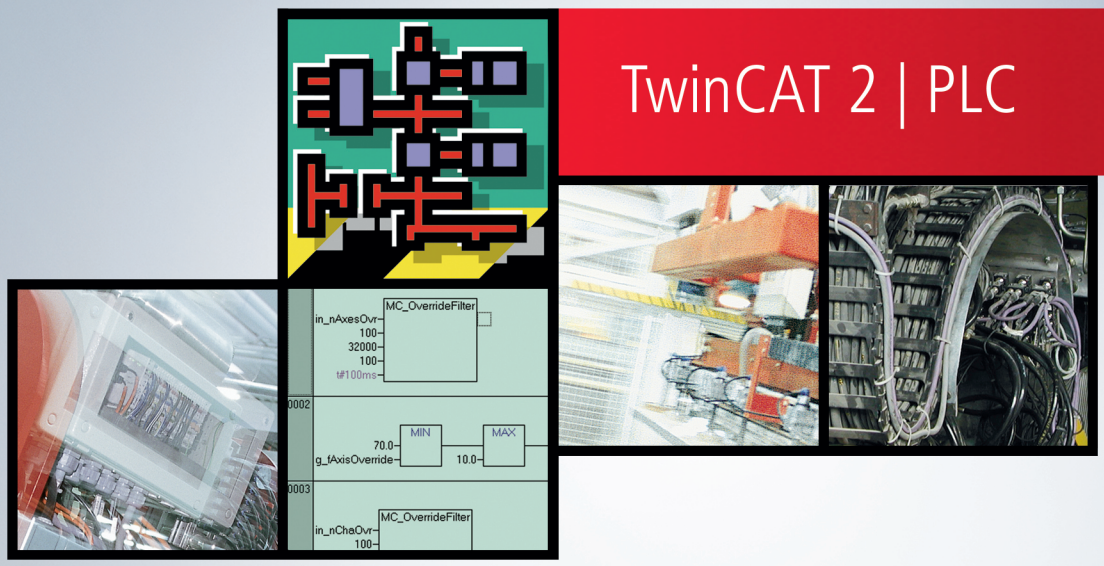
BECKHOFF New Automation Technology

Handbuch | DE

TwinCAT 2 PLC Control



13.06.2022 | Version: 1.0

1 Vorwort

1.1 Hinweise zur Dokumentation

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

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

Disclaimer

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

Marken

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

Patente

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

EtherCAT®

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

Copyright

1.2 Sicherheitshinweise

Sicherheitsbestimmungen

Beachten Sie die folgenden Sicherheitshinweise und Erklärungen!
Produktspezifische Sicherheitshinweise finden Sie auf den folgenden Seiten oder in den Bereichen Montage, Verdrahtung, Inbetriebnahme usw.

Haftungsausschluss

Die gesamten Komponenten werden je nach Anwendungsbestimmungen in bestimmten Hard- und Software-Konfigurationen ausgeliefert. Änderungen der Hard- oder Software-Konfiguration, die über die dokumentierten Möglichkeiten hinausgehen, sind unzulässig und bewirken den Haftungsausschluss der Beckhoff Automation GmbH & Co. KG.

Qualifikation des Personals

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

Erklärung der Symbole

In der vorliegenden Dokumentation werden die folgenden Symbole mit einem nebenstehenden Sicherheitshinweis oder Hinweistext verwendet. Die Sicherheitshinweise sind aufmerksam zu lesen und unbedingt zu befolgen!

⚠ GEFAHR

Akute Verletzungsgefahr!

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

⚠ WARNUNG

Verletzungsgefahr!

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

⚠ VORSICHT

Schädigung von Personen!

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

HINWEIS

Schädigung von Umwelt oder Geräten

Wenn der Hinweis neben diesem Symbol nicht beachtet wird, können Umwelt oder Geräte geschädigt werden.

Tipp oder Fingerzeig

Dieses Symbol kennzeichnet Informationen, die zum besseren Verständnis beitragen.

1.3 Hinweise zur Informationssicherheit

Die Produkte der Beckhoff Automation GmbH & Co. KG (Beckhoff) sind, sofern sie online zu erreichen sind, mit Security-Funktionen ausgestattet, die den sicheren Betrieb von Anlagen, Systemen, Maschinen und Netzwerken unterstützen. Trotz der Security-Funktionen sind die Erstellung, Implementierung und ständige Aktualisierung eines ganzheitlichen Security-Konzepts für den Betrieb notwendig, um die jeweilige Anlage, das System, die Maschine und die Netzwerke gegen Cyber-Bedrohungen zu schützen. Die von Beckhoff verkauften Produkte bilden dabei nur einen Teil des gesamtheitlichen Security-Konzepts. Der Kunde ist dafür verantwortlich, dass unbefugte Zugriffe durch Dritte auf seine Anlagen, Systeme, Maschinen und Netzwerke verhindert werden. Letztere sollten nur mit dem Unternehmensnetzwerk oder dem Internet verbunden werden, wenn entsprechende Schutzmaßnahmen eingerichtet wurden.

Zusätzlich sollten die Empfehlungen von Beckhoff zu entsprechenden Schutzmaßnahmen beachtet werden. Weiterführende Informationen über Informationssicherheit und Industrial Security finden Sie in unserem https://www.beckhoff.de/secguide.

Die Produkte und Lösungen von Beckhoff werden ständig weiterentwickelt. Dies betrifft auch die Security-Funktionen. Aufgrund der stetigen Weiterentwicklung empfiehlt Beckhoff ausdrücklich, die Produkte ständig auf dem aktuellen Stand zu halten und nach Bereitstellung von Updates diese auf die Produkte aufzuspielen. Die Verwendung veralteter oder nicht mehr unterstützter Produktversionen kann das Risiko von Cyber-Bedrohungen erhöhen.

Um stets über Hinweise zur Informationssicherheit zu Produkten von Beckhoff informiert zu sein, abonnieren Sie den RSS Feed unter https://www.beckhoff.de/secinfo.

Inhaltsverzeichnis

2 Kurzer Einblick in TwinCAT PLC Control

Was ist TwinCAT PLC Control?

Das TwinCAT PLC Control ist eine komplette Entwicklungsumgebung für Ihre Steuerung. Es ermöglicht dem SPS-Programmierer einen einfachen Einstieg in die mächtigen Sprachmittel der IEC. Die Benutzung der Editoren und der Debugging-Funktionen hat die ausgereiften Entwicklungsumgebungen höherer Programmiersprachen zum Vorbild.

Überblick über die Funktionalität von TwinCAT PLC Control

Wie ist ein Projekt strukturiert?

Ein Projekt wird in einer Datei abgelegt, die den Namen des Projekts trägt. Der erste Baustein, der in einem neuen Projekt angelegt wird, trägt als Vorschlag den Namen MAIN. Er wird von der vorgegebenen Task Standard aufgerufen. TwinCAT PLC Control unterscheidet verschiedene Arten von Objekten in einem Projekt: Bausteine, Datentypen und Ressourcen. Im Object Organizer finden Sie alle Objekte Ihres Projekts aufgelistet.

Wie erstelle ich mein Projekt?

Zuerst sollten Sie ein Zielsystem auswählen und eine Task vorgeben. Anschließend können Sie die für Ihr Problem notwendigen Bausteine anlegen. Sie können nun die Bausteine, die Sie benötigen, in den gewünschten Sprachen programmieren. Nach Abschluss der Programmierung können Sie das Projekt übersetzen, und eventuell angezeigte Fehler beseitigen.

Wie kann ich mein Projekt testen?

Sind alle Fehler beseitigt, loggen sich in der Steuerung ein und 'laden' Ihr Projekt in die Steuerung. Jetzt ist das TwinCAT PLC Control im Onlinebetrieb.
Sie können nun Ihr Projekt auf korrekten Ablauf testen. Belegen Sie hierzu manuell die Eingänge, und beobachten Sie, ob die Ausgänge wie gewünscht gesetzt werden. Des weiteren können Sie in den Bausteinen den Wertverlauf der lokalen Variablen beobachten. Im Watch- und Rezepturverwalter können Sie die Datensätze konfigurieren, deren Werte Sie betrachten wollen.
Im Falle eines Programmierfehlers können Sie Breakpoints setzen. Stoppt die Ausführung in einem solchen Breakpoint, so können Sie die Werte sämtlicher Projektvariablen zu diesem Zeitpunkt einsehen. Durch schrittweises Abarbeiten (Einzelschritt), können Sie die logische Korrektheit Ihres Programms überprüfen. Eine weitere Debugging-Funktion von TwinCAT PLC Control: Sie können Programmvariablen und Ein/Ausgängen auf bestimmte Werte setzen. Mit der Ablaufkontrolle können Sie überprüfen, welche Programmzeilen durchlaufen wurden. Die Traceaufzeichnung bietet Ihnen die Möglichkeit, den Verlauf von Variablen zyklusecht über einen längeren Zeitraum aufzuzeichnen und darzustellen. Ein Logbuch zeichnet Vorgänge bzw. Benutzeraktionen und interne Vorgänge während des Online-Modus chronologisch auf.

Das gesamte Projekt kann jederzeit dokumentiert oder in ein Textdatei exportiert werden.

Weitere Möglichkeiten

Das gesamte Projekt kann jederzeit dokumentiert, in eine Textdatei exportiert und in eine andere Sprache übersetzt werden.

ENI: Die Schnittstelle 'Engineering Interface' kann benutzt werden, um über den eigenständigen ENI Server auf eine externe Datenbank zur Quellcodeverwaltung/archivierung zuzugreifen. In dieser Datenbank können Bausteine, Datenstrukturen und Variablenlisten in Projekten oder Ressourcen verwaltet werden. Diese Schnittstelle steht auch anderen Clients des ENI Servers zur Verfügung.

Fazit

TwinCAT PLC Control ist ein vollständiges Entwicklungswerkzeug zur Programmierung Ihrer Steuerung, das Ihnen eine deutliche Zeitersparnis bei der Erstellung Ihrer Applikationen bringt.

2.1 Bestandteile eines Projekts

Ein Projekt beinhaltet alle Objekte eines Steuerungsprogramms. Ein Projekt wird in einer Datei mit dem Namen des Projekts gespeichert. Zu einem Projekt gehören folgende Objekte:
Bausteine, Datentypen, Ressourcen und Bibliotheken.

Baustein

Funktionen, Funktionsblöcke und Programme sind Bausteine, die durch Aktionen ergänzt werden können. Jeder Baustein besteht aus einem Deklarationsteil und einem Rumpf. Der Rumpf ist in einer der IEC-Programmiersprachen AWL, ST, AS, FUP, KOP oder CFC geschrieben.
TwinCAT PLC Control unterstützt alle IEC-Standardbausteine. Wenn Sie diese Bausteine in Ihrem Projekt benutzen wollen, müssen Sie die Bibliothek standard.lib in Ihr Projekt einbinden.
Bausteine können andere Bausteine aufrufen. Rekursionen sind jedoch nicht erlaubt.

Funktion

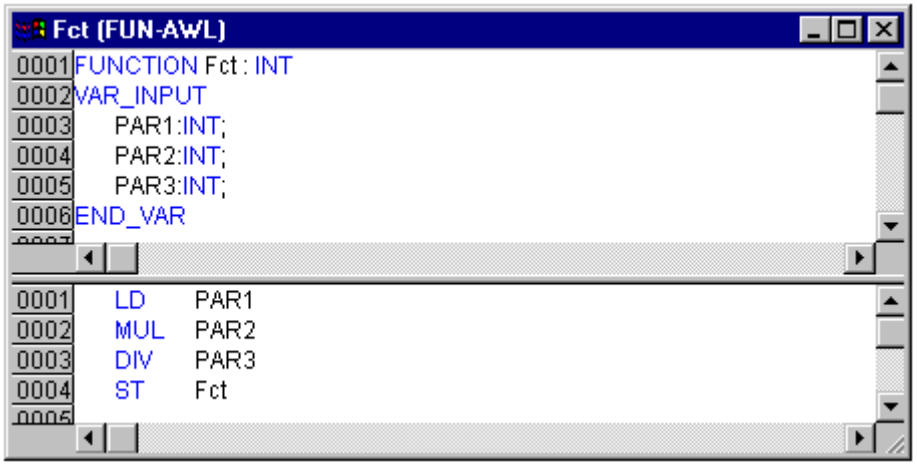
Eine Funktion ist ein Baustein, der bei der Ausführung genau ein Datenelement (das auch mehrelementig sein kann, wie z.B. Felder oder Strukturen) liefert, und dessen Aufruf in textuellen Sprachen als ein Operator in Ausdrücken vorkommen kann.
Bei der Deklaration einer Funktion ist darauf zu achten, dass die Funktion einen Typ erhalten muss. D.h. nach dem Funktionsnamen muss ein Doppelpunkt gefolgt von einem Typ eingegeben werden.

Eine korrekte Funktionsdeklaration sieht z.B. so aus:

FUNCTION Fct: INT

Außerdem muss der Funktion ein Ergebnis zugewiesen werden. Der Funktionsname wird benutzt wie eine Ausgabevariable.

Beispiel in AWL für eine Funktion, die drei Eingangsvariablen nimmt, und als Ergebnis das Produkt der ersten beiden, dividiert durch die letzte zurückliefert:



Der Aufruf einer Funktion kann in ST als ein Operand in Ausdrücken vorkommen.
Funktionen verfügen über keine internen Zustände. D.h. Aufrufe einer Funktion mit denselben Argumenten (Eingabeparametern) liefern immer denselben Wert (Ausgabe).

HINWEIS

Deklaration

Wird eine lokale Variable in einer Funktion als RETAIN deklariert, hat dies keine Auswirkung. Die Variable wird nicht im Retain-Bereich gespeichert.

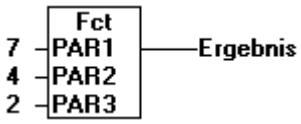
Beispiele für den Aufruf der oben beschriebenen Funktion:
in AWL:

```
LD 7
Fct 2,4
ST Ergebnis
```

in ST:

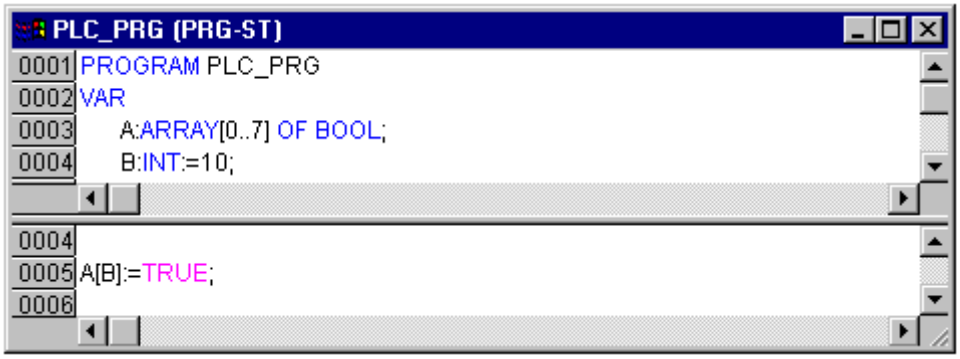
```
Ergebnis := Fct(7, 2, 4);
```

in FUP:



In AS kann ein Funktionsaufruf nur innerhalb eines Schrittes oder einer Transition erfolgen.

Wenn Sie in Ihrem Projekt eine Funktion mit Namen CheckBounds [▶ 341] definieren, können Sie damit Bereichsüberschreitungen in Ihrem Projekt automatisch überprüfen! Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.
Das folgende Beispielprogramm zum Testen der CheckBounds-Funktion greift außerhalb der Grenzen eines definierten Arrays zu. Die Funktion CheckBounds gewährleistet, dass der Wert TRUE nicht an die Stelle A[10], sondern an der oberen noch gültigen Bereichsgrenze A[7] zugewiesen wird. Mit der CheckBounds-Funktion können somit Zugriffe außerhalb von Array-Grenzen korrigiert werden.



Wenn Sie in Ihrem Projekt Funktionen mit Namen **CheckDivByte**, **CheckDivWord**, **CheckDivDWord** und **CheckDivReal** definieren, können Sie damit bei Verwendung des Operators DIV den Wert des Divisors überprüfen, beispielsweise um eine Division durch 0 zu verhindern. Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

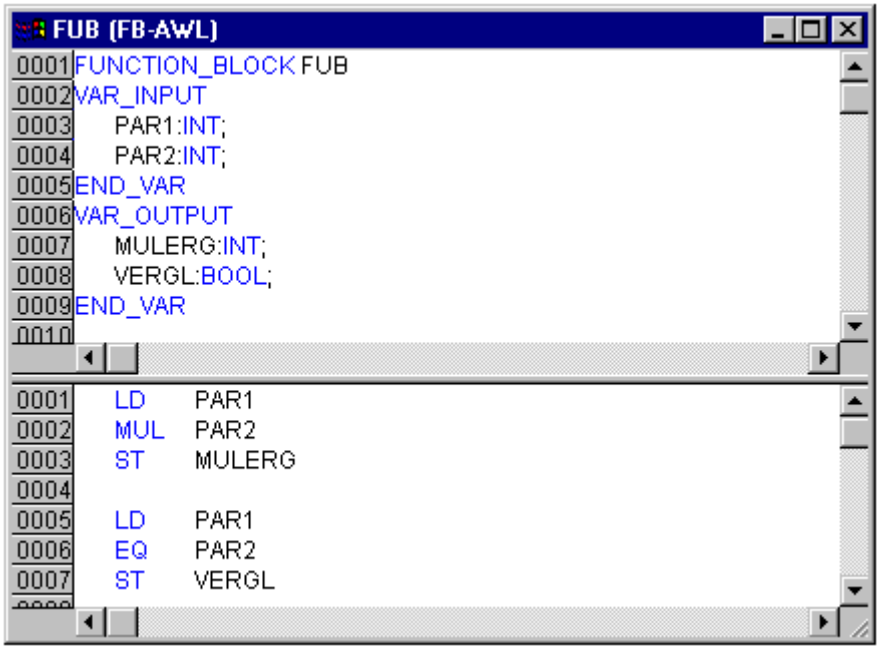
Wenn Sie die Funktionen **CheckRangeSigned** und **CheckRangeUnsigned** definieren, können Sie damit im online-Betrieb automatisch Bereichsüberschreitungen bei Variablen, die mit Unterbereichstypen deklariert sind, abfangen.

Die genannten Funktionsnamen sind aufgrund der hier beschriebenen Einsatzmöglichkeit reserviert.

Funktionsblock

Ein Funktionsblock ist ein Baustein, der bei der Ausführung einen oder mehrere Werte liefert. Ein Funktionsblock liefert keinen Rückgabewert im Gegensatz zu einer Funktion.
Es können Vervielfältigungen, genannt Instanzen (Kopien) eines Funktionsblocks geschaffen werden.

Beispiel in AWL für einen Funktionsblock mit zwei Eingabevariablen und zwei Ausgabevariablen. Eine Ausgabe ist das Produkt der beiden Eingaben, die andere ein Vergleich auf Gleichheit:



Instanzen von Funktionsblöcken
Es können Vervielfältigungen, genannt Instanzen (Kopien) eines Funktionsblocks geschaffen werden. Jede Instanz besitzt einen zugehörigen Bezeichner (den Instanznamen), und eine Datenstruktur, die ihre Eingaben, Ausgaben und internen Variablen beinhaltet. Instanzen werden wie Variablen lokal oder global deklariert, indem als Typ eines Bezeichners der Name des Funktionsblocks angegeben wird.

Beispiel für eine Instanz mit Namen INSTANZ des Funktionsblocks FUB:

INSTANZ: FUB;

Aufrufe von Funktionsblöcken geschehen stets über die oben beschriebenen Instanzen.
Nur auf die Ein- und Ausgabeparameter kann von außerhalb einer Instanz eines Funktionsblocks zugegriffen werden, nicht auf dessen interne Variablen.
Beispiel für den Zugriff auf eine Eingabevariable:
Der Funktionsblock fb hat eine Eingabevariable in1 vom Typ int.

```
PROGRAM prog
VAR
inst1:fb;
END_VAR
LD 17
ST inst1.in1
CAL inst1
END_PROGRAM
```

Die Deklarationsteile von Funktionsblöcken und Programmen können Instanzdeklarationen beinhalten. Instanzdeklarationen in Funktionen sind nicht zulässig.
Der Zugriff auf die Instanz eines Funktionsblocks ist auf den Baustein beschränkt, in dem sie instanziiert wurde, es sei denn, sie wurde global deklariert.
Der Instanzname einer Instanz eines Funktionsblocks kann als Eingabe einer Funktion oder eines Funktionsblocks benutzt werden.

Alle Werte bleiben von einer Ausführung des Funktionsblocks bis zur nächsten erhalten. Daher liefern Aufrufe eines Funktionsblocks mit denselben Argumenten nicht immer dieselben Ausgabewerte!

Enthält der Funktionsblock mindestens eine Retain-Variable, wird die gesamte Instanz im Retainbereich gespeichert.

Aufruf eines Funktionsblocks

Man kann die Eingabe- und Ausgabevariablen eines Funktionsblocks von einem anderen Baustein aus ansprechen, indem man eine Instanz des Funktionsblocks anlegt und über folgende Syntax die gewünschte Variable angibt:

<Instanzname>.<Variablenname>

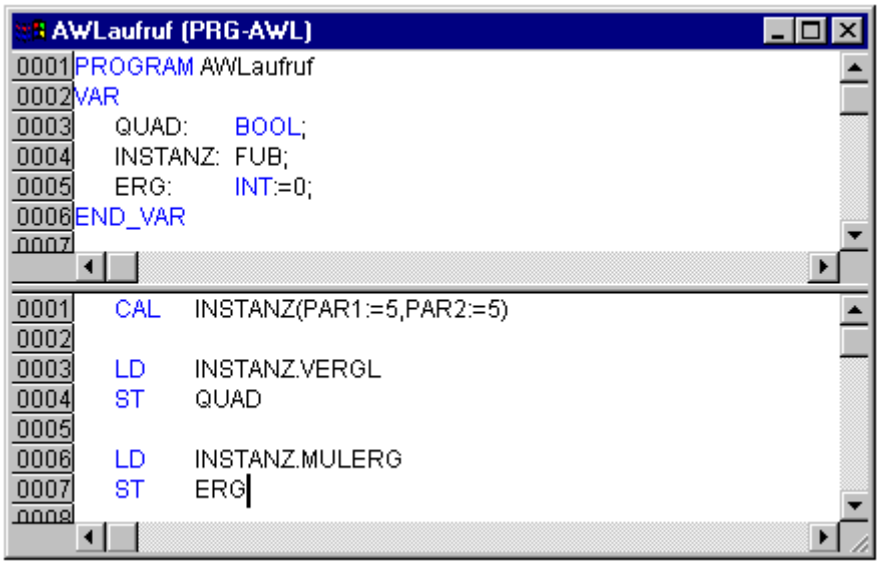
Wenn man die Eingabeparameter, also Werte der Inputvariablen, beim Aufruf setzen will, dann geschieht das bei den Textsprachen AWL und ST, indem man nach dem Instanznamen des Funktionsblocks in Klammer den Parametern Werte zuweist (die Zuweisung geschieht durch ":=" wie bei der Initialisierung von Variablen an der Deklarationsstelle).

Beachten Sie, dass Ein/Ausgabevariablen (VAR_IN_OUT) eines Funktionsblocks als Pointer übergeben werden. Ihnen können deshalb beim Aufruf keine Konstanten zugewiesen werden und es kann nicht lesend oder schreibend von außen auf sie zugegriffen werden. Beispiel für den Aufruf einer VAR_IN_OUT Variable inout1 des Funktionsblocks fubo in einem ST-Baustein:

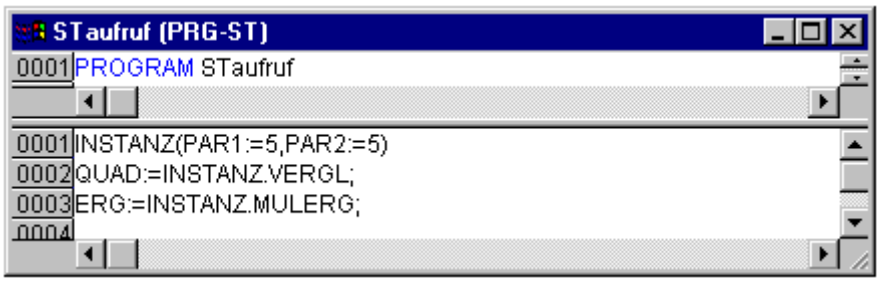
```
VAR

    inst:fubo;

    var1:int;

END_VAR


var1:=2;

inst(inout1:=var1);
```

Nicht zulässig wäre: inst(inout1:=2); bzw. inst.inout1:=2;

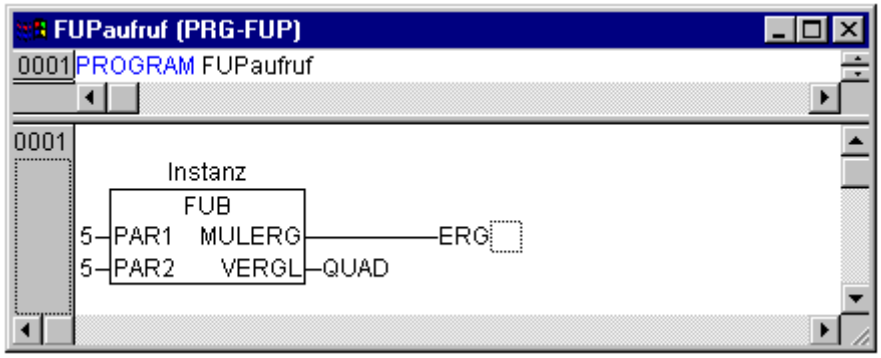
Beispiele für den Aufruf des oben beschriebenen Funktionsblocks FUB. Das Multiplikationsergebnis wird in der Variablen ERG abgelegt, das Ergebnis des Vergleichs wird in QUAD gespeichert. Es sei eine Instanz von FUB mit dem Namen INSTANZ deklariert.
In AWL wird die Instanz eines Funktionsblocks wie folgt aufgerufen:



Im Beispiel unten ist der Aufruf in ST abgebildet, wobei der Deklarationsteil gleich ist wie bei AWL:



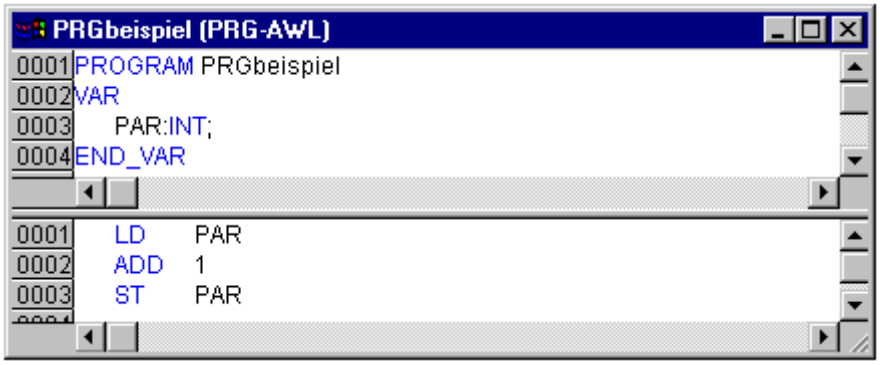
In FUP würde es wie folgt aussehen (Deklarationsteil ebenfalls wie bei AWL):



In AS können Aufrufe von Funktionsblöcken nur in Schritten vorkommen.

Programm

Ein Programm ist ein Baustein, der bei der Ausführung einen oder mehrere Werte liefert. Programme sind global im gesamten Projekt bekannt. Alle Werte bleiben von einer Ausführung des Programms bis zur nächsten erhalten.



Programme können von Programmen und Funktionsblöcken aufgerufen werden. Ein Programmaufruf in einer Funktion ist nicht erlaubt. Es gibt auch keine Instanzen von Programmen.
Wenn ein Baustein ein Programm aufruft, und es werden dabei Werte des Programms verändert, dann bleiben diese Veränderungen beim nächsten Aufruf des Programms erhalten, auch wenn das Programm von einem anderen Baustein aus aufgerufen wird.
Dies ist anders als beim Aufruf eines Funktionsblocks. Dort werden nur die Werte in der jeweiligen Instanz eines Funktionsblocks geändert. Diese Veränderungen spielen also auch nur eine Rolle, wenn dieselbe Instanz aufgerufen wird.

Beispiele für Aufrufe des oben beschriebenen Programms:

In AWL:

```
CAL PRGbeispiel
LD PRGbeispiel.PAR
ST ERG
```

In ST:

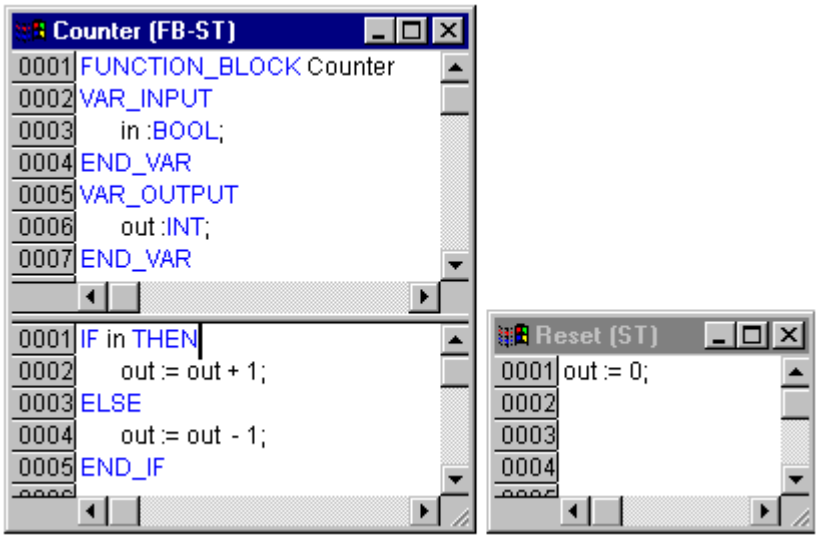
```
PRGbeispiel;
Erg := PRGbeispiel.PAR;
```

In FUP:

Wenn von einem Hauptprogramm aus zunächst die Variable PAR des Programms PRGbeispiel mit 0 initialisiert wird, und dann nacheinander Programme mit den obigen Programmaufrufen aufgerufen werden, dann wird das Ergebnis Erg in den Programmen die Werte 1,2 und 3 haben. Wenn man die Reihenfolge der Aufrufe vertauscht, ändern sich dementsprechend auch die Werte der jeweiligen Ergebnisparameter.

Aktion

Zu Funktionsblöcken und Programmen können Aktionen definiert werden. Die Aktion stellt eine weitere Implementation dar, die durchaus in einer anderen Sprache als die 'normale' Implementation erstellt werden kann. Jede Aktion erhält einen Namen.
Eine Aktion arbeitet mit den Daten des Funktionsblocks bzw. Programmes, zu dem sie gehört. Die Aktion verwendet die gleichen Ein-/ Ausgabevariablen und lokalen Variablen, wie die 'normale' Implementation.



In diesem Beispiel wird bei Aufruf des Funktionsblocks Counter die Ausgabevariable out erhöht bzw. erniedrigt in Abhängigkeit der Eingabevariablen in. Bei Aufruf der Aktion Reset des Funktionsblocks wird die Ausgabevariable out auf Null gesetzt. Es wird in beiden Fällen die gleiche Variable out beschrieben.

Eine Aktion wird aufgerufen mit <Programmname>.<Aktionsname> bzw. <Instanzname>.<Aktionsname>. Soll die Aktion innerhalb des eigenen Bausteins aufgerufen werden, so verwendet man in den Texteditoren nur den Aktionsnamen und in den grafischen den Funktionsblockaufruf ohne Instanzangabe.

Beispiele für Aufrufe der obigen Aktion:
Deklaration für alle Beispiele:

PROGRAM PLC_PRG
VAR
Inst : Counter;
END_VAR

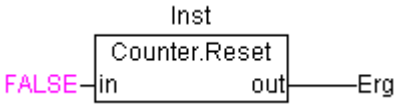
In AWL:

```
CAL Inst.Reset(In := FALSE)
LD Inst.out
ST ERG
```

In ST:

```
Inst.Reset(In := FALSE);
Erg := Inst.out;
```

In FUP:



Bei Bausteinen in Ablaufsprache spielen Aktionen eine besondere Rolle, siehe hierzu Kapitel Ablaufsprache [▶ 25].

Die IEC-Norm kennt keine Aktionen, außer die der Ablaufsprache.

Ressourcen

Die Ressourcen werden zum Konfigurieren und Organisieren Ihres Projektes und zur Verfolgung von Variablenwerten benötigt:

- Globale Variablen, die im ganzen Projekt verwendet werden können
- Steuerungskonfiguration zum Konfigurieren Ihrer Hardware
- Taskkonfiguration zur Steuerung Ihres Programms über Tasks
- Traceaufzeichnung zur grafischen Aufzeichnung von Variablenwerten
- Watch- und Rezepturverwalter zum Anzeigen und Vorbelegen von Variablenwerten

Siehe hierzu das Kapitel 'Die Ressourcen [▶ 169]'.

Bibliotheken

Sie können in Ihr Projekt eine Reihe von Bibliotheken einbinden, deren Bausteine, Datentypen und globale Variablen Sie genauso benutzen können, wie selbstdefinierte. Die Bibliothek 'standard.lib' steht Ihnen standardmäßig zur Verfügung.

Siehe hierzu das Kapitel 'Bibliotheksverwaltung [▶ 192]'.

Datentypen

Neben den Standarddatentypen können vom Benutzer eigene Datentypen definiert werden. Strukturen, Aufzählungstypen und Referenzen können angelegt werden.

Siehe hierzu 'Standard [▶ 302]-' und 'definierte Datentypen [▶ 302]' im Anhang.

2.2 Die Sprachen

2.2.1 Programmiersprachen

Das TwinCAT PLC Control unterstützt alle in der IEC 61131-3 beschriebenen Sprachen. Es gibt zwei textuelle Sprachen und drei grafische Sprachen.

Textuelle Sprachen

- Anweisungsliste (AWL, IL) [▶ 17]
- Strukturierter Text (ST) [▶ 19]

grafische Sprachen

- Funktionsplan (FUP, FBD) [▶ 29]
- Kontaktplan (KOP, LD) [▶ 29]
- Freigrafischer Funktionsplaneditor (CFC) [▶ 29]
- Ablaufsprache (AS, SFC) [▶ 25]

2.2.2 Anweisungsliste (AWL)

Eine Anweisungsliste (AWL) besteht aus einer Folge von Anweisungen. Jede Anweisung beginnt in einer neuen Zeile, und beinhaltet einen Operator und, je nach Art der Operation, einen oder mehrere durch Kommata abgetrennte Operanden.
Vor einer Anweisung kann sich ein Identifikator Marke befinden, gefolgt von einem Doppelpunkt (:).
Ein Kommentar muss das letzte Element in einer Zeile sein. Leere Zeilen können zwischen Anweisungen eingefügt werden.

Beispiel:

```
LD      17

ST      lint      (* Kommentar *)

GE      5

JMPC    next

LD      idword

EQ      istruct.sdword

STN     test

next:
```

Modifikatoren und Operatoren in AWL

In der Sprache AWL können folgende Operatoren und Modifikatoren verwendet werden.
Modifikatoren:

- C bei JMP, CAL, RET: Die Anweisung wird nur ausgeführt, wenn das Ergebnis des vorhergehenden Ausdrucks TRUE ist.
- N bei JMPC, CALC, RETC: Die Anweisung wird nur ausgeführt, wenn das Ergebnis des vorhergehenden Ausdrucks FALSE ist.
- N sonst: Negation des Operanden (nicht des Akku)

Im Folgenden finden Sie eine Tabelle aller Operatoren in AWL mit deren möglichen Modifikatoren und der jeweiligen Bedeutung:

Operator	Modifikatoren	Bedeutung
LD	N	Setze aktuelles Ergebnis gleich dem Operanden
ST	N	Speichere aktuelles Ergebnis an die Operandenstelle
S		Setze den Bool-Operand genau dann auf TRUE, wenn das aktuelle Ergebnis TRUE ist
R		Setze den Bool-Operand genau dann auf FALSE, wenn das aktuelle Ergebnis TRUE ist
AND	N, (	Bitweise AND
OR	N, (	Bitweise OR
XOR	(	Bitweise exklusives OR
ADD	(	Addition
SUB	(	Subtraktion
MUL	(	Multiplikation
DIV	(	Division
GT	(	>
GE	(	>=
EQ	(	=
NE	(	<>
LE	(	<=
LT	(	<
JMP	CN	Springe zur Marke
CAL	CN	Rufe Funktionsblock auf
RET	CN	Verlasse den Baustein und kehre ggf. zurück zum Aufrufer
)		Werte zurückgestellte Operation aus

Eine Auflistung sämtlicher IEC-Operatoren finden Sie im Anhang.

Beispiel für ein AWL-Programm unter Verwendung einiger Modifikatoren:

```
LD     TRUE      (*Lade TRUE in den
Akkumulator*)

ANDN   BOOL1     (*führe AND mit dem negierten Wert der
Variable BOOL1 aus*)

JMPC   marke     (*wenn das Ergebnis TRUE war, springe zur
Marke "marke"*)

LDN    BOOL2     (*Speichere den negierten Wert von *)

ST     ERG       (*BOOL2 in ERG*)

marke:


LD     BOOL2     (*Speichere den Wert von *)

ST     ERG       (*BOOL2 in ERG*)
```

Es ist in AWL auch möglich, Klammern nach einer Operation zu setzen. Als Operand wird dann der Wert der Klammer betrachtet.
Zum Beispiel:

```
LD 2

MUL 2

ADD 3

ST Erg
```

Hier ist der Wert von Erg 7. Mit Klammern:

```
LD 2
MUL( 2
ADD 3
)
ST Erg
```

Hier ergibt sich als Wert für Erg 10, denn die Operation MUL wird erst ausgewertet, wenn man auf ")" trifft; als Operand für MUL errechnet sich dann 5.

2.2.3 Strukturierter Text (ST)

Der Strukturierte Text besteht aus einer Reihe von Anweisungen, die wie in Hochsprachen bedingt ("IF..THEN..ELSE") oder in Schleifen (WHILE..DO) ausgeführt werden können.
Beispiel:

```
IF value < 7 THEN
    WHILE value < 8 DO
    value := value + 1;
    END_WHILE;
END_IF;
```

Ausdrücke

Ein Ausdruck ist ein Konstrukt, das nach seiner Auswertung einen Wert zurückliefert. Ausdrücke sind zusammengesetzt aus Operatoren und Operanden. Ein Operand kann eine Konstante, eine Variable, ein Funktionsaufruf oder ein weiterer Ausdruck sein.

Auswertung von Ausdrücken

Die Auswertung eines Ausdrucks erfolgt durch Abarbeitung der Operatoren nach bestimmten Bindungsregeln. Der Operator mit der stärksten Bindung wird zuerst abgearbeitet, dann der Operator mit der nächststärkeren Bindung, usw., bis alle Operatoren abgearbeitet sind. Operatoren mit gleicher Bindungsstärke werden von links nach rechts abgearbeitet.

Nachfolgend finden Sie eine Tabelle der ST-Operatoren in der Ordnung ihrer Bindungsstärke:

Operation	Symbol	Bindungsstärke
Einklammern	(Ausdruck)	Stärkste Bindung
Funktionsaufruf	Funktionsname (Parameterliste)	
Potenzieren	EXPT	
Negieren Komplementbildung	- NOT	
Multiplizieren Dividieren Modulo	* / MOD	
Addieren Subtrahieren	+ -	
Vergleiche	<,>,<=,>=	
Gleichheit Ungleichheit	= <>	
Bool AND	AND	
Bool XOR	XOR	
Bool OR	OR	Schwächste Bindung

Es gibt folgende Anweisungen in ST, tabellarisch geordnet samt Beispiel:

Anweisungsart	Beispiel
Zuweisung	A:=B; CV := CV + 1; C:=SIN(X);
Aufruf eines Funktionsblocks und Benutzung der FB-Ausgabe	CMD_TMR(IN := %IX5, PT := 300);A:=CMD_TMR.Q;
RETURN	RETURN;
IF	IF D:=B*B; IF D<0.0 THEN C:=A; ELSIF D=0.0 THEN C:=B; ELSE C:=D; END_IF;
CASE	CASE INT1 OF 1: BOOL1 := TRUE; 2: BOOL2 := TRUE; ELSE BOOL1 := FALSE; BOOL2 := FALSE; END_CASE;
FOR	FOR J:=101; FOR I:=1 TO 100 BY 2 DO IF ARR[I] = 70 THEN J:=I; EXIT; END_IF; END_FOR;
WHILE	WHILE J<= 100 AND ARR[J] <> 70 DO J:=J+2; END_WHILE;
REPEAT	REPEAT J:=J+2; UNTIL J= 101 OR ARR[J] = 70 END_REPEAT;
EXIT	EXIT;
Leere Anweisung	;

Anweisungen im Strukturierten Text

Wie der Name schon sagt, ist der Strukturierte Text auf strukturiertes Programmieren ausgelegt, d.h. für bestimmte häufig benutzte Konstrukte, wie etwa Schleifen, bietet ST vorgegebene Strukturen zur Programmierung. Dies hat die Vorteile geringerer Fehlerwahrscheinlichkeit und größerer Übersichtlichkeit des Programms. Vergleichen wir zum Beispiel zwei gleichbedeutende Programmsequenzen in AWL und ST:

Eine Schleife zum Berechnen von Zweierpotenzen in AWL:

```
LD Zaehler

EQ 0

JMPC ende

LD Var1

MUL 2

ST Var1

LD Zaehler

SUB 1

ST Zaehler

JMP schleife
```

```
ende:

LD Var1

ST Erg
```

Dieselbe Schleife in ST programmiert würde ergeben:

```
WHILE Zaehler<>0 DO

    Var1:=Var1*2;

    Zaehler:=Zaehler-1;

END_WHILE
Erg:=Var1;
```

Die Schleife in ST ist nicht nur kürzer zu programmieren, sondern auch wesentlich leichter zu lesen, vor allem, wenn man sich nun ineinander geschachtelte Schleifen in größeren Konstrukten vorstellt.
Die verschiedenen Strukturen im ST haben folgende Bedeutung:

Zuweisungsoperator

Auf der linken Seite einer Zuweisung steht ein Operand (Variable, Adresse), dem der Wert des Ausdrucks auf der rechten Seite zugewiesen wird mit dem Zuweisungsoperator :=
Beispiel:
Var1 := Var2 * 10;
Nach Ausführung dieser Zeile hat Var1 den zehnfachen Wert von Var2.

Aufruf von Funktionsblöcken in ST

Ein Funktionsblock in ST wird aufgerufen, indem man den Namen der Instanz des Funktionsblocks schreibt und anschließend in Klammer die gewünschten Werte der Parameter zuweist. Im folgenden Beispiel wird ein Timer aufgerufen mit Zuweisungen für die Parameter IN und PT. Anschließend wird die Ergebnisvariable Q an die Variable A zugewiesen.
Die Ergebnisvariable wird wie in AWL mit dem Namen des Funktionsblocks, einem anschließenden Punkt und dem Namen der Variablen angesprochen:

CMD_TMR(IN := %IX5, PT := 300);
A:=CMD_TMR.Q

2.2.3.1 FOR-Schleife

Mit der FOR-Schleife kann man wiederholte Vorgänge programmieren.
Syntax:

```
INT_Var :INT;


FOR <INT_Var> := <INIT_WERT> TO <END_WERT> {BY
<Schrittgröße>} DO

    <Anweisungen>

END_FOR;
```

Der Teil in geschweiften Klammern {} ist optional.
Die <Anweisungen> werden solange ausgeführt, solange der Zähler <INT_Var> nicht größer als der <END_WERT> ist. Dies wird vor der Ausführung der <Anweisungen> überprüft, so dass die <Anweisungen> niemals ausgeführt werden, wenn <INIT_WERT> größer als <END_WERT> ist.
Immer, wenn <Anweisungen> ausgeführt worden ist, wird <INT_Var> um <Schrittgröße> erhöht. Die Schrittgröße kann jeden Integerwert haben. Fehlt sie wird diese auf 1 gesetzt. Die Schleife muss also terminieren, da <INT_Var> nur größer wird.
Beispiel:

```
FOR Zaehler:=1 TO 5 BY 1 DO

    Var1:=Var1*2;

END_FOR;

Erg:=Var1;
```

Mit der Annahme, dass die Variable Var1 mit dem Wert 1 vorbelegt wurde, dann wird sie nach der FOR-Schleife den Wert 32 haben.

Der <END_WERT> darf nicht der Grenzwert des Zählers <INT_VAR> sein. Z.B. wenn die Variable Zaehler vom Typ SINT ist, darf der <END_WERT> nicht 127 sein, sonst erfolgt eine Endlosschleife.

2.2.3.2 REPEAT-Schleife

Die REPEAT-Schleife unterscheidet sich von den WHILE-Schleifen dadurch, dass die Abbruchbedingung erst nach dem Ausführen der Schleife überprüft wird. Das hat zur Folge, dass die Schleife mindestens einmal durchlaufen wird, egal wie die Abbruchbedingung lautet.
Syntax:

```
REPEAT

<Anweisungen>

UNTIL <Boolescher Ausdruck>

END_REPEAT;
```

Die <Anweisungen> werden solange ausgeführt, bis <Boolescher Ausdruck> TRUE ergibt.
Wenn <Boolescher Ausdruck> bereits bei der ersten Auswertung TRUE ergibt, dann werden <Anweisungen> genau einmal ausgeführt. Wenn <Boolescher_Ausdruck> niemals den Wert TRUE annimmt, dann werden die <Anweisungen> endlos wiederholt, wodurch ein Laufzeitfehler entsteht.

Der Programmierer muss selbst dafür sorgen, dass keine Endlosschleife entsteht, indem er im Anweisungsteil der Schleife die Bedingung verändert, also zum Beispiel einen Zähler hoch- oder runterzählt.

Beispiel:

```
REPEAT

    Var1 := Var1*2;

    Zaehler := Zaehler-1;

UNTIL

    Zaehler=0

END_REPEAT
```

2.2.3.3 WHILE-Schleife

Die WHILE-Schleife kann benutzt werden wie die FOR-Schleife, mit dem Unterschied, dass die Abbruchbedingung ein beliebiger boolscher Ausdruck sein kann. Das heißt, man gibt eine Bedingung an, die, wenn sie zutrifft, die Ausführung der Schleife zur Folge hat.
Syntax:

```
WHILE <Boolescher Ausdruck> DO

    <Anweisungen>

END_WHILE;
```

Die <Anweisungen> werden solange wiederholt ausgeführt, solange <Boolescher_Ausdruck> TRUE ergibt.
Wenn <Boolescher_Ausdruck> bereits bei der ersten Auswertung FALSE ist, dann werden die <Anweisungen> niemals ausgeführt. Wenn <Boolescher_Ausdruck> niemals den Wert FALSE annimmt, dann werden die <Anweisungen> endlos wiederholt, wodurch ein Laufzeitfehler entsteht.

Der Programmierer muss selbst dafür sorgen, dass keine Endlosschleife entsteht, indem er im Anweisungsteil der Schleife die Bedingung verändert, also zum Beispiel einen Zähler hoch- oder runterzählt.

Beispiel:

```
WHILE Zaehler<>0 DO

    Var1 := Var1*2;

    Zaehler := Zaehler-1;

END_WHILE
```

Die WHILE- und die REPEAT-Schleife sind in gewissem Sinne mächtiger als die FOR-Schleife, da man nicht bereits vor der Ausführung der Schleife, die Anzahl der Schleifendurchläufe wissen muss. In manchen Fällen wird man also nur mit diesen beiden Schleifenarten arbeiten können. Wenn jedoch die Anzahl der Schleifendurchläufe klar ist, dann ist eine FOR- Schleife zu bevorzugen, da sie keine endlosen Schleifen ermöglicht.

2.2.3.4 IF-Anweisung

Mit der IF-Anweisung kann man eine Bedingung abprüfen und abhängig von dieser Bedingung eine Anweisung ausführen.
Syntax:

```
IF <Boolscher_Ausdruck1> THEN

<IF_Anweisungen>

{ELSIF <Boolscher_Ausdruck2> THEN

<ELSIF_Anweisungen1>

.

.

ELSIF <Boolscher_Ausdruck n> THEN

<ELSIF_Anweisungen n-1>

ELSE

<ELSE_Anweisungen>}

END_IF;
```

Der Teile in geschweiften Klammern {} ist optional.
Wenn <Boolscher_Ausdruck1> TRUE ergibt, dann werden nur die <IF_Anweisungen> ausgeführt und keine der weiteren Anweisungen.
Andernfalls werden die Boolschen Ausdrücke, beginnend mit <Boolscher_Ausdruck2> der Reihe nach ausgewertet, bis einer der Ausdrücke TRUE ergibt. Dann werden nur die Anweisungen nach diesem Boolschen Ausdruck und vor dem nächsten ELSE oder ELSIF ausgewertet. Wenn keine der Boolschen Ausdrücke TRUE ergibt, dann werden ausschließlich die <ELSE_Anweisungen> ausgewertet.

Beispiel:

```
IF temp<17

THEN heizung_an := TRUE;

ELSE heizung_an := FALSE;

END_IF;
```

Hier wird die Heizung angemacht, wenn die Temperatur unter 17 Grad sinkt, ansonsten bleibt sie aus.

2.2.3.5 CASE-Anweisung

Mit der CASE-Anweisung kann man mehrere bedingte Anweisungen mit derselben Bedingungsvariablen in ein Konstrukt zusammenfassen.
Syntax:

```
CASE <Var1> OF

    <Wert 1>:

    <Anweisung 1>

    <Wert 2>:

    <Anweisung 2>

    <Wert3, Wert4, Wert5:

    <Anweisung 3>

    <Wert6 .. Wert10 :

    <Anweisung 4>

    ...

    <Wert n>:

    <Anweisung n>

ELSE

    <ELSE-Anweisung>

END_CASE;
```

Eine CASE-Anweisung wird nach folgenden Schema abgearbeitet:

- Wenn die Variable in <Var1> den Wert <Wert i> hat, dann wird die Anweisung <Anweisung i> ausgeführt.
- Hat <Var1> keinen der angegebenen Werte, dann wird die <ELSE-Anweisung> ausgeführt.
- Wenn für mehrere Werte der Variablen, dieselbe Anweisung auszuführen ist, dann kann man diese Werte mit Kommata getrennt hintereinander schreiben, und damit die gemeinsame Anweisung bedingen.
- Wenn für einen Wertebereich der Variablen, dieselbe Anweisung auszuführen ist, dann kann man den Anfangs- und Endwert getrennt durch zwei Punkte hintereinander schreiben, und damit die gemeinsame Anweisung bedingen.

Beispiel:

```
CASE INT1 OF

    1, 5:

    BOOL1 := TRUE;

    BOOL3 := FALSE;

    2:

    BOOL2 := FALSE;

    BOOL3 := TRUE;

    10..20:

    BOOL1 := TRUE;

    BOOL3:= TRUE;

ELSE

    BOOL1 := NOT BOOL1;
```

```
    BOOL2 := BOOL1 OR BOOL2;
END_CASE;
```

2.2.3.6 RETURN-Anweisung

Die RETURN-Anweisung kann man verwenden, um eine Baustein zu verlassen, beispielsweise abhängig von einer Bedingung.

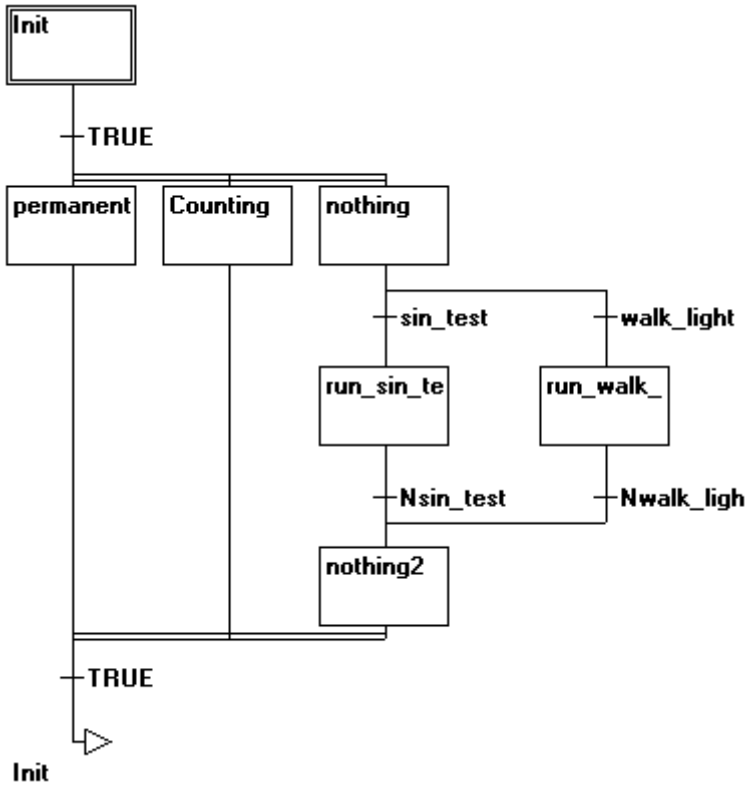
2.2.3.7 EXIT-Anweisung

Wenn die EXIT-Anweisung in einer FOR-, WHILE- oder REPEAT-Schleife vorkommt, dann wird die innerste Schleife beendet, ungeachtet der Abbruchbedingung.

2.2.4 Ablaufsprache (AS)

Die Ablaufsprache ist eine graphisch orientierte Sprache, die es ermöglicht, die zeitliche Abfolge verschiedener Aktionen innerhalb eines Programms zu beschreiben.

Beispiel für ein Netzwerk in der Ablaufsprache:



Schritt

Ein in Ablaufsprache geschriebener Baustein besteht aus einer Folge von Schritten, die über gerichtete Verbindungen (Transitionen) miteinander verbunden sind.

Es gibt zwei Arten von Schritten:

- Die vereinfachte Form besteht aus einer Aktion und einem Flag, das anzeigt, ob der Schritt aktiv ist. Ist zu einem Schritt die Aktion implementiert, so erscheint ein kleines Dreieck in der rechten oberen Ecke des Schrittes.

- Ein IEC-Schritt besteht aus einem Flag und einer oder mehreren zugewiesenen Aktionen oder boolschen Variablen. Die assoziierten Aktionen erscheinen rechts vom Schritt.

Aktion

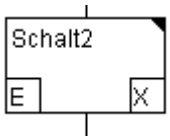
Eine Aktion kann eine Folge von Instruktionen in AWL oder in ST, eine Menge von Netzwerken in FUP oder in KOP oder wieder eine Ablaufstruktur (AS) enthalten. Bei den vereinfachten Schritten ist eine Aktion immer mit ihrem Schritt verbunden. Um eine Aktion zu editieren, führen Sie auf dem Schritt, zu dem die Aktion gehört, einen doppelten Mausklick aus, oder markieren Sie den Schritt und führen den Menübefehl, Extras"Zoom Aktion/Transition' aus. Zusätzlich sind eine Ein- und/oder Ausgangsaktion pro Schritt möglich.

Aktionen von IEC-Schritten hängen im Object Organizer direkt unter ihrem AS-Baustein und werden mit Doppelklick oder Drücken der <Eingabetaste> in ihren Editor geladen. Neue Aktionen können mit ‚Projekt"Aktion hinzufügen' erzeugt werden.

Eingangs- bzw. Ausgangsaktion

Einem Schritt kann zusätzlich zur Schritt-Aktion eine Eingangsaktion und eine Ausgangsaktion hinzugefügt werden. Eine Eingangsaktion wird nur einmal ausgeführt, gleich nachdem der Schritt aktiv geworden ist. Eine Ausgangsaktion wird nur einmal ausgeführt, bevor der Schritt deaktiviert wird. Ein Schritt mit Eingangsaktion wird durch ein 'E' in der linken unteren Ecke gekennzeichnet, die Ausgangsaktion durch ein 'X' in der rechten unteren Ecke. Die Ein- und Ausgangsaktion kann in einer beliebigen Sprache implementiert werden. Um eine Ein- bzw. Ausgangsaktion zu editieren, führen Sie auf die entsprechende Ecke im Schritt, einen doppelten Mausklick aus.

Beispiel eines Schrittes mit Ein- und Ausgangsaktion:



Transition/Transitionsbedingung

Zwischen den Schritten liegen sogenannte Transitionen.

Eine Transitionsbedingung muss den Wert TRUE oder FALSE haben. Somit kann sie aus einer Boolschen Variablen, Boolschen Adresse oder einer Boolschen Konstante bestehen.

Sie kann auch eine Folge von Instruktionen mit einem Boolschen Ergebnis in ST-Syntax (z.B. (i <= 100) AND b) oder einer beliebigen Sprache enthalten (siehe 'Extras' 'Zoom Aktion/Transition').

Aber eine Transition darf keine Programme, Funktionsblöcke oder Zuweisungen enthalten!

Neben Transitionen kann auch der Tipp-Modus benutzt werden, um zum nächsten Schritt weiterzuschalten, siehe SFCtip und SFCtipmode.

Aktiver Schritt

Nach dem Aufruf des AS-Bausteins wird zunächst die zum Initialschritt (doppelt umrandet) gehörende Aktion ausgeführt. Ein Schritt, dessen Aktion ausgeführt wird, heißt aktiv. Falls der Schritt aktiv ist, wird die zugehörige Aktion einmal pro Zyklus ausgeführt. Im Online Modus werden aktive Schritte blau dargestellt.

In einem Steuerungszyklus werden alle Aktionen ausgeführt, die zu aktiven Schritten gehören. Danach werden die jeweils nachfolgenden Schritte der aktiven Schritte aktiv, wenn die Transitionsbedingungen der nachfolgenden Schritte TRUE sind. Die jetzt aktiven Schritte werden erst im nächsten Zyklus ausgeführt.

Enthält der aktive Schritt eine Ausgangsaktion, wird auch diese erst im nächsten Zyklus ausgeführt, vorausgesetzt, die darauffolgende Transition ist TRUE.

IEC- Schritte

Neben den vereinfachten Schritten stehen die normkonformen IEC-Schritte in AS zur Verfügung. Um IEC-Schritte verwenden zu können, müssen Sie in Ihr Projekt die Bibliothek **TcSystem.Lib** einbinden.

Einem IEC-Schritt können beliebig viele Aktionen zugewiesen werden. IEC-Aktionen sind nicht wie bei den vereinfachten Schritten als Eingangs- Schritt- oder Ausgangsaktion fest einem Schritt zugeordnet, sondern liegen getrennt von den Schritten vor und können innerhalb ihres Bausteins mehrfach verwendet werden. Dazu müssen sie mit dem Befehl 'Extras"Aktion assoziieren' zu den gewünschten Schritten assoziiert werden.

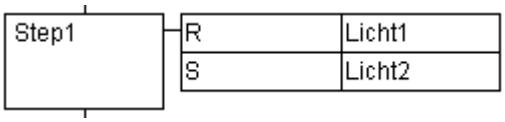
Neben Aktionen können auch boolsche Variablen zu Schritten zugewiesen werden.

Über sogenannte Bestimmungszeichen (Qualifier) wird die Aktivierung und Deaktivierung der Aktionen und boolschen Variablen gesteuert. Zeitliche Verzögerungen sind dabei möglich. Da eine Aktion immer noch aktiv sein kann, wenn bereits der nächste Schritt abgearbeitet wird, z.B. durch Bestimmungszeichen S (Set), kann man Nebenläufigkeiten erreichen.

Eine assoziierte boolsche Variable wird bei jedem Aufruf des AS-Bausteines gesetzt oder zurückgesetzt. Das heißt, ihr wird jedes mal entweder der Wert TRUE oder FALSE neu zugewiesen.

Die assoziierten Aktionen zu einem IEC-Schritt werden rechts vom Schritt in einem zweigeteilten Kästchen dargestellt. Das linke Feld enthält den Qualifier, evtl. mit Zeitkonstanten und das rechte den Aktionsnamen.

Beispiel für einen IEC-Schritt mit zwei Aktionen:



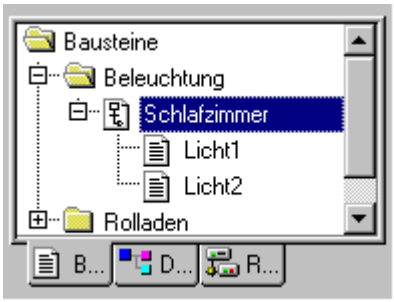
Zur leichteren Verfolgung der Vorgänge werden alle aktiven Aktionen im Onlinebetrieb, wie die aktiven Schritte blau dargestellt. Nach jedem Zyklus wird überprüft, welche Aktionen aktiv sind.

Beachten Sie hierzu auch die Einschränkung bei der Verwendung von Zeit-Qualifiern bei mehrmals verwendeten Aktionen im gleichen Zyklus !

Wird eine Aktion deaktiviert, so wird sie noch einmal ausgeführt. Das heißt, dass jede Aktion mindestens zweimal ausgeführt wird (auch eine Aktion mit Qualifier P). Bei einem Aufruf werden zuerst die deaktivierten Aktionen in alphabetischer Reihenfolge abgearbeitet und dann alle aktiven Aktionen wiederum in alphabetischer Reihenfolge.

Ob es sich bei einem neu eingefügten Schritt um einen IEC-Schritt handelt, ist abhängig davon, ob der Menübefehl ‚Extras"IEC-Schritte benutzen' angewählt ist.

Im Object Organizer hängen die Aktionen direkt unter ihrem jeweiligen AS-Baustein. Neue Aktionen können mit ‚Projekt"Aktion hinzufügen' erzeugt werden.



Zum Assoziieren der Aktionen zu IEC-Schritte stehen folgende Qualifier (Bestimmungszeichen) zur Verfügung.

N	Non-stored	die Aktion ist solange aktiv wie der Schritt
R	overriding Reset	die Aktion wird deaktiviert
S	Set (Stored)	die Aktion wird aktiviert und bleibt bis zu einem Reset aktiv
L	time Limited	die Aktion wird für eine bestimmte Zeit aktiviert
D	time Delayed	die Aktion wird nach einer bestimmten Zeit aktiv, sofern der Schritt noch aktiv ist
P	Pulse	die Aktion wird genau einmal ausgeführt, wenn der Schritt aktiv wird
SD	Stored and time Delayed	die Aktion wird nach einer bestimmten Zeit aktiviert und bleibt bis zu einem Reset aktiv
DS	Delayed and Stored	die Aktion wird nach einer bestimmten Zeit aktiviert, sofern der Schritt noch aktiv ist und bleibt bis zu einem Reset aktiv
SL	Stored and time Limited	die Aktion ist wird eine bestimmte Zeit aktiviert

Wird dieselbe Aktion in zwei unmittelbar aufeinanderfolgenden Schritten mit Qualifiern benutzt, die den zeitlichen Ablauf beeinflussen, kann bei der zweiten Verwendung der Zeit-Qualifier nicht mehr wirksam werden. Um dies zu umgehen, muss ein Zwischenschritt eingefügt werden, so dass in dem dann zusätzlich zu durchlaufenden Zyklus der Aktionszustand erneut initialisiert werden kann

Impilizite Variablen in AS

In der AS gibt es implizit deklarierte Variablen, die verwendet werden können. Zu jedem Schritt gehört ein Flag, welches den Zustand des Schritts speichert. Das Schritt-Flag (aktiver oder inaktiver Zustand des Schritts) heißt <StepName>.x bei IEC-Schritten bzw. nur <StepName> bei den vereinfachten Schritten. Diese boolsche Variable hat den Wert TRUE, wenn der zugehörige Schritt aktiv ist, und FALSE, wenn er inaktiv ist. Sie kann in jeder Aktion und Transition des AS-Bausteins benutzt werden.
Ob eine IEC-Aktion aktiv ist oder nicht, kann man abfragen mit der Variablen <AktionsName>.x.

Bei den IEC-Schritten kann mit den impliziten Variablen <StepName>.t die aktive Zeitdauer der Schritte abgefragt werden.
Auf die impliziten Variablen kann auch von anderen Programmen aus zugegriffen werden. Beispiel: boolvar1:=sfc.step1.x; Dabei ist step1.x die implizite boolsche Variable, die den Zustand von IEC-Schritt step1 im Baustein sfc1 darstellt.

Alternativzweig

Zwei oder mehr Zweige in AS können als Alternativverzweigungen definiert werden. Jeder Alternativzweig muss mit einer Transition beginnen und enden. Alternativverzweigungen können Parallelverzweigungen und weitere Alternativverzweigungen beinhalten. Eine Alternativverzweigung beginnt an einer horizontalen Linie (Alternativanfang) und endet an einer horizontalen Linie (Alternativende) oder mit einem Sprung.

Wenn der Schritt, der der Alternativanfangslinie vorangeht, aktiv ist, dann wird die erste Transition jeder Alternativverzweigung von links nach rechts ausgewertet. Die erste Transition von links, deren Transitionsbedingung den Wert TRUE hat, wird geöffnet und die nachfolgenden Schritte werden aktiviert (siehe aktiver Schritt).

Parallelzweig

Zwei oder mehr Verzweigungen in AS können als Parallelverzweigungen definiert werden. Jeder Parallelzweig muss mit einem Schritt beginnen und enden. Parallelverzweigungen können Alternativverzweigungen oder weitere Parallelverzweigungen beinhalten. Eine Parallelverzweigung beginnt bei einer doppelten Linie (Parallelanfang) und endet bei einer doppelten Linie (Parallelende) oder bei einem Sprung.

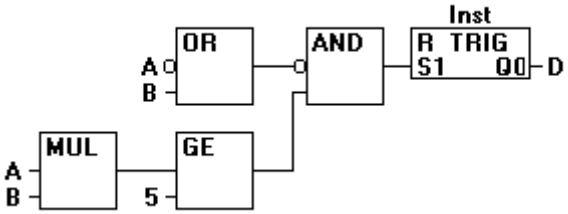
Wenn der der Parallelanfangs-Linie vorangehende Schritt aktiv ist, und die Transitionsbedingung nach diesem Schritt den Wert TRUE hat, dann werden die ersten Schritte aller Parallelverzweigungen aktiv (siehe aktiver Schritt). Diese Zweige werden nun alle parallel zueinander abgearbeitet. Der Schritt nach der Parallelende-Linie wird aktiv, wenn alle vorangehenden Schritte aktiv sind, und die Transitionsbedingung vor diesem Schritt den Wert TRUE liefert.

Sprung

Ein Sprung ist eine Verbindung zu dem Schritt, dessen Name unter dem Sprungsymbol angegeben ist. Sprünge werden benötigt, weil es nicht erlaubt ist, nach oben führende oder sich überkreuzende Verbindungen zu schaffen.

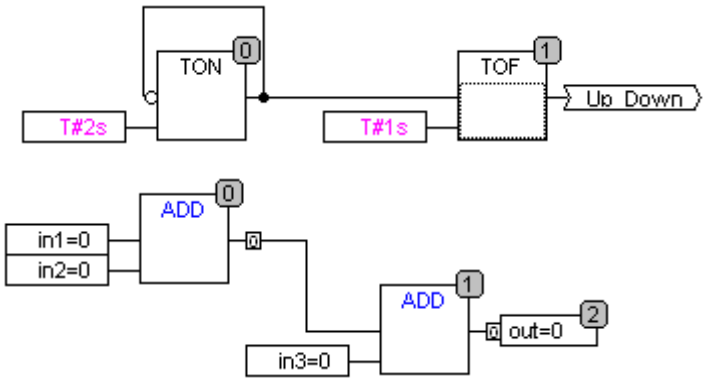
2.2.5 Funktionsplan (FUP)

Der Funktionsplan ist eine graphisch orientierte Programmiersprache. Er arbeitet mit einer Liste von Netzwerken, wobei jedes Netzwerk eine Struktur enthält, die jeweils einen logischen bzw. arithmetischen Ausdruck, den Aufruf eines Funktionsblocks, einen Sprung oder eine Return-Anweisung darstellt. Ein Beispiel für ein Netzwerk im Funktionsplan, wie es in TwinCAT PLC Control typischerweise aussehen könnte:



2.2.6 Freigraphischer Funktionsplaneditor (CFC)

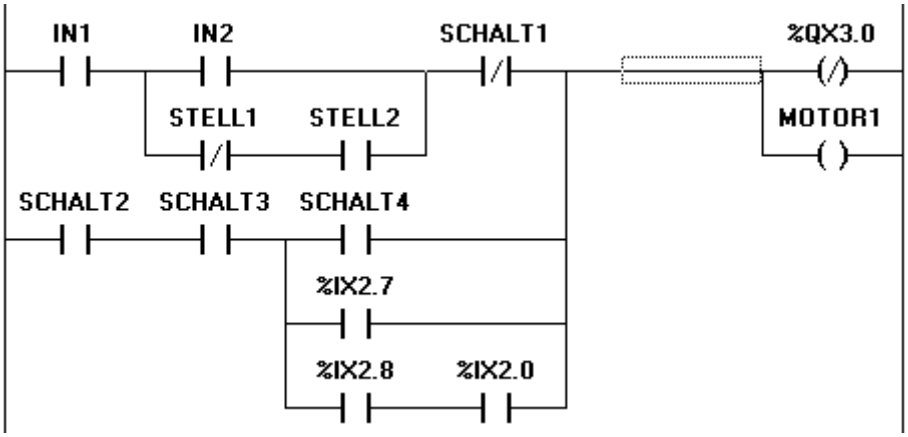
Der freigraphische Funktionsplaneditor arbeitet nicht wie der Funktionsplan FUP mit Netzwerken, sondern mit frei platzierbaren Elementen. Dies erlaubt beispielsweise Rückkoppelungen.
Beispiele für ein Netzwerk im Freigraphischen Funktionsplaneditor könnten typischerweise so aussehen:



2.2.7 Kontaktplan (KOP)

Der Kontaktplan ist eine ebenfalls graphisch orientierte Programmiersprache, die dem Prinzip einer elektrischen Schaltung angenähert ist. Einerseits eignet sich der Kontaktplan dazu, logische Schaltwerke zu konstruieren, andererseits kann man aber auch Netzwerke wie im FUP erstellen. Daher kann der KOP sehr gut dazu benutzt werden, um den Aufruf von anderen Bausteinen zu steuern. Der Kontaktplan besteht aus einer Folge von Netzwerken. Ein Netzwerk wird auf der linken und rechten Seite von einer linken und einer rechten vertikalen Stromleitung begrenzt. Dazwischen befindet sich ein Schaltplan aus Kontakten, Spulen und Verbindungslinien. Jedes Netzwerk besteht auf der linken Seite aus einer Folge von Kontakten, die von links nach rechts den Zustand "AN" oder "AUS" weitergeben, diese Zustände entsprechen den boolschen Werten TRUE und FALSE. Zu jedem Kontakt gehört eine boolsche Variable. Wenn diese Variable TRUE ist, dann wird der Zustand über die Verbindungslinie von links nach rechts weitergegeben, sonst erhält die rechte Verbindung den Wert AUS.

Beispiel für ein Netzwerk im Kontaktplan, wie es in TwinCAT PLC Control typischerweise aussehen könnte:



Kontakt

Jedes Netzwerk im KOP besteht auf der linken Seite aus einem Netzwerk von Kontakten (Kontakte werden dargestellt durch zwei parallele Linien: | |), die von links nach rechts den Zustand "An" oder "Aus" weitergeben. Diese Zustände entsprechen den boolschen Werten TRUE und FALSE. Zu jedem Kontakt gehört eine boolsche Variable. Wenn diese Variable TRUE ist, dann wird der Zustand über die Verbindungslinie von links nach rechts weitergegeben, sonst erhält die rechte Verbindung den Wert "Aus". Kontakte können parallel geschaltet sein, dann muss einer der Parallelzweige den Wert "An" übergeben, damit die Parallelverzweigung den Wert "An" übergibt, oder die Kontakte sind in Reihe geschaltet, dann müssen alle Kontakte den Zustand "An" übergeben, damit der letzte Kontakt den Zustand "An" weitergibt. Dies entspricht also einer elektrischen Parallel- bzw. Reihenschaltung. Ein Kontakt kann auch negiert sein, erkennbar am Schrägstrich im Kontaktsymbol: |/|. Dann wird der Wert der Linie weitergegeben, wenn die Variable FALSE ist.

Spule

Auf der rechten Seite eines Netzwerks im KOP befindet sich eine beliebige Anzahl sogenannter Spulen, dargestellt durch Klammern:( ). Diese können nur parallel geschaltet werden. Eine Spule gibt den Wert der Verbindungen von links nach rechts weiter, und kopiert ihn in eine zugehörige boolsche Variable. An der Eingangslinie kann der Wert AN (entspricht der boolschen Variablen TRUE) oder der Wert AUS anliegen (entsprechend FALSE). Kontakte und Spulen können auch negiert werden (im Beispiel ist z.B. der Kontakt SCHALT1 und die Spule %QX3.0 negiert). Wenn eine Spule negiert ist (erkennbar am Schrägstrich im Spulensymbol: (/)), dann kopiert sie den negierten Wert in die zugehörige boolsche Variable. Wenn ein Kontakt negiert ist, dann schaltet er nur dann durch, wenn die zugehörige boolsche Variable FALSE ist.

Funktionsblöcke im Kontaktplan

Neben Kontakten und Spulen können Sie auch Funktionsblöcke und Programme eingeben, diese müssen im Netzwerk einen Eingang und einen Ausgang mit boolschen Werten haben und können an denselben Stellen verwendet werden wie Kontakte, d. h. auf der linken Seite des KOP-Netzwerks.

Set/Reset-Spulen

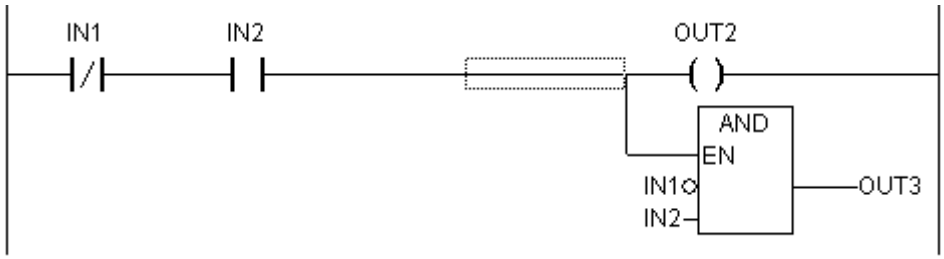
Spulen können auch als Set oder Reset-Spulen definiert sein. Eine Set-Spule (erkennbar am ‚S' im Spulensymbol: (S)) überschreibt in der zugehörigen boolschen Variablen niemals den Wert TRUE. D.h., wenn die Variable einmal auf TRUE gesetzt wurde, dann bleibt sie es auch. Eine Reset-Spule (erkennbar am ‚R' im Spulensymbol: (R)), überschreibt in der zugehörigen boolschen Variablen niemals den Wert FALSE: Wenn die Variable einmal auf FALSE gesetzt wurde, dann bleibt sie es auch.

KOP als FUP

Beim Arbeiten mit dem KOP kann leicht der Fall auftreten, dass Sie das Ergebnis der Kontaktschaltung zur Steuerung anderer Bausteine nutzen wollen. Dann können Sie einerseits das Ergebnis mit Hilfe der Spulen in einer globalen Variable ablegen, die an anderer Stelle weiter benutzt wird. Sie können aber auch den eventuellen Aufruf direkt in Ihr KOP-Netzwerk einbauen. Dazu führen Sie einen Baustein mit EN-Eingang ein. Solche Bausteine sind ganz normale Operanden, Funktionen, Programme oder Funktionsblöcke, die einen zusätzlichen Eingang haben, der mit EN beschriftet ist. Der EN-Eingang ist immer vom Typ BOOL und seine Bedeutung ist: der Baustein mit EN-Eingang wird dann ausgewertet, wenn EN den Wert TRUE hat. Ein EN-Baustein wird parallel zu den Spulen geschaltet, wobei der EN-Eingang mit der Verbindungslinie

zwischen den Kontakten und den Spulen verbunden wird. Wenn über diese Linie die Information AN transportiert wird, dann wird dieser Baustein ganz normal ausgewertet. Ausgehend von einem solchen EN-Baustein können Netzwerke wie in FUP erstellt werden.

Teil eines KOP-Netzwerks mit einem EN-Baustein



2.2.8 Debugging, Onlinefunktionen

Traceaufzeichnung [▶ 184]

Die Traceaufzeichnung bietet die Möglichkeit, den Werteverlauf von Variablen aufzuzeichnen, abhängig vom sogenannten Trigger-Ereignis. Dieses ist die steigende oder fallende Flanke einer vorher definierten boolschen Variablen (der Trigger-Variablen). Das TwinCAT PLC Control ermöglicht die Aufzeichnung von bis zu 20 Variablen. Ein Ringpuffer von 64 kB steht zur Verfügung.

Debugging [▶ 107]

Mit den Debugging-Funktionen von TwinCAT PLC Control wird Ihnen das Auffinden von Fehlern erleichtert.

Breakpoint

Ein Breakpoint ist eine Stelle im Programm, an der die Abarbeitung angehalten wird. Somit ist es möglich, die Werte von Variablen an einer bestimmten Programmstelle zu betrachten.
Breakpoints können in allen Editoren gesetzt werden. In den Texteditoren werden Breakpoints auf Zeilennummern gesetzt, in FUP und KOP auf Netzwerknummern, im CFC auf Bausteine und in AS auf Schritte.

In Funktionsblockinstanzen können keine Breakpoints gesetzt werden.

Einzelschritt

Einzelschritt bedeutet:

- in AWL: Das Programm bis zum nächsten CAL, LD oder JMP-Befehl ausführen.
- in ST: Die nächste Anweisung ausführen.
- in FUP, KOP: Das nächste Netzwerk ausführen
- in AS: Die Aktion zum nächsten Schritt ausführen.
- in CFC: Den nächsten Baustein (Box) im CFC-Programm ausführen.

Durch schrittweise Abarbeitung können Sie die logische Korrektheit Ihres Programms überprüfen.

Einzelzyklus

Wenn Einzelzyklus gewählt wurde, dann wird nach jedem Zyklus die Abarbeitung angehalten.

Werte Online verändern

Variablen können im laufenden Bertieb einmalig auf einen bestimmten Wert gesetzt werden (Wert schreiben) oder auch nach jedem Zyklus wieder neu mit einem bestimmten Wert beschrieben werden (Wert forcen). Sie können den Variablenwert im Online-Betrieb auch verändern, indem Sie einen Doppelklick darauf durchführen. Boolsche Variablen wechseln dadurch von TRUE auf FALSE bzw. umgekehrt, für alle anderen erhalten Sie einen Dialog 'Variable xy schreiben', in dem Sie den aktuellen Variablenwert editieren können.

Monitoring

Im Online Modus werden für alle am Bildschirm sichtbaren Variablen laufend die aktuellen Werte aus der Steuerung gelesen und dargestellt. Diese Darstellung finden Sie im Deklarations- und Programmeditor, außerdem können Sie im Watch- und Rezepturmanager aktuelle Variablenwerte ausgeben. Sollen Variablen aus Funktionsblock-Instanzen gemonitort werden, muss erst die entsprechende Instanz geöffnet werden.

Beim **Monitoring von VAR_IN_OUT Variablen** wird der dereferenzierte Wert ausgegeben.
Mit einfachem Klick auf das Kreuz oder mit Doppelklick auf die Zeile wird die Anzeige expandiert bzw. kollabiert.

Beim **Monitoring von Pointern** wird im Deklarationsteil sowohl der Pointer als auch der dereferenzierte Wert ausgegeben. Im Programmteil wird nur der Pointer ausgegeben:

+ --pointervar = ‘<‘pointervalue‘>‘

POINTER im dereferenzierten Wert werden ebenfalls entsprechend angezeigt.

Mit einfachem Klick auf das Kreuz oder mit Doppelklick auf die Zeile wird die Anzeige expandiert bzw. kollabiert.

Monitoring von ARRAY-Komponenten: Zusätzlich zu Array-Komponenten, die über eine Konstante indiziert sind, werden auch Komponenten angezeigt, die über eine Variable indiziert sind:
anarray[1] = 5
anarray[i] = 1
Besteht der Index aus einem Ausdruck (z.B. [i+j] oder [i+1]), kann die Komponente nicht angezeigt werden.

Simulation

Bei der Simulation wird das erzeugte Steuerungsprogramm nicht in der Steuerung abgearbeitet. Es stehen alle Onlinefunktionen zur Verfügung. Sie haben somit die Möglichkeit, die logische Korrektheit Ihres Programmes ohne Steuerungshardware zu testen.

Die Simulation steht Ihnen nur zur Verfügung, wenn Sie in der PLC Konfiguration die BCxx00 Klemmen-SPS (Codegenerierung 80C165) angewählt haben. Bei der SPS auf dem PC (Codegenerierung i386) erfolgt die Simulation direkt im Laufzeitsystem. Hierzu muss kein E/A vorhanden sein.

Logbuch

Das Logbuch zeichnet Benutzeraktionen, interne Vorgänge, Statusänderungen und Ausnahmezustände während des Online-Modus chronologisch auf. Es dient der Überwachung und der Fehlerrückverfolgung.

2.2.9 IEC 61131-3

Die Norm IEC 61131-3 ist ein internationaler Standard für Programmiersprachen von speicherprogrammierbaren Steuerungen. Die in TwinCAT PLC Control realisierten Programmiersprachen sind konform zu den Anforderungen der Norm. Nach diesem Standard besteht ein Programm aus folgenden Elementen:·

- Strukturen
- Bausteine
- Globale Variablen

Ausgewählte Literatur zum Thema IEC 61131-3:

- Peter Wratil: Moderne Programmiertechnik für Automatisierungssysteme. Würzburg: Vogel Buchverlag, 1996. ISBN 3-8023-1575-8
- Karl Pusch: Grundkurs IEC 1131. Würzburg: Vogel Buchverlag, 1999. ISBN 3-8023-1807-2
- Flavio Bonifatti et al.: IEC 1131-3 Programming Methology, Seyssins: CJ International, 1997. ISBN 2-9511585-0-5
- Karl-Heinz John, Michael Tiegelkamp: SPS -Programmierung mit IEC 61131-3, Berlin: Springer-Verlag, 2000. ISBN 3-540-66445-9
- Jens von Aspern: SPS-Softwareentwicklung mit IEC 61131, Heidelberg: Hüthig Verlag, 2000. ISBN 3-7785-2681-2

- R. W. Lewis: Programming industrial control systems using IEC 1131-3, London: IEC Publishing, 1998. ISBN 0 85296 950 3

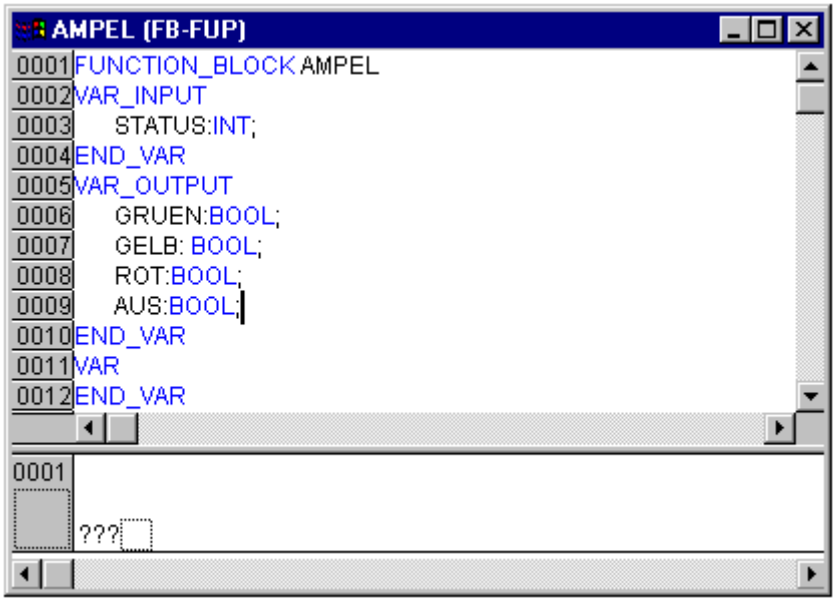
3 Ein Beispielprogramm

Das Beispielprogramm soll zur Steuerung einer Mini-Ampelanlage mit zwei Autoampeln an einer Kreuzung dienen. Beide Ampeln werden sich in ihren rot/grün-Phasen abwechseln, und um Unfälle zu vermeiden, sollen zwischen den Phasen auch noch gelb bzw. gelb/rot-Umschaltphasen vorsehen werden. Letztere werden kürzer dauern als erstere. In diesem Beispiel werden Sie sehen, wie sich zeitabhängige Programme mit den Sprachmitteln der IEC61131-3 darstellen lassen, wie man mit Hilfe vom TwinCAT PLC Control die verschiedenen Sprachen der Norm editiert, und wie man sie problemlos verbinden kann.

Bausteine erzeugen

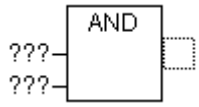
Starten Sie zunächst das TwinCAT PLC Control, und wählen Sie 'Datei' 'Neu'. Als Taskname wird Ihnen ‚Standard' vorgeschlagen. Den ersten Baustein sollten Sie PLC_PRG nennen. Für unseren Fall wählen wir als Sprache dieses Bausteins Ablaufsprache (AS). Erzeugen Sie nun zwei weitere Objekte mit dem Befehl 'Projekt' 'Objekt' 'einfügen' über die Menüleiste oder über das Kontextmenü (rechte Maustaste drücken im Object Organizer). Einen Funktionsblock in der Sprache Funktionsplan (FUP) namens AMPEL, sowie einen Baustein WARTEN, ebenfalls von dem Typ Funktionsblock, den wir als Anweisungsliste (AWL) programmieren wollen.
Im Baustein AMPEL werden wir die einzelnen Ampelphasen den Ampellichtern zuordnen, d.h. wir werden dafür sorgen, dass die rote Lampe bei der Phase rot und bei der Phase gelb/rot leuchtet, die gelbe Lampe bei der Phase gelb und gelb/rot, usw.
In WARTEN werden wir einen einfachen Timer programmieren, der als Eingabe die Dauer der Phase in Millisekunden bekommen wird, und der als Ausgabe TRUE liefert, sobald die Zeit abgelaufen ist.
PLC_PRG schließlich wird das alles miteinander verbinden, so dass das richtige Ampellicht zur richtigen Zeit, und mit der gewünschten Dauer leuchten wird.

Im Deklarationseditor deklarieren Sie als Eingabevariable (zwischen den Schlüsselwörtern VAR_INPUT und END_VAR) eine Variable namens STATUS vom Typ INT. STATUS wird fünf mögliche Zustände haben, nämlich jeweils einen für die Ampelphasen grün, gelb, gelb-rot, rot und aus. Ausgaben hat unsere Ampel dementsprechend vier, nämlich ROT, GELB, GRUEN (Umlaute werden für Variablen nicht akzeptiert) und AUS. Deklarieren Sie diese vier Variablen; dann sieht der Deklarationsteil des Funktionsblocks AMPEL folgendermaßen aus:

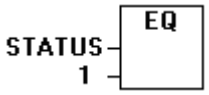


Funktionsblock AMPEL, Deklarationsteil

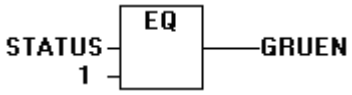
Nun gilt es, aus der Eingabe STATUS des Bausteins die Werte der Ausgabevariablen zu ermitteln. Gehen Sie dazu in den Rumpf des Bausteins. Klicken Sie in das Feld links neben dem ersten Netzwerk (das graue Feld mit der Nummer 1). Sie haben jetzt das erste Netzwerk selektiert. Wählen Sie nun den Menüpunkt 'Einfügen' 'Operator'. Es wird im ersten Netzwerk eine Box mit dem Operator AND und zwei Eingängen eingefügt:



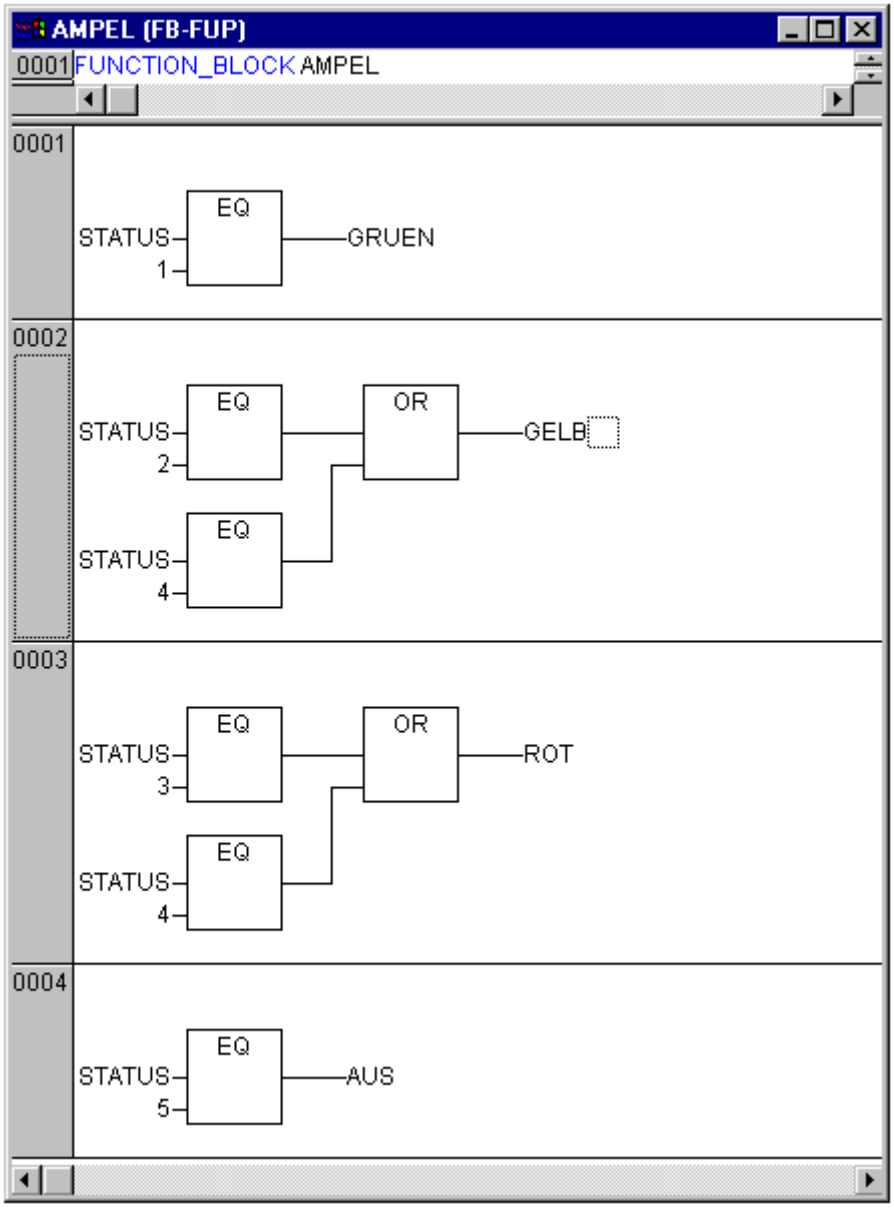
Klicken Sie auf den Text AND mit dem Mauszeiger und ändern Sie den Text in EQ. Selektieren Sie die drei Fragezeichen vom oberen der beiden Eingänge und tragen Sie die Variable STATUS ein. Anschließend selektieren Sie die unteren drei Fragezeichen und überschreiben ihn mit einer 1. Sie erhalten folgendes Netzwerk:



Klicken Sie nun an eine Stelle hinter der EQ Box. Es wird nun der Ausgang der EQ-Operation selektiert. Wählen Sie 'Einfügen' 'Zuweisung'. Die drei Fragezeichen ??? ändern Sie in GRUEN. Sie haben nun ein Netzwerk der folgenden Gestalt erstellt:



STATUS wird mit 1 verglichen, das Ergebnis wird GRUEN zugewiesen. Dieses Netzwerk schaltet also auf GRUEN, wenn der vorgegebene Statuswert 1 ist. Für die anderen Ampelfarben bzw. für AUS benötigen wir drei weitere Netzwerke. Sie erzeugen diese durch den Befehl 'Einfügen' 'Netzwerk (danach)'. Diese Netzwerke sollten Sie wie im Beispiel einrichten. Daraus ergibt sich der fertige Baustein:

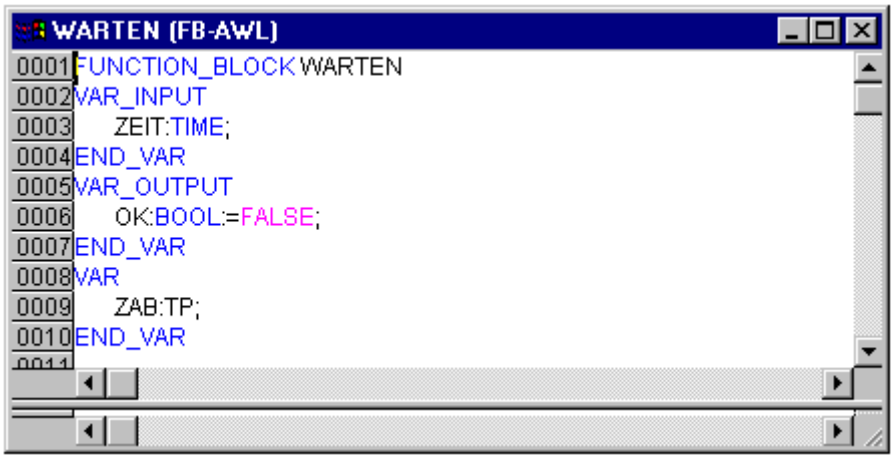


Um vor einem Operator einen weiteren Operator einzufügen, müssen Sie die Stelle selektieren, wo der Eingang, an den Sie den Operator anhängen wollen, in die Box mündet. Anschließend führen Sie ein 'Einfügen' 'Operator' aus. Ansonsten können Sie beim Erstellen dieser Netzwerke ebenso vorgehen wie beim ersten Netzwerk. Nun ist unser erster Baustein bereits fertig. AMPEL steuert uns, je nach Eingabe des Wertes STATUS, die jeweils gewünschte Ampelfarbe.

