DOKUMENTATION ISG-kernel

Handbuch
Externe Variablen in der CNC

Kurzbezeichnung:
EXTV

ISG Industrielle Steuerungstechnik GmbH
STEP, Gropiusplatz 10
D-70563 Stuttgart

www.isg-stuttgart.de
support@isg-stuttgart.de

Dokumentation Version: 1.16
08.11.2024

Vorwort

Rechtliche Hinweise

Diese Dokumentation wurde sorgfältig erstellt. Die beschriebenen Produkte und der Funktionsumfang werden jedoch ständig weiterentwickelt. Wir behalten uns das Recht vor, die Dokumentation jederzeit und ohne Ankündigung zu überarbeiten und zu ändern.

Aus den Angaben, Abbildungen und Beschreibungen in dieser Dokumentation können keine Ansprüche auf Änderung bereits gelieferter Produkte geltend gemacht werden.

Qualifikation des Personals

Diese Beschreibung wendet sich ausschließlich an ausgebildetes Fachpersonal der Steuerungs-, Automatisierungs- und Antriebstechnik, das mit den geltenden Normen, der zugehörigen Dokumentation und der Aufgabenstellung vertraut ist.

Zur Installation und Inbetriebnahme ist die Beachtung der Dokumentation, der nachfolgenden Hinweise und Erklärungen unbedingt notwendig. Das Fachpersonal ist verpflichtet, für jede Installation und Inbetriebnahme die zum betreffenden Zeitpunkt veröffentlichte Dokumentation zu verwenden.

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

Weiterführende Informationen

Unter den Links (DE)

https://www.isg-stuttgart.de/produkte/softwareprodukte/isg-kernel/dokumente-und-downloads

bzw. (EN)

https://www.isg-stuttgart.de/en/products/softwareproducts/isg-kernel/documents-and-downloads

finden Sie neben der aktuellen Dokumentation weiterführende Informationen zu Meldungen aus dem NC-Kern, Onlinehilfen, SPS-Bibliotheken, Tools usw.

Haftungsausschluss

Änderungen der Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig.

Marken und Patente

Der Name ISG®, ISG kernel®, ISG virtuos®, ISG dirigent® und entsprechende Logos sind eingetragene und lizenzierte Marken der ISG Industrielle Steuerungstechnik GmbH.

Die Verwendung anderer in dieser Dokumentation enthaltene Marken oder Kennzeichen durch Dritte kann zu einer Verletzung von Rechten der Inhaber der entsprechenden Bezeichnungen führen.

Copyright

Allgemeine- und Sicherheitshinweise

Verwendete Symbole und ihre Bedeutung

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

Symbole im Erklärtext

➢ Gibt eine Aktion an.

⇨ Gibt eine Handlungsanweisung an.

⚠ GEFAHR

Akute Verletzungsgefahr!

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

⚠ VORSICHT

Schädigung von Personen und Maschinen!

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

Achtung

Einschränkung oder Fehler

Dieses Symbol beschreibt Einschränkungen oder warnt vor Fehlern.

Hinweis

Tipps und weitere Hinweise

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

Beispiel

Allgemeines Beispiel

Beispiel zu einem erklärten Sachverhalt.

Programmierbeispiel

NC-Programmierbeispiel

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

Versionshinweis

Spezifischer Versionshinweis

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

Inhaltsverzeichnis

Abbildungsverzeichnis

1 Übersicht EXTV Parameter

Die Übersicht der Parameter der Externen Variablen ist tabellarisch in 4 Spalten sortiert

- In der 1. Spalte steht die eindeutige Kennung der Externen Variable, die sog. "ID". Diese setzt sich aus dem Präfix "P-EXTV" und einer eindeutigen 5-stelligen Nummer zusammen, z.B. P-EXTV-00001.
- In der 2. Spalte ist die Datenstruktur dargestellt, in der der Parameter definiert ist, z.B. var[i].
  Die Struktur dient der Kategorisierung, welche sich folgend im Kapitelaufbau widerspiegelt. Wenn bei 'Struktur' die Angabe fehlt, ist dies kein Fehler; in dem Fall gilt nur der Parameter in Spalte 3 alleine.
- In der 3. Spalte findet sich der "Parameter" mit seiner genauen Bezeichnung, z.B. name
  Wichtig zu erwähnen ist, dass "Struktur"+"Parameter" immer zusammen gehören und exakt so in der Liste der Externen Variablen konfiguriert werden müssen, z.B. var[i].name
- In der 4. Spalte wird die "Funktionalität" in einem zusammenfassenden Begriff/Kurzbeschreibung dargestellt, z.B. Name der externen Variablen.

ID	Struktur	Parameter	Funktionalität/ Kurzbeschreibung
P-EXTV-00001 [▶ 24]	var[i].	name	Name der Externen Variable
P-EXTV-00002 [▶ 24]	var[i].	byte_offset	Position der Externen Variable im Speicher
P-EXTV-00003 [▶ 25]	var[i].	type	Variablentyp
P-EXTV-00004 [▶ 25]	var[i].	scope	Gültigkeitsbereich der Variable
P-EXTV-00005 [▶ 26]	var[i].	synchronisation	Synchronisationsart der Variable
P-EXTV-00006 [▶ 26]	var[i].	access_rights	Zugriffsrecht der Variable
P-EXTV-00007 [▶ 27]	var[i].	array_elements	Anzahl der Elemente in einem Array aus Strukturelementen
P-EXTV-00008 [▶ 27]	var[i].	size	Größe der Variable
P-EXTV-00009 [▶ 28]	var[i].	create_hmi_interface	Freigabe für Zugriff durch HMI
P-EXTV-00010 [▶ 29]		number_used_variables	Anzahl konfigurierter externer Variablen
P-EXTV-00011 [▶ 29]		check_overlapping_variables	Plausibilitätsprüfung des Speicherlayouts
P-EXTV-00012 [▶ 30]		auto_memory_mode	automatisches Speicherlayout
P-EXTV-00013 [▶ 33]		init	Initialisierung mit Standardwerten
P-EXTV-00015 [▶ 20]	struct[i].	name	Name des Variablentyps
P-EXTV-00016 [▶ 21]	struct[i].element[j].	name	Name des Strukturelements
P-EXTV-00017 [▶ 21]	struct[i].element[j].	type	Typ des Strukturelements
P-EXTV-00018 [▶ 22]	struct[i].element[j].	synchronisation	Synchronisationsart des Strukturelements

ID	Struktur	Parameter	Funktionalität/ Kurzbeschreibung
P-EXTV-00019 [▶ 22]	struct[i].element[j].	access_rights	Zugriffsrecht des Strukturelements
P-EXTV-00020 [▶ 22]	struct[i].element[j].	array_elements	Feldgröße eines Strukturelements
P-EXTV-00021 [▶ 23]	struct[i].element[j].	size	Größe eines Strukturelements vom Typ VSTRING
P-EXTV-00022 [▶ 23]		use_exten-ded_string_var	Zeichenanzahl von Stringvariablen
P-EXTV-00047 [▶ 28]	var[i].	suppress_export	Unterdrücken des Variablenexports in die SPS-Beschreibung
P-EXTV-00048 [▶ 28]	var[i].	start	Variablen mit überlappendem Speicherbereich

2 Allgemeine Beschreibung

2.1 Verweise auf andere Dokumente

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

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

2.2 Kommentare in der ASCII-Listendatei

Kommentare können ganzzeilig oder am Ende einer Zeile eingefügt werden.

Bei ganzzeiligem Kommentar muss am Zeilenanfang das Kommentarzeichen "#' gefolgt von einem Leerzeichen eingefügt werden.

Soll am Ende einer Zeile ein Kommentar eingefügt werden, so muss vor dem Kommentar ein Leerzeichen vorhanden sein. Leerzeilen sind ebenfalls möglich.

Beispiel

Kommentare in ASCII-Listendatei

```
#
*************************************************************************
# Daten
#
*************************************************************************
#
# Auflistung Kommentare nach Zahlenwerten

dummy[1] 1 Kommentar
dummy[2] 1 # Kommentar
dummy[3] 1 ( Kommentar
dummy[4] 1 /* Kommentar
...
...
```

Wurde in der Zeile dem Listenparameter jedoch eine Zeichenkette als Wert zugeordnet, so muss ein evtl. nachfolgender Kommentar mit dem Zeichen '(' eröffnet werden. Die Kommentarzeichen Space, # und /* sind nicht zulässig!

Soll eine '(' selbst Bestandteil der Zeichenkette sein, so muss die Zeichenkette in Hochkommas ".." definiert werden (Verfügbar ab V3.1.3081.0, V3.1.3108.0).

```
# Auflistung Kommentare nach Strings

beispiel[0].bezeichnung STRING_1 (Kommentar mit '('Klammer nötig!)

beispiel[1].bezeichnung "STRING_(2)" (Kommentar mit '('Klammer nötig!)
```

3 Funktion und Eigenschaften

Externe Variablen beschreiben einen Datenspeicher, über den beliebige Werte zwischen CNC (NC-Kanal), Oberfläche (GUI) und SPS ausgetauscht werden können. Innerhalb der CNC sind die Daten kanalspezifisch (nur einem Kanal zugänglich) oder kanalübergreifend („global“) nutzbar. Die Bedeutung der einzelnen Speicherstellen (hier eine externe Variable) wird durch die Applikation (NC-Programm, GUI und SPS) bestimmt.

Die CNC selbst hat nur die Aufgabe, das Layout des Speichers festzulegen und dem Anwender im NC-Programm Zugriff auf die Variablen zu gewähren. Während der Konfigurationsphase werden jeder externen Variable ein Name, ein Typ, das Zugriffsrecht (Schreiben und/oder Lesen) zugewiesen. Optional können Speicheradressen und die Größe von Variablen selbst vergeben werden.

Es wird empfohlen die automatisierte Adressvergabe durch die CNC zu verwenden, um Probleme durch Alignment- oder Speicherüberlappungen zu vermeiden. Eine Kombination beider Adressvorgaben ist möglich, wird aber ebenfalls nicht empfohlen.

Festlegen des Speicherlayouts

Die Konfiguration der externen Variablen findet anhand einer ASCII Liste nur einmalig während des Hochlaufs statt.

Neben "einfachen" Variablen können benutzerdefinierte Strukturen festgelegt werden. Zudem sind eindimensionale Arrays von einfachen Variablen oder von Strukturen möglich. Auf sie kann indiziert zugegriffen werden.

Damit die Oberfläche oder eine SPS auf das - letztendlich von der CNC festgelegte - Speicherlayout korrekt zugreifen können, wird eine Beschreibung des Speichers benötigt. Möglich macht dies der Befehl #EXPORT VE. Es wird empfohlen, diesen vor dem erstmaligen Start der SPS auszuführen. Der Befehl erzeugt Deklarations-Dateien, die direkt in die SPS Programmierumgebung integriert werden können. Ein extrem(!) fehleranfälliges manuelles Nachbilden des Speicherlayouts auf Seiten der GUI oder SPS ist damit nicht erforderlich.

Achtung

Sind Adressen und Typen zwischen CNC und SPS bzw. GUI Variablen nicht deckungsgleich, so hat die CNC keine Möglichkeit dies zu überprüfen bzw. ein Fehlverhalten zu verhindern.

