; Die ersten 5 Fehler
                                             ; betreffen
                                             ; die Achsen
      ResetFlag Flag
      AchsenDeaktivierung
```

```
noerraxis:
Fret
```

10.4.4 Operationen an Strings

```
;-------------------------------------------------------------
; Beispiel für die Handhabung von Strings
;-------------------------------------------------------------
Function Beispiel
  Local       String1 as string
  Local       String2 as string
  Local       String3 as string
  Local       Länge as integer
  Local       Position as integer

  SetString "Probe-",String1              ; String1 enthält
                                          ; nun "Probe-"
  SetString "String",String2

  AddString String1,String2,String3       ; String3 enthält
                                          ; "String"
  Search    String3,'t',Position          ; Position gilt 2
  Search    String3,'Z',posizione         ; Position gilt -1

  Left      String3,7,String1             ; String1
                                          ; enthält "Probe-"

  Right     String3,2,String2             ; String2
                                          ; enthält "ng"

  Mid       String3,7,2,String3           ; String3
                                          ; enthält "St"

  ControlCh 65,String1                    ; String1
    ar
                                          ; enthält "A"

  Len       String3,Länge                 ; Länge gilt 2

  Str       Länge,String3                 ; String3
                                          ; enthält "2"

  Val       Position,String1              ; String1
                                          ; enthält "-1"

  AddString "Das Ergebnis ist
            ",String1,String2

  ; String2 enthält "Das Ergebnis ist -1"

Fret
```

10.4.5 Sequentielle / Parallele Ausführung

```
;-------------------------------------------------
; Beispiel einer Routine zur Nullstellung-Steuerung
; an einer Maschine mit 3 Achsen, unter Vermeidung
; von mechanischen Überschneidungen.
;
; Die Nullstellungs der verschiedenen Achsen werden von
; Funktionen ohne Text implementiert.
; Siehe das Beispiel "Nullstellung-Routine".
```

```
;
; Zuerst wird nur die Nullstellung-Vorgang an der
; Z-Achse ausgeführt (von der angenommen wird, daß sie
; nicht gleichzeitig mit den anderen erfolgen kann),
; danach werden gleichzeitig
; die Nullstellungs der Achsen X und Y
; ausgeführt.
;-----------------------------------------------


; Nachricht für den Bediener (in die Landessprache übersetzt)
DefMsg      MSG_SETP    DEU "Nullstellung der Achsen wird ausgeführt ..."
                        ENG "Homing in progress…"

Function NullstellungAchsen

   Message     MSG_SETP           ; informiert den Bediener

   Fcall       NullstellungAchseZ ; Z-Achsen - Nullstellung

   ; OK Z-Achsen - Nullstellung beendet

   StartTask   XAchseNullstellung ; startet Nullstellung X und Y
   StartTask   YAchseNullstellung

   WaitTask    XAchseNullstellung ; wartet Vervollständigung ab
   WaitTask    YAchseNullstellung

   DelMessage  MSG_SETP           ; löscht die Nachricht
                                  ; für den Bediener

Fret
```

10.4.6 Nullstellung-Routine

```
;-------------------------------------------------------------------
;  Beispiel einer Achsen-Nullstellung - Routine
;
;  Die Funktion führt folgenden Vorgänge aus:
;    1) deaktiviert die Software-Grenzen der Achse
;    2) setzt die Suchgeschwindigkeit für den Switch
;    3) bewegt die Achse zu einem inkrementalen Maß,
;       um das Erreichen des Switch sicherzustellen
;    4) wartet ab bis die Achse den Switch auslöst
;    5) stoppt die Achse und wartet das Ende der Bewegung ab
;    6) setzt die Geschwindigkeit (niedrig) zur Freigabe des Switch
;    7) setzt die Achse so weit zurück, daß der Switch
;       freigegeben wird
;    8) wartet die Freigabe des Switch ab
;    9) weist der Achse das neue Maß zu
;   10) aktiviert erneut die Default-Geschwindigkeit und die
;       Software-Grenzen
;
;   © TPA.
;-------------------------------------------------------------------

Function Nullstellung


   ResLi Achse ; deaktiviert die Software-Grenzen
   mPos
   ResLi Achse
   mNeg

   SetVe Achse ; deaktiviert die Software-Grenzen
   l     ,10
```

```
   MovIn Achse ; bewegt die Achse
   c     ,1000
         0
   WaitI Switc ; Switch abwarten
   nput  h,ON