Für den Timer im Baustein WARTEN benötigen wir einen Baustein aus der Standardbibliothek. Öffnen Sie also den Bibliotheksverwalter mit 'Fenster' 'Bibliotheksverwaltung'. Wählen Sie 'Einfügen' 'Weitere Bibliothek'. Der Dialog zum Öffnen von Dateien erscheint. Aus der Liste der Bibliotheken wählen Sie standard.lib.

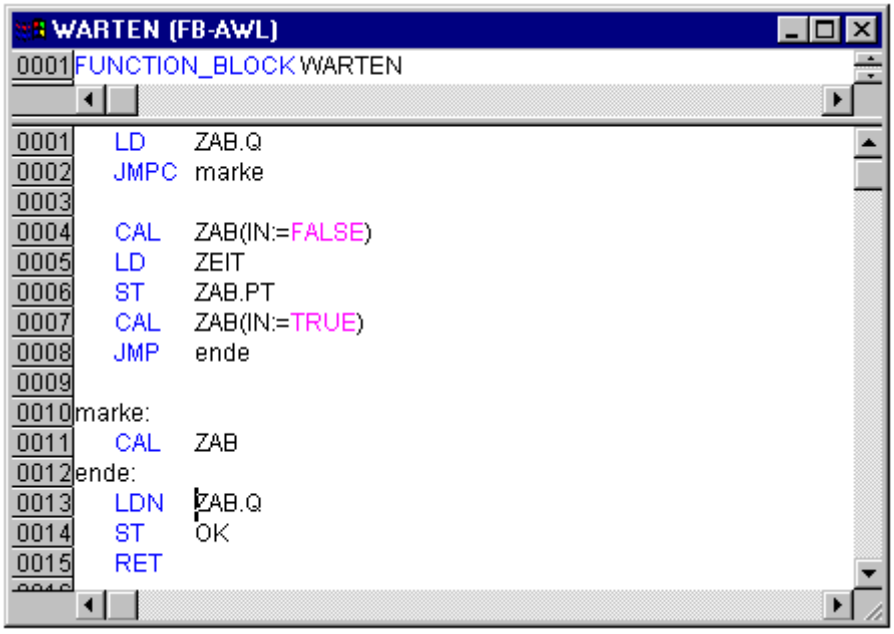
Gehen wir nun zum Baustein WARTEN. Dieser soll ein Timer werden, mit dem wir die Länge jeder Ampelphase angeben können. Unser Baustein erhält als Eingabevariable eine Variable ZEIT vom Typ TIME, und als Ausgabe liefert er einen boolschen Wert, den wir OK nennen wollen, und der TRUE sein soll, wenn die gewünschte Zeit abgelaufen ist. Diesen Wert besetzen wir mit FALSE vor, indem wir an das Ende der Deklaration (aber vor dem Strichpunkt)" := FALSE " einfügen. Für unsere Zwecke benötigen wir den Baustein TP, einen Pulsgeber. Dieser hat zwei Eingänge (IN, PT) und zwei Ausgänge (Q, ET). TP macht nun folgendes: Solange IN FALSE ist, ist ET 0 und Q FALSE. Sobald IN den Wert TRUE liefert, wird im Ausgang ET die Zeit in Millisekunden hochgezählt. Wenn ET den Wert PT erreicht, wird ET nicht mehr weitergezählt. Q liefert unterdessen solange TRUE, solange ET kleiner als PT ist. Sobald der Wert PT erreicht ist, liefert Q wieder FALSE. Übrigens: eine Kurzbeschreibung aller Bausteine aus der

Standardbibliothek finden Sie im Anhang. Um den Baustein TP im Baustein WARTEN verwenden zu können, müssen wir von TP eine lokale Instanz anlegen. Dazu deklarieren wir uns eine lokale Variable ZAB (für Zeit abgelaufen) vom Typ TP (zwischen den Schlüsselwörtern VAR, END_VAR). Der Deklarationsteil von WARTEN sieht somit wie folgt aus:



Funktionsblock WARTEN, Deklarationsteil

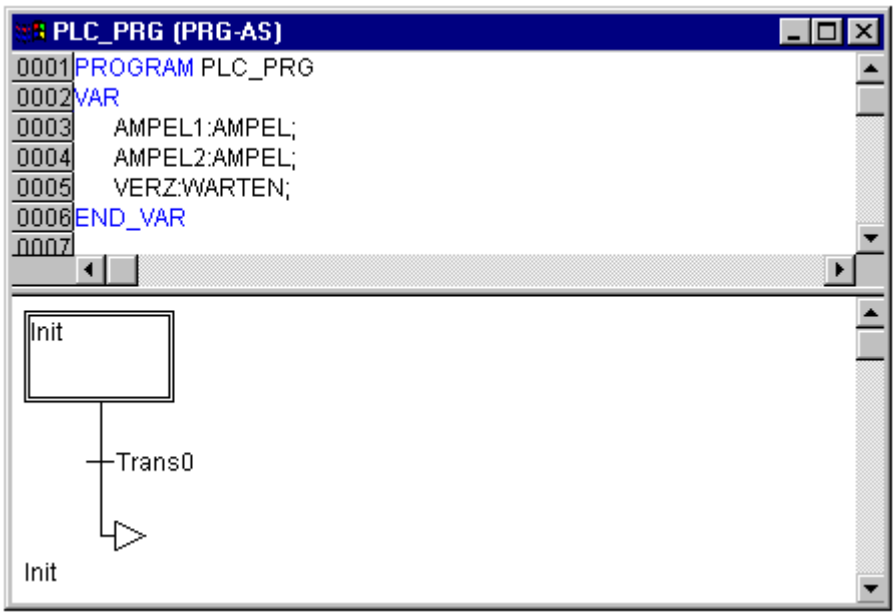
Um den gewünschten Timer zu realisieren, muss der Rumpf des Bausteins wie folgt ausprogrammiert werden:



Funktionsblock WARTEN, Anweisungsteil

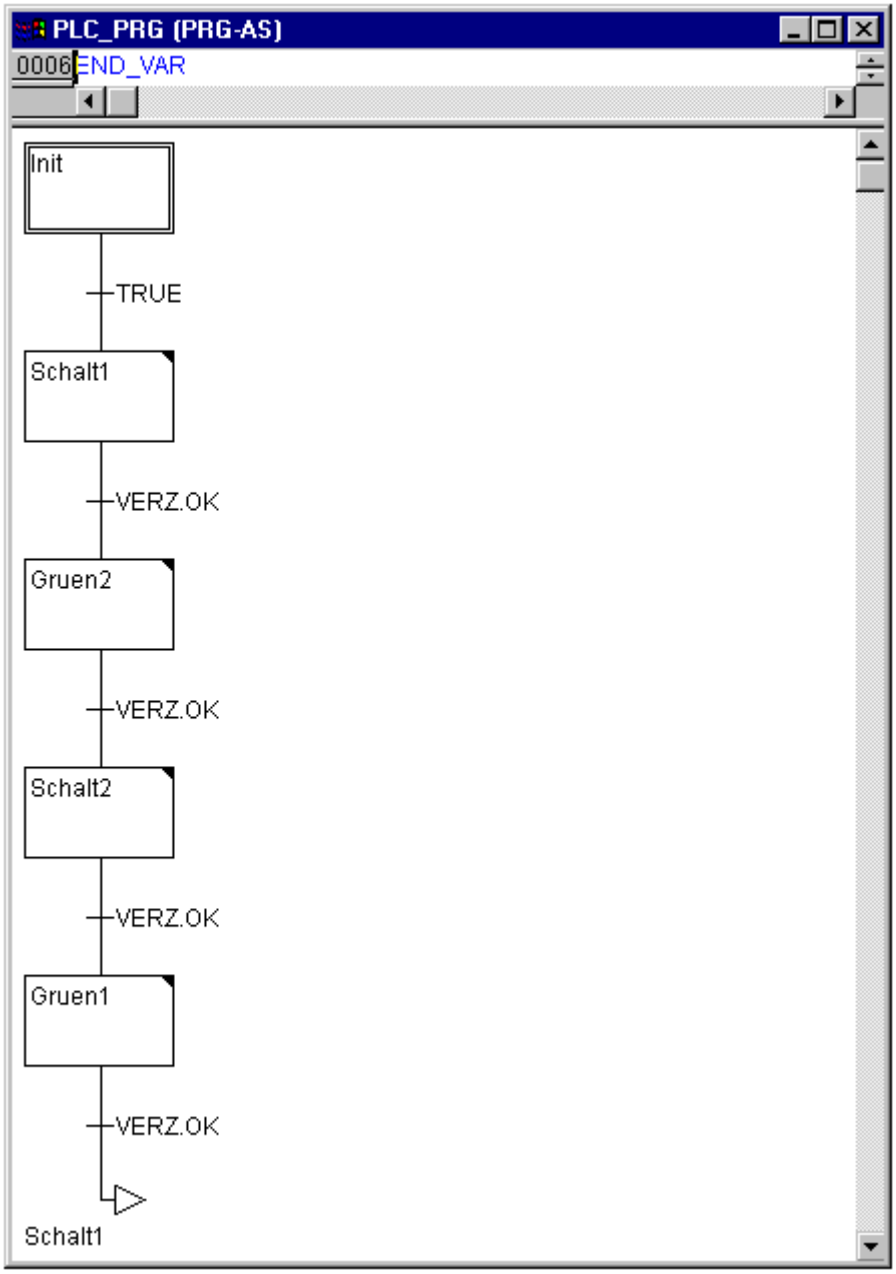
Zunächst wird abgefragt, ob Q bereits auf TRUE gesetzt ist (ob also bereits gezählt wird), in diesem Fall ändern wir nichts an der Belegung von ZAB, sondern rufen den Funktionsblock ZAB ohne Eingabe auf (um zu prüfen, ob die Zeit bereits abgelaufen ist). Andernfalls setzen wir die Variable IN in ZAB auf FALSE, und damit gleichzeitig ET auf 0 und Q auf FALSE. So sind alle Variablen auf den gewünschten Anfangszustand gesetzt. Nun speichern wir die benötigte Zeit aus der Variablen ZEIT in der Variablen PT, und rufen ZAB mit IN:=TRUE auf. Im Funktionsblock ZAB wird nun die Variable ET hochgezählt bis sie den Wert ZEIT erreicht, dann wird Q auf FALSE gesetzt. Der negierte Wert von Q wird nach jedem Durchlauf von WARTEN in OK gespeichert. Sobald Q FALSE ist, liefert also OK TRUE. Der Timer ist hiermit fertig.

Nun gilt es, unsere beiden Funktionsblöcke WARTEN und AMPEL im Hauptprogramm PLC_PRG zusammenzubringen. Zunächst deklarieren wir die Variablen, die wir brauchen. Das sind zwei Instanzen des Funktionsblocks AMPEL (AMPEL1, AMPEL2) und eine vom Typ WARTEN (VERZ wie Verzögerung). PLC_PRG sieht nun folgendermaßen aus:



Programm PLC_PRG, erste Ausbaustufe, Deklarationsteil

Das Anfangsdiagramm eines Bausteins in AS besteht stets aus einer Aktion "Init" einer nachfolgenden Transition "Trans0" und einem Sprung zurück zu Init. Wir sollten das etwas erweitern. Legen wir zunächst die Struktur des Diagramms fest, bevor wir die einzelnen Aktionen und Transitionen programmieren. Zuerst benötigen wir für jede Ampelphase einen Schritt. Fügen Sie diesen ein, indem Sie Trans0 markieren, und 'Einfügen' 'Schritt-Transition (danach)' wählen. Wiederholen Sie diesen Vorgang noch dreimal. Wenn Sie direkt auf den Namen einer Transition oder eines Schrittes klicken, dann wird dieser markiert, und Sie können ihn verändern. Nennen Sie die erste Transition nach Init "TRUE", alle anderen Transitionen "VERZ.OK". Die erste Transition schaltet also immer durch, alle anderen dann, wenn VERZ in OK TRUE ausgibt, also wenn die eingegebene Zeit abgelaufen ist. Die Schritte erhalten (von oben nach unten) die Namen Schalt1, Gruen2, Schalt2, Gruen1, wobei Init seinen Namen natürlich behält. "Schalt" soll jedes Mal eine Gelbphase bedeuten, bei Gruen1 wird AMPEL1 bei Gruen2 AMPEL2 grün sein. Ändern Sie zuletzt noch die Rücksprungadresse von Init nach Schalt1. Wenn Sie alles richtig gemacht haben, dann müsste das Diagramm nun folgendermaßen aussehen:

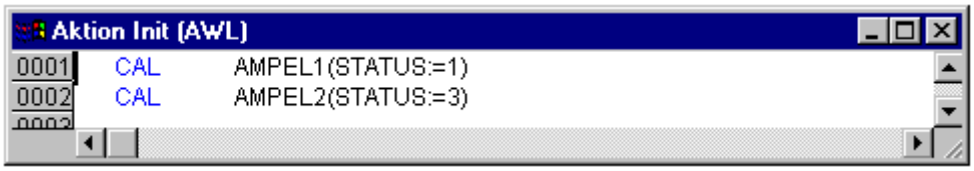


Programm PLC_PRG, erste Ausbaustufe, Anweisungsteil

Nun müssen wir die einzelnen Schritte ausprogrammieren. Wenn Sie auf dem Feld eines Schrittes einen Doppelklick ausführen, öffnen Sie einen Dialog zum Öffnen einer neuen Aktion. In unserem Fall werden wir als Sprache jeweils AWL (Anweisungsliste) verwenden.

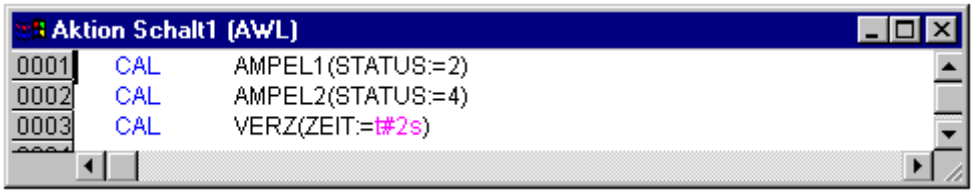
Aktionen und Transitionsbedingungen

In der Aktion zum Schritt Init werden die Variablen initialisiert, der STATUS von AMPEL1 soll 1 (grün) sein. Der Status von AMPEL2 soll 3 (rot) sein. Die Aktion Init sieht dann so aus:



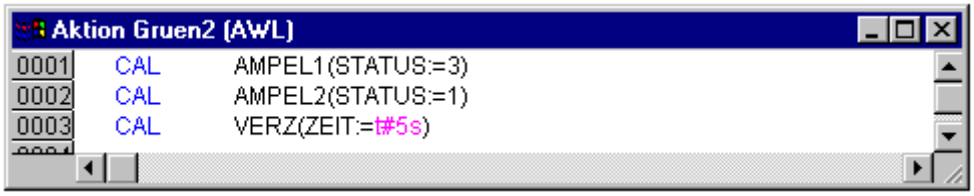
Aktion Init

Bei Schalt1 wechselt der STATUS von AMPEL1 auf 2 (gelb), der von AMPEL2 auf 4 (gelb-rot). Außerdem wird nun eine Verzögerungszeit von 2000 Millisekunden festgelegt. Die Aktion sieht nun wie folgt aus:



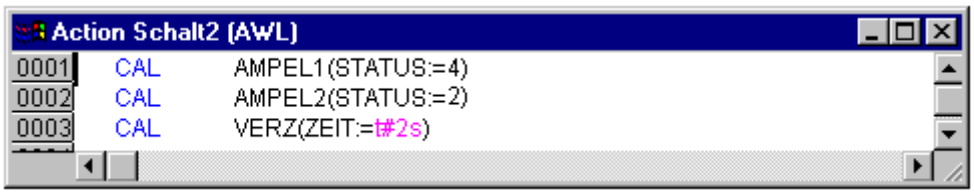
Aktion Schalt1

Bei Gruen2 ist AMPEL1 rot (STATUS:=3), AMPEL2 grün (STATUS:=1), und die Verzögerungszeit ist auf 5000 Millisekunden gesetzt.



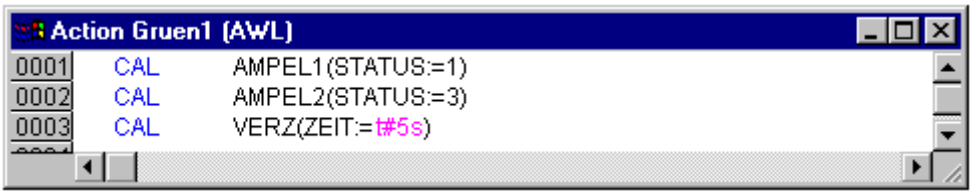
Aktion Gruen2

Bei Schalt2 wechselt der STATUS von AMPEL1 auf 4 (gelb-rot), der von AMPEL2 auf 2 (gelb). Es wird nun eine Verzögerungszeit von 2000 Millisekunden festgelegt.



Aktion Schalt2

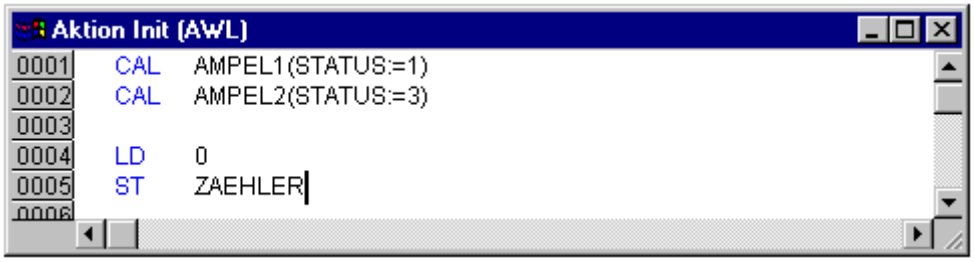
Bei Gruen1 ist AMPEL1 grün (STATUS:=1), AMPEL2 rot (STATUS:=3), und die Verzögerungszeit wird auf 5000 Millisekunden eingestellt.



Aktion Gruen1

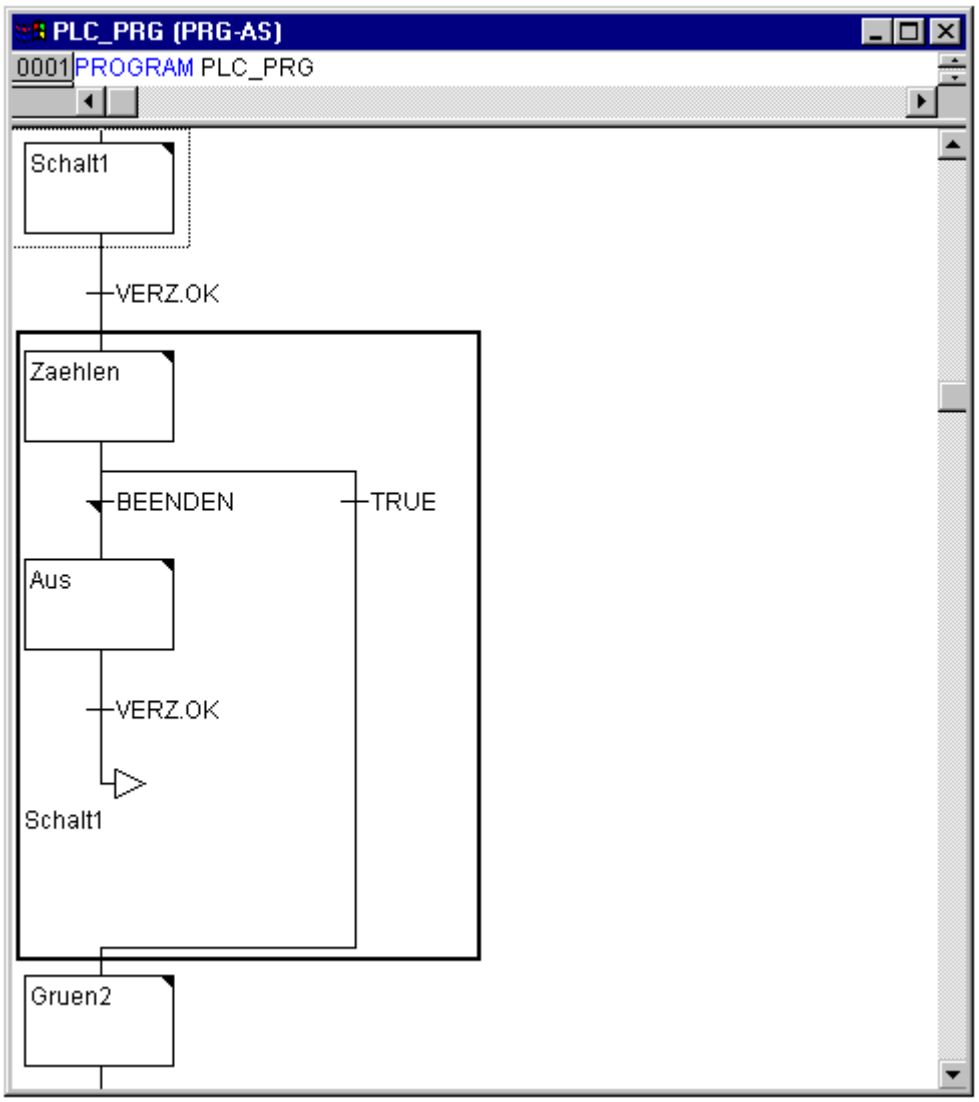
Damit ist die erste Ausbauphase unseres Programms beendet. Sie können es nun übersetzen und auch in der Simulation testen.

Damit sich in unserem Diagramm wenigstens eine Alternativverzweigung befindet, und damit wir unsere Ampelanlage nachts abstellen können, bauen wir in unser Programm nun einen Zähler ein, der nach einer bestimmten Zahl von Ampelzyklen die Anlage abstellt. Zunächst brauchen wir also eine neue Variable ZAEHLER vom Typ INT. Deklarieren Sie diese wie gehabt im Deklarationsteil von PLC_PRG, und initialisieren Sie sie in Init mit 0.



Aktion Init, zweite Fassung

Markieren Sie nun die Transition nach Schalt1 und fügen Sie einen Schritt und eine Transition danach ein. Markieren Sie die neu entstandene Transition und fügen Sie eine Alternativverzweigung links davon ein. Fügen Sie nach der linken Transition einen Schritt und eine Transition ein. Fügen Sie nach der nun neu entstandenen Transition einen Sprung nach Schalt1 ein. Wählen Sie die Transition unterhalb des ersten neuen Schrittes aus. Öffnen Sie das Kontextmenü und wählen Sie "Zoom Aktion/Transition" aus. Als Sprache für die Transition wählen Sie bitte AWL. Benennen Sie die neu entstandenen Teile wie folgt: Der obere der beiden neuen Schritte soll "Zählen" heißen, der untere "Aus". Die Transitionen heißen (von oben nach unten und von links nach rechts) BEENDEN, TRUE und VERZ.OK. Der neu entstandene Teil sollte also so aussehen wie der hier schwarz umrandete Teil:

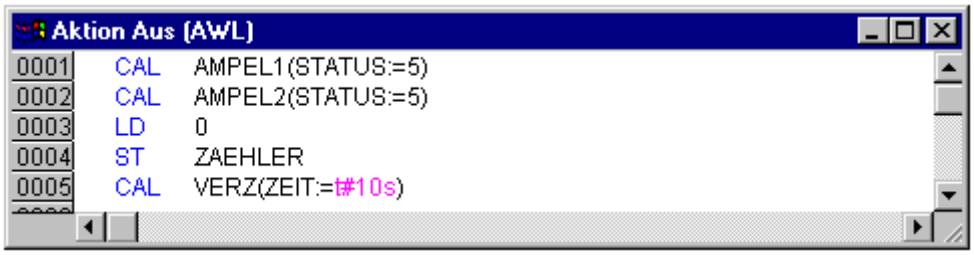


Ampelanlage

Es gibt also zwei neue Aktionen und eine neue Transitionsbedingung zu implementieren. Beim Schritt Zaehlen geschieht nichts anderes, als dass ZAEHLER um eins erhöht wird:

Die Transition BEENDEN überprüft, ob der Zähler größer als eine bestimmte Zahl ist, sagen wir mal 7:

Bei Aus wird der Status beider Ampeln auf 5 (AUS) gesetzt, der ZAEHLER wird auf 0 zurückgesetzt, und eine Verzögerungszeit von 10 Sekunden festgelegt:

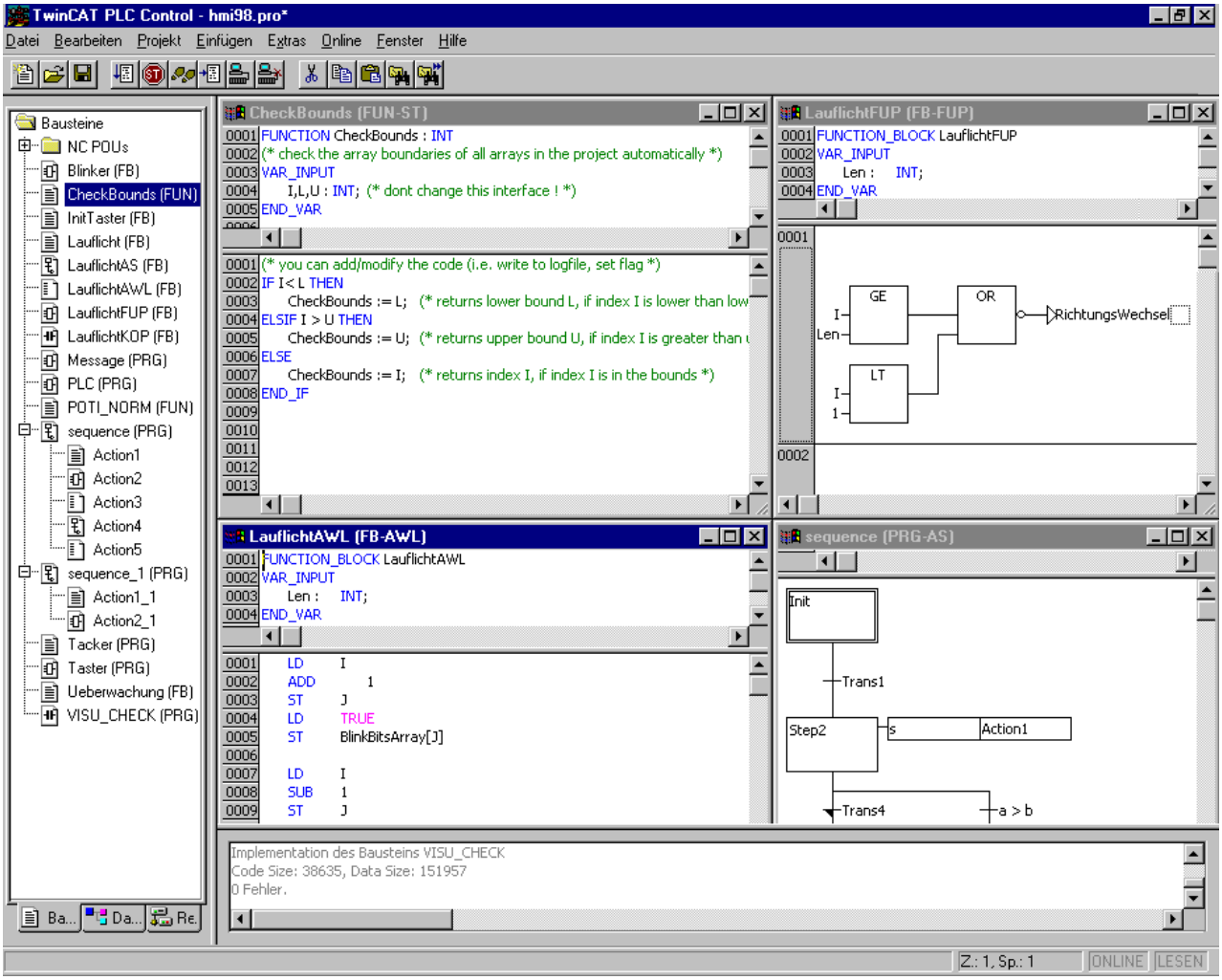


In unserer Ampelstadt wird es also nach sieben Ampelzyklen Nacht, für zehn Sekunden schaltet die Ampel sich aus, dann wird es wieder Tag, die Ampelanlage schaltet sich wieder ein, und das ganze geht wieder von vorn los.

Testen Sie nun Ihr Programm. Dazu müssen Sie es übersetzen ('Projekt' 'Alles übersetzen') und laden ('Online' 'Einloggen' und dann 'Online' 'Laden'). Wenn Sie nun 'Online' 'Start' ausführen, können Sie die zeitliche Abfolge der einzelnen Schritte ihres Hauptprogramms verfolgen. Das Fenster des Bausteins PLC_PRG hat sich nunmehr zum Monitor Fenster gewandelt. Mit Doppelklick auf das Pluszeichen im Deklarationseditor klappt die Variablendarstellung auf und Sie können die Werte der einzelnen Variablen beobachten.

4 Die Komponenten im Einzelnen

4.1 Hauptfenster

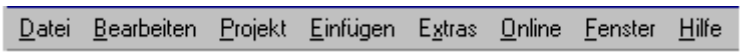


Folgende Elemente befinden sich im Hauptfenster von TwinCAT PLC Control (von oben nach unten):

- Die Menüleiste
- Die Funktionsleiste (optional) mit Schaltflächen zur schnelleren Ausführung von Menübefehlen.
- Der Object Organizer mit Registerkarten für Bausteine, Datentypen, Ressourcen und Visualisierung.
- Ein vertikaler Bildschirmteiler zwischen dem Object Organizer und dem Arbeitsbereich von TwinCAT PLC Control
- Der Arbeitsbereich, in dem sich die Editorfenster befinden
- Das Meldungsfenster (optional)
- Die Statusleiste (optional); mit Informationen über den derzeitigen Zustand des Projekts

Menüleiste

Die Menüleiste befindet sich am oberen Rand des Hauptfensters. Sie enthält alle Menübefehle.



Funktionsleiste

Durch Klicken mit der Maus auf ein Symbol ermöglicht die Funktionsleiste eine schnellere Auswahl eines Menübefehls. Die Auswahl der zur Verfügung gestellten Symbole passt sich automatisch an das aktive Fenster an. Der Befehl wird nur ausgeführt, wenn die Maustaste über das Symbol gedrückt und auch wieder ausgelassen wird. Wenn Sie den Mauszeiger eine kurze Zeit über einem Symbol in der Funktionsleiste halten, wird der Name des Symbols in einem Tooltip angezeigt. Die Anzeige der Funktionsleiste ist optional (siehe 'Projekt' 'Optionen' Kategorie Arbeitsbereich).

Object Organizer

Der Object Organizer befindet sich immer an der linken Seite von TwinCAT PLC Control. Unten befinden sich drei Registerkarten mit Symbolen für die drei Objektarten Bausteine, Datentypen und Ressourcen. Zum Wechseln zwischen den jeweiligen Objektarten klicken sie mit der Maus auf die entsprechende Registerkarte oder benutzen Sie die linke bzw. rechte Pfeiltaste. Wie Sie mit den Objekten im Object Organizer arbeiten, erfahren Sie im Kapitel Objekte [▶ 89].

Object Organizer

Bildschirmteiler

Der Bildschirmteiler ist die Grenze zwischen zwei nicht überlappenden Fenstern. In TwinCAT PLC Control gibt es Bildschirmteiler zwischen dem Object Organizer und dem Arbeitsbereich des Hauptfensters, zwischen der Schnittstelle (Deklarationsteil) und der Implementierung (Anweisungsteil) von Bausteinen und zwischen dem Arbeitsbereich und dem Meldungsfenster. Wenn Sie den Mauszeiger auf den Bildschirmteiler führen, können Sie damit den Bildschirmteiler verschieben. Dies geschieht durch Bewegen der Maus bei gedrückter linker Maustaste. Beachten Sie, dass der Bildschirmteiler stets an seiner absoluten Position bleibt, auch wenn die Fenstergröße verändert wird. Wenn der Bildschirmteiler nicht mehr vorhanden zu sein scheint, dann vergrößern Sie einfach Ihr Fenster.

Arbeitsbereich

Der Arbeitsbereich befindet sich an der rechten Seite im TwinCAT PLC Control-Hauptfenster. Alle Editoren für Objekte und der Bibliotheksverwaltung werden in diesem Bereich geöffnet. In der Titelleiste der Fenster erscheint der jeweilige Objektname, bei Bausteinen werden in einer Klammer dahinter zusätzlich je ein Kürzel für den Bausteintyp und die verwendete Programmiersprache angegeben.

Unter dem Menüpunkt 'Fenster' befinden sich alle Befehle zur Fensterverwaltung.

Meldungsfenster

Das Meldungsfenster befindet sich getrennt durch einen Bildschirmteiler unterhalb des Arbeitsbereiches im Hauptfenster. Es enthält alle Meldungen aus dem letzten Übersetzungs-, Überprüfungs- oder Vergleichvorgangs. Auch Suchergebnisse und die Querverweisliste können hier ausgegeben werden.

Wenn Sie im Meldungsfenster mit der Maus einen Doppelklick auf eine Meldung ausführen oder die <Eingabetaste> drücken, öffnet der Editor mit dem Objekt. Die betroffene Zeile des Objekts wird markiert. Mit den Befehlen 'Bearbeiten' 'Nächster Fehler' und 'Bearbeiten' 'Vorheriger Fehler' kann schnell zwischen den Fehlermeldungen gesprungen werden. Die Anzeige des Meldungsfensters ist optional (siehe 'Fenster' 'Meldungen').

Statusleiste

Die Statusleiste, die sich unten im Fensterrahmen des TwinCAT PLC Control-Hauptfensters befindet, zeigt Ihnen Informationen über das aktuelle Projekt und über Menübefehle an. Trifft eine Aussage zu, so erscheint der Begriff rechts in der Statusleiste in schwarzer Schrift, ansonsten in grauer Schrift. Wenn Sie im Online Modus arbeiten, so erscheint der Begriff Online in schwarzer Schrift, arbeiten Sie dagegen im Offline Modus, erscheint es in grauer Schrift. Im Online Modus erkennen Sie in der Statusleiste, ob Sie sich in der Simulation befinden (SIM), das Programm abgearbeitet wird (LÄUFT), ein Breakpoint gesetzt ist (BP) und Variablen geforced werden (FORCE). Bei Texteditoren wird die Zeilen- und Spaltennummer der aktuellen Cursorposition angegeben (z.B. Z.:5, Sp.:11). Wenn Sie im Überschreibmodus arbeiten, wird in der Statusleiste 'ÜB' schwarz angezeigt. Sie können zwischen dem Überschreib- und dem Einfügemodus wechseln, durch Betätigen der Taste <Einfg>. Wenn Sie einen Menübefehl angewählt, aber noch nicht betätigt haben, dann erscheint eine kurze Beschreibung in der Statusleiste. Die Anzeige der Statusleiste ist optional (siehe 'Projekt' 'Optionen' Kategorie Arbeitsbereich).

Kontextmenü Kurzform: <Umschalt>+<F10>

Anstatt die Menüleiste zu verwenden, um einen Befehl auszuführen, können Sie die rechte Maustaste verwenden. Das dann angezeigte Menü enthält die am häufigsten verwendeten Befehle für ein markiertes Objekt oder für den aktiven Editor. Die Auswahl der zur Verfügung gestellten Befehle passt sich automatisch an das aktive Fenster an.

4.2 Optionen

Im TwinCAT PLC Control können Sie die Ansicht Ihres Hauptfensters konfigurieren. Darüber hinaus können Sie noch weitere Einstellungen vornehmen. Dazu steht Ihnen der Befehl 'Projekt' Optionen' zur Verfügung. Die Einstellungen, die Sie hierbei vornehmen, werden, soweit nicht anders vermerkt, in der Datei "TwinCAT PLC Control.ini" gespeichert, und beim nächsten Start von TwinCAT PLC Control wiederhergestellt.

'Projekt''Optionen'

Mit diesem Befehl öffnet sich der Dialog zum Einstellen der Optionen. Die Optionen sind in verschiedene Kategorien eingeteilt. Wählen Sie auf der linken Seite des Dialogs die gewünschte Kategorie durch einen Mausklick oder mit Hilfe der Pfeiltasten aus und verändern Sie auf der rechten Seite die Optionen. Es stehen folgende Kategorien zur Verfügung:

- Laden & Speichern [▶ 46]
- Benutzerinformation [▶ 47]
- Editor [▶ 47]
- Arbeitsbereich [▶ 50]
- Farbe [▶ 51]
- Verzeichnisse [▶ 52]
- Logbuch [▶ 53]
- Übersetzungsoptionen [▶ 53]
- Kennworte [▶ 55]
- Sourcedownload [▶ 56]
- Symbolkonfiguration [▶ 57]
- Projektdatenbank [▶ 59]
- Makros [▶ 63]

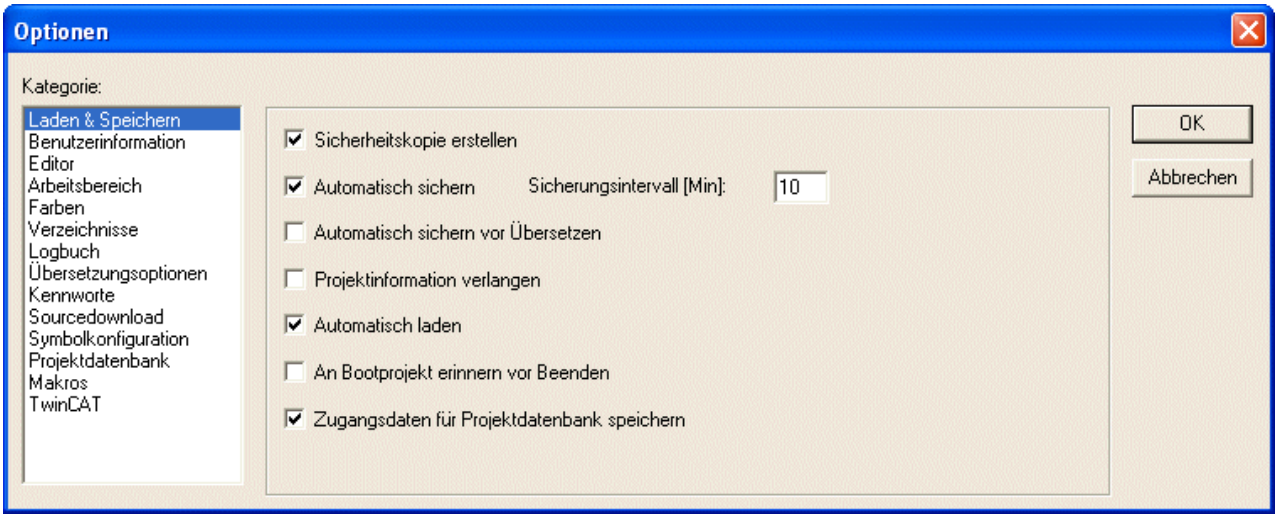
- TwinCAT [▶ 64]

Wenn ein Buscontroller (BCXXXX, BXXXXX) als Laufzeitsystem eingesetzt wird, dann wird die Kategorie "TwinCAT" durch folgende Kategorien ersetzt:

- Controller Settings [▶ 66]
- Controller Advanced

Laden & Speichern

Wenn Sie diese Kategorie wählen, erhalten Sie folgenden Dialog:



Optionsdialog der Kategorie Laden & Speichern

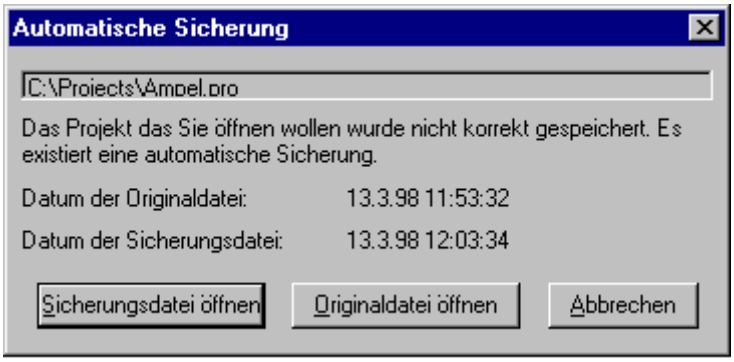
Sicherheitskopie erstellen:

Bei Aktivierung einer Option, erscheint ein Haken vor der Option. Wenn Sie die Option Sicherheitskopie erstellen wählen, dann speichert TwinCAT PLC Control die alte Datei bei jeder Sicherung in eine Sicherungsdatei mit dem Zusatz ".bak". So können Sie also stets die Version vor der letzten Speicherung wieder herstellen.

Automatisch sichern:

Wenn sie die Option 'Automatisch sichern' wählen, dann wird ihr Projekt während Sie daran arbeiten, ständig nach dem von Ihnen eingestellten Zeitintervall (**Sicherungsintervall**) in eine temporäre Datei mit dem Zusatz ".asd" gespeichert. Diese Datei wird beim normalen Beenden des Programms wieder gelöscht. Sollte TwinCAT PLC Control aus irgendeinem Grund nicht "normal" beendet werden (z.B. bei einem Stromausfall), dann wird die Datei nicht gelöscht.

Wenn Sie die Datei wieder öffnen, dann erscheint folgende Meldung:



Sie können nun entscheiden, ob Sie die Originaldatei oder die Sicherungsdatei öffnen wollen.

Automatisch sichern vor Übersetzen:

Das Projekt wird vor jedem Übersetzungslauf gespeichert. Dabei wird eine Datei mit dem Zusatz ".asd" angelegt, die sich verhält wie bei der Option 'Automatisch sichern' beschrieben.

Projektinformation verlangen:

Wenn sie die Option Projektinformation wählen, dann wird beim Abspeichern eines neuen Projekts, oder beim Abspeichern eines Projekts unter einem neuen Namen automatisch die Projektinformation aufgerufen. Die Projektinformationen können Sie mit dem Befehl 'Projekt' 'Projektinformation' einsehen und bearbeiten.

Automatisch laden

Wenn Sie die Option 'Automatisch laden' wählen, dann wird beim nächsten Start von TwinCAT PLC Control das zuletzt geöffnete Projekt automatisch geladen. Das Laden eines Projekts kann beim Start von TwinCAT PLC Control auch durch Angabe eines Projektes in der Befehlszeile erfolgen.

An Bootprojekt erinnern vor Beenden:

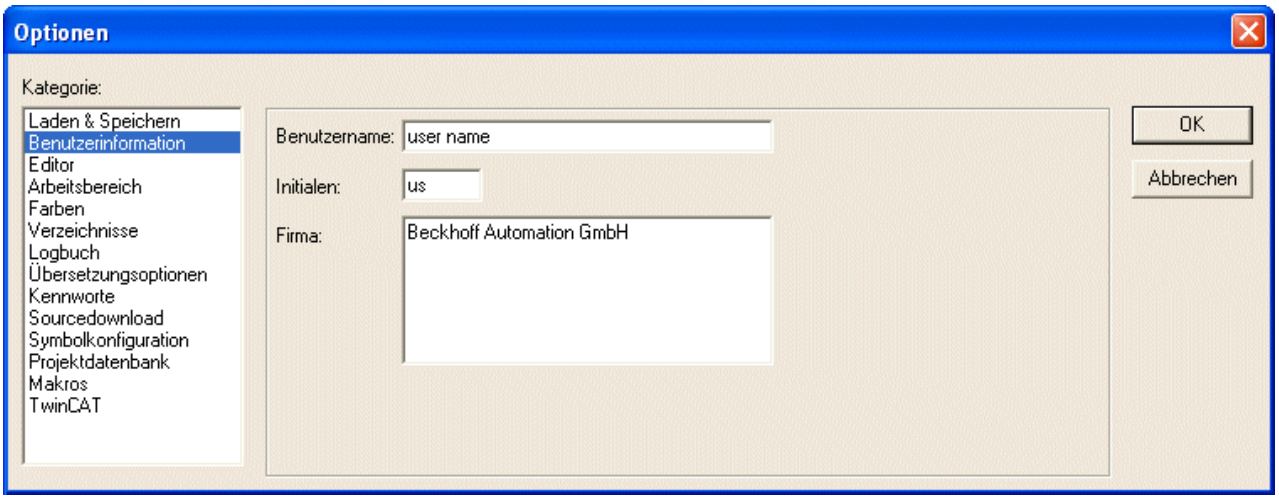
Wenn seit dem Erzeugen eines Bootprojekts das Projekt in modifizierter Form auf die Steuerung geladen wurde, ohne ein neues Bootprojekt zu erzeugen, wird der Anwender beim Verlassen des Projekts darauf aufmerksam gemacht: "Seit dem letzten Download ist kein Bootprojekt erzeugt worden. Trotzdem beenden ?

Zugangsdaten für Projektdatenbank (ENI-Zugangsdaten) speichern:

Benutzername und Passwort, wie sie gegebenenfalls für den Zugang zur ENI-Datenbank eingegeben wurden, werden mit dem Projekt gespeichert.

Benutzerinformation:

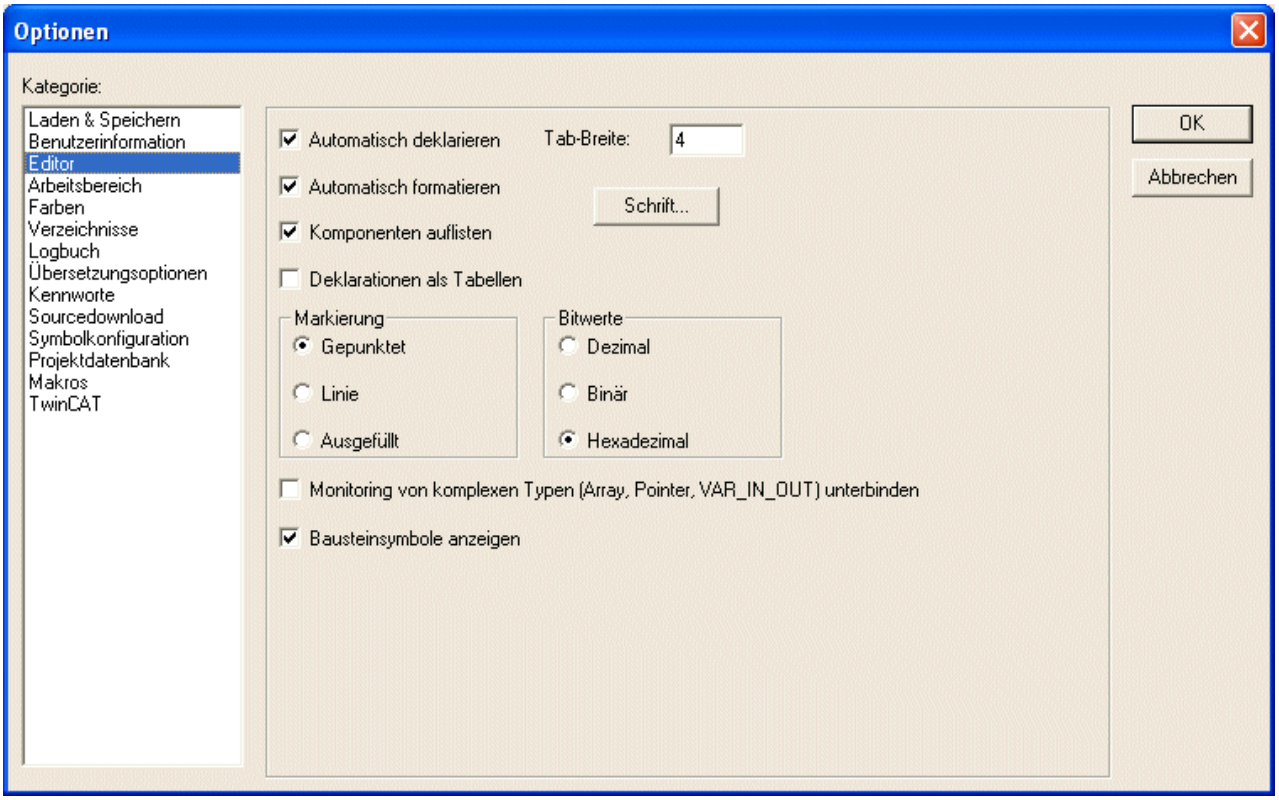
Wenn Sie diese Kategorie wählen, erhalten Sie folgenden Dialog:



Zur Benutzerinformation gehören der Name des Benutzers, seine Initialen und die Firma, bei der er arbeitet. Jeder der Einträge kann verändert werden und wird mit dem Projekt gespeichert.

Editor:

Wenn Sie diese Kategorie wählen, erhalten Sie folgenden Dialog:



Bei Aktivierung einer Option, erscheint ein Haken vor der Option. Sie können folgende Einstellungen zu den Editoren vornehmen:

- Automatisch deklarieren
- Automatisch formatieren
- Deklaration als Tabelle
- Tabulatorbreite
- Schriftart
- Darstellung der Markierung
- Darstellung der Bitwerte

Automatisch deklarieren:

Wenn die Option 'Automatisch deklarieren' gewählt wurde, so erscheint in allen Editoren nach Eingabe einer noch nicht deklarierten Variablen ein Dialog, mit dessen Hilfe diese Variable deklariert werden kann.

Automatisch formatieren:

Wenn die Option Automatisch formatieren gewählt wurde, führt TwinCAT PLC Control eine automatische Formatierung im Anweisungslisteneditor und im Deklarationseditor durch. Wenn eine Zeile verlassen wird, werden folgende Formatierungen durchgeführt:

- Operatoren, die in Kleinbuchstaben geschrieben sind, werden in Großbuchstaben dargestellt.
- Es werden Tabulatoren eingefügt, so dass sich eine einheitliche Spaltenaufteilung ergibt.

Komponenten auflisten:

Wenn diese Option aktiviert ist, steht die Intellisense-Funktion zur Verfügung. Wenn Sie dann an den Stellen, an denen ein Bezeichner eingegeben werden soll, nur einen Punkt eingeben, erhalten Sie eine Auswahlliste der dafür verfügbaren Variablen. Die Intellisense-Funktion gibt es in außer in den Editoren auch im Watch- und Rezepturverwalter und in der Trace-Konfiguration

Deklaration als Tabelle:

Wenn die Option 'Deklarationen als Tabelle' gewählt wurde, können Sie Variablen statt mit dem üblichen Deklarationseditor, als Tabelle editieren. Diese Tabelle ist wie ein Karteikasten geordnet, in dem Registerkarten für Eingabe-, Ausgabe-, lokale und EinAusgabevariablen sind. Für jede Variable stehen Ihnen die Felder Name, Adresse, Typ, Initial und Kommentar zur Verfügung.

Tab-Breite:

Im Feld '**Tab-Breite**' können Sie angeben, wie breit ein Tabulator in den Editoren dargestellt wird. Voreingestellt sind vier Zeichen breit, wobei die Zeichenbreite wiederum von der eingestellten Schriftart abhängt.

Monitoring von komplexen Typen (Array, Pointer, VAR_IN_OUT) unterbinden:

Wenn diese Option aktiviert ist, werden komplexe Datentypen wie Arrays, Pointer, VAR_IN_OUTs im Monitorfenster des Online Modus nicht dargestellt.

Baustein-Symbole anzeigen:

Wenn diese Option aktiviert ist, werden in den Bausteinboxen Symbole dargestellt, sofern diese als Bitmaps im Bibliotheksverzeichnis vorliegen. Der Name der Bitmap-Datei muss sich aus dem Bausteinnamen und der Erweiterung .bmp zusammensetzen. Beispiel: Für den Baustein TON liegt eine Datei TON.bmp vor.

Markierung:

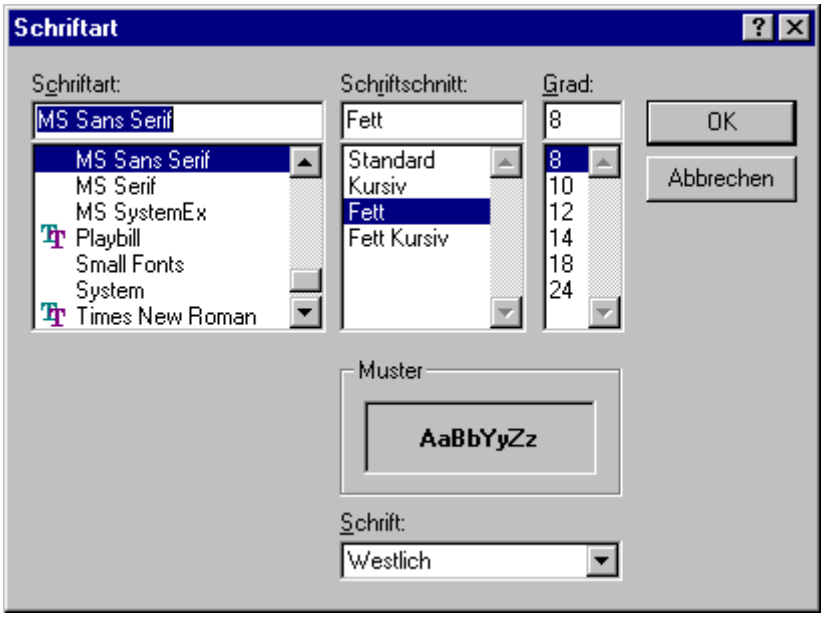
Wählen Sie hier, ob die aktuelle Markierung in den graphischen Editoren durch ein gepunktetes Rechteck (Gepunktet), durch ein Rechteck mit durchgezogener Linie oder durch ein ausgefülltes Rechteck (Ausgefüllt) angezeigt werden soll. Aktiviert ist die Auswahl, vor der ein (•) Punkt erscheint.

Bitwerte:

Wählen Sie, ob binäre Datentypen (BYTE, WORD, DWORD) beim Monitoring Dezimal, Hexadezimal oder Binär dargestellt werden sollen. Aktiviert ist die Auswahl, vor der ein (•) Punkt erscheint.

Schrift:

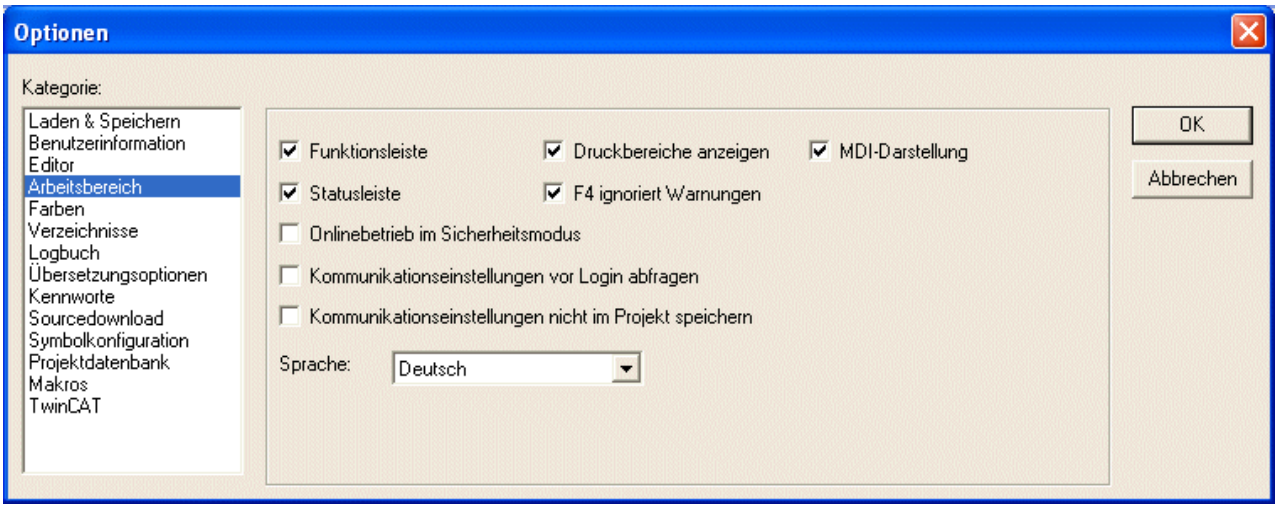
Mit Drücken der Schaltfläche 'Schrift' können Sie die Schrift in allen TwinCAT PLC Control-Editoren wählen. Die Größe der Schrift ist die Grundeinheit für alle Zeichnungsoperationen. Die Wahl einer größeren Schriftgröße vergrößert somit die Ausgabe und sogar den Ausdruck bei jedem Editor von TwinCAT PLC Control. Nachdem Sie den Befehl gewählt haben, öffnet der Dialog für die Wahl der Schriftart, Stil und Schriftgröße.



Dialog zur Einstellung der Schrift

Arbeitsbereich:

Wenn Sie die Kategorie Arbeitsbereich wählen, erhalten Sie folgenden Dialog:



Optionsdialog der Kategorie Arbeitsbereich

Funktionsleiste:

Wenn die Option Funktionsleiste gewählt wurde, wird die Funktionsleiste mit den Schaltflächen zur schnelleren Auswahl von Menübefehlen unterhalb der Menüleiste sichtbar.

Statusleiste:

Wenn die Option Statusleiste gewählt wurde, wird die Statusleiste am unteren Rand des TwinCAT PLC Control-Hauptfensters sichtbar.

Onlinebetrieb im Sicherheitsmodus:

Wenn die Option Onlinebetrieb im Sicherheitsmodus gewählt wurde, erscheint im Online Modus bei den Befehlen 'Start', 'Stop', 'Reset', 'Breakpoint an', 'Einzelzyklus', 'Werte schreiben', 'Werte forcen' und 'Forcen aufheben' ein Dialog mit der Sicherheitsabfrage, ob der Befehl wirklich ausgeführt werden soll.

Kommunikationseinstellungen vor Login abfragen:

Nach dem Befehl 'Online' 'Login' erscheint zunächst der Kommunikationsparameter-Dialog. Erst nachdem dieser mit OK geschlossen wurde, wird in den Online Modus gewechselt.

Kommunikationseinstellungen nicht im Projekt speichern:

Die Einstellungen des Kommunikationsparameter-Dialogs ('Online' 'Kommunikationsparameter') werden nicht mit dem Projekt gespeichert.

Druckbereiche anzeigen:

In jedem Editorfenster werden durch rot gestrichelte Linien die Begrenzungen des aktuell eingestellten Druckbereiches markiert. Dessen Größe hängt von den Druckereigenschaften (Papiergröße, Ausrichtung) und der Größe des "Content"-Bereichs der eingestellten Druckvorlage (Menü 'Datei' 'Einstellungen Dokumentation") ab.

F4 ignoriert Warnungen:

Nach einem Übersetzungslauf springt der Fokus beim Drücken von F4 im Meldungsfenster nur in Zeilen mit Fehlermeldungen, Warnungsausgaben werden ignoriert.

MDI-Darstellung:

Per Default ist diese Option (Multiple-Document-Interface) aktiviert, also das gleichzeitige Öffnen mehrerer Objekte (Fenster) möglich. Wird die Option deaktiviert (SDI-Modus, Single-Document-Interface), kann jeweils nur 1 Fenster im Arbeitsbereich geöffnet werden, das dann im Vollbildmodus dargestellt wird. Ausnahme: Die Aktion eines Programms kann auch im MDI-Modus gleichzeitig mit dem Programm dargestellt werden.

Sprache:

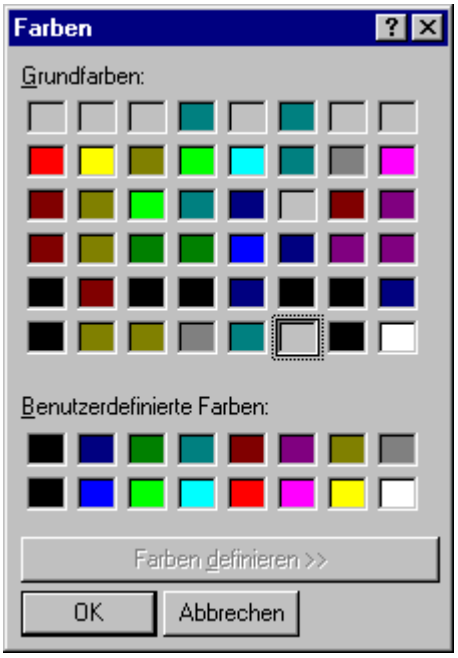
Hier erfolgt die Sprachumschaltung. Wählen Sie, in welcher Landessprache die Menü- und Dialogtexte sowie die Online Hilfe erscheinen sollen.

Farben:

Wenn Sie die Kategorie Farben wählen, erhalten Sie folgenden Dialog:

Optionsdialog der Kategorie Farbe

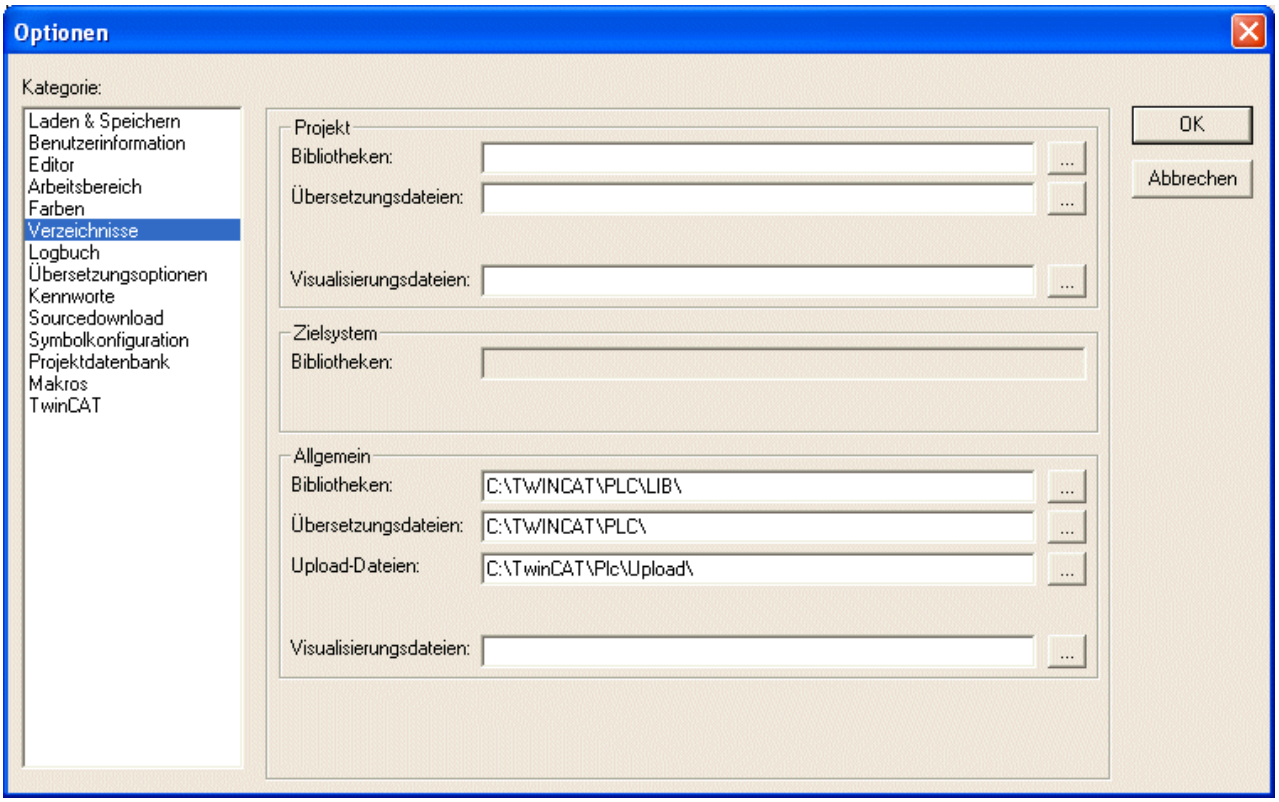
Sie können die voreingestellten Farbeinstellungen von TwinCAT PLC Control editieren. Sie können wählen, ob Sie die Farbeinstellungen für Zeilennummern (Voreinstellung:hellgrau), für Breakpoint Positionen (dunkelgrau), für einen gesetzten Breakpoint (hellblau), für die aktuelle Position (rot), für die durchlaufenen Positionen (grün) oder für das Monitoring boolescher Werte (blau) ändern wollen. Wenn Sie eine der angegebenen Schaltflächen gewählt haben, dann öffnet der Dialog zum Eingeben von Farben.



Dialog zur Einstellung der Farbe

Verzeichnisse:

Wenn Sie die Kategorie Verzeichnisse wählen, erhalten Sie folgenden Dialog:



In den Bereichen **Projekt** und **Allgemein** können Verzeichnisse eingetragen werden, die das TwinCAT PLC Control nach Bibliotheken durchsuchen bzw. für die Ablage von Übersetzungs-, Source-Upload- und Visualisierungs-Dateien verwenden soll. Wenn Sie die Schaltfläche (...) hinter einem Feld betätigen, dann öffnet der Dialog zum Auswählen eines Verzeichnisses. Für Bibliotheks- und Konfigurationsdateien können jeweils mehrere Pfade getrennt durch ein Semikolon ";" eingegeben werden.

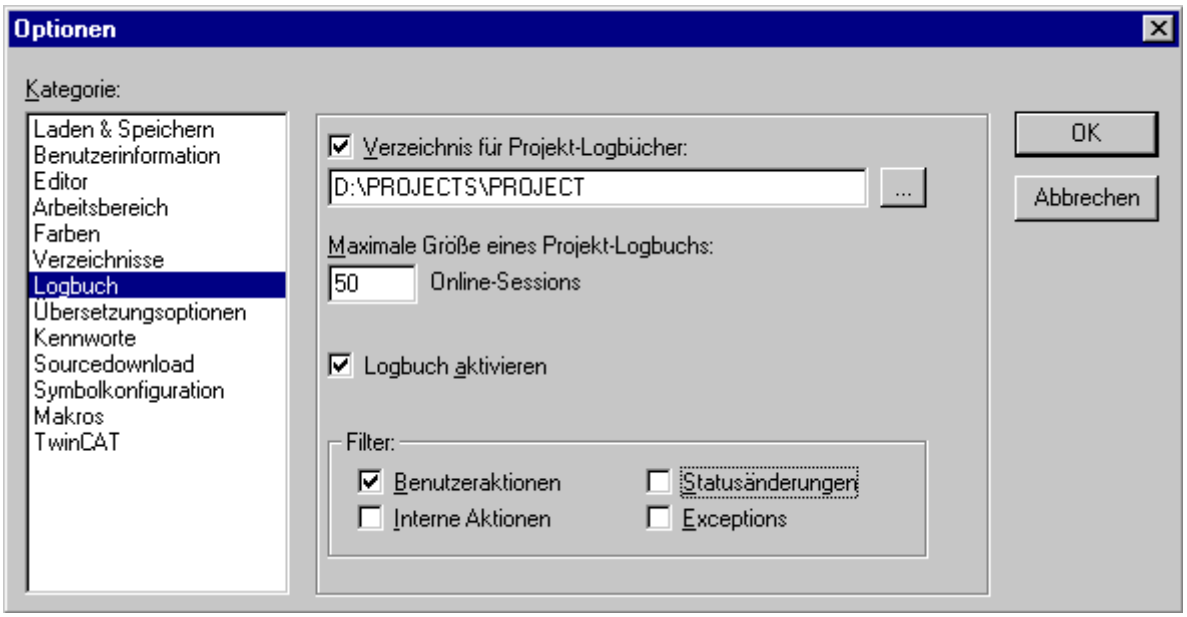
Bibliothekspfade können relativ zum aktuellen Projektverzeichnis eingegeben werden, indem ein "." vorangestellt wird. Wird z.B. ".\libs" angegeben, werden die Bibliotheken auch im Verzeichnis 'C:\Programme\projects\libs' gesucht, wenn das aktuelle Projekt im Verzeichnis 'C:\Programme\projects' liegt.

Die Angaben im Bereich **Projekt** werden mit dem Projekt gespeichert. Die Angaben im Bereich **Allgemein** werden in die ini-Datei des Programmiersystems geschrieben und gelten somit für alle Projekte.

Das TwinCAT PLC Control sucht generell zuerst in den bei 'Projekt' eingetragenen Verzeichnissen, dann in den unter 'Zielsystem' und zuletzt in den unter 'Allgemein' angegebenen. Liegen gleichnamige Dateien vor, wird die verwendet, die im zuerst durchsuchten Verzeichnis steht.

Logbuch:

Wenn Sie diese Kategorie wählen, wird folgender Dialog geöffnet:



In diesem Dialog können Sie eine Datei konfigurieren, die als Projekt-Logbuch alle Benutzeraktionen und internen Vorgänge während des Online-Modus chronologisch aufzeichnet.
Wird ein bestehendes Projekt geöffnet, für das noch kein Logbuch generiert wurde, öffnet ein Dialog, der darauf aufmerksam macht, dass nun ein Logbuch angelegt wird, das erstmals mit dem nächsten Login-Vorgang Einträge erhält.

Das Logbuch wird beim Speichern des Projekts automatisch als Binärdatei im Projektverzeichnis gespeichert.

Wenn Sie ein anderes Zielverzeichnis wünschen, können Sie die Option Verzeichnis für Projekt-Logbücher: aktivieren und im Editierfeld Speicherung den entsprechenden Pfad eingeben Über die Schaltfläche erhalten Sie dazu den Dialog '**Verzeichnis auswählen**'.

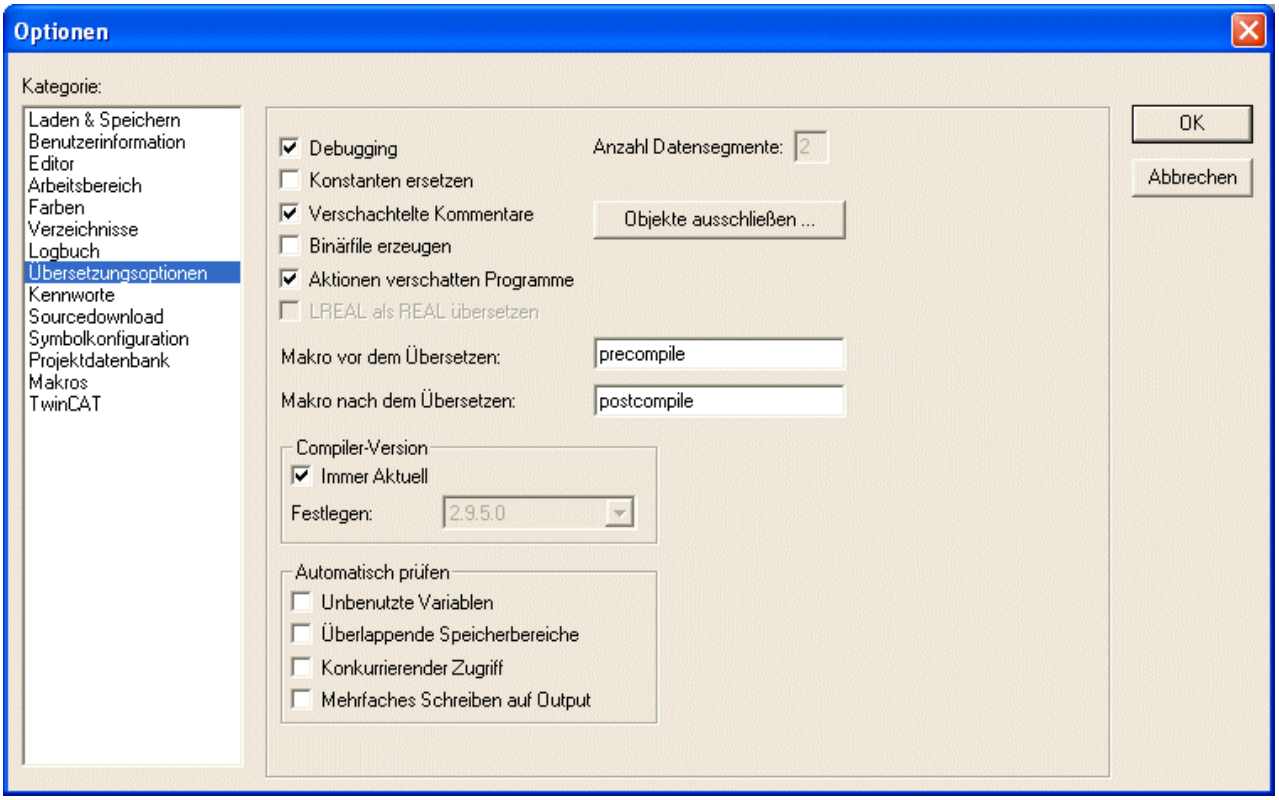
Die Logbuch-Datei erhält automatisch den Namen des Projekts mit der Erweiterung .log. Unter **Maximale Größe** eines Projekt-Logbuchs wird die Höchstanzahl der aufzuzeichnenden **Online-Sessions** festgelegt. Wird während der Aufzeichnung diese Anzahl überschritten, wird der jeweils älteste Eintrag zugunsten des neuesten gelöscht.
Die Funktion Logbuch kann im Optionsfeld Logbuch aktivieren ein- oder ausgeschaltet werden.

Im Bereich **Filter** können Sie wählen, welche Aktionen aufgezeichnet werden sollen: Benutzeraktion, Interne Aktion, Statusänderung, Exception. Nur Aktionen der hier mit einem Haken versehenen Kategorien werden im Logbuch-Fenster erscheinen bzw. in die Logbuch-Datei geschrieben.

Das Logbuch-Fenster können Sie mit dem Befehl 'Fenster' 'Logbuch' öffnen.

Übersetzungsoptionen:

Wenn Sie die Kategorie Übersetzungsoptionen wählen, erhalten Sie folgenden Dialog:



Optionsdialog der Kategorie Übersetzungsoptionen

Die An-/Abwahl der Optionen **Debugging**, Online Changes hat in TwinCAT für den PC momentan noch keine Auswirkungen. Diese Einstellungen müssen unter den TwinCAT Optionen durchgeführt werden ! Bei TwinCAT für Buscontroller hat die Einstellung zur Folge, dass keine Breakpoints mehr möglich sind. Dafür steht mehr Programmspeicher zur Verfügung.

Wenn die Option **Konstanten ersetzen** gewählt wurde, wird für jede Konstante direkt deren Wert geladen und im Online Modus werden die Konstanten in grün angezeigt. Forcen, Schreiben und Monitoring einer Konstante ist dann nicht mehr möglich. Ist die Option deaktiviert, wird der Wert über Variablenzugriff auf einen Speicherplatz geladen (dies ermöglicht zwar das Schreiben des Variablenwertes, bedeutet aber längere Bearbeitungszeit).

Wenn die Option **Verschachtelte Kommentare** aktiviert ist, können Kommentare ineinander verschachtelt eingegeben werden.

Beispiel:

```
(*
a:=inst.out; (* to be checked *)
b:=b+1;
*)
```

Der Kommentar, der mit der ersten Klammer beginnt, wird hier nicht bereits durch die Klammer nach 'checked' abgeschlossen, sondern erst durch die letzte Klammer.

Binärfile erzeugen:

Wenn die Option Binärfile erzeugen aktiviert ist, wird beim Übersetzen ein binäres Abbild des erzeugten Codes (Bootprojekt) im Projektverzeichnis angelegt. Der Dateiname: <projektname>.bin. Beachten Sie hierzu auch die Möglichkeit, mit dem Befehl 'Online' 'Bootprojekt erzeugen' das Bootprojekt und die Datei mit der zugehörigen Checksumme online auf der Steuerung bzw. offline im Projektverzeichnis abzulegen.

Aktionen verschatten Programme:

Diese Option ist per Default aktiviert, wenn ein neues Projekt angelegt wird. Sie bedeutet: Wenn eine lokale Aktion den gleichen Namen trägt wie eine Variable oder ein Programm, gilt folgende Rangfolge bei der Abarbeitung: lokale Variable vor lokaler Aktion vor globaler Variable vor Programm.

HINWEIS

Wird ein Projekt geöffnet, das mit einer früheren TwinCAT PLC Control Version erstellt wurde, ist die Option per Default deaktiviert. Somit wird die beim Erstellen gültige bisherige Rangfolge (lokale Variable vor globaler Variable vor Programm vor lokaler Aktion) beibehalten.

LREAL als REAL übersetzen:

Wenn diese Option aktiviert wird (Default: nicht aktiviert), werden beim Kompilieren des Projekts LREAL-Werte wie REAL-Werte behandelt. Dies kann verwendet werden, um Projekte plattformunabhängig zu entwickeln.

Datensegmente:

Mit der Angabe der Anzahl der Datensegmente können Sie festlegen, wie viele Speichersegmente in der Steuerung für die Daten Ihres Projekts reserviert werden sollen. Dieser Platz ist nötig, damit ein Online Change auch durchgeführt werden kann, wenn neue Variablen hinzugefügt wurden. Wenn beim Übersetzen die Meldung kommt "Die globalen Variablen brauchen zu viel Speicher, erhöhen Sie die hier eingetragene Anzahl. Diese Einstellung hat nur den Bus Controller BCxxxx Bedeutung. Bei TwinCAT für den PC ist die Anzahl der Datensegmente konstant. Die Größe kann aber unter den TwinCAT Optionen eingestellt werden.

Objekte ausschließen:

Diese Schaltfläche führt zum Dialog Objekte vom Übersetzen ausschließen:
Wählen Sie hier im dargestellten Baum die Projektbausteine aus, die bei einem Kompilierungslauf nicht übersetzt werden sollen, und aktivieren Sie die Option 'Nicht übersetzen'. Daraufhin werden die ausgeschlossenen Bausteine im Baum grün angezeigt. Um automatisch alle Bausteine auszuschließen, die im Programm nicht verwendet werden, drücken Sie die Schaltfläche 'Unbenutzte ausschließen'.

Makro:

Um auf den Übersetzungsvorgang Einfluss zu nehmen, können zwei Makros angegeben werden: Das Makro im Feld Makro vor dem Übersetzen wird vor dem Übersetzungslauf ausgeführt, das Makro im Feld Makro nach dem Übersetzen danach. Folgende Makro-Befehle können hier allerdings nicht ausgeführt werden: file new, file open, file close, file saveas, file quit, online, project compile, project check, project build, debug, watchlist
Alle im Dialog Übersetzungsoptionen festgelegten Einstellungen werden mit dem Projekt gespeichert.

Compiler Version:

Die für den Übersetzungsvorgang zu verwendende Compiler-Version kann hier definiert werden. In TwinCAT PLC Control Versionen nach V2.9 stehen jeweils sowohl die aktuelle als auch die bisherigen Compiler-Versionen (für jede Version / jedes Service Pack / jedes Patch) zurückgehend bis V2.9 zur Verfügung. Wenn Sie wollen, dass ein Projekt stets mit der neuesten Compiler-Version übersetzt werden soll, aktivieren Sie die Option 'Immer Aktuell'. Wenn das Projekt automatisch mit einer bestimmten Version übersetzt werden soll, stellen Sie diese über das Auswahlfeld bei 'Festlegen' ein.

Automatisch prüfen:

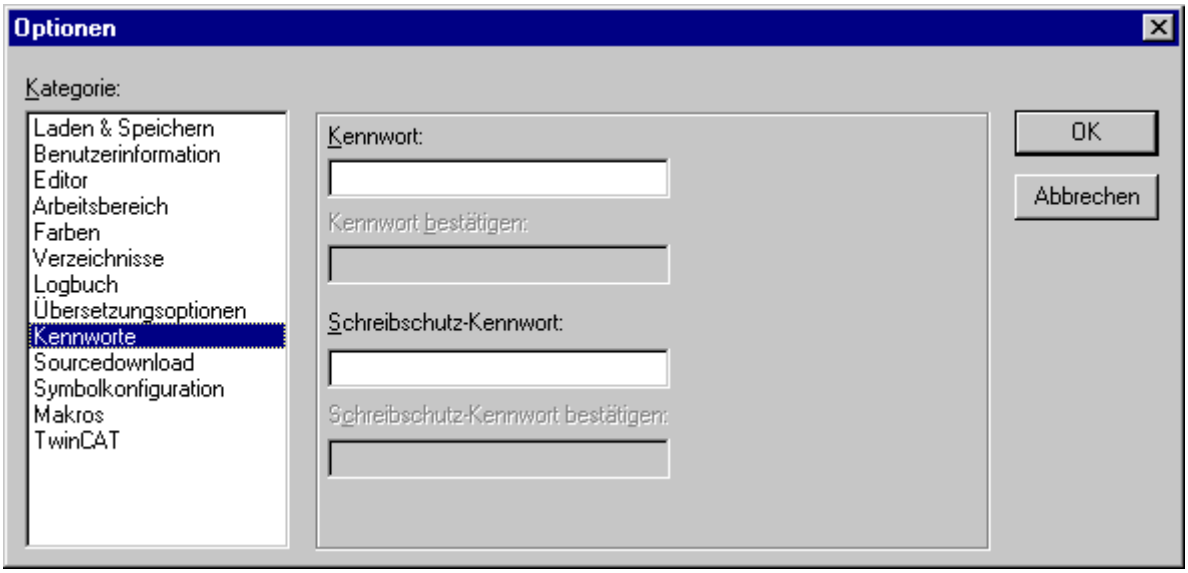
Zur Überprüfung der semantischen Korrektheit bei jedem Übersetzungslauf des Projekts können die folgenden Optionen aktiviert werden:
• Unbenutzte Variablen
• Überlappende Speicherbereiche
• Konkurrierender Zugriff
• Mehrfaches Speichern auf Output

Die Ergebnisse werden im Meldungsfenster ausgegeben. Diese Prüfungen können über das Befehlsmenü 'Überprüfen' im Menü 'Projekt' auch gezielt angestoßen werden

Alle im Dialog Übersetzungsoptionen festgelegten Einstellungen werden mit dem Projekt gespeichert.

Kennworte:

Wenn Sie die Kategorie Kennworte wählen, erhalten Sie folgenden Dialog:



Optionsdialog der Kategorie Kennworte

Zur Sicherung Ihrer Dateien vor unerwünschten Zugriffen bietet TwinCAT PLC Control die Möglichkeit, das Öffnen oder Verändern Ihrer Datei durch ein Kennwort absichern zu lassen.

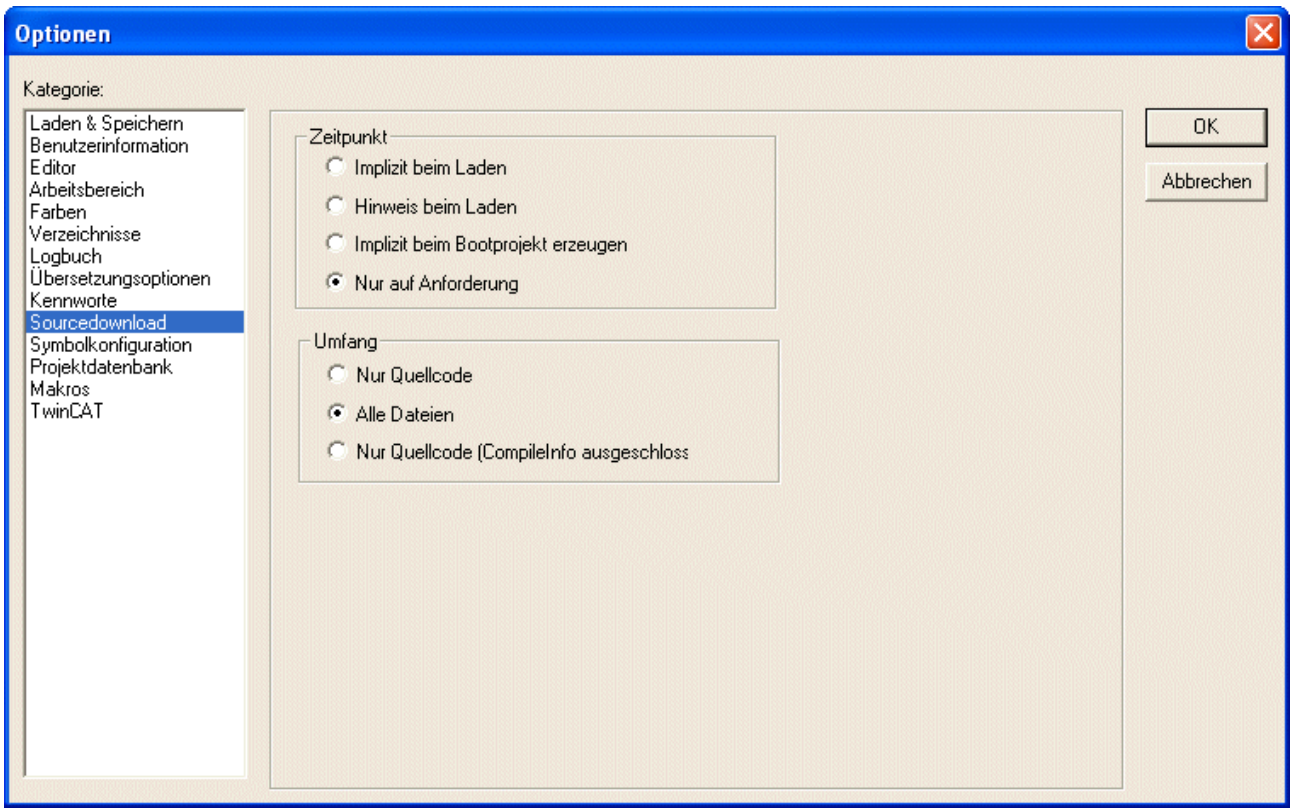
Geben Sie das gewünschte Kennwort im Feld **Kennwort** ein. Für jeden getippten Buchstaben erscheint im Feld ein Stern (*). Dasselbe Wort müssen Sie im Feld Kennwort bestätigen wiederholen. Schließen Sie den Dialog mit OK. Wenn die Meldung kommt: "Das Kennwort und seine Bestätigung stimmen nicht überein.", haben Sie sich bei einem der beiden Einträge vertippt. Wiederholen Sie deswegen am besten beide Einträge solange, bis der Dialog ohne Meldung schließt. Wenn Sie nun die Datei abspeichern und erneut öffnen erscheint ein Dialog, in dem Sie aufgefordert werden, das Kennwort einzugeben. Das Projekt wird nur dann geöffnet, wenn Sie das richtige Kennwort eingetippt haben, ansonsten meldet Ihnen TwinCAT PLC Control: "Das Kennwort ist nicht korrekt."

Neben dem Öffnen der Datei können Sie auch das Verändern der Datei mit einem Kennwort schützen. Dazu müssen Sie einen Eintrag in das Feld **Schreibschutz-Kennwort** vornehmen, und diesen Eintrag wieder im Feld darunter bestätigen. Ein schreibgeschütztes Projekt kann auch ohne Passwort geöffnet werden. Drücken Sie hierzu einfach die Schaltfläche Abbrechen, wenn TwinCAT PLC Control Sie beim Öffnen einer Datei auffordert, das Schreibschutz-Kennwort einzugeben. Nun können Sie das Projekt übersetzen, in die Steuerung laden, simulieren etc., aber Sie können es nicht verändern.

Wichtig ist natürlich, dass Sie sich selber beide Kennwörter gut merken. Die Kennworte werden mit dem Projekt gespeichert. Um differenziertere Zugriffsrechte zu schaffen, können Sie Arbeitsgruppen festzulegen ('Projekt' 'Objekt Zugriffsrechte' und 'Passwörter für Arbeitsgruppen').

Sourcedownload:

Wenn Sie diese Kategorie wählen, wird folgender Dialog geöffnet:



Sie können wählen, zu welchem **Zeitpunkt** und in welchem **Umfang** der Quellcode des Projekts in die Steuerung gespeichert wird. Die Daten werden hierzu gepackt.

Umfang:

- Die Option **Nur Quellcode** betrifft ausschließlich die Projekt-Datei (Zusatz .pro).
- Die Option **Alle Dateien** umfasst zusätzlich Dateien wie z.B. die zugehörigen Bibliotheken, Konfigurationsdateien usw.
- Die Option **Nur Quellcode (CompileInfo ausgeschlossen)** lädt wie oben, den reinen Quellcode, jedoch ohne CompileInfo. Somit ist diese Funktion eine reine Quellablage, ein Online Change nach einem Sourceupload von der SPS ist nicht möglich.
  Wegen des geringeren Datenumfangs ist diese Option für die BX Plattform interessant.