Einziger Schutz der CNC vor einem - zum Absturz der Steuerung führenden - ungültigen REAL64 Wert ist die Überprüfung auf "1.#INF", "-1.#INF" und "1.#SNAN" – Muster. Diese Überprüfung findet beim Lesen einer externen Variablen statt und führt im Fehlerfall zur Ausgabe der Meldungen mit der ID 21820 oder ID 21821.

Werte der lokalen Variablen sind über die Laufzeit eines NC-Programms hinweg gültig, d.h. sie werden z.B. beim Start eines neuen NC-Programms nicht abgelöscht.

Zeitpunkte – Schreiben/ Lesen

Für den Zeitpunkt des Schreibens/Lesens einer Variablen aus dem NC-Programm gibt es zwei Möglichkeiten:

- Der Zugriff auf eine Variable erfolgt synchron zur Abarbeitung des NC-Programms im Interpolator, d.h. die zeitliche Sequenz der NC-Befehle und Variablenzugriffe wird sichergestellt.
- Das Schreiben/Lesen einer Variablen wird durchgeführt zum Zeitpunkt der Decodierung (asynchron zur Abarbeitung des NC-Programms im Interpolator, also "vorab").

Die Vor- und Nachteile beider Varianten sind durch den Anwender zu bewerten.

Lesende synchrone Zugriffe führen zum Anhalten des Decoders, bis der synchron gelesene Wert dem Decoder zur Verfügung steht. Außerdem ist das Lesen synchroner Variablen z.B. bei Funktionen wie aktiver Werkzeugradiuskorrektur nicht zulässig, es wird die Meldung mit der ID 20651 ausgegeben.

Der Zugriff der GUI oder SPS auf die Daten erfolgt

- nicht synchronisiert zur NC-Programmbearbeitung. D.h., es kann nicht garantiert werden, wann Daten gelesen oder geschrieben werden. Also wann sie aus Sicht einer Anwendung gültig sind.
- Es kann ebenfalls nicht garantiert werden, dass Daten in Strukturen bzw. Arrays in sich konsistent übertragen werden.

Gewährleistet ist nur der (durch den Prozessor garantierte) korrekte Zugriff auf die Basistypen Byte, Word, Doubleword oder REAL64. Ein Zugriffsschutz hinsichtlich Zeitpunkt und Vollständigkeit muss programmtechnisch durch den Anwender gewährleistet werden.

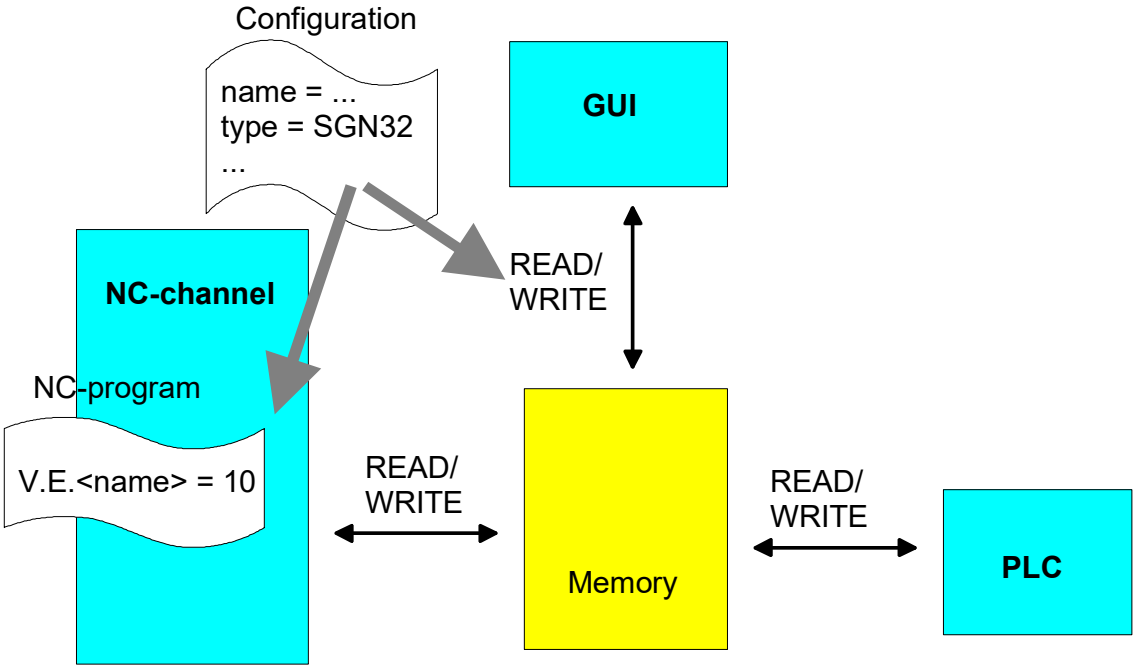

Abb. 1: Verwendung externer Variablen

4 Konfiguration und Initialisierung

4.1 Speichergröße

Der Gesamtspeicher, der für die externen Variablen jedes Kanals zur Verfügung gestellt stehen soll, muss vor dem Hochlauf der Steuerung festgelegt werden.

Bei TwinCAT-Systemen erfolgt die Dimensionierung im System-Manager in der HLI Maske (CNC Task-GEO) unter der Rubrik VE.

Die Dimensionierung des V.E.-Speichers erfolgt über den Parameter P-STUP-00037 (ext_var_max) in der Hochlaufliste.

Die hierbei festgelegte Zahl bestimmt die Anzahl an 24-Byte-Blöcken, aus denen der V.E.-Speicher jedes NC-Kanals besteht. Die globalen Variablen (kanalübergreifend) verwenden einen eigenen Speicherbereich der gleichen Größe. Der reservierte V.E.-Speicher jedes Kanals und der globale Speicher müssen zwingend größer sein, als der durch das Speicherlayout [▶ 12] belegte Speicherbereich.

Reicht der zur Verfügung stehende Speicher für die in der Liste konfigurierten externen Variablen nicht aus, erfolgt die Ausgabe der Fehlermeldung P-ERR-21519. In diesem Fall muss entweder der reservierte Speicher vergrößert werden oder die Größe und Anzahl der konfigurierten externen Variablen muss verringert werden.

4.2 Speicherlayout

Jeder Kanal hat Zugriff auf zwei unterschiedliche Speicherbereiche: Einer davon steht lokal nur für den aktuellen NC-Kanal zur Verfügung, der andere dagegen ist kanalübergreifend, d.h. er kann von allen NC-Kanälen gemeinsam genutzt werden. Die Konfiguration der externen Variablen wird pro NC-Kanal in einer Datei angegeben, in welcher globale und kanalspezifischen Variablen gemeinsam aufgeführt sind.

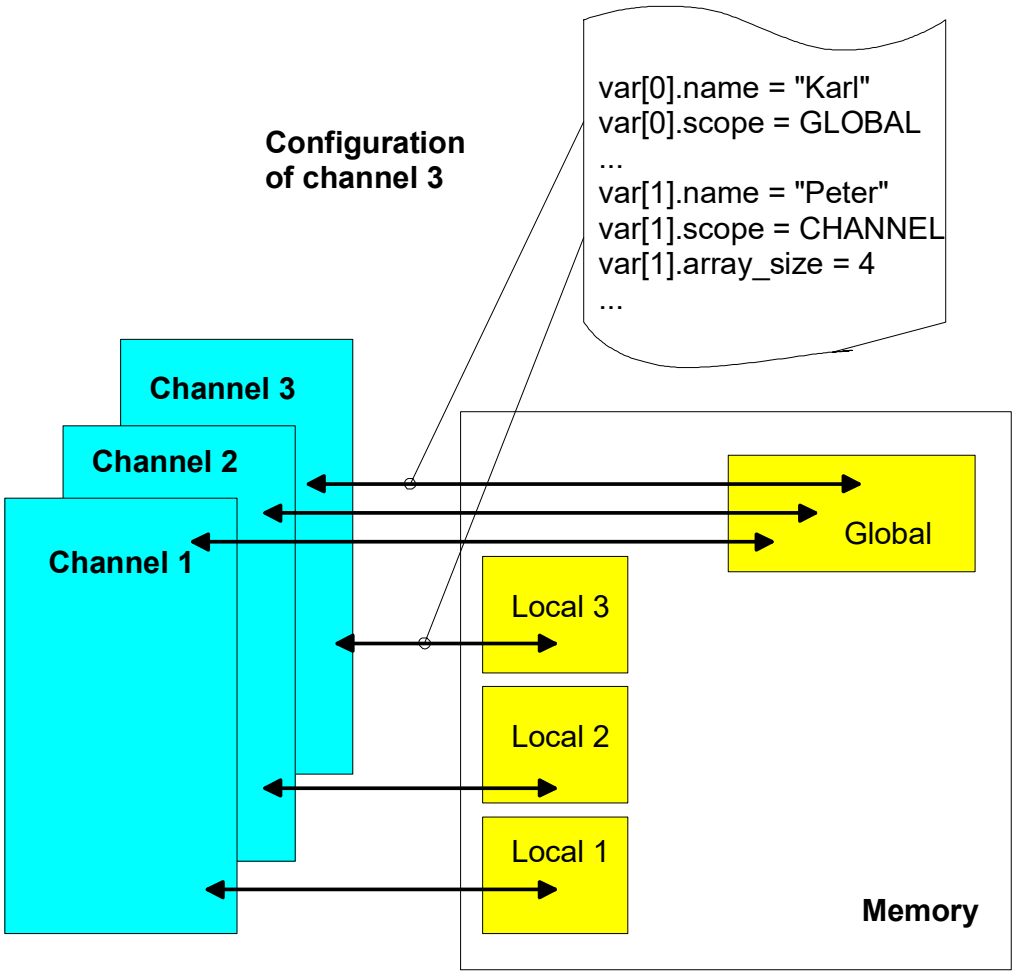

Abb. 2: Gültigkeitsbereich der Speicher

Der Speicherort einer V.E.-Variablen lässt sich manuell festlegen oder wird durch die CNC korrekt hinsichtlich Alignment und Speicherbedarf bestimmt. Im ersten Fall ist der Anwender für die Fehlerfreiheit des Speicherlayouts zuständig. Im anderen Fall ist die Gefahr einer Fehlkonfiguration nicht gegeben.

Bei einer manuellen Konfiguration des Speicherlayouts können V.E.-Variablen durch Angabe eines Adressoffsets (s. P-EXTV-00002 [▶ 24]) an jede beliebige Stelle des V.E.-Speichers gelegt. Mit P-EXTV-00008 [▶ 27] wird das Alignment zu einer im Speicher dahinter liegenden Variablen festgelegt. Bei komplexeren Strukturen kann dies von Bedeutung sein, wenn z. B. die nächste Variable vom Typ REAL64 ist.

Werden benutzerdefinierte Variablentypen genutzt muss in der SPS für Strukturen Byte-Alignment verwendet wird.

Die manuelle Adressvorgabe wird nicht empfohlen, da Speicher fragmentiert und überlappend angelegt sein könnte. Eine Überprüfung hierzu findet nur nach Aufforderung (s. P-EXTV-00011 [▶ 29]) statt, Bei automatischer Adressvergabe werden die Variablen im V.E.-Speicher hingegen nacheinander lückenlos mit korrektem Aligment abgelegt (siehe Abbildung).