   EndMo Achse ; stoppt die Achse
   v
   waitS Achse ; Achsen-Wartezeit
   till
   SetVe Achse ; setzt die Freigabe-Geschwindigkeit
   l     , 0.1

   MovIn Achse ; bewegt die Achse
   c     ,-100

   WaitI Switc ; Switch-Freigabe abwarten
   nput  h,OFF

   SetQu Achse ; weist das neue Maß zu
   ote   ,0

   SetVe Achse ; Wiederherstellung Geschwindigkeit
   l
   SetLi Achse ; Wiederherstellung Software-Grenzen
   mpos
   SetLi Achse
   mneg

Fret
```

10.4.7 ISO-Bewegungen

```
-----------------------------------------------------------------
; Beispiel einer ISO - Bewegung
;
; Mittels der ISOG0 und ISOG1-Anweisungen wird ein Profil erzeugt
;
;   © TPA.
;-----------------------------------------------------------------*

; Erklärung der ISO - Matrizen
; Matrix der Drehachsen
DrehMx[5] as double:off_X double:off_Y double:off_Z double:dis_X
double:dis_Y double:dis_Z double:delta double:gamma
;  Matrix des Werkzeughalters
WerkzghaltMx[1] as double:off_X double:off_Y double:off_Z double:alpha
double:beta
;  Matrix der Werkzeuge
WerkzgMx[10] as double:ut double

Function ISOInterpolation

   ; Einstellung der Standardwerte von Maschinenparametrisierung
   setval 90.0 MxRot[5].gamma
   setval 260.3 MxUtensili[10].ut
   setval WerkzgMx[10].ut ut

   ; Einstellung der Parameter des 'Algorithmus
   IsosetParam 0 50
   IsosetParam 1 50
   IsosetParam 2 0.9
   IsosetParam 3 60
   IsosetParam 4 30
```

```
    ; Maschineneinstellung: Erklärt die zu der Maschinenparametrisierung
    ; drei gebrauchten Matrizen und die in den ISO-; ; Bewegungen
    gebrauchten
    ; physikalischen Achsen.
    isoG216 DrehMx WerkzghaltMx WerkzgMx 31 X Y Z C B: IMPLIZITE KINEMATIK

    ; Einstellung der Gruppe von Parametern, die die Kinematik der Maschine
    beschreiben
    isoM6 X 5 1 10: IMPLIZITE KINEMATIK

    ; Einstellung der Anfangswerte
    setquote x 500
    setquote y 300
    setquote z 0
    setquote c 0
    setquote b 0
    setvel x
    setvel y
    setvel z
    setvel c
    setvel b
    setveli x y z c b

    ; Profilausführung
    isoG0 1001,X 998.0,Y 600.0,Z 0.0,C 90.0,B 45.0,50.0
    isoG1 1001,X 998.0,Y 600.0,Z 0.0,C 90.0,B 45.0,10000.0
    isoG1 1003,X 996.0,Y 600.0,Z 0.0,C 90.0,B 45.0,10000.0
    isoG1 1002,X 600.0,Y 600.0,Z 0.0,C 90.0,B 45.0,10000.0
    isoG1 1004,X 599.131759111665,Y 599.924038765061,Z 0,C 100,B
    45.0,10000.0
    isoG1 1006,X 598.289899283372,Y 599.69846310393,Z 0,C 110,B 45.0,10000.0
    isoG1 1005,X 597.5,Y 599.330127018922,Z 0,C 120,B 45.0,10000.0
    isoG1 1003,X 596.786061951567,Y 598.830222215595,Z 0,C 130,B
    45.0,10000.0
    isoG1 1002,X 596.169777784405,Y 598.213938048433,Z 0,C 140,B
    45.0,10000.0
    isoG1 1012,X 595.669872981078,Y 597.5,Z 0,C 150,B 45.0,10000.0
    isoG1 1011,X 595.301536896071,Y 596.710100716628,Z 0,C 160,B
    45.0,10000.0
    isoG1 1031,X 595.075961234939,Y 595.868240888335,Z 0,C 170,B
    45.0,10000.0
    isoG1 1102,X 595.0,Y 0.0,Z 0.0,C 180.0,B 45.0,10000.0
    waitstill X Y Z C B
fret
```

Tecnologie e Prodotti per l'Automazione Srl
Via Carducci 221
I - 20099 Sesto S.Giovanni (MI)
Tel. +39 02.36527550
Fax. +39 02.2481008
www.tpaspa.com

March 2021