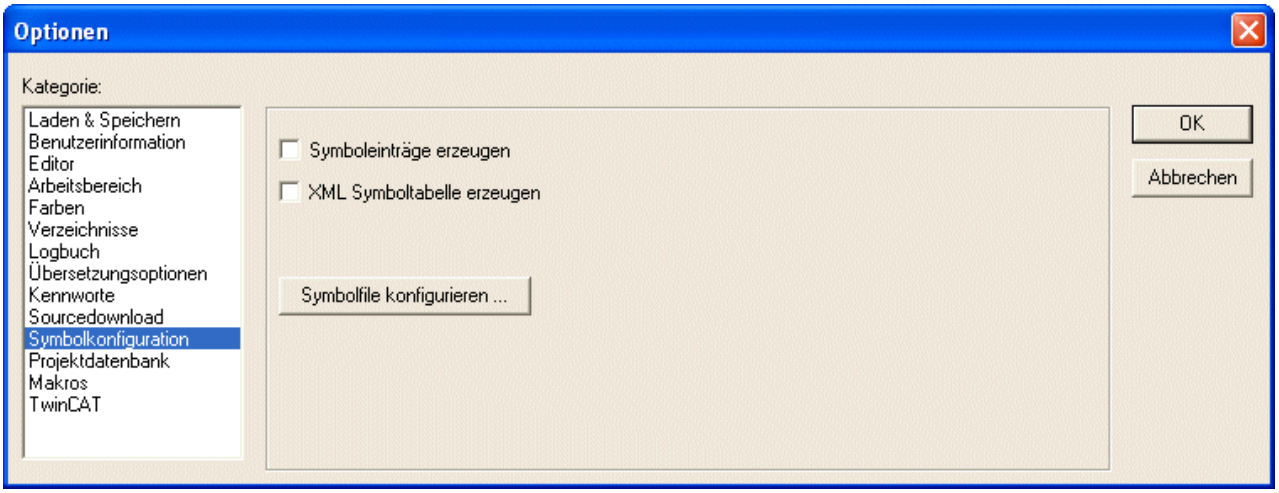
Zeitpunkt:

- Mit der Option **Implizit beim Laden** wird beim Befehl 'Online' 'Laden' der gewählte Dateiumfang automatisch in die Steuerung geladen.
- Mit der Option **Hinweis beim Laden** erhalten Sie beim Befehl 'Online' 'Laden' einen Dialog mit der Frage "Quellcode in die Steuerung schreiben ?" Drücken Sie Ja, wird der gewählte Dateiumfang automatisch in die Steuerung geladen, andernfalls schließen sie mit Nein.
- Mit der Option **Implizit beim Bootprojekt erzeugen** wird beim Befehl 'Online' 'Bootprojekt erzeugen' der gewählte Datenumfang automatisch in die Steuerung geladen.
- Mit der Option **Nur auf Anforderung** muss der gewählte Dateiumfang ausdrücklich über den Befehl 'Online' 'Quellcode laden' in die Steuerung geladen werden.

Das in der Steuerung gespeicherte Projekt können Sie unter 'Datei' 'Öffnen' mit 'Projekt aus der Steuerung öffnen' wieder hochladen. Die Daten werden dabei wieder entpackt.

Symbolkonfiguration

Diese Form der Symbolhaltung ist nur aus Kompatibilitätsgründen vorhanden und sollte auch nur für den TwinCAT OPC Server verwendet werden. Für alle anderen Zwecke existiert eine bei jedem Compile neu erstellte XML Datei mit dem Namen <projectname>.tpy

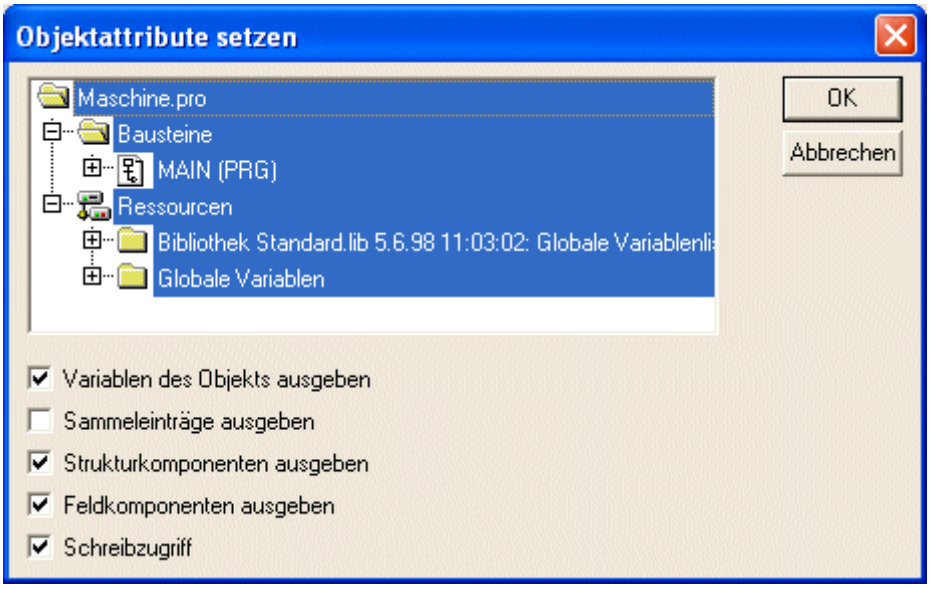


Der hier gebotene Dialog dient der Konfiguration der Symboldatei. Diese wird als Textdatei <Projektname>.sym bzw. Binärdatei <Projektname>.sdb im Projektverzeichnis angelegt.

Wenn die Option **Symboleinträge erzeugen** angewählt ist, werden automatisch bei jedem Übersetzen des Projekts Symboleinträge für die Projektvariablen in der Symboldatei angelegt.

Wenn zusätzlich die Option **XML-Datei erzeugen** aktiviert ist, wird außerdem eine XML-Version der Symboldatei erzeugt. Diese wird ebenfalls im Projektverzeichnis angelegt und erhält den Namen <Projektname>.SYM_XML.

Wenn Sie 'Symbolfile konfigurieren' betätigen, öffnet sich folgendes Fenster:



Wählen Sie im als Baumstruktur dargestellten Auswahleditor Projektbausteine aus und setzen Sie im unteren Dialogteil die gewünschten Optionen durch Mausklick auf das zugehörige Kästchen. Aktivierte Optionen sind mit einem Haken versehen.

Variablen des Objekts ausgeben: Die Variablen des gewählten Objektes werden ins Symbolfile ausgegeben.

Nur wenn die Option **Variablen des Objekts ausgeben** aktiviert ist, können folgende weitere Optionen wirksam werden:

Sammeleinträge ausgeben: Für Strukturen und Arrays des Objektes werden Einträge zum Zugriff auf die Gesamtvariablen erzeugt.

Strukturkomponenten ausgeben: Für Strukturen des Objektes wird für jede Variablenkomponente ein eigener Eintrag erzeugt.

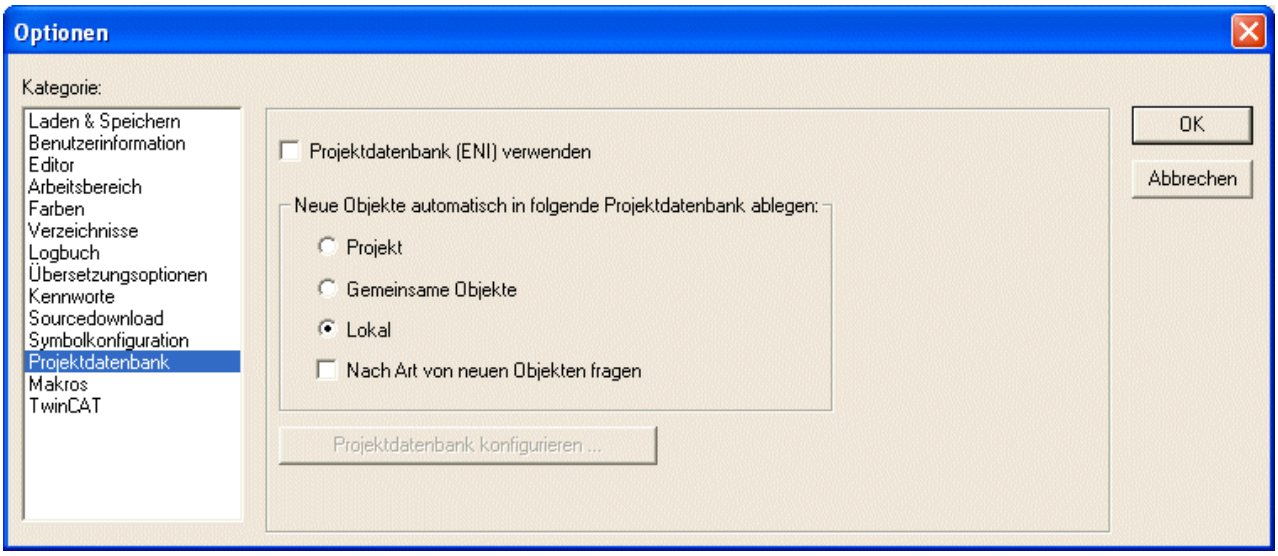
Feldkomponenten ausgeben: Für Arrays des Objektes wird für jede Variablenkomponente ein eigener Eintrag erzeugt.

Schreibzugriff: Die Variablen des Objektes dürfen vom OPC-Server verändert werden.

Nachdem die Optioneneinstellungen für die aktuelle Bausteinauswahl vorgenommen wurde, können andere Bausteine ausgewählt werden und ebenfalls mit einer Optionenkonfiguration versehen werden. Dies kann für beliebig viele Bausteinselektionen hintereinander ausgeführt werden. Wird der Dialog mit **OK** geschlossen, werden alle seit Öffnen des Dialogs vorgenommenen Konfigurationen übernommen.

Projektdatenbank

Wenn Sie die Kategorie Projektdatenbank wählen erscheint folgender Dialog:



In diesem Dialog wird festgelegt, ob das Projekt in einer Projektdatenbank verwaltet werden soll und es werden die für diesen Fall nötigen Konfigurationen der ENI-Schnittstelle vorgenommen.

Projektdatenbank (ENI) verwenden:

Aktivieren Sie diese Option, wenn Sie über einen ENI-Server auf eine Projektdatenbank zugreifen wollen, um alle oder bestimmte zum Projekt gehörigen Bausteine über diese Datenbank zu handhaben. Voraussetzung hierfür ist, dass ENI-Server und Projektdatenbank installiert sind und Sie als gültiger Datenbank-Benutzer definiert sind.

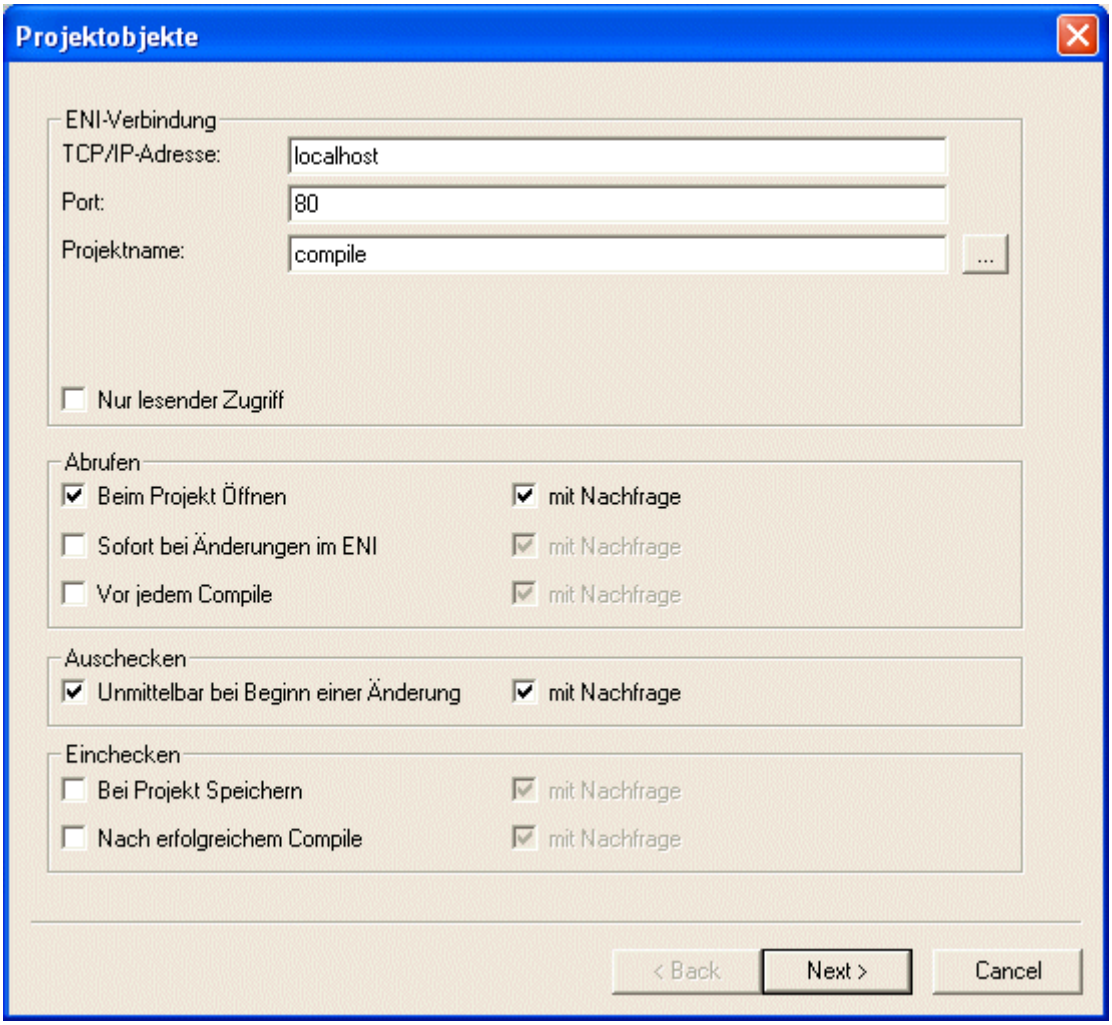
Wenn die Option aktiviert ist, stehen für jedes Objekt des Projekts die Funktionen (Einchecken, Abrufen etc.) der Projektdatenbank zur Verfügung. Zum einen werden dann gewisse Datenbankfunktionen automatisch ablaufen, wenn dies in den Optionen-Dialogen so konfiguriert ist, zum anderen können aber auch die Befehle des Menüs 'Projekt' 'Datenbankverknüpfung' zum gezielten Aufruf der Funktionen verwendet werden. Außerdem ist dann im Dialog für die Objekteigenschaften ein Registerblatt 'Datenbank-Verknüpfung' verfügbar, über den der Baustein einer bestimmten Datenbankkategorie zugeordnet werden kann.

Neue Objekte automatisch in folgende Projektdatenbank ablegen: Hier nehmen Sie eine Standardeinstellung vor: Wenn ein Objekt im Projekt neu eingefügt wird ('Objekt einfügen'), wird es automatisch der hier eingestellten Objektkategorie zugeordnet. Diese Zuordnung wird in den Objekteigenschaften ('Projekt' 'Objekt' 'Eigenschaften') wiedergegeben und kann dort auch für das Objekt verändert werden. Die möglichen Zuordnungen sind:
Projekt: Der Baustein wird in dem Datenbankverzeichnis abgelegt, das im Dialog ENI-Einstellungen/ Projektobjekte unter 'Projektname' definiert ist.
Gemeinsame Objekte: Der Baustein wird in dem Datenbankverzeichnis verwaltet, das im Dialog ENI-Einstellungen/Gemeinsame Objekte unter 'Projektname' definiert ist.
Lokal: Der Baustein wird nicht über das ENI in der Projektdatenbank verwaltet, sondern ausschließlich lokal im Projekt gespeichert.
Neben 'Projektobjekte' und 'Gemeinsame Objekte' gibt es eine dritte Datenbankkategorie 'Übersetzungsdateien' für solche Objekte, die erst beim Kompilieren eines Projekts entstehen, weswegen die Kategorie hier nicht relevant ist.

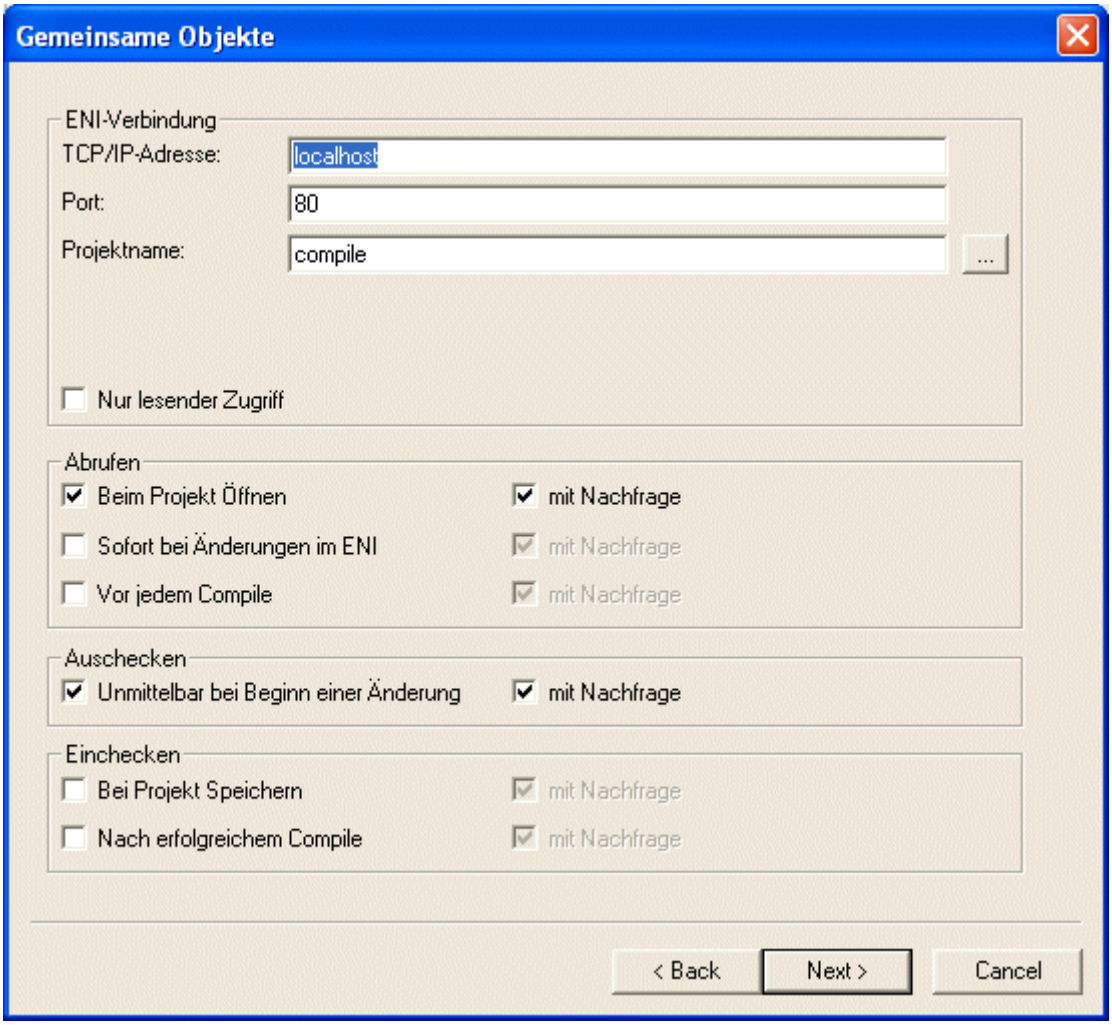
Nach Art von neuen Objekten fragen: Wenn diese Option aktiviert ist, wird bei jedem Einfügen eines neuen Objekts im Projekt der Dialog 'Objekt' 'Eigenschaften' geöffnet, in dem ausgewählt werden kann, welcher der oben genannten drei Objektkategorien der Baustein angehören soll. Damit kann also die Standardeinstellung überschrieben werden.

Projektdatenbank konfigurieren:

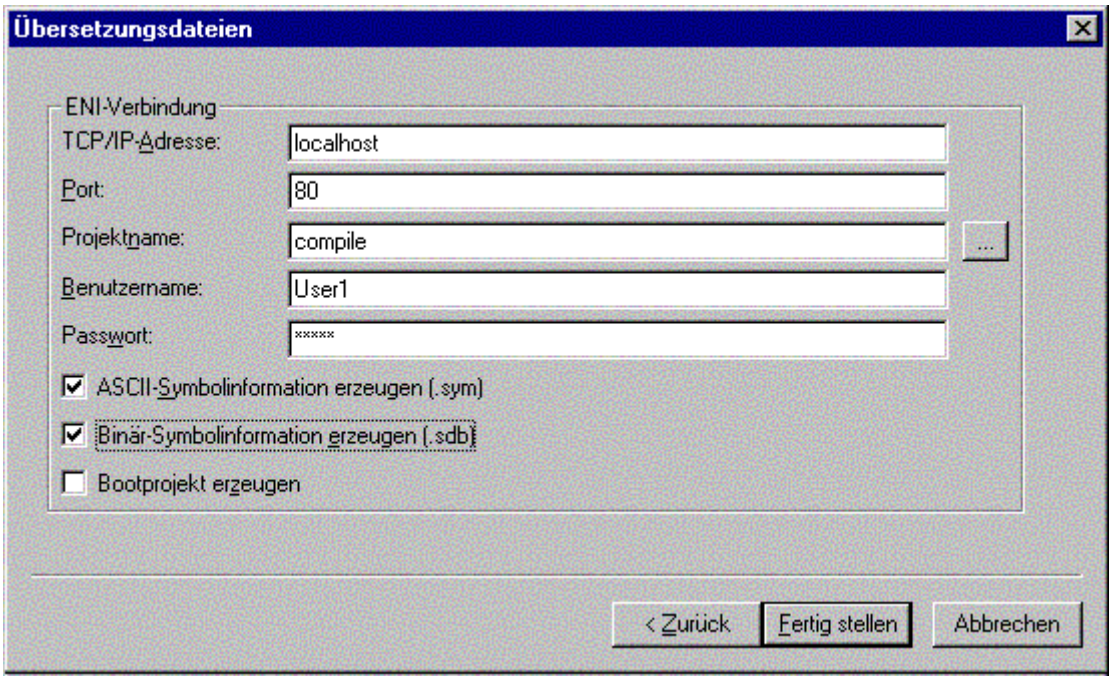
Diese Schaltfläche führt zu den ENI-Einstellungen, die in drei Dialogen vorgenommen werden.
Die zum Projekt gehörenden Objekte, die in der Datenbank verwaltet werden sollen, können den Datenbankkategorien 'Projektobjekte', 'Gemeinsame Objekte' oder 'Übersetzungsdateien' zugeordnet sein. Für jede dieser Kategorien wird in den folgenden Dialogen der Projektdatenbankoptionen festgelegt, in welchem Verzeichnis sie in der Datenbank liegen und welche Voreinstellungen für gewisse Datenbankfunktionen gelten:
Dialog ENI-Konfiguration / Projektobjekte



Dialog ENI-Konfiguration / Gemeinsame Objekte



Dialog ENI-Konfiguration / Übersetzungsdateien



Die Objekte werden in jedem Fall auch zusätzlich lokal, also mit dem Projekt gespeichert.

Die Dialoge erscheinen bei einer Erstkonfiguration nacheinander, wobei ein **Wizard** (über die Schaltfläche Weiter bzw. Next) den Benutzer führt. Dabei werden die im ersten Dialog vorgenommenen Einstellungen in die beiden anderen übernommen und es müssen nur gewünschte Abweichungen eingegeben werden.

Liegt bereits eine Konfiguration vor, sind die Dialoge in Form von drei Registerblättern in einem Fenster zusammengefasst.
Wurde vor der Konfiguration nicht bereits erfolgreich in die Datenbank eingeloggt (Login-Dialog über Menü 'Projekt' 'Datenbankverknüpfung' 'Login') wird der Login-Dialog zu diesem Zweck automatisch geöffnet werden.

Optionen für Projektobjekte, Gemeinsame Objekte und Übersetzungsdateien bezüglich Projektdatenbank

Diese Dialoge sind Bestandteil der Optionseinstellungen für die Projektdatenbank ('Projekt' 'Optionen' 'Projektdatenbank'). Hier wird definiert, mit welchen Zugangsparametern die Objekte der Kategorien 'Projekt' und 'Gemeinsame Objekte' in der Datenbank verwaltet werden. Im Dialog Übersetzungsdateien wird festgelegt, wie die Objekte der Kategorie Übersetzungsdateien in der Datenbank verwaltet werden.

Die Eingabefelder für ENI Verbindung sind bei allen drei Dialogen identisch:

ENI-Verbindung:

TCP/IP-Adresse: Adresse des Rechners, auf dem der ENI-Server läuft

Port: Default: 80; muss mit der Einstellung in der ENI-Server Konfiguration übereinstimmen

Projektname: Name des Verzeichnisses in der Datenbank, in dem die Objekte der betreffenden Kategorie abgelegt werden sollen. Falls das Verzeichnis in der Datenbank bereits existiert, können Sie es im Verzeichnisbaum der ENI Projekte auswählen, den Sie über die Schaltfläche ... erhalten. Wenn Sie sich allerdings vorher noch nicht über den Login-Dialog als ENI-Benutzer identifiziert haben, erscheint beim Drücken dieser Schaltfläche allerdings zunächst eben dieser Login-Dialog, wo Sie Benutzernamen und Passwort für den ENI-Zugang zu den drei Datenbank-Kategorien eingeben müssen.

Benutzername: Hier ist der Name des Benutzers einzugeben. (Nur bei Dialog Übersetzungsdateien)

Password: Passwort des Benutzers. (Nur bei Dialog Übersetzungsdateien)

Nur lesender Zugriff: Wenn diese Option aktiviert ist, kann auf die Daten des hier definierten Datenbankverzeichnisses nur lesend zugegriffen werden. (Nicht bei Dialog Übersetzungsdateien)

Optionen für Projektobjekte, Gemeinsame Objekte:

Abrufen:

Die Datenbankfunktion Abrufen (Menü 'Projekt' 'Datenbankverknüpfung') bedeutet, dass die aktuelle Version eines Bausteins aus der Datenbank ins lokal geöffnete Projekt kopiert wird, wobei die lokale Version überschrieben wird. Automatisch erfolgt dies für alle gegenüber der lokalen Projektversion veränderten Bausteine zu jedem der folgenden Zeitpunkte, der aktiviert (mit einem Haken markiert) ist:
Beim Projekt Öffnen: Wenn das Projekt in TwinCAT PLC Control geöffnet wird.
Sofort bei Änderungen im ENI: Wenn in der Datenbank eine neuere Version eines Bausteins eingecheckt wird; der Baustein wird dann unmittelbar im geöffneten Projekt aktualisiert und es wird eine entsprechende Meldung ausgegeben.
Vor jedem Compile: Vor jedem Übersetzungsvorgang in TwinCAT PLC Control.

Auschecken:

Die Datenbankfunktion Auschecken bedeutet, dass der Baustein als 'in Bearbeitung' markiert wird und für andere Benutzer gesperrt ist, bis er durch Einchecken oder Rückgängigmachen des Auscheckens wieder freigegeben wird.
Wenn die Option **Unmittelbar bei Beginn einer Änderung** aktiviert ist, dann erfolgt das Auschecken eines Bausteins automatisch, sobald mit dessen Bearbeitung im Projekt begonnen wird. Falls das Objekt bereits durch einen anderen Benutzer ausgecheckt ist (erkenntlich an einem roten Kreuz vor dem Objektnamen im Object Organizer), wird eine Meldung ausgegeben.

Einchecken:

Die Datenbankfunktion Einchecken bedeutet, dass eine neue Version eines Objekts in der Datenbank angelegt wird. Die alten Versionen bleiben erhalten. Die möglichen Zeitpunkte:
Bei Projekt Speichern: Wenn diese Option aktiviert ist, wird jeder veränderte Baustein automatisch bei jedem Speichern des Projekts eingecheckt.
Nach erfolgreichem Compile: Wenn diese Option aktiviert ist, wird nach jedem fehlerfreiem Übersetzungslauf des Projekts jedes veränderte Objekt eingecheckt.

Für die Punkte Abrufen, Aus- und Einchecken kann jeweils die Option **mit Nachfrage** aktiviert werden. In diesem Fall erscheint, bevor die betreffende Aktion ausgeführt wird, ein Dialog, in dem der Anwender nochmals bestätigen muss bzw. abbrechen kann.

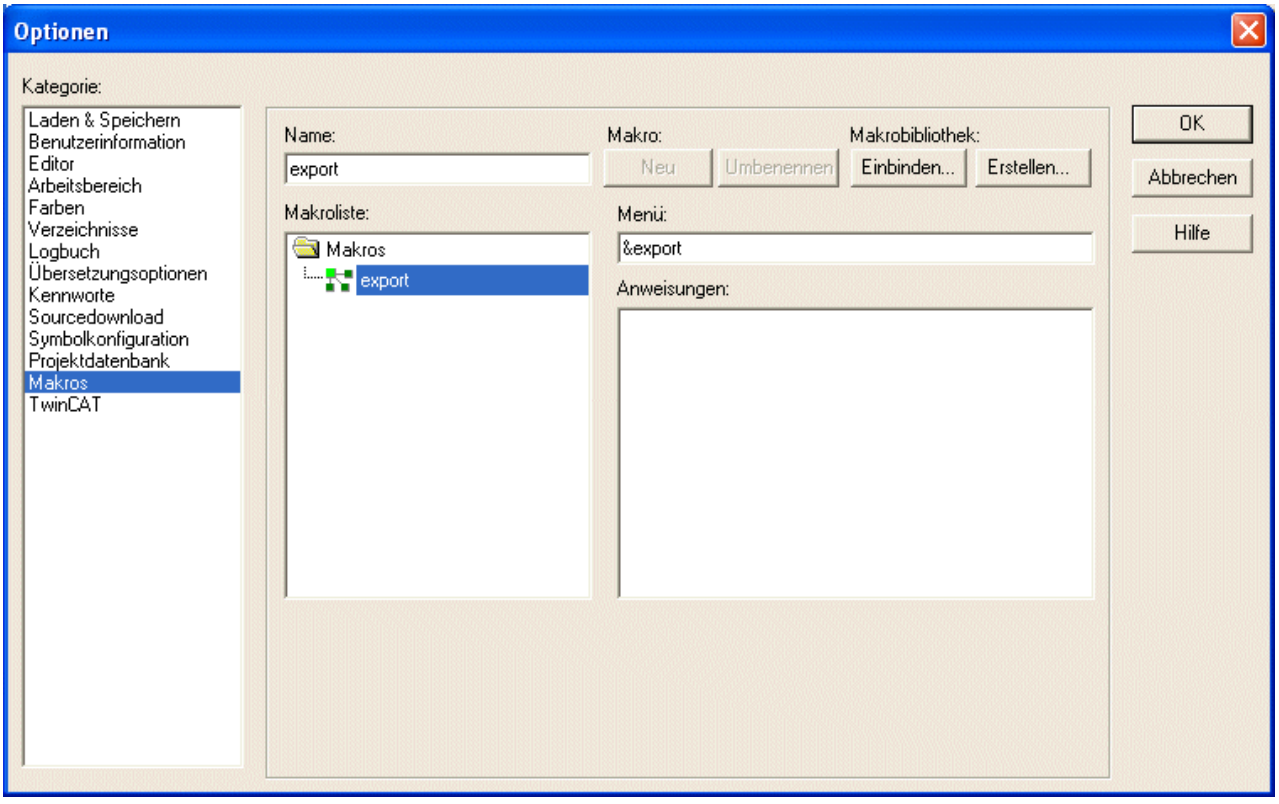
Nur bei Dialog Übersetzungsdateien:

ASCII-Symbolinformation erzeugen (.sym) **Binär-Symbolinformation erzeugen (.sdb)**	Wenn diese Option aktiviert ist, wird, sobald eine Symboldatei *.sym (Textformat) bzw. *.sdb (Binärformat) erzeugt wird, diese auch in die Datenbank geschrieben. Beim Erzeugen der Symbole gelten die in den Projektoptionen in der Kategorie 'Symbolkonfiguration' gesetzten Objektattribute.
Bootprojekt erzeugen	Wenn diese Option aktiviert ist, wird, sobald ein Bootprojekt erzeugt wird, dieses auch in der Datenbank abgelegt.

Wird **Abbrechen (bzw. Cancel)** gedrückt, kehrt man ebenfalls zum Hauptdialog zurück, wobei die Einstellungen auf Registerblatt 'Übersetzungsdateien' nicht gespeichert werden. Diejenigen, die bereits für Projektobjekte und Gemeinsame Objekte vorgenommen worden waren, bleiben jedoch erhalten.

Makros:

Wenn Sie diese Kategorie wählen, wird folgender Dialog geöffnet:



In diesem Dialog können aus den Kommandos des Batch-Mechanismus Makros definiert werden, die dann im Menü 'Bearbeiten' 'Makros' aufgerufen werden können.

Gehen Sie folgendermaßen vor, um neue Makros zu definieren:

1. Tragen Sie im Eingabefeld **Name** einen Namen für das zu erstellende Makro ein. Nach Drücken der Schaltfläche **Neu** wird dieser Name in die **Makroliste** übernommen und dort als selektiert markiert. Die Makroliste ist in Baumstruktur angelegt. Die lokal angelegten Makros stehen untereinander, eventuell eingebundene Makrobibliotheken (siehe unten) erscheinen mit dem Namen der Bibliotheksdatei. Über das Plus- bzw. Minuszeichen vor dem Bibliotheksnamen kann die Liste der Bibliothekselemente auf- oder zugeklappt werden.

2. Definieren Sie im Feld **Menü**, wie der Menüeintrag heißen soll, mit dem das Makro ins Menü 'Bearbeiten' 'Makros' eingehängt wird. Um einen Buchstaben als Short-Cut zu erhalten, muß diesem das Zeichen '&' vorangestellt werden. Beispiel: der Name "Ma&kro 1" erzeugt den Menüeintrag "Makro 1".

3. Im Editorfeld **Anweisungen** geben Sie nun die Kommandos für das in der Makroliste markierte Makro neu ein. Alle Kommandos des Batch-Mechanismus [▶ 298] und die im Zusammenhang mit diesen beschriebenen Schlüsselwörter sind zulässig, Sie erhalten eine Auflistung über die Schaltfläche **Hilfe**. Eine neue Anweisungszeile wird mit <Ctrl><Eingabetaste> eingefügt. Über die rechte Maustaste erhalten Sie das Kontextmenü mit den üblichen Texteditorfunktionen. Zusammengehörige Kommando-Bestandteile können mit Anführungszeichen zusammengefasst werden.

4. Falls Sie weitere Makros anlegen wollen, führen Sie die Schritte 1-3 erneut aus, bevor Sie den Dialog mit **OK** bestätigen und schließen.

Falls Sie ein Makro wieder **löschen** wollen, selektieren Sie es in der Makroliste und drücken Sie die Taste <Entf>.

Falls Sie ein Makro umbenennen wollen, selektieren Sie es in der Makroliste, geben unter **Name** einen anderen Namen ein und drücken dann die Schaltfläche **Umbenennen**.

Um ein bestehendes Makro zu **bearbeiten**, selektieren Sie es in der Makroliste und editieren Sie in den Eingabefeldern Menü und/oder Anweisungen. Die Änderungen werden mit OK übernommen.

Beim Verlassen des Dialogs mit **OK** wird die aktuelle Beschreibung der Makros im Projekt gespeichert.

Die Makro-Menüeinträge erscheinen in der Reihenfolge ihrer Definition im Menü 'Bearbeiten' 'Makros'.

Eine Prüfung des Makros erfolgt erst beim Ausführen des Menübefehls.

Makrobibliotheken:

Die Makros können in externen Makrobibliotheken gespeichert werden, die dann beispielsweise in anderen Projekten eingebunden werden können.

• Erstellen einer Makrobibliothek aus Makros des aktuellen Projekts:

Drücken Sie die Schaltfläche **Erstellen**. Sie erhalten den Dialog '**Objekte kopieren**', der alle verfügbaren Makros auflistet. Markieren Sie die gewünschten und bestätigen Sie mit OK. Daraufhin schließt der Auswahldialog und es öffnet der Dialog '**Makrobibliothek speichern'**. Geben Sie hier einen Namen und Pfad für die zu erstellende Bibliothek ein und drücken Sie die Schaltfläche Speichern. Nun wird die Bibliothek mit **<bibliotheksname>.mac** angelegt und der Dialog geschlossen.

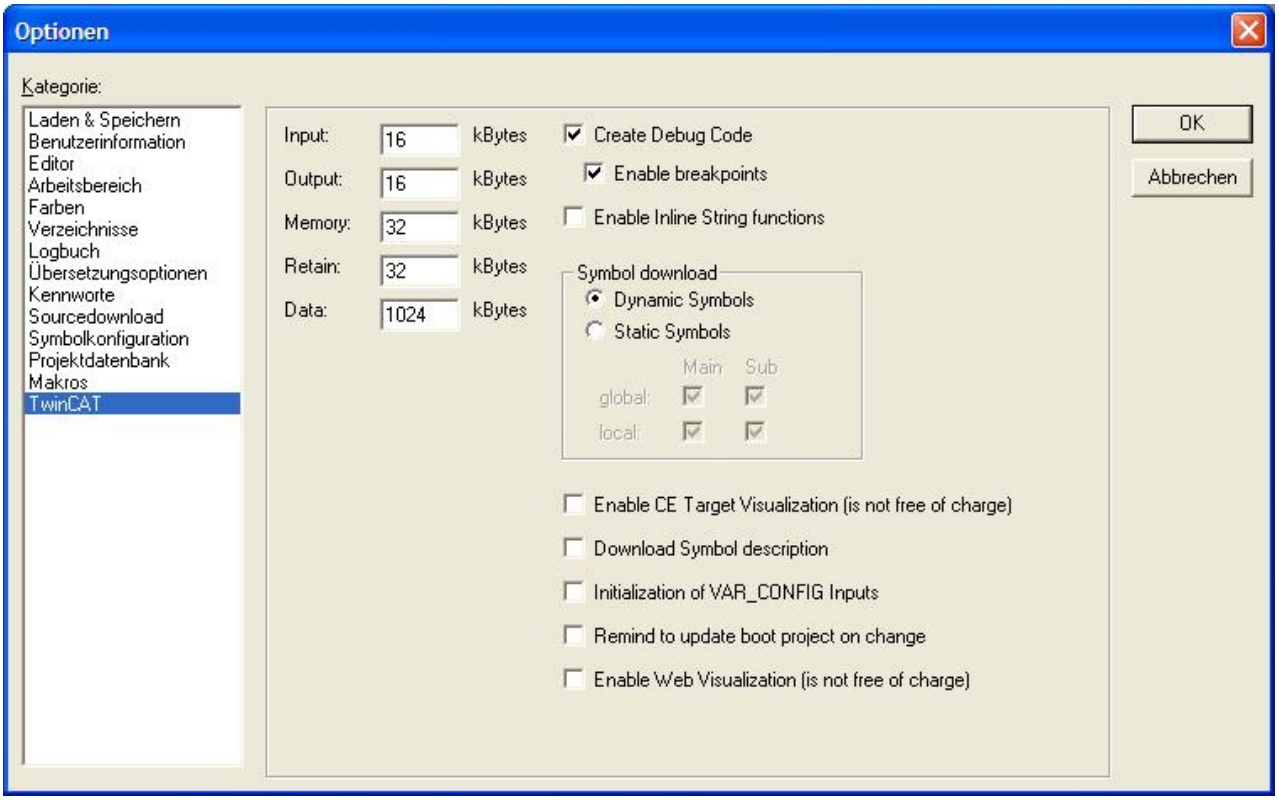
• Einbinden einer Makrobibliothek <bibliotheksname>.mac ins aktuelle Projekt:

Drücken Sie die Schaltfläche **Einbinden**. Es erscheint der Dialog **'Makrobibliothek öffnen',** der automatisch nur Dateien mit der Erweiterung *.mac anzeigt. Wählen Sie die gewünschte Bibliothek und drücken Sie die Schaltfläche Öffnen. Der Dialog schließt und die Bibliothek erscheint in der Baumstruktur der Makroliste.

Die Makros eines Projekts können auch exportiert werden ('Projekt' 'Exportieren').

TwinCAT:

Wenn Sie diese Kategorie wählen, wird folgender Dialog geöffnet:



In diesem Menü kann die Größe des adressierbaren Speichers für alle Eingangs-, Ausgangs-, und Merkervariablen eingestellt werden. Alle unlokierten Variablen liegen nicht in diesem Speicherbereich. In der Checkbox auf der rechten Seite kann generell entschieden werden, ob Breakpoints zugelassen sind oder nicht. Sollen von einem Visualisierungssystem Variablen per Namen von der SPS gelesen werden, so müssen alle nötigen Variablennamen im Laufzeitsystem der SPS bekannt sein.

Enable Inline String functions:

String-Funktionen sind nicht sicher bei Taskwechsel: Bei der Verwendung von Tasks dürfen String-Funktionen nur in einer Task eingesetzt werden. Wird die gleiche Funktion in verschiedenen Tasks benutzt, besteht die Gefahr des Überschreibens.

Mit der Anwahl dieser Funktionalität wird das Verhalten geändert und der Compiler löst dann diese Stringfunktionen direkt im PLC Code auf.

Symbol download:

Die Verwendung der **dynamischen Symbole** wird generell empfohlen. Bei dieser Art der Symbolhaltung wird bei Anforderung für ein bestimmtes Symbol das Symbol im Laufzeitsystem dynamisch zusammengebaut. Vorteile sind geringerer Speicherbedarf gegenüber der statischen Symbolhaltung und kürzere Compile- und Downloadzeiten. Die Anzahl der Handles ist auf 8192 begrenzt. Nicht benötigte Handles müssen wieder freigegeben werden.

Soll aus Kompatibilitätsgründen die **statische Symbolhaltung** verwendet werden, kann in diesem Menü der Umfang der im Laufzeitsystem der SPS abgelegten Variablen festgelegt werden. In einer Matrix können in den Zeilen die Haltung von **lokalen und globalen Variablennamen** festgelegt werden. Mit **"Haupt (Main)"** werden nur Struktur- oder Feldnamen erzeugt. Mit **"Unter (Sub)"** auch die Variablennamen für alle Struktur- oder Feldkomponenten.

Beispiel:

abc : ARRAY[1..2] OF BOOL;

Ist nur "Haupt (Main)" angewählt, so gibt es nur einen Variablennamen 'abc' und nur diese Variable ist symbolisch erreichbar. Ist auch noch "Unter (Sub) " angewählt, so können auch die Komponenten 'abc[1]' und 'abc[2]' per Symbol erreicht werden.

Enable CE Target Visualization:

Wenn diese Option angewählt ist, wird die Target-Visualisierung aktiviert. Die Bibliotheken **SysLibTargetVisu.lib** und **SysLibVisu.Lib** werden automatisch in den Bibliotheksverwalter eingefügt.

Download symbol description:

Bei Anwahl dieser Option wird die Symbolbeschreibung auf das Zielsystem geladen. Die .tpy Datei beschreibt das SPS Projekt und dient z.B. dem TwinCAT System Manager oder TwinCAT OPC als Input. Die Datei wird im Bootverzeichnis abgelegt und heißt entsprechend zum Laufzeitsystem CurrentPlc_1.tpy ... CurrentPlc_4.tpy.

Initialization of VAR_CONFIG inputs:

Die Aktivierung dieser Option bewirkt die Initialisierung der VAR_CONFIG Eingangsvariablen mit deren Initialisierungswert. Für den Fall, dass die VAR_CONFIG Variablen nicht verknüpft sind, ist dies über die SPS Diagnosebausteine erkennbar.

Der Initwert bleibt auch bei einer Nichtverknüpfung erhalten.

Remind to update boot project on change:

Wenn diese Option aktiviert ist, werden Sie bei jedem Download Ihres SPS-Projektes gefragt, ob direkt ein neues Bootprojekt erzeugt werden soll.

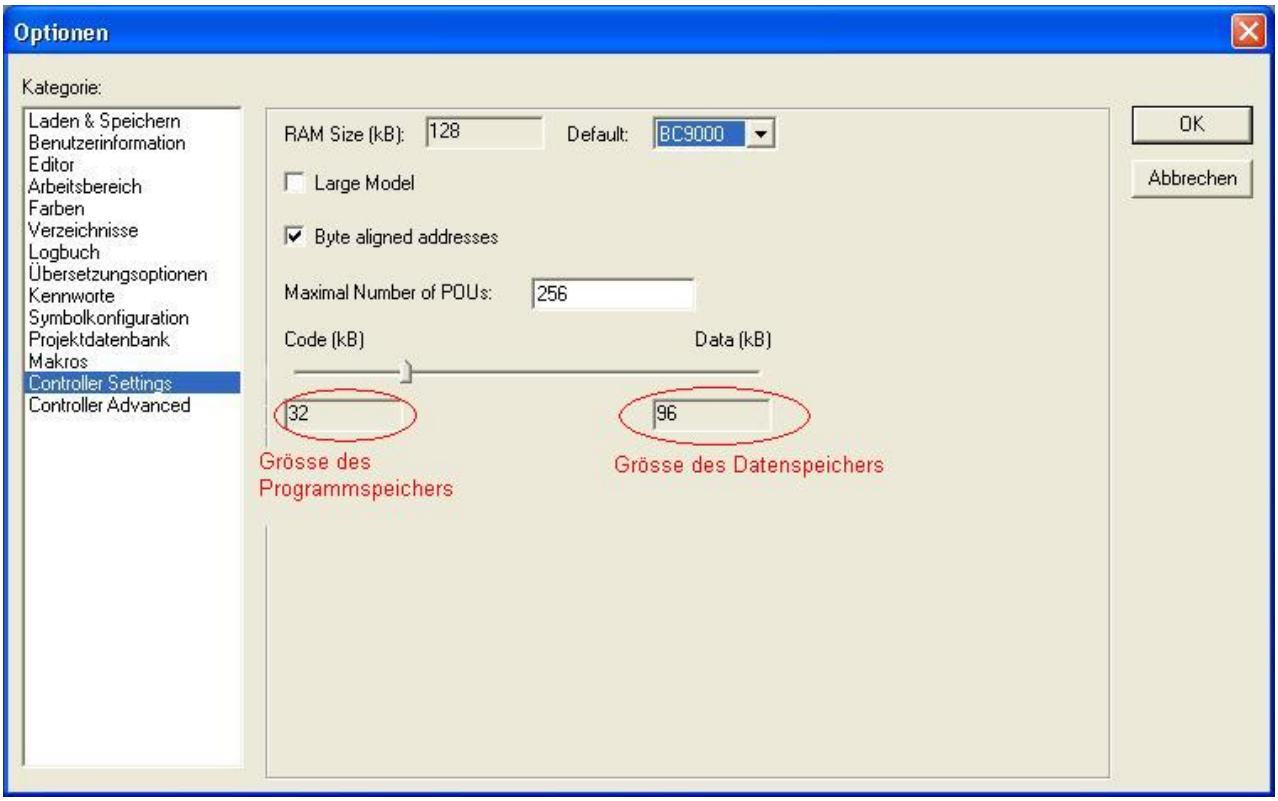
Enable Web Visualization:

Wenn diese Option angewählt ist, wird die Web-Visualisierung aktiviert.

Controller Settings

Diese Kategorie ist nur dann sichtbar, wenn ein Buscontroller als Laufzeitsystem verwendet wird.

Wenn Sie diese Kategorie wählen wird folgender Dialog geöffnet:



Hier kann man die Größe des Daten- bzw. des Programmspeichers einstellen.

- Durch das **Verschieben des Sliders nach links** wird **mehr Datenspeicher** zur Verfügung gestellt. Die Größe des Datenspeichers wird im rechten Anzeigefeld dargestellt.
- Durch das **Verschieben des Sliders nach rechts** wird **mehr Programmspeicher** zur Verfügung gestellt. Die Größe des Programmspeichers wird im linken Anzeigefeld dargestellt.
- Bei Aktivierung der Option "Large Model" ist die Speicheraufteilung fix, abhängig vom gewählten Buscontroller.

4.3 Projekte

Die Befehle, die sich auf ein ganzes Projekt beziehen, stehen unter den Menüpunkten 'Datei'und 'Projekt'. Einige der Befehle unter 'Projekt' arbeiten mit Objekten, und werden folglich im Kapitel Objekte [▶ 89] beschrieben.

Einträge unter Menüpunkt Datei:

'Datei' 'Neu'

Mit diesem Befehl legen Sie ein leeres Projekt mit dem Namen 'Unbenannt' an. Dieser Name muss beim Speichern geändert werden.

'Datei' 'Neu aus Vorlage'

Mit diesem Befehl kann ein beliebiges Projekt geöffnet werden, das als "Vorlage" verwendet werden soll, d.h. das Projekt muss nicht mit speziellen Einstellungen für diesen Zweck abgespeichert worden sein. Es erscheint der Dialog zur Auswahl einer Projektdatei, die dann mit Dateiname "Unbenannt" geöffnet wird.

'Datei' 'Öffnen'

Mit diesem Befehl öffnen Sie ein bereits bestehendes Projekt. Wenn bereits ein Projekt geöffnet ist und verändert wurde, dann fragt das TwinCAT PLC Control, ob dieses Projekt gespeichert werden soll oder nicht.

Der Dialog zum Öffnen einer Datei erscheint, und es muss eine Projektdatei mit dem Zusatz "*.pro" oder eine Bibliotheksdatei mit dem Zusatz "*.lib" ausgewählt werden. Diese Datei muss existieren; es ist nicht möglich, mit dem Befehl 'Öffnen' ein Projekt zu erzeugen.

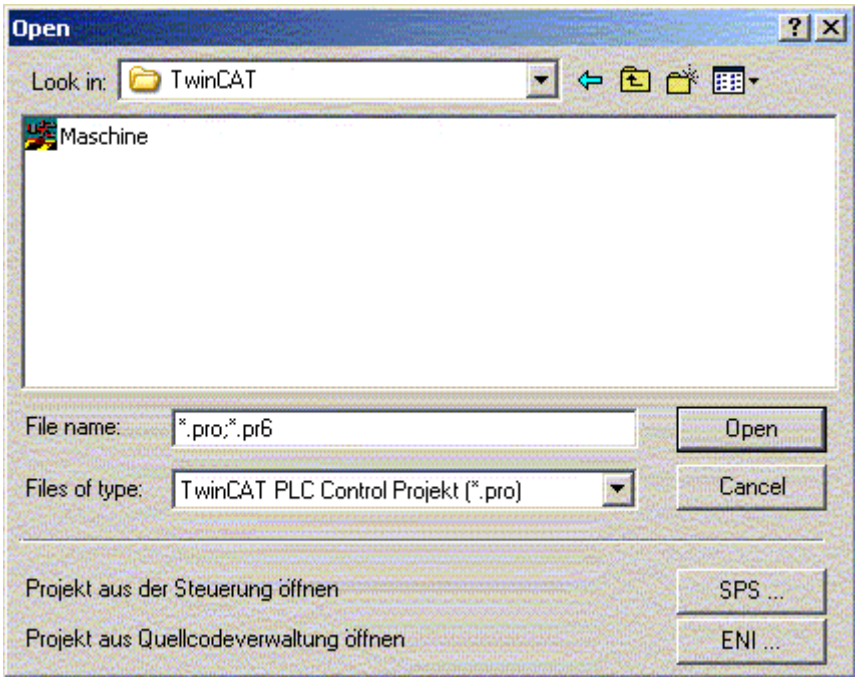
Um eine Projektdatei aus einer Steuerung hochzuladen, drücken Sie bei **Projekt aus der Steuerung öffnen** auf die Schaltfläche **SPS**. Besteht noch keine Verbindung mit der Steuerung, erhalten Sie zunächst den Dialog PLC Konfiguration zum Auswählen der Laufzeit. Ist eine Online-Verbindung hergestellt, wird geprüft, ob gleichnamige Projektdateien bereits im Verzeichnis auf Ihrem Rechner vorliegen. Ist dies der Fall, erhalten Sie den Dialog **Projekt aus der Steuerung laden**, in dem Sie entscheiden können, ob die lokalen Dateien durch die in der Steuerung verwendeten ersetzt werden sollen. (Dieser Vorgang entspricht in umgekehrter Richtung dem Vorgang 'Online' 'Quellcode laden', mit dem die Quelldatei des Projekts in der Steuerung gespeichert wird. Nicht zu verwechseln mit 'Bootprojekt erzeugen' !)

HINWEIS

Daten speichern

Beachten Sie, dass nach dem Hochladen eines Projekts dieses noch ohne Namen ist. Sie müssen es unter neuem Namen abspeichern.

Falls noch kein Projekt auf die Steuerung geladen war, erhalten Sie eine entsprechende Fehlermeldung.



Standarddialog zum Öffnen einer Datei in TwinCAT PLC Control

Die Option **Projekt aus der Quellcodeverwaltung öffnen** dient dazu, ein Projekt zu öffnen, das in einer ENI-Projektdatenbank verwaltet wird. Voraussetzung ist, dass Sie Zugang zu einem ENI-Server haben, der die Datenbank bedient. Bedienen Sie die Schaltfläche ENI... , um zuerst den Dialog 'Projektobjekte' zum Aufbau der Verbindung zum Server zu erhalten.

Geben Sie hier die entsprechenden Zugangsdaten (TCP/IP-Adresse, Port, Benutzername, Passwort, Nur lesender Zugriff) und das Verzeichnis der Datenbank (Projektname), aus dem die Objekte aus der Datenbank abgerufen werden sollen, ein und bestätigen Sie mit **Weiter.** Daraufhin schließt der Dialog und es öffnet sich der entsprechende für die Kategorie '**Gemeinsame Objekte**'. Geben Sie auch hier Ihre Zugangsdaten ein. Mit Fertigstellen wird dieser Dialog geschlossen und die Objekte der eingestellten Verzeichnisse automatisch abgerufen und geöffnet. Nun können Sie in den Projektoptionen die gewünschten Einstellungen vornehmen, die für die weitere Bearbeitung des Projekts gelten sollen. Wenn Sie das Projekt weiterhin in der Datenbank verwalten wollen, parametrieren Sie dementsprechend in den Dialogen der Kategorie Projektdatenbank.

Im Menü Datei sind unterhalb des Menüpunktes ‘**Beenden**’ die zuletzt geöffneten Projekte aufgelistet. Wenn Sie eines davon auswählen, wird dieses Projekt geöffnet.

Sind für das Projekt Kennworte oder Arbeitsgruppen definiert worden, so erscheint ein Dialog zur Eingabe des Passworts.

'Datei' 'Schließen'

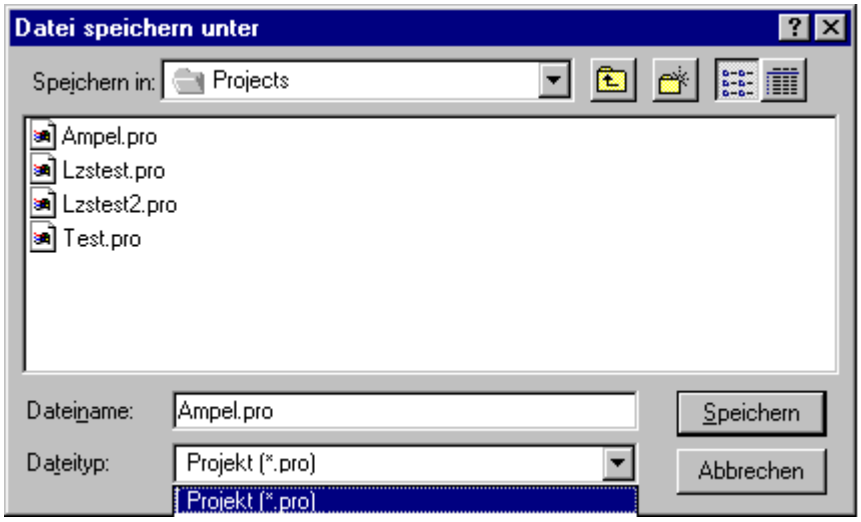
Mit diesem Befehl schließen Sie das aktuell geöffnete Projekt. Wenn das Projekt verändert wurde, fragt TwinCAT PLC Control, ob diese Veränderungen gespeichert werden sollen oder nicht. Wenn das zu speichernde Projekt den Namen "Unbenannt" trägt muss ein Name dafür festgelegt werden (siehe 'Datei "Speichern unter').

'Datei' 'Speichern' Kurzform: <Strg>+<S>

Mit diesem Befehl speichern Sie das Projekt, sofern es verändert wurde. Wenn das zu speichernde Projekt den Namen "Unbenannt" trägt, muss ein Name dafür festgelegt werden (siehe 'Datei' 'Speichern unter').

'Datei' 'Speichern unter'

Mit diesem Befehl kann das aktuelle Projekt in einer anderen Datei oder als Bibliothek gespeichert werden. Die ursprüngliche Projektdatei bleibt dabei unverändert. Nachdem der Befehl gewählt wurde, erscheint der Dialog zum Speichern. Wählen Sie entweder einen existierenden Dateinamen, oder geben Sie einen neuen Dateinamen ein und wählen Sie den gewünschten Dateityp.



Dialog zum 'Speichern unter'

Sie können das aktuelle Projekt auch als Bibliothek abspeichern, um es in anderen Projekten benutzen zu können. Wählen Sie den Dateityp **Interne Bibliothek (*.lib)**, wenn Sie Ihre Bausteine in TwinCAT PLC Control programmiert haben.

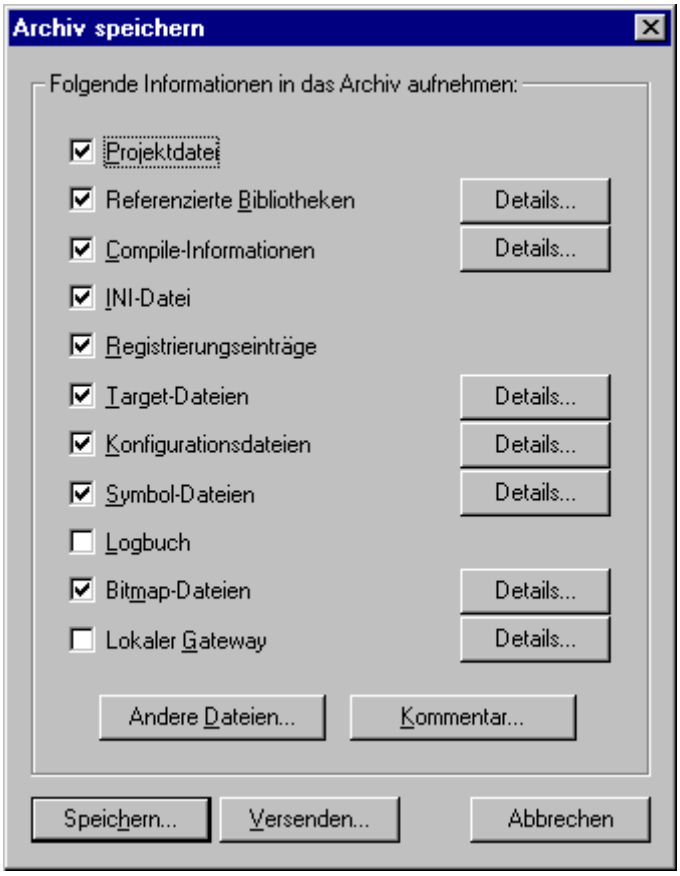
Wählen Sie den Dateityp Externe Bibliothek (*.lib), wenn Sie Bausteine in anderen Programmiersprachen (z.B. C) implementiert haben und einbinden wollen. Dies hat zur Folge, dass eine weitere Datei mit abgespeichert wird, die den Dateinamen der Bibliothek erhält, allerdings mit dem Zusatz "*.h". Diese Datei ist als C-Header-Datei aufgebaut und enthält die Deklarationen aller Bausteine, Datentypen und globalen Variablen.

Klicken Sie anschließend **OK**. Das aktuelle Projekt wird in der angegebenen Datei gespeichert. Wenn der neue Dateiname bereits existiert, wird gefragt, ob diese Datei überschrieben werden soll.

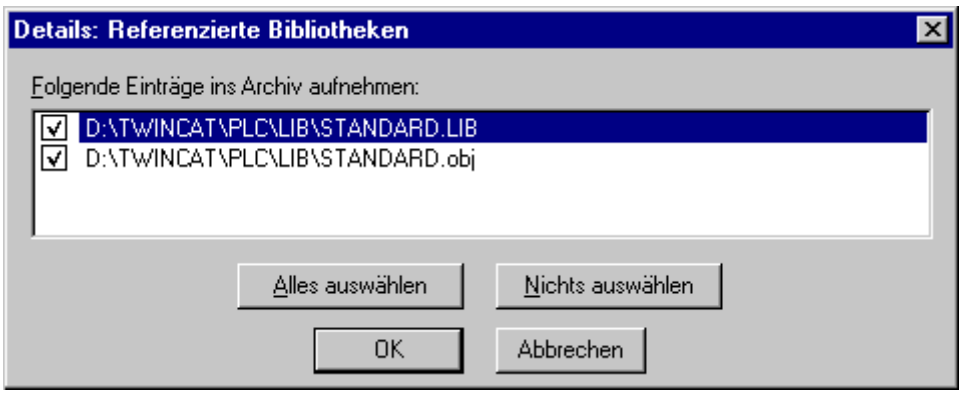
Bei 'Speichern als Bibliothek' wird das gesamte Projekt kompiliert. Wenn dabei ein Übersetzungsfehler auftritt, wird das Projekt nicht als Bibliothek gespeichert und es erscheint ein entsprechender Hinweis.

'Datei' 'Archiv speichern/versenden...'

Mit diesem Befehl kann ein komprimiertes zip-Archiv erstellt werden, das alle für ein Projekt relevanten Dateien enthält. Die zip-Datei kann im Dateisystem abgespeichert oder direkt in einer Email versendet werden.
Nach Ausführen des Befehls wird der Dialog 'Archiv speichern' geöffnet:



Hier wird definiert, welche Datei-Kategorien dem Projekt-Archiv hinzugefügt werden sollen. Eine Kategorie gilt als ausgewählt, wenn die zugehörige Kontrollbox mit einem Haken versehen ist. Dies wird erreicht über einen einfachen Mausklick in das Kästchen oder über einen Doppelklick auf die Kategoriebezeichnung. Für jede Kategorie, die ausgewählt ist, werden grundsätzlich alle relevanten Dateien in die zip-Datei kopiert (siehe unten, Tabelle). Für einige Kategorien kann jedoch eine Teilauswahl festgelegt werden. Dazu steht der Dialog 'Details' zur Verfügung, der über die zugehörige Schaltfläche Details geöffnet wird:

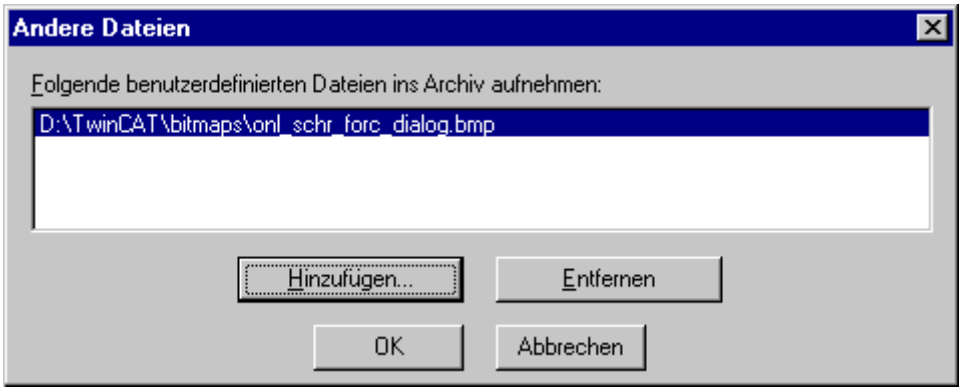


Der Dialog **Details** zeigt eine Liste aller in dieser Kategorie verfügbaren Dateien. Aktivieren bzw. deaktivieren Sie die gewünschten Dateien: Mit den Schaltflächen **Alles auswählen** bzw. **Nichts auswählen** können Sie jeweils alle Dateien der Liste erfassen, ein Mausklick in die Kontrollbox aktiviert bzw. deaktiviert eine einzelne Datei, ebenso ein Doppelklick auf den Eintrag. Außerdem kann durch Drücken der <Eingabe>-Taste ein Eintrag (de-)aktiviert werden, wenn er markiert ist.
Wenn der Details-Dialog mit **OK** geschlossen wird, wird die getroffene Auswahl übernommen. Die Einstellung wird bis zur endgültigen Erstellung des zip-Archivs gespeichert.

Im Hauptdialog **Archiv speichern** erkennt man die Kategorien, für die eine Teilauswahl vorgenommen wurde, am grauen Hintergrund der Kontrollbox.
Die folgende Tabelle zeigt, welche Dateikategorien vordefiniert sind und welche Dateien sie jeweils automatisch anziehen:

Kategorie	Zugehörige Dateien
Projektdatei	<Projektname>.pro (die TwinCAT PLC Control Projektdatei)
Referenzierte Bibliotheken	*.lib, *.obj, *.hex (Bibliotheken und ggfs. die zugehörigen obj- und hex-Dateien)
Compile-Informationen	*.ci (Information des letzten Übersetzungslaufs), *.ri (Information des letzten Downloads)
INI-Datei	TwinCAT PLC Ctrl.ini
Registrierungseinträge	
Symbol-Dateien	*.sdb, *.sym (Aus dem Projekt erzeugte Symbolinformation)
Logbuch	*.log (Projekt-Logbuch)
Bitmap-Dateien	*.bmp (Bitmaps, die in Projektbausteinen verwendet werden)

Um beliebige andere Dateien zum zip-Archiv hinzuzufügen, öffnen Sie über die Schaltfläche **Andere Dateien...** den gleichnamigen Dialog.



Hier kann eine benutzerdefinierte Liste von Dateien erstellt werden. Dazu wird über die Schaltfläche **Hinzufügen** der Standarddialog zum Öffnen einer Datei geöffnet. Wählen Sie eine Datei aus und bestätigen Sie mit **Öffnen**. Die Datei wird daraufhin in der Liste im Dialog 'Andere Dateien ...' eingefügt. Über die Schaltfläche Entfernen kann ein Eintrag in der Liste gelöscht werden . Wenn die Liste fertig erstellt ist, wird der Dialog mit OK geschlossen, um die Einträge bis zum Erstellen der zip-Datei zu speichern.

Um dem zip-Archiv eine Readme-Datei hinzuzufügen, drücken Sie die Schaltfläche **Kommentar....** Ein gleichnamiger Dialog öffnet, der ein Editierfeld enthält. Hier kann beliebiger Test eingegeben werden. Wird der Dialog mit OK geschlossen, wird bei der Erstellung des zip-Archivs eine Datei **Readme.txt** erstellt. Sie enthält den vom Benutzer eingegebenen Text, dem automatisch das Erstellungs(Build)datum hinzugefügt wird.

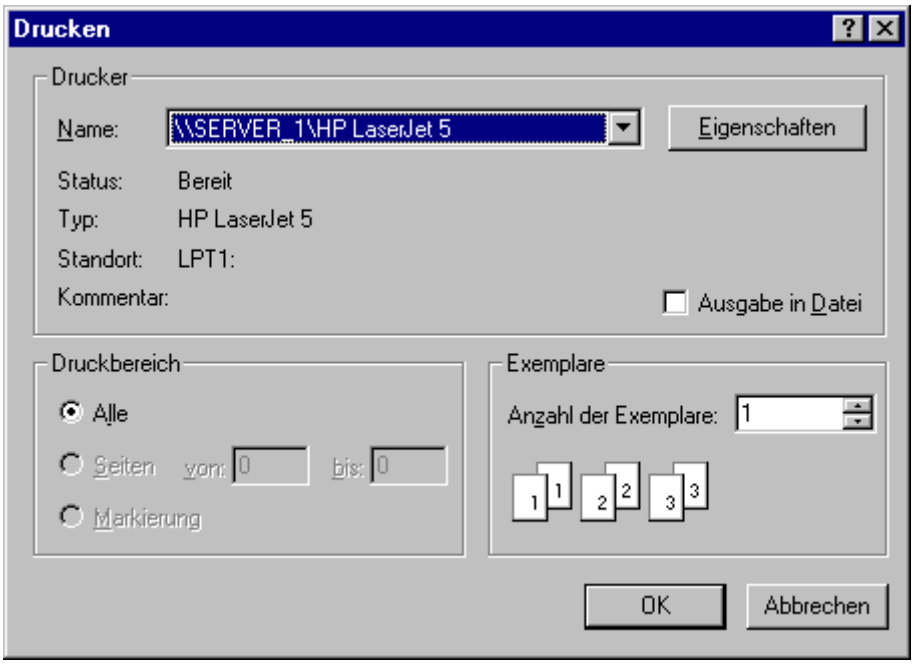
Generieren des zip-Archivs:

Wenn alle gewünschten Einstellungen vorgenommen wurden, kann im Hauptdialog das zip-Archiv erstellt werden. Folgende Schaltflächen stehen zur Verfügung:

- **Speichern...** erstellt und speichert die zip-Datei. Der Standarddialog zum Speichern einer Datei öffnet und es kann angegeben werden, wo die Datei abgelegt werden soll. Der Name der zip-Datei ist per Default <projektname>.zip. Wenn mit Speichern bestätigt wird, startet die Generierung des Archivs. Der Ablauf wird von einem Fortschrittsdialog begleitet und im Meldungsfenster mitprotokolliert.
- **Senden…** erstellt eine temporäre zip-Datei und generiert automatisch eine leere email, die das zip als Anhang enthält. Diese Funktion setzt eine korrekte Installation des MAPI (Messaging Application Programming Interface) voraus. Während die email erzeugt wird, erscheint ein Fortschrittsdialog und der Ablauf wird im Meldungsfenster mitprotokolliert. Die temporäre zip-Datei wird gelöscht, sobald sie der email als Anhang beigefügt ist.
- **Abbrechen:** Der Dialog wird ohne Erstellen eines zip-Archivs geschlossen, die vorgenommenen Einstellungen werden nicht gespeichert

'Datei' 'Drucken' Kurzform: <Strg>+<P>

Mit diesem Befehl wird der Inhalt des aktiven Fensters gedruckt. Nachdem der Befehl gewählt wurde, erscheint der Dialog zum Drucken. Wählen Sie die gewünschte Option, oder konfigurieren Sie den Drucker und klicken anschließend OK. Das aktive Fenster wird ausgedruckt. Farbausdrucke aus allen Editoren sind möglich.



Dialog zum Drucken

Sie können die**Anzahl der Exemplare** angeben und die Ausgabe in eine Datei umleiten. Mit der Schaltfläche **Eigenschaften** öffnen Sie den Dialog zur Druckereinrichtung. Das Layout Ihres Ausdruckes können Sie mit dem Befehl **'Datei' 'Einstellungen Dokumentation'** festlegen. Während des Druckens wird Ihnen in einer Dialogbox die Anzahl der bereits gedruckten Seiten mitgeteilt. Wenn Sie diese Dialogbox schließen, stoppt der Druckvorgang nach der nächsten Seite. Um Ihr gesamtes Projekt zu dokumentieren, benutzen Sie den Befehl **'Projekt''Dokumentieren'.** Wenn Sie eine Dokumentvorlage zu Ihrem Projekt erstellen wollen, dann öffnen Sie eine globale Variablenliste und benutzen den Befehl **'Extras' 'Doku-Vorlage erstellen'.**

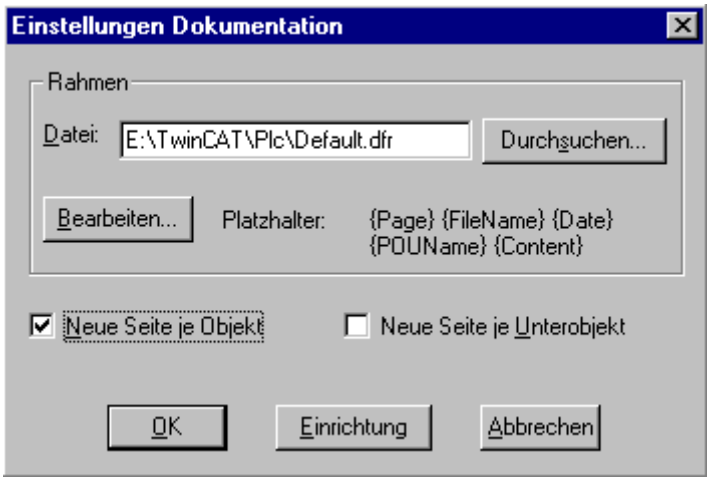
Ist der Fokus im Meldungsfenster, so wird der gesamte Inhalt ausgedruckt, und zwar zeilenweise, wie im Fenster dargestellt.

Möglicher Inhalt:

- Übersetzungsausgabe,
- Querverweisliste,
- Suchergebnis,
- Vergleichsergebnis,
- Batch-Protokollierung.

'Datei''Einstellungen Dokumentation'

Mit diesem Befehl können Sie das Layout der ausgedruckten Seiten festlegen. Es wird nun folgender Dialog geöffnet:



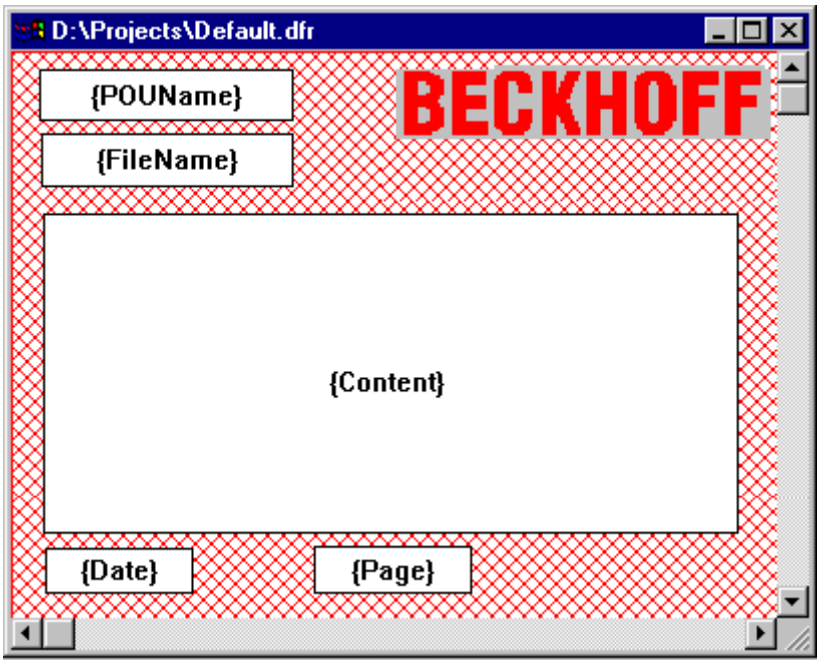
Dialog zur Einstellung des Seitenlayouts der Dokumentation

Im Feld **Datei** können Sie den Namen der Datei mit Zusatz ".dfr" eintragen, in die das Seitenlayout gespeichert werden soll. Standardmäßig wird die Vorlage in der Datei DEFAULT.DFR abgelegt.

Wenn Sie ein vorhandenes Layout verändern möchten, browsen Sie durch den Verzeichnisbaum auf der Suche nach der gewünschten Datei mit der Schaltfläche **'Durchsuchen'**.

Sie können ebenfalls auswählen, ob eine **neue Seite für jedes Objekt** und für jedes **Unterobjekt** begonnen wird. Mit der Schaltfläche 'Einrichtung' öffnen Sie die Druckereinrichtung.

Wenn Sie auf die Schaltfläche **Bearbeiten** klicken, erscheint die Vorlage zur Einstellung des Seitenlayouts. Hier können Sie Seitenzahlen, Datum, Datei- und Bausteinname, sowie Grafiken auf der Seite platzieren und den Textbereich, in den die Dokumentation gedruckt werden soll, festlegen.



Fenster zum Einfügen der Platzhalter auf dem Seitenlayout

Mit dem Menüpunkt **'Einfügen''Platzhalter'** und durch anschließende Auswahl einer der fünf Platzhalter (**Seite, Bausteinname, Dateiname, Datum, Inhalt**), können Sie durch Aufziehen eines Rechteckes auf dem Layout einen sogenannten Platzhalter einfügen. Diese werden im Ausdruck wie folgt ersetzt:

Befehl	Platzhalter	Beschreibung
Seite	{Page}	Hier erscheint die aktuelle Seitenzahl im Ausdruck.
Bausteinname	{POUName}	Hier erscheint der Name des aktuellen Bausteins.
Dateiname	{FileName}	Hier erscheint der Name des Projekts.
Datum	{Date}	Hier erscheint das aktuelle Datum.
Inhalt	{Content}	Hier erscheint der Inhalt des Bausteins.

Ferner können Sie mit **'Einfügen"Bitmap'** eine Bitmap-Grafik (z.B. Firmenlogo) in die Seite einfügen. Dabei müssen Sie nach Auswahl der Grafik ebenfalls ein Rechteck mit der Maus auf dem Layout aufziehen.

Wenn die Vorlage verändert wurde, fragt TwinCAT PLC Control beim Schließen des Fensters, ob diese Veränderungen gespeichert werden sollen oder nicht.

'Datei''Beenden' Kurzform: <Alt>+<F4>

Mit diesem Befehl beenden Sie TwinCAT PLC Control. Wenn ein Projekt geöffnet ist, wird es geschlossen wie in 'Datei"Speichern' beschrieben.

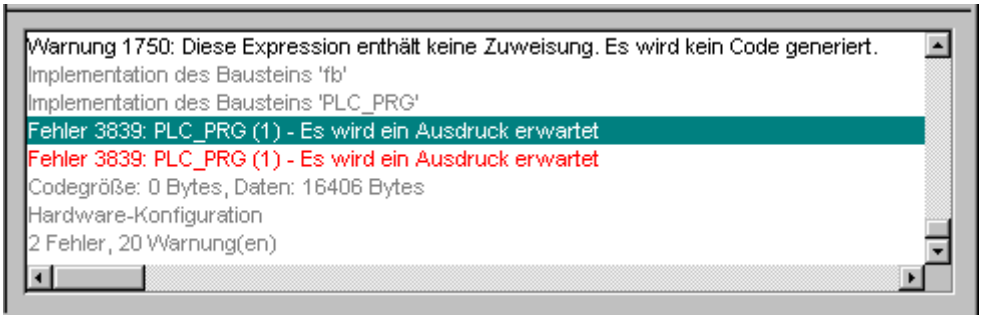
Es folgen jetzt die unter Projekt möglichen Aktionen:

'Projekt' 'Übersetzen' Kurzform: <Strg>+<F8>

Mit **'Projekt' 'Übersetzen'** wird das Projekt kompiliert. Der Übersetzungsvorgang ist grundsätzlich inkrementell, d.h. es werden nur die veränderten Bausteine neu übersetzt. Ein nicht inkrementeller Übersetzungsvorgang wird auch mit diesem Befehl erreicht, wenn vorher der Befehl **'Projekt' 'Alles bereinigen'** ausgeführt wurde.
Für das Zielsystem 'PC', das Online Change unterstützt, sind nach dem Übersetzungslauf alle Bausteine im Objekt Manager mit einem blauen Pfeil gekennzeichnet, die beim nächsten Download auf die Steuerung geladen werden.

Der Übersetzungslauf, der mit **'Projekt' 'Übersetzen'** durchgeführt wird, erfolgt automatisch, wenn über **'Online' 'Einloggen'** in die Steuerung eingeloggt wird.

Beim Übersetzen wird das Meldungsfenster geöffnet, in dem das Fortschreiten des Übersetzungsvorgangs und die während des Übersetzens eventuell auftretenden Fehler und Warnungen ausgegeben werden. Fehler und Warnungen sind mit Nummern gekennzeichnet. Über F1 erhalten Sie weitere Informationen zum aktuell markierten Fehler.
Eine Auflistung sämtlicher Fehlermeldungen [▶ 276] befindet sich im Anhang.



Ist die Option **Sichern vor Übersetzen** im Optionsdialog in der Kategorie Laden & Speichern gewählt, so wird das Projekt vor dem Übersetzen gespeichert.

HINWEIS

Die Querverweise entstehen während der Kompilierung und werden in der Übersetzungsinformation mitgespeichert. Um die Befehle **Aufrufbaum ausgeben**, **Querverweisliste ausgeben** und die Befehle **Unbenutzte Variablen**, **Konkurrierender Zugriff** und **Mehrfaches Schreiben auf Output** des Menüs **'Projekt' 'Überprüfen'** anwenden zu können, muss das Projekt nach einer Veränderung neu übersetzt werden.

'Projekt' 'Alles übersetzen'

Mit **'Projekt' 'Alles übersetzen'** wird im Gegensatz zum inkrementellen Übersetzen (**'Projekt' 'Übersetzen'**) das Projekt komplett neu kompiliert. Dabei wird allerdings die Download-Information nicht verworfen, wie es beim Befehl **'Alles bereinigen'** der Fall ist.

'Projekt' 'Alles bereinigen'

Mit diesem Befehl werden die Informationen des letzten Downloads und des letzten Übersetzungsvorgangs gelöscht.
Nach Anwählen des Befehls erscheint eine Dialogbox, die darauf hinweist, dass ein Login ohne erneuten Download nicht mehr möglich ist. Hier kann der Befehl abgebrochen oder bestätigt werden.

Ein Login ist nach **'Alles bereinigen'** nur dann möglich, wenn vorher die Datei *.ri mit den Projektinformationen des letzten Downloads explizit außerhalb des Projektverzeichnisses abgelegt worden war (siehe 'Download-Information laden') und nun vor dem Einloggen wieder geladen werden kann.

'Projekt' 'Download-Information laden'

Mit diesem Befehl kann die projektzugehörige Download-Information gezielt wieder geladen werden, wenn sie in ein anderes als das Projektverzeichnis gespeichert worden war. Nach Drücken des Befehls öffnet dazu der Standarddialog **'Datei Öffnen'**.

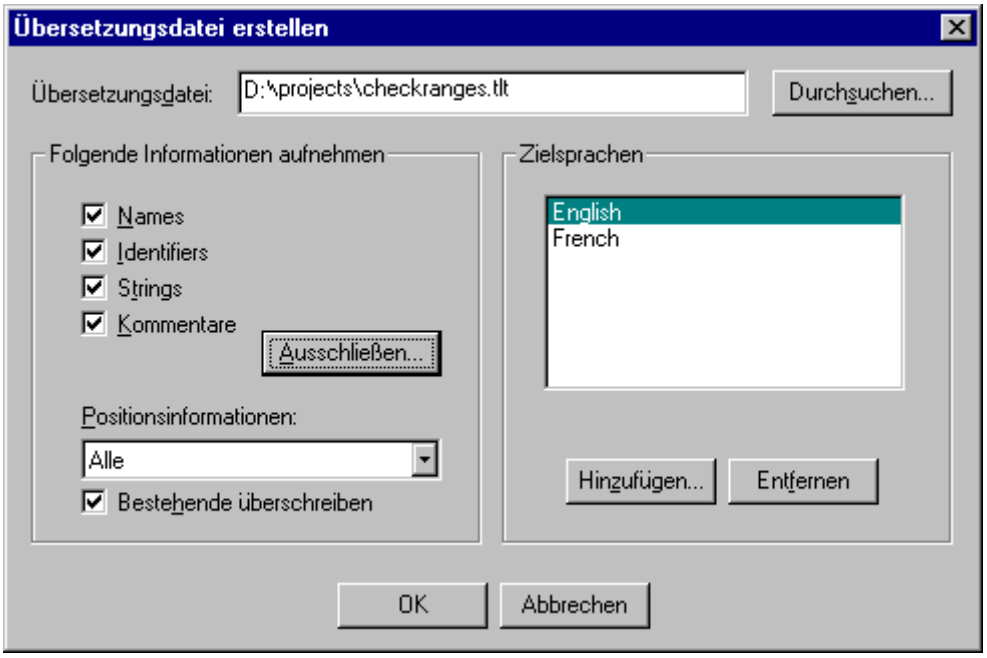
Die Download-Information wird automatisch bei jedem Download in eine Datei gespeichert, die den Namen **<Projektname><Targetidentifier>.ri** erhält und ins Projektverzeichnis gelegt wird. Sie wird bei jedem Öffnen des Projekts automatisch wieder mit geladen und dient beim erneuten Einloggen auf die Steuerung zum einen dazu, festzustellen, ob das Projekt auf der Steuerung dem gerade geöffneten entspricht (ID-Check). Außerdem wird geprüft, bei welchen Bausteinen sich der generierte Code verändert hat. Nur diese Bausteine werden bei Systemen, die Online Change unterstützen, beim Download erneut geladen. Wurde die *.ri-Datei allerdings über den Befehl **'Projekt' 'Alles bereinigen'** aus dem Projektverzeichnis gelöscht, kann die Download-Information mit 'Projekt' 'Download-Information laden' explizit von einem anderen Verzeichnis geladen werden, wenn die *.ri-Datei dort abgelegt wurde.

Dieser Menüpunkt dient dazu, die aktuelle Projektdatei in eine andere Sprache zu übersetzen. Dies geschieht durch das Einlesen einer Übersetzungsdatei, die aus dem Projekt heraus erzeugt wurde und extern mithilfe eines Texteditors mit Übersetzungstexten in der gewünschten Landessprache ergänzt wurde. Dazu gibt es zwei Untermenüpunkte:

- Übersetzungsdatei erstellen
- Projekt übersetzen

Übersetzungsdatei erstellen

Dieser Befehl des Menüs **'Projekt' 'In andere Sprache übersetzen'** führt zum Dialog **'Übersetzungsdatei erstellen'**:



Geben Sie im Feld **Übersetzungsdatei** einen Pfad ein, der anzeigt wo die Datei gespeichert werden soll. Die Default-Dateierweiterung ist ***.tlt,** es handelt sich um eine Textdatei. Ebenso möglich ist die Verwendung der Erweiterung ***.txt,** was empfehlenswert ist, falls die Datei beispielsweise in EXCEL oder WORD bearbeitet werden soll, da in diesem Fall die Daten in Tabellenform angeordnet werden.

Existiert bereits eine Übersetzungsdatei, die Sie bearbeiten wollen, geben Sie den Pfad dieser Datei ein bzw. verwenden Sie den über die Schaltfläche Durchsuchen erreichbaren Standard-Windows-Dialog zum Auswählen einer Datei.

Folgende Informationen aus dem Projekt können der zu erstellenden bzw. zu modifizierenden Übersetzungsdatei optional mitgegeben werden, so dass sie in dieser zur Übersetzung zur Verfügung stehen: **Names** (Namen, z.B. der Titel 'Bausteine' im Objekt Organizer), **Identifiers** (Bezeichner), **Strings**, **Kommentare**. Zusätzlich können die **Positionsinformationen** zu diesen Projektelementen übernommen werden.

Sind die entsprechenden Optionen mit einem Haken versehen, werden die Informationen als Sprachsymbole aus dem aktuellen Projekt in eine neu zu erstellende Übersetzungsdatei aufgenommen bzw. in einer bereits existierenden ergänzt. Falls die jeweilige Option nicht angewählt ist, werden sämtliche Informationen der betreffenden Kategorie, gleichgültig aus welchem Projekt sie stammen, aus der Übersetzungsdatei entfernt.

Positionsinformationen: Diese beschreibt mit den Angaben Dateipfad, Baustein, Zeile die Position des Sprachsymbols, das zur Übersetzung bereitgestellt wird. Zur Auswahl stehen hier drei Optionen:

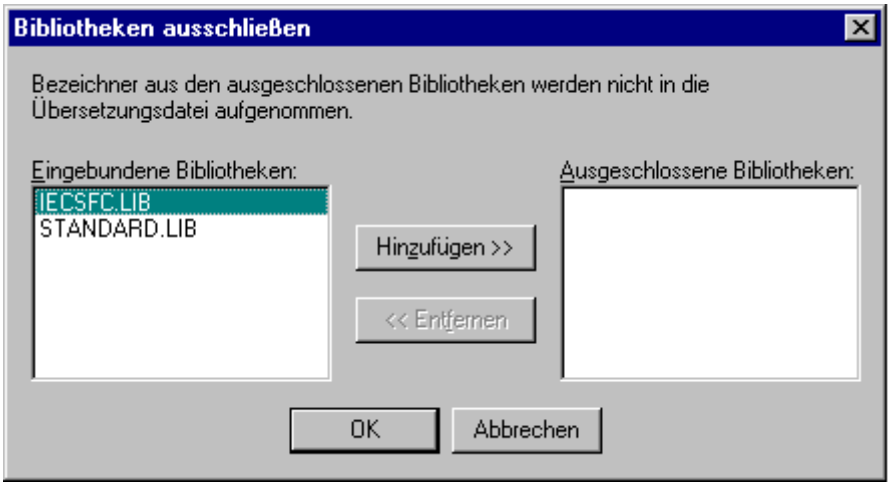
- 'Keine': Es werden keine Positionsinformationen generiert.
- 'Erstmaliges Auftreten': Es wird die Position in die Übersetzungsdatei aufgenommen, an der das zu übersetzende Element erstmalig auftritt.
- 'Alle': Es werden alle Positionen angegeben, an denen das betreffende Element im Projekt auftritt.

Falls eine früher erstellte Übersetzungsdatei editiert wird, die bereits mehr Positionsinformationen enthält als hier ausgewählt, so werden diese entsprechend gekürzt oder ganz gelöscht, gleichgültig aus welchem Projekt sie generiert wurden.

Pro Element (Sprachsymbol) werden maximal 64 Positionsinformationen generiert, selbst wenn der Anwender im Dialog Übersetzungsdatei erstellen unter Positionsinformationen "Alle" ausgewählt hat.

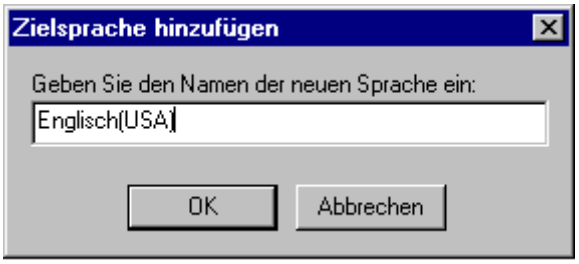
Bestehende überschreiben: Sämtliche bereits existierende Positionsinformationen in der Übersetzungsdatei, die aktuell bearbeitet wird, werden überschrieben, gleichgültig von welchem Projekt sie generiert wurden.

Zielsprachen: Diese Liste enthält Bezeichner für alle Sprachen, die in der Übersetzungsdatei enthalten sind bzw. nach Beenden des Dialogs **'Übersetzungsdatei erstellen'** aufgenommen werden sollen.
Die Schaltfläche **Ausschließen** öffnet den Dialog 'Bibliotheken ausschließen':



Hier können aus den ins Projekt eingebundenen Bibliotheken diejenigen ausgewählt werden, deren Bezeichnerinformationen nicht in die Übersetzungsdatei übernommen werden sollen. Dazu wird der betreffende Eintrag aus der linken Tabelle **Eingebundene Bibliotheken** mit der Maus ausgewählt und über die Schaltfläche **Hinzufügen** in die rechte Tabelle Ausgeschlossene Bibliotheken gebracht. Ebenso kann mit der Schaltfläche **Entfernen** ein dort gewählter Eintrag wieder gelöscht werden. Mit **OK** wird die Einstellung bestätigt und der Dialog geschlossen.

Die Schaltfläche **Hinzufügen** öffnet den Dialog '**Zielsprache hinzufügen':**



Im Editierfeld muss ein Sprachbezeichner eingegeben werden, der weder am Anfang noch am Ende ein Leerzeichen oder einen Umlaut (ä, ö, ü) enthalten darf.

Mit **OK** wird der Dialog **'Zielsprache hinzufügen'** geschlossen und die neue Zielsprache erscheint in der Zielsprachen-Liste.

Die Schaltfläche **Entfernen** löscht einen in der Liste selektierten Eintrag.
Ebenfalls über **OK** können Sie dann den Dialog **'Übersetzungsdatei erstellen'** bestätigen, um eine Übersetzungsdatei zu generieren.

Existiert bereits eine gleichnamige Übersetzungsdatei, erhalten Sie zunächst die folgende, mit Ja oder Nein zu beantwortende Sicherheitsabfrage:
"Die angegebene Übersetzungsdatei existiert bereits. Sie wird nun entsprechend geändert, wobei eine Sicherungskopie der bereits bestehenden Datei angelegt wird. Möchten Sie fortfahren ?"