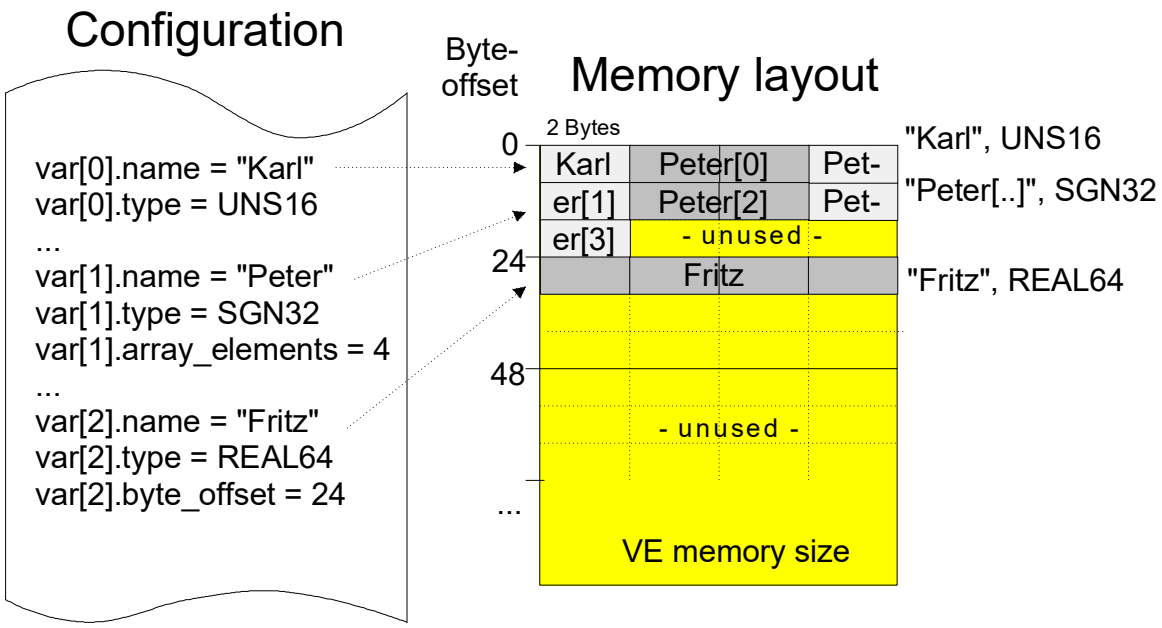

Abb. 3: Speicherlayout resultierend aus gegebener Konfiguration

Achtung

Ab CNC-Version V3.00.3019 werden die externen Variablen und Strukturen mit einem 8-Byte Alignment angelegt. Falls für eine Variable ein fester Startoffset über den Parameter byte_offset (P-EXTV-00002 [▶ 24]) oder index (P-EXTV-00038 [▶ 49]) vorgegeben ist bzw. size (P-EXTV-00008 [▶ 27]) genutzt wird, ist der Anwender für eine korrekte Adressvergabe zuständig, Es findet keine Alignmentberechnung statt.

Hinweis

Kompatibilität alter Konfigurationslisten

Konfigurationslisten nach einer älteren Syntax können auch weiterhin verwendet werden.
Siehe Speicherlayout bis V2.10.1025 [▶ 48]

Hinweis

Bei mehrkanaligen Konfigurationen wird empfohlen in allen Kanälen die gleichen globalen Einträge der V.E.-Variablen zu verwenden.

4.3 Parametrierung des Speicherlayouts von V.E. Variablen

Allgemeine Parameter

ID	Parameter	Bedeutung
P-EXTV-00010 [▶ 29]	number_used_variables	Anzahl konfigurierter externer Variablen
P-EXTV-00011 [▶ 29]	check_overlapping_variables	Plausibilitätsprüfung des Speicherlayouts
P-EXTV-00012 [▶ 30]	auto_memory_mode	automatisches Speicherlayout
P-EXTV-00013 [▶ 33]	init	Initialisierung mit Standardwerten
P-EXTV-00022 [▶ 23]	use_extended_string_var	Zeichenanzahl von Stringvariablen

Parameter für V.E. Strukturen

ID	Parameter	Bedeutung
P-EXTV-00015 [▶ 20]	struct[i].name	Name des Variablentyps
P-EXTV-00016 [▶ 21]	struct[i].element[j].name	Name des Strukturelements
P-EXTV-00017 [▶ 21]	struct[i].element[j].type	Typ des Strukturelements
P-EXTV-00018 [▶ 22]	struct[i].element[j]. synchronisation	Synchronisationsart des Strukturelements
P-EXTV-00019 [▶ 22]	struct[i].element[j]. access_rights	Zugriffsrecht des Strukturelements
P-EXTV-00020 [▶ 22]	struct[i].element[j]. array_elements	Feldgröße eines Strukturelements
P-EXTV-00021 [▶ 23]	struct[i].element[j].size	Größe eines Strukturelements vom Typ VSTRING

V.E.-Parameter einer Variablen

ID	Parameter	Bedeutung
P-EXTV-00001 [▶ 24]	var[i].name	Name der externen Variable
P-EXTV-00002 [▶ 24]	var[i].byte_offset	Position der externen Variable im Speicher
P-EXTV-00003 [▶ 25]	var[i].type	Variablentyp
P-EXTV-00004 [▶ 25]	var[i].scope	Gültigkeitsbereich der Variable
P-EXTV-00005 [▶ 26]	var[i].synchronisation	Synchronisationsart der Variable
P-EXTV-00006 [▶ 26]	var[i].access_rights	Zugriffsrecht der Variable
P-EXTV-00007 [▶ 27]	var[i].array_elements	Anzahl der Elemente in einem Array aus Struktu-relementen
P-EXTV-00008 [▶ 27]	var[i].size	Größe der Variable
P-EXTV-00009 [▶ 28]	var[i].create_hmi_interface	Freigabe für Zugriff durch HMI

ID	Parameter	Bedeutung
P-EXTV-00047 [▶ 28]	var[i].suppress_export	Unterdrücken des Variablenexports in die SPS-Beschreibung
P-EXTV-00048 [▶ 28]	var[i].start	Variablen mit überlappendem Speicherbereich

4.4 Syntax

Bei der Konfiguration der externen Variablen wird im Folgenden zwischen Definition der Variablentypen und der Variableninstanzierung unterschieden.

4.4.1 Kommentare in der ASCII-Listendatei

Kommentare können ganzzeilig oder am Ende einer Zeile eingefügt werden.

Bei ganzzeiligem Kommentar muss am Zeilenanfang das Kommentarzeichen "#' gefolgt von einem Leerzeichen eingefügt werden.

Soll am Ende einer Zeile ein Kommentar eingefügt werden, so muss vor dem Kommentar ein Leerzeichen vorhanden sein. Leerzeilen sind ebenfalls möglich.

Beispiel

Kommentare in ASCII-Listendatei

```
#
*****************************************************************************
# Daten
#
*****************************************************************************
#
# Auflistung Kommentare nach Zahlenwerten

dummy[1] 1 Kommentar
dummy[2] 1 # Kommentar
dummy[3] 1 ( Kommentar
dummy[4] 1 /* Kommentar
...
...
```

Wurde in der Zeile dem Listenparameter jedoch eine Zeichenkette als Wert zugeordnet, so muss ein evtl. nachfolgender Kommentar mit dem Zeichen '(' eröffnet werden. Die Kommentarzeichen Space, # und /* sind nicht zulässig!

Soll eine '(' selbst Bestandteil der Zeichenkette sein, so muss die Zeichenkette in Hochkommas "..." definiert werden (Verfügbar ab V3.1.3081.0, V3.1.3108.0).

```
# Auflistung Kommentare nach Strings

beispiel[0].bezeichnung STRING_1 (Kommentar mit '('Klammer nötig!)

beispiel[1].bezeichnung "STRING_(2)" (Kommentar mit '('Klammer nötig!)
```

4.4.2 Syntax und Interpretation der ASCII-Listendatei

Die in der ASCII-Listendatei enthaltenen Einträge werden von einem Interpreter in die entsprechenden internen Strukturen übernommen und danach auf Plausibilität geprüft. Damit ein sicherer Hochlauf der Steuerung immer gewährleistet ist, werden die bei der Plausibilitätsprüfung festgestellten fehlerhaften Einträge durch Standardwerte ersetzt.

Unbekannte Einträge werden nicht übernommen. Diese Unregelmäßigkeiten werden durch Warnmeldungen angezeigt. Es wird empfohlen, diesen Warnmeldungen nachzugehen und fehlerhafte Einträge in der ASCII-Listendatei zu bereinigen!

Hinweis

Für Daten vom Typ BOOLEAN gilt folgende Vereinbarung:

Wert	Bedeutung
0	Definition von FALSE
1	Definition von TRUE

Hinweis

Für Daten vom Typ STRING gilt folgende Vereinbarung:

Soll einem Listenparameter vom Typ STRING eine Zeichenkette zugewiesen werden, die Zeichen mit einer besonderen Bedeutung in ASCII-Listen enthält (z.B. Kommentarzeichen, Leerzeichen [▶ 17]), so muss diese Zeichenkette in Hochkommas ".." definiert werden (Verfügbar ab V3.1.3081.0, V3.1.3108.0).

```
example[0].name "STRING_WITH_COMMENT( # /*)_CHARACTERS"
```

Abschliessende Leerzeichen werden beim Einlesen verworfen. Der Eintrag..

```
example[0].name "STRING_WITH_POST_SPACES "
```

..ist gleichbedeutend mit

```
example[0].name "STRING_WITH_POST_SPACES"
```

Enthält die Zeichenkette nur Zeichen ohne besondere Bedeutung, sind keine Hochkommas erforderlich.

```
example[0].name STRING_WITH_STANDARD_CHARACTERS!
```

4.4.3 Datensätze zur Definition von Variablentypen (struct[i].*)

Neben den benutzerdefinierten Variablentypen können auch die elementaren Datentypen (BOOLEAN, ..., REAL64) und Zeichenketten vom Typ STRING verwendet werden:

Größe der Standard-Variablentypen

Variablentyp	Größe
BOOLEAN, UNS08, SGN08	1 Byte
UNS16, SGN16	2 Byte
UNS32, SGN32	4 Byte
REAL64	8 Byte
STRING	128 Byte Ab der CNC-Version V2.10.1025.00 können Stringvariablen inklusive Endekennung bis zu 128 Zeichen umfassen. Aus Gründen der Abwärtskompatibilität muss für die Verwendung von Stringvariablen mit mehr als 20 Zeichen der Parameter (P-EXTV-00022 [▶ 23]) use_extended_string_var = TRUE gesetzt werden. In Abhängigkeit des Parameters use_extended_string_var sind Stringvariable auf folgende Zeichenlängen (inklusive Endekennung) begrenzt: • use_extended_string_var = FALSE: 21 Byte (Standard) • use_extended_string_var = TRUE: 128 Byte
VSTRING	String mit variabler Größe zwischen 1 und 800 Byte Ab der CNC-Version V3.1.3039.00 können mit diesem Datentyp Stringvariablen mit individuell festgelegter Länge angelegt werden. Die gewünschte Größe der Zeichenkette inklusive Nullterminierung wird bei Strukturelementen im Parameter P-EXTV-00021 [▶ 23] oder bei Variablen im Parameter P-EXTV-00008 [▶ 27] angegeben. Die Standardgröße des Datentyps VSTRING sind 128 Bytes.