Nein kehrt ohne Aktion zum Dialog **'Übersetzungsdatei erstellen'** zurück. Wird **Ja** gewählt, so wird eine Kopie der bereits bestehenden Übersetzungsdatei mit dem Dateinamen "Backup_of_<Übersetzungsdatei>.xlt" im gleichen Verzeichnis angelegt und die betreffende Übersetzungsdatei gemäß der eingestellten Optionen modifiziert.

Beim Erzeugen einer Übersetzungsdatei geschieht folgendes:

- Für jede neue Zielsprache wird für jedes auszugebende Sprachsymbol ein Platzhalter ("##TODO") generiert.

- Wird eine bereits existierende Übersetzungsdatei bearbeitet, werden Dateieinträge von Sprachen, die in der Übersetzungsdatei, aber nicht in der Zielsprachen-Liste stehen, entfernt, gleichgültig aus welchem Projekt sie generiert wurden.

Bearbeiten der Übersetzungsdatei

Die Übersetzungsdatei ist als Textdatei zu öffnen und zu speichern. Die Zeichen ## kennzeichnen Schlüsselwörter. Die ##TODO-Platzhalter in der Datei können mit den gültigen Übersetzungstexten ersetzt werden. Pro Sprachsymbol wird ein durch ##NAME_ITEM und ##END_NAME_ITEM begrenzter Abschnitt angelegt (Für Kommentare entsprechend ##COMMENT_ITEM usw.).

Sehen Sie nachfolgend einen Beispiel-Abschnitt in der Übersetzungsdatei für den Namen eines im Projekt verwendeten Bausteins: ST_Visu. Die Zielsprachen Englisch(USA) und Französisch sind vorgesehen. In diesem Beispiel wurde auch die Positionsinformation für das zu übersetzende Projektelement mitgegeben:

vor der Übersetzung:

```
##NAME_ITEM

[D:\projects\Bspdt_22.pro::ST_Visualisierung::0]

ST_Visualisierung

##English :: ##TODO

##French :: ##TODO

##END_NAME_ITEM
```

nach der Übersetzung:

Anstelle der ##TODOs wurde nun der englische bzw. französische Ausdruck für 'Visualisierung' eingesetzt:

```
##NAME_ITEM

[D:\projects\Bspdt_22.pro::ST_Visualisierung::0]

ST_Visualisierung

##English :: ST_Visualization

##French :: ST_Visu

##END_NAME_ITEM
```

Es ist darauf zu achten, dass übersetzte Bezeichner und Namen gemäß der Norm gültig bleiben und dass Strings und Kommentare in die entsprechenden Klammerzeichen eingeschlossen werden.

Folgende Teile der Übersetzungsdatei sollten ohne genaue Kenntnis nicht modifiziert werden: Sprachblock, Flagblock, Positionsinformationen, Originaltexte.

Projekt übersetzen (in andere Sprache)

Dieser Befehl des Menüs **'Projekt' 'In andere Sprache übersetzen'** öffnet den Dialog 'Projekt in andere Sprache übersetzen'.

Das aktuelle Projekt kann unter Verwendung einer gültigen Übersetzungsdatei in eine andere Sprache übersetzt werden.

Wenn Sie die Sprachversion des Projekts, in der es erstellt wurde, erhalten wollen, speichern Sie eine Projektkopie vor dem Übersetzen unter einem anderen Namen. Ein Übersetzungslauf kann nicht rückgängig gemacht werden.

Geben Sie im Feld **Übersetzungsdatei** den Pfad der zu verwendenden Übersetzungsdatei an. Über **Durchsuchen** erhalten Sie den Standard-Windows-Dialog zum Auswählen einer Datei.

Im Feld **Zielsprache** erhalten Sie eine Liste der in der Übersetzungsdatei enthaltenen Sprachen-Bezeichner zur Auswahl der gewünschten Zielsprache.

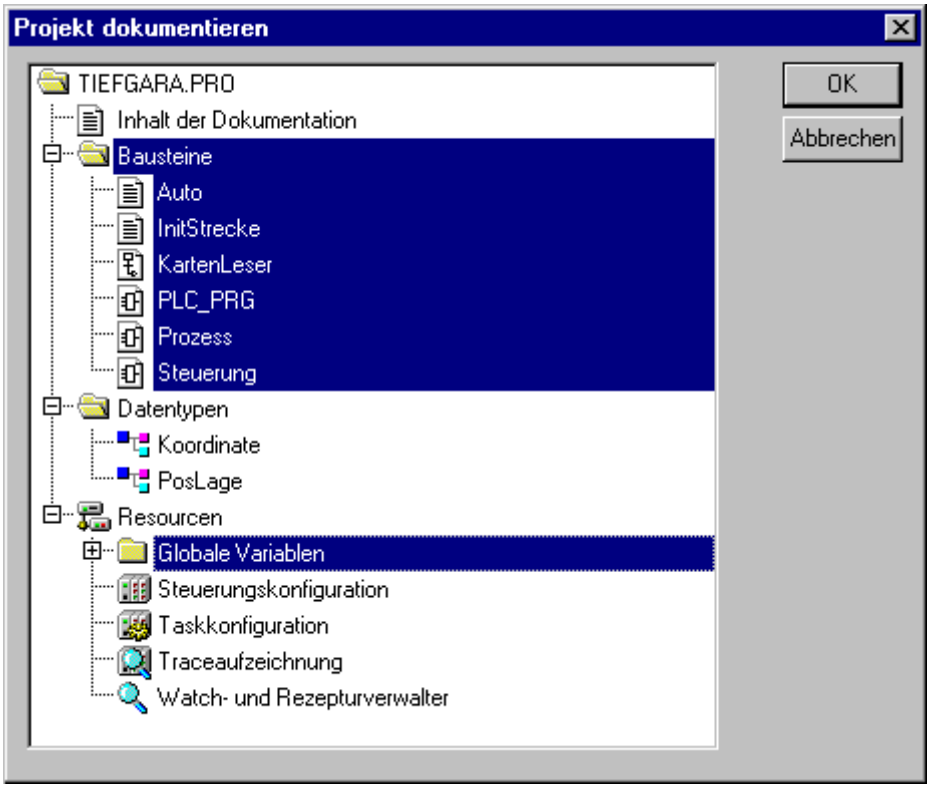
OK startet die Übersetzung des aktuellen Projekts mit Hilfe der angegebenen Übersetzungsdatei in die ausgewählte Zielsprache. Während der Übersetzung werden ein Fortschrittsdialog und gegebenenfalls Fehlermeldungen angezeigt. Nach der Übersetzung wird die Dialogbox sowie alle geöffneten Editierfenster des Projekts geschlossen.

Abbrechen schließt die Dialogbox ohne Modifizierung des aktuellen Projekts.
Falls die Übersetzungsdatei fehlerhafte Eingaben enthält, wird nach Drücken von OK eine Fehlermeldung ausgegeben, die Dateipfad und fehlerhafte Zeile ausgibt, z.B.: "[C:\Programme\projects\visu.tlt (78)]; Übersetzungstext erwartet".

'Projekt' 'Dokumentieren'

Dieser Befehl ermöglicht es Ihnen, eine Dokumentation ihres gesamten Projekts zu drucken. Zu einer vollständigen Dokumentation gehören:

- die Bausteine,
- der Inhalt der Dokumentation
- die Datentypen,
- die Ressourcen, globale Variablen, die Traceaufzeichnung, die Steuerungskonfiguration, die Taskkonfiguration, der Watch- und Rezepturverwalter)
- die Aufrufbäume von Bausteinen und Datentypen sowie
- die Querverweisliste. Für die letzten beiden Punkte muss das Projekt fehlerfrei übersetzt worden sein.



Dialog zur Projektdokumentation

Ausgedruckt werden die im Dialog selektierten Bereiche, die blau unterlegt sind.

Wenn Sie das gesamte Projekt selektieren wollen, selektieren Sie den Namen Ihres Projektes in der ersten Zeile. Wollen Sie dagegen nur ein einzelnes Objekt selektieren, klicken Sie auf das entsprechende Objekt bzw. bewegen das gepunktete Rechteck mit den Pfeiltasten auf das gewünschte Objekt. Objekte, die vor ihrem Symbol ein Pluszeichen haben, sind Organisationsobjekte, die weitere Objekte beinhalten. Mit einem Klick auf ein Pluszeichen wird das Organisationsobjekt aufgeklappt und mit einem Klick auf das nun erscheinende Minuszeichen kann es wieder zugeklappt werden. Wenn Sie ein Organisationsobjekt selektieren, sind alle zugehörigen Objekte ebenfalls selektiert. Bei gedrückter <Umschalt>-Taste können Sie einen Bereich von Objekten auswählen, bei gedrückter <Strg>-Taste, mehrere einzelne Objekte.

Wenn Sie Ihre Auswahl getroffen haben, klicken Sie auf OK. Es erscheint der Dialog zum Drucken. Das Layout der zu druckenden Seiten können Sie mit 'Datei**"Einstellungen Dokumentation'** festlegen.

'Projekt' 'Exportieren'

TwinCAT PLC Control bietet die Möglichkeit, Projekte zu exportieren, bzw. importieren. Damit haben Sie die Möglichkeit, Programme zwischen verschiedenen IEC-Programmiersystemen auszutauschen.

Bisher gibt es ein standardisiertes Austauschformat für Bausteine in AWL, ST und AS (das Common Elements-Format der IEC 61131-3). Für die Bausteine in KOP und FUP und die anderen Objekte hat TwinCAT PLC Control ein eigenes Ablageformat, da es hierfür kein textuelles Format in der IEC 61131-3 gibt.

Die ausgewählten Objekte werden in eine ASCII-Datei geschrieben.

Es können Bausteine, Datentypen und die Ressourcen exportiert werden. Zusätzlich können die Einträge im Bibliotheksverwalter, d.h. die Verknüpfungsinformation zu den Bibliotheken mit exportiert werden (nicht die Bibliotheken selbst !)

HINWEIS

Der Wieder-Import eines exportierten FUP- oder KOP-Bausteins schlägt fehl, wenn im graphischen Editor ein Kommentar ein einfaches Hochkomma (') enthält, da dieses als String-Beginn interpretiert wird.

Wenn Sie Ihre Auswahl im Dialogfenster getroffen haben (die Auswahl erfolgt wie bei **'Projekt' 'Dokumentieren'** beschrieben), können Sie noch entscheiden, ob Sie die Auswahl in eine Datei exportieren wollen, oder für jedes Objekt eine eigene Exportdatei generieren lassen. Schalten Sie hierzu entsprechend die Option **Eine Datei je Objekt** aus oder ein und klicken Sie dann auf **OK**.. Es erscheint der Dialog zum Speichern von Dateien. Geben Sie einen Dateinamen mit Zusatz ".exp" bzw. ein Verzeichnis für die einzelnen Objekt-Exportdateien an, die dann unter "Objektname.exp" dort angelegt werden.

'Projekt' 'Importieren'

Wählen Sie in dem erscheinenden Dialog zum Öffnen von Dateien die gewünschte Export-Datei aus.
Die Daten werden in das aktuelle Projekt importiert. Wenn ein gleichnamiges Objekt im Projekt bereits besteht, dann erscheint eine Dialogbox mit der Frage "Wollen Sie es ersetzen?": Antworten Sie mit **Ja**, wird das Objekt im Projekt ersetzt durch das Objekt aus der Importdatei, antworten Sie mit **Nein** erhält der Name des neuen Objekts als Ergänzung einen Unterstrich und eine Zählnummer ("_0", "_1", ..). Mit **Ja, alle** bzw. **Nein, alle** wird dies für alle Objekte durchgeführt.

Wird die Information zur Verknüpfung mit einer Bibliothek importiert, so wird die Bibliothek geladen und im Bibliotheksverwalter am Ende der Liste hinzugefügt. Wurde die Bibliothek bereits im Projekt geladen, so wird sie nicht erneut geladen. Ist allerdings in der Exportdatei, die importiert wird, ein anderer Speicherzeitpunkt für die Bibliothek angegeben, so wird der Bibliotheksnamen im Bibliotheksverwalter mit einem "*" gekennzeichnet (z.B. standard.lib*30.3.99 11:30:14), analog zum Laden eines Projektes. Kann die Bibliothek nicht gefunden werden, so erscheint der Informationsdialog : "Kann Bibliothek {<Pfad>\}<name> <date> <time>" nicht finden", analog zum Laden eines Projektes.
Im Meldungsfenster wird der Import protokolliert.

'Projekt' 'Kopieren'

Mit diesem Befehl können Sie Objekte (Bausteine, Datentypen und Ressourcen) sowie Verknüpfungen zu Bibliotheken aus anderen Projekten in Ihr Projekt kopieren. Wenn der Befehl gegeben wurde, dann öffnet zunächst der Standarddialog zum Öffnen von Dateien, wenn sie dort eine Datei ausgewählt haben, dann öffnet ein Dialog, in dem Sie die gewünschten Objekte auswählen können. Die Auswahl erfolgt wie bei **'Projekt' 'Dokumentieren'** beschrieben. Wenn ein gleichnamiges Objekt im Projekt bereits besteht, dann erhält der Name des neuen Objekts als letztes Zeichen einen Unterstrich und eine Zahlnummer ("_1", "_2" ...).

'Projekt' 'Vergleichen'

Dieser Befehl wird verwendet um zwei Projekte zu vergleichen, oder die aktuelle Version des geöffneten Projekts mit der, die zuletzt gespeichert worden war.

Übersicht:

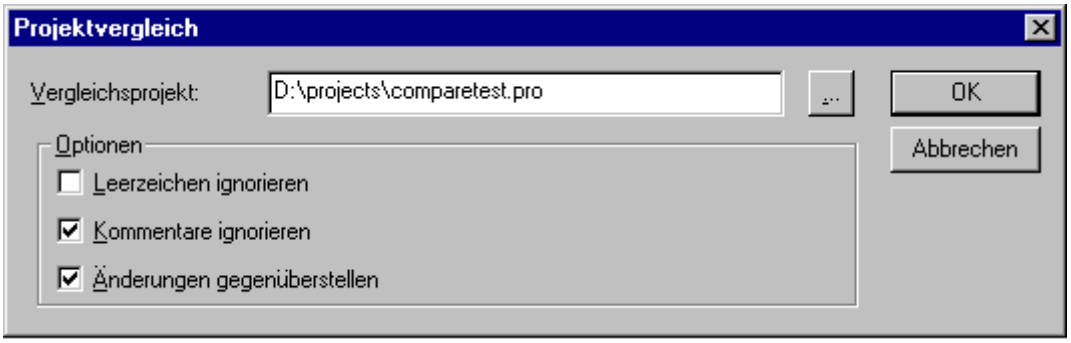
Im folgenden wird die Bezeichnung **'aktuelles Projekt'** für das Projekt verwendet, das momentan in Bearbeitung ist, und **'Vergleichsprojekt'** für das, das zum Vergleich aufgerufen wird. Nach Anwahl des Befehls wird das Projekt im **Vergleichsmodus** dargestellt. Mit **'Einheit'** wird im folgenden die kleinste Vergleichseinheit bezeichnet, die aus einer Zeile (Deklarationseditor, ST, AWL), einem Netzwerk (FUP, KOP) oder einem Element/Baustein (CFC,SFC) bestehen kann.

Im Vergleichsmodus werden aktuelles und Vergleichsprojekt in einem zweigeteilten Fenster gegenübergestellt und die als unterschiedlich befundenen Bausteine farblich gekennzeichnet. Bei Editorbausteinen gibt es auch für die Inhalte die direkte Gegenüberstellung. Vor dem Vergleichslauf können Filter bezüglich der Berücksichtigung von Leerzeichen und Kommentaren aktiviert werden. Außerdem kann gewählt werden, ob im Vergleichsmodus Veränderungen innerhalb bestehen bleibender Einheiten als solche dargestellt werden oder ob alle unterschiedlichen Einheiten als 'neu eingefügt' bzw. 'nicht mehr vorhanden' markiert werden. Die Version des Vergleichsprojekts kann für einzelne unterschiedliche Einheiten oder für einen ganzen Block gleich markierter ins aktuelle Projekt übernommen werden.

Solange der Vergleichsmodus aktiviert ist (siehe Statusleiste: COMPARE), kann das Projekt nicht editiert werden !

Durchführung Projektvergleich:

Nach Anwahl des Befehls öffnet der Dialog 'Projektvergleich':



Geben Sie den Pfad des **Vergleichsprojekts** ein. Über die Schaltfläche gelangen Sie zum Standarddialog für das Öffnen einer Datei, den Sie zur Projektauswahl zu Hilfe nehmen können. Wenn der Name des aktuellen Projekts eingetragen ist, wird die momentane Fassung des Projekts mit der bei der letzten Speicherung verglichen.

Falls das Projekt in einer ENI-Datenbank verwaltet wird, können Sie die lokal geöffnete mit der aktuellen Datenbankversion vergleichen. Aktivieren Sie hierzu die Option Vergleichen mit ENI-Projekt.

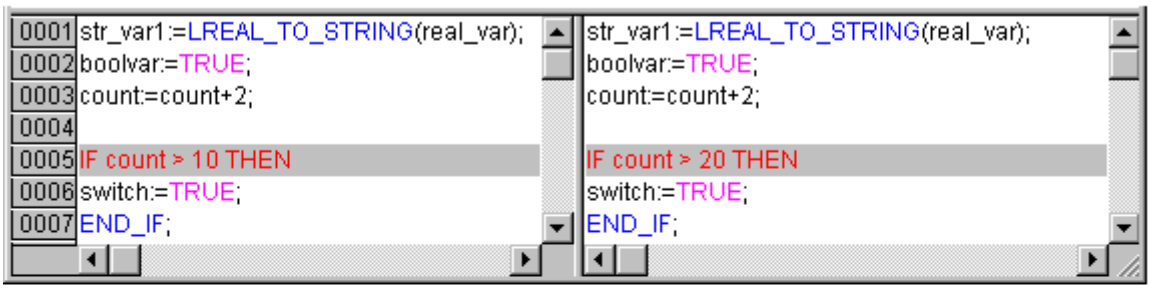
Folgende **Optionen** bezüglich des Vergleichs können aktiviert/deaktiviert werden:

- **Leerzeichen ignorieren**: Es werden keine Unterschiede gemeldet, die in unterschiedlicher Anzahl von Leerzeichen bestehen.
- **Kommentare ignorieren**: Es werden keine Unterschiede gemeldet, die Kommentare betreffen.
- **Änderungen gegenüberstellen**: Wenn die Option aktiviert ist: Für eine Einheit innerhalb eines Bausteins, die nicht gelöscht oder neu hinzugefügt, sondern nur verändert wurde, wird im zweigeteilten Fenster des Vergleichsmodus die Version des Vergleichsprojekts direkt der des aktuellen Projekts gegenübergestellt (rot markiert, siehe unten). Wenn die Option deaktiviert ist: Die betreffende Einheit wird im Vergleichsprojekt als 'nicht mehr vorhanden' und im aktuellen Projekt als 'neu eingefügt' dargestellt (siehe unten), also nicht direkt gegenübergestellt.

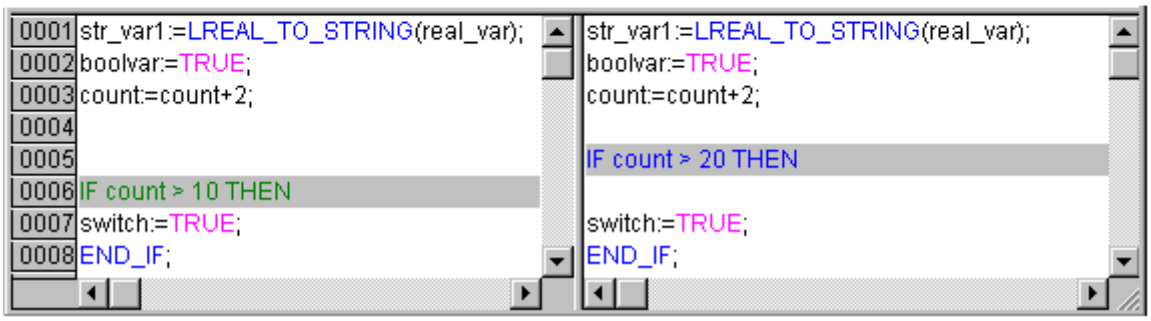
Beispiel:

Zeile 0005 wurde im aktuellen Projekt verändert (linke Fensterhälfte).

Option 'Oppose differences' activated:



Option 'Oppose differences' not activated:



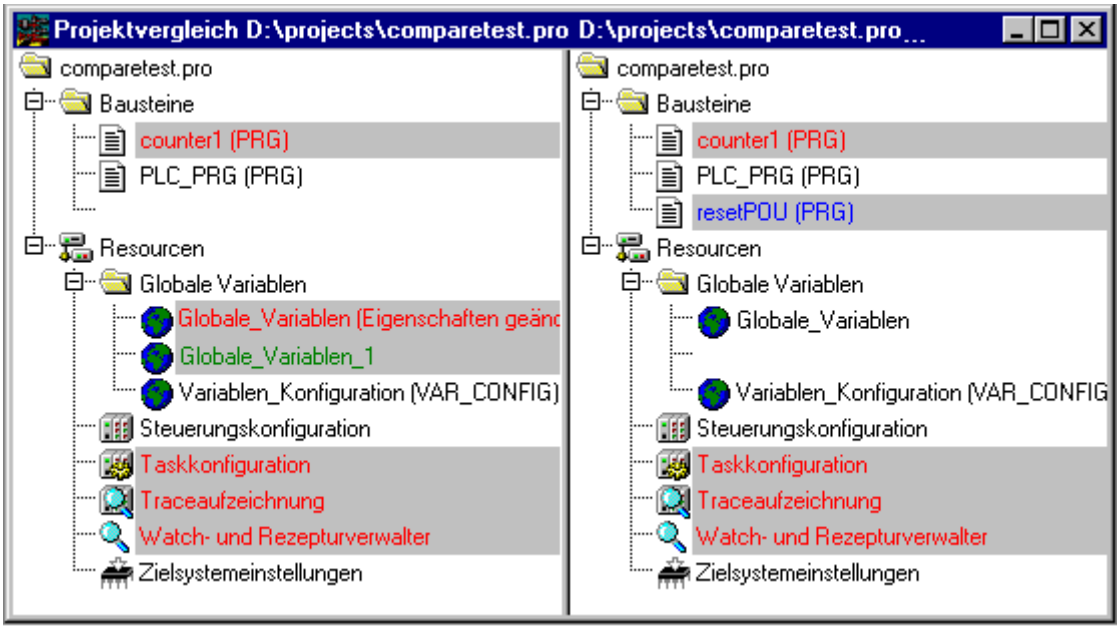
Wenn der Dialog Projektvergleich mit **OK** geschlossen wird, wird der Vergleich gemäß den Einstellungen durchgeführt.

Darstellung des Vergleichsergebnisses:

Die Ergebnisse werden zunächst im Strukturbaum des Projektes (Projektübersicht) dargestellt, von dem aus dann einzelne Bausteine geöffnet werden können, um deren inhaltliche Unterschiede zu sehen.

1. Projektübersicht im Vergleichsmodus:

Nach dem durchgeführten Projektvergleich öffnet ein zweigeteiltes Fenster, das den Strukturbaum des Projekts im **Vergleichsmodus** darstellt. In der Titelleiste steht: "Projektvergleich <Pfad aktuelles Projekt> - <Pfad Vergleichsprojekt>". Die linke Fensterhälfte zeigt das aktuelle Projekt, die rechte das Vergleichsprojekt. Die Projektübersicht zeigt an oberster Stelle den Projektnamen und entspricht ansonsten der Struktur des Object Organizers:



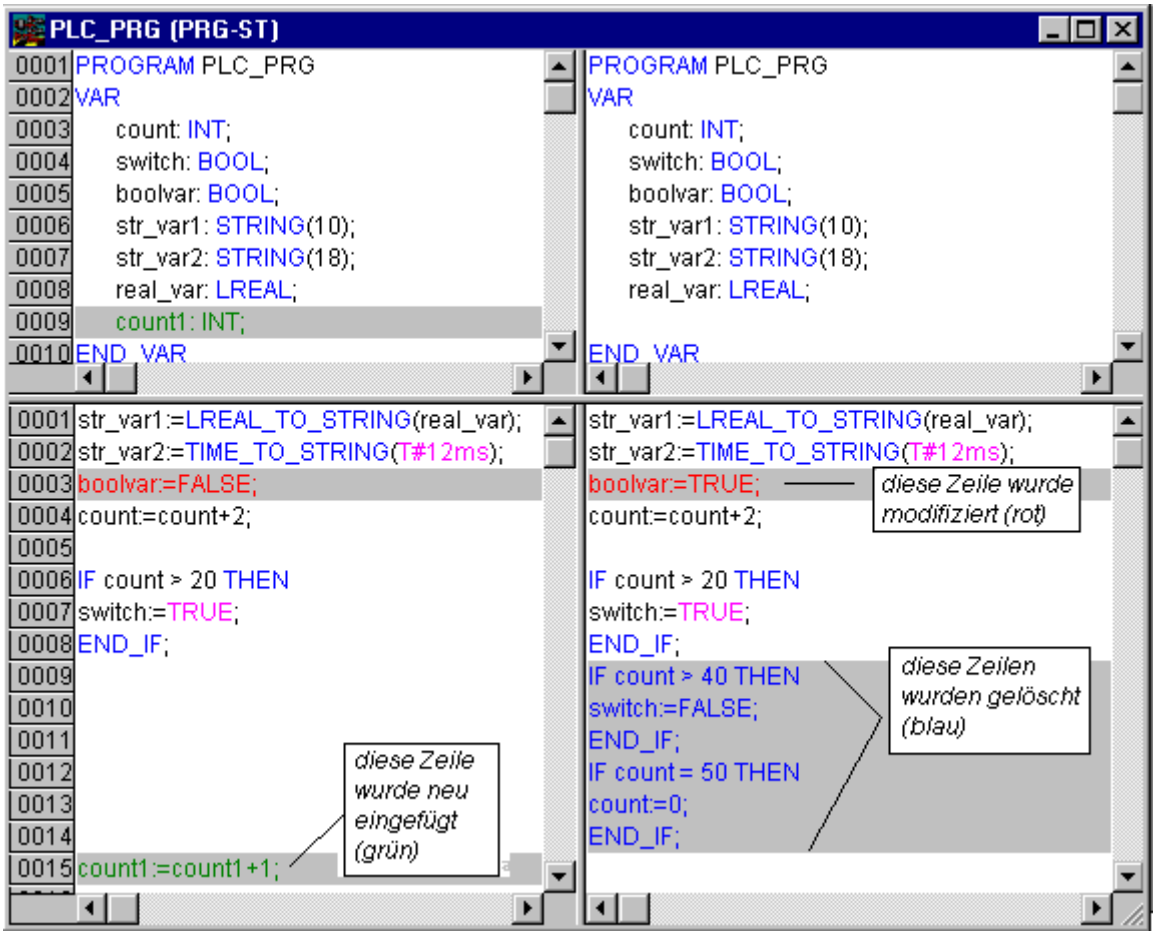
Bausteine, die Unterschiede aufweisen, werden mit einer Schattierung hinterlegt und durch die Textfarbe bzw. einen Textzusatz gekennzeichnet:

- **Rot**: Einheit wurde modifiziert; wird in beiden Fensterhälften rot dargestellt.
- **Blau**: Einheit ist nur im Vergleichsprojekt vorhanden; an gegenüberliegender Stelle in Strukturbaum des aktuellen Projekts wird eine Lücke eingefügt.
- **Grün**: Einheit ist nur im aktuellen Projekt vorhanden; an gegenüberliegender Stelle in Strukturbaum des Vergleichsprojekts wird eine Lücke eingefügt.
- **Schwarz**: Einheit, für die keine Unterschiede festgestellt wurde.
- "**(Eigenschaften geändert)**": Dieser Text erscheint hinter dem Bausteinnamen im Strukturbaum des aktuellen Projekts, wenn Unterschiede in den Bausteineigenschaften gefunden wurden.
- "**(Zugriffsrechte geändert)**": Dieser Text erscheint hinter dem Bausteinnamen im Strukturbaum des aktuellen Projekts, wenn Unterschiede in den Zugriffsrechten gefunden wurden.

2. Bausteininhalt im Vergleichsmodus:

Durch einen Doppelklick auf eine Zeile in der Projektübersicht, wird der betroffene **Baustein geöffnet**.

Handelt es sich um einen modifizierten (rot) Text- oder Grafikeditorbaustein, dann wird er in einem zweigeteilten Fenster geöffnet. Der Bausteininhalt des Vergleichsprojekts (rechts) wird wie in der Projektübersicht dem des aktuellen Projekts (links) gegenübergestellt. Für die unterschiedlichen Einheiten werden die bereits oben beschriebenen farblichen Kennzeichnungen angewendet.



Handelt es sich nicht um einen Editorbaustein, sondern beispielsweise um Taskkonfiguration, Zielsystemeinstellungen etc., wird, je nachdem ob der Doppelklick in der rechten oder linken Fensterhälfte der Projektübersicht angewendet wird, der Baustein des Vergleichsprojekts oder der des aktuellen Projekts in einem eigenen Fenster geöffnet. Für diese Projektbausteine erfolgt keine weitere inhaltliche Differenzierung der Unterschiede.

Arbeiten im Vergleichsmodus (Menü 'Extras', Kontextmenü):

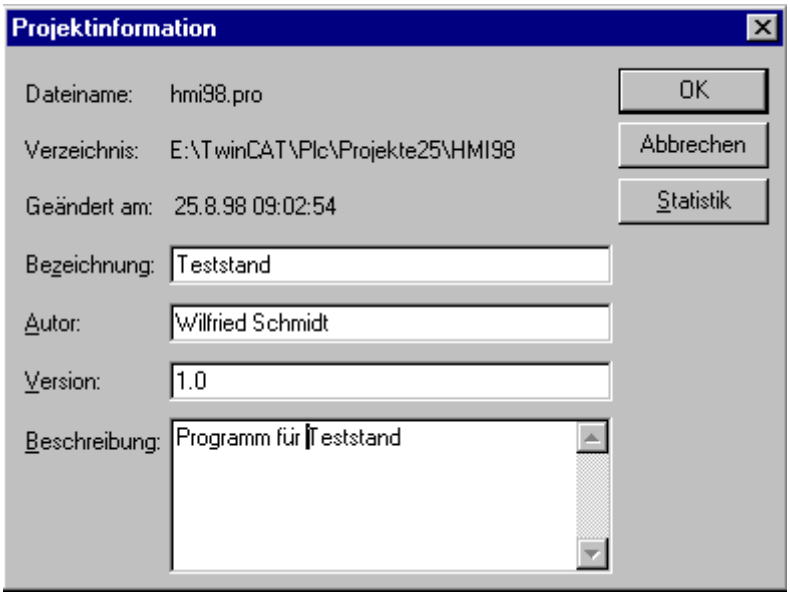
Steht der Cursor im zweigeteilten Vergleichsfenster auf einer Zeile, die eine Verschiedenheit anzeigt, bietet das Menü **Extras** bzw. das **Kontextmenü** (rechte Maustaste) je nachdem ob man sich in der Projektübersicht oder innerhalb eines Bausteins befindet, eine Auswahl der folgenden Befehle:

- **Nächster Unterschied (<F7>)**: Der Cursor springt zur nächsten Stelle (Zeile in der Projektübersicht/ Zeile bzw. Netzwerk im Baustein), die eine Verschiedenheit anzeigt.
- **Vorheriger Unterschied (<Umschalt><F7>)**: Der Cursor springt zur vorhergehenden Stelle (Zeile in der Projektübersicht/Zeile bzw. Netzwerk im Baustein), die eine Verschiedenheit anzeigt.
- **Änderung übernehmen (<Leertaste>)**: Für alle zusammenhängenden Einheiten (z.B. aufeinanderfolgende Zeilen), die die gleiche Änderungsmarkierung erhalten haben, wird die Version des Vergleichsprojekts ins aktuelle Projekt übernommen. Die betreffenden Einheiten erscheinen daraufhin in der entsprechenden Farbe in der linken Fensterhälfte. Handelt es sich um eine Einheit, die rot markiert war (Modifikation innerhalb), wird die Übernahme durch gelbe Schrift im aktuellen Projekt kenntlich gemacht.
- **Einzelne Änderung übernehmen (<Strg> <Leertaste>)**: Nur für die Vergleichseinheit, auf der aktuell der Cursor steht (z.B. Zeile in der Projektübersicht oder Zeile bzw. Netzwerk im Baustein), wird die Version des Vergleichsprojekts ins aktuelle Projekt übernommen. Die betreffenden Einheit erscheint daraufhin in der entsprechenden Farbe in der linken Fensterhälfte. Handelt es sich um eine Einheit, die rot markiert war (Modifikation innerhalb), wird die Übernahme durch gelbe Schrift im aktuellen Projekt kenntlich gemacht.
- **Eigenschaften übernehmen** (nur in der Projektübersicht): Für den Baustein, auf dem aktuell der Cursor steht, werden die Bausteineigenschaften aus dem Vergleichsprojekts ins aktuelle Projekt übernommen.

- **Zugriffsrechte übernehmen** (nur in der Projektübersicht): Für den Baustein, auf dem aktuell der Cursor steht, werden die Zugriffsrechte aus dem Vergleichsprojekts ins aktuelle Projekt übernommen.

'Projekt' 'Projektinformation'

Unter diesem Menüpunkt können Sie Informationen zu Ihrem Projekt abspeichern. Wenn der Befehl gegeben wurde, öffnet folgender Dialog:



Dialog zum Eingeben der Projektinformation

Als Projektinformation werden folgende Angaben angezeigt:

- Dateiname
- Verzeichnispfad
- Der Zeitpunkt der letzten Änderung (Geändert am)

Diese Angaben können nicht geändert werden. Darüber hinaus können Sie noch folgende eigene Angaben hinzufügen:

- eine Bezeichnung des Projekts,
- der Name des Autors,
- die Versionsnummer und
- eine Beschreibung des Projekts.

Diese Angaben sind optional. Bei Drücken der Schaltfläche Statistik erhalten Sie eine statistische Auskunft über das Projekt. Diese enthält die Angaben aus der Projektinformation, sowie die Anzahl der Bausteine, Datentypen, der lokalen und globalen Variablen, wie sie bei der letzten Übersetzung aufgezeichnet wurden.

Beispiel einer Projektstatistik

Wenn sie die Option **Projektinformation verlangen** in der Kategorie **Laden & Speichern** im Optionsdialog wählen, dann wird beim Abspeichern eines neuen Projekts, oder beim Abspeichern eines Projekts unter einem neuen Namen automatisch die Projektinformation aufgerufen.

'Projekt' 'Global Suchen'

Mit diesem Befehl können Sie nach dem Vorkommen eines Textes in Bausteinen, in Datentypen in den Objekten der globalen Variablen, in der Steuerungskonfiguration, in der Taskkonfiguration und in den Deklarationsteilen der Bibliotheken suchen. Wenn der Befehl eingegeben wurde, öffnet sich ein Dialog, in dem Sie die Bausteine und Objekte auswählen können, die durchsucht werden sollen. Die Auswahl erfolgt wie bei 'Projekt' 'Dokumentieren' beschrieben.

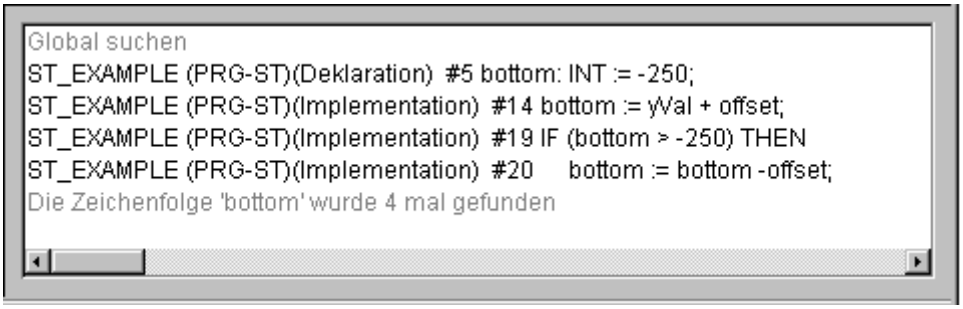
Wird die Auswahl mit **OK** bestätigt, erscheint der Standarddialog zum Suchen.

Dieser erscheint direkt, wenn der Befehl **'Global Suchen'** über das Symbol in der Menüleiste aufgerufen wurde, die Suche bezieht sich dann automatisch auf alle durchsuchbaren Teile des Projekts. Die zuletzt eingegebenen Suchstrings können über die Combobox des Feldes **Suche nach** ausgewählt werden. Wird ein Text in einem Objekt gefunden, so wird das Objekt in den zugehörigen Editor bzw. in den Bibliotheksverwalter geladen und die Fundstelle angezeigt. Das Anzeigen des gefundenen Textes sowie das Suchen und Weitersuchen verhalten sich analog zum Befehl **'Bearbeiten' 'Suchen'**.

Wenn Sie die Schaltfläche **Ins Meldungsfenster** anwählen, werden alle Verwendungsstellen der gesuchten Zeichenfolge in den ausgewählten Objekten zeilenweise in tabellarischer Form im Meldungsfenster aufgelistet. Abschließend wird die Anzahl der gefundenen Stellen angegeben.

Falls das Meldungsfenster nicht geöffnet war, wird es eingeblendet. Pro gefundener Stelle wird folgendes ausgegeben:

- Objektname
- Fundstelle im Deklarationsteil (Decl) oder im Implementationsteil (Impl) eines Bausteins
- Zeilen- bzw. Netzwerknummern
- komplette Zeile bei den textuellen Editoren
- komplette Texteinheit bei den grafischen Editoren



Wenn Sie im Meldungsfenster mit der Maus einen Doppelklick auf eine Zeile ausführen oder die <Eingabetaste> drücken, öffnet der Editor mit dem Objekt. Die betroffene Zeile des Objekts wird markiert. Mit den Funktionstasten <F4> und <Umschalt>+<F4> kann schnell zwischen den Ausgabezeilen gesprungen werden.

'Projekt' 'Global Ersetzen'

Mit diesem Befehl können Sie nach dem Vorkommen eines Textes in Bausteinen, Datentypen oder den Objekten der globalen Variablen suchen und diesen Text durch einen anderen Ersetzen. Bedienung und Ablauf verhalten sich ansonsten wie 'Projekt' 'Global Suchen' bzw. 'Bearbeiten' 'Ersetzen'. Allerdings werden die Bibliotheken nicht zur Auswahl angeboten und es ist keine Ausgabe ins Meldungsfenster möglich.

'Projekt' 'Überprüfen'

Mit diesem Befehl öffnen Sie ein Untermenü mit den folgenden Befehlen zur Überprüfung der semantischen Korrektheit des Projekts:

- Unbenutzte Variablen
- Überlappende Speicherbereiche
- Konkurrierender Zugriff
- Mehrfaches Speichern auf Output

Die Ergebnisse werden im Meldungsfenster ausgegeben.
Jede dieser Funktionen prüft den Stand des letzten Übersetzungslaufs. Das Projekt muss also mindestens einmal fehlerfrei übersetzt worden sein, bevor die Überprüfung durchgeführt werden kann, andernfalls sind die Menüpunkte "gegraut".

Unbenutzte Variablen

Diese Funktion des Menüs 'Projekt' 'Überprüfen' sucht nach Variablen, die deklariert sind, aber im Programm nicht verwendet werden. Sie werden mit Bausteinname und -zeile ausgegeben, z.B.: PLC_PRG (4) - var1. Variablen in Bibliotheken werden nicht berücksichtigt.

Die Ergebnisse werden im Meldungsfenster ausgegeben.

Überlappende Speicherbereiche

Diese Funktion Menüs 'Projekt' 'Überprüfen' prüft, ob bei der Zuweisung von Variablen mittels "AT"-Deklaration auf bestimmte Speicherbereiche Überschneidungen entstehen. Beispielsweise entsteht durch die Zuweisung der Variablen "var1 AT %QB21: INT" und "var2 AT %QD5: DWORD" eine Überschneidung, da sie das Byte 21 gemeinsam belegen. Die Ausgabe sieht dann folgendermaßen aus:

```
%QB21 wird durch folgende Variablen
referenziert:

PLC_PRG (3): var1 AT %QB21

PLC_PRG (7): var2 AT %QD5
```

Die Ergebnisse werden im Meldungsfenster ausgegeben.

Konkurrierender Zugriff

Diese Funktion Menüs 'Projekt' 'Überprüfen' sucht nach Speicherbereichen, die in mehr als einer Task referenziert werden. Zwischen lesendem oder schreibendem Zugriff wird dabei nicht unterschieden. Die Ausgabe lautet beispielsweise:

```
%MB28 wird in folgenden Tasks referenziert
:

Task1 - PLC_PRG (6): %MB28 [Nur-Lese-Zugriff]

Task2 - POU1.ACTION (1) %MB28 [Schreibzugriff]
```

Die Ergebnisse werden im Meldungsfenster ausgegeben.

Mehrfaches Speichern auf Output

Diese Funktion Menüs 'Projekt' 'Überprüfen' sucht nach Speicherbereichen, auf die in einem Projekt an mehr als einer Stelle schreibend zugegriffen wird. Die Ausgabe sieht beispielsweise so aus:

```
%QB24 wird an folgenden Stellen
beschrieben:

PLC_PRG (3): %QB24

PLC_PRG.POU1 (8): %QB24
```

Die Ergebnisse werden im Meldungsfenster ausgegeben.

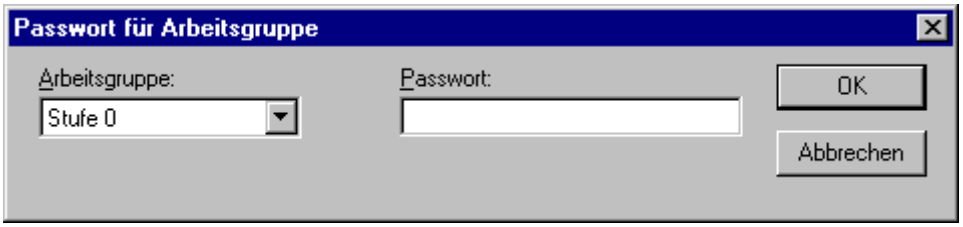
Arbeitsgruppen

Es können im TwinCAT PLC Control bis zu acht Gruppen mit unterschiedlichen Zugriffsrechten auf Bausteine, Datentypen und Ressourcen eingerichtet werden. Es können Zugriffsrechte für einzelne oder alle Objekte festgelegt werden. Jedes Öffnen eines Projekts geschieht als Mitglied einer bestimmten Arbeitsgruppe. Als solches Mitglied muss man sich mit einem Passwort autorisieren.

Die Arbeitsgruppen sind von 0 bis 7 durchnummeriert, wobei die Gruppe 0 die Administratorrechte besitzt, d.h. nur Mitglieder der Gruppe 0 dürfen Passwörter und Zugriffsrechte für alle Gruppen bzw. Objekte festlegen.

Wenn ein neues Projekt angelegt wird, dann sind zunächst alle Passwörter leer. Solange kein Passwort für die Gruppe 0 festgelegt wurde, betritt man das Projekt automatisch als Mitglied der Gruppe 0.

Wenn beim Laden des Projekts ein Passwort für die Arbeitsgruppe 0 festgestellt wird, dann wird beim Öffnen des Projekts für alle Gruppen die Eingabe eines Passworts verlangt. Dazu öffnet folgender Dialog:



Stellen Sie in der Combobox **Arbeitsgruppe** auf der linken Seite des Dialogs, die Gruppe ein, zu der Sie gehören, und geben Sie auf der rechten Seite das dazugehörige **Passwort** ein. Drücken Sie **OK**. Wenn das Passwort nicht mit dem gespeicherten Passwort übereinstimmt, kommt die Meldung:

"Das Kennwort ist nicht korrekt."

Erst wenn Sie das korrekte Kennwort eingegeben haben, wird das Projekt geöffnet.

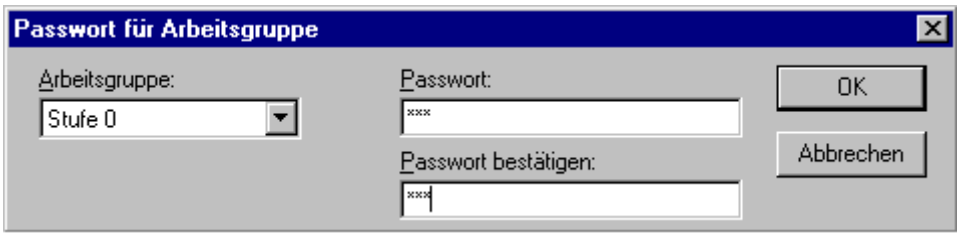
HINWEIS

Werden nicht für alle Arbeitsgruppen Passwörter vergeben, so kann man ein Projekt über eine Arbeitsgruppe, für die keines vergeben wurde, öffnen!

Mit dem Befehl **'Passwörter für Arbeitsgruppe'** können Sie Passwörter vergeben, mit dem Befehl 'Objekt' 'Zugriffsrechte' die Rechte für einzelne oder alle Objekte.

'Projekt' 'Passwörter für Arbeitsgruppen'

Mit diesem Befehl öffnet man den Dialog zur Passwortvergabe für Arbeitsgruppen. Dieser Befehl kann nur von Mitgliedern der Gruppe 0 ausgeführt werden. Wenn der Befehl gegeben wurde, öffnet folgender Dialog:



Dialog zur Passwortvergabe

In der linken Combobox Arbeitsgruppe können Sie die Gruppe auswählen. Für diese geben Sie das gewünschte Passwort im Feld Passwort ein. Für jeden getippten Buchstaben erscheint im Feld ein Stern (*). Dasselbe Wort müssen Sie im Feld Passwort bestätigen wiederholen. Schließen Sie den Dialog nach jeder Passworteingabe mit OK. Wenn die Meldung kommt:" Das Kennwort und seine Bestätigung stimmen nicht überein.", haben Sie sich bei einem der beiden Einträge vertippt. Wiederholen Sie deswegen am besten beide Einträge solange, bis der Dialog ohne Meldung schließt. Vergeben Sie gegebenenfalls erst danach ein Passwort für die nächste Gruppe durch erneuten Aufruf des Befehls.
Mit dem Befehl 'Objekt' 'Zugriffsrechte' können Sie die Rechte für einzelne oder alle Objekte vergeben

4.4 Objekte

Im Folgenden wird beschrieben, wie man mit Objekten arbeitet und welche Hilfen Ihnen zur Verfügung stehen den Überblick über ein Projekt zu behalten (Ordner, Aufrufbaum, Querverweisliste,..).

Objekt

Als "Objekt" werden Bausteine, Datentypen und die Ressourcen (globale Variablen, die Traceaufzeichnung, die Steuerungskonfiguration, die Taskkonfiguration und der Watch- und Rezepturverwalter) bezeichnet. Die zur Strukturierung des Projektes eingefügten Ordner sind zum Teil mit impliziert. Sämtliche Objekte eines Projekts stehen im Object Organizer.

Wenn Sie den Mauszeiger eine kurze Zeit über einen Baustein im Object Organizer halten, wird die Art des Bausteins (Programm, Funktion oder Funktionsblock) in einem Tooltip angezeigt; bei den globalen Variablen das Schlüsselwort (VAR_GLOBAL, VAR_CONFIG).

Mit Drag&Drop können Sie Objekte (und auch Ordner, siehe 'Ordner') innerhalb ihrer Objektart verschieben. Selektieren Sie hierzu das Objekt und verschieben es bei gedrückter linker Maustaste an den gewünschten Ort. Kommt es durch die Verschiebung zu einer Namenskollision, wird das neu eingefügt Element durch eine angehängte fortlaufende Nummer (z.B. "Objekt_1") eindeutig gekennzeichnet.

Ordner

Um den Überblick bei größeren Projekten zu behalten, sollte man seine Bausteine, Datentypen und globalen Variablen sinnvoll in Ordnern gruppieren.

Es ist eine beliebige Schachtelungstiefe von Ordnern möglich. Befindet sich vor dem geschlossenen Ordnersymbol ein Pluszeichen , beinhaltet dieser Ordner Objekte und/oder weitere Ordner. Mit einem Klick auf das Pluszeichen wird der Ordner aufgeklappt und die untergeordneten Objekte erscheinen. Mit einem Klick auf das nun voranstehende Minuszeichen kann er wieder zugeklappt werden.

Im Kontextmenü finden Sie die Befehle '**Knoten Expandieren**' und **'Knoten Kollabieren'** mit der gleichen Funktionalität.

Mit Drag&Drop können Sie die Objekte und auch die Ordner innerhalb ihrer Objektart verschieben. Selektieren Sie hierzu das Objekt und verschieben es bei gedrückter linker Maustaste an den gewünschten Ort. Weitere Ordner können Sie mit 'Neuer Ordner' einfügen.

Ordner haben keinerlei Einfluss auf das Programm, sondern dienen lediglich der übersichtlichen Strukturierung Ihres Projektes

Beispiel von Ordnern im Object Organizer

'Neuer Ordner'

Mit dem Befehl wird ein neuer Ordner als Ordnungsobjekt eingefügt. Ist ein Ordner selektiert, wird der neue unter diesen Ordner angelegt, ansonsten auf gleicher Ebene. Ist eine Aktion selektiert, wird der neue Ordner auf der Ebene des Bausteins eingefügt, zu dem die Aktion gehört. Das Kontextmenü des Object Organizers, das diesen Befehl beinhaltet, erscheint, wenn ein Objekt oder die Objektart selektiert ist und Sie die rechte Maustaste oder <Umschalt>+<F10> drücken.

Der neu eingefügte Ordner erhält zunächst die Bezeichnung 'Neuer Ordner'. Beachten Sie folgende Namenskonvention für Ordner:

- Ordner, die sich auf gleicher Hierarchieebene befinden, müssen unterschiedliche Namen tragen. Ordner auf unterschiedlichen Ebenen können gleiche Namen erhalten.
- Ein Ordner kann nicht den gleichen Namen erhalten wie ein Objekt, das sich auf derselben Ebene befindet.

Ist bereits ein Ordner mit Namen 'Neuer Ordner' auf gleicher Ebene vorhanden, erhält jeder zusätzliche mit diesem Namen automatisch eine angehängte fortlaufende Nummer (z.B. "Neuer Ordner 1"). Ein Umbenennen auf einen bereits verwendeten Namen ist nicht möglich.

'Knoten Expandieren' 'Knoten Kollabieren'

Mit dem Befehl Expandieren werden die Objekte sichtbar aufgeklappt, die sich unter dem gewählten Objekt befindet, mit Kollabieren werden die untergeordneten Objekte nicht mehr angezeigt. Bei Ordnern können Sie auch mit Doppelklick oder drücken der <Eingabetaste> die Ordner auf- und zuklappen. Das Kontextmenü des Object Organizers, das diesen Befehl beinhaltet, erscheint, wenn ein Objekt oder die Objektart selektiert ist und Sie die rechte Maustaste oder <Umschalt>+<F10> drücken.

'Projekt''Objekt löschen' Kurzform: <Entf>

Mit diesem Befehl wird das aktuell markierte Objekt (ein Baustein, eine Datentyp oder globale Variablen) oder ein Ordner mit den darunter liegenden Objekten aus dem Object Organizer entfernt, und ist somit im Projekt gelöscht. Zuvor wird zur Sicherheit noch einmal abgefragt. Wenn das Editorfenster des Objekts geöffnet war, wird es automatisch geschlossen. Verwendet man zum Löschen den Befehl 'Bearbeiten''Ausschneiden', wird das Objekt zusätzlich in die Zwischenablage geschoben.

Projekt''Objekt einfügen' Kurzform: <Einfg>

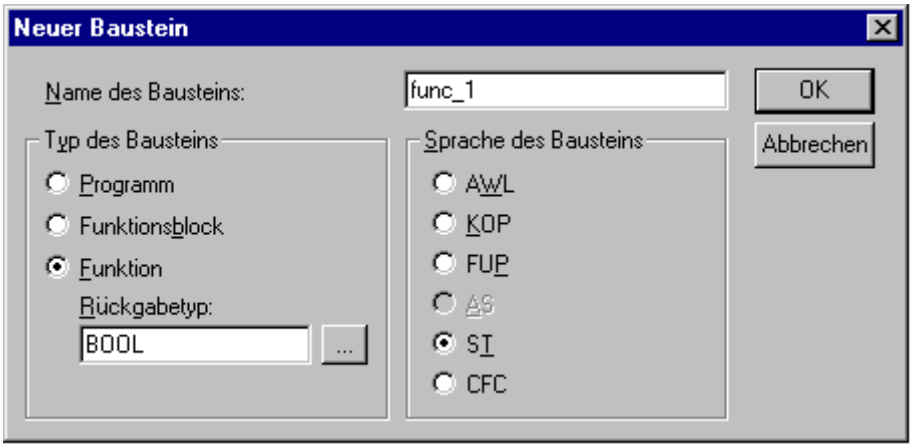
Mit diesem Befehl erstellen Sie ein neues Objekt. Die Art des Objekts (Baustein, Datentyp oder globale Variablen) hängt von der gewählten Registerkarte im Object Organizer ab. In der erscheinenden Dialogbox geben Sie den **Namen** des neuen Objektes an.

Beachten Sie hierbei folgende Einschränkungen:

- Der Bausteinname darf keine Leerzeichen enthalten
- Ein Baustein darf nicht den gleichen Namen erhalten wie ein anderer Baustein, bzw. wie ein Datentyp
- Ein Datentyp darf nicht den gleichen Namen erhalten wie ein anderer Datentyp, bzw. wie ein Baustein.
- Eine globale Variablenliste darf nicht den gleichen Namen erhalten wie eine andere globale Variablenliste.
- Eine Aktion darf nicht den gleichen Namen erhalten wie eine andere Aktion desselben Bausteins.

In allen anderen Fällen sind Namensübereinstimmungen erlaubt. So können beispielsweise Aktionen verschiedener Bausteine gleiche Namen erhalten und eine Visualisierung den gleichen wie ein Baustein.

Handelt es sich um einen Baustein, muss zusätzlich der Typ des Bausteins (Programm, Funktion oder Funktionsblock) und die Sprache, in der er programmiert werden soll, gewählt werden. Als **Typ des Bausteins** ist 'Programm' voreingestellt, als **Sprache des Bausteins** die des zuletzt angelegten Bausteins. Soll ein Baustein des Typs Funktion erstellt werden, muss im Texteingabefeld Rückgabetyp der gewünschte Datentyp eingegeben werden. Dabei sind alle elementaren Datentypen und definierten Datentypen (Arrays, Strukturen, Enumerationen, Alias) erlaubt. Die Eingabehilfe (z.B. über **<F2>**) kann benützt werden



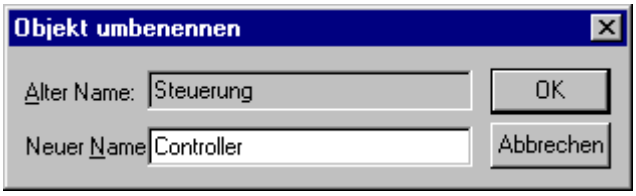
Nach dem Bestätigen der Eingabe über **OK**, was nur möglich ist, wenn nicht gegen oben genannte Namenskonventionen verstoßen wird, wird das neue Objekt im Object Organizer angelegt und das passende Eingabefenster erscheint.

Verwendet man den Befehl **'Bearbeiten' 'Einfügen'**, wird das in der Zwischenablage befindliche Objekt eingefügt und es erscheint kein Dialog. Verstößt der Name des eingefügten Objekts gegen die Namenskonventionen (siehe oben), wird er durch eine mit einem Unterstrich angehängte fortlaufende Nummer (z.B. "Rechtsabbieger_1") eindeutig gemacht.

Wenn das Projekt über die ENI Schnittstelle mit einer Projektdatenbank verknüpft ist, kann diese Verknüpfung so konfiguriert sein, dass beim Anlegen eines neuen Objekts nachgefragt wird, in welcher Datenbankkategorie es verwaltet werden soll. In diesem Fall erhalten Sie den Dialog 'Objekteigenschaften' zur Auswahl der Datenbankkategorie.

'Projekt''Objekt umbenennen' Kurzform: <Leertaste>

Mit diesem Befehl geben Sie dem aktuell ausgewählten Objekt oder Ordner einen neuen Namen. Beachten Sie, dass der Name des Objekts noch nicht verwendet worden sein darf. War das Bearbeitungsfenster des Objekts geöffnet ist, dann ändert sich sein Titel bei der Neubenennung des Objekts automatisch.



Dialog zum Umbenennen eines Bausteins

'Projekt' 'Objekt konvertieren'

Dieser Befehl kann nur auf Bausteine angewandt werden. Sie können Bausteine in den Sprachen ST, FUP, KOP und AWL in eine der drei Sprachen AWL, FUP und KOP übersetzen. Dazu muss das Projekt übersetzt sein. Wählen Sie die Sprache, in die Sie konvertieren wollen, und geben Sie dem neuen Baustein einen neuen Namen. Beachten Sie, dass der neue Name des Bausteins noch nicht verwendet worden sein darf. Dann können Sie OK drücken und der neue Baustein wird Ihrer Bausteinliste hinzugefügt.

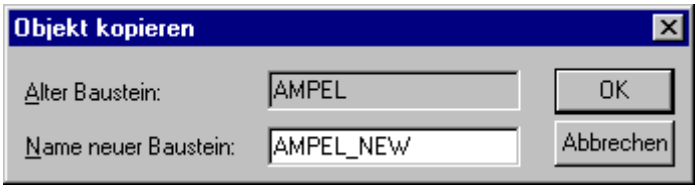
Die Art der Verarbeitung beim Konvertierungsvorgang entspricht der, die für einen Kompilierungslauf gilt.

Dialog zur Konvertierung eines Bausteins

'Projekt' 'Objekt kopieren'

Mit diesem Befehl wird ein ausgewähltes Objekt kopiert und unter neuem Namen abgespeichert. Geben Sie im erscheinenden Dialog den Namen des neuen Objektes ein. Beachten Sie, dass der Name des Objekts noch nicht verwendet worden sein darf.

Verwendet man dagegen den Befehl '**Bearbeiten' 'Kopieren**', wird das Objekt in die Zwischenablage kopiert und es erscheint kein Dialog.



Dialog zum Kopieren eines Bausteins

'Projekt' 'Objekt bearbeiten' Kurzform: <Eingabetaste>

Mit diesem Befehl laden Sie ein im Object Organizer markiertes Objekt in den jeweiligen Editor. Ist bereits ein Fenster mit diesem Objekt geöffnet, so erhält es den Fokus, d.h. es wird in den Vordergrund geholt, und kann nun bearbeitet werden. Es gibt noch zwei weitere Möglichkeiten, um ein Objekt zu bearbeiten:

- Doppelklick mit der Maus auf das gewünschte Objekt oder

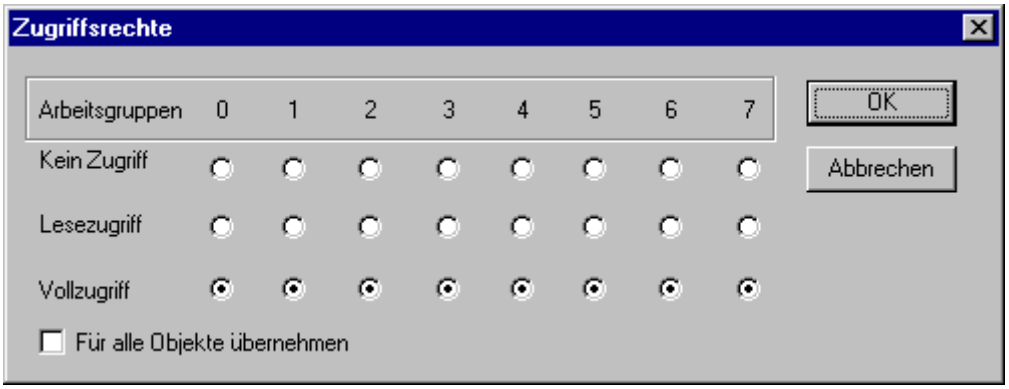
- Tippen Sie im Object Organizer die ersten Buchstaben des Objektnamens ein. Daraufhin öffnet sich ein Dialog, in dem alle Objekte der eingestellten Objektart mit diesen Anfangsbuchstaben zur Auswahl stehen. Selektieren Sie das gewünschte Objekt und klicken auf die Schaltfläche Öffnen, um das Objekt in sein Bearbeitungsfenster zu laden. Dabei wird dieses Objekt auch im Object Organizer markiert und alle im Objektpfad hierarchisch oberhalb des Objekts liegenden Ordner und Objekte werden expandiert. Diese Möglichkeit wird bei der Objektart Ressourcen nur für globale Variablen unterstützt.

Die letzte Möglichkeit ist vor allem in Projekten mit vielen Objekten hilfreich.

Dialog zur Wahl des zu öffnenden Objektes

'Projekt' 'Objekt Zugriffsrechte'

Mit diesem Befehl öffnet man den Dialog zur Vergabe der Zugriffsrechte der verschiedenen Arbeitsgruppen. Es öffnet folgender Dialog:



Dialog zur Vergabe von Zugriffsrechten

Mitglieder der Arbeitsgruppe 0 können nun für jede Arbeitsgruppe individuell Zugriffsrechte vergeben. Dabei sind drei Einstellungen möglich:

- **Kein Zugriff**: Das Objekt kann nicht von einem Mitglied der Arbeitsgruppe geöffnet werden.
- **Lesezugriff**: Das Objekt kann von einem Mitglied der Arbeitsgruppe zum Lesen geöffnet, aber nicht geändert werden.
- **Vollzugriff**: Das Objekt kann von einem Mitglied der Arbeitsgruppe geöffnet und geändert werden.

Die Einstellungen beziehen sich entweder auf das im Object Organizer aktuell markierte Objekt, oder falls die Option 'Für alle Objekte übernehmen' gewählt wird, auf sämtliche Bausteine, Datentypen und Ressourcen des Projekts. Die Zuordnung zu einer Arbeitsgruppe erfolgt beim Öffnen des Projektes durch eine Passwortabfrage, sofern ein Passwort für die Arbeitsgruppe 0 vergeben wurde.

'Projekt' 'Objekt Eigenschaften'

Für das im Object Organizer markierte Objekt öffnet dieser Befehl den Dialog 'Eigenschaften'.

Auf dem Registerblatt **Zugriffsrechte** findet sich derselbe Dialog, der auch beim Befehl 'Projekt' 'Objekt Zugriffsrechte' erhalten wird und der wie dort beschrieben bedient werden kann.

Ob weitere Registerblätter zur Einstellung von Objekteigenschaften verfügbar sind, und welche dies sind, hängt davon ab, welches Objekt und welche Projekteinstellungen vorliegen:

- Wenn eine globale Variablenliste im Objekt Organizer markiert ist, dann enthält das Registerblatt 'Globale Variablenliste' den Dialog **Globale Variablenliste**, in dem die Parameter für die Aktualisierung der Liste konfiguriert sind. Die Einträge können hier verändert werden. Bei der Neuerstellung einer globalen Variablenliste wird dieser Dialog mit dem Befehl 'Objekt einfügen' geöffnet, wenn im Objekt Organizer der Ordner 'Globale Variablen' bzw. einer der darunterstehenden Einträge markiert ist.
- Wenn das Projekt mit einer **Datenbank** verknüpft ist steht jeweils ein weiteres Registerblatt mit dem Titel Datenbank-Verknüpfung zur Verfügung. Hier wird die aktuelle Zuordnung des Objekts mit einer der Datenbankkategorien bzw. zur Kategorie 'Lokal' angezeigt und kann auch verändert werden.

'Projekt' 'Datenbankverknüpfung'

Dieser Befehl steht zur Verfügung, wenn in den Projektoptionen, Kategorie '**Projektdatenbank**' die Option '**Projektdatenbank (ENI) verwenden**' aktiviert ist. Er führt zu einem Untermenü mit Befehlen zur Verwaltung des Objekts bzw. Projekts in der aktuell über die ENI-Schnittstelle verknüpften Datenbank:

- Login: Anmelden des Benutzers beim ENI Server

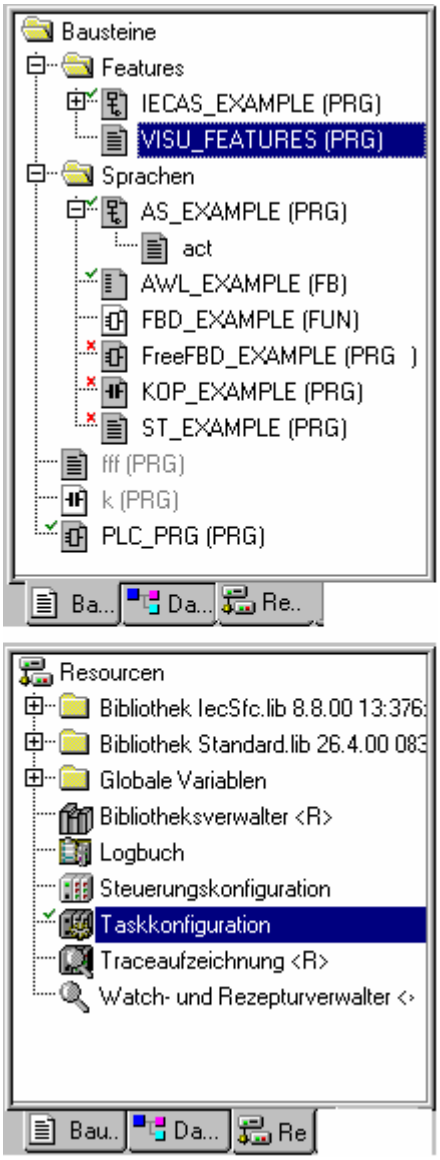
Wenn ein Objekt im Object Organizer markiert ist und der Befehl 'Datenbankverknüpfung' aus dem Kontextmenü (rechte Maustaste) gewählt wird, können für dieses Objekt über die folgenden Befehle die entsprechenden Datenbankfunktionen aufgerufen werden:

- Festlegen
- Abrufen
- Auschecken
- Einchecken
- Auschecken rückgängig
- Unterschiede anzeigen
- Versionsgeschichte anzeigen

Wenn der Befehl **'Datenbankverknüpfung'** im Menü **'Projekt'** angewählt wird, erscheinen zusätzliche Menüpunkte, die alle Objekte des Projekts betreffen:

- Mehrfach Festlegen
- Alles abrufen
- Mehrfach Auschecken
- Mehrfach Einchecken
- Mehrfach Auschecken rückgängig
- Projekt Versionsgeschichte
- Version Labeln
- Ressource-Objekte einfügen
- Status auffrischen

Die nachfolgenden Kapitel beschreiben die Befehle im einzelnen.



Grau schattiertes Icon:

Objekt wird in der Datenbank verwaltet.

Grüner Haken vor Objektnamen:

Objekt wurde vom aktuell geöffneten TwinCAT PLC Control Projekt aus ausgecheckt.

Rotes Kreuz vor Objektnamen:

Objekt ist momentan von einem anderen Benutzer ausgecheckt.

<R> hinter Objektnamen:

Auf das Objekt kann nur lesend zugegriffen werden.

Einige Objekte (Taskkonfiguration, Tracekonfiguration, Steuerungskonfiguration, Zielsystemeinstellungen, Watch- und Rezepturverwalter) sind grundsätzlich mit einem <R> versehen, solange sie nicht ausgecheckt sind. In diesem Fall bedeutet dies, dass keine automatische Abfrage '... Objekt auschecken..?.." erscheint, wenn mit dem Editieren des Objekts begonnen wird; es heißt jedoch nicht automatisch, dass kein Schreibzugriff möglich ist. Wenn kein Schreibzugriff möglich ist, erkennen Sie dies daran, dass der Befehl 'Auschecken' nicht anwählbar ist.

'Projekt' 'Datenbankverknüpfung' 'Festlegen'

Es wird festgelegt, ob das im Object Organizer markierte Objekt in der Datenbank oder nur lokal (im Projekt) verwaltet werden soll. Dazu erscheint ein Dialog, in dem eine der zwei Datenbankkategorien 'Projekt' und 'Gemeinsame Objekte' oder aber die Kategorie 'Lokal' gewählt werden kann.

Die Icons aller Objekte, die in der Datenbank verwaltet werden, erscheinen im Object Organizer grau schattiert.

'Projekt' 'Datenbankverknüpfung' 'Abrufen'

Die aktuelle Version des im Object Organizer markierten Objekts wird aus der Datenbank abgerufen und ersetzt die lokale Version. Im Gegensatz zum Auschecken, siehe unten, wird das Objekt in der Datenbank nicht für die Bearbeitung durch andere Benutzer gesperrt.

'Projekt' 'Datenbankverknüpfung' 'Auschecken'

Das im Object Organizer markierte Objekt wird aus der Datenbank ausgecheckt und dadurch für die Bearbeitung durch andere Benutzer gesperrt.

Beim Aufrufen des Befehls öffnet der Dialog '**Datei auschecken**'. Es kann ein Kommentar eingegeben werden, der in der Versionsgeschichte des Objekts in der Datenbank zusammen mit dem Auscheckvorgang gespeichert wird.

Nach Bestätigen des Dialogs mit OK wird das ausgecheckte Objekt im Object Organizer mit einem grünen Haken vor dem Bausteinnamen gekennzeichnet, für andere Benutzer erscheint es mit einem roten Kreuzchen markiert und ist damit für diese nicht bearbeitbar.

'Projekt' 'Datenbankverknüpfung' 'Einchecken'

Das im Object Organizer markierte Objekt wird in die Datenbank eingecheckt. Damit wird in der Datenbank eine neue Version des Objekts angelegt. Die alten Versionen bleiben erhalten.

Beim Aufrufen des Befehls öffnet der Dialog 'Datei einchecken'. Es kann ein Kommentar eingegeben werden, der in der Versionsgeschichte des Objekts in der Datenbank zusammen mit dem Auscheckvorgang gespeichert wird.

Nach Bestätigen des Dialogs mit OK verschwindet der grüne Haken vor dem Bausteinnamen im Object Organizer.

'Projekt' 'Datenbankverknüpfung' 'Auschecken rückgängig'

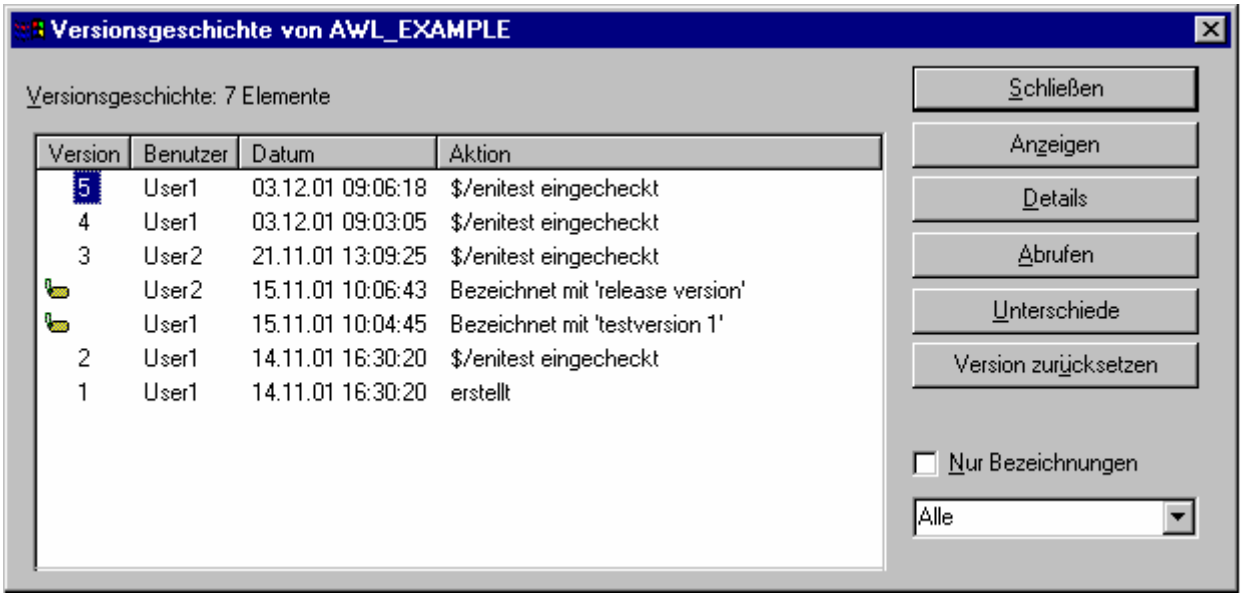
Das Auschecken des im Object Organizer markierten Objekts und die lokal in diesem Objekt vorgenommenen Änderungen werden rückgängig gemacht. Es erscheint kein Dialog. Das Objekt bleibt mit unveränderter Version und wieder für andere Bearbeiter freigegeben in der Datenbank. Der rote Haken vor dem Bausteinnamen im Object Organizer verschwindet.

'Projekt' 'Datenbankverknüpfung' 'Unterschiede anzeigen'

Der Baustein, der aktuell zur Bearbeitung geöffnet ist, wird in einem zweigeteilten Fenster dargestellt, das die lokale, bearbeitete Version der letzten aktuellen Version aus der Datenbank gegenüberstellt. Die Unterschiede der Versionen werden optisch wie beim Projektvergleich dargestellt.

'Projekt' 'Datenbankverknüpfung' 'Versionsgeschichte anzeigen'

Ein Dialog 'Versionsgeschichte von <Objektname> wird geöffnet, der in einer Tabelle alle Versionen, die für das aktuell bearbeitete Objekt in der Datenbank eingecheckt bzw. gelabelt wurden, auflistet. Angegeben sind:



Version: Datenbankabhängige Nummerierung der zeitlich nacheinander eingecheckten Versionen des Objekts. Gelabelte Versionen erhalten keine Versionsnummer, sondern sind mit einem Label-Icon gekennzeichnet.

Benutzer: Name des Benutzers, der die Aktion am Objekt durchgeführt hat

Datum: Datum und Uhrzeit der Aktion

Aktion: Art der Aktion, die am Objekt durchgeführt wurde. Datenbankabhängig, z.B. 'erstellt' (das Objekt wurde in der Datenbank erstmals eingecheckt), 'eingecheckt' oder 'bezeichnet mit <label>' (diese Version des Objekts wurde mit einem Bezeichner versehen)

Die Schaltflächen:

Schließen: Der Dialog wird geschlossen

Anzeigen: Die in der Tabelle markierte Version wird in einem Fenster geöffnet. In der Titelleiste steht "ENI: <Name des Projekts in der Datenbank>/<Objektname>

Details: Der Dialog **'Details der Versionsgeschichte'** öffnet:

Datei (Name des Projekts und des Objekts in der Datenbank), **Version** (s.o.), **Datum** (s.o.), **Benutzer** (s.o.), **Kommentar** (Kommentar, der beim Einchecken bzw. Labeln eingegeben wurde). Über die Schaltflächen **Nächste** bzw. **Vorherige** kann zu den Details des nächsten bzw. vorherigen Eintrags im Dialog 'Versionsgeschichte von ..' gesprungen werden

Abrufen: Die in der Tabelle markierte Version wird aus der Datenbank in das TwinCAT PLC Control geladen und ersetzt die lokale Version.

Unterschiede: Wird in der Tabelle nur eine Version des Objekts markiert bewirkt der Befehl, dass diese mit der aktuellen Datenbankversion verglichen wird. Sind zwei Versionen markiert, werden diese verglichen. Die Unterschiede werden in einem zweigeteilten Fenster wie beim Projektvergleich.

Version zurücksetzen: Die in der Tabelle markierte Version wird als aktuelle Datenbankversion gesetzt. Die später eingefügten Versionen werden gelöscht. Dies kann benützt werden, um einen früheren Stand wiederherzustellen und als aktuellen zu führen.

Nur Bezeichnungen: Wenn diese Option aktiviert ist, erscheinen nur die mit einem Label versehenen Versionen in der Tabelle zur Auswahl. In dem Auswahlfeld

Auswahlbox unterhalb der Optionswahl 'Nur Bezeichnungen': Hier sind die Namen aller Benutzer aufgelistet, die bereits Datenbankaktionen an den Objekten des Projekts durchgeführt haben. Wählen Sie 'Alle' oder einen der Namen, um für alle oder nur für die durch einen bestimmten Benutzer bearbeiteten Objekte die Versionsgeschichte zu erhalten.

'Projekt' 'Datenbankverknüpfung' 'Mehrfach Festlegen'

Mit diesem Befehl kann für mehrere Objekte des aktuellen Projekts gleichzeitig festgelegt werden, in welcher Datenbankkategorie sie verwaltet werden sollen. Es erscheint zunächst derselbe Dialog '**Objekteigenschaften**' wie beim Befehl '**Festlegen**'. Hier wählen Sie die gewünschte Kategorie und schließen den Dialog mit OK. Daraufhin öffnet sich der Dialog '**ENI-Auswahl**', der die Bausteine des Projekts auflistet, die für die eingestellte Kategorie in Frage kommen (beispielsweise erscheinen bei eingestellter Kategorie 'Ressourcen' nur die Ressourcen-Bausteine des Projekts zur Auswahl). Die Darstellung entspricht der im Object Organizer verwendeten Baumstruktur. Markieren Sie die gewünschten Bausteine und bestätigen mit OK.

'Projekt' 'Datenbankverknüpfung' 'Alles abrufen'

Für das geöffnete Projekt wird die aktuelle Version aller Objekte der Kategorie Projekt aus der Datenbank abgerufen. Wurden in der Datenbank Objekte hinzugefügt, werden diese nun ebenfalls lokal eingefügt, wurden in der Datenbank Objekte gelöscht, werden diese lokal nicht gelöscht, aber automatisch der Kategorie 'Lokal' zugeordnet. Bei Objekten der Kategorie Ressourcen werden nur diejenigen aus der Datenbank abgerufen, die bereits im lokalen Projekt angelegt sind. Zur Bedeutung des Abrufens siehe oben 'Abrufen'.

'Projekt' 'Datenbankverknüpfung' 'Mehrfach Auschecken'

Es können mehrere Objekte gleichzeitig ausgecheckt werden. Dazu öffnet sich der Dialog '**ENI-Auswahl**', der in einer wie im Object Organizer gestalteten Baumstruktur die Bausteine des Projekts auflistet. Markieren Sie die Bausteine, die ausgecheckt werden sollen, und bestätigen mit OK. Zur Bedeutung des Auscheckens siehe oben 'Auschecken'.

'Projekt' 'Datenbankverknüpfung' 'Mehrfach Einchecken'

Es können mehrere Objekte gleichzeitig ausgecheckt werden. Das Vorgehen entspricht dem beim Mehrfach Auschecken. Zur Bedeutung des Eincheckens siehe oben, 'Einchecken'.

'Projekt' 'Datenbankverknüpfung' 'Projekt Versionsgeschichte'

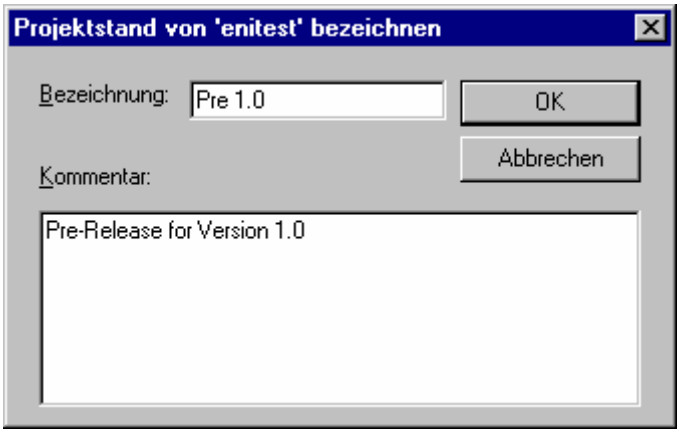
Wählen Sie diesen Befehl, um die Versionsgeschichte für das aktuelle Projekt einsehen zu können.

Sie erhalten den Dialog 'Versionsgeschichte von <Name des Projekts in der Datenbank>', in dem in chronologischer Folge die Aktionen (Erstellen, Einchecken, Labeln) für alle projektzugehörigen Objekte aufgelistet sind. Die Anzahl dieser Objekte wird hinter Versionsgeschichte: angegeben. Zur Bedienung des Dialogs sehen Sie bitte oben unter '**Versionsgeschichte**' für ein Einzelobjekt, beachten Sie jedoch folgende Unterschiede:

- Die Befehl 'Version zurücksetzen' steht nur für Einzelobjekte zur Verfügung
- Der Befehl 'Abrufen' bedeutet, dass alle Objekte aus der in der Tabelle markierten Projektversion ins lokale Projekt abgerufen werden. Dies bedeutet, dass lokale Objekte mit der älteren Version überschrieben werden. Lokale Objekte, die in dieser älteren Version noch nicht im Projekt enthalten waren, werden jedoch nicht aus der lokalen Version entfernt !

'Projekt' 'Datenbankverknüpfung' Projekt Version Labeln'

Dieser Befehl dient dazu, den aktuellen Stand der Objekte unter einer Bezeichnung zusammenzufassen, die es später erlaubt, genau diesen Stand wieder abzurufen. Es öffnet ein Dialog 'Projektstand von <Name des Projekts in der Datenbank>'. Geben eine **Bezeichnung** (Label) für den Projektstatus ein und optional einen **Kommentar**. Wenn Sie mit OK bestätigen, schließt der Dialog und die Bezeichnung und die Aktion des Labelns ("bezeichnet mit...") erscheinen in der Tabelle der Versionsgeschichte sowohl eines Einzelobjekts als auch der des Projekts. Eine gelabelte Version erhält keine Versionsnummer, sondern ist am Label-Icon in der Spalte 'Version' erkennbar. Ist die Option 'Nur Bezeichnungen' aktiviert, werden nur gelabelte Versionen angezeigt.

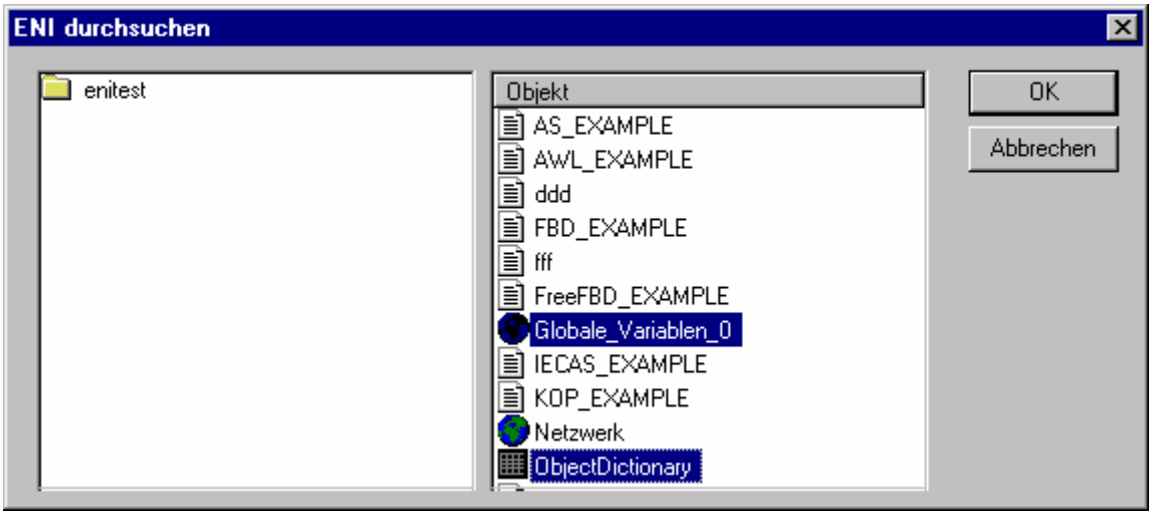


'Projekt' 'Datenbankverknüpfung' 'Gemeinsame-Objekte einfügen'

Dieser Befehl dient dazu, zusätzliche Objekte der Kategorie 'Gemeinsame Objekte', die in der Datenbank verfügbar sind, in das lokal geöffnete Projekt einzubinden. Bei Objekten der Kategorie Projekt ist dies nicht nötig, da beim 'Alles Abrufen' automatisch alle aktuell vorhandenen Datenbankobjekte ins lokale Projekt geladen werden, auch solche, die dort noch nicht angelegt sind. Bei Objekten der Kategorie 'Gemeinsam' jedoch werden beim 'Alles Abrufen' nur die bereits im Projekt eingebundenen Objekte berücksichtigt.

Fügen Sie ein zusätzliches Objekt folgendermaßen ein:

Der Befehl öffnet den Dialog '**ENI durchsuchen**', in dem alle Objekte aufgelistet werden, die in dem links angegebenen Projektverzeichnis in der Datenbankprojekt liegen. Wählen Sie die gewünschte Ressource und drücken Sie OK oder führen Sie eine Doppelklick darauf aus. Damit wird das Objekt in das lokal geöffnete Projekt eingefügt.



'Projekt' 'Datenbankverknüpfung' 'Status auffrischen'

Dieser Befehl aktualisiert die Anzeige im Object Organizer, so dass der aktuelle Status der Objekte bezüglich Datenbank dargestellt wird.

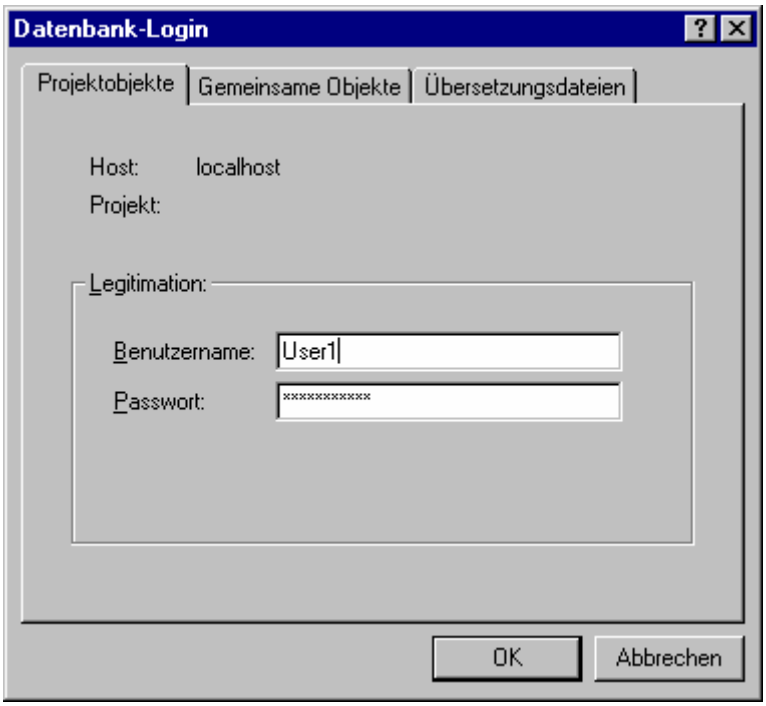
'Projekt' 'Datenbankverknüpfung' 'Login'

Dieser Befehl öffnet den Datenbank-Login-Dialog, in dem sich der Anwender beim ENI Server für jede Datenbankkategorie anmelden muss, um für das Projekt Verbindung zur jeweiligen Datenbank zu erhalten. Die Zugangsdaten müssen also im ENI Server (ENI Administration, Benutzerverwaltung) und gegebenenfalls auch in der Benutzerverwaltung der Datenbank bekannt sein. Nach Ausführen des Befehls öffnet zunächst der Login-Dialog für die Kategorie 'Projektobjekte'.

Angezeigt wird folgendes:

- **Datenbank**: Projektobjekte
- **Host**: Rechneradresse des ENI Servers (Host), wie sie auch in den Projektoptionen / Kategorie Projektdatenbank im Feld TCP/IP Adresse angegeben wird.
- **Projekt**: Name des Projekts in der Datenbank (siehe ebenfalls in Projektoptionen, Kategorie Projektdatenbank, Projektobjekte, Feld 'Projektname')

Geben Sie **Benutzername** und **Passwort** im Bereich **Legitimation** ein.



Drücken Sie OK, um die Eingaben zu bestätigen. Daraufhin schließt der Dialog für die Projektobjekte und es wird automatisch der Login-Dialog für die Gemeinsamen Objekte geöffnet. Geben Sie auch hier die entsprechenden Zugangsdaten ein, bestätigen mit OK und verfahren daraufhin genauso im dritten Login-Dialog, der sich für die Kategorie Übersetzungsdateien öffnet.

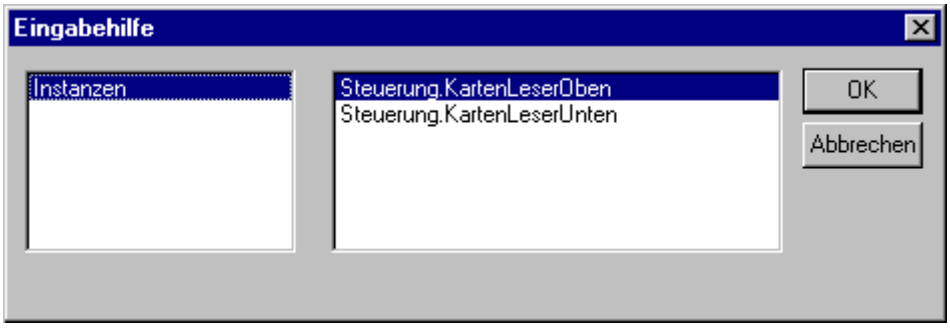
'Projekt' 'Aktion hinzufügen'

Mit diesem Befehl erzeugt man zum selektierten Baustein im Object Organizer eine Aktion. Im erscheinenden Dialog wählt man den Aktionsnamen und die Sprache aus, in der die Aktion implementiert werden soll.
Die neue Aktion wird im Object Organizer unter ihren Baustein gehängt. Vor dem Baustein erscheint nun ein Pluszeichen. Durch einen einfachen Mausklick auf das Pluszeichen erscheinen die Aktionsobjekte und vor dem Baustein ein Minuszeichen. Durch erneutes Klicken auf das Minuszeichen werden die Aktionen nicht mehr angezeigt und es erscheint wieder das Pluszeichen. Dies können Sie auch mit den Kontextmenübefehlen '**Knoten Expandieren**' und '**Knoten Kollabieren**' erreichen
Mit Doppelklick auf die Aktion oder durch Drücken <**Eingabetaste**> wird eine Aktion zum Editieren in ihren Editor geladen.

'Projekt' 'Instanz öffnen'

Mit diesem Befehl kann man im Online Modus die Instanz des im Object Organizer ausgewählten Funktionsblock öffnen und anzeigen lassen. Ebenso gelangt man durch einen Doppelklick auf den Funktionsblock im Object Organizer zu einem Auswahldialog, der die Instanzen des Funktionsblocks sowie die Implementation auflistet. Wählen Sie hier die gewünschte Instanz bzw. die Implementation und bestätigen Sie mit **OK**. Daraufhin wird das Gewünschte in einem Fenster dargestellt.

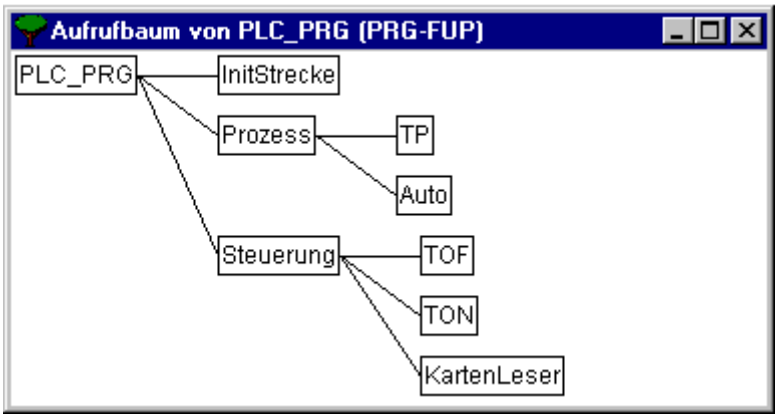
Instanzen können erst nach dem Einloggen geöffnet werden! (Projekt wurde korrekt übersetzt und über '**Online**' '**Einloggen**' an die Steuerung übertragen).



Dialog zum Öffnen einer Instanz

'Projekt''Aufrufbaum ausgeben'

Mit diesem Befehl öffnen Sie ein Fenster, in dem der Aufrufbaum des im Object Organizer ausgewählten Objekts dargestellt wird. Hierfür muss das Projekt übersetzt sein. Der Aufrufbaum beinhaltet sowohl Aufrufe von Bausteinen, als auch Verweise auf verwendete Datentypen.



Beispiel für einen Aufrufbaum

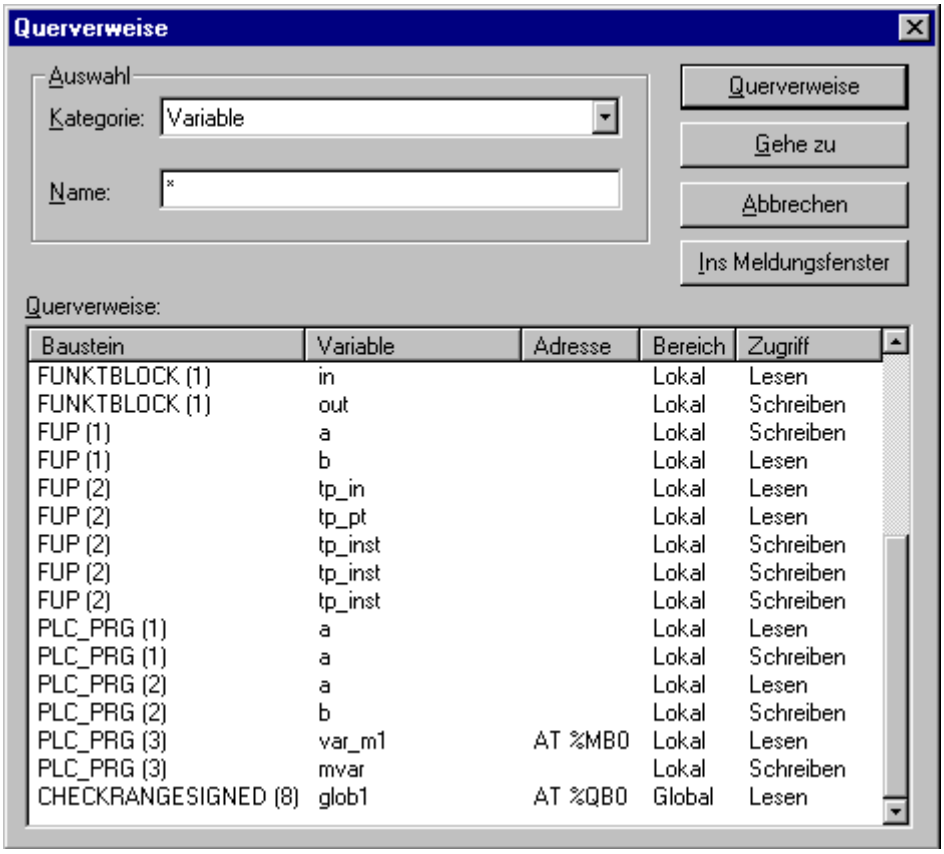
'Projekt' 'Querverweisliste ausgeben'

Mit diesem Befehl öffnen Sie einen Dialog, der die Ausgabe aller Verwendungsstellen zu einer Variable, einer Adresse oder einem Bausteins ermöglicht. Hierfür muss das Projekt übersetzt sein. Wählen Sie zuerst die Kategorie 'Variable', 'Adresse' oder 'Baustein' und geben Sie dann den Namen des gewünschten Elements ein. Um alle Elemente der eingestellten Kategorie zu erhalten, geben Sie bei Name ein "*" ein.

Mit Klicken auf die Schaltfläche **Querverweise** erhalten Sie die Liste aller Verwendungsstellen. Es wird neben dem Baustein und der Zeilen- bzw. Netzwerknummer der Variablenname und die eventuelle Adressanbindung angegeben. In der Spalte Bereich steht, ob es sich um eine lokale oder globale Variable handelt, in der Spalte Zugriff steht, ob an der jeweiligen Stelle über 'Lesen' oder 'Schreiben' auf die Variable zugegriffen wird.

Wenn Sie eine Zeile der Querverweisliste markieren und die Schaltfläche Gehe zu betätigen oder einen Doppelklick auf die Zeile ausführen, wird der Baustein in seinem Editor an der entsprechenden Stelle angezeigt. Auf diese Weise können Sie zu allen Verwendungsstellen ohne aufwendige Suche springen.

Um sich das Handling zu erleichtern, können Sie mit der Schaltfläche **Ins Meldungsfenster** die aktuelle Querverweisliste ins Meldungsfenster übernehmen und von dort zum jeweiligen Baustein wechseln



Dialog und Beispiel einer Querverweisliste

4.5 Editierfunktionen

Über die folgenden Befehle verfügen Sie in allen Editoren und zum Teil im Object Organizer. Die Befehle befinden sich sämtlich unter dem Menüpunkt 'Bearbeiten'.

'Bearbeiten' 'Rückgängig' Kurzform: <Strg>+<Z>

Dieser Befehl macht im aktuell geöffneten Editorfenster bzw. im Object Organizer die letzte ausgeführte Aktion rückgängig. Durch mehrfaches Ausführen dieses Befehls können alle Aktionen rückgängig gemacht werden bis zu dem Zeitpunkt, an dem das Fenster geöffnet wurde. Dies gilt für alle Aktionen in den Editoren für Bausteine, Datentypen und globalen Variablen und im Object Organizer.

Mit **'Bearbeiten''Wiederherstellen'** können Sie eine rückgängig gemachte Aktion erneut wieder ausführen.

'Bearbeiten' 'Wiederherstellen' Kurzform: <Strg>+<Y>

Mit dem Befehl können Sie eine rückgängig gemachte Aktion ('Bearbeiten' 'Rückgängig') im aktuell geöffneten Editorfenster bzw. im Object Organizer wiederherstellen.
So oft wie zuvor der Befehl 'Rückgängig' ausgeführt wurde, so oft kann auch 'Wiederherstellen' ausgeführt werden.

Die Befehle **'Rückgängig'** und **'Wiederherstellen'** beziehen sich jeweils auf das aktuelle Fenster. Jedes Fenster führt seine eigene Aktionsliste. Wenn Sie in mehreren Fenstern Aktionen rückgängig machen wollen, aktivieren Sie jeweils das entsprechende Fenster. Beim Rückgängigmachen oder Wiederherstellen im Object Organizer muss dort der Fokus liegen.

'Bearbeiten' 'Ausschneiden' Kurzform: <Strg>+<X> oder <Umschalt>+<Entf>

Dieser Befehl schiebt die aktuelle Markierung aus dem Editor in die Zwischenablage. Die Markierung wird aus dem Editor entfernt. Beim Object Organizer gilt dies analog mit dem markierten Objekt, wobei nicht alle Objekte gelöscht werden können, z.B. die Steuerungskonfiguration. Beachten Sie, dass nicht alle Editoren das Ausschneiden unterstützen, und dass es in einigen Editoren eingeschränkt sein kann. Die Form der Auswahl hängt vom jeweiligen Editor ab: In den Texteditoren (AWL, ST, Deklarationen) ist die Markierung eine Liste von Zeichen. Im FUP- und im KOP-Editor ist die Auswahl eine Menge von Netzwerken, die jeweils durch ein gepunktetes Rechteck im Netzwerk-Zahlenfeld markiert sind bzw. eine Box mit allen vorangehenden Linien, Boxen und Operanden. Im AS-Editor ist die Auswahl ein Teil einer Schrittfolge, umgeben von einem gepunkteten Rechteck.

Um den Inhalt der Zwischenablage einzufügen, benutzen Sie den Befehl **'Bearbeiten''Einfügen'**. Im AS-Editor können Sie ebenso die Befehle '**Extras''Parallelzweig einfügen (rechts)**' bzw. '**Extras''Einfügen danach'** benutzen.

Um eine Auswahl in die Zwischenablage einzufügen ohne sie zu entfernen, benutzen Sie den Befehl '**Bearbeiten''Kopieren'**.

Um einen markierten Bereich zu entfernen, ohne die Zwischenablage zu verändern, benutzen Sie den Befehl '**Bearbeiten''Löschen'**.

'Bearbeiten''Kopieren' Symbol: Kurzform: <Strg>+<C>

Dieser Befehl kopiert die aktuelle Markierung vom Editor in die Zwischenablage. Der Inhalt des Editorfensters wird dabei nicht verändert. Beim Object Organizer gilt dies analog mit dem markierten Objekt, wobei nicht alle Objekte kopiert werden können, z.B. die Steuerungskonfiguration. Beachten Sie, dass nicht alle Editoren das Kopieren unterstützen, und dass es in einigen Editoren eingeschränkt sein kann. Für die Form der Auswahl gelten dieselben Regeln wie bei '**Bearbeiten''Ausschneiden'**.

'Bearbeiten''Einfügen' Kurzform: <Strg>+<V>

Fügt den Inhalt der Zwischenablage an die aktuelle Position im Editorfenster ein. In den graphischen Editoren ist dieser Befehl nur dann ausführbar, wenn durch das Einfügen wieder eine korrekte Struktur entsteht. Beim Object Organizer wird das Objekt aus der Zwischenablage eingefügt. Beachten Sie, dass das Einfügen nicht von allen Editoren unterstützt wird, und das es in einigen Editoren eingeschränkt sein kann. Die aktuelle Position wird je nach Typ des Editors unterschiedlich definiert: Bei den Texteditoren (AWL, ST, Deklarationen) ist die aktuelle Position die Position, des blinkenden Cursors (eine kleine senkrechte Linie, die man per Mausklick positionieren kann). Im FUP- und im KOP-Editor ist die aktuelle Position das erste Netzwerk mit einem gepunkteten Rechteck im Netzwerknummernbereich. Der Inhalt der Zwischenablage wird vor diesem Netzwerk eingefügt. Wurde eine Teilstruktur kopiert, so wird diese vor dem markierten Element eingefügt. Im AS-Editor ist die aktuelle Position durch die Auswahl festgelegt, die mit einem gepunkteten Rechteck umgeben ist. Der Inhalt der Zwischenablage wird, abhängig von der Markierung und dem Inhalt der Zwischenablage, vor dieser Markierung, oder in einem neuen Zweig (parallel oder alternativ) links der Markierung eingefügt.

Im AS können auch die Befehle 'Extras**''Parallelzweig einfügen (rechts)'** bzw. **'Extras''Einfügen danach'** benutzt werden, um den Inhalt der Zwischenablage einzufügen.

'Bearbeiten''Löschen' Kurzform: <Entf>

Löscht den markierten Bereich aus dem Editorfenster. Der Inhalt der Zwischenablage wird dabei nicht verändert. Beim Object Organizer gilt dies analog mit dem markierten Objekt, wobei nicht alle Objekte ausgeschnitten werden können, z.B. die Steuerungskonfiguration. Für die Form der Auswahl gelten dieselben Regeln wie bei '**Bearbeiten''Ausschneiden'**. Im Bibliotheksverwalter ist die Auswahl der aktuelle gewählte Bibliotheksname.

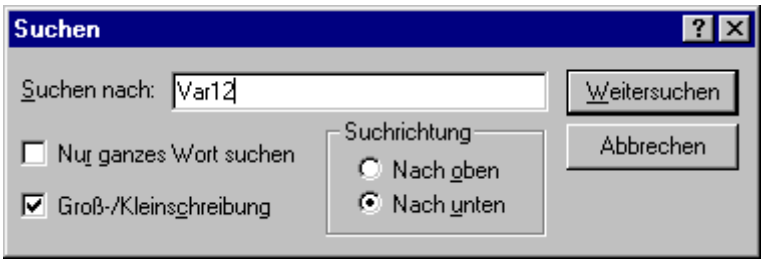
'Bearbeiten''Suchen'

Mit diesem Befehl suchen Sie nach einer bestimmten Textstelle im aktuellen Editorfenster. Es öffnet der Dialog für die Suche. Dieser bleibt geöffnet, bis die Schaltfläche **Abbrechen** gedrückt wird.

Im Feld **Suche nach** können Sie die zu suchende Zeichenfolge eingeben.

Außerdem können Sie auswählen, ob Sie den zu suchenden Text als **ganzes Wort** suchen wollen, oder auch als Teil eines Worts, ob bei der Suche die **Groß-/Kleinschreibung** beachtet werden soll und ob die Suche ausgehend von der aktuellen Cursorposition **nach oben** oder **nach unten** erfolgen soll. Die Schaltfläche **Weitersuchen** startet die Suche. Diese beginnt an der gewählten Position und erfolgt in der gewählten Suchrichtung. Wenn die Textstelle gefunden wurde, dann wird diese markiert. Wenn die Textstelle nicht gefunden wurde, wird das gemeldet. Die Suche kann mehrmals hintereinander durchgeführt werden bis der Anfang, bzw. das Ende des Inhalts des Editorfensters erreicht ist.

Beachten Sie, dass der gefundene Text vom Dialog zum Suchen verdeckt sein kann.



Dialog zum Suchen

'Bearbeiten' 'Weitersuchen' Kurzform: <F3>

Mit diesem Befehl führen Sie einen Suchbefehl mit denselben Parametern durch, wie bei der letzten Ausführung des Befehls **'Bearbeiten''Suchen'**.

'Bearbeiten' 'Ersetzen'

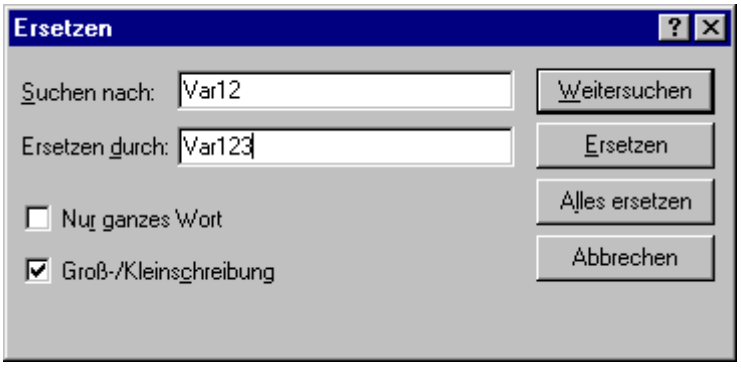
Mit diesem Befehl suchen Sie nach einer bestimmten Textstelle, genau wie beim Befehl **'Bearbeiten' 'Suchen'**, und ersetzen sie durch eine andere. Nachdem Sie den Befehl gewählt haben, öffnet der Dialog für 'Ersetzen'. Dieser Dialog bleibt so lang geöffnet, bis die Schaltfläche **Abbrechen** bzw. **Schließen** gedrückt wird.

Im Feld **Suchen nach** erscheint automatisch die Textstelle, die Sie vorher im Editor markiert haben, Sie können aber auch die zu suchende Zeichenfolge neu eingeben. Die Schaltfläche **Ersetzen** ersetzt dann das zuerst gefundene editierbare Vorkommen dieser Zeichenfolge durch den Text, der im Feld **Ersetzen durch** eingegeben wurde. Über Weitersuchen können Sie zur nächsten Stelle, an der die Zeichenfolge gefunden wird, gelangen. Mit der Schaltfläche **Alles ersetzen** wird die gesuchte Zeichenfolge im gesamten Projekt, sofern es sich um editierbare Stellen handelt, durch die gewünschte ersetzt.

Beachten Sie, dass an schreibgeschützten Textstellen der Text nicht ersetzt werden kann (Teile der Task- und Steuerungskonfiguration, Bibliotheken). Zeichenfolgen in editierbaren Teilen der Konfiguratoren (Task-, Programmname, Bezeichner für Ein-/Ausgänge) können ersetzt werden.

Die zuletzt eingegebenen Suchstrings und Ersetzungsstrings können über die jeweilige Combobox der Felder ausgewählt werden.

Nach dem Ersetzungsvorgang wird gemeldet, wie oft der Text ersetzt wurde.



Dialog zum Suchen und Ersetzen

'Bearbeiten' 'Eingabehilfe' Kurzform: <F2>

Mit diesem Befehl erhalten Sie einen Dialog zur Auswahl von möglichen Eingaben an der aktuellen Cursorposition im Editorfenster. Wählen Sie in der linken Spalte die gewünschte Kategorie der Eingabe aus, markieren in der rechten Spalte den gewünschten Eintrag und bestätigen Sie Ihre Wahl mit OK. Damit wird Ihre Auswahl an dieser Position eingefügt.

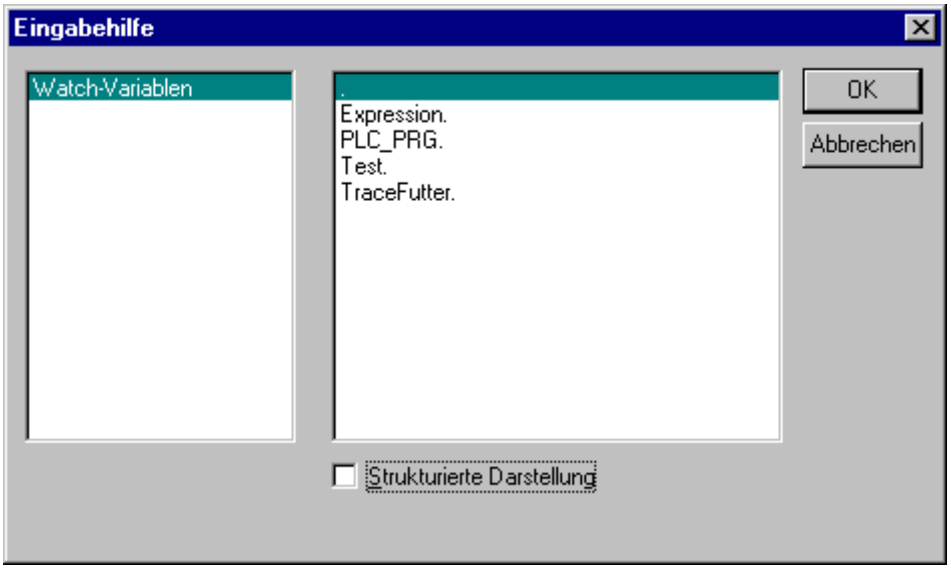
Die jeweils angebotenen Kategorien sind abhängig von der aktuellen Cursorposition im Editorfenster, d.h. davon was an dieser Stelle eingetragen werden kann (z.B. Variablen, Operatoren, Bausteine, Konvertierungen usw.).

Ist die Option **Mit Argumenten** aktiviert, werden beim Einfügen des gewählten Elements die zu übergebenden Argumente mit angegeben Beispiele: Auswahl von Funktionsblock fu1, der die Eingangsvariable var_in definiert hat: fu1(var_in:=);

Einfügen von Funktion func1, die als Übergabeparameter var1 und var2 braucht: func1(var1,var2);

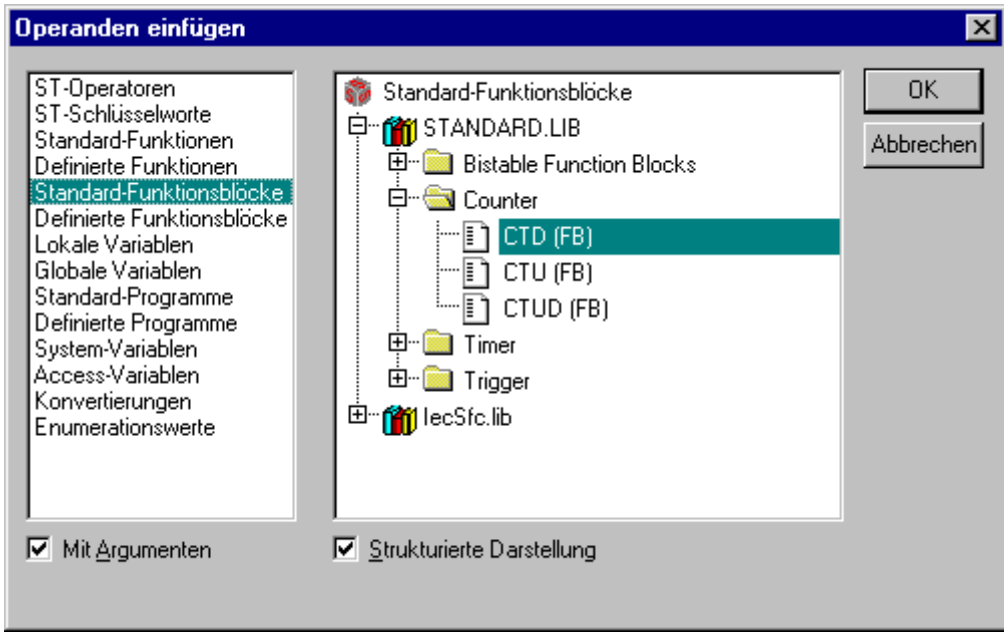
Grundsätzlich ist ein Wechsel zwischen nicht strukturierter und strukturierter Darstellung der zur Verfügung stehenden Elemente möglich. Dies geschieht durch Aktivieren/Deaktivieren der Option **Strukturierte Darstellung.**

- **Nicht-strukturierte Darstellung**



Die Bausteine, Variablen oder Datentypen in jeder Kategorie sind einfach linear alphabetisch sortiert. An manchen Positionen (z.B. in der Watchliste) werden mehrstufige Variablennamen benötigt. Dann zeigt der Dialog zur Eingabehilfe eine Liste aller Bausteine sowie einen einzelnen Punkt für die globalen Variablen. Nach jedem Bausteinnamen steht ein Punkt. Wird ein Baustein durch Doppelklick bzw. Drücken der <Eingabetaste> markiert, öffnet sich die Liste der zugehörigen Variablen. Liegen Instanzen und Datentypen vor, kann weiter aufgeklappt werden. Durch **OK** wird die selektierte Variable übernommen.

- **Strukturierte Darstellung**



Ist **Strukturierte Darstellung** angewählt, werden die Bausteine, Variablen oder Datentypen hierarchisch geordnet. Dies ist möglich für Standard-Programme, Standard-Funktionen, Standard-Funktionsblöcke, Definierte Programme, Definierte Funktionen, Definierte Funktionsblöcke, Globale Variablen, Lokale Variablen, Definierte Typen, Watch-Variablen. Die optische und hierarchische Darstellung entspricht der des Object Organizers, sind Elemente in Bibliotheken betroffen, werden diese in alphabetischer Reihenfolge an oberster Stelle eingefügt und die jeweilige Hierarchie wie im Bibliotheksverwalter dargestellt.

Die Ein- und Ausgangsvariablen von Funktionsblöcken, die als Lokale oder Globale Variablen deklariert sind, sind in der Kategorie 'Lokale Variablen' bzw. 'Globale Variablen' unterhalb des Instanznamens aufgelistet (z.B. Inst_TP.ET, Inst_TP.IN, ...). Man gelangt dorthin, indem man den Instanznamen .(z.B. Inst_TP) anwählt und mit **OK** bestätigt.

Wenn hier die Instanz eines Funktionsblocks markiert ist, kann die Option **Mit Argumenten** angewählt werden. Dann werden in den Textsprachen ST und AWL und bei der Taskkonfiguration der Instanzname und die Eingangsparameter des Funktionsblocks eingefügt.

z.B. bei Auswahl Inst (DeklarationInst : TON;) wird eingefügt:

```
Inst(IN:= ,PT:=)
```

Ist die Option nicht angewählt, wird nur der Instanzname eingefügt. Bei den grafischen Sprachen oder im Watch-Fenster wird generell nur der Instanzname eingefügt.

Komponenten von Strukturen werden analog zu Funktionsblockinstanzen dargestellt.

Für Enumerationen werden die einzelnen Enumerationswerte unter dem Enumerationstyp aufgelistet. Die Reihenfolge: Enums aus Bibliotheken, Enums aus Datentypen, lokale Enums aus Bausteinen.

Generell gilt, dass Zeilen, die Unterobjekte enthalten, nicht auswählbar sind (außer Instanzen, s.o.), sondern nur auf- und zuklappbar analog zu den mehrstufigen Variablennamen.

Wird die Eingabehilfe im Watch- und Rezepturverwalter bzw. bei der Auswahl der Trace-Variablen im Trace-Konfigurationsdialog aufgerufen, so ist es möglich, eine Mehrfachauswahl zu treffen. Bei gedrückter <Umschalt>-Taste können Sie einen Bereich von Variablen auswählen, bei gedrückter <Strg>-Taste, mehrere einzelne Variablen. Die gewählten Variablen werden markiert. Werden bei der Bereichsmarkierung Zeilen ausgewählt, die keine gültigen Variablen enthalten (z.B. Bausteinnamen), so werden diese Zeilen nicht in die Auswahl übernommen. Mit der Einzelauswahl sind solche Zeilen nicht markierbar.

Im Watchfenster und in der Tracekonfiguration ist es möglich, Strukturen, Arrays oder Instanzen aus der Eingabehilfe zu übernehmen. Da ein Doppelklick der Maustaste mit dem Auf- und Zuklappen des Elements belegt ist, kann die Auswahl in diesen Fällen nur durch **OK** bestätigt werden.

Daraufhin werden die gewählten Variablen zeilenweise ins Watchfenster eingetragen, d.h. jede gewählte Variable wird in eine Zeile geschrieben. Bei den Tracevariablen wird jede Variable in einer Zeile der Tracevariablen-Liste eingetragen.

Wird beim Eintragen der gewählten Variablen die maximale Anzahl der Tracevariablen von 20 überschritten, erscheint die Fehlermeldung "Es sind höchstens 20 Variablen erlaubt". Weitere ausgewählte Variablen werden dann nicht mehr in die Liste übernommen.

Einige Einträge (z.B. Globale Variablen) werden erst nach einem Übersetzungslauf in der Eingabehilfe aktualisiert.

'Bearbeiten' 'Variablen Deklaration' Kurzform: <Umschalt>+<F2>

Mit diesem Befehl erhalten Sie den Dialog zur Variablendeklaration, der sich bei eingestellter Projektoption 'Automatisch deklarieren' auch öffnet, sobald Sie im Deklarationseditor eine neue Variable eingeben.

'Bearbeiten''Nächster Fehler' Kurzform: <F4>

Nach dem fehlerhaften Übersetzen eines Projektes kann mit diesem Befehl der nächste Fehler bzw. die nächste Warnung angezeigt werden. Es wird jeweils das entsprechende Editorfenster aktiviert und die fehlerhafte Stelle markiert und gleichzeitig im Meldungsfenster die passende Fehlermeldung bzw. Warnung unterlegt. Sollen die Warnungen beim schrittweisen Durchgehen mit F4 ignoriert werden, muss die Option 'F4 ignoriert Warnungen' im Menü **'Projekt' 'Optionen' 'Arbeitsbereich'** aktiviert sein.

'Bearbeiten' 'Vorheriger Fehler' Kurzform: <Umschalt>+<F4>

Nach dem fehlerhaften Übersetzen eines Projektes kann mit diesem Befehl die vorherige Fehlermeldung bzw. Warnung angezeigt werden. Es wird jeweils das entsprechende Editorfenster aktiviert und die fehlerhafte Stelle markiert und gleichzeitig im Meldungsfenster die passende Fehlermeldung bzw. Warnung unterlegt. Sollen die Warnungen beim schrittweisen Durchgehen mit F4 ignoriert werden, muss die Option 'F4 ignoriert Warnungen' im Menü **'Projekt' 'Optionen' 'Arbeitsbereich'** aktiviert sein

'Bearbeiten' 'Makros'

Unter diesem Menüpunkt erscheinen alle Makros, die für das aktuelle Projekt definiert wurden. (Zur Erstellung siehe 'Projekt' 'Optionen' 'Makros'). Wird das gewünschte Makro angewählt und ist es ausführbar, öffnet sich der Dialog 'Makro ausführen'. Hier erscheinen der Makroname und die aktuelle Befehlszeile. Über die Schaltfläche **Abbrechen** kann die Abarbeitung des Makros gestoppt werden, wobei die aktuelle Befehlszeile noch zu Ende abgearbeitet wird. Dabei wird eine entsprechende Meldung ins Message-Fenster und im online-Betrieb in das Logbuch ausgegeben: "<Makro>: Ausführung durch Anwender abgebrochen".

Makros können sowohl offline als auch online ausgeführt werden. Es werden jedoch jeweils nur die im jeweiligen Mode verfügbaren Befehle ausgeführt

4.6 Onlinefunktionen

Die zur Verfügung stehenden Onlinebefehle sind unter dem Menüpunkt 'Online' gesammelt. Die Ausführung einiger Befehle ist abhängig vom aktiven Editor. Die Onlinebefehle stehen erst nach dem Einloggen zur Verfügung.

Durch die Funktionalität 'Online Change' haben Sie die Möglichkeit, Änderungen des Programms auf der laufenden Steuerung vorzunehmen. Sehen Sie hierzu **'Online' 'Einloggen'.**

'Online''Einloggen' Kurzform: <F11>

Dieser Befehl verbindet das Programmiersystem mit der Steuerung (oder startet das Simulationsprogramm) und wechselt in den Online Modus.

Wenn das aktuelle Projekt seit dem Öffnen bzw. seit der letzten Veränderung nicht übersetzt wurde, so wird es jetzt übersetzt (wie bei 'Projekt' 'Übersetzen'). Treten beim Übersetzen Fehler auf, so wechselt das TwinCAT PLC Control nicht in den Online Modus.

Wenn das aktuelle Projekt seit dem letzten Download auf die Steuerung verändert, aber nicht geschlossen wurde, und wenn nicht mit dem Befehl **'Projekt' 'Alles bereinigen'** die letzten Download-Informationen gelöscht wurden, wird nach dem Befehl 'Einloggen' ein Dialog mit der Abfrage geöffnet: "Das Programm wurde geändert. Sollen die Änderungen geladen werden? (Online Change)". Mit **Ja** bestätigen Sie, dass beim Einloggen die geänderten Teile des Projekts auf die Steuerung geladen werden sollen. Mit **Nein** erfolgt ein Einloggen, ohne dass die seit dem letzten Download vorgenommenen Änderungen auf die Steuerung geladen werden. Mit **Abbrechen** brechen Sie den Befehl ab. Mit **Alles laden** wird das komplette Projekt erneut auf die Steuerung geladen.

Nach erfolgreichem Einloggen stehen alle Onlinefunktionen zur Verfügung (soweit die entsprechenden Einstellungen in **'Optionen'** Kategorie Übersetzungsoptionen eingegeben wurden).

Um vom Online Modus zurück in den Offline Modus zu wechseln, benutzen Sie den Befehl **'Online''Ausloggen'**

Im Fehlerfall:

Fehler: "Es konnte keine Verbindung zur Steuerung aufgenommen werden"

Überprüfen Sie, ob Sie Ihr TwinCAT gestartet haben (TwinCAT Icon grün). Sollte das TwinCAT Icon in der Taskbar (unten) rot sein, so müssen Sie das TwinCAT System über rechte Maustaste auf das TwinCAT Icon, dann 'System' und 'Start' starten.

Fehler: "Das Programm wurde geändert! Soll das neue Programm geladen werden?"

Das aktuelle Projekt im Editor passt nicht zu dem derzeit in der Steuerung geladenen Programm (bzw. zu dem gestarteten Simulationsprogramm). Monitoring und Debugging ist deshalb nicht möglich. Sie können nun "Nein" wählen, sich wieder ausloggen und das richtige Projekt öffnen oder mit "Ja" das aktuelle Projekt in die Steuerung laden.

Meldung: "Das Programm wurde geändert! Sollen die Änderungen geladen werden? (ONLINE CHANGE)"

Das Projekt läuft auf der Steuerung. Das Zielsystem unterstützt 'Online Change' und das Projekt ist gegenüber dem letzten Download bzw. dem letzten Online Change auf die Steuerung verändert worden. Sie können nun entscheiden, ob diese Änderungen bei laufendem Steuerungsprogramm geladen werden sollen oder ob der Befehl abgebrochen werden soll. Sie können aber auch den gesamten übersetzten Code laden, indem Sie die Schaltfläche Alles laden wählen.

'Online''Ausloggen' Kurzform: <F12>

Die Verbindung zur Steuerung wird abgebaut und in den Offline Modus gewechselt. Um in den Online Modus zu wechseln, benutzen Sie den Befehl '**Online''Einloggen**'.

'Online''Laden'

Dieser Befehl lädt das kompilierte Projekt in die Steuerung (Download, nicht zu verwechseln mit **'Online' 'Quellcode laden'**!).

Die Download-Informationen werden in einer Datei <projectname>0000000ar.ri gespeichert, die beim Online Change verwendet wird, um das aktuelle Programm mit dem zuletzt in die Steuerung geladenen zu vergleichen, so dass nur die geänderten Programmteile erneut geladen werden. Diese Datei wird mit dem Befehl **'Projekt' 'Alles bereinigen'** gelöscht !

'Online''Start' Kurzform: <F5>

Dieser Befehl startet die Abarbeitung des Anwenderprogramms in der Steuerung bzw. in der Simulation. Der Befehl kann ausgeführt werden, unmittelbar nach dem Befehl **'Online''Laden'** oder nachdem das Anwenderprogramm in der Steuerung mit dem Befehl **'Online''Stop'** gestoppt wurde oder wenn das Anwenderprogramm auf einem Breakpoint steht oder wenn **'Online''Einzelzyklus'** ausgeführt wurde.

'Online''Stop' Kurzform <Umschalt>+<F8>

Stoppt die Abarbeitung des Anwenderprogramms in der Steuerung bzw. in der Simulation zwischen zwei Zyklen. Benutzen Sie den Befehl **'Online"Start'**, um die Programmabarbeitung fortzusetzen.

'Online' 'Reset'

Dieser Befehl setzt mit Ausnahme der Persistent-Variablen (VAR PERSISTENT) alle Variablen auf den Wert zurück, mit dem sie initialisiert wurden (also auch die mit VAR RETAIN deklarierten Variablen !). Variablen, die nicht explizit mit einem Initialisierungswert versehen wurden, werden auf die Standardinitialwerte gesetzt (Integer-Zahlen beispielsweise auf 0). Bevor alle Variablen überschrieben werden, erfolgt eine Sicherheitsabfrage durch das TwinCAT PLC Control. Die Situation entspricht der bei einem Stromausfall oder beim Aus-/Einschalten der Steuerung während das Programm läuft. Benutzen Sie den Befehl '**Online' 'Start**', um die Steuerung und damit die Programmabarbeitung erneut zu starten.

'Online' 'Urlöschen'

Dieser Befehl setzt alle Variablen, auch die persistenten (PERSISTENT) auf den Initialisierungswert zurück und löscht das Anwenderprogramm auf der Steuerung. Die Steuerung wird in den Urzustand zurückversetzt.

'Online' 'Breakpoint an/aus' Kurzform: <F9>

Dieser Befehl setzt einen Breakpoint an der aktuellen Position im aktiven Fenster. Ist an der aktuellen Position bereits ein Breakpoint gesetzt, so wird dieser entfernt. Die Position, an der ein Breakpoint gesetzt werden kann, hängt von der Sprache ab, in der der Baustein im aktiven Fenster geschrieben ist. In den Texteditoren (AWL, ST) wird der Breakpoint auf die Zeile, in der der Cursor steht, gesetzt, wenn diese Zeile eine Breakpointposition ist (zu erkennen an der dunkelgrauen Farbe des Zeilennummernfeldes). Zum Setzen bzw. Entfernen eines Breakpoints in den Texteditoren können Sie auch auf das Zeilennummernfeld klicken. Im FUP und KOP wird der Breakpoint auf das aktuell markierte Netzwerk gesetzt. Zum Setzen bzw. Entfernen eines Breakpoints im FUP- bzw. KOP-Editor können Sie auch auf das Netzwerknummerfeld klicken. Im AS wird der Breakpoint auf den aktuell markierten Schritt gesetzt. Zum Setzen bzw. Entfernen eines Breakpoints kann im AS auch <Umschalt> mit Doppelklick verwendet werden. Ist ein Breakpoint gesetzt, so wird das Zeilennummernfeld bzw. das Netzwerknummernfeld bzw. der Schritt mit hellblauer Hintergrundfarbe dargestellt. Wenn bei der Programmabarbeitung ein Breakpoint erreicht ist, dann stoppt das Programm, und das entsprechende Feld wird mit einer roten Hintergrundfarbe dargestellt. Um das Programm fortzusetzen, benutzen Sie die Befehle **'Online"Start'**, **'Online"Einzelschritt in'** oder **'Online"Einzelschritt über'**.

Zum Setzen bzw. Entfernen von Breakpoints können Sie auch den Breakpoint-Dialog benutzen.

'Online' 'Breakpoint-Dialog'

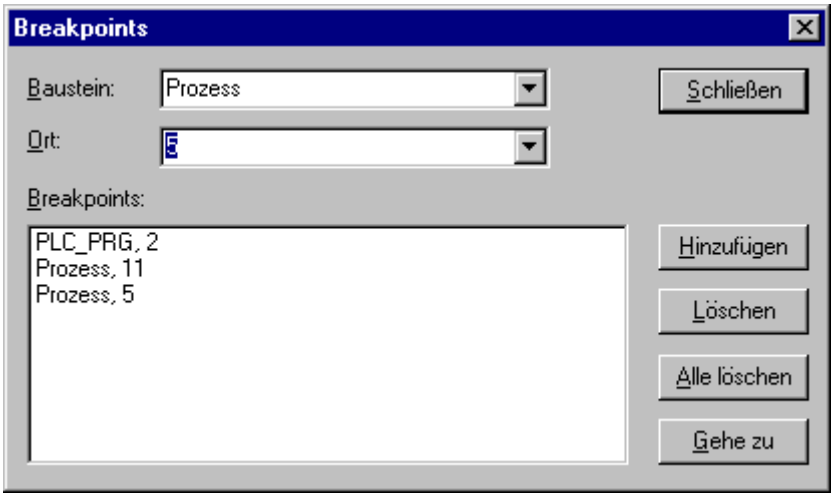
Dieser Befehl öffnet einen Dialog zum Editieren von Breakpoints im gesamten Projekt. Der Dialog zeigt zudem alle aktuell gesetzten Breakpoints an.

Zum Setzen eines Breakpoints wählen Sie in der Combobox Baustein einen Baustein und in der Combobox Ort die Zeile bzw. das Netzwerk, wo Sie den Breakpoint setzen möchten und drücken die Schaltfläche 'Hinzufügen'. Der Breakpoint wird in die Liste aufgenommen.

Um einen Breakpoint zu löschen, markieren Sie in der Liste der gesetzten Breakpoints den zu löschenden aus und drücken die Schaltfläche '**Löschen'**. Mit der Schaltfläche **'Alle löschen'** werden alle Breakpoints gelöscht.

Um zu der Stelle im Editor zu gehen, an der ein bestimmter Breakpoint gesetzt wurde, markieren Sie in der Liste der gesetzten Breakpoints den entsprechenden und drücken die Schaltfläche '**Gehe zu**'.

Zum Setzen bzw. Entfernen von Breakpoints können Sie auch den Befehl '**Online' 'Breakpoint an/aus'** benutzen.



Dialog zum Editieren der Breakpoints

'Online' 'Einzelschritt über' Kurzform: <F10>

Mit diesem Befehl wird ein Einzelschritt ausgeführt, wobei bei Aufrufen von Bausteinen erst nach dessen Abarbeitung angehalten wird. Im AS wird eine komplette Aktion abgearbeitet.

Wenn die aktuelle Anweisung der Aufruf einer Funktion oder eines Funktionsblocks ist, dann wird die Funktion oder der Funktionsblock komplett ausgeführt. Benutzen Sie den Befehl '**Online' 'Einzelschritt in**', um an die erste Anweisung einer aufgerufenen Funktion bzw. eines aufgerufenen Funktionsblocks zu kommen.
Wenn die letzte Anweisung erreicht ist, dann geht das Programm zur nächsten Anweisung des aufrufenden Bausteins weiter.

'Online''Einzelschritt in' Kurzform: <F8>

Es wird ein Einzelschritt abgearbeitet, wobei bei Aufrufen von Bausteinen vor der Ausführung der ersten Anweisung des Bausteins angehalten wird. Gegebenenfalls wird in einen aufgerufenen Baustein gewechselt. Wenn die aktuelle Position ein Aufruf einer Funktion oder eines Funktionsblocks ist, dann geht der Befehl zur ersten Anweisung des aufgerufenen Bausteins weiter. In allen anderen Situationen verhält sich der Befehl genau wie '**Online''Einzelschritt** über'.

'Online''Einzelzyklus' Kurzform: <Strg>+<F5>

Dieser Befehl führt einen einzelnen Steuerungszyklus aus und stoppt nach diesem Zyklus. Dieser Befehl kann kontinuierlich wiederholt werden, um in einzelnen Zyklen fortzufahren. Einzelzyklus endet, wenn der Befehl '**Online''Start**' ausgeführt wird.

'Online' 'Werte schreiben' Kurzform: <Strg>+<F7>

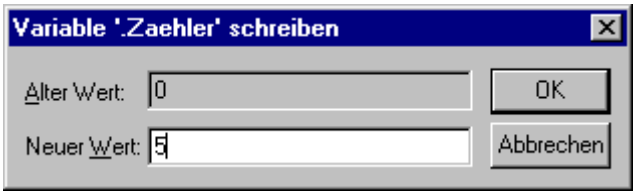
Mit diesem Befehl werden zu Beginn eines Zyklus - einmalig ! - eine oder mehrere Variablen auf benutzerdefinierte Werte gesetzt.

Es können die Werte aller einelementigen Variablen verändert werden, die auch im Monitoring sichtbar sind.

Bevor der Befehl 'Werte Schreiben' ausgeführt werden kann, muss ein Variablenwert zum Schreiben vorbereitet, d.h. definiert werden.

1. Definieren der Werte:

- Bei nicht-boolschen Variablen wird ein doppelter Mausklick auf die Zeile, in der die Variable deklariert ist, durchgeführt, oder die Variable wird markiert und die <Eingabetaste> gedrückt. Daraufhin erscheint die Dialogbox 'Variable <x> schreiben', wo der Wert eingegeben werden kann, der auf die Variable geschrieben werden soll.



Dialog zum Schreiben eines neuen Variablenwertes

- Bei Boolschen Variablen wird durch doppelten Mausklick auf die Zeile, in der die Variable deklariert ist, der Wert getoggelt (Wechsel zwischen TRUE, FALSE und keinem neuen Wert) ohne dass ein Dialog erscheint.

Der zum Schreiben vorgesehene neue Wert wird türkisfarben in spitzen Klammern hinter dem bisherigen Deklarationswert angezeigt, z.B. a=0 <:=34>.

Ausnahme in der Anzeige der zu schreibenden Werte: Im FUP- und KOP-Editor steht der Wert ohne spitze Klammern türkisfarben neben dem Variablennamen.

Das Wertedefinieren kann für beliebig viele Variablen durchgeführt werden.

Die Werte, die für Variablen zum Schreiben eingetragen wurden, können auf die gleiche Weise auch korrigiert bzw. wieder gelöscht werden. Genauso möglich ist dies im **'Online' 'Schreiben/Forcen-Dialog'** (siehe unten).

Die zum Schreiben vorgemerkten Werte werden in einer **Schreibliste (Watchlist)** gespeichert, wo sie bleiben, bis sie tatsächlich geschrieben, gelöscht oder durch den Befehl 'Werte forcen' in eine Forceliste verschoben werden.

2. Schreiben der Werte:

Der Befehl zum Schreiben der in der Schreibliste gesetzten Werte ist an zwei Stellen zu finden:

- Befehl 'Werte schreiben' im Menü 'Online'.
- Schaltfläche 'Werte schreiben' im Dialog 'Editieren der Schreibliste und der Forceliste'.

Wird der Befehl '**Werte schreiben**' ausgeführt, werden alle in der Schreibliste enthaltenen Werte einmalig am Zyklusbeginn auf die entsprechenden Variablen in der Steuerung geschrieben und damit aus der Schreibliste gelöscht. (Wird der Befehl '**Werte forcen**' ausgeführt, werden die betroffenen Variablen ebenfalls aus der Schreibliste gelöscht und in die Forceliste übernommen!)

In der Ablaufsprache können die Einzelwerte, aus denen sich ein Transitions-Ausdruck zusammensetzt, nicht mit 'Werte schreiben' verändert werden. Dies beruht darauf, dass beim Monitoring der 'Gesamtwert' des Ausdrucks, nicht die Werte der Einzelvariablen dargestellt werden (z.B. "a AND b" wird nur dann als TRUE dargestellt, wenn wirklich beide Variablen den Wert TRUE haben).
Im FUP dagegen wird in einem Ausdruck, der beispielsweise als Eingang eines Funktionsblocks verwendet wird, nur die erste Variable gemonitort. Somit ist auch ein 'Werte schreiben' nur für diese Variable möglich.

'Online' 'Werte forcen' Kurzform: <F7> (Werte forcen)

Mit diesem Befehl werden eine oder mehrere Variablen dauerhaft auf benutzerdefinierte Werte gesetzt. Das Setzen erfolgt dabei im Laufzeitsystem jeweils am Anfang und am Ende des Zyklus.

Der zeitliche Ablauf in einem Zyklus: 1.Eingänge lesen, 2. Werte forcen 3. Code abarbeiten, 4. Werte forcen 5. Ausgänge schreiben.

Die Funktion ist so lange aktiv, bis sie durch den Benutzer explizit aufgehoben wird (Befehl 'Online' 'Forcen aufheben') oder das Programmiersystem ausloggt.

Zum Setzen der neuen Werte wird, wie unter 'Online' 'Werte schreiben' (siehe 1. Werte definieren) beschrieben, zunächst eine **Schreibliste** erzeugt. Die in der Schreibliste enthaltenen Variablen sind im Monitoring entsprechend gekennzeichnet. Die Schreibliste wird in eine **Forceliste** übertragen, sobald der Befehl '**Online' 'Werte forcen**' ausgeführt wird. Möglicherweise existiert bereits eine aktive Forceliste, die

dann entsprechend aktualisiert wird. Die Schreibliste wird geleert und die neuen Werte rot als 'geforced' dargestellt. Modifikationen in der Forceliste werden jeweils beim nächsten 'Werte Forcen' auf das Programm übertragen.

Die Forceliste entsteht beim ersten Forcen der in der Schreibliste enthaltenen Variablen, während die Schreibliste bereits **vor** dem ersten Schreiben der enthaltenen Variablen existiert.

Der Befehl zum Forcen einer Variable (und dadurch Aufnahme in die Forceliste) findet sich an folgenden Stellen:

- Befehl 'Werte forcen' im Menü 'Online'.
- Schaltfläche 'Werte forcen' im Dialog 'Editieren der Schreibliste und der Forceliste'.

In der Ablaufsprache können die Einzelwerte, aus denen sich ein Transitions-Ausdruck zusammensetzt, nicht mit 'Werte forcen' verändert werden. Dies beruht darauf, dass beim Monitoring der 'Gesamtwert' des Ausdrucks, nicht die Werte der Einzelvariablen dargestellt wird (z.B. "a AND b" wird nur dann als TRUE dargestellt, wenn wirklich beide Variablen den Wert TRUE haben).

Im FUP dagegen wird in einem Ausdruck, der beispielsweise als Eingang eines Funktionsblocks verwendet wird, nur die erste Variable gemonitort. Somit ist ein 'Werte forcen' nur für diese Variable möglich.

'Online''Forcen aufheben' Kurzform: <Umschalt>+<F7>

Der Befehl beendet das Forcen von Variablenwerten in der Steuerung. Die Variablen ändern ihren Wert wieder normal.

Geforcte Variablen sind im Monitoring an der roten Darstellung ihrer Werte zu erkennen. Es besteht die Möglichkeit, pauschal die komplette Forcelist zu löschen oder aber nur einzelne Variablen daraus vor Ausführen des Befehls zum Aufheben des Forcens vorzumerken.

Um die komplette Forceliste zu löschen, also **für alle Variablen** das Forcen aufzuheben, wählen Sie eine der folgenden Möglichkeiten:

- Befehl 'Forcen aufheben' im Menü 'Online'.
- Schaltfläche 'Forcen aufheben' im Dialog 'Editieren der Schreibliste und der Forceliste'.
- Löschen der kompletten Forceliste über den Dialog 'Löschen der Schreib-/Forcelisten'. Diesen öffnet man mit dem Befehl 'Online'"Forcen aufheben'.

Um das Forcen nur **für einzelne Variablen** aus der Forceliste aufzuheben, müssen Sie diese Variablen zunächst dafür vormerken. Wählen Sie dafür eine der folgenden Möglichkeiten. Die zum Forcen vorgemerkten Variablen sind danach am türkisfarbenen Zusatz <**Forcen aufheben>** erkenntlich.

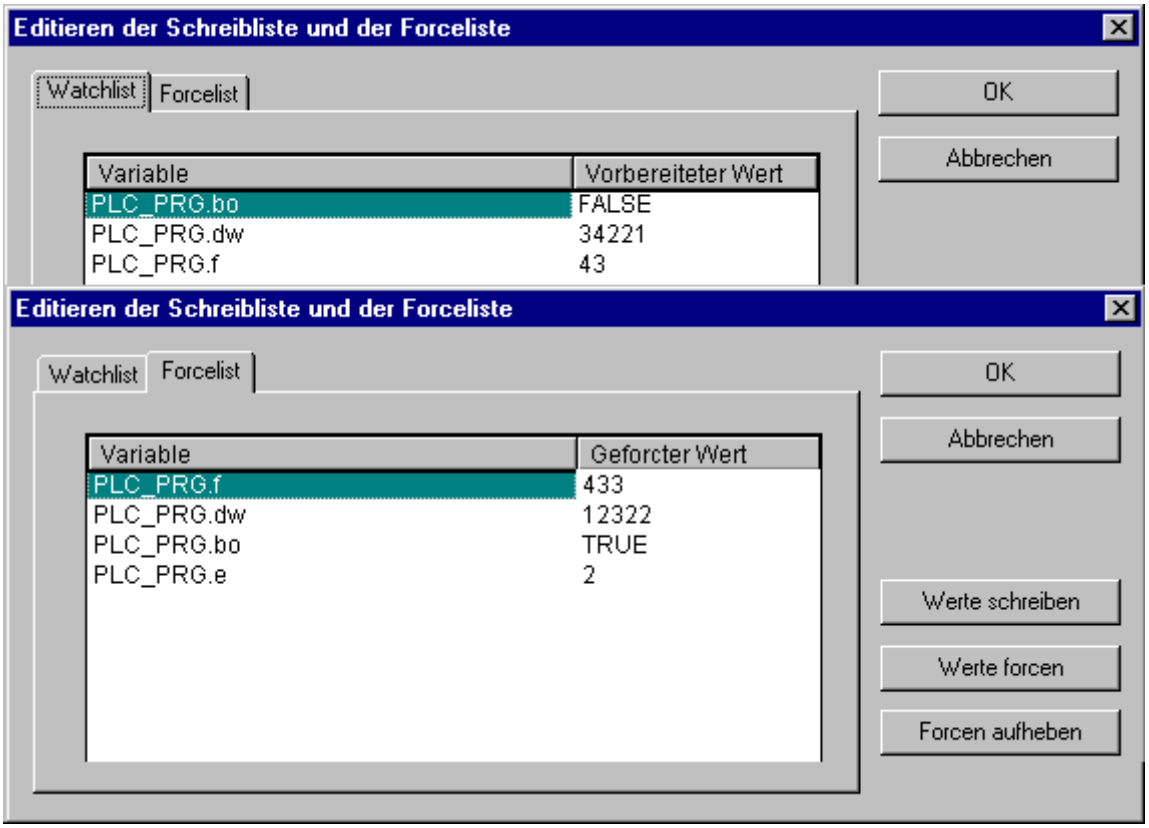
- Ein doppelter Mausklick auf auf die Zeile, in der eine nicht-boolsche geforcte Variable deklariert ist, öffnet den Dialog 'Variable <x> schreiben'. Drücken Sie hier die Schaltfläche <Forcen für diese Variable aufheben>.
- Mit wiederholten doppelten Mausklicks auf auf die Zeile, in der eine boolsche geforcte Variable deklariert ist, können Sie bis zur Anzeige <Forcen aufheben> hinter der Variable toggeln.
- Löschen Sie den Wert im Editierfeld der Spalte 'Geforcter Wert' im Schreiben-/Forcen-Dialog, den Sie über das 'Online'-Menü öffnen können.

Ist für alle gewünschten Variablen die Einstellung "<Forcen aufheben>" hinter dem Wert im Deklarationsfenster sichtbar, führen Sie den Befehl **'Forcen'** aus, der den neuen Inhalt der Forceliste auf das Programm überträgt.

Wenn beim Ausführen des Befehls 'Forcen aufheben' die aktuelle Schreibliste (siehe 'Online' 'Werte schreiben') nicht leer ist, erscheint der Dialog 'Löschen der Schreib-/Forcelisten', in dem der Benutzer entscheiden muss, ob er nur das **Forcen aufheben** oder auch die **Schreibliste** verwerfen will, oder beides.

'Online' 'Schreiben/Forcen-Dialog'

Dieser Befehl führt zu einem Dialog, der in zwei Registern die aktuelle Schreibliste (Watchlist) und Forceliste (Forcelist) darstellt. In einer Tabelle werden jeweils der Variablename und deren zum Schreiben vorbereitete bzw. geforcte Wert dargestellt.



Die Variablen gelangen durch die Befehle **'Online' 'Werte schreiben'** in die Watchlist und werden durch den Befehl **'Online' 'Werte forcen'** in die Forcelist verschoben. Die Werte können hier in den Spalten 'Vorbereiteter Wert' bzw. 'Geforcter Wert' editiert werden, indem per Mausklick auf den Eintrag ein Editierfeld geöffnet wird. Bei nicht typkonsistenter Eingabe wird eine Fehlermeldung ausgegeben. Wird ein Wert gelöscht, bedeutet dies, dass der Eintrag aus der Schreibliste entfernt wird bzw. die Variable zum Aufheben des Forcens vorgemerkt wird, sobald der Dialog mit einem anderen Befehl als **Abbrechen** verlassen wird.

Folgende Befehle, die denen im Online-Menü entsprechen, stehen über Schaltflächen zur Verfügung:

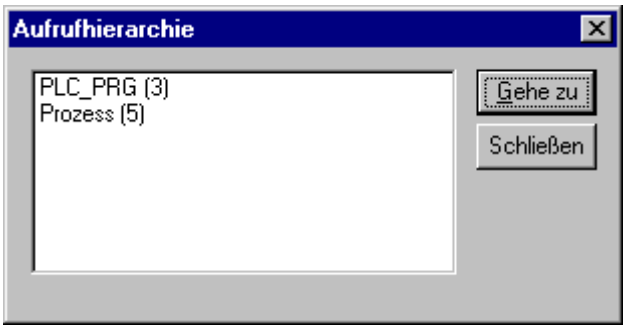
Werte forcen: Alle Einträge der aktuelle Schreibliste werden in die Forcliste verschoben, d.h. die Werte der Variablen in der Steuerung werden geforct. Alle Variablen, die mit 'Forcen aufheben' markiert sind, werden nicht mehr geforct. Der Dialog wird danach geschlossen.

Werte schreiben: Alle Einträge der aktuellen Schreibliste werden einmalig auf die entsprechenden Variablen in der Steuerung geschrieben. Der Dialog wird danach geschlossen.

Forcen aufheben: Alle Einträge der Forceliste werden gelöscht bzw. wenn eine Schreibliste vorhanden ist, geht der Dialog 'Löschen der Schreib-/Forcelisten' auf, in dem der Benutzer entscheiden muss, ob er nur das **Forcen aufheben** oder auch die **Schreibliste verwerfen** will, oder beides. Der Dialog wird danach bzw. nach Schließen des Auswahldialogs geschlossen.

'Online''Aufrufhierarchie'

Diesen Befehl können Sie starten, wenn die Simulation an einem Breakpoint stoppt. Es wird ein Dialog mit einer Liste der Bausteine, die sich momentan im Aufruf-Stack befinden, ausgegeben.



Beispiel für eine Aufrufhierarchie

Der erste Baustein ist stets das in dem momentan eingestellten Debug Task aufgerufene Programm, denn hier beginnt die Abarbeitung. Der letzte Baustein ist stets der Baustein in dem die Abarbeitung momentan steht. Nachdem einer der Bausteine ausgewählt wurde, und die Schaltfläche **'Gehe zu'** gedrückt wurde, wird der ausgewählte Baustein in ein Fenster geladen, und die Zeile, bzw. das Netzwerk, in dem sich die Abarbeitung befindet, wird angezeigt.

'Online' 'Ablaufkontrolle'

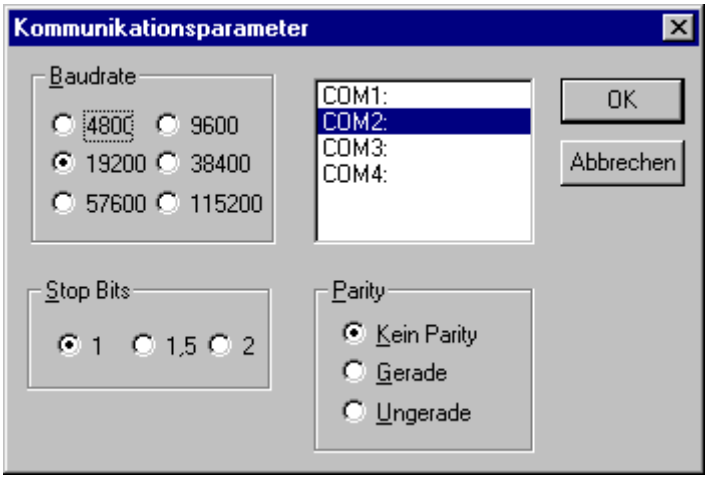
Ist die **Ablaufkontrolle** gewählt, so erscheint ein Haken vor dem Menüpunkt. Danach wird jede Zeile, bzw. jedes Netzwerk, das während des letzten Steuerungszyklus ausgeführt wurde, markiert. Das Zeilennummernfeld bzw. das Netzwerknummernfeld der durchlaufenen Zeilen bzw. Netzwerke wird grün dargestellt. Im AWL-Editor wird am linken Rand jeder Zeile ein weiteres Feld eingefügt, in dem der aktuelle Inhalt des Akkumulators angezeigt wird. In den graphischen Editoren zum Funktionsplan und Kontaktplan wird in allen Verbindungslinien, die keine boolschen Werte transportieren, ein weiteres Feld eingefügt. Wenn diese Aus- und Eingänge belegt werden, dann wird der Wert, der über die Verbindungslinie transportiert wird, in diesem Feld angezeigt. Verbindungslinien die ausschließlich boolsche Werte transportieren, werden dann blau eingefärbt, wenn sie TRUE transportieren, so kann der Informationsfluss ständig mitverfolgt werden.

'Online''Simulation'

Dieser Menüpunkt ist nur bei einem Programm für den BusController BC anwählbar. Bei TwinCAT für den PC kann direkt in dem Laufzeitsystem, das auch später die Steuerung übernimmt ohne angeschlossene E/A gearbeitet werden. Ist Simulation ausgewählt, so erscheint ein Haken vor dem Menüpunkt. Im Simulationsmodus läuft das Benutzerprogramm auf demselben PC unter Windows. Dieser Modus wird benutzt, um das Projekt zu testen. Die Kommunikation zwischen dem PC und der Simulation benutzt den Windows Message Mechanismus. Wenn das Programm nicht im Simulationsmodus ist, dann läuft das Programm auf der Steuerung. Die Kommunikation zwischen dem PC und der Steuerung läuft typischerweise über die serielle Schnittstelle. Der Status dieses Flags wird mit dem Projekt gespeichert.

'Online' 'Kommunikationsparameter'

Dieser Dialog ist nur für TwinCAT mit dem BusController (BC) anwählbar. In einem Dialog können die Parameter für die Übertragung über die serielle Schnittstelle eingegeben werden. Wichtig ist, dass diese Parameter mit den in der Steuerung eingestellten übereinstimmen.



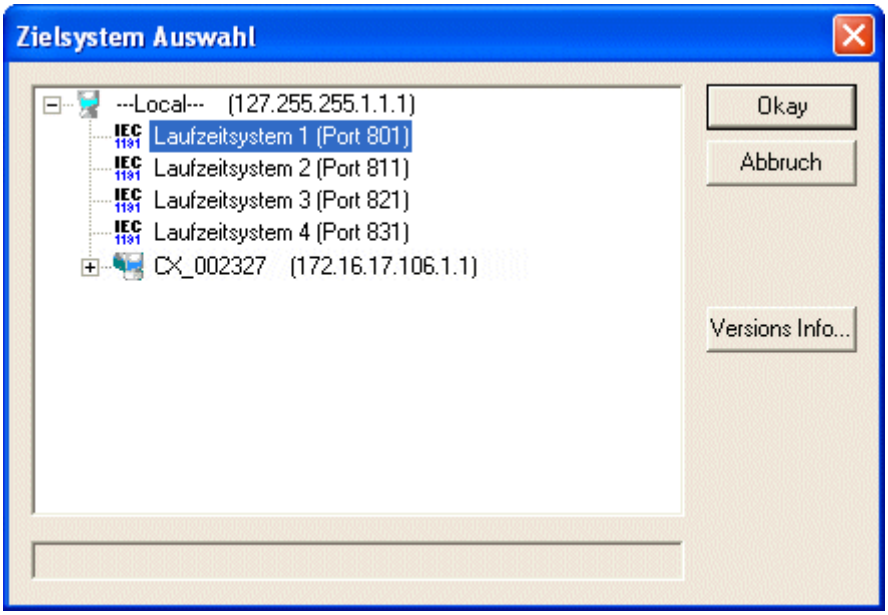
Dialog zur Einstellung der Kommunikationsparameter

Eingestellt werden können die Baudrate, ob mit geradem, ungeradem oder keinem Parity übertragen werden soll, die Zahl der Stopbits sowie die Schnittstelle (COM1, COM2 etc.), über die übertragen werden soll. Die gewählten Parameter werden mit dem Projekt gespeichert.

'Online' 'Auswahl des Zielsystems'

Nur TwinCAT PC:

Hier besteht die Möglichkeit das Zielsystem (d.h. das Laufzeitsystem in dem das SPS Programm läuft) auszuwählen. Sie sehen alle lokalen (maximal vier) und remote vorhandenen Laufzeitsysteme.



Nach Abspeichern des SPS-Projektes, wird die Information des gewählten Zielsystems (Port-Nummer) im Projekt mit abgelegt, damit sie z.B. beim Import oder Rescan im TwinCAT System Manager zur Verfügung steht.

'Online' 'Quellcode laden'

Mit diesem Befehl wird der Quellcode des Projekts in die Steuerung geladen. Dieser ist nicht zu verwechseln mit dem Code, der beim Übersetzen des Projekts entsteht ! Welche Optionen für den Download gelten (Zeitpunkt, Umfang) können Sie im Dialog **'Projekt' 'Optionen' 'Sourcedownload'** einstellen.

'Online' 'Erzeugen eines Bootprojekts'

Wird dieser Befehl online ausgeführt, wird das kompilierte Projekt so auf der Steuerung abgelegt, daß die Steuerung es bei einem Neustart automatisch laden kann. Je nach Zielsystem erfolgt die Speicherung des Bootprojekts auf unterschiedliche Weise. Beispielsweise wird auf dem PC eine Datei im TwinCAT Bootverzeichnis angelegt: TCPLC_P_x.wbp (x steht für die Nummer des Laufzeitsystem 1 bis 4).

'Online' 'Erzeugen eines Bootprojekts (offline)'

Wird dieser Befehl ausgeführt, wird das kompilierte Projekt in das Verzeichnis des Projektes (*.pro) abgelegt. Je nach Zielsystem erfolgt die Speicherung des Bootprojektes auf unterschiedliche Weise. Auf einem WinXP PC wird im Projekt-Ordner eine Datei TcPLC_P_x.wbp erzeugt. x steht dabei für das jeweilige Laufzeitsystem. Damit das Bootprojekt bei einem Neustart des Zielsystems automatisch geladen wird, muss die erzeugte Datei manuell in das entsprechende TwinCAT Boot Verzeichnis kopiert werden.

'Online' 'Datei in Steuerung schreiben'

Dieser Befehl dient dazu, eine beliebige Datei in die Steuerung zu laden. Er öffnet den Dialog 'Datei in Steuerung schreiben', in dem Sie die gewünschte Datei markieren. Nach Schließen des Dialogs über die Schaltfläche 'Öffnen' wird die Datei in die Steuerung geladen und dort unter demselben Namen abgelegt. Das Laden wird durch eine Fortschrittsanzeige begleitet.

Mit dem Befehl '**Online' 'Datei aus Steuerung laden**' können Sie eine auf der Steuerung abgelegte Datei wieder laden.

'Online' 'Datei aus Steuerung laden'

Mit diesem Befehl können Sie eine über '**Online' 'Datei in Steuerung schreiben**' auf der Steuerung abgelegte Datei wieder laden. Sie erhalten den Dialog 'Datei aus Steuerung laden'. Geben Sie unter **Dateiname** den Namen der gewünschten Datei ein und wählen Sie das Zielverzeichnis, in das sie geladen werden soll (**Speichern in**), sobald der Dialog über die Schaltfläche 'Speichern' geschlossen wird.

Folgende Menü-Einträge gibt es nur, wenn die Steuerungsplattform (das Zielsystem) ein BCxxxx ist:

'Online' 'Koppler'

Hier werden Ihnen spezielle Befehle zur Verfügung gestellt.

Klemmbus Reset: Ein Reset auf dem Klemmenbus wird durchgeführt.

Koppler Reset: Der Koppler wird neu gestartet.

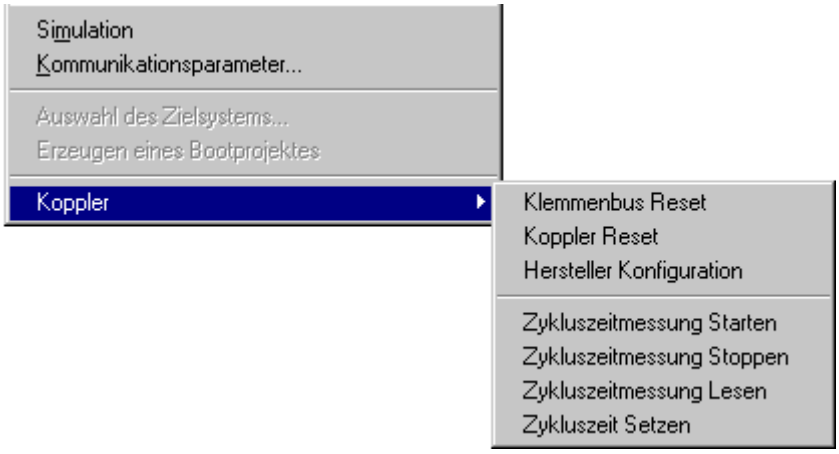
Hersteller Konfiguration: Der Auslieferungszustand des Kopplers wird wieder herstellt (Bootprojekt wird gelöscht), danach muss der Koppler neu gestartet durch einen Koppler Reset.

Zyklusmessung starten: Die Zykluszeitmessung auf dem Koppler wird gestartet.

Zyklusmessung stoppen: Die Zykluszeitmessung auf dem Koppler wird gestoppt.

Zyklusmessung lesen: Es werden minimale Zykluszeit (seit Messanfang), maximale Zykluszeit, mittlere Zykluszeit (über die letzten 200 Zyklen) und aktuelle Zykluszeit sowie die Anzahl der PLC-Zyklen gelesen.

Zykluszeit setzen: Hier wird die Soll-Zykluszeit und die Hintergrundbearbeitungszeit eingestellt. Wenn die Ist-Zykluszeit größer ist als die Soll-Zykluszeit kommt der nächste Zyklus entsprechend später (man hat dann keine konstanten Zyklus mehr). Generell sollte die Soll-Zykluszeit wie folgt berechnet werden: 1.25 * mittlere Zykluszeit (aus der Messung), Hintergrundbearbeitungszeit: 0,25 * mittlere Zykluszeit. Die Zeiten können nur auf eine 1 ms genau eingestellt werden.



4.7 Fenster

Unter dem Menüpunkt 'Fenster' finden Sie alle Befehle zur Fensterverwaltung. Das sind sowohl Befehle zum automatischen Anordnen Ihrer Fenster, als auch zum Öffnen des Bibliothekverwalters und zum Wechseln zwischen Ihren geöffneten Fenstern. Am Ende des Menüs finden Sie eine Auflistung aller geöffneten Fenster in der Reihenfolge, in der sie geöffnet wurden. Mit einem Mausklick auf den jeweiligen Eintrag wechseln Sie zum gewünschten Fenster. Vor dem aktiven Fenster erscheint ein Haken.

'Fenster' 'Nebeneinander'

Mit diesem Befehl ordnen Sie alle Fenster im Arbeitsbereich nebeneinander an, so dass sie sich nicht überlappen und den gesamten Arbeitsbereich ausfüllen.

'Fenster' 'Untereinander'

Mit diesem Befehl ordnen Sie alle Fenster im Arbeitsbereich untereinander an, so dass sie sich nicht überlappen und den gesamten Arbeitsbereich ausfüllen.

'Fenster' 'Überlappend'

Mit diesem Befehl ordnen Sie alle Fenster im Arbeitsbereich kaskadenförmig hintereinander an

'Fenster' 'Symbole anordnen'

Mit diesem Befehl ordnen Sie alle minimierten Fenster im Arbeitsbereich in einer Reihe am unteren Ende des Arbeitsbereiches an.

'Fenster' 'Alle Schließen'

Mit diesem Befehl schließen Sie alle geöffneten Fenster im Arbeitsbereich.

'Fenster' 'Meldungen' Kurzform: <Umschalt>+<Esc>

Mit diesem Befehl öffnen bzw. schließen Sie das Meldungsfenster mit den Meldungen aus dem letzten Übersetzungs-, Überprüfungs- oder Vergleichsvorgang. Ist das Meldungsfenster geöffnet, so erscheint im Menü ein Haken vor dem Befehl.

'Fenster' 'Bibliotheksverwaltung'

Mit diesem Befehl öffnen Sie den Bibliotheksverwalter.

‘Fenster’ ’Logbuch’

Mit diesem Befehl öffnen Sie das Logbuch-Fenster, in dem Protokolle der Online-Sessions angezeigt werden können.

4.8 Hilfesystem

Sollten Sie während Ihrer Arbeit irgendwelche Probleme mit TwinCAT PLC Control haben, steht Ihnen eine Online-Hilfe zur Verfügung. Dort finden Sie alle Informationen, die sich auch in diesem Handbuch befinden. Die Hilfe verweist direkt auf das Beckhoff Information System. Das Beckhoff Information System muss installiert sein.

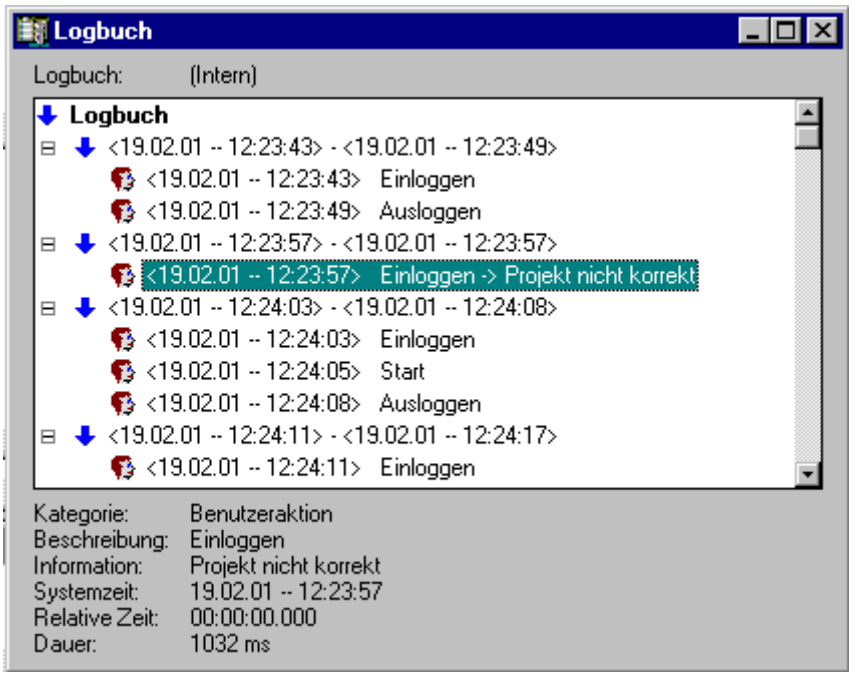
4.9 Logbuch

Das Logbuch speichert in chronologischer Reihenfolge Aktionen, die während einer Online-Session auftreten. Dazu wird zu jedem Projekt eine binäre Logdatei (*.log) angelegt. Darüber hinaus kann der Anwender Auszüge aus dem jeweiligen Projekt-Logbuch in einem externen Logbuch abspeichern.

Das Logbuch-Fenster kann im Offline- und Online-Modus geöffnet werden und kann somit online auch als unmittelbarer Monitor dienen.

'Fenster' 'Logbuch'

Zum Öffnen wählen Sie den Menüpunkt 'Fenster"Logbuch'.



Über dem Protokollfenster steht hinter **Logbuch**: der Dateiname des momentan angezeigten Logbuchs. Falls es sich dabei um das Logbuch des aktuellen Projekts handelt, wird "(Intern)" angezeigt.

Im Protokollfenster werden die protokollierten Einträge dargestellt. Der neueste Eintrag wird jeweils unten angehängt.

Es werden nur Aktionen der Kategorien dargestellt, die im Menü 'Projekt' 'Optionen' 'Logbuch' im Bereich 'Filter' aktiviert sind !

Unterhalb des Protokollfensters wird jeweils die zum im Fenster selektierten Eintrag verfügbare Information angezeigt:

Kategorie: Die Kategorie des einzelnen Logbuch-Eintrags. Möglich sind folgende vier Kategorien:

- **Benutzeraktion**: Der Anwender hat eine Online-Aktion (typischerweise aus dem Online-Menü) ausgeführt.
- **Interne Aktion**: Es ist eine interne Aktion in der Online-Schicht ausgeführt worden (z.B. Delete Buffers oder Init Debugging)
- **Statusänderung**: Der Status des Laufzeitsystems hat sich geändert (z.B. von Running auf Break, falls ein Breakpoint angelaufen wurde)
- **Exception**: Eine Ausnahme ist aufgetreten, z.B. ein Kommunikationsfehler

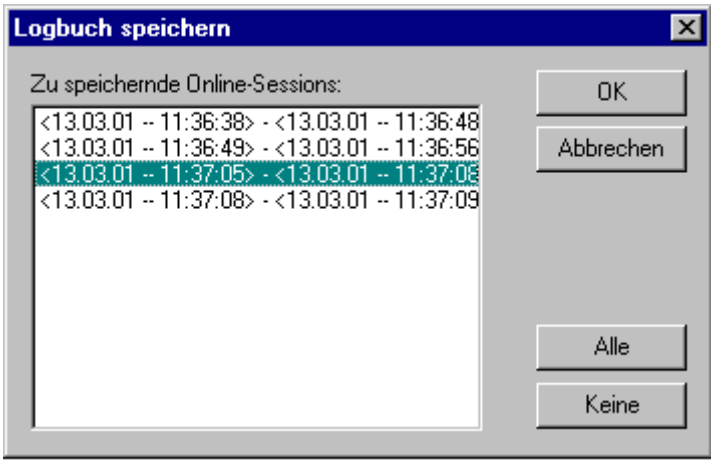
Beschreibung: Die Art der Aktion. Benutzeraktionen heißen ebenso wie ihre korrespondierenden Menübefehle, alle anderen Aktionen sind englischsprachig und heißen ähnlich der zugehörigen OnlineXXX()-Funktion.

Information: Dieses Feld enthält eine Beschreibung über einen möglicherweise während der Aktion aufgetretenen Fehler. Das Feld ist leer, falls kein Fehler aufgetreten ist.

Systemzeit: Die momentane Systemzeit bei Beginn der Aktion; sekundengenau.
Relative Zeit: Die Zeit relativ zum Beginn der Online-Session; millisekundengenau.
Dauer: Die Dauer der Aktion in Millisekunden.

Laden...
Über den Standarddialog zum Öffnen einer Datei kann eine externe Logbuch-Datei *.log geladen und angezeigt werden. Das im Projekt befindliche Logbuch wird durch den Befehl nicht überschrieben. Wird das Logbuchfenster geschlossen und danach wieder geöffnet oder wird eine neue Online-Session gestartet, so wird die geladene Version wieder durch das Projekt-Logbuch ersetzt.

Speichern
Dieser Menüpunkt ist nur anwählbar, wenn aktuell das Projekt-Logbuch angezeigt wird. Es ermöglicht die Speicherung eines Auszugs des Projekt-Logbuchs in einer externen Datei. Dazu wird folgender Dialog eingeblendet, in dem die zu speichernden Online-Sessions ausgewählt werden können



Projekt-Logbuch anzeigen
Dieser Befehl ist nur anwählbar, wenn aktuell ein externes Logbuch angezeigt wird. Er schaltet die Darstellung wieder auf das Projekt-Logbuch zurück.

Speicherung des Projekt-Logbuchs

Unabhängig von einer möglichen Speicherung des Logbuchs in einer externen Datei (siehe oben) wird das Projekt-Logbuch automatisch in einer binären Datei <projectname>.log gespeichert. Wird im Dialog **'Projekt' 'Optionen' 'Logbuch'** nicht explizit ein anderer Pfad angegeben, erfolgt die Ablage im gleichen Verzeichnis, in dem das Projekt gespeichert wird.

Die maximal zu speichernde Anzahl von Online-Sessions kann im Dialog 'Projekt' 'Optionen' 'Logbuch' eingestellt werden. Wird diese Zahl während der laufenden Aufzeichnung überschritten, so wird die jeweils älteste Session zugunsten der neuesten aus dem Aufzeichnungspuffer gelöscht.

5 Die Editoren

Alle Editoren für Bausteine bestehen aus einem Deklarationsteil und einem Rumpf. Diese sind getrennt durch einen Bildschirmteiler, den man nach Bedarf verschieben kann, indem man ihn mit der Maus anklickt, und mit gedrückter Maustaste nach oben oder unten fährt Der Rumpf kann aus einem Text- oder Grafikeditor bestehen, der Deklarationsteil ist immer ein Texteditor.

Druckgrenzen

Die vertikalen und horizontalen Seitenbegrenzungen, die beim Drucken des Editorinhaltes gelten, sind durch rot gestrichelte Linien sichtbar, falls die Option '**Druckbereiche anzeigen**' in den Projektoptionen im Dialog '**Arbeitsbereich**' angewählt wurde. Dabei gelten die Vorgaben des eingestellten Druckers sowie die im Menü '**Datei**' '**Einstellungen Dokumentation**' ausgewählte Größe der Druckvorlage. Ist kein Drucker bzw. keine Druckvorlage eingestellt, wird von einer Default-Belegung ausgegangen (Default.DFR und Standard-Drucker). Die horizontalen Druckgrenzen werden so eingezeichnet, als wäre bei den 'Einstellungen Dokumentation' die Optionen 'Neue Seite je Objekt' bzw. 'Neue Seite je Unterobjekt' angewählt. Die unterste Grenze ist nicht dargestellt.

Eine exakte Anzeige der Druckbereichsgrenzen ist nur bei einem auf 100% eingestellten Zoomfaktor gewährleistet.

Kommentar

Benutzerkommentare müssen in die speziellen Zeichenfolgen "(*" und "*)" eingeschlossen werden. Beispiel: (*Dies ist ein Kommentar.*)
Kommentare sind in allen Texteditoren und dort an beliebiger Stelle erlaubt, d.h. in allen Deklarationen, den Sprachen AWL und ST und in selbstdefinierten Datentypen. Wird das Projekt unter Verwendung einer **Dokuvorlage** ausgedruckt, erscheint in textbasierten Programmteilen der bei der Variablendeklaration eingegebene Kommentar jeweils hinter der Variable.

In den graphischen Editoren FUP und KOP können zu jedem Netzwerk Kommentare eingegeben werden. Suchen Sie hierzu das Netzwerk aus, das Sie kommentieren möchten und aktivieren Sie 'Einfügen' 'Kommentar'. Im CFC gibt es spezielle Kommentarbausteine, die beliebig platziert werden können.

In AS können Sie Kommentare zum Schritt im Dialog zum Editieren von Schrittattributen eingeben.

Auch **verschachtelte Kommentare** sind erlaubt, wenn die entsprechende Option im Dialog '**Projekt**' '**Optionen**' '**Übersetzungsoptionen**' aktiviert ist.
Wenn Sie den Mauszeiger im Online Modus eine kurze Zeit über einer Variablen halten, wird der Typ und gegebenenfalls die Adresse und der Kommentar der Variablen in einem Tooltip angezeigt.

Zoom zu aufgerufenem Baustein

Dieser Befehl steht im Kontextmenü (<F2>) oder im Menü Extras zur Verfügung, wenn der Cursor in Texteditoren auf dem Namen eines aufgerufenen Bausteins steht bzw. wenn in grafischen Editoren die Box eines Bausteins markiert ist. Zoom öffnet den betreffenden Baustein in seinem Editorfenster. Handelt es sich um einen Baustein aus einer Bibliothek, so wird der Bibliotheksverwalter aufgerufen und der entsprechende Baustein angezeigt.

Instanz öffnen

Dieser Befehl entspricht dem Befehl 'Projekt' 'Instanz öffnen' . Er steht im Kontextmenü oder im Menü Extras zur Verfügung, wenn der Cursor in Texteditoren auf dem Namen eines Funktionsblocks steht bzw. wenn in grafischen Editoren die Box eines Funktionsblocks markiert ist.

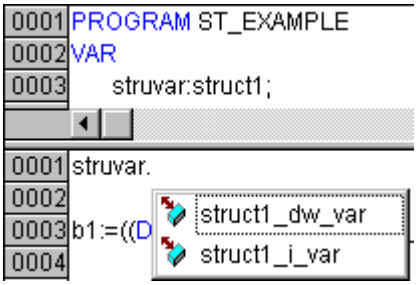
Intellisense Funktion

Wenn in den Projektoptionen in der Kategorie '**Editor**' die Option '**Komponenten auflisten**' aktiviert ist, steht in allen Editoren, im Watch- und Rezepturverwalter, in der Visualisierung und in der Tracekonfiguration die **"Intellisense Funktion"** zur Verfügung:

- Wird anstelle eines Bezeichners ein Punkt "." eingegeben, öffnet sich eine Auswahlliste aller lokalen und globalen Variablen. Aus dieser Liste kann ein Element selektiert und durch Drücken der Eingabetaste hinter dem Punkt eingefügt werden. Das Einfügen funktioniert ebenfalls nach einem Doppelklick auf das Listenelement.
- Wird als Bezeichner eine Funktinsblockinstanz oder eine als Struktur definierte Variable, gefolgt von einem Punkt, eingegeben, öffnet sich nach Eingabe des Punktes eine Auswahlliste der Ein- und Ausgangsvariablen des Funktionsblocks bzw. der Strukturkomponenten.

Beispiel:

Eingabe von "struvar." -> die Komponenten der Struktur struct1 werden angeboten



5.1 Deklarationseditor

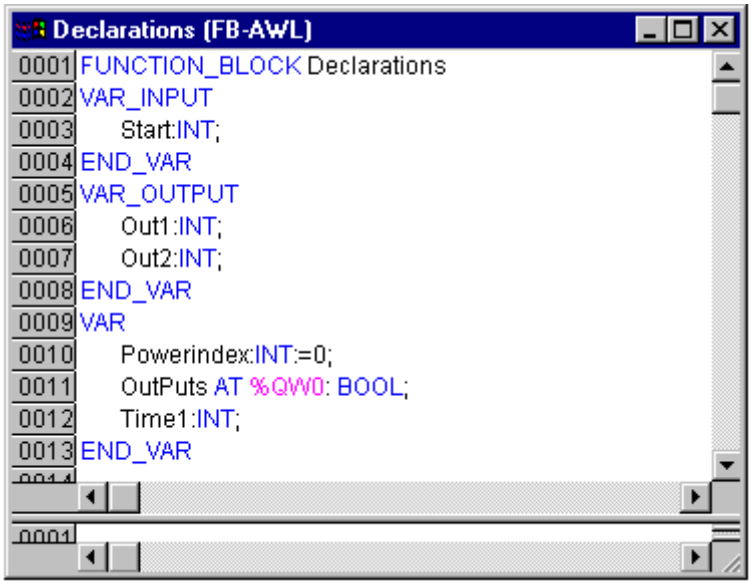
Der Deklarationseditor wird verwendet bei der Variablendeklaration von Bausteinen und globalen Variablen, zur Datentypdeklaration, und im Watch- und Rezepturverwalter. Er verfügt über die Windows-üblichen Funktionalitäten und auch die der IntelliMouse kann genützt werden, wenn der entsprechende Treiber installiert ist.

Im Überschreibmodus wird in der Statusleiste 'ÜB' schwarz angezeigt, zwischen Überschreib- und Einfügemodus kann mit der Taste <Einfg> gewechselt werden.
Die Variablendeklaration wird durch Syntaxcoloring unterstützt.

Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste).

Deklarationsteil

Im Deklarationsteil eines Bausteins werden alle Variablen deklariert, die nur in diesem Baustein verwendet werden. Dies können Eingabevariablen, Ausgabevariablen, EinAusgabevariablen, lokale Variablen, remanente Variablen und Konstanten sein. Die Deklarationssyntax orientiert sich am Standard der IEC61131-3. Ein Beispiel für eine korrekte Variablendeklaration im TwinCAT PLC Control-Editor:



Deklarationseditor

Eingabevariablen

Zwischen den Schlüsselwörtern VAR_INPUT und END_VAR werden alle Variablen deklariert, die als Eingabevariablen eines Bausteins dienen, das heißt, an der Aufrufstelle kann der Wert der Variablen beim Aufruf mitgegeben werden.

Beispiel:

```
VAR_INPUT
    in1:INT; (* 1. Eingabevariable*)
END_VAR
```

Ausgabevariablen

Zwischen den Schlüsselwörtern VAR_OUTPUT und END_VAR werden alle Variablen deklariert, die als Ausgabevariablen eines Bausteins dienen, das heißt, diese Werte werden dem aufrufenden Baustein zurückgeliefert, dort können diese abgefragt und weiterverwendet werden.

Beispiel:

```
VAR_OUTPUT
    out1:INT; (* 1. Ausgabevariable*)
END_VAR
```

Ein-Ausgabevariablen

Zwischen den Schlüsselwörtern VAR_IN_OUT und END_VAR werden alle Variablen deklariert, die als Ein- und Ausgabevariablen eines Bausteins dienen.

Bei dieser Variablen wird direkt der Wert der übergebenen Variablen verändert ("Übergabe als Pointer", Call-by-Reference). Deshalb kann der Eingabewert für eine solche Variable keine Konstante sein. Deshalb können auch VAR_IN_OUT Variablen eines Funktionsblocks nicht von außen direkt über <Funktionsblockinstanz>.<Ein-/Ausgabevariable> gelesen oder beschrieben werden !

Beispiel:

```
VAR_IN_OUT
    inout1:INT; (* 1. EinAusgabevariable *)
END_VAR
```

Lokale Variablen

Zwischen den Schlüsselwörtern VAR und END_VAR werden alle lokalen Variablen eines Bausteins deklariert. Diese haben keine Verbindung nach außen, das heißt, es kann nicht von außen auf sie zugegriffen werden.

Beispiel:

```
VAR
    loc1:INT; (* 1. Lokale Variable*)
END_VAR
```

Remanente Variablen

Remanente Variablen können ihren Wert über die übliche Programmlaufzeit hinaus behalten. Dazu gehören Retain-Variablen und Persistente Variablen.

- Retain-Variablen werden mit dem Schlüsselwort RETAIN gekennzeichnet. Diese Variablen behalten ihren Wert nach einem Shutdown von TwinCAT. Beim Kommando 'Online' 'Reset' wird die Steuerung mit Initialwerten gestartet. Ein Anwendungsbeispiel für Retain Variablen wäre ein Stückzähler in einer Fertigungsanlage, der nach einem Stromausfall weiterzählen soll.

Alle anderen Variablen werden neu initialisiert, entweder mit ihren initialisierten Werten oder mit den Standardinitialisierungen.

Beispiel:

```
VAR RETAIN
    rem1:INT; (* Remanente Variable*)
END_VAR
```

HINWEIS

Lokale Variable als RETAIN deklariert

Wenn eine lokale Variable in einem Programm als RETAIN deklariert ist, wird genau diese Variable im Retain-Bereich gespeichert (wie eine globale Retain-Variable).

Wenn eine lokale Variable in einem Funktionsblock als RETAIN deklariert ist, wird die komplette Instanz dieses Funktionsblocks im Retain-Bereich gespeichert (alle Daten des Bausteins), wobei jedoch nur die deklarierte Retain-Variable als solche behandelt wird.

Wenn eine lokale Variable in einer Funktion als RETAIN deklariert ist, hat dies keine Auswirkung. Die Variable wird nicht im Retain-Bereich gespeichert !

Persistente Variablen

Neben den remanenten Variablen, die mit RETAIN spezifiziert werden, gibt es eine weitere Klasse von remanenten Variablen. Diese Variablen werden mit ihrem Instanzpfad und Symbolnamen gesichert. Die Generierung von Symbolen muss dazu angewählt sein. Während die mit RETAIN gesicherten Variablen nach einem "Rebuild all" des SPS Programms und anschließendem Kaltstart nicht mehr zur Verfügung stehen, bleiben persistente Variablen bestehen. Bei einem Reset der SPS werden auch RETAIN Variablen neu initialisiert, persistente Variablen lassen sich nur mit einem **Urlöschen** initialisieren. Die persistenten Variablen werden beim Herunterfahren des TwinCAT System in eine Datei gesichert und beim Neustart wieder mit ihren gesicherten Werten vorinitialisiert.

Beispiel:

```
VAR PERSISTENT
    Rem2:INT; (* Persistente Variable*)
END_VAR
```

Zur Überprüfung der Richtigkeit der PERSISTENT und RETAIN Variablen gibt es in den Systemvariablen in der Struktur SYSTEMINFO [▶ 341] ein Byte bootDataFlags. Dieses Byte zeigt den Zustand nach dem Laden der Bootdaten an. Die oberen vier Bit signalisieren den Zustand der persistenten Daten, die unteren vier Bit den Zustand der retain Daten. Um diese Informationen nutzen zu können, muss die Bibliothek "TcSystem.lib" eingebunden werden.