Achtung

Nachträgliche Konfiguration auf 128 Byte große Stringvariablen

Falls in einer bestehenden Konfiguration nachträglich auf 128-Byte große Stringvariablen gewechselt wird (use_extended_string_var [▶ 23] = TRUE), müssen die Adressen der externen Variablen im Speicher (Index, Byte-Offset) und die PLC angepasst werden, damit nachfolgende Variablen nicht überschrieben werden!

Die Definition der Variablentypen erfolgt in der gleichen Konfigurationsliste wie das Festlegen einer Instanz einer externen Variablen.

Im Konfigurationsfile können Kommentare ganzzeilig, oder am Ende einer Zeile eingefügt werden. Bei ganzzeiligem Kommentar muss am Zeilenanfang das Kommentarzeichen "'#", gefolgt von einem Leerzeichen eingefügt werden.

Soll am Ende einer Zeile ein Kommentar eingefügt werden, so muss vor dem Kommentar lediglich ein Leerzeichen vorhanden sein. Wurde in der Zeile jedoch ein String definiert, so muss dem Kommentar das Kommentarzeichen "(" vorangestellt werden.

Leerzeilen sind ebenfalls möglich.

Das Schlüsselwort 'struct' ersetzt das in älteren CNC-Versionen gebräuchliche Token 'type'. Aufgrund der Abwärtskompatibilität wird jedoch 'type' auch weiterhin unterstützt.

Strukturname	Index
struct[i]	i = 0 ... 49 (Maximale Anzahl Datensätze zur Definition externer Variablentypen)

4.4.3.1 Name des Variablentyps (P-EXTV-00015)

P-EXTV-00015	Name des Variablentyps
Beschreibung	Über den Namen wird der Variablentyp identifiziert. Gross- und Kleinbuchstaben werden unterschieden.
Parameter	struct[i].name
Datentyp	STRING
Datenbereich	Maximal 26 Zeichen (Länge Variablenname ab CNC-Version 2.10.1504 *)
Dimension	----
Standardwert	-
Anmerkungen	* Maximal 20 Zeichen vor CNC-Version 2.10.1504

4.4.4 Datensätze zur Definition der Elemente eines Variablentyps (struct[i].element[j].*)

Strukturname	Index
struct[i].element[j]	j = 0 ... 49 (Maximale Anzahl Datensätze zur Definition der Elemente eines Variablentyps)

4.4.4.1 Name des Strukturelements (P-EXTV-00016)

P-EXTV-00016	Name des Strukturelements
Beschreibung	Über den Name wird das Strukturelement identifiziert. Groß- und Kleinbuchstaben werden unterschieden.
Parameter	struct[i].element[j].name
Datentyp	STRING
Datenbereich	Maximal 20 Zeichen
Dimension	----
Standardwert	-
Anmerkungen	

4.4.4.2 Typ des Strukturelements (P-EXTV-00017)

P-EXTV-00017	Typ des Strukturelements
Beschreibung	Die Kennung gibt den Datentyp des Strukturelements an. Neben elementaren Datentypen (SGN08, …,REAL64) kann hier auch der Datentyp STRING oder ein benutzerdefinierter Datentyp angegeben werden.
Parameter	struct[i].element[j].type
Datentyp	STRING
Datenbereich	BOOLEAN, SGN08, UNS08, SGN16, UNS16, SGN32; UNS32, REAL64, STRING, VSTRING oder benutzerdefiniert
Dimension	----
Standardwert	UNS32
Anmerkungen	

4.4.4.3 Synchronisationsart des Strukturelements (P-EXTV-00018)

P-EXTV-00018	Synchronisationsart des Strukturelements
Beschreibung	Standardmäßig erben alle Strukturelemente die Synchronisationsart für den Schreib-/Lesezugriff (siehe Synchronisation des Zugriffs durch NC-Kanal [▶ 40]) von der Variableninstanzierung. Mit diesem Parameter kann die Synchronisationsart für das Strukturelement individuell festgelegt werden.
Parameter	struct[i].element[j].synchronisation
Datentyp	BOOLEAN
Datenbereich	TRUE, FALSE
Dimension	----
Standardwert	TRUE *
Anmerkungen	* Die Synchronisationsart wird von der Variableninstanziierung geerbt (siehe Datensätze zur Definition externer Variablen (var[i].*) [▶ 24])

4.4.4.4 Zugriffsrecht des Strukturelements (P-EXTV-00019)

P-EXTV-00019	Zugriffsrecht des Strukturelements
Beschreibung	Standardmäßig erben alle Strukturelemente die Zugriffsrechte von der Variableninstanziierung. Mit diesem Parameter kann das Zugriffsrecht für das Strukturelement individuell festgelegt werden.
Parameter	struct[i].element[j].access_rights
Datentyp	STRING
Datenbereich	READ_WRITE, READ_ONLY, WRITE_ONLY, INHERIT_ACCESS
Dimension	----
Standardwert	INHERIT_ACCESS
Anmerkungen	Das Zugriffsrecht wird von der Variableninstanziierung geerbt (siehe Datensätze zur Definition externer Variablen (var[i].*) [▶ 24])

4.4.4.5 Arraygröße des Strukturelements (P-EXTV-00020)

P-EXTV-00020	Anzahl der Elemente in einem Array aus Strukturelementen
Beschreibung	Ist das Strukturelement ein Array aus Strukturelementen, so muss die Anzahl der Elemente angegeben werden. Ist die Größe 0, so ist das Strukturelement eine einzelne Variable und kein Array.
Parameter	struct[i].element[j].array_elements
Datentyp	UNS16
Datenbereich	1 ... MAX(UNS16)
Dimension	----
Standardwert	0
Anmerkungen	

4.4.4.6 Größe eines Strukturelements vom Typ VSTRING (P-EXTV-00021)

P-EXTV-00021	Größe eines Strukturelements vom Typ VSTRING (Datentyp mit variabler Stringlänge)
Beschreibung	Ab CNC-Version V3.1.3039.00 kann für die Definition der externen Variablen ein neuer Datentyp VSTRING verwendet werden. Für diesen Datentyp kann die Größe des String-Elements individuell festgelegt werden kann. Die gewünschte Größe wird in diesem Parameter inklusive Nullterminierung angegeben, d.h. bei einem Wert size = 128 stehen 127 Nutzzeichen in dem Element zur Verfügung.
Parameter	struct[i].element[j].size
Datentyp	UNS32
Datenbereich	1 ... 800 Byte
Dimension	Byte
Standardwert	128 Byte (127 Nutzzeichen mit Null-Terminierung)
Anmerkungen	Der Parameter steht ab CNC-Version V3.1.3039.00 zur Verfügung. Er wird nur für die Größenangabe für Stringelemente vom Typ VSTRING verwendet und hat für die anderen Datentypen keine Bedeutung!

4.4.5 Festlegung der Zeichenanzahl von Stringvariablen (P-EXTV-00022)

P-EXTV-00022	Zeichenanzahl von Stringvariablen
Beschreibung	Mit dem Parameter kann die zulässige Zeichenanzahl von Stringvariablen von 21 auf 128 Zeichen (jeweils inklusive Endmarke) erhöht werden. Falls die Adressen der V.E. Variablen in 24-Byte Blöcken (siehe Speicherlayout [▶ 12]) vorgegeben sind, ist bei den 128 Byte großen Variablen vom Typ STRING zu beachten, dass sie im Speicherlayout mehrere 24-Byte Blöcke belegen und der Index entsprechend hochgezählt (vergl. Variablenarrays) werden muss.
Parameter	use_extended_string_var
Datentyp	BOOLEAN
Datenbereich	TRUE, FALSE
Dimension	----
Standardwert	FALSE
Anmerkungen	

4.4.6 Datensätze zur Definition externer Variablen (var[i].*)

Strukturname	Index
var[i]	i = 0 ... 214 (Maximale Anzahl Datensätze zur Definition der Variablen)

4.4.6.1 Variablenname (P-EXTV-00001)

P-EXTV-00001	Name der externen Variable
Beschreibung	Über den Namen wird die Variable identifiziert. Aus diesem wird mit dem Präfix „V.E.“ ein Gesamtname für den NC-Kanal und die Oberfläche zusammengesetzt (z.B. **V.E.<name>**). Dieser Gesamtname wird dann im NC-Programm bei der Variablenprogrammierung verwendet. Groß- und Kleinbuchstaben werden unterschieden.
Parameter	var[i].name
Datentyp	STRING
Datenbereich	Maximal 26 Zeichen (Länge Variablenname ab CNC-Version 2.10.1504 *)
Dimension	----
Standardwert	-
Anmerkungen	* Maximal 20 Zeichen vor CNC-Version 2.10.1504

4.4.6.2 Byteoffset (P-EXTV-00002)

P-EXTV-00002	Position der externen Variable im Speicher
Beschreibung	Abweichend von dem implizit durch „type“ (P-EXTV-00003 [▶ 25]) bzw. der Größe einer Struktur festgelegten Datengröße (siehe Datensätze zur Definition der Elemente eines Variablentyps (struct[i].element[j].*) [▶ 21]) kann die Position einer Variablen im Speicher festgelegt werden. Die Angabe dieses Parameters ist optional. Alle Variablen mit byte_offset = -1 werden im V.E.-Speicher lückenlos hintereinander (beginnend mit Offset 0) angereiht.
Parameter	var[i].byte_offset
Datentyp	SGN32
Datenbereich	0 ... MAX(SGN32)
Dimension	----
Standardwert	-1
Anmerkungen	**Es wird empfohlen, die Adressberechnungen der Steuerung zu überlassen und den Parameter nicht einzusetzen. Eventuelle Speicherüberlappungen durch eine Fehlkonfiguration werden nur mit dem Paramater** P-EXTV-00011 [▶ 29] **erkannt!**

4.4.6.3 Variablentyp (P-EXTV-00003)

P-EXTV-00003	Typ der externen Variable
Beschreibung	Die Kennung gibt den Datentyp der Variable an. Neben elementaren Datentypen (SGN08, …, REAL64) und dem Datentyp STRING kann hier auch ein benutzerdefinierter Variablentyp angegeben werden (P-EXTV-00015 [▶ 20]).
Parameter	var[i].type
Datentyp	STRING
Datenbereich	BOOLEAN, SGN08, UNS08, SGN16, UNS16, SGN32; UNS32, REAL64, STRING, VSTRING oder benutzerdefiniert
Dimension	----
Standardwert	UNS32
Anmerkungen	Bei einer automatischen Adressvergabe bestimmt der Parameter implizit die Adresse der **nächsten** Variablen wobei Rechnerarchitektur und in der Steuerung hinterlegte Alignmentstrategien berücksichtigt werden. Der lesende oder schreibende Zugriff auf eine Variable erfolgt **unabhängig von einer manuellen oder automatischen Adressvergabe** immer mit dem hinterlegten Typ.