Bitnummer	Beschreibung
0	RETAIN Variablen: LOADED (fehlerfrei geladen)
1	RETAIN Variablen: INVALID (es wurde die Sicherungskopie geladen, weil keine gültige Datei vorhanden war)
2	RETAIN Variablen: REQUESTED (RETAIN Variablen sollten geladen werden, Einstellung im TwinCAT System Control))
3	reserviert
4	PERSISTENT Variablen: LOADED (fehlerfrei geladen)
5	PERSISTENT Variablen: INVALID (es wurde die Sicherungskopie geladen, weil keine gültige Datei vorhanden war)
6	reserviert
7	reserviert

Beim Shutdown (Stop) von TwinCAT werden die PERSISTENT und die RETAIN Daten in zwei Dateien auf die Festplatte geschrieben. Der Pfad kann im TwinCAT System Control über die TwinCAT System Eigenschaften (Reiter PLC) angegeben werden. Die Standardeinstellung ist "<Laufwerk>:\TwinCAT\Boot". Die Dateien haben alle einen festen Namen und eine feste Endung:

Dateiname	Beschreibung
TCPLC_P_x.wbp	Bootprojekt (x = Nummer des Laufzeitsystems)
TCPLC_R_x.wbp	RETAIN Variablen (x = Nummer des Laufzeitsystems)
TCPLC_T_x.wbp	PERSISTENT Variablen (x = Nummer des Laufzeitsystems)
TCPLC_R_x.wb~	Sicherungskopie der RETAIN Variablen (x = Nummer des Laufzeitsystems)
TCPLC_T_x.wb~	Sicherungskopie der PERSISTENT Variablen (x = Nummer des Laufzeitsystems)

Kann beim Shutdown (Stop) von TwinCAT die Datei der persistenten und/oder retain Variablen nicht geschrieben werden, so wird standardmäßig die Sicherungsdatei geladen. In der SPS ist dann im bootDataFlags das Bit 1 (für die RETAIN Variablen) oder/und das Bit 5 (für die PERSISTENT Variablen) gesetzt.

Soll die Sicherungsdatei auf keinen Fall verwendet werden, so muss man in der NT Registry eine Einstellung vornehmen. Hierzu ist mit dem Registry-Editor unter

```
[HKEY_LOCAL_MACHINE\SOFTWARE\Beckhoff\TwinCAT\Plc]
"ClearInvalidRetainData"=dword:00000000
"ClearInvalidPersistentData"=dword:00000000
```

der Wert von "ClearInvalidRetainData" auf 1 bzw. von "ClearInvalidPersistentData" auf 1 zu setzen. Die Defaulteinstellung ist 0.

Konstanten, Typed Literals

Konstanten werden mit dem Schlüsselwort **CONSTANT** gekennzeichnet. Sie können lokal oder global deklariert werden.

Syntax:

```
VAR CONSTANT
    <Bezeichner>:<Typ> :=
<Initialisierung>;
END_VAR
```

Beispiel:

```
VAR CONSTANT
    con1:INT:=12; (* 1. Konstante*)
END_VAR
```

Eine Auflistung möglicher Konstanten finden Sie im Anhang. Sehen Sie dort auch zur Möglichkeit der Verwendung von getypten Konstanten (Typed Literals).

Externe Variablen

Mit dem Schlüsselwort **EXTERNAL** werden globale Variablen gekennzeichnet, die in den Baustein importiert werden sollen. Sie erscheinen im Online-Modus ebenfalls im Watchfenster des Deklarationsteils.

Stimmt die VAR_EXTERNAL-Deklaration nicht mit der globalen Deklaration überein, so wird folgende Fehlermeldung ausgegeben: "Deklaration von '<var>' stimmt nicht mit globaler Deklaration überein!"

Existiert die globale Variable nicht, so wird folgende Fehlermeldung ausgegeben: "Unbekannte globale Variable: '<var>'!"

Beispiel:

```
VAR EXTERNAL
    var_ext1:INT:=12; (* 1. externe Variable *)
END_VAR
```

Schlüsselwörter

Schlüsselwörter sind in allen Editoren in Großbuchstaben zu schreiben. Schlüsselwörter dürfen nicht als Variablennamen verwendet werden.

Variablendeklaration

Eine Variablendeklaration hat folgende Syntax:

```
<Bezeichner> {AT
<Adresse>}:<Typ> {:= <Initialisierung>};
```

Die Teile in geschweiften Klammern {} sind optional.

Für den Bezeichner, also den Namen von Variablen ist zu beachten, dass er keine Leerstellen und Umlaute enthalten darf, er darf nicht doppelt deklariert werden und nicht identisch mit Schlüsselwörtern sein. Groß-/Kleinschreibung bei Variablen wird nicht beachtet, das heißt VAR1, Var1 und var1 sind keine unterschiedlichen Variablen. Unterstriche sind in Bezeichner signifikant, z.B. werden "A_BCD" und "AB_CD" als unterschiedliche Bezeichner interpretiert. Mehrfach aufeinanderfolgende Unterstriche am Anfang eines Bezeichners oder in einem Bezeichner sind nicht erlaubt.

Die Bezeichnerlänge sowie der signifikante Bereich sind unbegrenzt.

Alle Variablendeklarationen und Datentypelemente können Initialisierungen enthalten. Sie erfolgen mit dem Zuweisungsoperator " := ". Für Variablen von elementaren Typen sind diese Initialisierungen Konstanten. Die Default-Initialisierung ist für alle Deklarationen 0.

Beispiel:

```
var1:INT:=12; (* Integervariable mit
Initialwert 12*)
```

Wenn Sie eine Variable direkt an eine bestimmte Adresse binden wollen, dann müssen Sie die Variable mit dem Schlüsselwort **AT** deklarieren. Zur schnelleren Eingabe von Deklarationen verwenden Sie den **Kurzformmodus**. In Funktionsblöcken können Variablen auch mit unvollständigen Adreßangaben spezifiziert werden. Um solche Variablen in einer lokalen Instanz zu nutzen, muss dafür ein Eintrag in der **Variablenkonfiguration** vorgenommen werden.

Beachten Sie die Möglichkeit des automatischen Deklarierens!

AT-Deklaration

Wenn Sie eine Variable direkt an eine bestimmte Adresse binden wollen, dann müssen Sie die Variable mit dem Schlüsselwort AT deklarieren. Der Vorteil einer solchen Vorgehensweise ist, dass man einer Adresse einen aussagekräftigeren Namen geben kann, und dass man eine eventuelle Veränderung eines Ein- oder Ausgangssignals nur an einer Stelle (nämlich in der Deklaration) zu ändern braucht.

Beachten Sie, dass man auf Variablen, die auf einen Eingang gelegt sind, nicht schreibend zugreifen kann. Als weitere Einschränkung gilt, dass AT-Deklarationen nur für lokale und globale Variablen gemacht werden können, also nicht für Ein- und Ausgabevariablen von Bausteinen.

Beispiele:

```
schalter_heizung7 AT %QX0.0: BOOL;
lichtschrankenimpuls AT %IX7.2: BOOL;
ablage AT %MX2.2: BOOL;
```

Wenn boolsche Variablen auf eine Byte-, Word- oder DWORD-Adresse gelegt werden, belegen sie 1 Byte mit TRUE bzw. FALSE, nicht nur das erste Bit nach dem Offset !

'Einfügen' 'Deklarations Schlüsselworte'

Mit diesem Befehl öffnen Sie eine Liste aller Schlüsselwörter, die im Deklarationsteil eines Bausteins benutzt werden können. Nachdem ein Schlüsselwort ausgewählt, und die Wahl bestätigt wurde, wird das Wort an der aktuellen Cursorposition eingefügt. Die Liste erhalten Sie auch, wenn Sie die Eingabehilfe aufrufen und die Kategorie Deklarationen auswählen.

'Einfügen' 'Typen'

Mit diesem Befehl erhalten Sie eine Auswahl der möglichen Typen für eine Variablendeklaration. Die Liste erhalten Sie auch, wenn Sie die Eingabehilfe <F2> aufrufen.

Die Typen sind eingeteilt in die Kategorien:

- Standard-Typen BOOL, BYTE etc.·
- Definierte Typen Strukturen, Aufzählungstypen etc.
- Standard-Funktionsblöcke für Instanzdeklarationen
- Definierte Funktionsblöcke für Instanzdeklarationen

TwinCAT PLC Control unterstützt sämtliche Standard-Typen der Norm IEC1131-3. Beispiele für die Verwendung der verschiedenen Typen befinden sich im Anhang.

Syntaxcoloring

In den Texteditoren und im Deklarationseditor werden Sie bei der Implementierung und der Variablendeklaration optisch unterstützt. Fehler werden vermieden bzw. schneller entdeckt, dadurch dass der Text farbig dargestellt wird.

Ein ungeschlossener Kommentar, der dadurch Anweisungen auskommentiert, wird sofort bemerkt; Schlüsselwörter werden nicht versehentlich falsch geschrieben usw.

Es gilt folgende Farbgebung:

Blau	Schlüsselwörter
Grün	Kommentare in den Texteditoren
Rosa	Spezielle Konstanten (z.B. TRUE/FALSE, T#3s, %IX0.0)
Rot	Fehlerhafte Eingabe (z.B. ungültige Zeitkonstante, Schlüsselwort klein geschrieben,...)
Schwarz	Variablen, Konstanten, Zuweisungsoperatoren, ...

Kurzformmodus

Der Deklarationseditor von TwinCAT PLC Control bietet Ihnen die Möglichkeit des Kurzformmodus. Dieser wird aktiviert, wenn Sie eine Zeile mit <Strg><Eingabetaste> beenden. Folgende Kurzformen werden unterstützt:· Alle Bezeichner bis auf den letzten Bezeichner einer Zeile werden zu Variablenbezeichnern der Deklaration. · Der Typ der Deklaration wird durch den letzten Bezeichner der Zeile bestimmt, hierbei gilt:

B oder BOOL	ergibt BOOL
I oder INT	ergibt INT
R oder REAL	ergibt REAL
S oder STRING	ergibt STRING

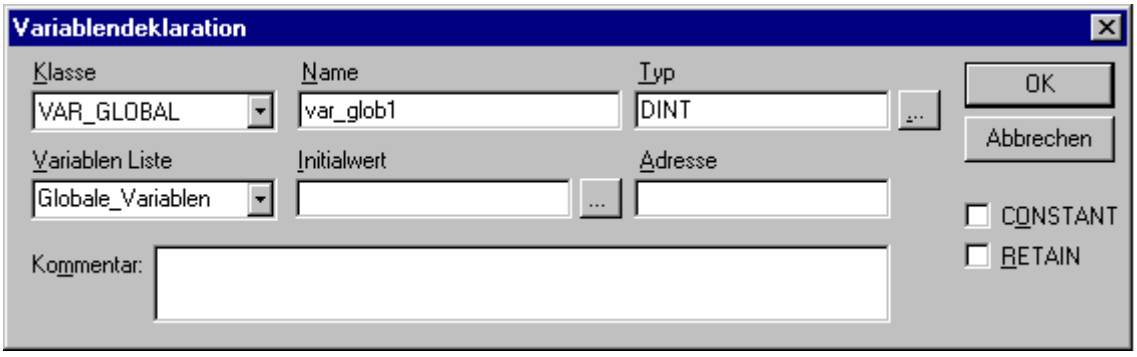
Wenn durch diese Regeln kein Typ festgelegt werden konnte, dann ist der Typ BOOL und der letzte Identifikator wird nicht als Typ benutzt (Beispiel 1.). Jede Konstante wird, je nach Typ der Deklaration, zu einer Initialisierung oder einer Stringlänge (Beispiele 2. und 3.). Eine Adresse (wie im %MD12) wird um das AT ... Attribut erweitert (Beispiel 4.). Ein Text nach einem Strichpunkt (;) wird ein Kommentar (Beispiel 4.). Alle anderen Zeichen in der Zeile werden ignoriert (wie z.B. das Ausrufezeichen in Beispiel 5.).

Beispiele:

Kurzform	Deklaration
A	A: BOOL;
A B I 2	A, B: INT := 2;
ST S 2; Ein String	ST: STRING(2); (* Ein String *)
X %MD12 R 5; Reelle Zahl	X AT %MD12: REAL := 5.0;(* Reelle Zahl *)
B !	B: BOOL;

Automatisch deklarieren

Wenn die Option Automatisch deklarieren in der Kategorie Editor des Optionsdialogs gewählt wurde, so erscheint in allen Editoren nach Eingabe einer noch nicht deklarierten Variablen ein Dialog, mit dessen Hilfe diese Variable deklariert werden kann.



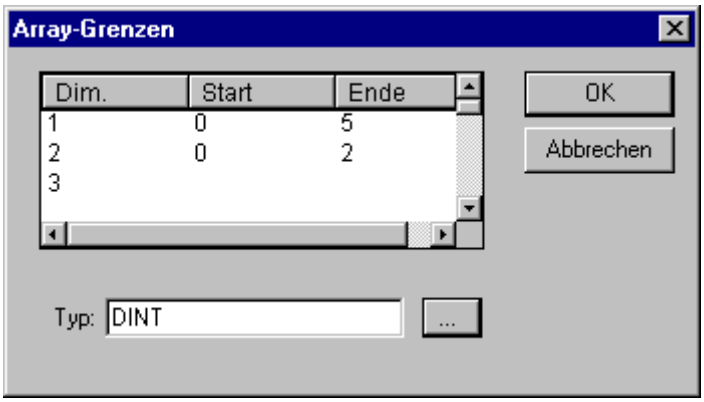
Dialog zur Variablendeklaration

Wählen Sie mit Hilfe der Combobox **Klasse**, ob es sich um eine lokale Variable (**VAR**) Eingabevariable (**VAR_INPUT**), Ausgabevariable (**VAR_OUTPUT**), EinAusgabevariable (**VAR_IN_OUT**) oder eine globale Variable (**VAR_GLOBAL**) handelt. Mit den Optionen **CONSTANT** und **RETAIN** können sie definieren, ob es sich um eine Konstante bzw. eine remanente Variable handelt.

Das Feld **Name** wurde mit dem im Editor eingegebenen Variablenname vorbelegt, das Feld Typ mit BOOL.

Mit der Schaltfläche ... erhalten Sie den Dialog der Eingabehilfe zur Auswahl aller möglichen Datentypen.

Wird als Typ der Variable ARRAY (Feld) ausgewählt, so erscheint der Dialog zur Eingabe der Array-Grenzen.



Deklarationseditor für Arrays

Für jede der drei möglichen Dimensionen (**Dim**.) können unter **Start** und **Ende** die Arraygrenzen eingegeben werden, indem durch Mausklick auf das entsprechende Feld ein Editierrahmen geöffnet wird. Im Feld **Typ** wird der Datentyp des Arrays angegeben. Über die Schaltfläche ... kann hierzu ein Eingabehilfedialog aufgerufen werden.

Beim Verlassen des Array-Grenzen-Dialogs über die Schaltfläche **OK** wird aus den Angaben das Feld 'Typ' im Dialog Variablendeklaration im IEC-Format belegt. Beispiel: ARRAY [1..5, 1..3] OF INT

Im Feld **Initialwert** können Sie den Initialwert der zu deklarierenden Variable eintragen. Ist diese ein Array oder eine gültige Struktur, können Sie über die Schaltfläche oder <F2> einen speziellen Initialisierungsdialog öffnen, bzw. bei anderen Typen den Eingabehilfedialog.

Im Initialisierungsdialog für ein Array erhalten Sie eine Auflistung der Array-Elemente und können mit Mausklick auf die Stelle hinter ":=" ein Editierfeld zum Eintragen des Initialwerts eines Elements öffnen.

Im Initialisierungsdialog für eine Struktur werden die einzelnen Komponenten in Baumstruktur dargestellt. In Klammern hinter dem Variablennamen stehen Typ und Default-Initialwert der Komponente, dahinter folgt jeweils ":=". Bei Mausklick auf das Feld hinter ":=" öffnet ein Editierfeld, in dem Sie den gewünschten Initialwert eingeben. Ist eine Komponente ein Array, können im Initialisierungsdialog durch Mausklick auf das Pluszeichen vor dem Array-Namen die einzelnen Felder des Arrays aufgeklappt und mit Initialwerten editiert werden.

Nach Verlassen des Initialisierungsdialogs mit OK erscheint im Feld Initialwert des Deklarationsdialogs die Initialisierung des Arrays bzw. der Struktur im IEC-Format.
Beispiel: x:=5,feld:=2,3,struct2:=(a:=2,b:=3)

Im Feld Adresse können Sie die zu deklarierende Variable an eine IEC-Adresse binden (AT-Deklaration).

Gegebenenfalls geben Sie einen Kommentar ein. Der Kommentar kann mittels der Tastenkombination <Strg>+<Eingabetaste> mit Zeilenumbrüchen versehen werden.
Durch Drücken von OK wird der Deklarationsdialog geschlossen und die Variable gemäß IEC-Syntax in den entsprechenden Deklarationseditor eingetragen.

Den Dialog zur Variablendeklaration erhalten Sie ebenfalls auf Anfrage über den Befehl 'Bearbeiten' 'Variablen' Deklaration'.

Steht der Cursor auf einer Variablen, kann im Offline Modus mit <Shift> <F2> das Autodeclare-Fenster mit den aktuellen variablenbezogenen Einstellungen geöffnet werden.

Zeilennummern im Deklarationseditor

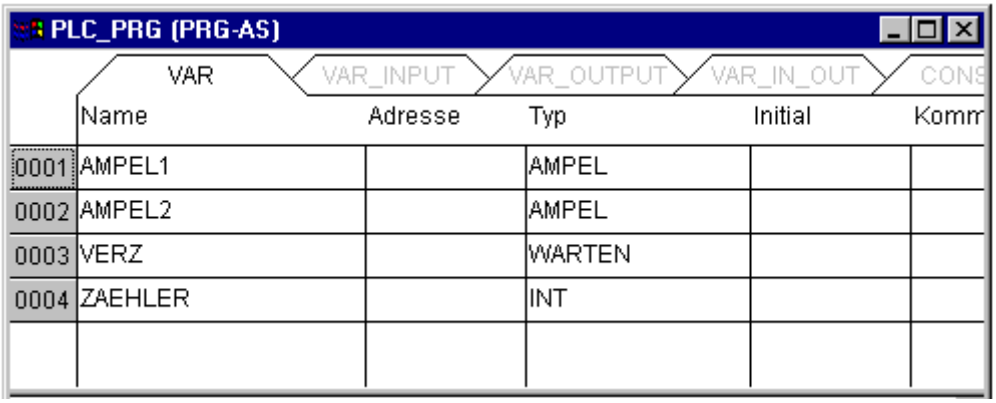
Im Offline Modus markiert ein einfacher Klick auf eine spezielle Zeilennummer die gesamte Textzeile. Im Online Modus lässt ein einzelner Klick auf eine bestimmte Zeilennummer die Variable in dieser Zeile auf- oder zuklappen, falls es sich um eine strukturierte Variable handelt.

Deklarationen als Tabelle

Ist die Option Deklarationen als Tabelle im Optiondialog in der Kategorie Editor eingestellt, erhalten Sie den Deklarationseditor in einer tabellarischen Darstellung. Wie in einem Karteikasten können Sie einzeln die Registerkarten der jeweiligen Variablenarten auswählen und die Variablen editieren. Für jede Variable erhalten Sie folgende Felder zum Eintrag:

Name	Geben Sie den Bezeichner der Variablen ein.
Adresse	Geben Sie gegebenenfalls die Adresse der Variablen ein (AT-Deklaration)
Typ	Geben Sie den Typ der Variablen ein. (Bei der Instanziierung eines Funktionsblocks, den Funktionsblock)
Initial	Geben Sie eine eventuelle Initialisierung der Variablen ein. (entsprechend dem Zuweisungsoperator " := ")
Kommentar	Geben Sie hier einen Kommentar ein.

Die beiden Darstellungsarten des Deklarationseditors können problemlos gewechselt werden. Im Online Modus gibt es für die Darstellung des Deklarationseditors keine Unterschiede. Um eine neue Variable zu editieren, führen Sie den Befehl 'Einfügen' 'Neue Deklaration' aus.



Deklarationseditor als Tabelle

'Einfügen' 'Neue Deklaration'

Mit diesem Befehl tragen Sie eine neue Variable in die Deklarationstabelle des Deklarationseditors ein. Befindet sich die aktuelle Cursorposition in einem Feld der Tabelle, wird die neue Variable vor dieser Zeile eingefügt, ansonsten ans Ende der Tabelle angefügt. Außerdem können Sie ans Ende der Tabelle eine neue Deklaration anfügen, indem Sie im letzten Feld der Tabelle die rechte Pfeiltaste oder die Tabulatortaste betätigen. Sie erhalten eine Variable, die als Vorbelegung im Feld **Name** 'Name', und im Feld **Typ** 'Bool' stehen hat. Diese Werte sollten Sie in die gewünschten Werte ändern. Name und Typ genügen für eine vollständige Variablendeklaration.

Pragma-Anweisung

Die Pragma-Anweisung dient zum Steuern des Übersetzungsvorgangs. Sie wird steht mit zusätzlichem Text in einer Programmzeile oder in einer eigenen Zeile des Deklarationseditors.

Die Pragma-Anweisung wird in geschweifte Klammern gefasst (Groß- oder Kleinschreibung wird nicht berücksichtigt):

{ <Anweisungstext> }

Kann der Compiler den Anweisungstext nicht sinnvoll interpretieren, so wird das gesamte Pragma wie ein Kommentar behandelt und überlesen. Es wird jedoch eine Warnung ausgegeben: "Ignoriere Compilerdirektive '<Anweisungstext>'!".

Abhängig vom Pragmatyp und -inhalt wirkt ein Prama auf die Zeile, in der es steht, bzw. auf alle folgenden Zeilen, bis es mit einem entsprechenden Pragma wieder aufgehoben wird oder bis dasselbe Pragma mit anderen Parametern ausgeführt oder das Ende der Datei erreicht wird. Als Datei wird hierbei verstanden: Deklarationsteil, Implementationsteil, Globale Variablenliste, Typdeklaration.

Die öffnende Klammer darf unmittelbar auf einen Variablennamen folgen. Öffnende und schließende Klammer müssen sich in derselben Zeile befinden.

```
Folgendes Pragma steht derzeit zur
Verfügung:
```

```
Pragma {flag [<flags>] [off|on]}
```

Mit diesem Pragma können die Eigenschaften einer Variablendeklaration beeinflusst werden.
<flags> kann eine Kombination der folgenden Flags sein:

noinit	Die Variable wird nicht initialisiert.
nowatch	Die Variable kann nicht gemonitort werden.
noread	Die Variable wird ohne Leserecht in die Symboldatei exportiert.
nowrite	Die Variable wird ohne Schreibrecht in die Symboldatei exportiert.
noread, nowrite	Die Variable wird nicht in die Symboldatei exportiert.

Mit der Modifikation "**on**" wirkt das Pragma auf alle folgenden Variablendeklarationen, bis es vom Pragma {flag off} aufgehoben wird, bzw. bis es von einem anderen {flag <flags> on}-Pragma überschrieben wird. Ohne die Modifikation mit "on" oder "off" wirkt das Pragma nur auf die aktuelle Variablendeklaration (das ist die Deklaration, die mit dem nächsten Strichpunkt abgeschlossen wird).
Beispiele:
Die Variable a wird nicht initialisiert und nicht gemonitort. Die Variable b wird nicht initialisiert:

```
VAR
    a : INT {flag noinit, nowatch};
    b : INT {flag noinit};
END_VAR
```

```
VAR
    {flag noint, nowatch on}
    a : INT {flag noinit on};
    b : INT;
    {flag off}
END_VAR
```

Beide Variablen werden nicht initialisiert:

```
{flag noinit on}
VAR
    a : INT;
    b : INT;
END_VAR
{flag off}
```

```
VAR
    {flag noinit on}
    a : INT;
    b : INT;
    {flag off}
END_VAR
```

Die Flags "noread" und "nowrite" dienen dazu, in einem Baustein, der Lese- und/oder Schreibrecht hat, einzelne Variablen mit einem eingeschränkten Zugriffsrecht zu versehen. Der Default für eine Variable ist die Einstellung, die der Baustein hat, in dem die Variable deklariert ist. Hat eine Variable weder Lese- noch Schreibrecht, dann wird sie nicht in die Symboldatei exportiert.

Beispiele:

Der Baustein wird mit Lese- und Schreibrecht versehen, dann kann mit den folgenden Pragmas Variable a nur mit Schreibrecht, Variable b überhaupt nicht exportieren:

```
VAR
    a : INT {flag noread};
    b : INT {flag noread, nowrite};
END_VAR
```

Beide Variablen a und b werden nicht in die Symboldatei exportiert:

```
{flag noread, nowrite on}
VAR
    a : INT;
    b : INT;
END_VAR
{flag off}
```

Das Pragma wirkt additiv auf alle untergeordneten Variablendeklarationen.

Beispiel: (alle verwendeten Bausteine werden mit Lese- und Schreibrecht exportiert)

```
a : afb;
...
FUNCTION_BLOCK afB
VAR
    b : bfb {flag nowrite};
    c : INT;
END_VAR
...
FUNCTION_BLOCK bfB
VAR
    d : INT {flag noread};
    e : INT {flag nowrite};
END_VAR
```

"a.b.d": Wird nicht exportiert.
"a.b.e": Wird nur mit Leserecht exportiert.
"a.c": Wird mit Lese- und Schreibrecht exportiert.

Pragma für Anzeige/Nicht-Anzeige von Deklarationsteilen im Bibliotheksverwalter

Mittels der Pragmas {library public} und {library private} kann in einer in TwinCAT-SPS erstellten Bibliothek definiert werden, welche Zeilen/Zeilenteile des Deklarationsteils später bei der Verwendung der Bibliothek in einem Projekt im Bibliotheksverwalter angezeigt bzw. nicht angezeigt werden sollen. Die Darstellung des Implementationsteils bleibt davon unbeeinflusst. Damit können beispielsweise Kommentare oder bestimmte Variablendeklarationen der Bibliothek für den Benutzer unsichtbar gemacht werden. Die Pragmas gelten jeweils für den Rest derselben bzw. die nachfolgenden Zeilen, solange, bis sie durch das jeweils andere Pragma aufgehoben werden.

Syntax:

{library public} Der nachfolgende Text wird im Bibliotheksverwalter angezeigt.
{library private} Der nachfolgende Text wird nicht angezeigt.

Beispiel: Sehen Sie unten den Deklarationsteil einer Bibliothek, die in TwinCAT SPS erstellt wird. Der Kommentar "(* this is for all *)" soll nach dem Einbinden der Bibliothek im Bibliotheksverwalter angezeigt werden, "(* but this is not for all ")" dagegen nicht. Die Variablen *local* und *in3* sollen ebenfalls nicht sichtbar sein:

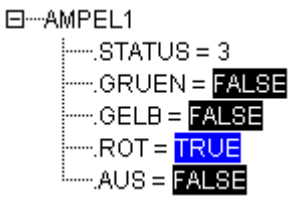
```
{library public}
    (*this is for all*)
{library private}
    (*this is not for all*)
{library public}
FUNCTION afun   : BOOL
VAR_INPUT
    in      : BOOL;
END_VAR
{library private}
VAR
    local   : BOOL;
END_VAR
{library public}
VAR_INPUT
    in2     : BOOL;
{library private}
```

```
    in3     : BOOL;
{library public}
END_VAR
```

Deklarationseditoren im Online Modus

Im Online Modus wird der Deklarationseditor zu einem Monitor-Fenster. In jeder Zeile steht eine Variable, gefolgt von einem Gleichheitszeichen (=) und dem Wert der Variablen. Wenn die Variable zu diesem Zeitpunkt undefiniert ist, erscheinen drei Fragezeichen (???). Bei Funktionsblöcken werden nur für geöffnete Instanzen (Befehl 'Projekt' 'Instanz öffnen') die Werte angezeigt.

Vor jeder mehrelementigen Variablen steht eine Pluszeichen. Durch das Drücken der <Eingabetaste> oder nach einem Doppelklick auf eine solche Variable klappt diese auf, im Beispiel wäre die Struktur Ampel1 aufgeklappt:



Bei einer aufgeklappten Variablen werden alle ihre Komponenten nachfolgend aufgelistet. Vor der Variablen erscheint ein Minuszeichen. Mit einem weiteren Doppelklick bzw. durch Drücken der <Eingabetaste> klappt die Variable zu und das Pluszeichen erscheint erneut.

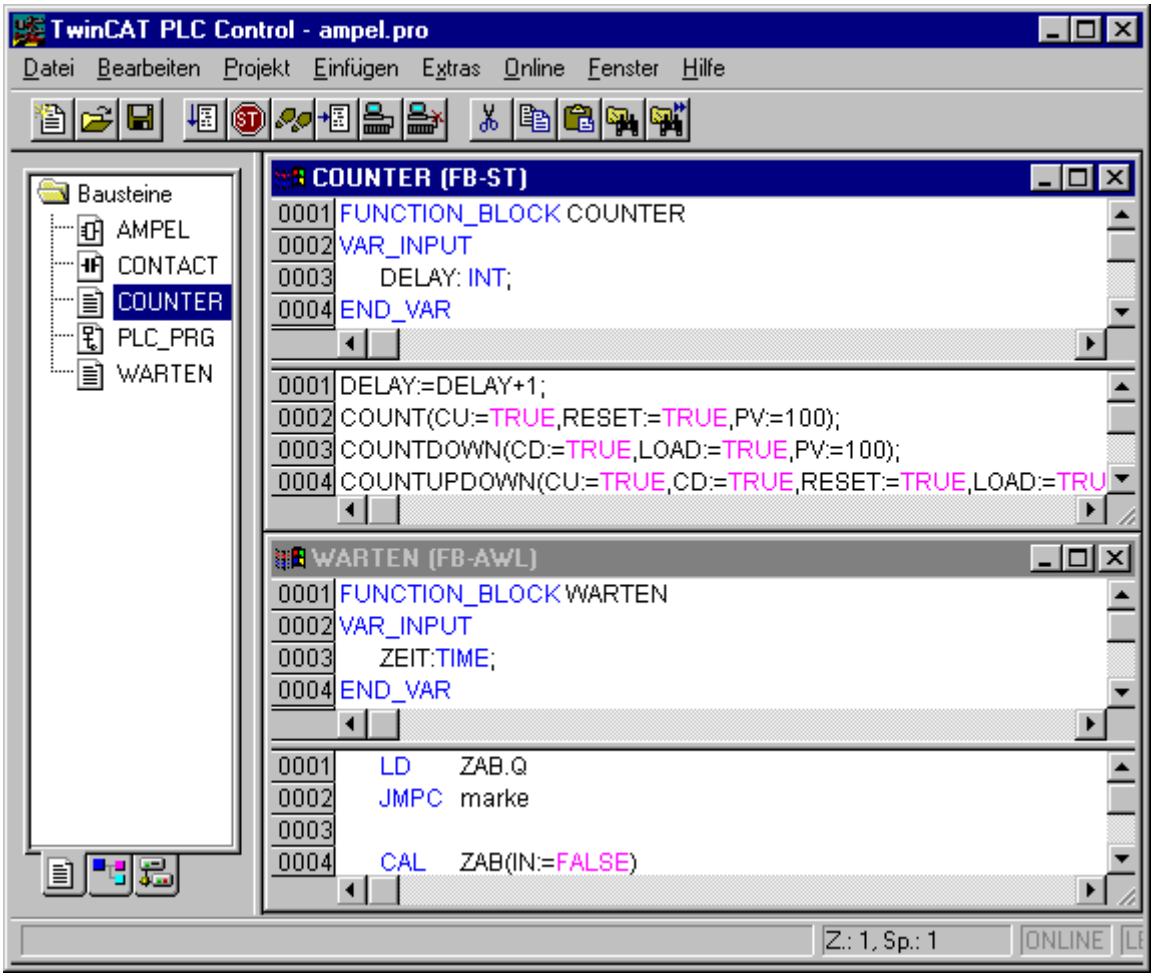
Durch Drücken der <Eingabetaste> oder einen Doppelklick auf eine einelementige Variable öffnet den Dialog zum Schreiben einer Variablen. Hier ist es möglich, den aktuellen Wert der Variablen zu ändern. Bei boolschen Variablen erscheint kein Dialog, sie werden getoggelt.

Der neue Wert wird hinter der Variable türkisfarben in spitzen Klammern angezeigt und bleibt unverändert. Wenn der Befehl '**Online''Werte schreiben**' gegeben wird, dann werden alle Variablen auf die gewählten Werte gesetzt und wieder schwarz dargestellt. Wenn der Befehl 'Online''Werte forcen' gegeben wird, dann werden alle Variablen auf die gewählten Werte gesetzt, bis der Befehl '**Forcen aufheben**' erfolgt. In diesem Fall wechselt die Farbe des Force-Wertes auf rot.

5.2 Texteditoren

Die Texteditoren (der Anweisungslisteneditor [▶ 135] und der Editor für Strukurierten Text [▶ 136]) von TwinCAT PLC Control verfügen über die üblichen Funktionalitäten von Windows Texteditoren. Die Implementierung in den Texteditoren wird durch Syntaxcoloring unterstützt.

Wenn Sie im Überschreibmodus arbeiten, wird in der Statusleiste 'ÜB' schwarz angezeigt. Sie können zwischen dem Überschreib- und dem Einfügemodus wechseln, durch Betätigen der Taste <Einfg>.



Texteditoren für Anweisungsliste und Strukturierter Text

Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste oder <Strg>+<F10>. Folgende Menübefehle benutzen speziell die Texteditoren:

'Einfügen''Operator'

Mit diesem Befehl werden in einem Dialog alle Operatoren angezeigt, die in der aktuellen Sprache verfügbar sind. Wird einer der Operatoren ausgewählt, und die Liste mit **OK** geschlossen, dann wird der markierte Operator an die aktuelle Cursorposition eingefügt.
'Einfügen''Operand'

Mit diesem Befehl werden in einem Dialog alle Variablen angezeigt. Sie können wählen, ob Sie eine Liste der globalen, der lokalen oder der Systemvariablen dargestellt haben wollen. Wird einer der Operanden ausgewählt, und der Dialog mit **OK** geschlossen, dann wird der markierte Operand an die aktuelle Cursorposition eingefügt.
'Einfügen''Funktion'

Mit diesem Befehl werden in einem Dialog alle Funktionen angezeigt. Sie können wählen, ob Sie eine Liste der benutzerdefinierten oder der Standardfunktionen dargestellt haben wollen. Wird eine der Funktionen ausgewählt, und der Dialog mit OK geschlossen, dann wird die markierte Funktion an der aktuellen Cursorposition eingefügt. Wurde im Dialog die Option **Mit Argumenten** angewählt, so werden die erforderlichen Eingabevariablen der Funktion mit eingefügt.
'Einfügen''Funktionsblock'

Mit diesem Befehl werden in einem Dialog alle Funktionsblöcke angezeigt. Sie können wählen, ob Sie eine Liste der benutzerdefinierten oder der Standardfunktionsblöcke dargestellt haben wollen. Wird einer der Funktionsblöcke ausgewählt, und der Dialog mit OK geschlossen, dann wird der markierte Funktionsblock an die aktuelle Cursor-position eingefügt. Wurde im Dialog die Option Mit Argumenten angewählt, so werden die erforderlichen Eingabevariablen des Funktionsblocks mit eingefügt.

Bausteinaufruf mit Ausgangsparametern

Die Ausgangsparameter eines aufgerufenen Bausteins können in den textuellen Sprachen AWL und ST bereits direkt im Aufruf zugewiesen werden.

Beispiel: Ausgangsparameter out1 von afbinst wird Variable a zugewiesen.

```
AWL: CAL afbinst(in1:=1, out1=>a)
ST: afbinst(in1:=1, out1=>a);
```

Die Texteditoren im Online Modus

Die Onlinefunktionen in den Editoren sind Breakpoint Setzen und Einzelschrittabarbeitung (Steppen). Zusammen mit dem Monitoring hat der Anwender so die Debugging-Funktionalität eines modernen Windows-Hochsprachendebuggers.

Im Online Modus wird das Texteditor-Fenster vertikal zweigeteilt. Auf der linken Seite des Fensters befindet sich dann der normale Programmtext, auf der rechten Seite finden Sie die Variablen dargestellt, deren Werte in der jeweiligen Zeile geändert werden.

Die Darstellung ist dieselbe, wie im Deklarationsteil. D.h. wenn die Steuerung läuft, dann werden die momentanen Werte der jeweiligen Variablen dargestellt.

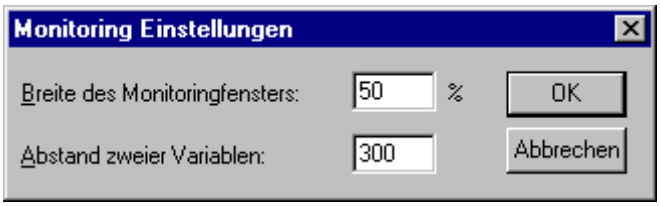
Beim Monitoring von Ausdrücken oder Bit-adressierten Variablen ist folgendes zu beachten: Bei Ausdrücken wird stets der Wert des gesamten Ausdrucks dargestellt.

Beispiel: a AND b wird als blau bzw. mit ":=TRUE" angezeigt, wenn a und b TRUE sind). Bei Bit-adressierten Variablen wird immer der angesprochene Bit-Wert gemonitort (z.B. wird a.3 blau bzw. mit :=TRUE dargestellt, wenn a den Wert 4 hat).

Wenn Sie den Mauszeiger eine kurze Zeit über einer Variablen halten, wird der Typ, die Adresse und der Kommentar der Variablen in einem Tooltip angezeigt.

'Extras' 'Monitoring Einstellungen'

Mit diesem Befehl können Sie Ihr Monitoring-Fenster konfigurieren. In den Texteditoren wird beim Monitoring das Fenster aufgeteilt in eine linke Hälfte, in der das Programm steht, und eine rechte Hälfte, in der alle Variablen, die in der entsprechenden Programmzeile stehen, gemonitort werden. Sie können einstellen, welche Breite der Monitoring-Bereich im Textfenster bekommen soll, und welchen Abstand zwei Monitoring-Variablen in einer Zeile haben sollen. Die Abstandsangabe 1 entspricht dabei einer Zeilenhöhe in der gewählten Schriftart.



Monitoring Einstellungen-Dialog

Breakpointpositionen im Texteditor

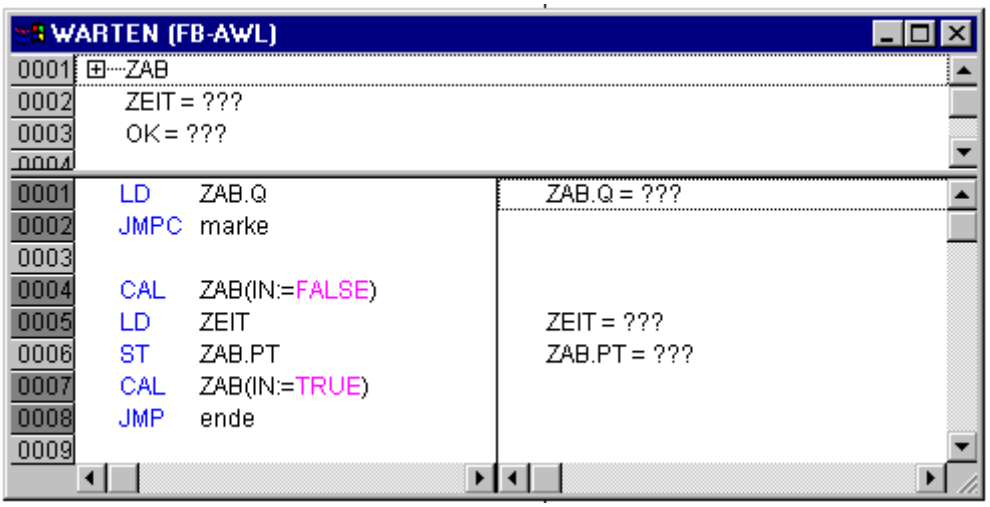
Da intern in TwinCAT PLC Control mehrere AWL-Zeilen zu einer C-Code-Zeile zusammengefaßt werden, können nicht in jeder Zeile Breakpoints gesetzt werden. Breakpointpositionen sind alle Stellen im Programm, an denen sich Variablenwerte ändern können oder an denen der Programmfluß verzweigt (Ausnahme: Funktionsaufrufe. Hier muss gegebenenfalls ein Breakpoint in der Funktion gesetzt werden). An den dazwischenliegenden Positionen ist ein Breakpoint auch nicht sinnvoll, da sich an den Daten seit der vorhergehenden Breakpointposition nichts geändert haben kann. Damit ergeben sich folgende Breakpointpositionen in der AWL:

- Am Anfang des Bausteins
- Auf jedem LD, LDN (oder falls ein LD direkt nach einer Marke steht, auf dieser)
- Bei jedem JMP, JMPC, JMPCN· Bei jeder Marke
- Bei jedem CAL, CALC, CALCN
- Bei jedem RET, RETC, RETCN
- Am Ende des Bausteins

Für Strukturierten Text ergeben sich folgende Breakpointpositionen:

- Bei jeder Zuweisung
- Bei jeder RETURN und EXIT-Anweisung
- in Zeilen, in denen Bedingungen ausgewertet werden (WHILE, IF, REPEAT)
- Am Ende des Bausteins

Breakpoint Positionen sind dadurch gekennzeichnet, dass das Zeilennummernfeld in einem dunkleren Grau dargestellt ist.



AWL-Editor mit möglichen Breakpointpositionen (dunklere Nummernfelder)

Wie setzt man einen Breakpoint?

Um einen Breakpoint zu setzen, klickt der Anwender mit der Maus das Zeilennummernfeld der Zeile an, in der er den Breakpoint setzen möchte. Ist das ausgewählte Feld eine Breakpointposition, so wechselt die Farbe des Zeilennummernfeldes von dunkelgrau nach hellblau und der Breakpoint wird in der Steuerung aktiviert.

Löschen von Breakpoints

Entsprechend wird, um einen Breakpoint zu löschen, das Zeilennummernfeld der Zeile mit dem zu löschenden Breakpoint angeklickt. Setzen und Löschen von Breakpoints kann auch über Menü (**'Online' 'Breakpoint an/aus'),** über Funktionstaste <F9> oder das Symbol in der Funktionsleiste ausgewählt werden.

Was passiert an einem Breakpoint?

Ist in der Steuerung ein Breakpoint erreicht, so wird am Bildschirm der Ausschnitt mit der entsprechenden Zeile dargestellt. Das Zeilennummernfeld der Zeile, in der die Steuerung steht, ist rot.

In der Steuerung stoppt die Bearbeitung des Anwenderprogramms. Steht das Programm auf einem Breakpoint, so kann die Bearbeitung mit **'Online' 'Start'** fortgesetzt werden.

Außerdem kann mit **'Online' 'Einzelschritt über'** bzw. **'Einzelschritt in'** nur bis zur nächsten Breakpointposition gegangen werden. Ist die Anweisung, auf der man steht, ein CAL-Befehl oder steht in den Zeilen bis zur nächsten Breakpointposition ein Funktionsaufruf, so wird dieser mit **'Einzelschritt über'** übersprungen, mit **'Einzelschritt in'** wird in den aufgerufenen Baustein verzweigt.

Zeilennummern des Texteditors

Die Zeilennummern des Texteditors geben die Nummer jeder Textzeile einer Implementierung eines Bausteins an. Im Offline Modus markiert ein einfacher Klick auf eine spezielle Zeilennummer die gesamte Textzeile. Im Online Modus zeigt die Hintergrundfarbe der Zeilennummer den Breakpoint-Zustand jeder Zeile an:

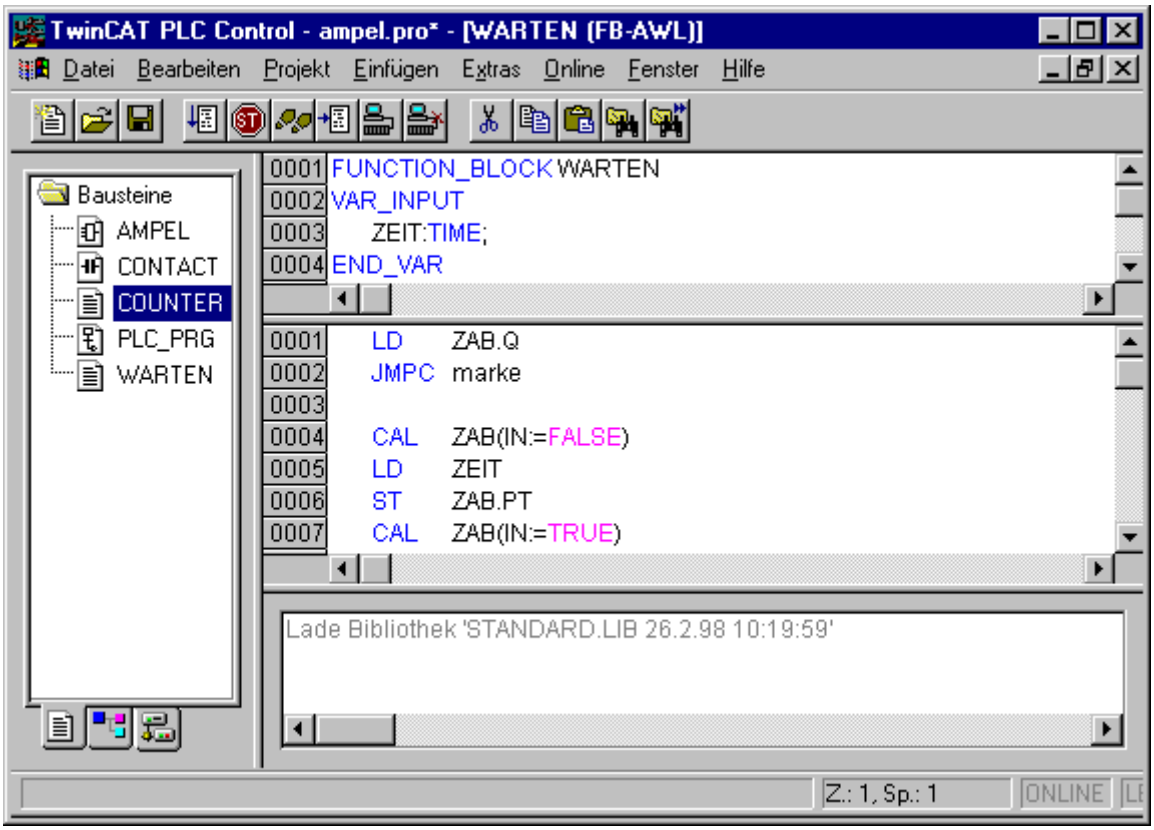
- dunkelgrau: Diese Zeile ist eine mögliche Position für einen Breakpoint.
- hellblau: in dieser Zeile wurde ein Breakpoint gesetzt.

- rot: die Programmabarbeitung befindet sich an diesem Punkt.

Im Online Modus wechselt ein einfacher Mausklick den Breakpointzustand dieser Zeile.

5.3 Anweisungslisteneditor

So sieht ein in AWL geschriebener Baustein unter dem entsprechenden TwinCAT PLC Control-Editor aus:



AWL-Editor

Alle Editoren für Bausteine bestehen aus einem Deklarationsteil und einem Rumpf. Diese sind getrennt durch einen Bildschirmteiler.

Der Anweisungslisteneditor ist ein Texteditor mit den üblichen Funktionalitäten von Windows Texteditoren. Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste oder <Strg>+<F10>.

Ein mehrzeiliger Bausteinaufruf ist möglich.

Beispiel:

```
CAL CTU_inst(
CU:=%IX10,
PV:=(
LD A
ADD 5
)
)
```

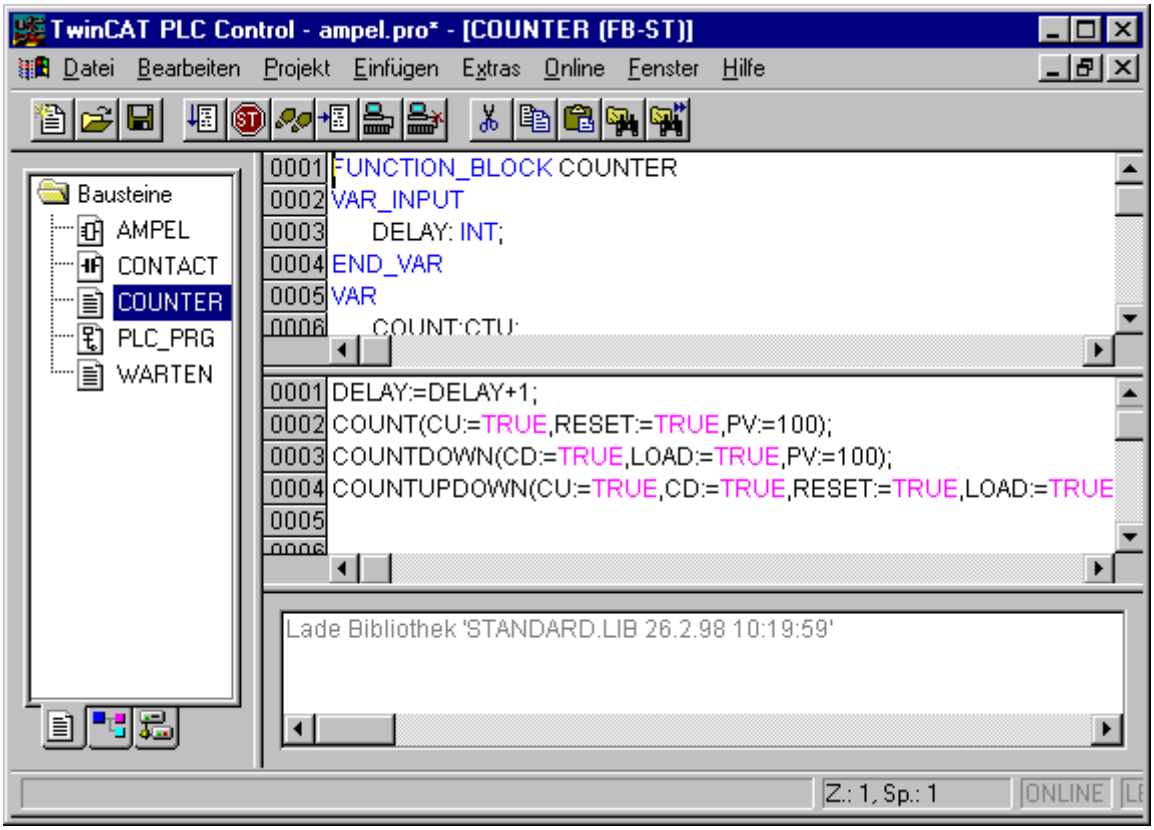
AWL im Online Modus

Mit dem Befehl **'Online' 'Ablaufkontrolle'** wird im AWL-Editor auf der linken Seite jeder Zeile ein weiteres Feld eingefügt, in dem der Akkumulatorinhalt dargestellt wird.

Zu weiteren Informationen über den AWL-Editor im Online Modus siehe oben 'Die Texteditoren im Online Modus'.

5.4 Strukturierter Text Editor

So sieht ein in ST geschriebener Baustein unter dem entsprechenden TwinCAT PLC Control-Editor aus:



Editor für Strukturierten Text

Alle Editoren für Bausteine bestehen aus einem Deklarationsteil und einem Rumpf. Diese sind getrennt durch einen Bildschirmteiler.

Der Editor für Strukturierten Text ist ein Texteditor mit den üblichen Funktionalitäten von Windows Texteditoren. Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste.

Für Informationen über die Sprache, lesen Sie das Kapitel Strukturierter Text (ST).

5.5 Graphische Editoren

Die Editoren der beiden graphisch orientierten Sprachen Ablaufsprache AS [▶ 160], KOP [▶ 142] und FUP [▶ 138] und freigraphischer Funktionsplan CFC [▶ 147] haben viele Gemeinsamkeiten. In den nachfolgenden Abschnitten werden diese Punkte zusammengefasst. Die Implementierung in den grafischen Editoren wird durch Syntaxcoloring unterstützt.

Zoom

Objekte wie Bausteine, Aktionen, Transitionen etc. in den Sprachen AS, KOP, FUP und CFC können mit einer Zoom-Funktion vergrößert oder verkleinert werden. Dabei werden alle Elemente des Fensterinhalts des Implementationsteils erfasst, der Deklarationsteil bleibt unverändert.

Standardmäßig wird jedes Objekt mit der Zoomstufe 100% angezeigt. Die eingestellte Zoomstufe wird als Objekteigenschaft im Projekt abgespeichert.

Das Ausdrucken der Projektdokumentation erfolgt immer in der Darstellung 100% !
Die Zoomstufe kann über eine Auswahlliste in der Symbolleiste eingestellt werden. Es sind Werte zwischen 25% und 400% auswählbar, individuelle Werte zwischen 10% und 500% können manuell eingegeben werden.

Die Zoomstufen-Auswahl steht nur zur Verfügung, wenn der Cursor in einem in einer graphischen Sprache erstellten Objekt oder in einem Visualisierungsobjekt steht.

Die Cursorpositionen in den Editoren können auch im gezoomten Zustand des Objekts weiterhin selektiert und auch mit den Pfeiltasten erreicht werden. Die Textgröße richtet sich nach dem Zoomfaktor und der eingestellten Schriftgröße.

Die Ausführung aller Menüpunkte zur Bedienung des Editors (z.B. Einfügen einer Box) entsprechend der Cursorposition ist bei jeder Zoomstufe und unter Beibehaltung dieser möglich.

Im Online Modus wird jedes Objekt entsprechend der eingestellten Zoomstufe dargestellt, die Online-Funktionalitäten sind uneingeschränkt verfügbar.
Bei Verwendung der IntelliMouse kann ein Objekt vergrößert/verkleinert werden, indem die <STRG>-Taste gedrückt und gleichzeitig das Rad vorwärts/rückwärts gedreht wird.

Netzwerk

In den Editoren KOP und FUP wird das Programm in einer Liste von Netzwerken angeordnet. Jedes Netzwerk ist auf der linken Seite mit einer fortlaufenden Netzwerknummer gekennzeichnet und enthält eine Struktur, die jeweils einen logischen bzw. arithmetischen Ausdruck, einen Programm-, Funktions- oder Funktionsblockaufruf, einen Sprung oder eine Return-Anweisung darstellt.

Sprungmarken

Zu jedem Netzwerk gehört eine Sprungmarke, die wahlweise auch leer sein kann. Diese Marke wird editiert, indem man in die erste Zeile des Netzwerks, unmittelbar neben die Netzwerknummer klickt. Jetzt kann man eine Marke gefolgt von einem Doppelpunkt eingeben.

Netzwerkkommentare

Zu jedem Netzwerk kann ein mehrzeiliger Kommentar vergeben werden. In **'Extras' 'Optionen'** kann die Anzahl der Zeilen, die maximal für einen Netzwerkkommentar zur Verfügung stehen sollen, im Feld **Maximale Kommentargröße** eingegeben werden (der voreingestellte Wert ist hier 4), sowie die Zahl der Zeilen, die generell für Kommentare freigelassen werden sollen (**Minimale Kommentargröße**). Ist hier z.B. 2 eingestellt, so stehen an jedem Netzwerkanfang nach der Labelzeile zwei leere Kommentarzeilen. Der Vorgabewert ist hier 0, was den Vorteil hat, dass mehr Netzwerke in den Bildschirmbereich passen.

Ist die minimale Kommentargröße größer als 0, so kann, um einen Kommentar einzugeben, einfach in die Kommentarzeile geklickt und der Kommentar eingegeben werden. Andernfalls muss zunächst das Netzwerk, zu dem ein Kommentar eingegeben werden soll, ausgewählt und mit **'Einfügen' 'Kommentar'** eine Kommentarzeile eingefügt werden. Kommentare werden im Unterschied zu Programmtext grau dargestellt.

Im **Kontaktplaneditor** können außerdem Kommentare für einzelne Kontakte und Spulen vergeben werden. Dazu wird die Option Kommentare pro Kontakt aktiviert und die gewünschte Anzahl Zeilen, die dafür vorgesehen und angezeigt werden soll, im Feld Zeilen für Variablenkommentar eingegeben. Daraufhin erscheint ein Kommentarfeld über dem Kontakt bzw. der Spule und es kann Text eingetragen werden.

Wenn die Option Kommentare pro Kontakt aktiviert ist, kann im Kontaktplaneditor außerdem die Anzahl der Zeilen (**Zeilen für Variablentext:**) definiert werden, die für den Variablennamen des Kontakts bzw. der Spule verwendet wird, damit auch lange Namen durch die Verwendung von mehreren Zeilen komplett dargestellt werden können.

Für den Kontaktplaneditor kann eingestellt werden, daß die Netzwerke mit Umbrüchen versehen werden sollen, sobald die eingestellte Fensterbreite dazu führt, daß nicht mehr alle Elemente eines Netzwerks sichtbar sind.

'Einfügen' 'Netzwerk (danach)' oder 'Einfügen' 'Netzwerk (davor)' Kurzform: <Umschalt>+<T> (Netzwerk danach)

Um ein neues Netzwerk im FUP- oder KOP-Editor einzufügen, wählt man den Befehl **'Einfügen' 'Netzwerk (danach)'** oder **'Einfügen' 'Netzwerk (davor)',** je nachdem, ob man das neue Netzwerk vor oder nach dem aktuellen Netzwerk einfügen will. Das aktuelle Netzwerk ändert man durch Mausklick auf die Netzwerknummer. Man erkennt es am gepunkteten Rechteck unter der Nummer. Mit der <Umschalttaste> und Mausklick wird der ganze Bereich von Netzwerken zwischen dem aktuellen und dem angeklickten Netzwerk ausgewählt.

Die Netzwerkeditoren im Online Modus

In den Editoren FUP und KOP können Breakpoints nur auf Netzwerke gesetzt werden. Das Netzwerknummernfeld eines Netzwerks, auf das ein Breakpoint gesetzt wurde, wird blau dargestellt. Die Bearbeitung stoppt dann vor dem Netzwerk, auf dem der Breakpoint steht. In diesem Fall wird das Netzwerknummernfeld rot dargestellt. Bei der Einzelschrittabarbeitung (Steppen) wird von Netzwerk zu Netzwerk gesprungen.

Alle Werte werden an den Ein- und Ausgängen der Netzwerkbausteine gemonitort.

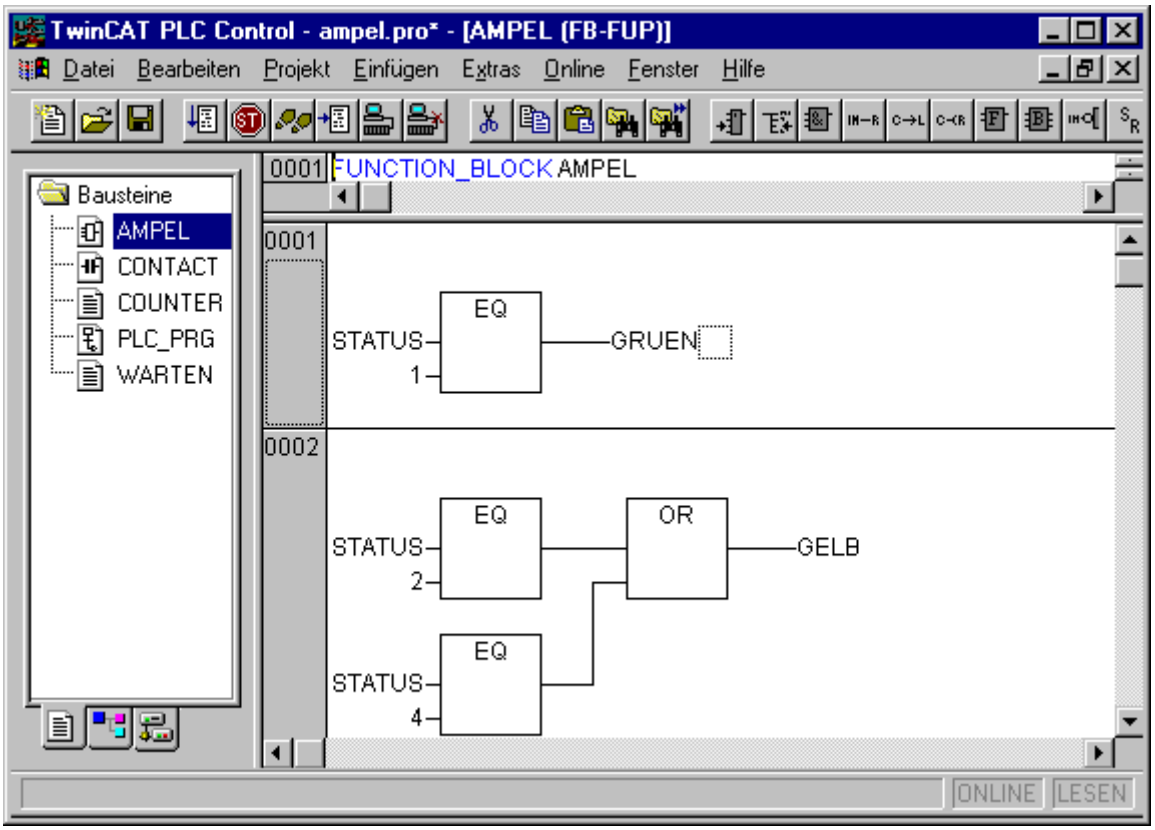
Beim Monitoring von Ausdrücken oder Bit-adressierten Variablen ist folgendes zu beachten: Bei Ausdrücken, z.B. a AND b als Transitionsbedingung oder Funktionsblockseingang, wird stets der Wert des gesamten Ausdrucks dargestellt (a AND b wird als blau bzw. mit :=TRUE angezeigt, wenn a und b TRUE sind). Bei Bit-adressierten Variablen wird immer der angesprochene Bit-Wert gemonitort (z.B. wird a.3 blau bzw. mit :=TRUE dargestellt, wenn a den Wert 4 hat).

Die Ablaufkontrolle starten Sie mit dem Menübefehl **'Online' 'Ablaufkontrolle'**. Mit ihr können Sie die aktuellen Werte, die in den Netzwerken über die Verbindungslinien transportiert werden, einsehen. Wenn die Verbindungslinien keine boolschen Werte transportieren, dann wird der Wert in einem extra eingefügten Feld angezeigt. Die Monitorfelder für Variablen, die nicht verwendet werden (z.B. bei der Funktion SEL) werden grau schattiert dargestellt. Wenn die Linien boolsche Werte transportieren, dann werden sie, für den Fall, dass sie TRUE transportieren, blau eingefärbt. So kann der Informationsfluss während des Steuerungslaufs mitverfolgt werden.

Wenn Sie den Mauszeiger eine kurze Zeit über einer Variablen halten, wird der Typ, die Adresse und der Kommentar der Variablen in einem Tooltip angezeigt.

5.6 Funktionsplaneditor

So sieht ein in FUP geschriebener Baustein unter dem entsprechenden TwinCAT PLC Control-Editor aus:

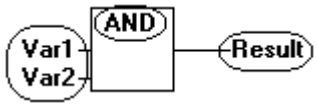


Editor für Funktionsplan
Der Funktionsplaneditor ist ein graphischer Editor. Er arbeitet mit einer Liste von Netzwerken, wobei jedes Netzwerk eine Struktur enthält, die jeweils einen logischen bzw. arithmetischen Ausdruck, den Aufruf eines Funktionsblocks, einer Funktion, eines Programms, einen Sprung oder eine Return-Anweisung darstellt. Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste).

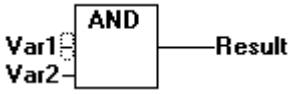
Cursorpositionen im FUP

Jeder Text ist eine mögliche Cursorposition. Der selektierte Text ist blau hinterlegt und kann nun geändert werden. Ansonsten ist die aktuelle Cursorposition durch ein gepunktetes Rechteck gekennzeichnet. Es folgt eine Aufzählung aller möglichen Cursorpositionen mit einem Beispiel:

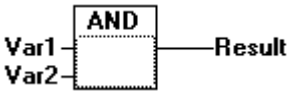
1) Jedes Textfeld (mögliche Cursorpositionen schwarz umrahmt):



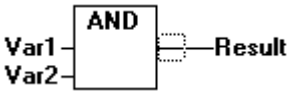
2) Jeder Eingang:



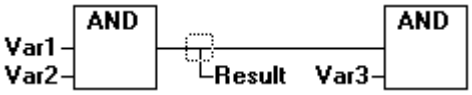
3) Jeder Operator, Funktion oder Funktionsbaustein:



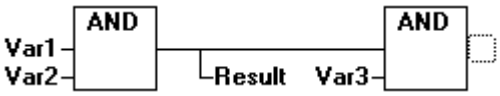
4) Ausgänge, wenn danach eine Zuweisung oder ein Sprung kommt:



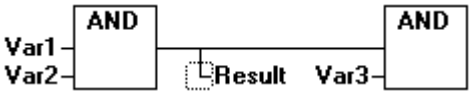
5) Das Linienkreuz über einer Zuweisung, einem Sprung oder einer Return-Anweisung:



6) Hinter dem äußerst rechten Objekt eines jeden Netzwerkes ("letzte Cursorposition", dies ist auch die Cursorposition, wenn ein Netzwerk selektiert wurde):



7) Das Linienkreuz unmittelbar vor einer Zuweisung:



Wie man den Cursor setzt

Der Cursor kann durch Klicken der Maus oder mit Hilfe der Tastatur auf eine bestimmte Position gesetzt werden. Mit den Pfeiltasten wird jeweils zur nächstliegenden Cursorposition in der ausgewählten Richtung gesprungen. Alle Cursorpositionen, einschließlich der Textfelder, können so erreicht werden. Ist die letzte Cursorposition selektiert, so kann mit den Pfeiltasten <nach oben> bzw. <nach unten> die letzte Cursorposition des vorhergehenden bzw. nachfolgenden Netzwerkes selektiert werden. Ein leeres Netzwerk enthält nur drei Fragezeichen "???". Durch Klicken hinter diese wird die letzte Cursorposition selektiert.

'Einfügen' 'Zuweisung' Kurzform: <Strg>+<A>

Dieser Befehl fügt eine Zuweisung ein. Eingefügt wird, je nach selektierter Position, unmittelbar vor dem selektierten Eingang (Cursorposition 2), unmittelbar nach dem selektierten Ausgang (Cursorposition 4), unmittelbar vor dem selektierten Linienkreuz (Cursorposition 5) oder am Ende des Netzwerkes (Cursorposition 6). Zu einer eingefügten Zuweisung kann anschließend der eingetragene Text "???" selektiert und durch die Variable, an die zugewiesen werden soll, ersetzt werden. Dazu können Sie auch die Eingabehilfe verwenden. Um zu einer existierenden Zuweisung eine weitere Zuweisung hinzuzufügen, benutzen Sie den Befehl 'Einfügen' 'Ausgang'.

'Einfügen' 'Sprung' Kurzform: <Strg>+<L>

Dieser Befehl fügt einen Sprung ein. Eingefügt wird, je nach selektierter Position, unmittelbar vor dem selektierten Eingang (Cursorposition 2), unmittelbar nach dem selektierten Ausgang (Cursorposition 4), unmittelbar vor dem selektierten Linienkreuz (Cursorposition 5) oder am Ende des Netzwerkes (Cursorposition 6). Zu einem eingefügten Sprung kann anschließend der eingetragene Text "???" selektiert und durch die Sprungmarke, an die gesprungen werden soll, ersetzt werden.

'Einfügen' 'Return' Kurzform: <Strg>+<R>

Dieser Befehl fügt eine RETURN-Anweisung ein. Eingefügt wird, je nach selektierter Position, unmittelbar vor dem selektierten Eingang (Cursorposition 2), unmittelbar nach dem selektierten Ausgang (Cursorposition 4), unmittelbar vor dem selektierten Linienkreuz (Cursorposition 5) oder am Ende des Netzwerkes (Cursorposition 6).

'Einfügen' 'Baustein' Kurzform: <Strg>+<B>

Mit diesem Befehl können Operatoren, Funktionen, Funktionsblöcke und Programme eingefügt werden. Es wird zunächst stets ein "AND"-Operator eingefügt. Dieser kann durch Selektieren und Überschreiben des Typ-Textes ("AND") in jeden anderen Operator, in jede Funktion, in jeden Funktionsblock und in jedes Programm umgewandelt werden. Mit der Eingabehilfe (<F2>) können Sie den gewünschten Baustein auswählen. Hat der neugewählte Baustein eine andere Mindestanzahl von Eingängen, so werden diese angehängt. Hat der neue Baustein eine kleinere Höchstzahl von Eingängen, so werden die letzten Eingänge gelöscht.

Bei Funktionen und Funktionsblöcken werden die formalen Namen der Ein- und Ausgänge angezeigt.

Bei Funktionsblöcken existiert ein editierbares Instanz-Feld über der Box. Wird durch Ändern des Typ-Textes ein anderer Funktionsblock aufgerufen, der nicht bekannt ist, wird eine Operator-Box mit zwei Eingängen und dem angegeben Typ angezeigt. Ist das Instanz-Feld angewählt, kann über <F2> die Eingabehilfe mit den Kategorien zur Variablenauswahl aufgerufen werden.

Der neue Baustein wird abhängig von der selektierten Position (siehe Cursorpositionen) eingefügt:

- Ist ein Eingang selektiert (Cursorposition 2), so wird der Baustein vor diesem Eingang eingefügt. Der erste Eingang dieses Bausteins wird mit dem Zweig links vom selektierten Eingang verbunden. Der Ausgang des neuen Bausteins wird mit dem selektierten Eingang verbunden.
- Ist ein Ausgang selektiert (Cursorposition 4), dann wird der Baustein nach diesem Ausgang eingefügt. Der erste Eingang des Bausteins wird mit dem selektierten Ausgang verbunden. Der Ausgang des neuen Bausteins wird mit dem Zweig, mit dem der selektierte Ausgang verbunden war, verbunden.
- Ist ein Baustein, eine Funktion oder ein Funktionsblock selektiert (Cursorposition 3), so wird das alte Element durch den neuen Baustein ersetzt. Die Zweige werden, soweit möglich, wie vor der Ersetzung verbunden. Wenn das alte Element mehr Eingänge hatte als das neue, dann werden die unverknüpfbaren Zweige gelöscht. Das gleiche gilt für die Ausgänge.
- Ist ein Sprung oder ein Return selektiert, so wird der Baustein vor diesem Sprung, bzw. Return eingefügt. Der erste Eingang des Bausteins wird mit dem Zweig links des selektierten Elements verbunden. Der Ausgang des Bausteins wird mit dem Zweig rechts des selektierten Elements verbunden.
- Ist die letzte Cursorposition eines Netzwerks selektiert (Cursorposition 6), so wird der Baustein nach dem letzten Element eingefügt. Der erste Eingang des Bausteins wird mit dem Zweig links der selektierten Position verbunden.

Alle Eingänge des Bausteins, die nicht verbunden werden konnten, erhalten den Text "???". Dieser Text muss angeklickt und in die gewünschte Konstante bzw. Variable geändert werden.
Steht rechts von einem eingefügten Baustein ein Ast, so wird dieser dem ersten Bausteinausgang zugeordnet. Ansonsten bleiben die Ausgänge unbelegt.

'Einfügen' 'Eingang' Kurzform: <Strg>+<U>

Dieser Befehl fügt einen Operatoreingang ein. Die Zahl der Eingänge ist bei vielen Operatoren variabel (z.B. ADD kann 2 oder mehr Eingänge haben). Um einen solchen Operator um einen Eingang zu erweitern, muss der Eingang, vor dem ein weiterer eingefügt werden soll (Cursorposition 1), oder der Operator selbst (Cursorposition 3), wenn ein unterster Eingang angefügt werden soll, selektiert werden.

Der eingefügte Eingang ist mit dem Text "???" belegt. Dieser Text muss angeklickt und in die gewünschte Konstante bzw. Variable geändert werden. Dazu können Sie auch die Eingabehilfe verwenden.

'Einfügen' 'Ausgang'

Dieser Befehl fügt zu einer existierenden Zuweisung eine zusätzliche Zuweisung hinzu. Diese Funktionalität dient dem Erstellen sogenannter Zuweisungskämme, d.h. der Zuweisung des aktuell an der Leitung anliegenden Wertes an mehrere Variablen.

Ist das Linienkreuz über einer Zuweisung (Cursorposition 5) bzw. der unmittelbar davor liegende Ausgang selektiert (Cursorposition 4), so wird nach den bereits vorhandenen Zuweisungen eine weitere angefügt.

Ist das Linienkreuz direkt vor einer Zuweisung selektiert (Cursorposition 4), so wird vor dieser Zuweisung eine weitere eingefügt.

Der eingefügte Ausgang ist mit dem Text "???" belegt. Dieser Text muss angeklickt und in die gewünschte Variable geändert werden. Dazu können Sie auch die Eingabehilfe verwenden.

'Extras' 'Negation' Kurzform: <Strg>+<N>

Mit diesem Befehl können Sie Eingänge, Ausgänge, Sprünge oder RETURN-Anweisungen negieren. Das Symbol für die Negation ist ein kleiner Kreis auf einer Verbindung.

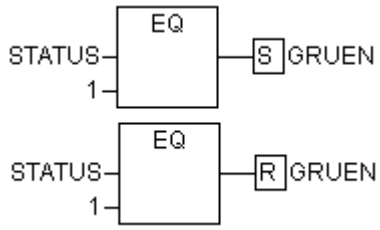
Wenn ein Eingang selektiert ist (Cursorposition 2), dann wird dieser Eingang negiert.

Wenn ein Ausgang selektiert ist (Cursorposition 4), dann wird dieser Ausgang negiert.

Wenn ein Sprung oder ein Return markiert ist, dann wird der Eingang dieses Sprungs, bzw. Returns negiert. Eine Negation kann durch erneutes Negieren gelöscht werden.

'Extras' 'Set/Reset' Symbol:

Mit diesem Befehl können Ausgänge als Set bzw. Reset Ausgänge definiert werden. Ein Gatter mit Set Ausgang wird mit [S] und ein Gatter mit Reset Ausgang mit [R] dargestellt.



Set/Reset Ausgänge in FUP

Ein Set Ausgang wird auf TRUE gesetzt, wenn das zugehörige Gatter TRUE liefert. Der Ausgang behält nun diesen Wert, auch wenn das Gatter wieder auf FALSE zurückspringt. Ein Reset Ausgang wird auf FALSE gesetzt, wenn das zugehörige Gatter TRUE liefert. Der Ausgang behält seinen Wert, auch wenn das Gatter wieder auf FALSE zurückspringt. Bei mehrfachen Ausführen des Befehls wechselt der Ausgang zwischen Set-, Reset- und normalen Ausgang.

'Extras' 'Zoom' Kurzform: <Alt>+<Eingabetaste>

Mit diesem Befehl laden Sie einen selektierten Baustein in seinen Editor (Cursorposition 3). Handelt es sich um einen Baustein aus einer Bibliothek, so wird der Bibliotheksverwalter aufgerufen und der entsprechende Baustein angezeigt.

'Extras' 'Instanz öffnen'

Dieser Befehl entspricht dem Befehl 'Projekt' 'Instanz öffnen'.

Ausschneiden, Kopieren, Einfügen und Löschen in FUP

Die Befehle zum 'Ausschneiden', 'Kopieren', 'Einfügen' oder 'Löschen', befinden sich unter dem Menüpunkt 'Bearbeiten'.

Ist ein Linienkreuz selektiert (Cursorposition 5), so werden die darunter liegenden Zuweisungen, Sprünge oder RETURN-Anweisungen ausgeschnitten, gelöscht oder kopiert.

Ist ein Operator, eine Funktion oder ein Funktionsbaustein selektiert (Cursorposition 3), so werden das selektierte Objekt selbst, sowie alle an den Eingängen anliegenden Äste mit Ausnahme des ersten Astes ausgeschnitten, gelöscht oder kopiert. Ansonsten wird der gesamte vor der Cursorposition liegende Ast ausgeschnitten, gelöscht oder kopiert.

Nach dem Kopieren oder Ausschneiden liegt der gelöschte bzw. kopierte Teil in der Zwischenablage und kann nun beliebig oft eingefügt werden. Dazu muss zunächst die Einfügeposition ausgewählt werden.

Gültige Einfügepositionen sind Eingänge und Ausgänge. Wenn in die Zwischenablage ein Operator, eine Funktion oder ein Funktionsbaustein geladen wurde (zur Erinnerung: in diesem Fall liegen alle anliegenden Zweige außer dem ersten, mit in der Zwischenablage), wird der erste Eingang mit dem Ast vor der Einfügeposition verbunden.

Andernfalls wird der gesamte vor der Einfügeposition liegende Ast durch den Inhalt der Zwischenablage ersetzt. In jedem Fall wird das letzte eingefügte Element mit dem rechts von der Einfügeposition liegenden Ast verbunden.

Durch Ausschneiden und Einfügen lässt sich nun folgendes Problem lösen: In der Mitte eines Netzwerks wird ein neuer Operator eingefügt. Der rechts vom Operator liegende Ast ist nun mit dem ersten Eingang verbunden, muss aber mit dem 2. Eingang verbunden sein. Man selektiert nun den ersten Eingang und führt ein **'Bearbeiten' 'Ausschneiden'** aus. Anschließend selektiert man den zweiten Eingang und führt ein '**Bearbeiten' 'Einfügen**' aus. Somit hängt der Ast nun am 2. Eingang.

Der Funktionsplan im Online Modus

Im Funktionsplan können Breakpoints nur auf Netzwerke gesetzt werden. Wenn ein Breakpoint auf ein Netzwerk gesetzt wurde, dann wird das Netzwerknummernfeld blau dargestellt. Die Bearbeitung stoppt dann vor dem Netzwerk, auf dem der Breakpoint steht. In diesem Fall wird das Netzwerknummernfeld rot. Beim Steppen (Einzelschritt) wird von Netzwerk zu Netzwerk gesprungen.

Zu jeder Variablen, wird der aktuelle Wert dargestellt.

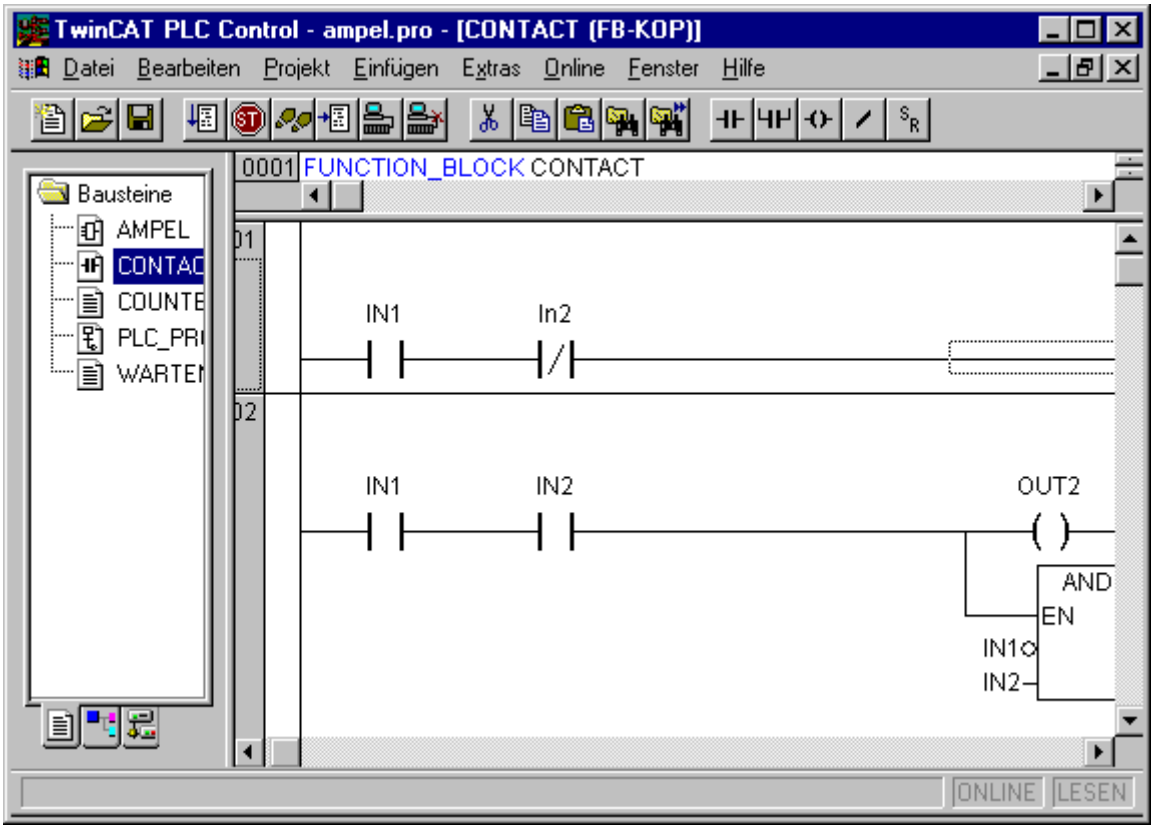
Ausnahme: Wenn der Eingang eines Funktionsblocks ein Ausdruck ist, wird nur die erste Variable des Ausdrucks gemonitort

Ein Doppelklick auf eine Variable öffnet den Dialog zum Schreiben einer Variablen. Hier ist es möglich, den aktuellen Wert der Variablen zu ändern. Bei boolschen Variablen erscheint kein Dialog, sie werden getoggelt. Der neue Wert wird rot und bleibt unverändert. Wenn der Befehl **'Online''Werte schreiben'** gegeben wird, dann werden alle Variablen auf die gewählten Werte gesetzt und wieder schwarz dargestellt. Die Ablaufkontrolle starten Sie mit dem Menübefehl **'Online' 'Ablaufkontrolle**'. Mit ihr können Sie die aktuellen Werte, die in den Netzwerken über die Verbindungslinien transportiert werden, einsehen. Wenn die Verbindungslinien keine boolschen Werte transportieren, dann wird der Wert in einem eigens eingefügten Feld angezeigt. Wenn die Linien boolsche Werte transportieren, dann werden sie, für den Fall, dass sie TRUE transportieren, blau eingefärbt. So kann der Informationsfluss während des Steuerungslaufs mitverfolgt werden.

Wenn Sie den Mauszeiger eine kurze Zeit über einer Variablen halten, wird der Typ, die Adresse und der Kommentar der Variablen in einem Tooltip angezeigt.

5.7 Kontaktplaneditor

So sieht ein in KOP geschriebener Baustein im TwinCAT PLC Control-Editor aus:



Baustein im Kontaktplan

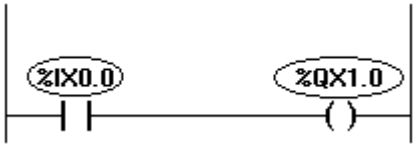
Alle Editoren für Bausteine bestehen aus einem Deklarationsteil und einem Rumpf. Diese sind getrennt durch einen Bildschirmteiler.

Der KOP-Editor ist ein graphischer Editor. Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste oder <Strg>+<F10>. Für Informationen über die Elemente, siehe Kontaktplan (KOP).

Cursorpositionen im KOP-Editor

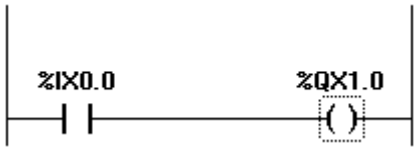
Folgende Stellen können Cursorpositionen sein, wobei Funktionsblock- und Programmaufrufe wie Kontakte behandelt werden können. Bausteine mit EN-Eingängen und daran geknüpfte andere Bausteine werden behandelt wie im Funktionsplan. Information über das Editieren dieser Netzwerkteile finden Sie im Kapitel über den FUP-Editor.

1. Jedes Textfeld (mögliche Cursorpositionen schwarz umrahmt)

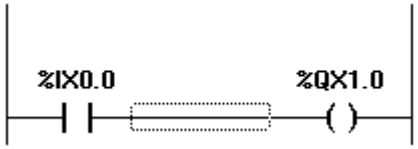


2. Jeder Kontakt oder Funktionsblock

3. Jede Spule



4. Die Verbindungslinie zwischen den Kontakten und den Spulen.



Folgende Menübefehle benutzt speziell der Kontaktplan:

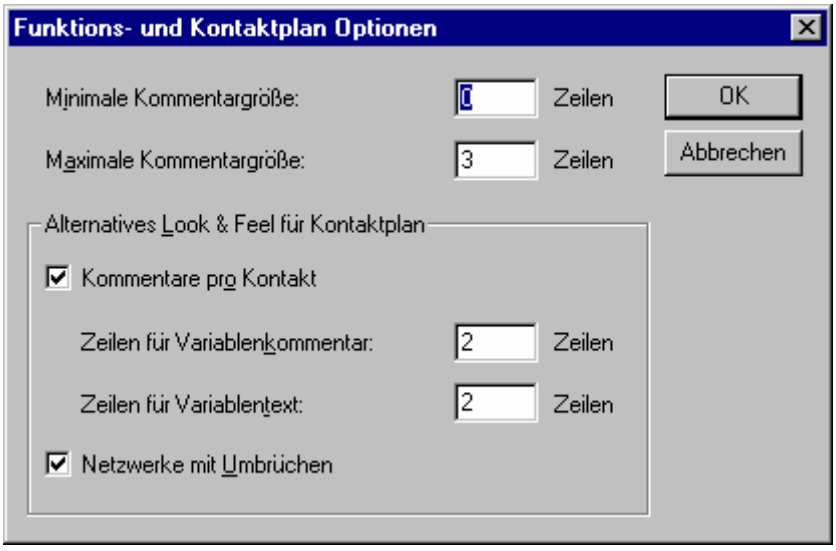
'Einfügen''Kontakt' Kurzform: <Strg>+<O>

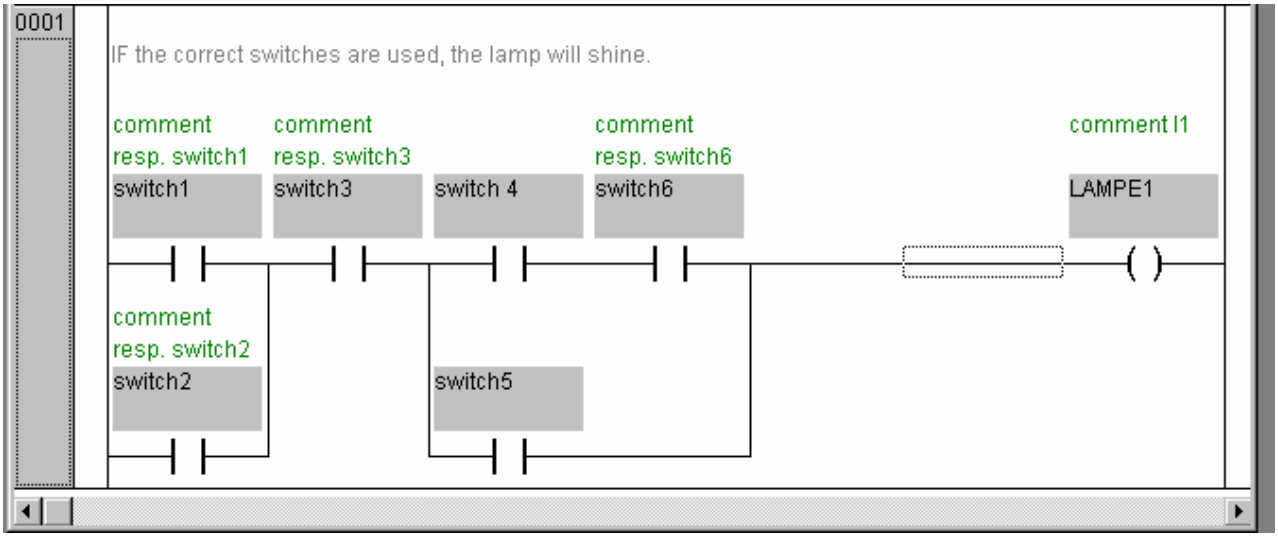
Mit diesem Befehl fügen Sie im KOP-Editor einen Kontakt vor der markierten Stelle im Netzwerk ein.

Ist die markierte Stelle eine Spule (Cursorposition 3), oder die Verbindungslinie zwischen den Kontakten und den Spulen (Cursorposition 4), dann wird der neue Kontakt seriell zur bisherigen Kontaktschaltung geschaltet.

Der Kontakt erhält als Vorbelegung den Text "???". Sie können diesen Text anklicken und in die gewünschte Variable bzw. die gewünschte Konstante ändern. Dazu können Sie auch die Eingabehilfe verwenden. Wenn Sie über den Befehl 'Extras' 'Optionen' im Dialog 'Funktionsplan- und Kontaktplan Optionen' die Option Kommentare pro Kontakt aktiviert haben, können Sie im selben Dialog neben einer gewünschten Anzahl Zeilen für den Variablenkommentar auch eine bestimmte Anzahl Zeilen für den Variablennamen vorgeben. Dies ist sinnvoll bei langen Variablennamen, um das Netzwerk horizontal kompakt zu halten.

Beachten Sie außerdem die Option Netzwerke mit Umbrüchen, die Sie ebenfalls über den Befehl '**Extras' 'Optionen**' im Dialog '**Funktionsplan- und Kontaktplan Optionen**' einschalten können.





'Einfügen''Paralleler Kontakt' Kurzform: <Strg>+<R>

Mit diesem Befehl fügen Sie im KOP-Editor einen Kontakt parallel zur markierten Stelle im Netzwerk ein. Ist die markierte Stelle eine Spule (Cursorposition 3), oder die Verbindung zwischen den Kontakten und den Spulen (Cursorposition 4), dann wird der neue Kontakt parallel zur gesamten bisherigen Kontaktschaltung geschaltet. Der Kontakt erhält als Vorbelegung den Text "???". Sie können diesen Text anklicken und in die gewünschte Variable bzw. die gewünschte Konstante ändern. Dazu können Sie auch die Eingabehilfe verwenden.

'Einfügen' 'Funktionsblock' Kurzform: <Strg>+<B>

Diesen Befehl verwenden Sie, um einen Funktionsblock oder ein Programm als Baustein einzufügen. Dazu muss die Verbindung zwischen den Kontakten und den Spulen markiert sein (Cursorposition 4) oder eine Spule (Cursorposition 3). Der Eingabehilfe-Dialog öffnet sich, wo Sie aus den zur Verfügung stehenden Standard- und selbst definierten Bausteinen auswählen können.

Der erste Eingang des neu eingefügten Bausteins wird auf die Eingangsverbindung, der erste Ausgang auf die Ausgangsverbindung gelegt, daher müssen diese Variablen unbedingt vom Typ BOOL sein. Alle anderen Ein- und Ausgänge des Bausteins werden mit dem Text "???" besetzt. Diese Vorbelegungen können in andere Konstanten, Variablen oder Adressen geändert werden. Dazu können Sie auch die Eingabehilfe verwenden.

'Einfügen''Spule' Kurzform: <Strg>+<L>

Mit diesem Befehl fügen Sie im KOP-Editor eine Spule parallel zu den bisherigen Spulen ein. Wenn die markierte Stelle die Verbindung zwischen den Kontakten und den Spulen ist (Cursorposition 4), dann wird die neue Spule als letzte eingefügt. Wenn die markierte Stelle eine Spule ist (Cursorposition 3), dann wird die neue Spule direkt darüber eingefügt. Die Spule erhält als Vorbelegung den Text "???". Sie können diesen Text anklicken und in die gewünschte Variable ändern. Dazu können Sie auch die Eingabehilfe verwenden.

Bausteine mit EN-Eingängen

Wenn Sie mit Ihrem KOP-Netzwerk zur Steuerung von Aufrufen anderer Bausteine verwenden wollen, dann müssen Sie einen Baustein mit einem EN-Eingang einfügen. Ein solcher Baustein wird parallel zu den Spulen geschaltet. Ausgehend von solch einem Baustein können Sie das Netzwerk wie im Funktionsplan weiterentwickeln. Die Befehle zum Einfügen an einen EN-Baustein finden Sie unter dem Menüpunkt '**Einfügen' 'Einfügen an Baustein**'.

Ein Operator, ein Funktionsblock oder eine Funktion mit EN-Eingang verhält sich wie der entsprechende Baustein im Funktionsplan, mit dem Unterschied, dass seine Ausführung über den EN-Eingang gesteuert wird. Dieser Eingang wird an der Verbindungslinie zwischen Spulen und Kontakten angeschlossen. Wenn diese Verbindung die Information "An" transportiert, dann wird der Baustein ausgewertet.