4.4.6.4 Gültigkeitsbereich (P-EXTV-00004)

P-EXTV-00004	Gültigkeitsbereich der externen Variable
Beschreibung	Beim Gültigkeitsbereich wird zwischen einem kanalspezifischen und einem kanalübergreifenden, globalen unterschieden.
Parameter	var[i].scope
Datentyp	STRING
Datenbereich	GLOBAL, CHANNEL
Dimension	----
Standardwert	CHANNEL
Anmerkungen	

4.4.6.5 Synchronisationsart (P-EXTV-00005)

P-EXTV-00005	Synchronisationsart der externen Variable
Beschreibung	Der Schreib-/Lesezugriff erfolgt normalerweise synchron zur Bearbeitung. In Einzelfällen kann diese implizite Synchronisierung unterdrückt werden (siehe Synchronisation des Zugriffs durch NC-Kanal [▶ 40]). Falls es sich bei der Variable um eine Variablenstruktur handelt, wird die Synchronisationsart an alle untergeordneten Strukturelemente vererbt. Zusätzlich kann bei der Typdefinition die Synchronisationsart für jedes Strukturelement individuell festgelegt werden (P-EXTV-00018 [▶ 22]).
Parameter	var[i].synchronisation
Datentyp	BOOLEAN
Datenbereich	TRUE, FALSE
Dimension	----
Standardwert	TRUE
Anmerkungen	Synchrone Variablen führen beim Lesen immer zum Anhalten des Decoders bis der synchron gelesene Wert dem Decoder zur Verfügung steht. Außerdem ist das Lesen synchroner Variablen z.B. bei Funktionen wie aktiver Werkzeugradiuskorrektur nicht zulässig, es wird die Meldung mit der ID 20651 ausgegeben.

4.4.6.6 Zugriffsrecht (P-EXTV-00006)

P-EXTV-00006	Zugriffsrecht der externen Variable
Beschreibung	In der Grundeinstellung ist ein Schreib-/Lesezugriff auf die Variablen möglich, welcher über den Zugriffsschutz eingeschränkt werden kann. Falls es sich bei der Variable um eine Variablenstruktur handelt, wird das Zugriffsrecht an alle untergeordneten Strukturelemente vererbt. Zusätzlich kann bei der Typdefinition das Zugriffsrecht für jedes Strukturelement individuell festgelegt werden (P-EXTV-00019 [▶ 22]).
Parameter	var[i].access_rights
Datentyp	STRING
Datenbereich	READ_WRITE, READ_ONLY, WRITE_ONLY
Dimension	----
Standardwert	READ_WRITE
Anmerkungen	Das Zugriffsrecht gilt nur für die CNC und nicht für die SPS. Eine Variable, die mit dem Zugriffsrecht READ_ONLY definiert ist kann im NC-Programm nur gelesen werden. In der SPS kann auf diese Variable geschrieben werden.

4.4.6.7 Arraygröße (P-EXTV-00007)

P-EXTV-00007	Anzahl der Elemente in einem Array der Variablen
Beschreibung	Wird eine externe Variable nicht nur einmal sondern als Array dieser Variable benötigt, so ist die Anzahl der Feldelemente festzulegen. Ist die Variable kein Array, dann ist 0 anzugeben.
Parameter	var[i].array_elements
Datentyp	UNS16
Datenbereich	1 ... MAX(UNS16)
Dimension	----
Standardwert	0
Anmerkungen	

4.4.6.8 Variablengröße (P-EXTV-00008)

P-EXTV-00008	Größe der externen Variable
Beschreibung	Abweichend von dem implizit durch „type“ (P-EXTV-00003 [▶ 25]) bzw. der Größe einer Struktur festgelegten Datengröße siehe Datensätze zur Definition der Elemente eines Variablentyps (struct[i].element[j].*) [▶ 21]) kann der Offset und damit die **Anfangsposition der im Speicher nachfolgenden** globalen Variablen „verschoben“ werden. Die Angabe der Variablengröße ist nur notwendig, falls zusätzliche Alignmentbytes berücksichtigt werden müssen. Bei Arrays gibt der Parameter die Größe eines Einzelelements an. Bei Variablen vom Typ VSTRING wird in diesem Parameter die Größe für die Stringvariable (inklusive Nullterminierung) festgelegt. Die Größe kann für jede Variable unterschiedlich gewählt werden. Standardmäßig wird die Variable mit einer Größe von 128 Byte angelegt.
Parameter	var[i].size
Datentyp	UNS32
Datenbereich	0 ... MAX(UNS32)
Dimension	----
Standardwert	0
Anmerkungen	Bei einem Wert von 0 wird der Wert dieses Parameters aus dem in P-EXTV-00003 [▶ 25] eingestellten Datentyp abgeleitet. **Achtung**: Ist der Wert kleiner als der tatsächlich benötigte Speicher, so wird die Variable von der nachfolgenden Variablen überschrieben.

4.4.6.9 HMI-Zugriffsfreigabe (P-EXTV-00009)

P-EXTV-00009	Freigabe für Zugriff durch HMI
Beschreibung	Für jede Variable kann zusätzlich der Zugriff über die Oberfläche durch ein entsprechendes Kommunikationsobjekt ermöglicht werden, falls das Flag gesetzt ist.
Parameter	var[i].create_hmi_interface
Datentyp	BOOLEAN
Datenbereich	TRUE, FALSE
Dimension	----
Standardwert	FALSE
Anmerkungen	Für Variablen mit benutzerdefiniertem Datentyp hat dieser Parameter keine Auswirkung.

4.4.6.10 Variablenexport in PLC-Beschreibung (P-EXTV-00047)

P-EXTV-00047	Unterdrücken des Variablenexports in die PLC-Beschreibung
Beschreibung	Mit Hilfe des NC-Befehls #EXPORT VE (s. [FCT-C22]) kann die CNC-Beschreibung der externen Variablen in eine äquivalente PLC-Beschreibung exportiert werden. Durch Setzen dieses Parameters auf den Wert TRUE kann der Export für diese Variable unterdrückt werden. In der PLC-Beschreibung werden dann entsprechende Alignment-Bytes vorgesehen.
Parameter	var[i].suppress_export
Datentyp	BOOLEAN
Datenbereich	TRUE, FALSE
Dimension	----
Standardwert	FALSE
Anmerkungen	Dieser Parameter ist ab den CNC-Versionen V.2.11.2027.01, V.2.11.2807.18 bzw. V3.1.3052.01 verfügbar.

4.4.6.11 Überlappende Variablen (P-EXTV-00048)

P-EXTV-00048	Variablen mit überlappendem Speicherbereich
Beschreibung	In diesem Parameter kann eine bereits definierte externe Variable, ein Struktur- oder Array-Element angegeben werden. Die aktuelle Variable wird dann mit der Adresse der hier angegebenen Variable bzw. des Struktur- oder Arrayelements angelegt, so dass sich die Variablen im Speicher überlappen.
Parameter	var[i].start
Datentyp	STRING
Datenbereich	Beliebiger, bereits definierter Variablenname
Dimension	----
Standardwert	-
Anmerkungen	Dieser Parameter steht unter TwinCAT nicht zur Verfügung.

4.4.7 Anzahl konfigurierter externer Variablen (P-EXTV-00010)

P-EXTV-00010	**Anzahl konfigurierter externer Variablen**
Beschreibung	Bei lückenloser Belegung von *var[i].** wird hier der Index der zuletzt definierten Variable +1 eingetragen (i_{last}+1). Bei Belegung mit Lücken von *var[i].** wird hier der höchste Index der definierten Variablen +1 eingetragen (i_{max}+1). Ist der Wert kleiner als die tatsächlich konfigurierte Variablenanzahl, so sind nach Hochlauf der Steuerung auch nur die Variablen bis zu diesem Wert verfügbar.
Parameter	number_used_variables
Datentyp	UNS16
Datenbereich	0 ... MAX(UNS16)
Dimension	----
Standardwert	0
Anmerkungen	*anzahl_belegt (Alte Syntax bis V2.11.2034.0)*

4.4.8 Plausibilitätsprüfung des Speicherlayouts (P-EXTV-00011)

P-EXTV-00011	**Plausibilitätsprüfung des Speicherlayouts**
Beschreibung	Mit Hilfe des Parameters var[i].byte_offset P-EXTV-00002 [▶ 24] können externe Variablen an eine beliebige, evtl. falsche, Speicheradresse gelegt werden. Mit diesem Parameter kann eine Plausibilitätsprüfung des Speicherlayouts der externen Variablen aktiviert werden. Falls Variablen sich im Speicher überlappen, gibt die CNC im Steuerungshochlauf die Fehlermeldung P-ERR-21848 aus und die überlappende Variable wird gelöscht.
Parameter	check_overlapping_variables
Datentyp	BOOLEAN
Datenbereich	TRUE, FALSE
Dimension	----
Standardwert	FALSE
Anmerkungen	Dieser Parameter ist ab den CNC-Versionen V.2.11.2027.01, V.2.11.2807.18 bzw. V3.1.3052.01 verfügbar.

4.4.9 Methode für automatisches Speicherlayout (P-EXTV-00012)

P-EXTV-00012	Methode für automatisches Speicherlayout
Beschreibung	Ab CNC-Version V2.10.1025.00 werden die externen Variablen von der CNC automatisch hintereinander im Speicher ohne Lücken angelegt. Mit Hilfe des Parameters var[i].byte_offset P-EXTV-00002 [▶ 24] bzw. durch Angabe eines 24-Byte Bereichs var[i].index P-EXTV-00038 können externe Variablen auch an eine beliebige Speicheradresse gelegt werden. Falls die automatische und die manuelle Adressvergabe kombiniert werden, kann in diesem Parameter festgelegt werden, wie die CNC die automatischen Adressen der Variablen vergibt.
Parameter	auto_memory_mode
Datentyp	STRING
Datenbereich	START_VE_MEMORY: Alle Variablen mit automatisch vergebener Adresse werden lückenlos an den Anfangsbereich des externen Variablenspeichers angelegt. LAST_USED_ADDRESS: Die Variable mit automatisch vergebener Adresse wird immer nach dem letzten, durch die vorangegangenen Variablen belegten Speicherbereich angelegt.
Dimension	----
Standardwert	START_VE_MEMORY
Anmerkungen	Dieser Parameter ist ab den CNC-Versionen V.2.11.2027.01, V.2.11.2807.18 bzw. V3.1.3052.01 verfügbar. Für ältere CNC-Versionen ist die START_VE_MEMORY Einstellung wirksam.

Beispiel für auto_memory_mode = START_VE_MEMORY:

```
auto_memory_mode START_VE_MEMORY

var[0].name                         var_1
var[0].type                         SGN32
var[0].scope                        GLOBAL
var[0].synchronisation              FALSE
var[0].access_rights                READ_WRITE
#
var[1].name                         var_2
var[1].type                         REAL64
var[1].scope                        GLOBAL
var[1].synchronisation              TRUE
var[1].access_rights                READ_WRITE
var[1].byte_offset                  16
#
var[2].name                         var_3
var[2].type                         SGN32
var[2].scope                        GLOBAL
var[2].synchronisation              TRUE
var[2].access_rights                READ_WRITE
```

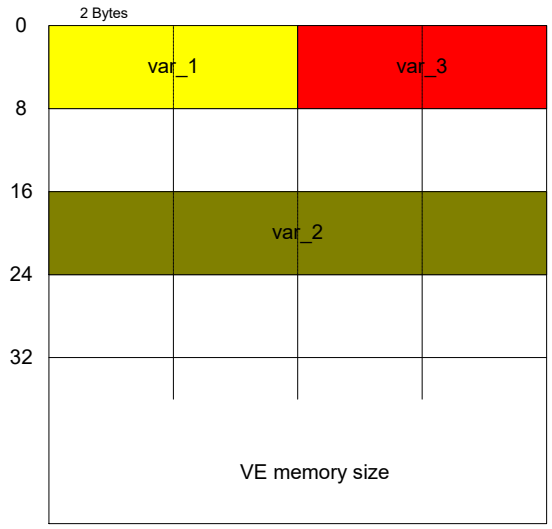

Abb. 4: Resultierendes Speicherlayout

Beispiel für auto_memory_mode = LAST_USED_ADDRESS:

```
auto_memory_mode LAST_USED_ADDRESS

var[0].name                         var_1
var[0].type                         SGN32
var[0].scope                        GLOBAL
var[0].synchronisation              FALSE
var[0].access_rights                READ_WRITE
#
var[1].name                         var_2
var[1].type                         REAL64
var[1].scope                        GLOBAL
var[1].synchronisation              TRUE
var[1].access_rights                READ_WRITE
var[1].byte_offset                  16
#
var[2].name                         var_3
var[2].type                         SGN32
var[2].scope                        GLOBAL
var[2].synchronisation              TRUE
var[2].access_rights                READ_WRITE
```

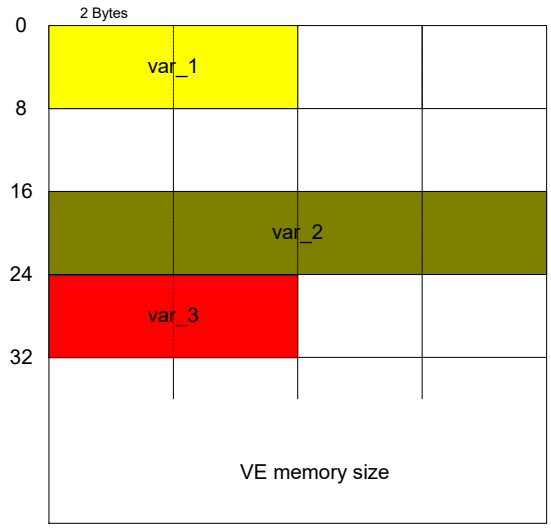

Abb. 5: Resultierendes Speicherlayout

4.4.10 Initialisierung bei Steuerungsstart (P-EXTV-00013)

P-EXTV-00013	**Initialisierung der externen Variablen mit Standardwerten.**
Beschreibung	Mithilfe dieses Parameters können der externen Variablen bei Steuerungsstart Initialwerte zugewiesen werden. Dieser Parameter wird von der Steuerung nur bei Steuerungsstart ausgewertet, nachdem das Speicherlayout der externen Variablen festgelegt wurde. Hinter dem Schlüsselwort init erwartet die CNC einen NC-Programm konformen Syntaxstring mit einer Wertzuweisung einer externen Variablen (s. [PROG//13-Variablen und Variablenrechnung]). Das Schlüsselwort init darf mehrfach in der Konfigurationsdatei der externen Variable vorkommen. Die Initialanweisungen werden in der Reihenfolge, wie sie in der Konfigurationsdatei stehen, abgearbeitet. Für die Wertzuweisung können arithmetische Operationen, String-Operationen und andere externe Variablen verwendet werden. Weitere Decodervariablen oder Anweisungen sind nicht zulässig.
Parameter	init
Datentyp	STRING
Datenbereich	NC-Programm konformer Syntaxstring mit Wertzuweisung an externe Variable
Dimension	----
Standardwert	-
Anmerkungen	Dieser Parameter ist ab den CNC-Versionen V.2.11.2027.01, V.2.11.2807.18 bzw. V3.1.3052.01 verfügbar.

Achtung

Die Initialisierungswerte der externen Variablen P-EXTV-00013 werden nur beim Steuerungsstart ausgewertet. Beim Nachladen einer externen Variablenliste werden die Werte nicht übernommen!

Beispiel für die Initialisierung von ext. Variablen während dem Steuerungshochlauf:

```
init      V.E.var1_real64 = 1234.5
init      V.E.var1_sgn32 = ROUND[1 + 10 / 3]
init      V.E.var1_string = "Hello" + " world!"
init      V.E.var2_real64 = 2.0 * V.E.var1_real64

init      V.E.arr_sgn32[0] = 1
init      V.E.arr_sgn32[1] = 2

init      V.E.vector.x = 10.0
init      V.E.vector.y = 20.0
init      V.E.vector.z = 30.0
```

4.4.11 Beispiel einer Konfigurationsliste

Beispiele für Typdefinitionen:

```
use_extended_string_var                TRUE #Strings mit 128 Zeichen

struct[0].name                         VECTOR_T
struct[0].element[0].name              x
struct[0].element[0].type              REAL64
struct[0].element[1].name              y
struct[0].element[1].type              REAL64
struct[0].element[2].name              z
struct[0].element[2].type              REAL64
#
struct[1].name                         TARGET_POINT_T
struct[1].element[0].name              point
struct[1].element[0].type              VECTOR_T
struct[1].element[1].name              valid
struct[1].element[1].type              BOOLEAN
struct[1].element[1].access_rights     READ_ONLY
struct[1].element[1].synchronisation   TRUE
#
struct[2].name                         TRAJEKTORIE_T
struct[2].element[0].name              nbr_points
struct[2].element[0].type              SGN32
struct[2].element[1].name              name
struct[2].element[1].type              STRING
struct[2].element[2].name              points
struct[2].element[2].type              TARGET_POINT_T
struct[2].element[2].array_elements    10
```

Beispiele für Variablendefinitionen:

```
number_used_variables                5
#
var[0].name                          var_global_1
var[0].type                          UNS32
var[0].scope                         GLOBAL
var[0].synchronisation               FALSE
var[0].access_rights                 READ_WRITE
#
var[1].name                          var_chan_1
var[1].type                          SGN32
var[1].scope                         CHANNEL
var[1].synchronisation               TRUE
var[1].access_rights                 READ_WRITE
#
var[2].name                          array_chan_1
var[2].type                          SGN16
var[2].scope                         CHANNEL
var[2].synchronisation               TRUE
var[2].access_rights                 READ_WRITE
var[2].array_elements                20
#
var[3].name                          var_chan_2
var[3].type                          STRING
var[3].scope                         CHANNEL
var[3].synchronisation               TRUE
var[3].access_rights                 READ_WRITE
#
var[4].name                          trajektorie
var[4].type                          TRAJEKTORIE_T
var[4].scope                         CHANNEL
var[4].synchronisation               FALSE
var[4].access_rights                 READ_WRITE
```

4.4.12 Beispiel zu V.E.-Strukturen

Im Beispiel soll in einer V.E.-Liste die Handhabung von V.E.-Strukturen erläutert werden.

Folgende Aufgabe:

Ein Positionsverlauf wird mit einem Namen bezeichnet und hat eine definierte Anzahl von Positionen.

Jede dieser Positionen besteht aus X, Y, Z und einer Kennung für die Gültigkeit.

Es sind 5 unterschiedliche Verläufe möglich, jede dieser Verläufe hat maximal 12 Punkte

Struktur Kurve:

- Position
- Name

Struktur Position

- X
- Y
- Z
- Gültigkennung

```
#*********************************************
# TC_CHANNEL_DESC_5: Externe Variablen
#*********************************************
use_extended_string_var                          1
# ----Definition der Strukturen -------
# -------Struktur Positionsverlauf -------
#
struct[0].name                                    typcurve
struct[0].element[0].name                         point
struct[0].element[0].type                         typ_pos
struct[0].element[0].array_elements               12
struct[0].element[1].name                         curve_name
struct[0].element[1].type                         STRING

#------- Struktur raumliche Position -------
struct[1].name                                    typ_pos
struct[1].element[0].name                         X
struct[1].element[0].type                         REAL64
struct[1].element[1].name                         Y
struct[1].element[1].type                         REAL64
struct[1].element[2].name                         Z
struct[1].element[2].type                         REAL64
struct[1].element[3].name                         pos_is_valid
struct[1].element[3].type                         BOOLEAN
#
#------- Variablen -------
number_used_variables                            1
#
var[0].name                                      curve
var[0].type                                      typcurve
var[0].scope                                     GLOBAL
var[0].synchronisation                           FALSE
var[0].access_rights                             READ_WRITE
var[0].array_size                                5
#
Ende
```

Hinweis

Die Eingabe von Strukturen und Variablen erfolgt Case Sensitive.

Wird beim Datentyp anstelle von STRING der Typ mit String angegeben, dann wird der Fehler P-ERR-21441 ausgegeben.

Die verwendbaren Datentypen sind im Parameter P-EXTV-00003 [▶ 25] aufgelistet.

Hinweis

Nur verwendete Strukturen werden auf syntaktische Korrektheit geprüft.

Die Prüfung erfolgt beim Steuerungshochlauf.

(Belegung eines Punktes aus dem NC-Programm

%Setpoint.nc

N020 V.E.curve[0].point[2].X=11

N030 V.E.curve[0].point[2].Y=22

N040 V.E.curve[0].point[2].Z=33

N080 M30

4.5 Einbindung in NC-Hochlauf

Die Konfiguration der externen Variablen wird jedem NC-Kanal getrennt über eine ASCII-Liste bekannt gegeben. Für jeden Kanal wird in der zentralen Hochlaufbeschreibung des Systems ein Dateinamen angegeben, welcher die Konfiguration der externen Variablen diesen Kanals festlegt.

Beispiel: Auszug aus der Hochlaufbeschreibung für zwei Kanäle

```
# ----------------------------------------
# Konfigurierungsdaten Listen
# ----------------------------------------
#
Listen                              ASCII
default_sda_mds                     ..\listen\default_sda.lis
hand_mds                            ..\listen\hand_mds.lis
rtconf_lis                          ..\listen\rtconf.lis
#
sda_mds[0]                          ..\listen\sda_mds1.lis
werkz_data[0]                       ..\listen\werkz_d1.lis
nullp_data[0]                       ..\listen\nullp_d1.lis
pzv_data[0]                         ..\listen\pzv_d1.lis
ve_var[0]                           ..\listen\ext_var1.lis
hmi[0].objects                      default
channel[0].objects                  default
#
sda_mds[1]                          ..\listen\sda_mds2.lis
werkz_data[1]                       ..\listen\werkz_d2.lis
nullp_data[1]                       ..\listen\nullp_d2.lis
pzv_data[1]                         ..\listen\pzv_d2.lis
ve_var[1]                           ..\listen\ext_var2.lis
hmi[1].objects                      default
channel[1].objects                  default
```

5 Anwendung und Zugriff auf Variablen

5.1 NC-Programm

Der Zugriff des NC-Kanals auf die externen Variablen erfolgt aufgrund der Schreib-/Leseanweisung auf die Variable **V.E.<name>** im NC-Programm. Die im NC-Programm zur Verfügung stehenden Variablen setzen sich aus dem Präfix **V.E.** und dem in der Konfigurationsliste der Variablen angegeben Namen **<name>** zusammen. **V.E.** Variablen dürfen maximal aus 20 Zeichen bestehen.