Wenn einmal ein Baustein mit EN-Eingang angelegt wurde, kann mit diesem Baustein ein Netzwerk wie im Funktionsplan angelegt werden. D.h. in eine EN-Baustein können Daten von üblichen Operatoren, Funktionen, Funktionsblöcken fließen, und ein EN-Baustein kann Daten an solche üblichen Bausteine transportieren.

Wenn Sie also im KOP-Editor ein Netzwerk wie in FUP programmieren wollen, müssen Sie nur in ein neues Netzwerk zuerst einen EN-Operator einfügen, anschließend können Sie von diesem Baustein aus ihr Netzwerk weiterbilden wie im FUP-Editor. Ein so gebildetes Netzwerk verhält sich wie das entsprechende Netzwerk in FUP.

'Einfügen''Baustein mit EN'

Mit diesem Befehl fügen Sie einen Funktionsblock, einen Operator, eine Funktion oder ein Programm mit EN-Eingang in ein KOP-Netzwerk ein.

Die markierte Stelle muss die Verbindung zwischen den Kontakten und den Spulen sein (Cursorposition 4) oder eine Spule (Cursorposition 3). Der neue Baustein wird parallel zu den Spulen unterhalb derselben eingefügt und trägt zunächst die Bezeichnung "AND". Diese Bezeichnung können Sie in die gewünschte ändern. Dazu können Sie auch die Eingabehilfe verwenden. Zur Verfügung stehen Standard- und selbst definierte Bausteine..

'Einfügen''Einfügen an Baustein'

Mit diesem Befehl können Sie an einen bereits eingefügten Baustein (auch ein Baustein mit EN-Eingang), weitere Elemente anfügen. Die Befehle unter diesem Menüpunkt sind an denselben Cursorpositionen ausführbar wie die entsprechenden Befehle im Funktionsplan (siehe Kapitel 5.7.).

Mit **Eingang** fügen Sie einen neuen Eingang an den Baustein an.

Mit **Ausgang** fügen Sie einen neuen Ausgang an den Baustein an.

Mit Baustein fügen Sie einen weiteren Baustein an. Die Vorgehensweise entspricht der, die unter 'Einfügen' 'Baustein' beschrieben ist.

Mit **Zuweisung** können Sie eine Zuweisung zu einer Variablen einfügen. Zunächst wird diese durch drei Fragezeichen "???" dargestellt, die Sie editieren und durch die gewünschte Variable ersetzen können. Die Eingabehilfe steht hierbei zur Verfügung.

'Einfügen''Sprung'

Mit diesem Befehl fügen Sie im KOP-Editor einen Sprung parallel am Ende zu den bisherigen Spulen ein. Wenn die eingehende Leitung den Wert "An" liefert, dann wird der Sprung an die bezeichnete Marke durchgeführt.

Die markierte Stelle muss die Verbindung zwischen den Kontakten und den Spulen sein (Cursorposition 4) oder eine Spule (Cursorposition 3).Der Sprung erhält als Vorbelegung den Text "???". Sie können diesen Text anklicken und in die gewünschte Sprungmarke ändern.

'Einfügen''Return'

Mit diesem Befehl fügen Sie im KOP-Editor eine RETURN-Anweisung parallel am Ende zu den bisherigen Spulen ein. Wenn die eingehende Leitung den Wert "An" liefert, die Abarbeitung des Bausteins in diesem Netzwerk abgebrochen.Die markierte Stelle muss die Verbindung zwischen den Kontakten und den Spulen sein (Cursorposition 4) oder eine Spule (Cursorposition 3).

'Extras''Dahinter Einfügen'

Mit diesem Befehl fügen Sie im KOP-Editor den Inhalt der Zwischenablage als seriellen Kontakt nach der Markierungsstelle ein. Dieser Befehl ist nur möglich, wenn der Inhalt der Zwischenablage und die markierte Stelle Netzwerke aus Kontakten sind.

'Extras''Darunter Einfügen' Kurzform: <Strg>+<U>

Mit diesem Befehl fügen Sie im KOP-Editor den Inhalt der Zwischenablage als parallelen Kontakt unter der Markierungsstelle ein. Dieser Befehl ist nur möglich, wenn der Inhalt der Zwischenablage und die markierte Stelle Netzwerke aus Kontakten sind.

'Extras''Darüber Einfügen'

Mit diesem Befehl fügen Sie im KOP-Editor den Inhalt der Zwischenablage als parallelen Kontakt über der Markierungsstelle ein. Dieser Befehl ist nur möglich, wenn der Inhalt der Zwischenablage und die markierte Stelle Netzwerke aus Kontakten sind.

'Extras''Negation' Kurzform: <Strg>+<N>

Mit diesem Befehl negieren Sie einen Kontakt, eine Spule, eine Sprung- oder RETURN-Anweisung oder Ein- bzw. Ausgang von EN-Bausteinen an der aktuellen Cursorposition (Cursorposition 2 und 3).

Zwischen den runden Klammern der Spule, bzw. zwischen den geraden Strichen des Kontakts erscheint ein Schrägstrich ((/) bzw. |/|). Bei Sprüngen, Returns, Ein- bzw. Ausgängen von EN-Bausteinen erscheint wie im FUP-Editor ein kleiner Kreis auf der Verbindung.

Die Spule schreibt nun den negierten Wert der Eingangsverbindung in die zugehörige boolsche Variable. Ein negierter Kontakt schaltet genau dann den Zustand des Eingangs auf den Ausgang, wenn die zugehörige boolsche Variable den Wert FALSE liefert.

Wenn ein Sprung oder ein Return markiert ist, dann wird der Eingang dieses Sprungs, bzw. Returns negiert. Eine Negation kann durch erneutes Negieren gelöscht werden.

'Extras''Set/Reset'

Wenn Sie diesen Befehl auf eine Spule ausführen, dann erhalten Sie eine Set-Spule. Eine solche Spule überschreibt niemals den Wert TRUE in der zugehörigen boolschen Variablen. Das heißt, wenn der Wert dieser Variablen einmal auf TRUE gesetzt wurde, dann bleibt er für immer auf TRUE. Eine Set-Spule wird mit einem "S" im Spulensymbol gekennzeichnet.

Wenn sie diesen Befehl erneut ausführen, dann erhalten Sie eine Reset-Spule. Eine solche Spule überschreibt niemals den Wert FALSE in der zugehörigen boolschen Variablen. Das heißt, wenn der Wert dieser Variablen einmal auf FALSE gesetzt wurde, dann bleibt er für immer auf FALSE. Eine Reset-Spule wird mit einem "R" im Spulensymbol gekennzeichnet.

Wenn Sie diesen Befehl öfters ausführen, dann wechselt diese Spule zwischen Set-, Reset- und normaler Spule.

'Extras' 'Instanz öffnen'

Dieser Befehl entspricht dem Befehl 'Projekt' 'Instanz öffnen'.

Der Kontaktplan im Online Modus

Im Online Modus werden im Kontaktplan alle Kontakte und Spulen, die im Zustand "An" (TRUE) sind, blau eingefärbt, ebenso werden alle Leitungen über die "An" transportiert wird, blau gefärbt. An den Ein- und Ausgängen von Funktionsblöcken werden die Werte der entsprechenden Variablen angezeigt.

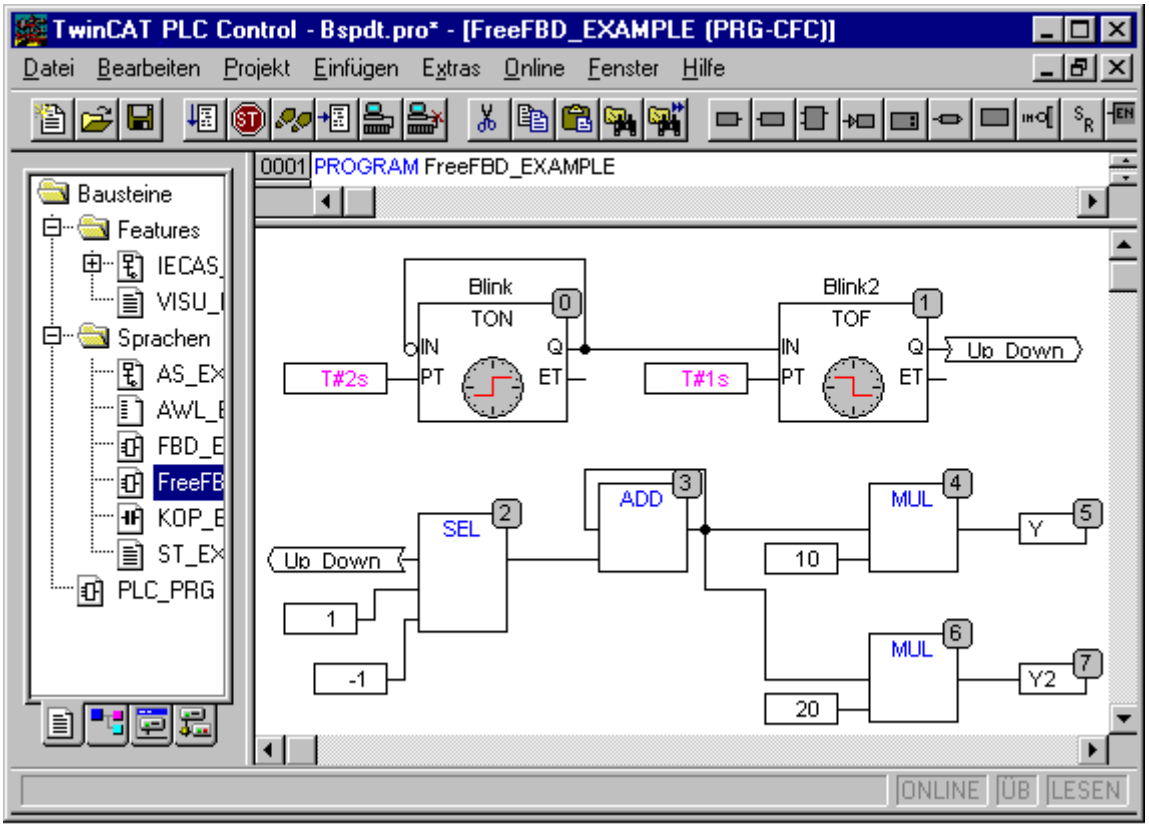
Breakpoints können nur auf Netzwerke gesetzt werden; beim **Steppen** wird von Netzwerk zu Netzwerk gesprungen.

Bei eingeschalteter **Ablaufkontrolle** ('Online' 'Ablaufkontrolle') werden die Nummernfelder der durchlaufenen Netzwerke grün markiert.

Wenn Sie den Mauszeiger eine kurze Zeit über einer Variablen halten, wird der Typ, die Adresse und der Kommentar der Variablen in einem Tooltip angezeigt.

5.8 Freigrafischer Funktionsplaneditor

So sieht ein Baustein aus, der mit dem freigrafischen Funktionsplaneditor (CFC) erstellt wurde:



Editor für freigrafischen Funktionsplan

Beim freigrafischen Funktionsplaneditor werden keine Netzwerke verwendet, sondern die Elemente können frei platziert werden. Zu den Elementen der Abarbeitungsliste gehören der Baustein, Eingang, Ausgang, Sprung, Label, Return und Kommentar. Die Ein- und Ausgänge dieser Elemente können durch Ziehen einer Verbindung mit der Maus verbunden werden. Die Verbindungslinie wird automatisch gezeichnet. Dabei wird unter Berücksichtigung der bestehenden Verbindungen die kürzeste Verbindungslinie gezeichnet. Beim Verschieben von Elementen werden die Verbindungslinien automatisch angepasst. Kann eine Verbindungslinie aus Platzgründen nicht gezeichnet werden, so wird eine rote Linie zwischen Eingang und zugehörigem Ausgang dargestellt. Sobald genügend Platz vorhanden ist, wird diese Linie in eine Verbindungslinie umgewandelt.

Ein Vorteil des freigrafischen gegenüber des gewöhnlichen Funktionsplaneditors FUP ist, dass Rückkopplungen direkt eingefügt werden können.

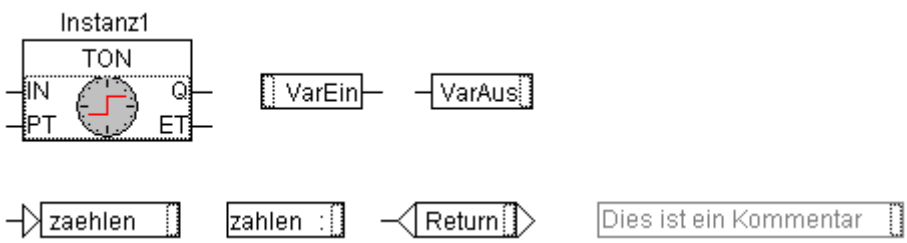
Die wichtigsten Befehle finden Sie im Kontextmenü.

Cursorpositionen im CFC

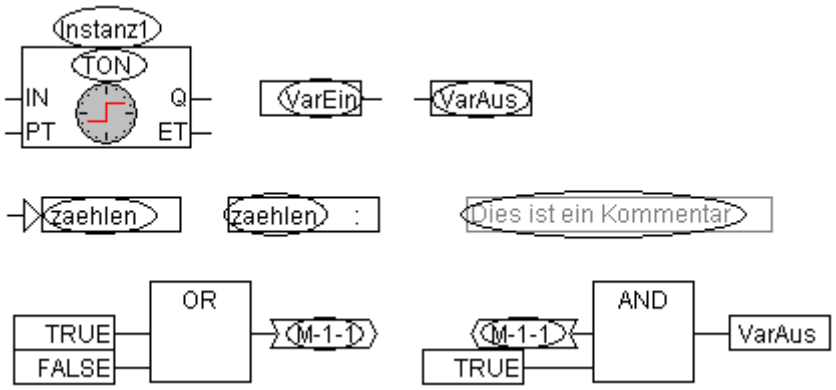
Jeder Text ist eine mögliche Cursorposition. Der selektierte Text ist blau hinterlegt und kann geändert werden.

Ansonsten ist die aktuelle Cursorposition durch ein gepunktetes Rechteck gekennzeichnet. Es folgt eine Aufzählung aller möglichen Cursorpositionen mit Beispielen:

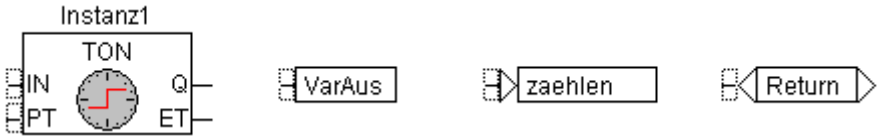
1. Rümpfe der Elemente Baustein, Eingang, Ausgang, Sprung, Label, Return und Kommentar:



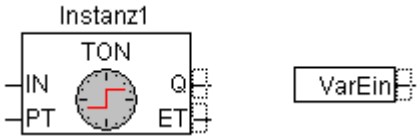
2. Textfelder der Elemente Baustein, Eingang, Ausgang, Sprung, Label und Kommentar, ferner die Textfelder der Verbindungsmarken:



3. Eingänge der Elemente Baustein, Ausgang, Sprung und Return:



4. Ausgänge der Elemente Baustein und Eingang:



'Einfügen' 'Baustein' Kurzform: <Strg>+<B>

Mit diesem Befehl können Operatoren, Funktionen, Funktionsblöcke und Programme eingefügt werden. Es wird zunächst stets ein "AND"-Operator eingefügt. Dieser kann durch Selektieren und Überschreiben des Textes in jeden anderen Operator, in jede Funktion, in jeden Funktionsblock und in jedes Programm umgewandelt werden. Mit der Eingabehilfe können Sie den gewünschten Baustein aus der Liste der unterstützten Bausteine auswählen. Hat der neue Baustein eine andere Mindestanzahl von Eingängen, so werden diese angehängt. Hat der neue Baustein eine kleinere Höchstzahl von Eingängen, so werden die letzten Eingänge gelöscht.

'Einfügen' 'Eingang' Kurzform: <Strg>+<E>

Mit diesem Befehl wird ein Eingang eingefügt. Der eingetragene Text "???" kann selektiert und durch eine Variable oder Konstante ersetzt werden. Dazu können Sie auch die Eingabehilfe verwenden.

'Einfügen' 'Ausgang' Kurzform: <Strg>+<A>

Mit diesem Befehl wird ein Ausgang eingefügt. Der eingetragene Text "???" kann selektiert und durch eine Variable ersetzt werden. Dazu können Sie auch die Eingabehilfe verwenden. Es wird der Wert, der am Eingang des Ausgangs anliegt, dieser Variablen zugewiesen.

'Einfügen' 'Sprung' Kurzform: <Strg>+<J>

Mit diesem Befehl wird ein Sprung eingefügt. Der eingetragene Text "???" kann selektiert und durch die Sprungmarke, an die gesprungen werden soll, ersetzt werden.
Die Sprungmarke wird mit dem Befehl 'Einfügen' 'Label' eingefügt.

'Einfügen' 'Label' Kurzform: <Strg>+<L>

Mit diesem Befehl wird ein Label eingefügt. Der eingetragene Text "???" kann selektiert und durch die Sprungmarke ersetzt werden. Im Online Modus wird automatisch ein RETURN-Label zur Markierung des Bausteinendes eingefügt.

Der Sprung wird mit dem Befehl '**Einfügen' 'Sprung**' eingefügt.

'Einfügen' 'Return' Kurzform: <Strg>+<R>

Mit diesem Befehl wird eine RETURN-Anweisung eingefügt.

Beachten Sie, dass im Online Modus automatisch eine Sprungmarke mit der Bezeichnung RETURN in der ersten Spalte und nach dem letzten Element im Editor eingefügt wird, die beim Steppen vor dem Verlassen des Bausteins angesprungen wird.

'Einfügen' 'Kommentar' Kurzform: <Strg>+<K>

Mit diesem Befehl wird ein Kommentar eingefügt.
Eine neue Zeile innerhalb des Kommentars, erhalten Sie mit <Strg> + <Eingabetaste>.

'Einfügen' 'Bausteineingang' Kurzform: <Strg>+<U>

Dieser Befehl fügt einen Bausteineingang ein. Die Zahl der Eingänge ist bei vielen Operatoren variabel (z.B. ADD kann 2 oder mehr Eingänge haben).
Um einen solchen Operator um einen Eingang zu erweitern, muss der Operator selbst (Cursorposition 1) selektiert werden.

'Einfügen' 'In-Pin', 'Einfügen' 'Out-Pin'

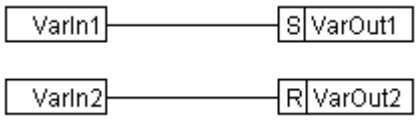
Diese Befehle stehen zur Verfügung, sobald ein Makro zur Bearbeitung geöffnet ist. Sie dienen dem Einfügen von In- bzw. Out-Pins als Ein- und Ausgänge des Makros. Sie unterscheiden sich von den normalen Ein- und Ausgängen der Bausteine durch die Darstellungsform und dadurch, dass sie keinen Positionsindex erhalten.

'Extras' 'Negieren' Kurzform: <Strg>+<N>

Mit diesem Befehl können Sie Eingänge, Ausgänge, Sprünge oder RETURN-Anweisungen negieren. Das Symbol für die Negation ist ein kleiner Kreis auf einer Verbindung.
Wenn ein Eingang eines Bausteins, Ausgangs, Sprungs oder Returns selektiert ist (Cursorposition 3), dann wird dieser Eingang negiert.
Wenn ein Ausgang eines Bausteins oder Eingangs selektiert ist (Cursorposition 4), dann wird dieser Ausgang negiert.
Eine Negation kann durch erneutes Negieren gelöscht werden.

'Extras' 'Set/Reset'

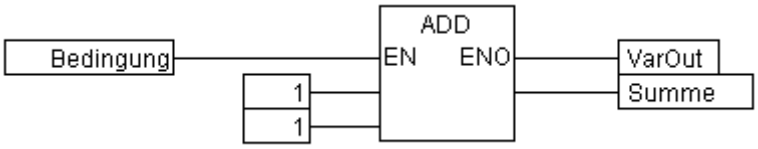
Dieser Befehl kann nur für selektierte Eingänge der Elemente Ausgang (Cursorposition 3) ausgeführt werden.
Das Symbol für Set ist S, das für Reset ist R.



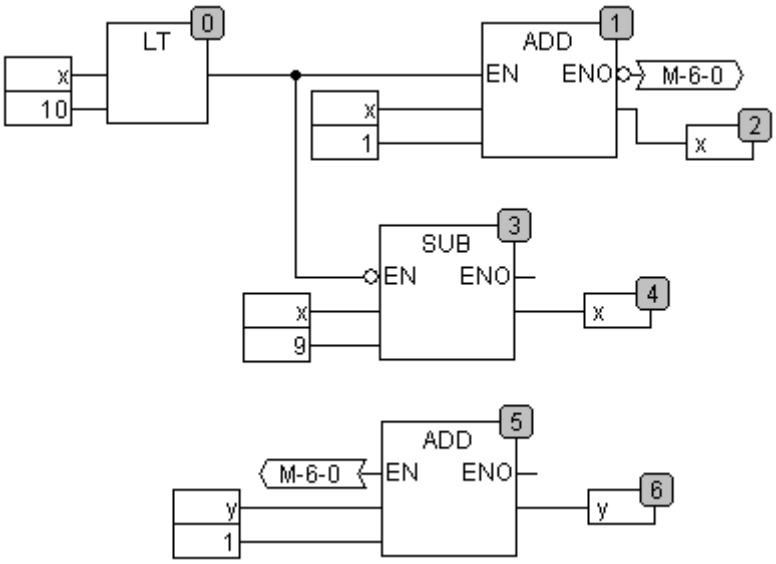
VarOut1 wird auf TRUE gesetzt, falls VarIn1 TRUE liefert. VarOut1 behält diesen Wert, auch wenn VarIn1 wieder auf FALSE zurückspringt.
VarOut2 wird auf FALSE gesetzt, falls VarIn2 TRUE liefert. VarOut2 behält diesen Wert, auch wenn VarIn2 wieder auf FALSE zurückspringt.
Bei mehrfacher Ausführung des Befehls wechselt der Ausgang zwischen Set-, Reset- und normalem Zustand.

'Extras' 'EN/ENO' Kurzform: <Strg>+<O>

Mit diesem Befehl erhält ein selektierter Baustein (Cursorposition 3) einen zusätzlichen boolschen Freigabe-Eingang EN (Enable In) und einen boolschen Ausgang ENO (Enable Out).



In diesem Beispiel wird ADD nur dann ausgeführt, wenn die boolsche Variable Bedingung TRUE ist. Nach der Ausführung von ADD wird VarOut ebenfalls auf TRUE gesetzt. Falls die Variable Bedingung gleich FALSE ist, wird ADD nicht abgearbeitet und VarOut erhält den Wert FALSE. Untenstehendes Beispiel zeigt, wie der Wert von ENO für weitere Bausteine verwendet werden kann.

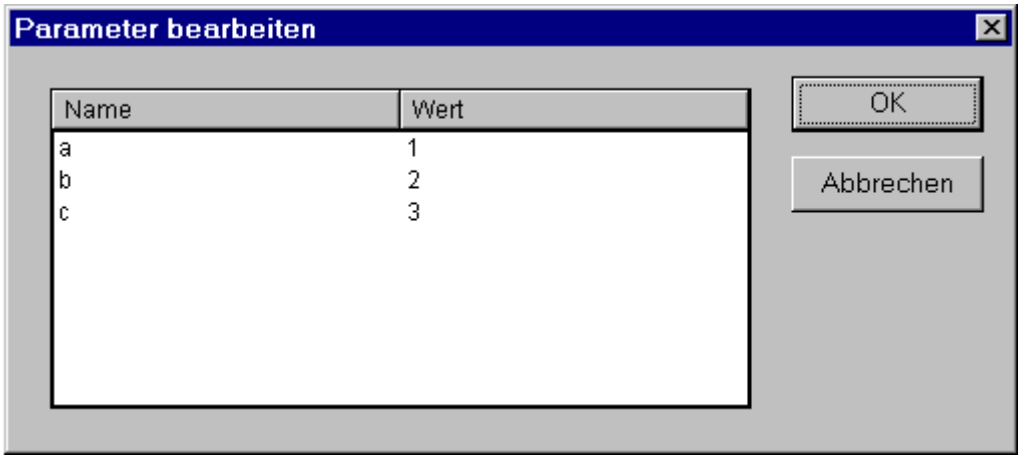


x soll dabei mit 1 und y mit 0 initialisiert sein. Die Nummern in der rechten Ecke der Bausteine geben die Abarbeitungsreihenfolge an.

x wird solange um eins erhöht, bis es den Wert 10 annimmt. Da dann der Ausgang des Bausteins LT(0) FALSE liefert, wird SUB(3) und ADD(5) ausgeführt. x wird also auf den Wert 1 gesetzt und y wird um 1 erhöht. Danach wird wieder LT(0) ausgeführt, solange x kleiner als 10 ist. y zählt also, wie oft x die Werte 1 bis 10 durchläuft.

'Extras' 'Eigenschaften...'

Im freigrafischen Funktionsplaneditor werden konstante Eingangsparameter (VAR_INPUT CONSTANT) von Funktionen und Funktionsblöcken nicht direkt angezeigt. Diese können angezeigt und in in ihrem Wert verändert werden, wenn man den Rumpf des betreffenden Bausteins selektiert (Cursorposition 1) und den Befehl 'Extras"Eigenschaften...' wählt oder einfach auf den Rumpf doppelklickt. Es öffnet sich der Dialog Parameter bearbeiten:



Die Werte der konstanten Eingangsparameter (VAR_INPUT CONSTANT) können verändert werden. Dazu muss der Parameterwert in der Spalte Wert markiert werden. Durch erneuten Mausklick oder durch Drücken der Leertaste kann dieser bearbeitet werden. Bestätigt wird die Änderung eines Wertes durch Drücken der <Eingabetaste>; durch Drücken der <Escape-Taste> werden die Änderungen verworfen. Mit der Schaltfläche **OK** werden alle Änderungen gespeichert.

Elemente selektieren

Um ein Element zu selektieren, klickt man mit der Maus auf den Rumpf des Elements (Cursorposition 1). Um mehrere Elemente zu markieren, drücken Sie die <Umschalt>-Taste und mit der Maus nacheinander auf die entsprechenden Elemente oder ziehen Sie bei gedrückter linker Maustaste ein Fenster über die zu markierenden Elemente auf.
Mit dem Befehl '**Extras' 'Alles Markieren**' können alle Elemente selektiert werden.

Elemente verschieben

Ein oder mehrere selektierte Elemente können mit den Pfeiltasten bei gedrückter <Umschalt>-Taste verschoben werden. Eine weitere Möglichkeit ist, die Elemente mit gedrückter linker Maustaste zu verschieben. Diese Elemente werden durch Loslassen der linken Maustaste abgelegt, sofern sie nicht andere Elemente überdecken oder die vorgesehene Größe des Editors überschreiten. In diesem Fall erhalten die markierten Elemente wieder in ihrer ursprünglichen Position und es ertönt eine Warnung.

Elemente kopieren

Ein oder mehrere selektierte Elemente werden mit dem Befehl '**Bearbeiten' 'Kopieren'**, kopiert und mit '**Bearbeiten' 'Einfügen'** eingefügt.

Verbindungen erstellen

Ein Eingang eines Elementes kann mit genau einem Ausgang eines Elementes verbunden werden. Ein Ausgang eines Elementes kann mit mehreren Eingängen von Elementen verbunden werden.
Es gibt mehrere Möglichkeiten, einen Eingang eines Elementes E2 mit dem Ausgang eines Elementes E1 zu verbinden.

Mit der linken Maustaste auf den Ausgang des Elements E1 (Cursorposition 4) klicken, die linke Maustaste gedrückt halten, den Mauszeiger auf den Eingang des Elements E2 (Cursorposition 3) ziehen und dort die linke Maustaste loslassen. Während des Ziehvorganges mit der Maus wird eine Verbindung vom Ausgang des Elements E1 zum Mauszeiger gezeichnet.

Mit der linken Maustaste auf den Eingang des Elements E2 klicken, die linke Maustaste gedrückt halten, den Mauszeiger auf den Ausgang des Elements E1 ziehen und dort die linke Maustaste loslassen.

Eines der Elemente E1 oder E2 verschieben (Cursorposition 1) und durch Loslassen der linken Maustaste so ablegen, dass sich der Ausgang von Element E2 und Eingang von Element E1 berühren.

Falls das Element E2 ein Baustein mit einem freien Eingang ist, kann mit der Maus auch eine Verbindung von einem Ausgang von E1 in den Rumpf von E2 gezogen werden. Beim Loslassen der Maustaste wird automatisch eine Verbindung mit dem obersten freien Eingang von E2 hergestellt. Wenn der Baustein E2 keinen freien Eingang hat, aber ein Operator ist, der um einen Eingang erweiterbar ist, dann wird automatisch ein neuer Eingang erzeugt.

Mit Hilfe dieser Methoden können auch der Ausgang und Eingang eines Bausteins miteinander verbunden werden (Rückkopplung). Zum Erstellen einer Verbindung zwischen zwei Pins klicken Sie mit der linken Maustaste auf einen Pin, halten die Taste gedrückt und ziehen dabei die Verbindung zum gewünschten Pin, wo Sie die Taste wieder loslassen. Wird während des Verbindungsziehens der Arbeitsbereich des Editors verlassen, so wird automatisch gescrollt. Für einfache Datentypen erfolgt während des Verbindens eine Typprüfung. Sind die Typen der beiden Pins nicht kompatibel, so ändert sich der Cursor auf "Verboten". Für komplexe Datentypen erfolgt keine Überprüfung.

Verbindungen löschen

Es gibt mehrere Möglichkeiten, eine Verbindung zwischen dem Ausgang eines Elementes E1 und einem Eingang eines Elementes E2 zu löschen:

Den Ausgang von E1 selektieren (Cursorposition 4) und die <Entf-Taste> drücken oder den Befehl **'Bearbeiten"Löschen'** ausführen. Ist der Ausgang von E1 mit mehreren Eingängen verbunden, so werden mehrere Verbindungen gelöscht.

Den Eingang von E2 selektieren (Cursorposition 4) und die <Entfernen-Taste> drücken oder den Befehl '**Bearbeiten"Löschen**' ausführen.

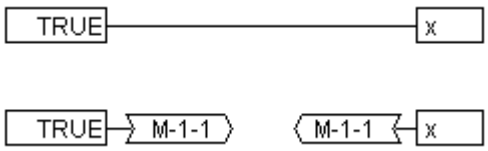
Mit der Maus den Eingang von E2 selektieren, die linke Maustaste dabei gedrückt halten und die Verbindung vom Eingang von E2 wegziehen. Wird die linke Maustaste dann in einem freien Bereich losgelassen, so wird die Verbindung gelöscht.

Verbindungen ändern

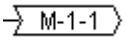
Eine Verbindung zwischen dem Ausgang eines Elementes E1 und dem Eingang eines Elementes E2 kann leicht in eine Verbindung zwischen dem Ausgang von E1 und einem Eingang eines Elements E3 geändert werden. Dazu wird mit der Maus auf den Eingang von E2 geklickt (Cursorposition 3), die linke Maustaste dabei gedrückt gehalten, der Mauszeiger auf den Eingang von E3 bewegt und dort losgelassen.

'Extras' 'Verbindungsmarke'

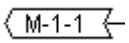
Verbindungen können an Stelle von Verbindungslinien auch mit Hilfe von Konnektoren (Verbindungsmarken) dargestellt werden. Dabei werden der Ausgang und der zugehörige Eingang mit einem Konnektor, der einen eindeutigen Namen hat, versehen.
Liegt bereits eine Verbindung zwischen zwei Elementen vor, die nun in Konnektor-Darstellung angezeigt werden soll, so wird zunächst der Ausgang der Verbindungslinie markiert (Cursorposition 3) und der Menüpunkt 'Extras'Verbindungsmarke angewählt. Nachfolgende Darstellung zeigt eine Verbindung vor und nach dem Anwählen letzteren Menüpunktes.



Vom Programm wird standardmäßig ein eindeutiger Konnektorname vergeben, der mit M beginnt, aber verändert werden kann. Der Konnektorname wird als Parameter des Ausgangs gespeichert, kann jedoch sowohl am Eingang als auch am Ausgang editiert werden:

Editieren des Konnektornamens am Ausgang: 

Wird der Text im Konnektor ersetzt, so wird der neue Konnektorname bei allen zugehörigen Konnektoren an Eingängen übernommen. Es kann jedoch kein Name gewählt werden, der bereits zu einer anderen Verbindungsmarke gehört, da somit die Eindeutigkeit des Konnektornamens verletzt wäre. Eine entsprechende Meldung wird in diesem Fall ausgegeben.

Editieren des Konnektornamens am Eingang: 

Wird der Text im Konnektor ersetzt, so wird er auch in der zugehörigen Verbindungsmarke am anderen Baustein ersetzt. Verbindungen in Konnektordarstellung können wieder in gewöhnliche Verbindungen umgewandelt werden, indem man die Ausgänge der Verbindungen markiert (Cursorposition 4) und den Menüpunkt '**Extras"Verbindungsmarke**' erneut anwählt.

Inputs/Outputs "On the fly" einfügen

Ist exakt ein Input- bzw. Output-Pin eines Elementes selektiert, kann unmittelbar durch Eingabe einer Zeichenfolge über die Tastatur das entsprechende Input- bzw. Output-Element eingefügt werden und dessen Editorfeld mit der Zeichenfolge gefüllt werden.

Abarbeitungsreihenfolge

Beim freigrafischen Funktionsplaneditor erhalten die Elemente Baustein, Ausgang, Sprung, Return und Label jeweils eine Abarbeitungsnummer. In dieser Reihenfolge werden die einzelnen Elemente zur Laufzeit berechnet.
Beim Einfügen eines Elements wird die Nummer automatisch in topologischer Reihenfolge vergeben (von links nach rechts und von oben nach unten). Wurde die Reihenfolge bereits verändert, erhält das neue Element die Nummer seines topologischen Nachfolgers und alle höheren Nummern werden um eins erhöht.
Beim Verschieben eines Elementes bleibt die Nummer erhalten.
Die Reihenfolge hat Einfluss auf das Ergebnis und muss in bestimmten Fällen geändert werden.
Wird die Reihenfolge angezeigt, so erscheint bei den Elementen in der rechten oberen Ecke die jeweilige Abarbeitungsnummer.

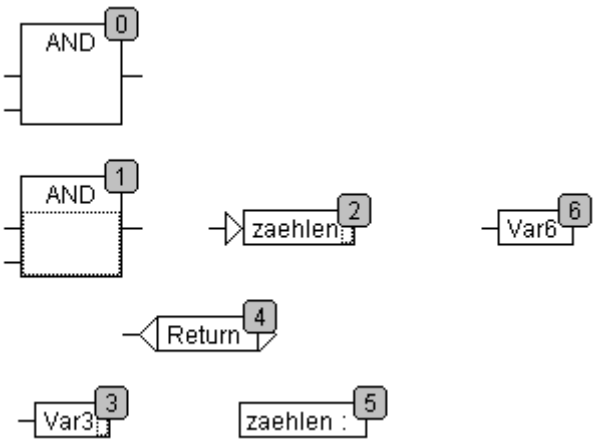
'Extras' 'Reihenfolge' 'Anzeigen'

Mit diesem Befehl wird die Anzeige der Abarbeitungsreihenfolge an- bzw. angeschaltet. Standardmäßig wird die Abarbeitungsreihenfolge angezeigt (erkennbar am Haken (*) vor dem Menüpunkt).
Bei den Elementen Baustein, Ausgang, Sprung, Return und Label erscheint in der rechten oberen Ecke ihre jeweilige Abarbeitungsnummer.

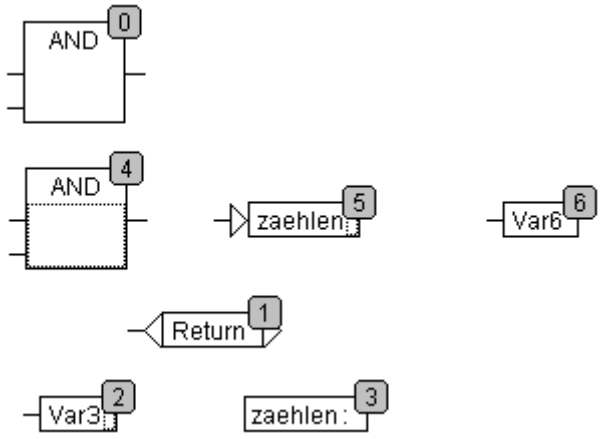
'Extras' 'Reihenfolge' 'Topologisch anordnen'

Elemente sind in topologischer Reihenfolge angeordnet, wenn die Abarbeitung von links nach rechts und von oben nach unten stattfindet, d.h. bei topologisch angeordneten Elementen nimmt die Nummer von links nach rechts und von oben nach unten zu. Die Verbindungen spielen keine Rolle. Es zählt nur die Lage der Elemente.

Wird der Befehl **'Extras"Reihenfolge"Topologisch anordnen'** ausgeführt, so werden alle **markierten** Elemente topologisch angeordnet. Alle Elemente der Selektion werden dabei aus der Abarbeitungsliste herausgenommen. Danach werden die Elemente der Selektion einzeln von rechts unten nach links oben wieder in die verbliebende Abarbeitungsliste eingefügt. Jedes markierte Element wird dabei in der Abarbeitungsliste vor dem topologischen Nachfolger eingefügt, d.h. es wird vor dem Element eingefügt, das bei einer topologischen Anordnung danach abgearbeitet werden würde, wenn alle Elemente des Editors in topologischer Reihenfolge angeordnet wären. Dies wird an einem Beispiel verdeutlicht.



Die Elemente mit der Nummer 1, 2 und 3 sind selektiert. Wird jetzt der Befehl '**Topologisch anordnen**' angewählt, werden die drei selektierten Elemente zunächst aus der Abarbeitungsliste herausgenommen. Dann werden nacheinander Var3, der Sprung und der AND-Operator wieder eingefügt. Var3 wird vor das Label eingeordnet und bekommt die Nummer 2. Danach wird der Sprung eingeordnet und erhält zunächst die 4, nach Einfügen des AND die 5. Es ergibt sich folgende neue Abarbeitungsreihenfolge:



Beim Ablegen eines neu erzeugten Bausteins wird dieser standardmäßig vor ihren topologischen Nachfolger in die Abarbeitungsliste einsortiert.
'Extras''Reihenfolge''Eins vor'

Mit diesem Befehl werden alle selektierten Elemente mit Ausnahme des Elements, das sich am Anfang der Abarbeitungsreihenfolge befindet, innerhalb der Abarbeitungsreihenfolge um einen Platz nach vorne verschoben.

'Extras''Reihenfolge''Eins zurück'

Mit diesem Befehl werden alle selektierten Elemente mit Ausnahme des Elements, das sich am Ende der Abarbeitungsreihenfolge befindet, innerhalb der Abarbeitungsreihenfolge um einen Platz nach hinten verschoben.

'Extras''Reihenfolge''An den Anfang'

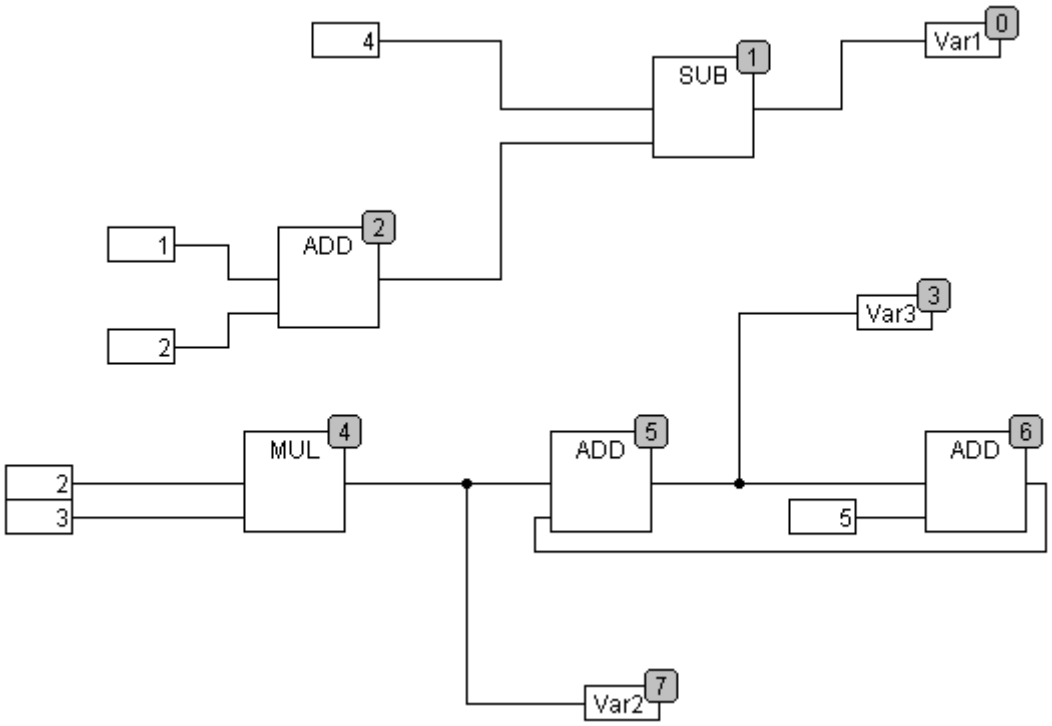
Mit diesem Befehl werden alle selektierten Elemente an den Anfang der Abarbeitungsreihenfolge geschoben, wobei die Reihenfolge innerhalb der selektierten Elemente beibehalten wird. Ebenso bleibt die Reihenfolge innerhalb der nicht selektierten Elemente beibehalten.

'Extras' 'Reihenfolge' 'Ans Ende'

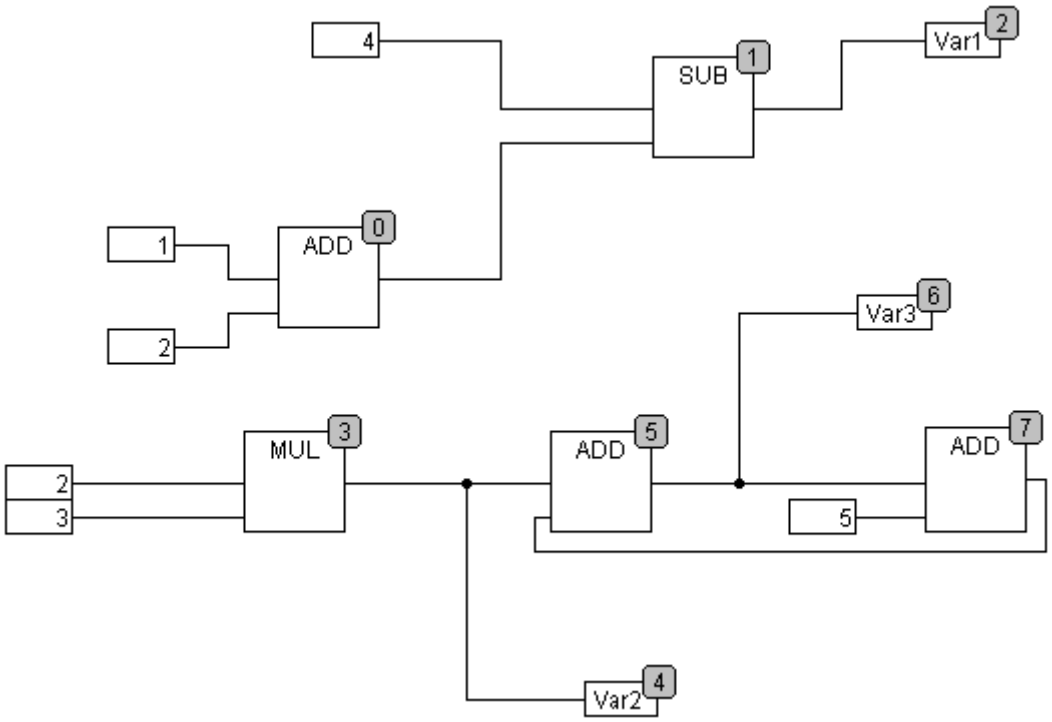
Mit diesem Befehl werden alle selektierten Elemente an das Ende der Abarbeitungsreihenfolge geschoben, wobei die Reihenfolge innerhalb der selektierten Elemente beibehalten wird. Ebenso bleibt die Reihenfolge innerhalb der nicht selektierten Elemente beibehalten.

'Extras' 'Reihenfolge' 'Alles nach Datenfluss anordnen'

Dieser Befehl wird **auf alle** Elemente angewandt. Die Abarbeitungsreihenfolge wird bestimmt vom Datenfluss der Elemente und nicht von Ihrer Lage.
Untenstehende Abbildung zeigt Elemente, die topologisch angeordnet sind.



Nach Anwahl des Befehls ergibt sich folgende Anordnung:



Bei Anwahl des Befehls werden zunächst alle Elemente topologisch sortiert. Danach wird eine neue Abarbeitungsliste zusammengestellt. Ausgehend von den bekannten Werten der Eingänge wird ermittelt, welche der noch nicht nummerierten Elemente als nächstes abgearbeitet werden kann. Im oberen "Netzwerk" kann z.B. der Baustein ADD sofort abgearbeitet werden, da die Werte, die an seinen Eingängen anliegen (1 und 2) bekannt sind. Erst danach kann der Baustein SUB abgearbeitet werden, da das Ergebnis von ADD bekannt sein muss usw.
Rückkopplungen werden allerdings als letztes eingefügt.
Der Vorteil der datenflussmäßigen Reihenfolge ist, dass eine Ausgangsbox, die mit dem Ausgang eines

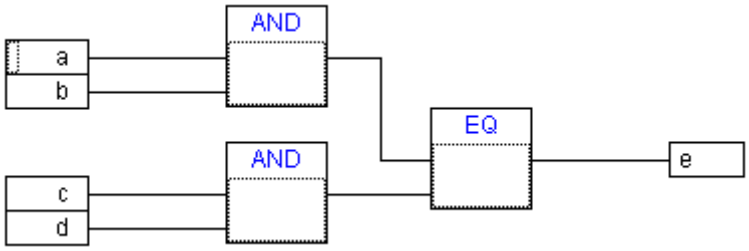
Bausteins verbunden ist, in der Datenflussreihenfolge unmittelbar auf diesen folgt, was bei der topologischen Anordnung nicht immer der Fall ist. Die topologische Reihenfolge liefert unter Umständen also ein anderes Ergebnis als die Reihenfolge nach Datenfluss, wie man an den obigen Beispielen erkennt.

'Extras' 'Makro erzeugen'

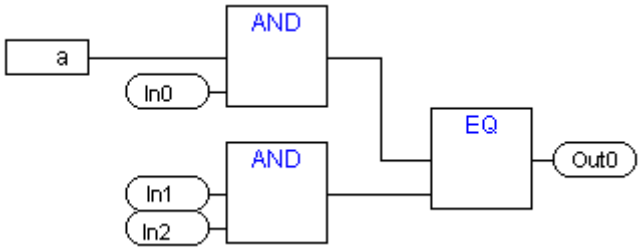
Mit diesem Befehl können mehrere Bausteine, die gleichzeitig selektiert sind, zu einem Block zusammengefasst werden, der als Makro mit einem Namen versehen werden kann. Makros können nur über Kopieren/Einfügen vervielfältigt werden, wobei jede Kopie ein eigenes Makro darstellt, dessen Namen unabhängig gewählt werden kann. Makros sind somit keine Referenzen. Alle Verbindungen, die durch die Erzeugung des Makros "gekappt" werden, erzeugen In- bzw. Out-Pins am Makro. Verbindungen zu Inputs erzeugen einen In-Pin. Als Name neben dem Pin erscheint ein Default-Name der Form In<n>. Für Verbindungen zu Outputs erscheint Out<n>. Betroffene Verbindungen, welche vor der Erzeugung des Makros Verbindungsmarken hatten, erhalten die Verbindungsmarke am PIN des Makros.

Ein Makro erhält zunächst den Default-Namen "MAKRO". Dieser kann im Namensfeld der Makro-Verwendung geändert werden. Wird das Makro editiert, so wird der Name des Makros in der Titelleiste des Editorfensters an den Bausteinnahmen angehängt angezeigt.

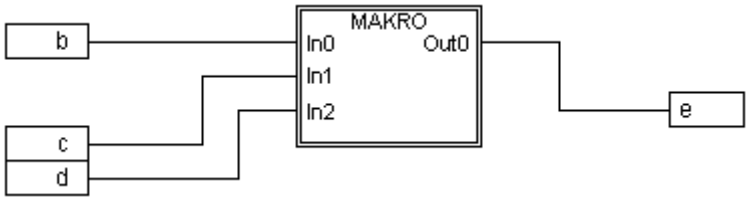
Beispiel:
Selektion



Makro



Im Editor:



'Extras' 'In Makro springen'

Durch diesen Befehl oder durch Doppelklick auf den Rumpf des Makros wird das Makro im Editorfenster des zugehörigen Bausteins zum Bearbeiten geöffnet. Der Name des Makros wird angehängt an den Bausteinnamen in der Titelleiste angezeigt.

Die bei der Erstellung erzeugten Pin-Boxen für die Ein- und Ausgänge des Makros können wie die normalen Baustein-Ein- und Ausgänge. Sie können also verschoben, gelöscht, hinzugefügt etc. werden. Sie unterscheiden sich lediglich in der Darstellung und besitzen keinen Positionsindex. Zum Hinzufügen können

Sie die Schaltflächen (Eingang) bzw. (Ausgang) benützen, die in der Symbolleiste angeboten werden. Pin-Boxen haben abgerundete Ecken. Der Text der Pin-Box entspricht dem Namen des Pins in der Makrodarstellung.

Die Reihenfolge der Pins an der Makro-Box richtet sich nach der Abarbeitungsreihenfolge der Elemente des Makros. Niedriger Reihenfolgeindex vor hohem, oberer Pin vor unterem.

Die Abarbeitungsreihenfolge innerhalb des Makros ist geschlossen, d.h. das Makro wird als ein Block gerechnet, und zwar an der Position des Makros im übergeordneten Baustein. Die Befehle zur Manipulation der Reihenfolge wirken sich somit nur innerhalb des Makros aus.

'Extras' 'Makro expandieren'

Mit diesem Befehl wird das selektierte Makro wieder expandiert und die enthaltenen Elemente an der Position des Makros im Baustein eingefügt. Die Verbindungen zu den Pins des Makros werden wieder als Verbindungen zu den Ein- bzw. Ausgängen der Elemente dargestellt. Kann die Expansion des Makros aus Platzmangel nicht an der Position der Makrobox erfolgen, so wird das Makro solange nach rechts und unten verschoben, bis genügend Platz zur Verfügung steht.

'Extras' 'Eine Makroebene zurück', 'Extras' 'Alle Makroebenen zurück'

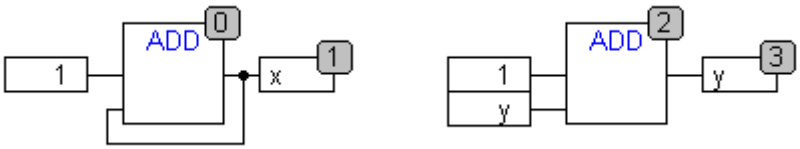
Diese Befehle stehen auch in der Symbolleiste zur Verfügung, sobald ein Makro zur Bearbeitung geöffnet ist. Sind Makros ineinander geschachtelt, kann in die darüber liegende bzw. in die oberste Darstellungsebene zurückgeschaltet werden.

Rückkopplungen

Im freigraphischen Funktionsplaneditor können im Gegensatz zum gewöhnlichen Funktionsplaneditor Rückkopplungen direkt dargestellt werden. Dabei muss beachtet werden, dass für den Ausgang eines Bausteins generell eine interne Zwischenvariable angelegt wird. Bei Operatoren ergibt sich der Datentyp der Zwischenvariable aus dem größten Datentyp der Eingänge.

Der Datentyp einer Konstanten ermittelt sich aus dem kleinstmöglichen Datentyp, d.h. für die Konstante '1' wird der Datentyp SINT angenommen. Wird nun eine Addition mit Rückkopplung und der Konstante '1' durchgeführt, so liefert der erste Eingang den Datentyp SINT und der zweite ist aufgrund der Rückkopplung undefiniert. Somit ist die Zwischenvariable auch vom Typ SINT. Der Wert der Zwischenvariable wird erst danach der Ausgangsvariablen zugewiesen.

Untenstehende Abbildung zeigt einmal eine Addition mit Rückkopplung und einmal direkt mit einer Variablen. Die Variablen x und y sollen dabei vom Typ INT sein.



Zwischen den beiden Additionen gibt es Unterschiede:
Die Variable y kann mit einem Wert ungleich 0 initialisiert werden, die Zwischenvariable der linken Addition jedoch nicht.
Die Zwischenvariable der linken Addition hat den Datentyp SINT, die der rechten den Datentyp INT. Die Variablen x und y haben ab dem 129ten Aufruf unterschiedliche Werte. Die Variable x, obwohl vom Typ INT, erhält den Wert -127, weil die Zwischenvariable einen Überlauf hat. Die Variable y erhält dagegen den Wert 129.

'Extras' 'Instanz öffnen'

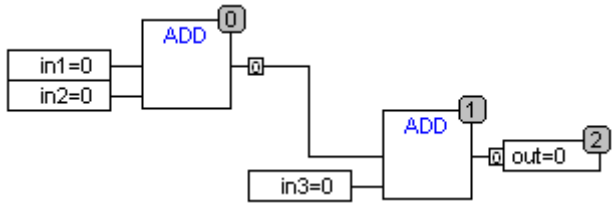
Dieser Befehl entspricht dem Befehl '**Projekt' 'Instanz öffnen'.**

CFC im Online Modus'

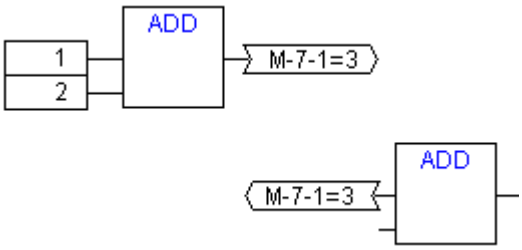
Monitoring:

Die Werte für Eingänge und Ausgänge werden innerhalb der Input- bzw. Output-Boxen dargestellt. Konstanten werden nicht gemonitort. Für nicht boolsche Variablen werden die Boxen entsprechend den angezeigten Werten vergrößert. Für boolsche Verbindungen werden der Variablenname sowie die

Verbindung blau dargestellt, wenn der Wert TRUE ist, ansonsten bleiben sie schwarz.
Interne boolsche Verbindungen werden Online ebenfalls im Zustand TRUE blau angezeigt, ansonsten schwarz. Der Wert von internen nicht boolschen Verbindungen wird in einer kleinen Box mit abgerundeten Ecken am Ausgangs-Pin der Verbindung angezeigt.



PINs in Makros werden wie Ein- bzw. Ausgangsboxen gemonitort.



Nicht boolsche Verbindungen mit Verbindungsmarken zeigen Ihren Wert innerhalb der Verbindungsmarke an. Für boolsche Verbindungen werden die Leitungen sowie die Markennamen wiederum blau dargestellt, falls die Leitung TRUE führt, ansonsten schwarz.

Ablaufkontrolle:

Bei eingeschalteter Ablaufkontrolle werden die durchlaufenen Verbindungen mit der in den Projekt-Optionen eingestellten Farbe markiert.

Breakpoints:

Haltepunkte können auf alle Elemente gesetzt werden, die auch einen Abarbeitungsreihenfolgen-Index besitzen. Die Abarbeitung des Programms wird vor dem Ausführen des jeweiligen Elements angehalten, d.h. für Bausteine und Ausgänge vor dem Zuweisen der Eingänge, für Sprungmarken vor dem Ausführen des Elements mit dem nächsten Index. Als Haltepunktposition im Breakpoint-Dialog wird der Abarbeitungsreihenfolgen-Index des Elements verwendet.

Das Setzen der Haltepunkte erfolgt auf ein selektiertes Element mit der Taste F9 oder über den Menüpunkt 'Breakpoint an/aus', im 'Online'- oder 'Extras'-Menü oder im Kontext-Menü des Editors. Ist auf einem Element ein Haltepunkt gesetzt, so wird mit dem nächsten Ausführen des Befehls 'Breakpoint an/aus' dieser wieder gelöscht und umgekehrt. Zusätzlich kann der Haltepunkt auf einem Element durch Doppelklick auf dieses getoggelt werden.

Die Darstellung der Breakpoints erfolgt mit den in den Projekt-Optionen eingestellten Farben.

RETURN-Marke:

Im Online-Modus wird automatisch eine Sprungmarke mit der Bezeichnung "RETURN" in der ersten Spalte und nach dem letzten Element im Editor erzeugt. Diese Marke markiert das Ende des Bausteins und wird beim Steppen vor dem Verlassen des Bausteins angesprungen. In Makros werden keine RETURN-Marken eingefügt.

Steppen:

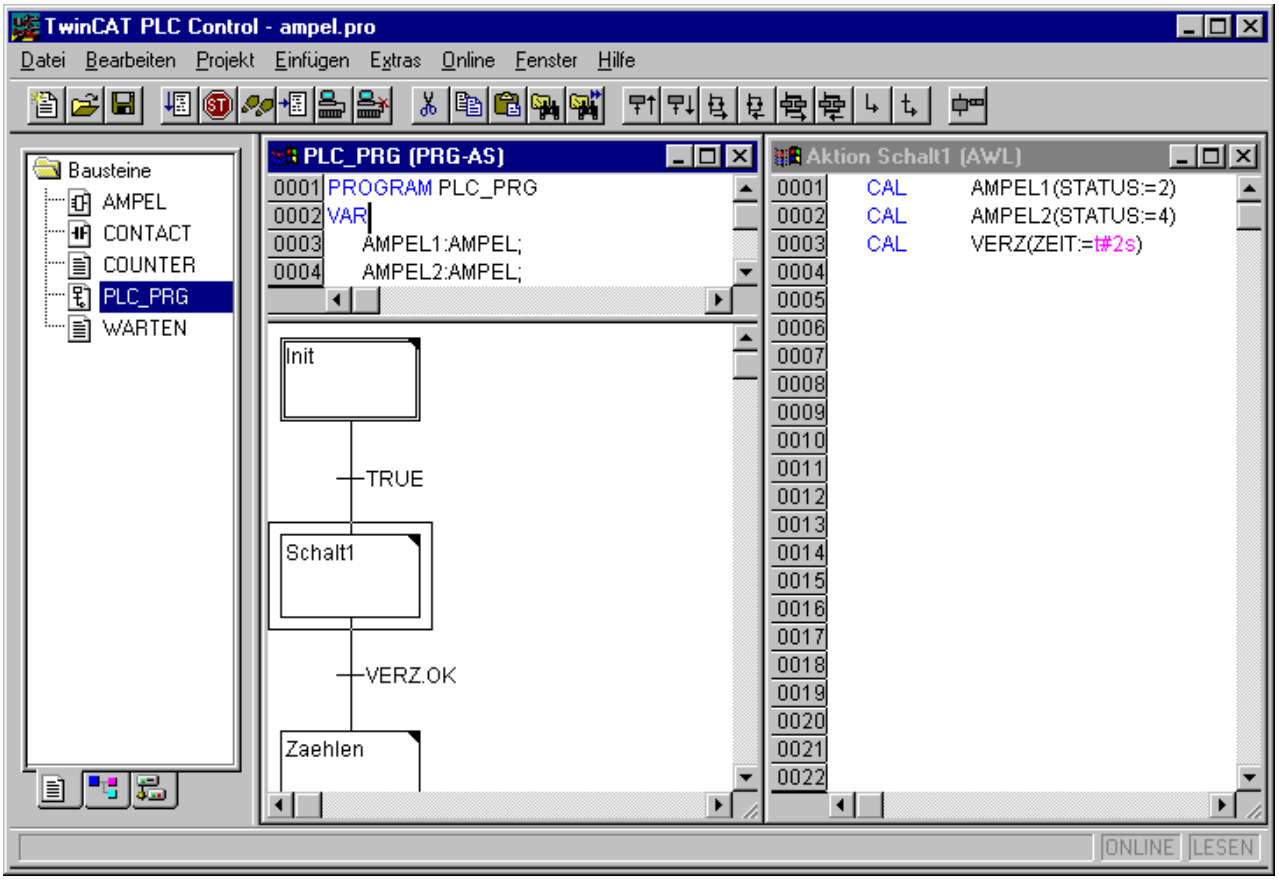
Bei 'Einzelschritt über' wird immer zum Element mit dem nächsthöheren Reihenfolgen-Index gesprungen. Ist das aktuelle Element ein Makro oder ein Baustein, so wird bei 'Einzelschritt in' in die Implementierung desselben verzweigt. Wird von dort ein 'Einzelschritt über' durchgeführt, wird auf das Element weitergesprungen, dessen Reihenfolgen-Index dem des Makros folgt.

'Extras' 'Zoom' Kurzform: <Alt> + <Enter>

Mit diesem Befehl kann die Implementierung eines Bausteins geöffnet werden, wenn der Baustein selektiert ist.

5.9 Ablaufspracheneditor

So sieht ein in AS geschriebener Baustein im TwinCAT PLC Control-Editor aus:



Alle Editoren für Bausteine bestehen aus einem Deklarationsteil und einem Rumpf. Diese sind getrennt durch einen Bildschirmteiler.

Der Ablaufspracheneditor ist ein graphischer Editor. Die wichtigsten Befehle finden Sie im Kontextmenü (rechte Maustaste). Tooltips zeigen sowohl im Offline- als auch im Online Modus und gezoomtem Status die vollständigen Namen bzw. Ausdrücke von Schritten, Transitionen, Sprüngen, Sprungmarken, Qualifiern oder assoziierten Aktionen.

Für Informationen über die Ablaufsprache, siehe Kapitel Ablaufsprache(AS) [▶ 160].

Der Editor für die Ablaufsprache muss auf die Besonderheiten von AS eingehen. Dazu dienen die folgenden Menüpunkte:

Blöcke markieren im AS

Ein markierter Block ist eine Menge von AS-Elementen, die von einem gepunkteten Rechteck umgeben sind. (Im Beispiel oben etwa ist der Schritt Schalt1 markiert.)

Man kann ein Element (einen Schritt, eine Transition oder einen Sprung) auswählen, indem man den Mauszeiger auf dieses Element setzt, und die linke Maustaste drückt, oder indem man die Pfeiltasten benützt. Um eine Menge von mehreren Elementen zu markieren, drücken Sie zusätzlich zu einem bereits markierten Block die <Umschalttaste>, und wählen das Element in der linken oder rechten unteren Ecke der Menge aus. Die sich ergebende Auswahl ist die kleinste zusammenhängende Menge von Elementen, die diese beiden Elemente beinhaltet.

Beachten Sie, dass alle Befehle nur dann ausgeführt werden können, wenn sie nicht den Sprachkonventionen widersprechen.

'Einfügen"Schritt-Transition (davor)' Kurzform: <Strg>+<T>

Dieser Befehl fügt im AS-Editor einen Schritt gefolgt von einer Transition vor dem markierten Block ein.

'Einfügen''Schritt-Transition (danach)' Kurzform: <Strg>+<E>

Dieser Befehl fügt im AS-Editor einen Schritt gefolgt von einer Transition nach der ersten Transition im markierten Block ein.

'Einfügen' 'Alternativzweig (rechts)' Kurzform: <Strg>+<A>

Dieser Befehl fügt im AS-Editor eine Alternativverzweigung als rechte Verzweigung des markierten Blocks ein. Der markierte Block muss hierfür mit einer Transition beginnen und enden. Der neue Zweig besteht dann aus einer Transition.

'Einfügen' 'Alternativzweig (links)'

Dieser Befehl fügt im AS-Editor eine Alternativverzweigung als linke Verzweigung des markierten Blocks ein. Der markierte Block muss hierfür mit einer Transition beginnen und enden. Der neue Zweig besteht dann aus einer Transition.

'Einfügen' 'Parallelzweig (rechts)' Kurzform: <Strg>+<L>

Dieser Befehl fügt im AS-Editor eine parallele Verzweigung als rechte Verzweigung des markierten Blocks ein. Der markierte Block muss hierfür mit einem Schritt beginnen und enden. Der neue Zweig besteht dann aus einem Schritt. Um Sprünge auf die entstandene Parallelverzweigung zu ermöglichen, muss sie mit einer Sprungmarke versehen werden.

'Einfügen' 'Parallelzweig (links)'

Dieser Befehl fügt im AS-Editor eine parallele Verzweigung als linke Verzweigung des markierten Blocks ein. Der markierte Block muss hierfür mit einem Schritt beginnen und enden. Der neue Zweig besteht dann aus einem Schritt. Um Sprünge auf die entstandene Parallelverzweigung zu ermöglichen, muss sie mit einer Sprungmarke (siehe 'Extras' 'Marke zu Parallelzweig hinzufügen') versehen werden.

'Einfügen' 'Sprung' Kurzform: <Strg>+<U>

Dieser Befehl fügt im AS-Editor einen Sprung am Ende des Zweigs ein, zu dem der markierte Block gehört. Die Verzweigung muss hierfür eine Alternativverzweigung sein. Zu einem eingefügten Sprung kann anschließend der eingetragene Text 'Step' selektiert und durch den Schrittnamen, zu dem gesprungen werden soll, ersetzt werden.

'Einfügen' 'Transition-Sprung'

Dieser Befehl fügt im AS-Editor eine Transition gefolgt von einem Sprung am Ende der ausgewählten Verzweigung ein. Die Verzweigung muss hierfür eine parallele Verzweigung sein.

Zu einem eingefügten Sprung kann anschließend der eingetragene Text 'Step' selektiert und durch den Schrittnamen bzw. die Sprungmarke einer Parallelverzweigungen, zu dem/der gesprungen werden soll, ersetzt werden.

'Einfügen' 'Eingangsaktion hinzufügen'

Mit diesem Befehl können Sie zu einem Schritt eine Eingangsaktion hinzufügen. Eine Eingangsaktion wird nur einmal ausgeführt, gleich nachdem der Schritt aktiv geworden ist. Die Eingangsaktion kann in einer beliebigen Sprache implementiert werden.

Ein Schritt mit Eingangsaktion wird durch ein 'E' in der linken unteren Ecke gekennzeichnet.

'Einfügen' 'Ausgangsaktion hinzufügen'

Mit diesem Befehl können Sie einem Schritt eine Ausgangsaktion hinzufügen. Eine Ausgangsaktion wird nur einmal ausgeführt, bevor der Schritt deaktiviert wird. Die Ausgangsaktion kann in einer beliebigen Sprache implementiert werden.

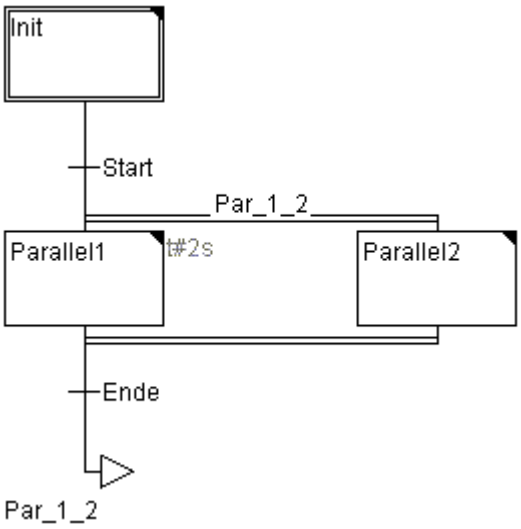
Ein Schritt mit Ausgangsaktion wird durch ein 'X' in der rechten unteren Ecke gekennzeichnet.

'Extras' 'Paralellzweig einfügen (rechts)'

Dieser Befehl fügt den Inhalt der Zwischenablage als rechte Parallelverzweigung des markierten Blocks ein. Dafür muss der markierte Block mit einem Schritt beginnen und enden. Der Inhalt der Zwischenablage muss ebenfalls ein AS-Block sein, der mit einem Schritt beginnt und endet.

'Extras' 'Marke zu Parallelzweig hinzufügen'

Um eine neu eingefügte Parallelverzweigung mit einer Sprungmarke zu versehen, muss die vor der Parallelverzweigung liegende Transition markiert werden und der Befehl 'Marke zu Parallelzweig hinzufügen' ausgeführt werden. Daraufhin wird die Parallelverzweigung mit einem Standardnamen "Parallel" und einer angehängten laufenden Nummer versehen, die nach den Regeln für Bezeichnernamen editiert werden können. Im nachfolgenden Beispiel wurde "Parallel" durch "Par_1_2" ersetzt und der Sprung nach Transition "Ende" auf diese Sprungmarke gelenkt.



Sprungmarke löschen

Eine Sprungmarke wird durch Löschen des Sprungmarken-Textes gelöscht.
'Extras' 'Einfügen danach'

Dieser Befehl fügt den AS-Block in der Zwischenablage nach dem ersten Schritt bzw. der ersten Transition des markierten Blocks ein (normales Kopieren fügt ihn vor dem markierten Block ein). Das wird nur dann ausgeführt, wenn die resultierende AS-Struktur nach den Sprachnormen korrekt ist.

'Extras' 'Zoom Aktion/Transition' Kurzform: <Alt>+<Eingabetaste>

Die Aktion des ersten Schritts des markierten Blocks bzw. der Transitionsrumpf der ersten Transition des markierten Blocks wird in der jeweiligen Sprache, in der er geschrieben ist, in den Editor geladen. Wenn die Aktion oder der Transitionsrumpf leer ist, dann muss die Sprache ausgewählt werden, in der er geschrieben werden soll.

'Extras' 'Lösche Aktion/Transition'

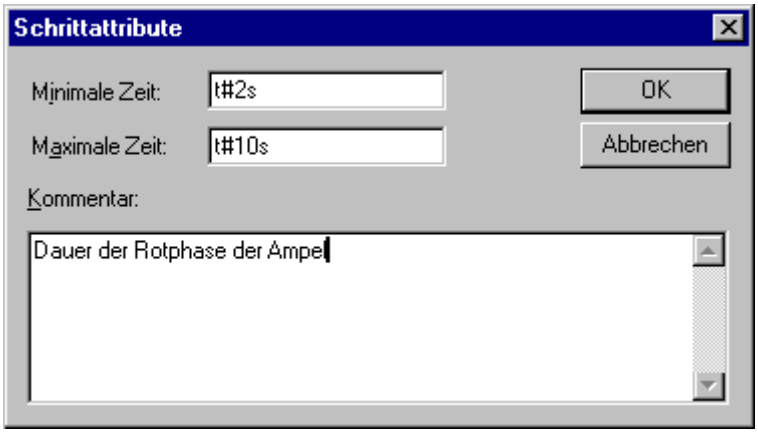
Mit diesem Befehl können Sie die Aktionen des ersten Schritts des markierten Blocks bzw. der Transitionsrumpf der ersten Transition des markierten Blocks löschen.

Ist bei einem Schritt nur entweder die Aktion, die Eingangsaktion oder die Ausgangsaktion implementiert, so wird diese mit dem Befehl gelöscht. Andernfalls erscheint ein Dialog, in dem gewählt werden kann, welche Aktion bzw. Aktionen gelöscht werden sollen.

Steht der Cursor in einer Aktion eines IEC-Schritts, wird nur diese Assoziation gelöscht. Ist ein IEC-Schritt mit einer asoziierten Aktion selektiert, so wird diese Assoziation gelöscht. Bei einem IEC-Schritt mit mehreren Aktionen erscheint ein Dialog zur Auswahl.

'Extras' 'Schritt Attribute'

Mit diesem Befehl öffnen Sie einen Dialog, in dem Sie Attribute zu dem markierten Schritt editieren können.

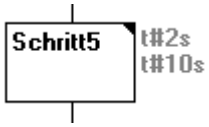


Dialog zum Editieren von Schrittattributen

Sie können drei verschiedene Einträge im Schrittattribute-Dialog vornehmen. Unter **Minimale Zeit** geben Sie die Zeitdauer ein, die die Abarbeitung dieses Schritts mindestens dauern soll. Unter **Maximale Zeit** geben Sie die Zeitdauer ein, die die Abarbeitung des Schrittes höchstens dauern soll. Beachten Sie, dass die Einträge vom Typ **TIME** sind , d.h. verwenden Sie eine TIME-Konstante (z.B. T#3s) oder eine Variable vom Typ TIME.

Unter **Kommentar** können Sie einen Kommentar zum Schritt eingeben. Im Dialog 'Ablaufsprachen Optionen', den Sie über **'Extras' 'Optionen'** öffnen, können Sie dann einstellen, ob im AS-Editor die Kommentare oder die Zeiteinstellung zu Ihren Schritten dargestellt werden soll. Rechts neben dem Schritt erscheint dann entweder der Kommentar oder die Zeiteinstellungen.

Bei Überschreiten der Maximalzeit werden AS-Flags [▶ 163] gesetzt, die der Benutzer abfragen kann.



Im Beispiel ist ein Schritt dargestellt, dessen Ausführung mindestens zwei und höchstens zehn Sekunden dauern soll. Im Online Modus wird zusätzlich zu diesen beiden Zeiten angezeigt, wie lange der Schritt bereits aktiv ist.

AS-Flags

Wenn im AS ein Schritt länger aktiv ist, als in seinen Attributen angegeben, dann werden einige spezielle Flags gesetzt. Darüber hinaus gibt es weitere Variablen, die Sie definieren können, um den Ablauf in der Ablaufsprache zu steuern. Um die Flags zu benutzen, müssen Sie sie global oder lokal, als Aus- oder Eingabevariable deklarieren.

SFCEnableLimit: Diese spezielle Variable ist vom Typ BOOL. Wenn sie TRUE ist, werden Zeitüberschreitungen bei den Schritten in SFCError registriert. Ansonsten werden Zeitüberschreitungen ignoriert.

SFCInit: Wenn diese boolsche Variable TRUE ist, dann wird die Ablaufsprache auf den Init-Schritt zurückgesetzt. Die anderen AS-Flags werden ebenfalls zurückgesetzt (Initialisierung). Solange die Variable TRUE ist, bleibt der Init-Schritt gesetzt (aktiv), wird aber nicht ausgeführt. Erst wenn SFCInit wieder auf FALSE gesetzt wird, wird der Baustein normal weiterbearbeitet.

SFCReset: Diese Variable vom Typ BOOL verhält sich ähnlich wie SFCInit. Im Unterschied zu dieser wird allerdings nach der Initialisierung der Initschritt weiter abgearbeitet. Dies könnte beispielsweise dazu benützt werden, um im Initschritt das SFCReset-Flag gleich wieder auf FALSE zu setzen

SFCQuitError: Solange diese boolsche Variable TRUE ist, wird die Abarbeitung des AS-Diagramms angehalten, eine eventuelle Zeitüberschreitung in der Variablen SFCError wird dabei zurückgesetzt. Wenn die Variable wieder auf FALSE gesetzt wird, werden alle bisherigen Zeiten in den aktiven Schritten zurückgesetzt.

SFCPause: Solange diese boolsche Variable TRUE ist, wird die Abarbeitung des AS-Diagramms angehalten.

SFCTrans: Diese boolsche Variable wird TRUE, wenn eine Transition schaltet.

SFCError: Diese boolsche Variable wird TRUE, wenn in einem AS-Diagramm eine Zeitüberschreitung aufgetreten ist. Wenn im Programm nach der ersten Zeitüberschreitung eine weitere auftritt, wird diese nicht mehr registriert, wenn die Variable SFCError vorher nicht wieder zurückgesetzt wurde.

SFCErrorStep: Diese Variable ist vom Typ STRING. Wird durch SFCError eine Zeitüberschreitung registriert, wird in dieser Variable der Name des Schritts gespeichert, der die Zeitüberschreitung verursacht hat.

SFCErrorPOU: Diese Variable vom Typ STRING erhält im Falle einer Zeitüberschreitung den Namen des Bausteins, in dem die Zeitüberschreitung aufgetreten ist.

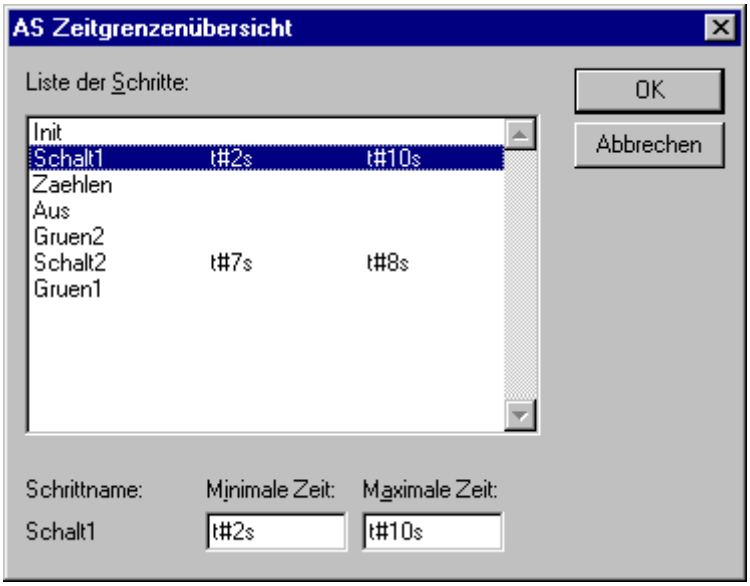
SFCCurrentStep: Diese Variable ist vom Typ STRING. In dieser Variablen wird der Name des Schritts gespeichert, der aktiv ist, unabhängig von der Zeitüberwachung. Bei einer Parallelverzweigung wird der Schritt im äußersten rechten Zweig gespeichert.
Wenn einmal eine Zeitüberschreitung aufgetreten ist, und die Variable SFCError nicht wieder zurückgesetzt wurde, wird keine weitere Zeitüberschreitung registriert.

SFCErrorAnalyzation: Diese Variable vom Typ STRING gibt den Transitionsausdruck bzw. jede Variable eines zusammengesetzten Ausdrucks aus, welche zu einem FALSE der Transition führt und damit zu einer Zeitüberschreitung im vorangehenden Schritt. Voraussetzung dafür ist die Deklaration des Flags SFCError, das die Zeitüberschreitung registriert. SFCErrorAnalyzation greift auf eine Funktion AppendErrorString der Bibliothek **TcSystem.Lib** zurück. Der Ausgabestring trennt mehrere Komponenten durch das Zeichen "|".

SFCTip, SFCTipMode: Diese Variablen vom Typ BOOL erlauben den Tipp-Betrieb des SFC. Wenn dieser durch SFCTipMode=TRUE eingeschaltet ist, kann nur zum nächsten Schritt weitergeschaltet werden, indem SFCTip auf TRUE gesetzt wird. Solange SFCTipMode auf FALSE gesetzt ist, kann zusätzlich auch über die Transitionen weitergeschaltet werden.

'Extras' 'Zeitenüberblick'

Mit diesem Befehl öffnen Sie ein Fenster, in dem Sie die Zeiteinstellungen Ihrer AS-Schritte editieren können:



Zeitgrenzenübersicht zu einem AS-Baustein

In der Zeitgrenzenübersicht werden alle Schritte Ihres AS-Bausteins dargestellt. Wenn Sie zu einem Schritt eine Zeitbegrenzung angegeben haben, dann wird diese rechts vom Schritt angezeigt (zuerst die Untergrenze, dann die Obergrenze). Außerdem können Sie die Zeitbegrenzungen editieren. Klicken Sie

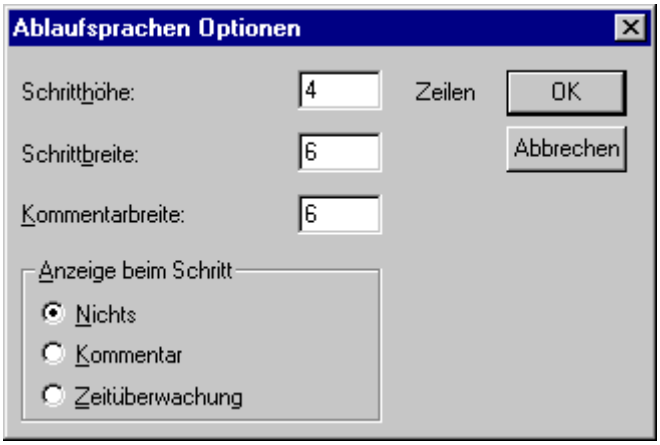
dazu in der Übersicht auf den gewünschten Schritt. Der **Schrittname** wird dann unten im Fenster angezeigt, gehen Sie in das Feld **Minimale Zeit** oder **Maximale Zeit**, und geben Sie dort die gewünschte Zeitbegrenzung ein.

Beachten Sie, dass die Einträge vom Typ **TIME** sind, d.h. verwenden Sie eine TIME-Konstante (z.B. T#3s) oder eine Variable vom Typ **TIME**. Wenn Sie das Fenster mit OK schließen, werden alle Veränderungen abgespeichert.

Im Beispiel haben die Schritte 2 und 6 eine Zeitbegrenzung. Schalt1 dauert mindestens zwei und höchstens zehn Sekunden. Schalt2 dauert mindestens sieben und höchstens acht Sekunden.

'Extras' 'Optionen'

Mit diesem Befehl öffnen Sie einen Dialog, in dem Sie verschiedene Optionen zu Ihrem AS-Baustein einstellen können.



Im AS-Optionen-Dialog können Sie fünf Einträge vornehmen. Unter **Schritthöhe** können Sie eingeben, wie viele Zeilen ein AS-Schritt in Ihrem AS-Editor hoch sein soll. 4 ist hier die Standardeinstellung. Unter **Schrittbreite** können Sie eingeben wie viele Spalten ein Schritt breit sein soll. 6 ist hier die Standardeinstellung. Die **Kommentarbreite** definiert die Anzahl der Spalten, die dargestellt werden, wenn Sie den Kommentar beim Schritt mit anzeigen lassen.

Unter **Anzeige beim Schritt** können Sie einstellen, welche der Eingaben, die Sie unter '**Extras' 'Schritt Attribute**' gemacht haben, angezeigt werden sollen. Sie können **Nichts** anzeigen lassen, den **Kommentar** oder die **Zeitüberwachung**.

'Extras' 'Aktion assoziieren'

Mit diesem Befehl können Aktionen und boolsche Variablen zu IEC-Schritten assoziiert werden. Rechts neben den IEC-Schritt wird ein weiteres zweigeteiltes Kästchen für die Assoziation einer Aktion angehängt. Vorbelegt ist es im linken Feld mit dem Qualifier 'N' und dem Namen 'Action'. Beide Vorbelegungen können geändert werden. Dazu können Sie die Eingabehilfe benutzen.

Neue Aktionen für IEC-Schritte werden im Object Organizer zu einem AS-Baustein mit dem Befehl '**Projekt' 'Aktion hinzufügen'** angelegt.

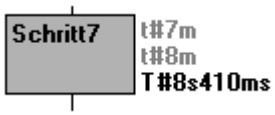
'Extras' 'IEC-Schritte benutzen'

Ist dieser Befehl aktiviert (erkennbar am Haken vor dem Menüpunkt und am gedrückten Symbol in der Funktionsleiste), werden beim Einfügen von Schritt-Transitionen und Parallelzweigen statt den vereinfachten Schritten IEC-Schritte eingefügt.

Ist diese Option gewählt, wird beim Anlegen eines AS-Bausteins der Init-Schritt als IEC-Schritt angelegt.

Die Ablaufsprache im Online Modus

Beim Ablaufspracheneditor werden im Onlinebetrieb die aktuell aktiven Schritte als blaue Schritte angezeigt. Wenn Sie es unter '**Extras' 'Optionen**' eingestellt haben, dann wird neben den Schritten die Zeitüberwachung dargestellt. Unter den von Ihnen eingegebenen Unter- und Obergrenzen erscheint eine dritte Zeitangabe von der Sie ablesen können, wie lange der Schritt bereits aktiv ist.



Im Bild ist der abgebildete Schritt bereits seit 8 Sekunden und 410 Millisekunden aktiv. Er muss aber mindestens 7 Minuten aktiv sein, bevor der Schritt verlassen wird.

Mit '**Online' 'Breakpoint an/aus**' kann ein Breakpoint auf einen Schritt gesetzt werden, außerdem in einer Aktion an den für die verwendete Sprache zugelassenen Stellen. Die Bearbeitung hält dann vor Ausführung dieses Schrittes bzw. Programmstelle der Aktion an. Schritte bzw. Programmstellen, auf die ein Breakpoint gesetzt ist, sind hellblau markiert.

Sind in einer Parallelverzweigung mehrere Schritte aktiv, so wird der aktive Schritt, dessen Aktion als nächstes bearbeitet wird, rot dargestellt.
Wurden IEC-Schritte verwendet, werden alle aktiven Aktionen im Onlinebetrieb blau dargestellt.

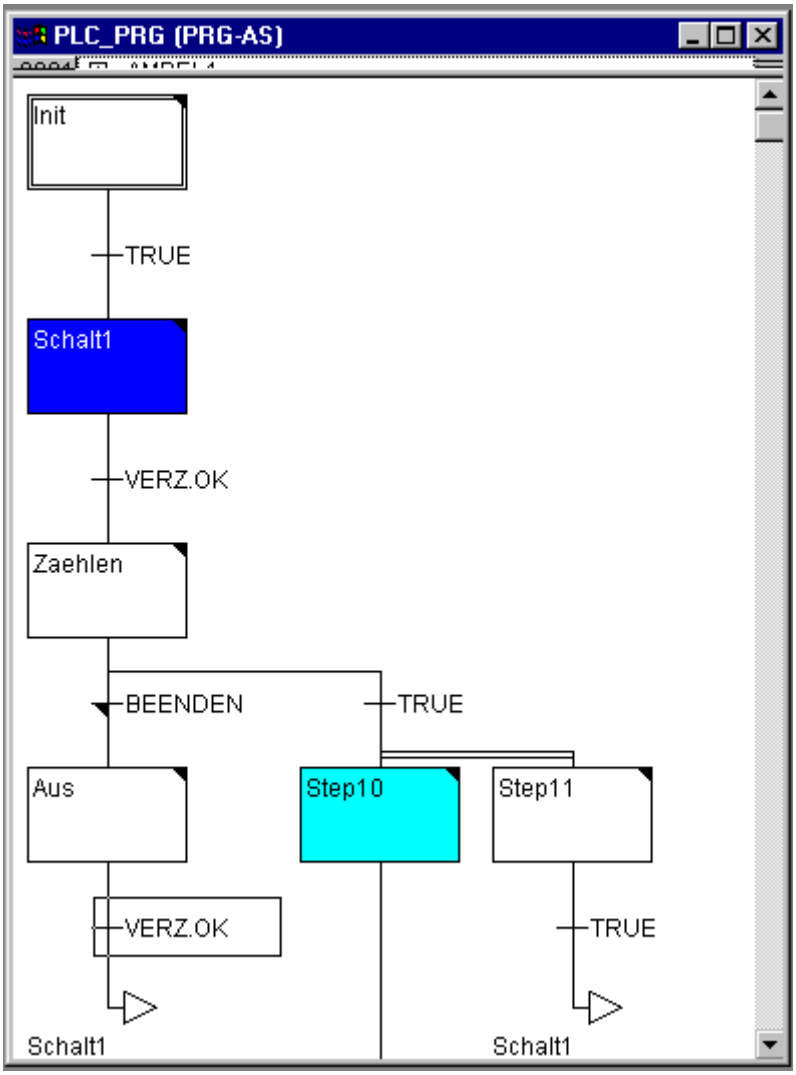
Auch in AS wird ein schrittweises Steppen unterstützt:

Mit dem Befehl '**Online' 'Einzelschritt über**' wird stets zum nächsten Schritt gesteppt, dessen Aktion ausgeführt wird. Ist die aktuelle Position

- ein Schritt in einem linearen Ablauf eines Bausteins oder ein Schritt im rechtesten Parallelzweig eines Bausteins, so wird der AS-Baustein verlassen und zum Aufrufer zurückgekehrt. Ist der Baustein das Hauptprogramm, beginnt der nächste Zyklus.
- ein Schritt im nicht rechtesten Zweig einer Parallelverzweigung, so wird zum aktiven Schritt im nächsten Parallelzweig gesprungen.
- die letzte Breakpoint-Position innerhalb einer 3S-Aktion, so wird zum Aufrufer des SFC gesprungen.
- die letzte Breakpoint-Position innerhalb einer IEC-Aktion, so wird zum Aufrufer des SFC gesprungen.
- die letzte Breakpoint-Position innerhalb einer Eingangsaktion/oder Ausgangsaktion, so wird zum ersten aktiven Schritt gesprungen.

Mit '**Online' 'Einzelschritt in**' kann zusätzlich in Aktionen hineingesteppt werden. Soll in eine Eingangs-, Ausgangs- oder IEC-Aktion gesprungen werden, muss dort ein Breakpoint gesetzt sein. Innerhalb der Aktionen stehen dem Anwender alle Debugging-Funktionalitäten des entsprechenden Editors zur Verfügung.

Wenn Sie den Mauszeiger im Deklarationseditor eine kurze Zeit über einer Variablen halten, wird der Typ, die Adresse und der Kommentar der Variablen in einem Tooltip angezeigt



Ablaufsprache im Online Modus mit einem aktiven Schritt (Schalt1) und einem Breakpoint (Step10)

⚠ VORSICHT

Undefinierter Zustand

Wenn Sie einen Schritt umbenennen und Online Change durchführen während genau dieser Schritt aktiv ist, stoppt das Programm in undefiniertem Zustand.

Abarbeitungsreihenfolge der Elemente einer Schrittkette:

1. Zunächst werden alle Action Control Block Flags der IEC-Aktionen zurückgesetzt, die in dieser Schrittkette verwendet werden. (Nicht jedoch die Flags von IEC-Aktionen, die innerhalb von Aktionen aufgerufen werden).
2. Für alle Schritte wird in der Reihenfolge, die sie in der Schrittkette einnehmen (von oben nach unten und von links nach rechts) überprüft, ob die Bedingung für die Ausführung der Ausgangsaktion gegeben ist und gegebenenfalls diese ausgeführt.
3. Für alle Schritte wird in der Reihenfolge, die sie in der Schrittkette einnehmen, überprüft, ob die Bedingung für die Eingangsaktion gegeben ist und gegebenenfalls diese ausgeführt.
4. Für alle Schritte wird in der Reihenfolge, die sie in der Schrittkette einnehmen, folgendes durchgeführt:
   - Gegebenenfalls wird die abgelaufene Zeit in die dazugehörige Schrittvariable kopiert.
   - Gegebenenfalls wird eine Zeitüberschreitung überprüft und die AS-Error-Flags werden entsprechend bedient.
   - Bei Nicht-IEC-Schritten wird nun die dazugehörige Aktion ausgeführt.

5. Die IEC-Aktionen, die in der Schrittkette verwendet werden, werden in alphabetischer Reihenfolge ausgeführt. Dabei wird in zwei Durchläufen durch die Liste der Aktionen gegangen. Im ersten Durchlauf werden alle im aktuellen Zyklus deaktivierten IEC-Aktionen ausgeführt. Im zweiten Durchlauf werden alle im aktuellen Zyklus aktiven IEC-Aktionen ausgeführt.
6. Die Transitionen werden ausgewertet: Wenn der Schritt im aktuellen Zyklus aktiv war und die nachfolgende Transition TRUE liefert (und eventuell die minimal aktive Zeit bereits abgelaufen ist), dann wird der nachfolgende Schritt aktiviert.