Programmierbeispiel

VE-Variablenzugriff in der CNC

```
N100 $IF V.E.CHANNEL_WR >= 100     (Entspr. dem Wert von V.E.CHANNEL_WR)
                                   (wird in die verschiedenen Fälle)
                                   (verzweigt.)
N110 G01 X100 Y100 F1000

N120 $ELSE
N130 G01 X100 YV.E.CHANNEL_WR F1000   (Geradeninterpolation in)
                                      (Y-Richtung mit dem Wert)
                                      (von CHANNEL_WR)
N140 $ENDIF

N150 V.E.GLOBAL_SWR = V.A.ABS.X    (Der externen Variablen wird die)
                                   (absolute X-Koordinate zugewiesen)
N160 G01 XV.E.GLOBAL_SWR           (Geradeninterpolation in X-Richtung)
                                   (mit dem Wert von V.E.GLOBAL_SWR)
```

Programmierbeispiel

VE-Variablenzugriff in der CNC ab Version V2.10.1025.00

```
N010 $IF V.E.trajektorie.name != „"
N020   V.E.name = V.E.trajektorie.name
N030   P1 = 0
N040   $WHILE P1 < V.E.trajektorie.nbr_points
N050     $IF V.E.trajektorie.points[P1].valid == TRUE
N060       G0 X = V.E.trajektorie.points[P1].point.x
              Y = V.E.trajektorie.points[P1].point.y
              Z = V.E.trajektorie.points[P1].point.z
N070     $ENDIF
N080     P1 += 1
N090   $ENDWHILE
N100 $ENDIF
N110 V.E.name = „"
N120 M30
```

5.1.1 Synchronisation des Zugriffs durch NC-Kanal

Bei einem lesenden oder schreibenden synchronen Variablenzugriff erwartet der Anwender eventuell eine zeitliche Sequenz, wie im NC-Programm angegeben. Dieses Verhalten kann durch die Synchronisationsart der Variablen festgelegt werden.

Ein asynchroner Lese-/Schreibzugriff (siehe Bild, Fall b)) wird unmittelbar während der Dekodierung des NC-Programms durchgeführt.

Da die NC-Programmdekodierung in einer planerischen Phase zeitlich vor der eigentlichen Abarbeitung der NC-Befehle durchgeführt wird, darf ein synchronen Variablenzugriff dann nicht einfach während der Programmdekodierung durchgeführt werden, sondern muss zur aktuellen Programmabarbeitung (Achsbewegung) synchron erfolgen. Ein synchroner Zugriff wird folgendermaßen sichergestellt:

Synchrones LESEN (siehe Bild, Fall a))**:** Beim Lesen wird die Dekodierung solange angehalten, bis die aktuelle Programmbearbeitung an der zuletzt dekodierten NC-Programmzeile angelangt ist (implizites #FLUSH WAIT [PROG// NC-Kanal leeren]). Der Wert wird dann gelesen und dem Decoder zur Verfügung gestellt. Dann erst wird die Decodierung fortgesetzt. Da ein #FLUSH WAIT während bestimmter satzübergreifender NC- Funktionalitäten (z.B. aktiver Splineinterpolation, aktiver Werkzeugradiuskorrektur) nicht erlaubt ist, ist auch ein synchrones LESEN während dieser Funktionalität nicht möglich. In diesen Fällen wird der Fehler ID 20651 ausgegeben.

Synchrones SCHREIBEN (siehe Bild, Fall a))**:** Ein Schreibzugriff wird wie ein sonstiges NC-Kommando während der Dekodierung eingeplant und erst später bei der NC-Bearbeitung tatsächlich durchgeführt.

Da diese Synchronisierung, insbesondere das Anhalten der Dekodierung bei einem Lesezugriff, unerwünschte Laufzeiteinflüsse hat, kann diese implizite Synchronisation bei der Definition der Variablen ausgeschaltet werden. Dies ist natürlich nur möglich, wenn der Zeitpunkt des Lesezugriffs nicht synchron zur Bearbeitung erfolgen muss bzw. dies durch einen expliziten Synchronisationspunkt (z.B. explizites #FLUSH WAIT) im NC-Programm sowieso sichergestellt ist.

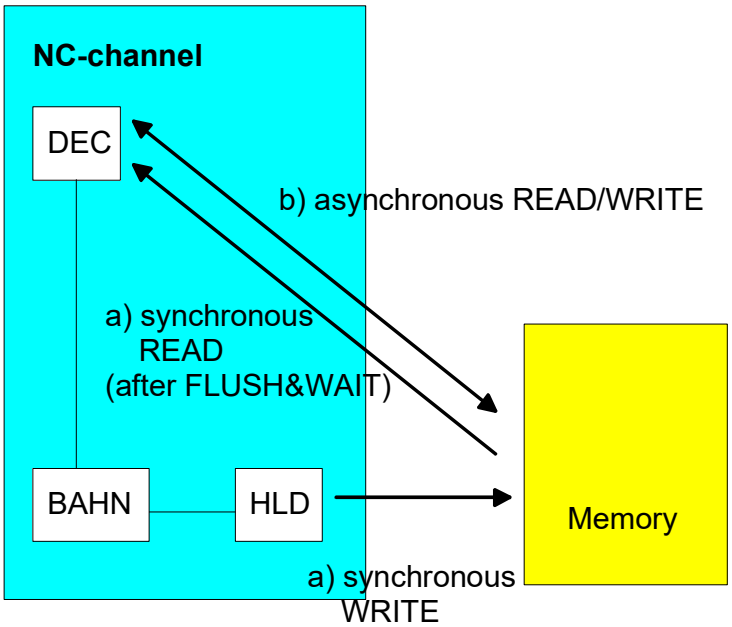

Abb. 6: Asynchroner/synchroner Zugriff der Dekodierung (DEC) und Abarbeitung (BAHN) über SPS-Treiber (HLD)

Achtung

Die Ausgabe synchronisierter V.E.-Variablen erfolgt steuerungsintern über die gleiche Schnittstelle wie die von Technologiefunktionen (M-/H-/T-Funktionen) mit der Synchronisationsart MOS. Der Anwender hat daher sicherzustellen, dass alle an dieser Schnittstelle anstehenden Daten von der SPS ausgelesen werden, da sonst die Synchronität zur Ausgabe im NC-Kanal nicht gewährleistet ist.

5.2 Oberfläche (P-EXTV-00030 - P-EXTV-00037)

Die Oberfläche (HMI) hat Zugriff auf NC-Schnittstellen und NC-Daten über sogenannte HMI-Objekte. Das Protokoll des Zugriffs wird über eine DLL gekapselt, welche eine Windows-Applikation den Schreib-/Lesezugriff auf HMI-Objekte anbietet. Für die Oberfläche können automatisch entsprechend der Konfigurationsliste Schnittstellenobjekte angelegt werden, wenn dies für die Variable entsprechend konfiguriert ist.

Für jede angegebene Variable werden dann zwei Objekte angelegt, je eines für den Schreib- und eines für den Lesezugriff. Die findet unabhängig von den Zugriffsrechten des NC-Kanals statt, d.h. selbst wenn der NC-Kanal nur lesenden Zugriff auf die Variable hat (z.B. access_rights = READ_ONLY), kann die Oberfläche auch schreibend zugreifen. Bei einem Array wird ein Oberflächenobjekt für jedes Arrayelement angelegt.

Falls gewünscht kann das Format der Namen der Oberflächenobjekte durch Angabe entsprechender Schablonen in der Liste beliebig angepasst werden (ab Version V254). Hierbei sind als Platzhalter %s für den Namen und bei einem Array nachfolgend zusätzlich %d für den Index anzugeben.

P-EXTV-00030	**HMI-Lesezugriff bei globalem Array**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_rd_global_array
Standardwert	**cnc_ve_**%s**_rd[**%d**]**
Anmerkungen	Platzhalter %s für den Namen und nachfolgend %d für den Arrayindex

P-EXTV-00031	**HMI-Schreibzugriff bei globalem Aray**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_wr_global_array
Standardwert	**cnc_ve_**%s**_wr[**%d**]**
Anmerkungen	Platzhalter %s für den Namen und nachfolgend %d für den Arrayindex

P-EXTV-00032	**HMI-Lesezugriff bei kanalspezifischem Array**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_rd_channel_array
Standardwert	**mc_ve_**%s**_rd[**%d**]**
Anmerkungen	Platzhalter %s für den Namen und nachfolgend %d für den Arrayindex

P-EXTV-00033	**HMI-Schreibzugriff bei kanalspezifischem Array**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_wr_channel_array
Standardwert	**mc_ve_**%s**_wr[**%d**]**
Anmerkungen	Platzhalter %s für den Namen und nachfolgend %d für den Arrayindex

P-EXTV-00034	**HMI-Lesezugriff bei globalen Variablen**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_rd_global
Standardwert	**cnc_ve_**%s**_rd**
Anmerkungen	Platzhalter %s für den Namen

P-EXTV-00035	**HMI-Schreibzugriff bei globalen Variablen**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_wr_global
Standardwert	**cnc_ve_**%s**_wr**
Anmerkungen	Platzhalter %s für den Namen

P-EXTV-00036	**HMI-Lesezugriff bei kanalspezifischen Variablen**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_rd_channel
Standardwert	**mc_ve_**%s**_rd**
Anmerkungen	Platzhalter %s für den Namen

P-EXTV-00037	**HMI-Schreibzugriff bei kanalspezifischen Variablen**
Beschreibung	Format des Namens des HMI-Objektes durch Angabe einer entsprechenden Schablone.
Parameter	name_wr_channel
Standardwert	**mc_ve_**%s**_wr**
Anmerkungen	Platzhalter %s für den Namen

Beispiel: Vergabe der Namen für die Oberflächenschnittstelle

```
# ************************************
#
# ************************************
#
name_rd_global_array              cnc_test1_%s_rd[%d]
name_wr_global_array              cnc_test1_%s_wr[%d]
name_rd_channel                   mc_test2_%s_rd
name_wr_channel                   mc_test2_%s_wr

...

var[0].name                       G_ARRAY5
var[0].type                       SGN32
var[0].scope                      GLOBAL
var[0].synchronisation            FALSE
var[0].access_rights              READ_WRITE
var[0].array_size                 5
var[0].create_hmi_interface       TRUE # HMI-Objekt wird angelegt
#

var[1].name                       L_BOOLEAN
var[1].type                       BOOLEAN
var[1].scope                      CHANNEL
var[1].synchronisation            FALSE
var[1].access_rights              READ_WRITE
var[1].array_size                 1
var[1].create_hmi_interface       TRUE # HMI-Objekt wird angelegt
```

Durch obigen Auszug der Konfigurationsliste werden folgende HMI-Objekte angelegt:

```
cnc_test1_G_ARRAY5_rd[0]
cnc_test1_G_ARRAY5_wr[0]
cnc_test1_G_ARRAY5_rd[1]
cnc_test1_G_ARRAY5_wr[1]
cnc_test1_G_ARRAY5_rd[2]
cnc_test1_G_ARRAY5_wr[2]
cnc_test1_G_ARRAY5_rd[3]
cnc_test1_G_ARRAY5_wr[3]
cnc_test1_G_ARRAY5_rd[4]
cnc_test1_G_ARRAY5_wr[4]

mc_test2_L_BOOLEAN_wr
mc_test2_L_BOOLEAN_rd
```

5.3 SPS

Nach Hochlauf der Steuerung erhält das SPS-Run-Time-System Zugriff auf die gemeinsamen Speicherbereiche für V.E.-Variablen zwischen CNC und SPS. Diese werden in kanalspezifische und einen globalen Speicherbereich unterschieden.

gpVE[iChannelIndex]r

gpVEGlob

Um die Nachbildung des Speicherbereiches für die SPS zu vereinfachen wird empfohlen diesen nach Anlegen mit dem #EXPORT VE –Befehl zu exportieren.
Weitere Informationen siehe [FCT-C22// Beschreibung]

Manuelles Nachbilden der Variablenstruktur in der SPS

Das nachfolgende SPS-Beispiel zeigt, wie die V.E.-Variablen durch manuelles Nachbilden der Variablenstrukturen in der SPS verwendet werden können.

Programmierbeispiel

Zugriff V.E.-Variablen in der SPS

Definition der Variablenstrukturen für die V.E.-Variablen:

```
TYPE VECTOR_T :
STRUCT
  x : LREAL;
  y : LREAL;
  z : LREAL;
END_STRUCT
END_TYPE

TYPE TARGET_POINT_T :
STRUCT
  point : VECTOR_T;
  valid : BOOL;
END_STRUCT
END_TYPE

TYPE TRAJEKTORIE_T :
STRUCT
  nbr_points : DINT;
  name : STRING(127);
  points : ARRAY [0..9] OF TARGET_POINT_T;
END_STRUCT
END_TYPE
```

Definition der Strukturen für den kompletten, belegten V.E.-Speicherbereich:

```
TYPE VE_GLOBAL:
STRUCT
  var_global_1 : DINT;
END_STRUCT
END_TYPE
```

```
TYPE VE_CHAN_1:
STRUCT
  var_chan_1 : DINT;
  array_chan_1 : ARRAY [0..19] OF INT;
  name : STRING(127);
  Trajektorie : TRAJEKTORIE_T;
END_STRUCT
END_TYPE
```

PLC-Programm zum Zugriff auf die V.E.-Variablen von Kanal 1:

```
PROGAM V_E
  VAR
    p_ve_chan_1 : POINTER TO VE_CHAN_1;
  END_VAR

(* Kanal 1 mit Index 0 - gpVECh[0]*)
p_ve_chan_1 := ADR (gpVECh[0]^.ext_var32[0]);

IF (p_ve_chan_1^.name = ,')
THEN
  p_ve_chan_1^.trajektorie.name := ,My Path!';
  p_ve_chan_1^.trajektorie.nbr_points := 2;

  p_ve_chan_1^.trajektorie.points[0].valid := TRUE;
  p_ve_chan_1^.trajektorie.points[0].point.x := 100.0;
  p_ve_chan_1^.trajektorie.points[0].point.y := 200.0;
  p_ve_chan_1^.trajektorie.points[0].point.z := 300.0;

  p_ve_chan_1^.trajektorie.points[1].valid := TRUE;
  p_ve_chan_1^.trajektorie.points[1].point.x := 200.0;
  p_ve_chan_1^.trajektorie.points[1].point.y := 400.0;
  p_ve_chan_1^.trajektorie.points[1].point.z := 600.0;
END_IF;
```

6 Anhang

6.1 Konfigurationssyntax bis V2.10.1025

Beispiel einer ASCII Liste:

```
# ************************************
# Externe Variablen V254
# ************************************
#
number_used_variables            2
#
var[0].name                      GLOBAL_SWR (Globale VE)
var[0].index                     0
var[0].type                      SGN16
var[0].scope                     GLOBAL
var[0].synchronisation           FALSE
var[0].access_rights             READ_ONLY
var[0].array_size                0
var[0].size                      2 # 2 Byte pro Element
var[0].create_hmi_interface      FALSE
#
var[1].name                      CHANNEL_WR (Kanal VE)
var[1].index                     0
var[1].type                      SGN32
var[1].scope                     CHANNEL
var[1].synchronisation           FALSE
var[1].access_rights             READ_ONLY
var[1].array_size                10
var[1].size                      4 # 4 Byte pro Element
var[1].create_hmi_interface      FALSE
#
Ende
```

Bezeichner	Wertebereich	Standard	Bedeutung
number_used_variables *(Alte Syntax bis V2.11.2034.0: anzahl_belegt)*	[0; MAX_UNS16]	0	Bei lückenloser Belegung von *var[i].** wird hier der Index der zuletzt definierten Variable +1 eingetragen (i_{last}+1). Bei Belegung mit Lücken von *var[i].** wird hier der höchste Index der definierten Variablen +1 eingetragen (i_{max}+1).
*var[i].**	i:= [0; 214]		Datensätze zur Definition der Variablen
name	ASCII-String		Siehe P-EXTV-00001
index	[0; MAX_SGN32]	-1	Der Index legt die Position im Speicher fest, an welcher die Variable abgelegt wird. Der gesamte Speicher ist dabei als Feld von 24 Byte großen Einheiten aufgebaut. Ein Index von –1 zeigt an, dass der Eintrag nicht belegt ist.
type	[BOOLEAN, SGN08, UNS08, SGN16, UNS16, SGN32; UNS32, REAL64, STRING]	UNS32	Siehe P-EXTV-00003
scope	[GLOBAL; CHANNEL]	CHANNEL	Siehe P-EXTV-00004
synchronisation	[TRUE, FALSE]	TRUE	Siehe P-EXTV-00005
access_rights	[READ_WRITE, READ_ONLY, WRITE_ONLY]	READ_WRITE	Siehe P-EXTV-00006
array_size	[0; MAX_UNS16]	0	Wird eine externe Variable nicht nur einmal sondern als Array dieser Variable benötigt, so ist die Anzahl der Elemente festzulegen. Ist die Variable kein Array, ist 0 anzugeben.
size	[0; MAX_UNS32]	4	Siehe P-EXTV-00008
create_hmi_interface	[TRUE, FALSE]	FALSE	Siehe P-EXTV-00009

6.1.1 Speicherlayout bis V2.10.1025

Über die Konfiguration wird die Sicht des Kanals auf den Speicher und somit dessen logische Strukturierung definiert. Der gesamte Speicher ist als Feld von 24-Byte-Blöcken (Union der Typen aller Inhalte) gerastert. Bei der Definition jeder Variable wird die Startposition durch Angabe des Blockindex in diesem Feld angegeben.

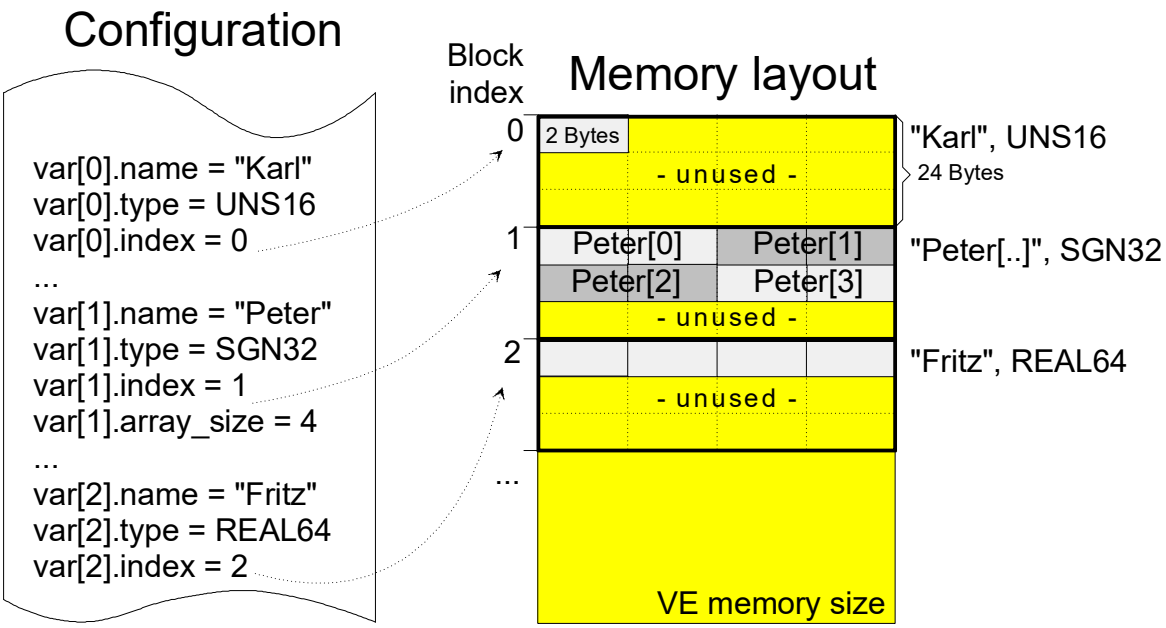

Abb. 7: Speicherlayout resultierend aus gegebener Konfiguration

Eine V.E.-Variable (auch V.E.-Array) wird zusammenhängend ab der angegebenen Startposition abgelegt. Ist sie größer als ein 24-Byte Raster, so wird der folgende Speicherbereich (Raster) mit hinzugezogen. Grundsätzlich ist es auch möglich, mehrere logische Variablen auf die identische Speicherstelle zu legen, d.h. mehrere Sichtweisen auf eine Speicherstelle freizugeben. Eine Überlappungsfreiheit der einzelnen Variablen wird durch die NC nicht überwacht.

Werden einzelne Variablen in den Speicher gelegt, so bleibt je nach Variablengröße nicht genutzter Speicher pro Speicherraster übrig. Dieser ist nicht weiter adressierbar.

6.1.2 Speicherblockindex (P-EXTV-00038)

P-EXTV-00038	**Index des Speicherblocks der externen Variable im Speicher**
Beschreibung	Der Index legt die Position im Speicher fest, an welcher die Variable abgelegt wird. Der gesamte Speicher ist dabei als Feld von 24 Byte großen Einheiten aufgebaut. Ein Index von -1 zeigt an, dass der Eintrag nicht belegt ist. Siehe Speicherlayout bis V2.10.1025 [▶ 48]
Parameter	var[i].index
Datentyp	SGN32
Datenbereich	0 ... MAX(SGN32)
Dimension	----
Standardwert	-1
Anmerkungen	

6.2 Verweise auf andere Dokumente

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

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

6.3 Literaturverzeichnis

Programmieranleitung CNC

HLI Manual

Export von V.E. Variablen in SPS Strukturen

6.4 Anregungen, Korrekturen und neueste Dokumentation

Sie finden Fehler, haben Anregungen oder konstruktive Kritik? Gerne können Sie uns unter documentation@isg-stuttgart.de kontaktieren. Die aktuellste Dokumentation finden Sie in unserer Onlinehilfe (DE/EN):

QR-Code Link: https://www.isg-stuttgart.de/documentation-kernel/

Der o.g. Link ist eine Weiterleitung zu:

https://www.isg-stuttgart.de/fileadmin/kernel/kernel-html/index.html

Hinweis

Mögliche Änderung von Favoritenlinks im Browser:

Technische Änderungen der Webseitenstruktur betreffend der Ordnerpfade oder ein Wechsel des HTML-Frameworks und damit der Linkstruktur können nie ausgeschlossen werden.

Wir empfehlen, den o.g. „QR-Code Link“ als primären Favoritenlink zu speichern.

PDFs zum Download:

DE:
https://www.isg-stuttgart.de/produkte/softwareprodukte/isg-kernel/dokumente-und-downloads

EN:
https://www.isg-stuttgart.de/en/products/softwareproducts/isg-kernel/documents-and-downloads

E-Mail: documentation@isg-stuttgart.de

Stichwortverzeichnis

P

ISG Industrielle Steuerungstechnik GmbH
STEP, Gropiusplatz 10
D-70563 Stuttgart

www.isg-stuttgart.de
support@isg-stuttgart.de