Folgendes ist zur Implementierung von Aktionen zu beachten:

Es kann vorkommen, dass eine Aktion in einem Zyklus mehrfach ausgeführt wird, weil sie in mehreren Schrittketten assoziiert ist. (Beispielsweise könnte ein SFC zwei IEC-Aktionen A und B besitzen, die beide in SFC implementiert sind, und die beide die IEC-Aktion C aufrufen, dann können im selben Zyklus die IEC-Aktionen A und B aktiv sein und in beiden IEC-Aktionen kann wiederum die IEC-Aktion C aktiv sein, dann würde C zweimal aufgerufen).

Wird dieselbe IEC-Aktion gleichzeitig in verschiedenen Ebenen eines SFC verwendet, könnte dies durch die oben beschriebene Abarbeitungsreihenfolge zu unerwünschten Effekten führen. Deshalb wird in diesem Fall eine Fehlermeldung ausgegeben.

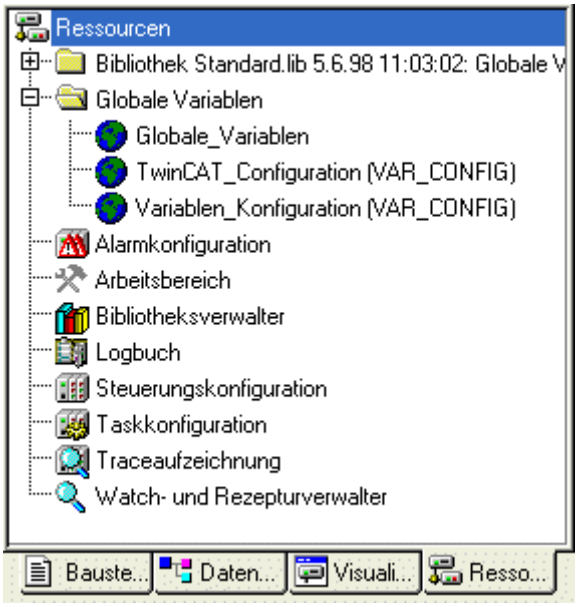
Möglicherweise kann dies bei der Bearbeitung von Projekten, die mit älteren Versionen von TwinCAT PLC Control erstellt wurden, auftreten.

Beim Monitoring von Ausdrücken (z.B. A AND B) in Transitionen wird nur der "Gesamtwert" der Transition dargestellt.

6 Die Ressourcen

In der Registerkarte Ressourcen des Object Organizers befinden sich Objekte zum Konfigurieren und Organisieren Ihres Projektes und zur Verfolgung von Variablenwerten:

- Globale Variablen [▶ 169], die im ganzen Projekt verwendet werden können, sowohl die Globalen Variablen des Projektes als auch die der eingebundenen Bibliotheken.
- Alarmkonfiguration [▶ 173] zum Konfigurieren von Alarmklassen und -gruppen, die dann beispielsweise in der Visualisierung zur Anzeige und Bedienung gebracht werden können.
- Arbeitsbereich als Überblick über alle aktuell eingestellten Projektoptionen
- Bibliotheksverwalter [▶ 192] zur Verwaltung aller ans Projekt angebundenen Bibliotheken
- Logbuch [▶ 118] zur chronologischen Aufzeichnung der Aktionen, die zwischen Ein- und Ausloggen ablaufen
- Steuerungskonfiguration [▶ 181] zum Konfigurieren Ihrer Hardware
- Taskkonfiguration [▶ 182] zur Steuerung Ihres Programms über Tasks
- Traceaufzeichnung [▶ 184] zur grafischen Aufzeichnung von Variablenwerten
- Watch- und Rezepturverwalter [▶ 189] zum Anzeigen und Vorbelegen von Variablenwerten



Ressourcen

6.1 Globale Variablen

Im Object Organizer befinden sich in der Registerkarte Ressourcen im Ordner Globale Variablen standardmäßig drei Objekte (in Klammern die vorbelegten Namen der Objekte):

- Globale Variablenliste
- Variablenkonfiguration

Alle in diesen Objekten definierte Variablen sind im ganzen Projekt bekannt. Ist der Ordner Globale Variablen nicht aufgeklappt (Pluszeichen vor dem Ordner), öffnen Sie ihn mit Doppelklick oder Drücken der <Eingabetaste> in der Zeile. Wählen Sie das entsprechende Objekt aus. Mit dem Befehl 'Objekt bearbeiten' öffnet ein Fenster mit den bisher definierten globalen Variablen.

Mehrere Variablenlisten

Wenn Sie eine große Anzahl globaler Variablen deklariert haben, und Sie Ihre globale Variablenliste besser strukturieren wollen, dann können Sie weitere Variablenlisten anlegen. Normale globale Variablen (**VAR_GLOBAL**) und Variablenkonfigurationen (**VAR_CONFIG**) müssen in getrennten Objekten definiert werden. Selektieren Sie im Object Organizer den Ordner Globale Variablen oder eines der bestehenden Objekte mit globalen Variablen und führen Sie den Befehl '**Projekt' 'Objekt einfügen**' aus. Geben Sie dem Objekt in der erscheinenden Dialogbox einen entsprechenden Namen. Damit wird eine weiteres Objekt mit dem Schlüsselwort VAR_GLOBAL angelegt. Wenn Sie ein Objekt mit Variablenkonfiguration haben möchten, ändern Sie das Schlüsselwort entsprechend in VAR_CONFIG.

Globale Variablen

Als globale Variablen können "normale" Variablen, Konstanten oder remanente Variablen deklariert werden, die im gesamten Projekt bekannt sind.

Anlegen einer Globalen Variablenliste

Zum Neuanlegen einer globalen Variablenliste markieren Sie im Objekt Organizer bei den **Ressourcen** den Eintrag '**Globale Variablen**' bzw. eine dort bereits angelegte Globale Variablenliste. Wenn Sie dann den Befehl '**Projekt' 'Objekt' 'Einfügen**' wählen, öffnet der Dialog Globale Variablenliste.

Dieser Dialog öffnet übrigens zur Anzeige einer bereits vorgenommenen Konfiguration der im Objekt Organizer markierten Globalen Variablenliste auch beim Befehl '**Projekt' 'Objekt' 'Eigenschaften**'.

Editieren der Listen für Remanente Globale Variablen

Wenn es vom Laufzeitsystem unterstützt wird, kann mit remanenten Variablen gearbeitet werden.

Es gibt zwei Arten von remanenten globalen Variablen :

Persistente und Retain Variablen bleiben bei einem kontrollierten Beenden des Laufzeitsystems (Shutdown) bzw. einem 'Online' 'Reset Kalt' oder einem Download erhalten.

Persistente Variablen sind nicht automatisch auch Retain-Variablen !

Remanente Variablen erhalten zusätzlich das Schlüsselwort RETAIN bzw. PERSISTENT...

Syntax:

```
VAR_GLOBAL RETAIN

(* Variablendeklarationen *)

END_VAR
```

```
VAR_GLOBAL PERSISTENT

(* Variablendeklarationen *)

END_VAR
```

Globale Konstanten

Globale Konstanten erhalten zusätzlich das Schlüsselwort **CONSTANT**.

Syntax:

```
VAR_GLOBAL CONSTANT

(* Variablendeklarationen *)

END_VAR
```

Variablenkonfiguration

In Funktionsbausteinen können bei Variablen, die zwischen den Schlüsselwörtern VAR und END_VAR definiert sind, Adressen für Eingänge und Ausgänge angegeben werden, die nicht vollständig definiert sind. Nicht vollständig definierte Adressen werden mit einem Stern gekennzeichnet.

Beispiel:

```
FUNCTION_BLOCK locio

VAR
```

```
    loci AT %I*: BOOL := TRUE;

    loco AT %Q*: BOOL;

END_VAR
```

Hier werden zwei lokale I/O-Variablen definiert, eine local-In (%I*) und eine local-Out (%Q*).

Wenn Sie lokale I/Os konfigurieren wollen, steht zur Variablenkonfiguration standardmäßig im Object Organizer in der Registerkarte **Ressourcen** das Objekt **Variablen_Konfiguration** zur Verfügung. Das Objekt kann umbenannt werden und es können weitere Objekte für die Variablenkonfiguration angelegt werden.

Der Editor für Variablenkonfiguration arbeitet wie der Deklarationseditor. Variablen zur lokalen I/O-Konfiguration müssen zwischen den Schlüsselwörtern VAR_CONFIG und END_VAR stehen. Der Name einer solchen Variable besteht aus einem vollständigen Instanzpfad, wobei die einzelnen Baustein- und Instanznamen durch Punkte voneinander getrennt sind. Die Deklaration muß eine Adresse enthalten, deren Klasse (Eingang/Ausgang) dem der nicht vollständig spezifizierten Adresse (%I*, %Q*) im Funktionsbaustein entspricht. Auch der Datentyp muß mit der Deklaration im Funktionsbaustein übereinstimmen.

Konfigurationsvariablen, deren Instanzpfad nicht gültig ist , weil die Instanz nicht existiert, werden als Fehler gekennzeichnet. Umgekehrt wird ebenfalls ein Fehler gemeldet, wenn für eine Instanzvariable keine Konfiguration existiert. Um eine vollständige Liste aller benötigten Konfigurationsvariablen zu erhalten, kann der Menübefehl '**Alle Instanzpfade**' im Menü **'Einfügen'** benutzt werden.

Beispiel:

In einem Programm gebe es folgende Definition von Funktionsbausteinen:

```
PROGRAM PLC_PRG

VAR

    Hugo: locio;

    Otto: locio;

END_VAR
```

Dann sieht eine korrekte Variablenkonfiguration folgendermaßen aus:

```
VAR_CONFIG

    PLC_PRG.Hugo.loci AT %IX1.0 : BOOL;

    PLC_PRG.Hugo.loco AT %QX0.0 : BOOL;

    PLC_PRG.Hugo.loci AT %IX1.0 : BOOL;

    PLC_PRG.Otto.loco AT %QX0.3 : BOOL;

END_VAR
```

Hier werden zwei lokale I/O-Variablen definiert, eine local-In (%I*) und eine local-Out (%Q*).

HINWEIS

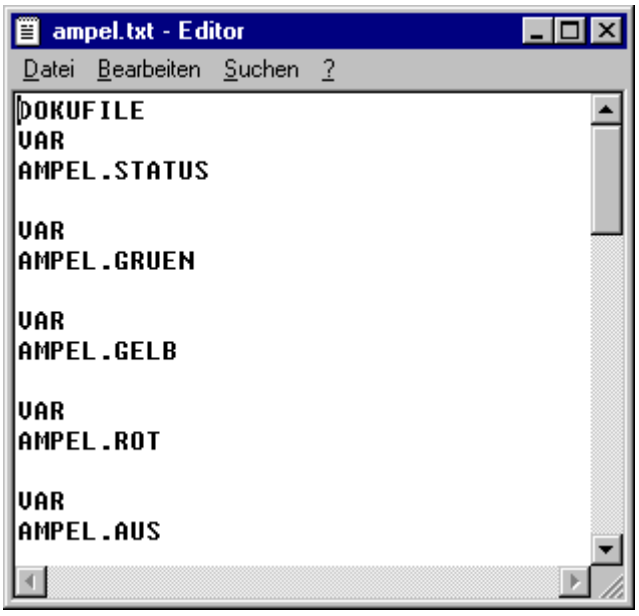
Achten Sie darauf, daß ein Ausgang, der in der Variablenkonfiguration verwendet wird, nicht im Projekt direkt oder über eine Variable (AT-Deklaration) beschrieben wird (AT-Deklaration), da das nicht beachtet wird.

'Einfügen''Alle Instanzpfade'

Mit diesem Befehl wird ein VAR_CONFIG - END_VAR-Block erzeugt, der alle im Projekt vorhandenen Instanzpfade enthält. Bereits vorhandene Deklarationen werden nicht neu eingefügt, um bestehende Adressen zu erhalten. Dieser Menüpunkt steht im Fenster der Variablenkonfiguration zur Verfügung, wenn das Projekt kompiliert ist ('**Projekt''Alles übersetzen**').

Dokumentvorlage

Wenn Sie ein Projekt mehrfach dokumentieren müssen, etwa mit deutschen und mit englischen Kommentaren, oder wenn Sie mehrere ähnliche Projekte dokumentieren wollen, die dieselben Variablennamen benutzen, dann können Sie sich viel Arbeit sparen, indem Sie eine Dokumentvorlage erstellen mit dem Befehl '**Extras' 'Doku-Vorlage erstellen'.** Die erstellte Datei können Sie in einen beliebigen Texteditor laden, und editieren. Die Datei beginnt mit der Zeile **DOKUFILE**, dann folgt eine Auflistung der Projektvariablen, wobei zu jeder Variablen drei Zeilen vorgegeben sind, eine Zeile **VAR**, die anzeigt, wann eine neue Variable kommt, dann eine Zeile mit dem Namen der Variablen, und schließlich eine leere Zeile. Diese Zeile können Sie nun ersetzen durch einen Kommentar zu der Variablen. Variablen, die Sie nicht dokumentieren wollen, löschen Sie einfach aus dem Text. Sie können beliebig viele Dokumentvorlagen zu Ihrem Projekt erstellen.



Windows-Editor mit Dokumentvorlage

Um eine Dokumentvorlage zu benutzen, geben Sie den Befehl **'Extras' 'Doku-Vorlage auswählen**'. Wenn Sie nun das gesamte Projekt dokumentieren, oder Teile Ihres Projekts drucken, dann wird im Programmtext zu allen Variablen der Kommentar eingefügt, den Sie in der Doku-Vorlage erstellt haben. Dieser Kommentar erscheint nur im Ausdruck!

'Extras' 'Doku-Vorlage erstellen'

Mit diesem Befehl erstellen Sie eine Dokumentvorlage. Der Befehl steht zur Verfügung, wenn ein Objekt der Globalen Variablen selektiert ist. Es öffnet sich der Dialog zum Abspeichern von Dateien unter einem neuen Namen. Im Feld für den Dateinamen ist der Zusatz *.txt bereits eingegeben. Wählen Sie einen beliebigen Namen aus. Es wird nun eine Textdatei erstellt, in der sämtliche Variablen Ihres Projekts aufgelistet sind.

'Extras' 'Doku-Vorlage auswählen'

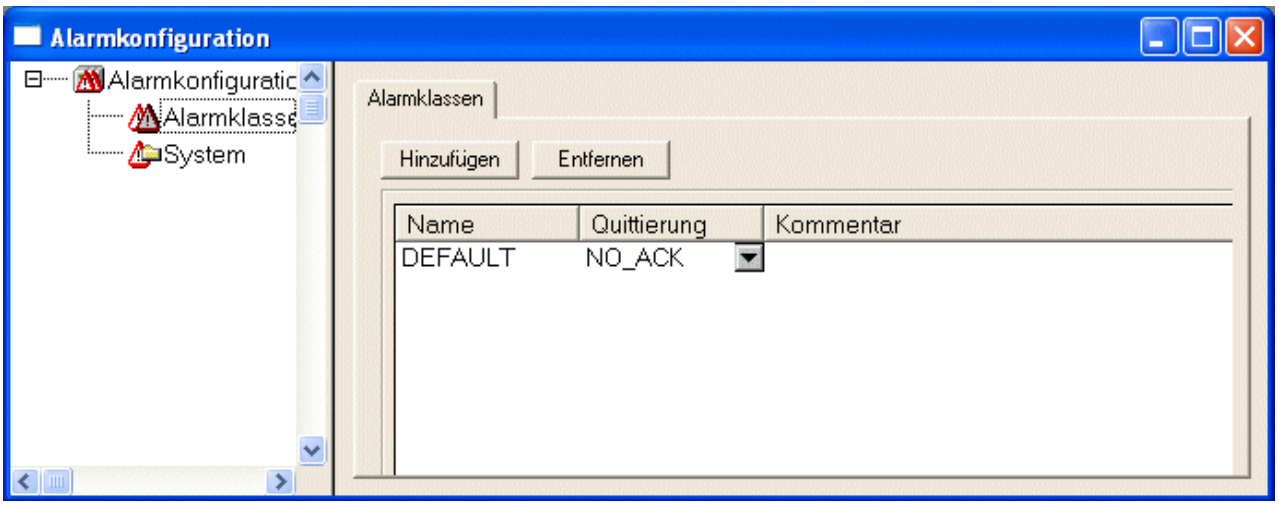
Mit diesem Befehl wählen Sie eine Dokumentvorlage aus. Der Dialog zum Öffnen von Dateien öffnet. Wählen Sie die gewünschte Dokumentvorlage aus und drücken Sie OK. Wenn Sie nun das gesamte Projekt dokumentieren, oder Teile Ihres Projekts drucken, dann wird im Programmtext zu allen Variablen der Kommentar eingefügt, den Sie in der Doku-Vorlage erstellt haben. Dieser Kommentar erscheint nur im Ausdruck! Zum Erstellen einer Dokumentvorlage verwenden Sie den Befehl '**Extras' 'Doku-Vorlage erstellen'.**

6.2 Alarmkonfiguration

Mit Hilfe des in TwinCAT PLC Control integrierten Alarmsystems ist es möglich, kritische Prozesszustände festzustellen, diese aufzuzeichnen oder dem Benutzer über eine Visualisierung zu veranschaulichen. Die Alarmbehandlung kann in TwinCAT PLC Control, optional aber auch auf der Steuerung durchgeführt werden. Sehen Sie hierzu die Zielsystemeinstellungen im Dialog 'Visualisierung'.
Zur Konfiguration steht die 'Alarmkonfiguration' im Object Organizer im Register Ressourcen zur Verfügung. Hier werden Alarmklassen und Alarmgruppen definiert. Die Alarmklasse dient der Typisierung eines Alarms, das heißt sie verleiht ihm bestimmte Parameter. Die Alarmgruppe dient dem konkreten Konfigurieren eines oder mehrerer Alarme (denen jeweils eine bestimmte Klasse und weitere Parameter zugewiesen werden) zur Verwendung im Projekt. Sie bietet also die Möglichkeit der Strukturierung der verfügbaren Alarme. Die verschiedenen Alarmgruppen werden vom Benutzer unterhalb des Gliederungspunkts 'System' eingehängt und definiert.
Zur Visualisierung von Alarmen steht das Element 'Alarmtabelle' in der Visualisierung [▶ 234] bereit. In dieser Tabelle kann der Benutzer Alarme überwachen und bestätigen.
Um eine Historie, d.h. Aufzeichnung von Alarm-Events in einer Log-Datei zu erhalten, muss eine solche angegeben werden und für jede Gruppe das Speicherverhalten definiert werden.

Wenn Sie den Eintrag 'Alarmkonfiguration' in den Ressourcen anwählen, öffnet der Dialog 'Alarmkonfiguration' mit einem zweigeteilten Fenster, dessen Funktionsweise dem der Steuerungskonfiguration oder Taskkonfiguration entspricht. Links wird der Konfigurationsbaum dargestellt, rechts der zum im Baum selektierten Eintrag passende Konfigurationsdialog.

Öffnen Sie durch einen Mausklick auf das Pluszeichen vor dem Eintrag 'Alarmkonfiguration' den aktuell vorliegenden Baum. Bei einer Neukonfiguration enthält er nur die Einträge 'Alarmklassen' und 'System'.



6.2.1 Alarmsystem, Begriffe

Die Verwendung eines Alarmsystems in TwinCAT PLC Control folgt den folgenden allgemeingültigen Beschreibungen und Definitionen zu Alarmen:

- **Alarm:** Generell wird ein Alarm als eine spezielle Bedingung (Wert eines Ausdrucks) angesehen.
- **Priorität:** Die Priorität oder auch „Severity“ eines Alarms beschreibt wie schwerwiegend oder auch wichtig die Alarmbedingung ist. Höchste Priorität ist "0", niedrigste mögliche Angabe ist "255".
- **Alarmzustand:** Ein für die Alarmüberwachung konfigurierter Ausdruck/Variable kann die folgenden Zustände einnehmen: NORM (kein Alarmzustand), INTO (Alarm ist eben eingetreten, "Alarm kommt"), ACK (Alarm ist eingetreten und wurde vom Benutzer bestätigt), OUTOF (Alarmzustand wurde wieder beendet, "Alarm ist gegangen', aber noch nicht bestätigt !)
- **Unterzustand:** Eine Alarmbedingung kann Grenzen (Lo, Hi) und "extreme" Grenzen besitzen (LoLo, HiHi). Beispiel: Der Wert eines Ausdrucks steigt an und überschreitet zunächst die HI-Grenze, woraufhin der HI-Alarm eintritt. Wenn der Wert weiter steigt und noch vor Bestätigen des HI-Alarms die

HIHI-Grenze überschreitet, wird der HI-Alarm intern automatisch bestätigt und es existiert nur noch der HIHI-Alarmzustand in der Alarmliste (interne Liste zur Verwaltung der Alarme). Der HI-Zustand wird in diesem Fall als Unterzustand bezeichnet.

- **Bestätigung von Alarmen:** Eine Hauptanwendung von Alarmen ist es, einem Benutzer eine Alarmsituation mitzuteilen. Dabei ist es oftmals notwendig sicherzustellen, dass dieser die Benachrichtigung auch erhalten hat (siehe mögliche Aktionen in der Alarmklassenkonfiguration). Der Benutzer muss den Alarm "bestätigen" damit dieser aus der Alarmliste gelöscht wird.
- **Alarm-Event:** Ein Alarm-Event darf nicht mit einer Alarmbedingung verwechselt werden. Während eine Alarmbedingung über einen längeren Zeitpunkt gelten kann, beschreibt ein Alarm-Event das momentane Auftreten einer Veränderung, beispielsweise vom Normalzustand zum Alarmzustand. Mögliche Alarm-Events: INTO, OUTOF, ACK,

In TwinCAT PLC Control werden folgende Möglichkeiten unterstützt.

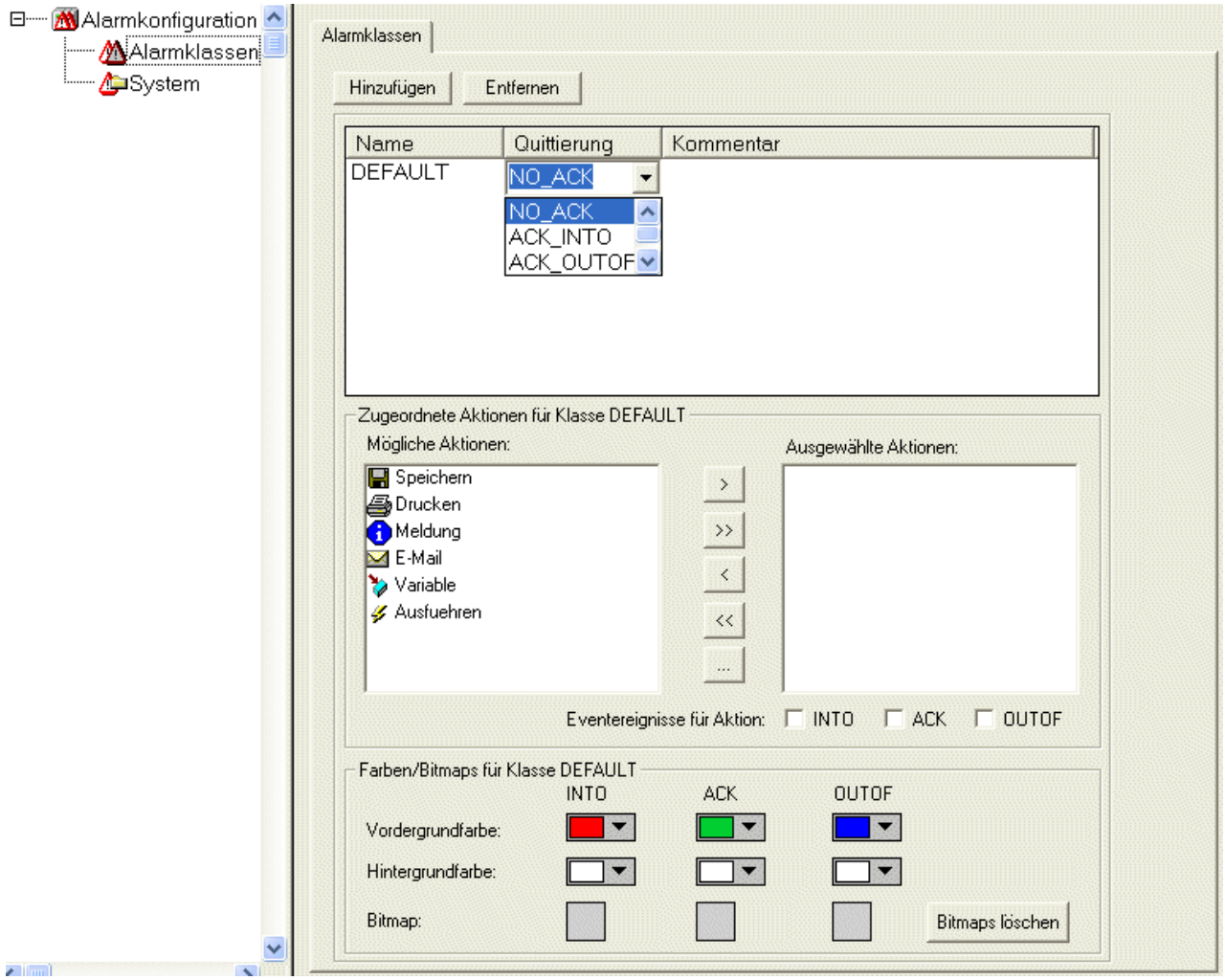
- Deaktivierung der Alarmerzeugung einzelner Alarme, sowie ganzer Alarmgruppen
- Selektion der darzustellenden Alarme über Alarmgruppen sowie der Priorität einzelner Alarme
- Speicherung aller aufgetretenen AlarmEvents
- Visualisierungselement Alarmtabelle in der TwinCAT PLC Control Visualisierung

6.2.2 Alarmklassen

Alarmklassen dienen der allgemeinen Beschreibung bestimmter Alarmkriterien wie z.B. die Quittierungsphilospie (Bestätigung eines Alarms durch den Benutzer), die Aktionsausführung (was soll bei bestimmten Alarmzuständen automatisch passieren) und die Visualisierung in der Alarmtabelle. Alarmklassen werden global in der Alarmkonfiguration definiert und stehen dann jeder Alarmgruppe als "Basiskonfiguration" zur Verfügung

Konfiguration von Alarmklassen:

Markieren Sie den Eintrag 'Alarmklassen' im Alarm-Konfigurationsbaum. Der Konfigurationsdialog 'Alarmklassen' erscheint:



Konfigurationsdialog 'Alarmklassen'

Drücken Sie die Schaltfläche **Hinzufügen**, um eine Alarmklasse anzulegen. Daraufhin wird im oberen Fenster eine Zeile eingefügt, die zunächst nur die Einstellung "NOACK" (no acknowledgement) in der Spalte 'Quittierung' anzeigt. Vergeben Sie einen Namen für die Alarmklasse, indem Sie mit einem Mausklick auf das Feld unterhalb **Name** einen Editier-Rahmen öffnen und wählen Sie bei Bedarf einen anderen Quittierungstyp aus der Auswahlliste unterhalb **Quittierung**.

Es gibt folgende Quittierungstypen:

NO_ACK: Keine Bestätigung des Alarms durch den Benutzer erforderlich

ACK_INTO: Eine "gekommene Alarmbedingung" (Status "INTO", Alarm tritt ein) muss durch den Benutzer quittiert werden.

ACK_OUTOF: Ein "gegangener Alarm" (Status "OUTOF", Alarm beendet) muss durch den Benutzer quittiert werden.

ACK_ALL: Gegangene und gekommene Alarmbedingungen müssen durch den Benutzer quittiert werden.

Zusätzlich können Sie einen **Kommentar** eingeben.

Einträge für weitere Alarmklassen werden jeweils unten an die Liste angehängt.
Mit der Schaltfläche **Entfernen** wird der gerade selektierte Eintrag aus der Liste gelöscht.

Zugeordnete Aktionen für Klasse <Klassenname>:

Jeder Alarmklasse kann eine Liste von Aktionen zugeordnet werden, die beim Eintreten von Alarm-Events ausgelöst werden sollen.
Markieren Sie in der Liste **Mögliche Aktionen** die gewünschte und bringen Sie sie über die Schaltfläche ">" in das Feld **Ausgewählte Aktionen**. Über die Schaltfläche "**>>**" können Sie gleichzeitig alle Aktionen auswählen. Ebenso entfernen Sie über "**<**" bzw. "**<<**" eine oder alle Aktionen wieder aus der Auswahl.
Wenn eine ausgewählte Aktion in der Liste markiert ist, kann über die Schaltfläche "**...**" ein entsprechender Dialog geöffnet werden, in dem die gewünschte E-Mail-Adresse, Druckereinstellung und jeweils ein Meldungstext definiert werden können.

Es werden folgende Aktionstypen unterstützt:

Aktion	Beschreibung	Einstellungen, die im zugehörigen Dialog vorgenommen werden.
Speichern:	Der Alarm-Event wird intern gespeichert, um beispielsweise in eine Log-Datei ausgegeben zu werden. **Bitte beachten**: Dazu muss diese Datei in der Alarmgruppenkonfiguration definiert worden sein!	Diese Einstellungen müssen bei der Konfiguration der Alarmspeicherung vorgenommen werden.
Drucken:	Eine Meldung wird an einen Drucker ausgegeben	**Druckerschnittstelle:** Wählen Sie einen der auf dem lokalen System definierten Drucker aus; **Textausgabe:** Meldungstext (s.u.), der ausgedruckt werden soll
Meldung:	Es wird eine Meldungsbox mit dem zu definierenden Text geöffnet.	**Meldung:** Meldungstext (s.u.), der in einem eigenen Meldungsfenster ausgegeben werden soll.
E-Mail:	Eine E-Mail wird verschickt, die den zu definierenden Meldungstext enthält.	**Von:** E-Mail Adresse des Absenders; **An:** E-Mail Adresse des Empfängers; **Betreff:** Betreff-Text; **Nachricht:** Meldungstext (s.u.); **Server:** Name des E-Mail Servers
Variable:	Einer Variablen des aktuellen Projekts wird der Alarmstatus bzw. ein Meldungstext zugewiesen.	**Variable:** Variablenname: Eine Variable kann über die Eingabehilfe ausgewählt werden (<F2>): Eine boolesche Variable kann verwendet werden, um die Alarmstati NORM=0 und INTO=1 anzuzeigen, eine ganzzahlige Variable zeigt die Alarmstati NORM =0, INTO =1, ACK =2, OUTOF =4 an; eine String-Variable wird der Meldungstext zugewiesen, der im Feld **Message** definiert ist (s.u.)
Ausführen:	Ein externes Programm wird gestartet sobald der Alarm-Event (s.u.) eintritt.	**Ausführbare Datei:** Name der Datei, die ausgeführt werden soll (z.B. notepad.exe); über die Schaltfläche "..." kann der Standarddialog zum Auswählen einer Datei zu Hilfe genommen werden; **Parameter:** der exe-Datei entsprechende Parameter, die dem Aufruf angehängt werden sollen.

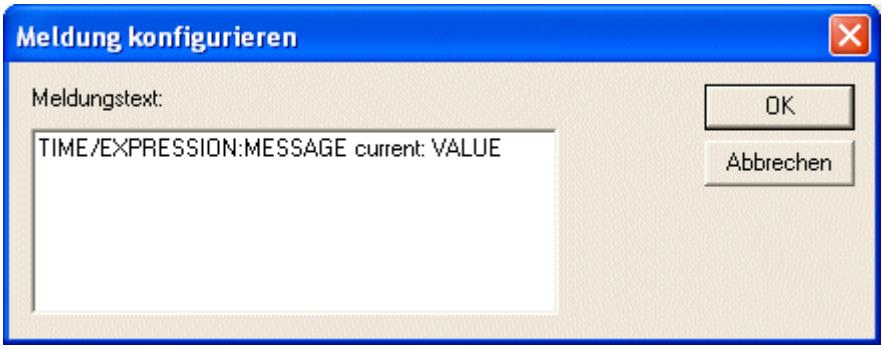
Definition des Meldungstexts:

Bei der Konfiguration der Aktionen 'Meldung','E-Mail', 'Drucken', 'Variable' und ggfs. 'Ausführen' können Sie einen Meldungstext definieren, der bei Eintritt des Alarmevents (s.u.) ausgegeben werden soll.
Zeilenumbrüche sind mit <Strg>+<Eingabe> einzufügen.
Folgende **Platzhalter** können verwendet werden:

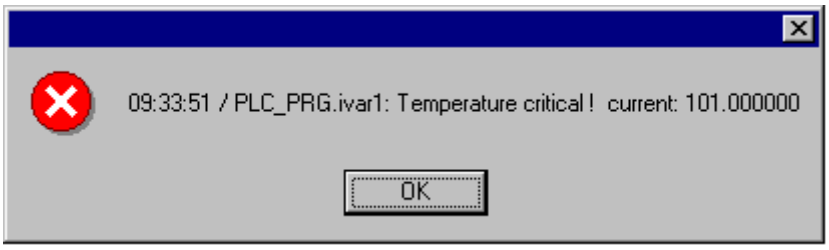
MESSAGE	Der in der Alarmgruppenkonfiguration für den Alarm definierte Meldungstext (Spalte Meldung) wird ausgegeben.
DATE	Das Datum des Wechsels zum zugehörigen Status (INTO)
TIME	Die aktuelle Uhrzeit des Alarmeintritts wird ausgegeben
EXPRESSION	Der Ausdruck (in Alarmgruppe definiert), der den Alarm ausgelöst hat
PRIORITY	Priorität des Alarms (in Alarmgruppe definiert.
VALUE	Aktueller Wert des Ausdrucks
TYPE	Alarmtyp (in Alarmgruppe definiert)
CLASS	Alarmklasse (in Alarmgruppe definiert)
TARGETVALUE	Zielwert bei Alarmtypen DEV+ und DEV- (in Alarmgruppe definiert)
DEADBAND	Toleranz des Alarms (in Alarmgruppe definiert)
ALLDEFAULT	Alle Angaben zum Alarm werden, wie für die Ausgabe in eine Speicherdatei (Historie) beschrieben, ausgegeben

Beispiel:

Tragen Sie zur Definition der Meldungsbox (Aktion 'Meldung') folgendes in das Meldungsfenster ein:



Geben Sie außerdem bei der Definition des Alarms in der Alarmgruppe im Tabellenfeld 'Meldung' folgendes ein: "Temperature critical !". Die Alarmmeldung wird dann folgendermaßen ausgegeben:



Der Meldungstext kann über eine *.vis-Datei oder eine Übersetzungsdatei *.tlt bei einer Sprachumschaltung des Projekts ebenfalls berücksichtigt werden. Um in die Übersetzungsdatei *.tlt aufgenommen zu werden muss die entsprechende Zeichenfolge allerdings – wie alle visualisierungs-bezogenen Texte – zu Beginn und am Ende mit "#"-Zeichen versehen werden (z.B. im oben gezeigten Beispiel: "#Temperature critical !#" und "TIME /EXPRESSION: MESSAGE #current#: VALUE", um die entsprechenden Textteile als ALARMTEXT_ITEM in der Übersetzungsdatei zu erhalten.)

Eine **Speicherdatei** zur Aktion 'Speichern' wird innerhalb der Alarmgruppenkonfiguration definiert.

Eventereignisse für Aktion:

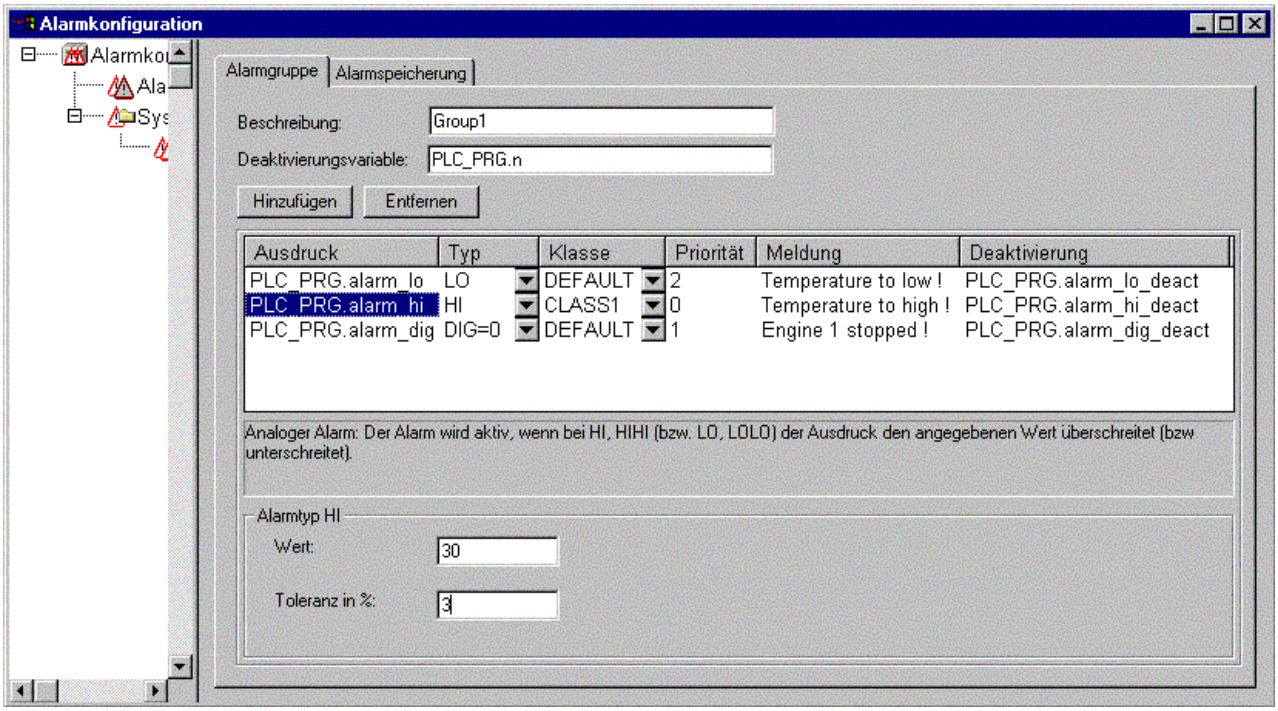
Zu jeder Aktion wird festgelegt bei welchen **Alarm-Events** sie ausgelöst wird.
Aktivieren Sie die gewünschten Events:
INTO Der Alarm tritt ein.
ACK Eine Bestätigung durch den Benutzer erfolgt.
OUTOF Der Alarmzustand wird beendet.

Farben/Bitmaps für Klasse:

Jeder Alarmklasse können verschiedene Farben und Bitmaps zugeordnet werden, welche dann beim Visualisieren der Alarmtabelle für die Unterscheidung der Alarme verwendet werden. Wählen Sie jeweils **Vordergrundfarbe** und **Hintergrundfarbe** für die möglichen Alarm-Events INTO, ACK und OUTOF (siehe oben). Dazu öffnet der Standarddialog zur Farbauswahl wenn Sie auf die Pfeilsymbole klicken, bzw. der Dialog zur Auswahl einer Bitmap-Datei wenn Sie auf das jeweilige graue Quadrat klicken.

6.2.3 Alarmgruppen

Alarmgruppen dienen der Strukturierung verschiedener Alarme. Jeder Alarm ist genau einer Alarmgruppe zugeordnet und wird von dieser verwaltet. Alle Alarme einer Gruppe können eine gemeinsame Deaktivierungsvariable und gemeinsame Parameter hinsichtlich der Alarmspeicherung zugewiesen bekommen. Die Gruppe kann also der Strukturierung der verfügbaren Alarme dienen. Auch ein einzelner Alarm muss in einer Alarmgruppe konfiguriert werden.
Eine hierarchische Gliederung von Alarmgruppen in der Alarmkonfiguration kann mit Hilfe von Ordner-Elementen hergestellt werden. Wenn eine Alarmgruppe im Alarm-Konfigurationsbaum selektiert wird, wird automatisch der **Dialog Alarmgruppe** angezeigt:



Konfigurationsdialog 'Alarmgruppe'

Im Feld **Beschreibung** können Sie eine Bezeichnung für die Alarmgruppe eingeben.
Als **Deaktivierungsvariable** kann eine boolesche Projektvariable eingetragen werden, die bei steigender Flanke die Alarmauslösung für alle Alarme der vorliegenden Gruppe deaktiviert und bei fallender Flanke wieder aktiviert.
Über die Schaltfläche **Hinzufügen** können einzelne Alarme der Gruppe hinzugefügt werden die durch folgende Parameter definiert werden:

Ausdruck: Die Projektvariable, auf die sich der Alarm bezieht. Nehmen Sie für die korrekte Eingabe die Eingabehilfe <F2> bzw. die Intellisense-Funktion zu Hilfe. Es kann auch ein Ausdruck eingetragen werden (z.B. "a + b")

Typ: Folgende Alarmtypen können verwendet werden: (Beachten Sie den jeweils zugehörigen Kommentar, der unterhalb der Tabelle angezeigt wird)

- **DIG=0:** Digitaler Alarm, wird aktiv, wenn der Ausdruck den Wert FALSE annimmt.
- **DIG=1:** Digitaler Alarm, wird aktiv, wenn der Ausdruck den Wert TRUE annimmt.

- **LOLO:** Analoger Alarm, wird aktiv, wenn der Ausdruck den unter 'Alarmtyp LOLO' angegebenen Wert unterschreitet. Eine Toleranzangabe in Prozent des Wertes ist möglich. Innerhalb des Toleranzbereichs wird auch nach Unterschreiten des LOLO-Wertes noch kein Alarm ausgelöst.
- **LO:** entsprechend wie LOLO
- **HI:** Analoger Alarm, wird aktiv, wenn der Ausdruck den unter 'Alarmtyp HIHI' angegebenen Wert überschreitet. Eine Toleranzangabe in Prozent des Wertes ist möglich. Innerhalb des Toleranzbereichs wird auch nach Überschreiten des HI-Wertes noch kein Alarm ausgelöst.
- **HIHI:** entsprechend wie bei HI
- **DEV-:** Deviation (Abweichung vom Zielwert); Alarm wird aktiv, wenn der Ausdruck den unter 'Alarmtyp DEV-' angegebenen Zielwert + prozentuale Abweichung unterschreitet. Prozentuale Abweichung = Zielwert* (Abweichung in %) /100.
- **DEV+:** Deviation (Abweichung vom Zielwert); Alarm wird aktiv, wenn der Ausdruck den unter 'Alarmtyp DEV-' angegebenen Zielwert + die angegebene Abweichung überschreitet. Prozentuale Abweichung = Zielwert* (Abweichung in %) /100.
- **ROC:** Rate of Change (Änderungsrate pro Zeiteinheit); Alarm wird aktiv, wenn der Ausdruck sich zum vorherigen Wert zu stark verändert hat. Der alarmauslösende Grenzwert der Änderungsintensität wird definiert durch die Anzahl der Einheiten (Wertänderung), die sich pro Sekunde, Minute oder Stunde ändern.

Klasse: Wählen Sie die gewünschte Alarmklasse. Sie erhalten die vor der letzen Speicherung des Projekts in der Alarmklassen-Konfiguration definierten Klassen zur Auswahl.

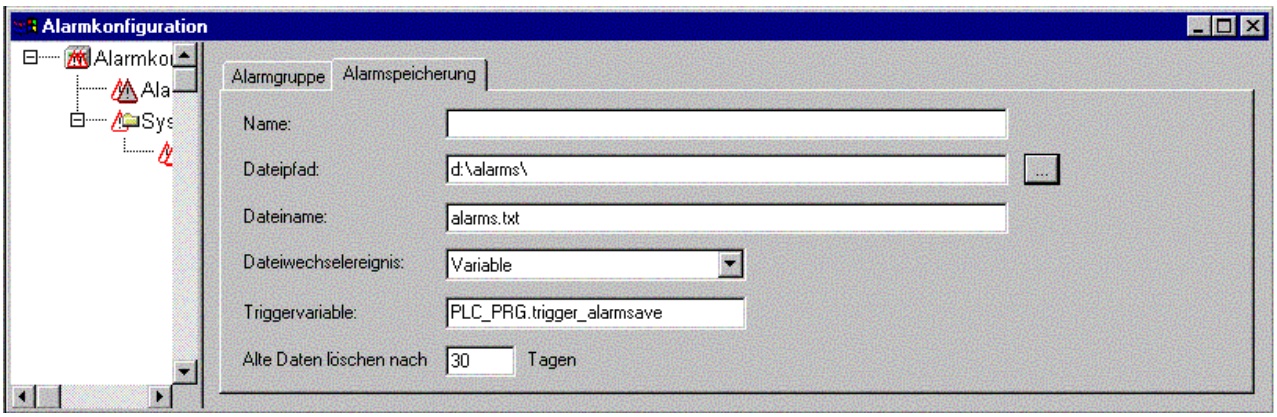
Priorität: Hier können Prioritäten von 0-255 vergeben werden, wobei 0 die höchste Priorität hat. Die Priorität wirkt sich auf die Sortierung in der Alarmtabelle aus.

Meldung: Definieren Sie hier den Text für die Meldung, die innerhalb der Alarmtabelle oder über das Makro "MESSAGE" innerhalb der jeweiligen Aktionen ausgegeben werden kann. Diese Box muss vom Benutzer mit OK bestätigt werden, was jedoch nicht automatisch den Alarm bestätigt ! Zur Alarmbestätigung muss auf die Alarmliste zugegriffen werden, was über das Visualisierungselement 'Alarmtabelle' möglich ist, bzw. über das Datum des Eintrags des Alarms, das aus einer Speicherdatei zu entnehmen ist, welche optional erzeugt werden kann.

Deaktivierung Hier kann eine Projektvariable eingetragen werden, die bei einer steigenden Flanke die Auslösung des Alarms deaktiviert. Beachten Sie jedoch , dass dieser Eintrag durch einen im Feld 'Deaktivierungsvariable' (siehe oben) vorgenommenen Eintrag überschrieben wird.

6.2.4 Alarmspeicherung

Für jede Alarmgruppe kann eine Datei definiert werden, in der die Alarm-Events gespeichert werden, wenn (!) ein "Speichern" in der Aktionenliste der betroffenen Klasse aktiviert worden ist.
Selektieren Sie die Alarmgruppe im Alarm-Konfigurationsbaum und wählen das Dialog-Registerblatt 'Alarmspeicherung':



Konfigurationsdialog 'Alarmspeicherung'

Folgende Eingaben sind möglich:

Dateipfad: Verzeichnispfad für die unter Dateiname angegebene Datei; über die Schaltfläche "..." erhalten Sie den Standarddialog zum Auswählen eines Verzeichnisses.

Dateiname: Name der Datei, die die Alarm-Events speichern soll (z.B. "alarmlog"). Automatisch wird die Datei dann mit dem hier definierten Namen angelegt, dem eine Ziffer angehängt wird sowie die Erweiterung ".alm". Die Ziffer gibt die Version der log-Datei an. Die erste Speicherdatei wird mit einer 0 versehen, weitere, die aufgrund der unter 'Dateiwechselereignis' definierten Bedingungen entstehen, aufsteigend mit 1, 2 usw. (Beispiele -> "alarmlog0.alm", "alarmlog1.alm). Das Format der Speicherdatei kann über den Dialog 'Einstellungen Dokumentationsrahmen' definiert werden.

Dateiwechselereignis: Geben Sie hier die Bedingung an, unter der eine neue Datei zur Speicherung angelegt werden soll. Mögliche Bedingungen: Niemals, nach einer Stunde, nach einem Tag, nach einer Woche, nach einem Monat, nach einer steigenden Flanke der unter 'Triggervariable' angegebenen Variable, nach Erreichen einer bestimmten, unter 'Maximale Eintraganzahl' angegebene Anzahl von Einträgen.

Triggervariable bzw. Maximale Eintraganzahl: siehe Dateiwechselereignis:

Alte Daten löschen nach .. Stunden: Anzahl der Tage nach Erstellungsdatum, nach denen alle Alarmspeicherdateien außer der aktuellen gelöscht werden.

Die Speicherdatei (Historie) enthält die folgenden Einträge:

Datum/Zeit in DWORD	Datum	Zeit	Event	Ausdruck	Alarmtyp	Grenzwert	Toleranz	akt. Wert	Klasse	Priorität	Meldung
1046963332	6.3.03	16:08:52	INTO	PLC_PRG.b	LO -30	5 -31	Alarm_high	0	cl1	3	Meldung1
1046963333	6.3.03	16:08:53	ACK	PLC_PRG.n	HIHI	35	Warnng	9	cl3	0	Meldung2

BEISPIEL:

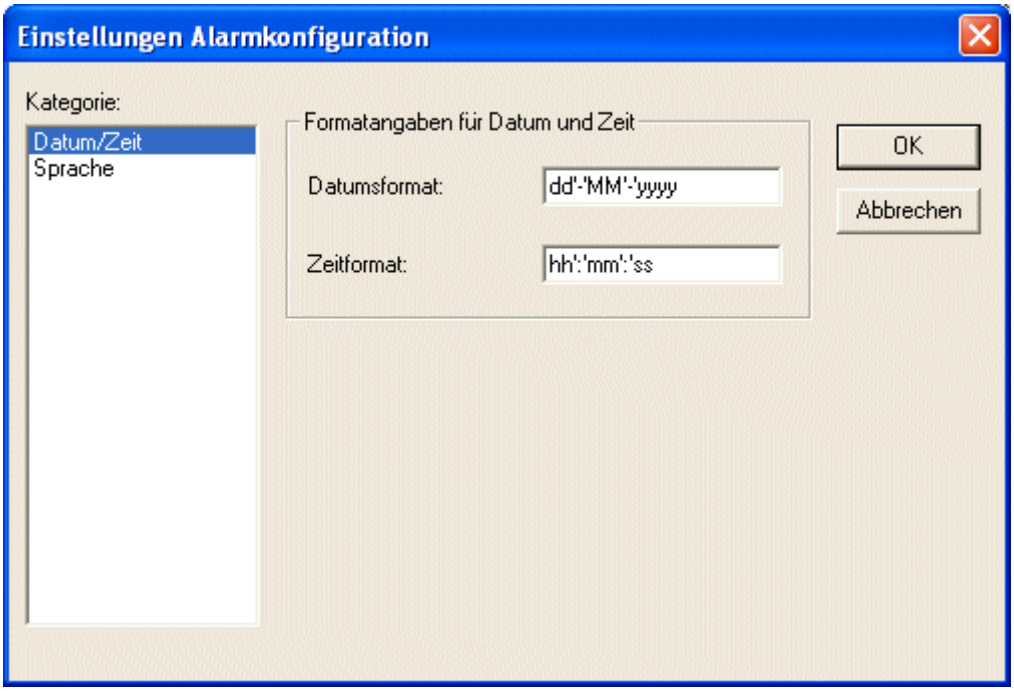
```
1046963332,6.3.03 16:08:52,INTO,PLC_PRG.ivar5,HIHI,,,,    9.00,a_class2,0,
1046963333,6.3.03 16:08:53,INTO,PLC_PRG.ivar4,ROC,2,,,    6.00,a_class2,2,
1046963333,6.3.03 16:08:53,INTO,PLC_PRG.ivar3,DEV-,,,,   -6.00,a_class2,5,
1046963334,6.3.03 16:08:54,INTO,PLC_PRG.ivar2,LOLO,-35,,3,  -47.00, warning, 10, warning: low temperature !
1046963334,6.3.03 16:08:54,INTO,PLC_PRG.ivar1,HI,20,,5,   47.00,a_classl,2,temperature to high ! Acknowledge !
```

6.2.5 Menü Extras: Einstellungen

Kategorie Datum/Zeit:
Hier definieren Sie, in welchem Format Datums- und Zeitangaben in der Speicherdatei für die Alarme dargestellt werden sollen. Geben Sie die Formate gemäß folgender Syntax an, Bindestriche und Doppelpunkte werden in einfache Hochkommas gesetzt:

für Datum: dd'-'MM'-'yyyy -> z.B. "12.Jan-1993"

für Uhrzeit: hh':'mm':'ss -> z.B. "11:10:34"

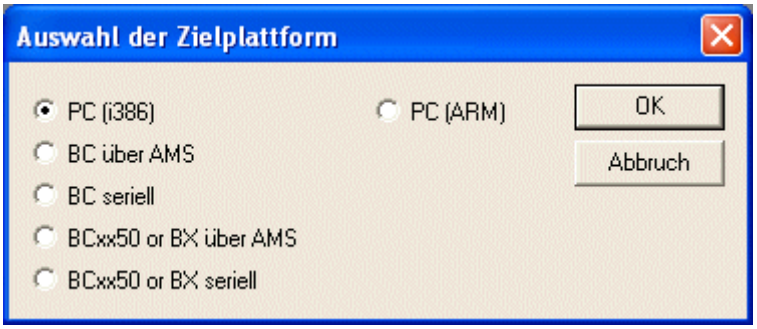


Sprache:

Wählen Sie hier die Sprachdatei, die für die Sprachumschaltung verwendet werden soll, also auch die Texte der Alarmkonfiguration enthalten muss. Sehen Sie hierzu folgende Beschreibungen:

- Visualisierung, Einstellung der Sprache [▶ 251]
- 'Projekt' 'In andere Sprache übersetzen' [▶ 67]

6.3 Steuerungskonfiguration



Die Steuerungskonfiguration ist abhängig von der jeweils zu konfigurierenden Hardware. Mit dem TwinCAT PLC Control können die folgenden Hardwareplattformen programmiert werden.

- PC (i386): Steuerungsprogramm läuft auf dem PC (z.B. CX10x0)
- PC (ARM): Steuerungsprogramm läuft auf einem CX9000.
- BCxxxx: Steuerungsprogramm läuft auf einem Buscontroller (Klemmen-SPS)
- BCxx50 oder BXxxxxx

Soll das Steuerungsprogramm auf der BCxxxx/BCxx50 oder BXxxxx Klemmen-SPS laufen, so gibt es zwei unterschiedliche Möglichkeiten das Programm zu laden:

- Programm wird über AMS geladen (abhängig vom Feldbus)
- Programm wird über eine serielle Schnittstelle geladen

6.4 Taskkonfiguration

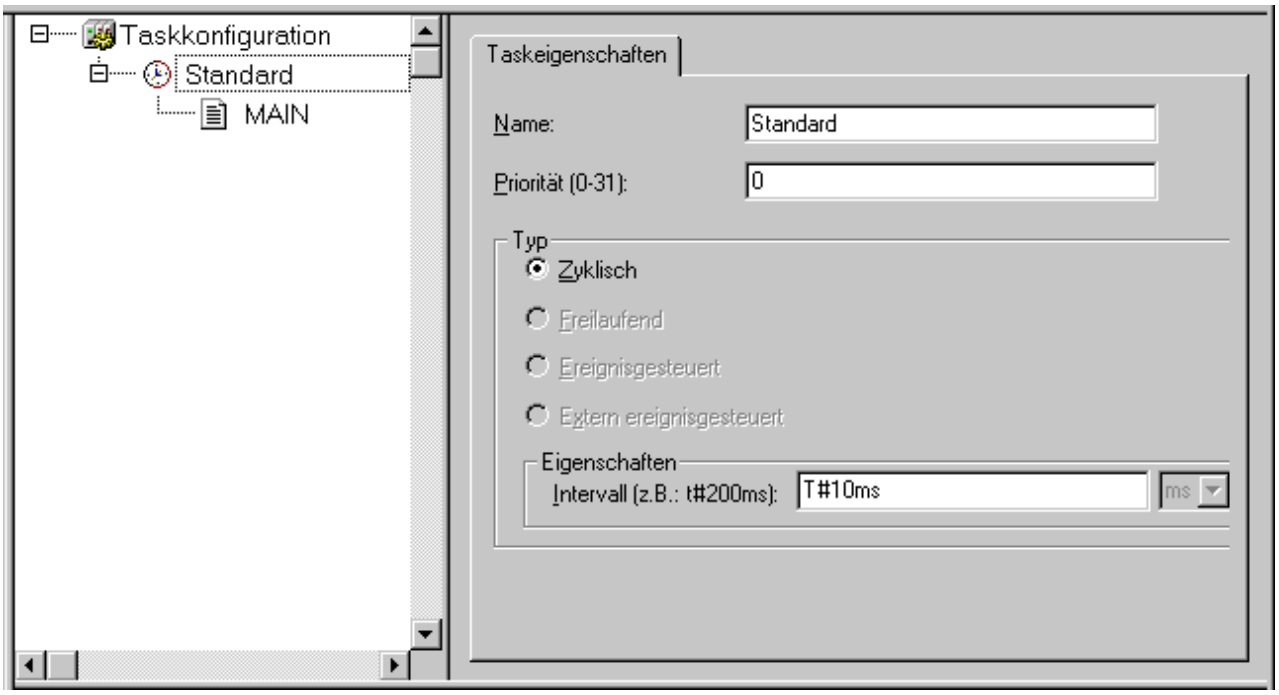
Eine Task ist eine zeitliche Ablaufeinheit eines IEC-Programms. Sie ist definiert durch einen Namen, eine Priorität und einen Typ, der festlegt, welche Bedingung Ihren Start auslöst. Diese Bedingung kann nur zeitlich definiert sein (Zyklusintervall).

Jeder Task kann eine Folge von Programmen zugeordnet werden, die beim Ausführen der Task abgearbeitet werden sollen.

Durch Zusammenwirken von Priorität und Bedingung wird festgelegt, in welcher zeitlichen Abfolge die Tasks abgearbeitet werden.

Im Online Modus kann die Task-Abarbeitung in einer grafischen Darstellung verfolgt werden.

Die Taskkonfiguration befindet sich als Objekt in der Registerkarte Ressourcen im Object Organizer. Der Task-Editor erscheint in einem zweigeteilten Fenster.



Im linken Fensterteil werden die Tasks in einem Konfigurationsbaum dargestellt. In der ersten Zeile steht 'Taskkonfiguration', darunter folgen die Einträge für die einzelnen Tasks. Unterhalb jedes Taskeintrags hängen die zugehörigen Programmaufrufe.

Im rechten Fensterteil wird zu dem im Konfigurationsbaum markierten Eintrag der Eigenschaftendialog geöffnet. Hier können die einzelnen Tasks und Programmaufrufe definiert werden, jeweils der Name, die Priorität und der Typ.

HINWEIS

Datenverlust

Sie sollten nicht in mehreren Tasks gleiche String-Funktionen verwenden, da in diesem Fall bei der Abarbeitung der Tasks die Gefahr des Überschreibens besteht.

Arbeiten im Taskkonfigurator

Die wichtigsten Befehle finden Sie im **Kontextmenü** (rechte Maustaste).

Am Kopf der Taskkonfiguration steht das Wort ”Taskkonfiguration”, wenn sich vor dem Wort ein Pluszeichen befindet, dann ist die nachfolgende Liste zugeklappt. Mit Doppelklick auf die Liste oder Drücken der <Eingabetaste> klappen Sie diese auf. Es erscheint ein Minuszeichen und mit erneutem Doppelklick, klappt die Liste wieder zu. An jede Task ist eine Liste von Programmaufrufen angehängt; diese Liste können Sie ebenfalls auf- und zuklappen.

• Mit dem Befehl ‘**Einfügen’ ‘Task einfügen**’ wird eine Task eingefügt.

• Mit dem Befehl '**Einfügen' 'Task anhängen**' wird eine Task am Ende des Konfigurationsbaums eingefügt.

• Mit dem Befehl ‘**Einfügen’ ‘Programmaufruf einfügen**’ wird ein Programmaufruf zu einer Task eingefügt.

Die Konfiguration eines im Konfigurationsbaum selektierten Eintrags erfolgt im Eigenschaften-Dialog im rechten Fensterteil durch Aktivieren/Deaktivieren von Optionen bzw. Einträge in Eingabefelder. Dies ist entweder der Dialog zum Festlegen der Taskeigenschaften oder der Dialog zum Eintragen des Programmaufrufs. Die vorgenommenen Einstellungen werden sofort in den Konfigurationsbaum übernommen und dort angezeigt, sobald der Fokus wieder dorthin gesetzt wird.

Ein Task- oder Programmname kann direkt im Konfigurationsbaum editiert werden. Dazu wird mit einem Mausklick auf den Namen oder durch Drücken der <Leertaste>, wenn ein Eintrag markiert ist, ein Editierrahmen geöffnet, in dem die Bezeichnung geändert werden kann.

Mit den Pfeiltasten kann im Konfigurationsbaum der nächste bzw. vorangehende Eintrag selektiert werden.

‘Einfügen’ ‘Task einfügen’ oder ‘Einfügen’ ‘Task anhängen’

Mit diesem Befehl fügen Sie der Taskkonfiguration eine neue Task hinzu.

Ist ein Taskeintrag selektiert, steht der Befehl ‘**Task einfügen’** zur Verfügung. Die neue Task wird nach der selektierten eingefügt. Ist das Wort Taskkonfiguration selektiert, steht der Befehl ‘**Task anhängen**’ zur Verfügung und die neue Task wird ans Ende der bestehenden Liste angehängt.

Es öffnet sich der Dialog zur Festlegung von Taskeigenschaften.

Geben Sie die gewünschten Attribute ein:

- **Name**: ein Name für die Task, mit der sie im Konfigurationsbaum erscheint; der Name kann auch dort editiert werden, indem durch Anklicken oder Drücken der Leertaste ein Editierfeld geöffnet wird
- **Priorität (0-3)**: (eine Zahl zwischen 0 und 3, wobei gilt: 0 die höchste, 3 die niedrigste Priorität darstellt),
- **Typ**:
  - **Zyklisch**: Die Task wird entsprechend der bei Intervall eingegebenen Zeit zyklisch gestartet.

Freilaufend, Ereignisgesteuert oder Extern ereignisgesteuert: Diese Tasktypen werden nicht unterstützt!.

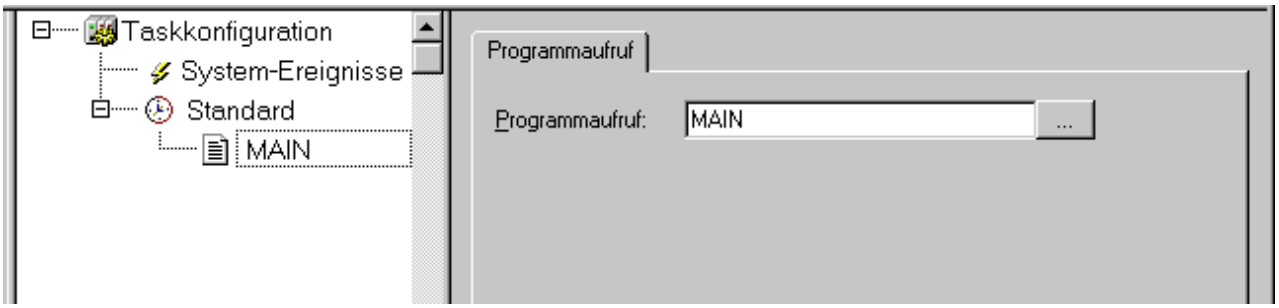
Eigenschaften:

- **Intervall** (für Typ 'Zyklisch'): die Zeitspanne, nach der die Task erneut gestartet werden soll. Wird eine Zahl eingegeben, kann im Auswahlfeld dahinter die Einheit Millisekunden [ms] oder Mikrosekunden [µs] gewählt werden. Eingaben in Millisekunden erscheinen dann nach dem nächsten Fokuswechsel im TIME-Format (z.B. t#200ms); sie können auch direkt so in die Eingabezeile geschrieben werden. Bei Angaben in Mikrosekunden wird weiterhin nur die Zahl dargestellt (z.B. 300).

‘Einfügen’ ‘Programmaufruf anhängen’ oder ‘Einfügen’ ‘Programmaufruf einfügen’

Mit diesen Befehlen öffnen Sie den Dialog zum Eintrag eines Programmaufrufs zu einer Task in der Taskkonfiguration.

Bei ‘Programmaufruf einfügen’ wird der neue Programmaufruf vor dem Cursor eingefügt und bei ‘Programmaufruf anhängen’ ans Ende der bestehenden Liste angehängt.



Geben Sie in das Feld Programmaufruf einen gültigen Programmnamen aus Ihrem Projekt an, oder öffnen Sie mit der Schaltfläche ... oder mit <F2> die Eingabehilfe zur Auswahl gültiger Programmnamen. Der Programmname kann auch im Konfigurationsbaum noch verändert werden, wenn der Programmeintrag selektiert ist. Dazu wird entweder durch einen Mausklick auf den Namen oder durch Drücken der Leertaste ein Editierfeld geöffnet. Wenn das ausgewählte Programm Eingabevariablen erfordert, dann geben Sie diese in der üblichen Form, und vom deklarierten Typ (z.B. prg(invar:=17)) an.

‘Extras’ ‘Debug Task festlegen’

Mit diesem Befehl kann im Online Modus in der Taskkonfiguration eine Task festgelegt werden, in der das Debuggen stattfinden soll. Im Konfigurationsbaum erscheint dann hinter dem Taskeintrag der Text "[DEBUG]".

Die Debugging-Funktionalitäten beziehen sich dann nur auf diesen Task. d.h. das Programm stoppt bei einem Breakpoint nur, wenn das Programm von der eingestellten Task durchlaufen wird.

Die Festlegung der Debug Task wird im Projekt gespeichert und bei Einloggen/Download automatisch wieder gesetzt.

Debug beenden

Um den "Debug-Modus" zu beenden

- wählen Sie "Taskkonfiguration"
- öffnen Sie das Kontextmenü
- wählen Sie „Debug Task festlegen“

'Extras' 'Aufrufhierarchie anzeigen'

Wenn beim Debuggen an einem Breakpoint gestoppt wird, kann über diesen Befehl die Aufrufhierarchie des betreffenden Bausteins ermittelt werden. Dazu muss die Debug-Task im Konfigurationsbaum selektiert sein. Es öffnet sich das Fenster 'Aufrufhierarchie von Task <Taskname>' mit der Anzeige des Bausteins, in dem der Breakpoint liegt (z.B. "prog_x (2)" für Zeile 2 von Baustein prog_x) . Danach folgen in rücklaufender Reihenfolge die Einträge für die aufrufenden Bausteinpositionen. Wird die Schaltfläche Gehe zu betätigt, springt der Fokus zur markierten Position.

6.5 Traceaufzeichnung

Traceaufzeichnung bedeutet, dass der Werteverlauf von Variablen über einen bestimmten Zeitraum hin aufgezeichnet wird. Diese Werte werden in einen Ringspeicher geschrieben (Tracebuffer). Ist der Speicher voll, so werden die "ältesten" Werte vom Speicheranfang her wieder überschrieben.

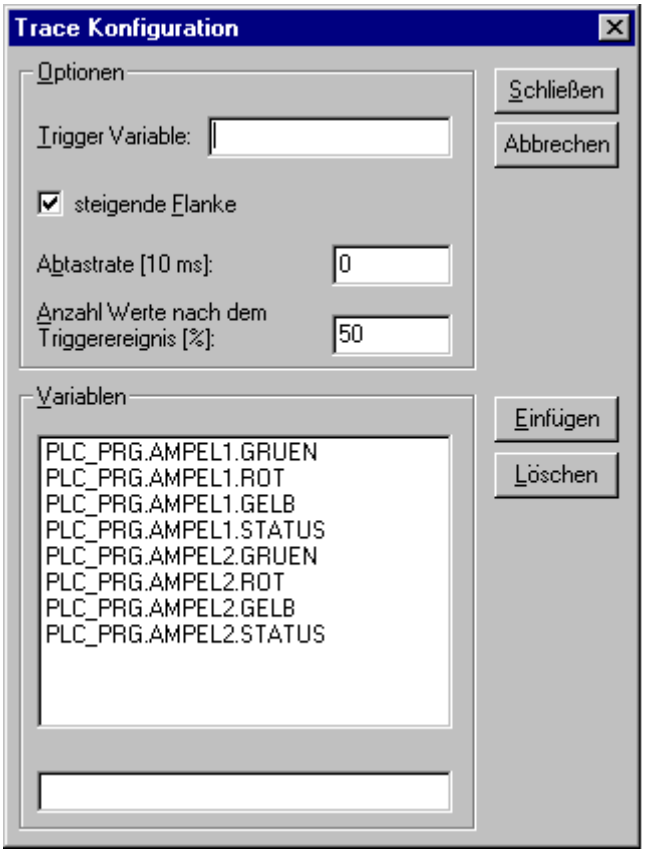
Maximal können 20 Variablen gleichzeitig aufgezeichnet werden. Da die Größe des Tracebuffers in der Steuerung einen fixen Wert besitzt, können bei sehr vielen oder sehr breiten Variablen (DWORD) weniger Werte aufgezeichnet werden.

Um einen Trace aufzeichnen zu können, öffnen Sie das Objekt Traceaufzeichnung in der Registerkarte Ressourcen im Object Organizer. Danach müssen die Tracevariablen, die aufgezeichnet werden sollen, eingegeben werden (siehe **'Extras' 'Tracekonfiguration'**).

Nachdem Sie die Konfiguration mit '**Trace definieren**' an die Steuerung geschickt haben und die Aufzeichnung in der Steuerung gestartet ('**Trace starten**') haben, werden die Werte der Variablen aufgezeichnet. Mit '**Trace lesen**' werden die zuletzt aufgezeichneten Werte ausgelesen und grafisch als Kurven dargestellt.

'Extras' 'Tracekonfiguration'

Mit diesem Befehl erhalten Sie den Dialog zur Eingabe der aufzuzeichnenden Variablen sowie diverser Traceparameter für die Traceaufzeichnung.



Dialog zur Tracekonfiguration

Die Liste der aufzuzeichnenden **Variablen** ist zunächst leer. Um eine Variable anzufügen, muss diese in das Feld unter der Liste eingegeben werden. Anschließend kann sie mit der Schaltfläche **Einfügen** oder der <Eingabetaste> an die Liste angefügt werden. Sie können auch die **Eingabehilfe** verwenden.

Eine Variable wird aus der Liste gelöscht, indem sie selektiert und anschließend die Schaltfläche **Löschen** gedrückt wird.

In das Feld **Trigger Variable** kann eine boolsche oder analoge Variable eingetragen werden. Sie können hier auch die Eingabehilfe verwenden. Die Trigger Variable beschreibt die Abbruchbedingung des Traces. Im **Trigger Level** geben Sie an, bei welchem Wert einer analogen Trigger Variable das Triggerereignis eintritt. Wenn in der **Trigger Flankepositiv** gewählt wurde, tritt das Triggerereignis nach einer steigenden Flanke einer boolschen Trigger Variablen ein bzw. wenn eine analoge Trigger Variable den Trigger Level von unten nach oben durchschreitet. Bei **negativ** entsprechend nach einer fallenden Flanke bzw. Durchschreiten von oben nach unten getriggert. Bei **beide** wird nach fallender und steigender Flanke bzw. positivem und negativem Durchlauf getriggert, bei **keine** gibt es kein Triggerereignis.

In der **Trigger Position** geben Sie an, welcher Prozentsatz der Messwerte vor Eintreten des Triggerereignisses aufgezeichnet wird. Geben Sie hier beispielsweise 25 ein, werden 25% der Messwerte vor und 75% der Messwerte nach dem Triggerereignis dargestellt, dann wird der Trace abgebrochen.

Mit dem Feld **Abtastrate**, können Sie den Zeitabstand zwischen zwei Aufzeichnungen in Millisekunden angeben. Die Vorbelegung "0" bedeutet: ein Abtastvorgang pro Zyklus.

Wählen Sie den Modus des Abrufens der aufgezeichneten Werte: Bei **Einzel** wird einmal die **Anzahl der vorgegebenen Messungen** dargestellt. Bei **Fortlaufend** wird das Auslesen der Aufzeichnung der vorgegebenen Messwert-Anzahl immer wieder neu gestartet. Geben Sie beispielsweise für Anzahl '35' ein, umfasst die erste Darstellung die ersten Messwerte 1 bis 35, dann wird automatisch die Aufzeichnung der nächsten 35 Messwerte (36-70) abgerufen, usw. Bei **Manuell** erfolgt ein Auslesen der Traceaufzeichnung gezielt mit ‚**Extras"Trace lesen**'.

Der Abrufmodus funktioniert unabhängig davon, ob eine Trigger Variable gesetzt ist. Ist keine Trigger Variable angegeben, wird der Tracebuffer mit der Anzahl der vorgegebenen Messwerte gefüllt und beim Abruf wird der Pufferinhalt gelesen und dargestellt.

Über die Schaltfläche **Speichern** wird die erstellte Tracekonfiguration in einer Datei gespeichert. Sie erhalten hierzu das Standardfenster ‚Datei speichern unter'.

Über die Schaltfläche **Laden** können Sie eine abgespeicherte Tracekonfiguration wieder laden. Sie erhalten hierzu das Standardfenster ‚Datei öffnen'.

Beachten Sie, dass **Speichern** und **Laden** aus dem Konfigurationsdialog nur die Konfiguration, nicht die Werte einer Traceaufzeichnung betrifft (im Gegensatz zu den Menübefehlen 'Extras' Trace speichern' und 'Extras' 'Trace laden').
Ist das Feld **Trigger Variable** leer, läuft die Traceaufzeichnung endlos und kann explizit mit **'Extras' 'Trace stoppen'** abgebrochen werden.

'Extra' 'Trace starten'

Mit diesem Befehl wird die Tracekonfiguration in die Steuerung übertragen und die Traceaufzeichnung in der Steuerung gestartet.

'Extra' 'Trace lesen'

Mit diesem Befehl wird der aktuelle Tracebuffer aus der Steuerung gelesen und die Werte der ausgewählten Variablen werden dargestellt.

'Extra' 'Trace automatisch lesen'

Mit diesem Befehl wird der aktuelle Tracebuffer aus der Steuerung automatisch gelesen und die Werte werden fortlaufend dargestellt. Wird der Tracebuffer automatisch gelesen, befindet sich ein Haken vor dem Menüpunkt.

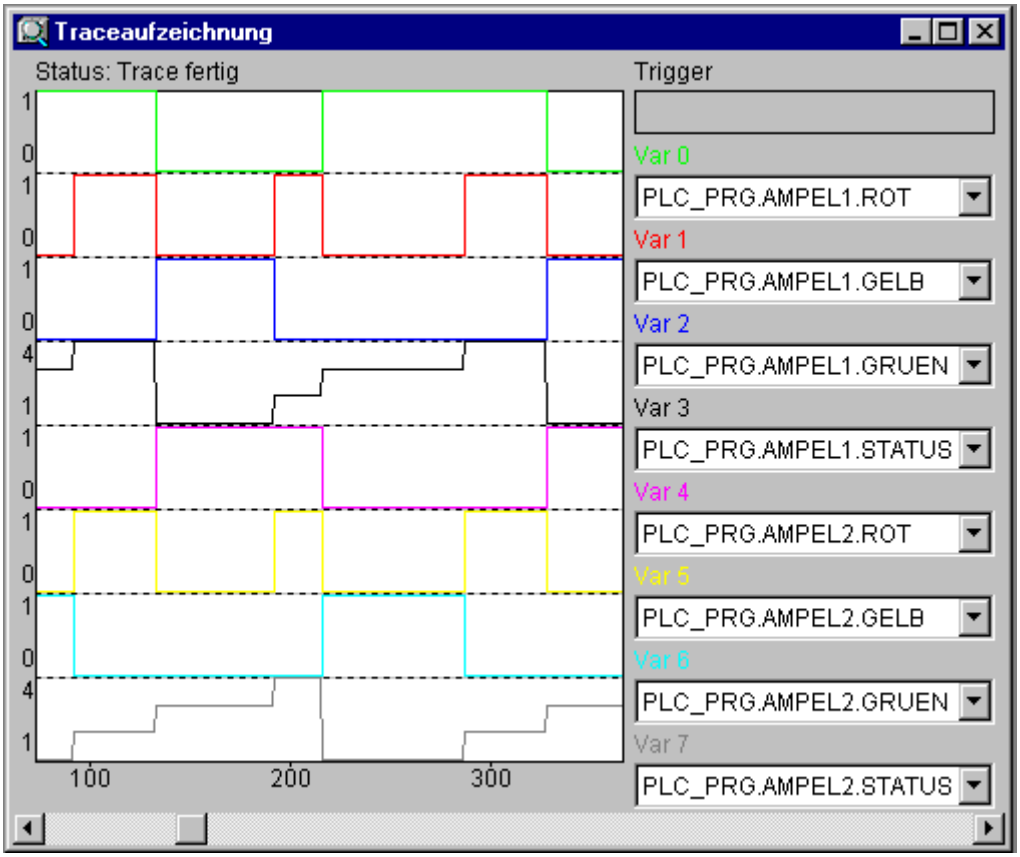
'Extra' 'Trace stoppen'

Dieser Befehl stoppt die Traceaufzeichnung in der Steuerung. Vor einer erneuten Aufzeichnung muss die Tracedefinition geladen und erneut der Trace gestartet werden.

Auswahl der darzustellenden Variablen

Die Comboboxen rechts neben dem Fenster für die Darstellung der Kurven enthalten jeweils alle in der Tracekonfiguration definierten Tracevariablen. Wird eine Variable aus der Liste ausgewählt, so wird diese, nachdem ein Tracebuffer gelesen wurde, in der entsprechenden Farbe ausgegeben (Var 0 grün etc.). Variablen können auch dann ausgewählt werden, wenn bereits Kurven ausgegeben sind.

Es können maximal bis zu acht Variablen gleichzeitig im Tracefenster beobachtet werden.



Darstellung der Traceaufzeichnung von acht verschiedenen Variablen ohne Trigger

Ist ein Tracebuffer geladen, so werden die Werte aller darzustellenden Variablen ausgelesen und dargestellt. Wenn keine Abtastrate eingestellt ist, wird die X-Achse mit der fortlaufenden Nummer des aufgezeichneten Wertes beschriftet.

In der Statusanzeige des Tracefensters (1. Zeile) wird angezeigt, ob der Tracebuffer noch nicht voll ist und wann der Trace fertig ist. Wurde ein Wert für die Abtastrate angegeben, dann gibt die x-Achse die Zeit des Wertes an. Dem "ältesten" aufgezeichneten Wert wird die Zeit 0 zugeordnet. Im Beispiel werden z.B. die Werte der letzten 25s angezeigt.

Die Y-Achse wird mit Integerwerten beschriftet. Die Skalierung ist so aus-gelegt, das der niedrigste und der höchste Wert in den Bildbereich passen. Im Beispiel hat Var5 den niedrigsten Wert 6, als höchsten Wert 11 angenommen, daher auch die Einstellung der Skala am linken Rand.

Ist die Triggerbedingung erfüllt, so wird an der Schnittstelle zwischen den Werten vor Eintreten der Triggerbedingung und danach eine senkrechte gestrichelte Linie ausgegeben.

Ein gelesener Speicher bleibt bis zum Projektwechsel oder Verlassen des Systems erhalten.

'Extras' 'Cursor ausgeben'

Der schnellste Weg, einen Cursor im Graphikfenster zu setzen, ist, mit der linken Maustaste innerhalb des Fensters zu klicken. Der Cursor kann mit der Maus beliebig verschoben werden.

Eine weitere Möglichkeit ist der Befehl 'Extras' 'Cursor ausgeben. Mit diesem Befehl erscheint in der Traceaufzeichnung eine vertikale Linie. Sie können diese Linie mit der Maus oder den Pfeiltasten nach rechts und links verschieben. Durch Drücken von <Strg>+<links>, bzw. von <Strg>+<rechts> erhöhen Sie die Geschwindigkeit der Bewegung. Über dem Graphikfenster, können Sie die aktuelle x-Position des Cursors lesen. Neben Var0,.Var1, ... ,VarN wird der Wert der jeweiligen Variable dargestellt. Wenn der Mauszeiger sich im Graphikfenster befindet, und die linke Maustaste gedrückt wird, dann wird ebenfalls ein Cursor angezeigt.

Durch zusätzliches Drücken der <Umschalt>-Taste verschieben Sie die andere Linie, die den Differenzbetrag zur ersten Linie anzeigt.

'Extras' 'Mehrkanal'

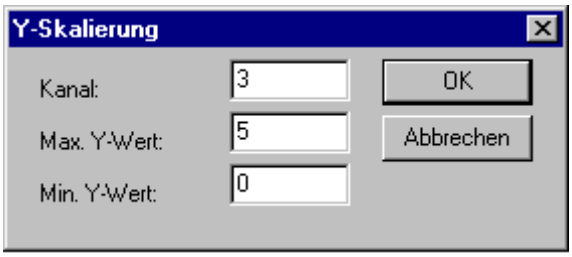
Mit diesem Befehl kann zwischen einkanaliger und mehrkanaliger Darstellung der Traceaufzeichnung gewechselt werden. Bei mehrkanaliger Darstellung befindet sich ein Haken vor dem Menüpunkt. Voreingestellt ist die mehrkanalige Darstellung. Hier wird das Darstellungsfenster auf die bis zu acht darzustellenden Kurven aufgeteilt. Zu jeder Kurve wird am Rand der maximale und der minimale Wert ausgegeben. Bei einkanaliger Darstellung werden alle Kurven mit dem gleichen Skalierungsfaktor dargestellt und überlagert. Dies kann von Nutzen sein, um Abweichungen von Kurven darstellen zu können.

'Extras' 'Koordinatennetz"

Mit diesem Befehl können Sie das Koordinatennetz im Darstellungsfenster ein- und ausschalten. Ist es eingeschaltet, erscheint ein Haken (*) vor dem Menübefehl.

'Extras' 'Y-Skalierung'

Mit diesem Befehl können Sie die vorgegebene Y-Skalierung einer Kurve (**Kanal**) in der Tracedarstellung ändern. Geben Sie im Dialog die Nummer der gewünschten Kurve (Kanal) und den neuen höchsten (**Max. Y-Wert)** und den neuen niedrigsten Wert (**Min. Y-Wert)** auf der y-Achse an. Mit Doppelklick auf eine Kurve erhalten Sie ebenfalls den Dialog. Der Kanal und die bisherigen Werte sind vorbelegt.



Dialog zur Einstellung der Y-Skalierung

'Extras' 'Strecken'

Mit diesem Befehl können die ausgegebenen Werte der Traceaufzeichnung gestreckt (gezoomt) werden. Die Anfangsposition wird mit der horizontalen Bildlaufleiste eingestellt. Bei mehrmaligem aufeinanderfolgendem Strecken, wird ein immer kürzerer Traceausschnitt im Fenster angezeigt. Dieser Befehl ist das Gegenstück zu '**Extras"Komprimieren**'.

'Extras' 'Komprimieren'

Mit diesem Befehl können die ausgegebenen Werte der Traceaufzeichnung komprimiert werden, d.h. nach diesem Befehl kann der Verlauf der Tracevariablen innerhalb einer größeren Zeitspanne betrachtet werden. Eine mehrmalige Ausführung des Befehls ist möglich. Dieser Befehl ist das Gegenstück zu '**Extras"Strecken**'.

'Extras' 'Trace speichern'

Mit diesem Befehl kann eine Traceaufzeichnung (Werte + Konfiguration) abgespeichert werden. Es öffnet sich der Dialog zum Speichern einer Datei. Der Dateiname erhält den Zusatz "*.trc". Die gespeicherte Traceaufzeichnung kann mit '**Extras' 'Trace laden**' wieder geladen werden.

'Extras' 'Trace laden'

Mit diesem Befehl kann eine abgespeicherte Traceaufzeichnung (Werte + Konfiguration) wieder geladen werden. Es öffnet sich der Dialog zum Öffnen einer Datei. Wählen Sie die gewünschte Datei mit dem Zusatz "*.trc". Mit **'Extras' 'Trace speichern**' kann eine Traceaufzeichnung abgespeichert werden.

'Extras' 'Trace in ASCII-File'

Mit diesem Befehl kann eine Traceaufzeichnung in eine ASCII-Datei abgespeichert werden. Es öffnet sich ein der Dialog zum Speichern einer Datei. Der Dateiname erhält den Zusatz "*.txt". In der Datei werden die Werte nach folgendem Schema abgelegt:

TwinCAT PLC Control Trace

D:\TWINCAT PLC CONTROL\PROJECTS\AMPEL.PRO

Zyklus PLC_PRG.ZAEHLER PLC_PRG.LIGHT1

0 2 1

1 2 1

2 2 1

.....

Wurde in der Tracekonfiguration keine Abtastrate eingestellt, so steht in der ersten Spalte der Zyklus, d.h. jeweils ein Wert pro Zyklus wurde erfasst. Im anderen Fall wird hier der Zeitpunkt in ms eingetragen, an dem die Werte der Variablen ab dem Start der Traceaufzeichnung abgespeichert wurden.

In den darauffolgenden Spalten werden die entsprechenden Werte der Tracevariablen abgespeichert. Die Werte sind jeweils durch ein Leerzeichen voneinander getrennt.

Die zugehörigen Variablennamen werden in der dritten Zeile der Reihenfolge nach nebeneinander dargestellt (PLC_PRG.ZAEHLER, PLC_PRG.LIGHT1).

6.6 Watch- und Rezepturverwalter

Mit Hilfe des Watch- und Rezepturverwalters können die Werte von ausgesuchten Variablen angezeigt werden. Der Watch- und Rezepturverwalter ermöglicht auch die Variablen mit bestimmten Werten vorzubelegen und auf einmal an die Steuerung zu übertragen ('**Rezeptur schreiben**'). Genauso können aktuelle Werte der Steuerung als Vorbelegung in den Watch- und Rezepturverwalter eingelesen und abgespeichert werden ('**Rezeptur lesen**'). Hilfreich sind diese Funktionen z.B. für die Einstellung und Erfassung von Regelungsparametern.

Alle erzeugten Watchlisten ('**Einfügen' 'Neue Watchliste**') werden in der linken Spalte des Watch- und Rezepturverwalter angezeigt und können mit einem Mausklick oder den Pfeiltasten ausgewählt werden. Im rechten Bereich des Watch- und Rezepturverwalters werden die jeweils zugehörigen Variablen angezeigt.

Um mit dem Watch- und Rezepturverwalter zu arbeiten, öffnen Sie das Objekt **Watch- und Rezepturverwalter** in der Registerkarte **Ressourcen** im Object Organizer.

Im **Offline Modus** kann man im Watch- und Rezepturverwalter mehrere Watchlisten durch den Befehl '**Einfügen' 'Neue Watchliste**' erzeugen.

Zum Eingeben der zu beobachtenden Variablen kann eine Liste aller Variablen mit der Eingabehilfe aufgerufen werden oder man gibt die Variablen mit der Tastatur nach folgender Notation ein:

<Bausteinname>.<Variablenname>.

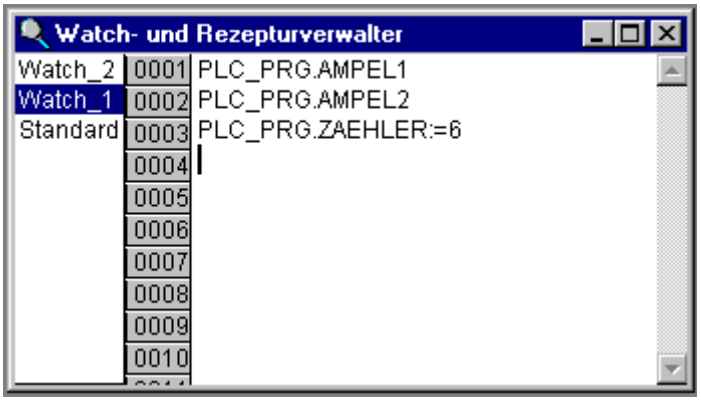
Bei globalen Variablen fehlt der Bausteinname. Sie beginnen mit einem Punkt. Der Variablenname kann wiederum mehrstufig sein. Adressen können direkt eingegeben werden.

Beispiel für eine mehrstufige Variable:

PLC_PRG.Instanz1.Instanz2.Struktur.Komponentenname

Beispiel für eine globale Variable:

.global1.component1



Watch- und Rezepturverwalter im Offline Modus

Die Variablen der Watchliste können mit konstanten Werten vorbelegt werden, d.h. im Online Modus können diese Werte mit dem Befehl '**Extras' 'Rezeptur schreiben'** in die Variablen geschrieben werden. Dazu muss der konstante Wert mit := der Variablen zugewiesen werden:

Beispiel:

PLC_PRG.TIMER:=50

Im Beispiel ist die Variable PLC_PRG.ZAEHLER mit dem Wert 6 vorbelegt.

'Einfügen' 'Neue Watchliste'

Mit diesem Befehl fügen Sie in den Watch- und Rezepturverwalter eine neue Watchliste ein. Geben Sie im erscheinenden Dialog den gewünschten Namen der Watchliste ein.

'Extras' 'Watchliste Umbenennen'

Mit diesem Befehl kann der Name einer Watchliste im Watch- und Rezepturverwalter geändert werden. Geben Sie im erscheinenden Dialog den neuen Namen der Watchliste ein.

'Extras' 'Watchliste speichern'

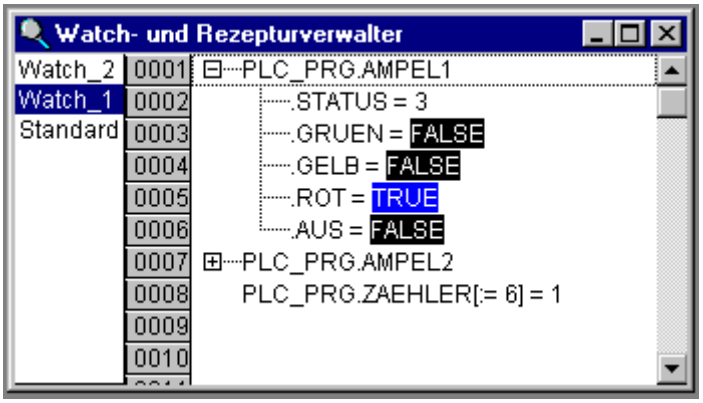
Mit diesem Befehl kann eine Watchliste abgespeichert werden. Es öffnet sich der Dialog zum Speichern einer Datei. Der Dateiname ist vorbelegt mit dem Namen der Watchliste und erhält den Zusatz "*.wtc". Die gespeicherte Watchliste kann mit 'Extras' 'Watchliste laden' wieder geladen werden.

'Extras' 'Watchliste laden'

Mit diesem Befehl kann eine abgespeicherte Watchliste wieder geladen werden. Es öffnet sich der Dialog zum Öffnen einer Datei. Wählen Sie die gewünschte Datei mit dem Zusatz "*.wtc". Im erscheinenden Dialog können Sie der Watchliste einen neuen Namen geben. Vorbelegt ist der Dateiname ohne Zusatz. Mit **'Extras' 'Watchliste speichern'** kann eine Watchliste abgespeichert werden.

Watch- und Rezepturverwalter im Online Modus

Im Online Modus werden die Werte der eingegebenen Variablen angezeigt. Strukturierte Werte (Arrays, Strukturen oder Instanzen von Funktionsblöcken) sind durch ein Pluszeichen vor dem Bezeichner gekennzeichnet. Mit Mausklick auf das Pluszeichen oder Drücken der <Eingabetaste>, wird die Variable auf- bzw. zugeklappt. Um neue Variablen einzugeben, kann die Anzeige durch den Befehl '**Extra"Monitoring aktiv**' ausgeschaltet werden. Nachdem die Variablen eingegeben sind, können Sie mit demselben Befehl wieder das Anzeigen der Werte aktivieren.



Watch- und Rezepturverwalter im Online Modus

Im Offline Modus können Variablen mit konstanten Werten vorbelegt werden (durch Eingabe von := <Wert> nach der Variablen). Im Online Modus können nun diese Werte mit dem Befehl 'Extras' 'Rezeptur schreiben' in die Variablen geschrieben werden. Mit dem Befehl 'Extras' 'Rezeptur lesen', wird die Vorbelegung der Variablen mit dem aktuellen Wert der Variablen ersetzt.

Es werden nur die Werte **einer** Watchliste geladen, die im Watch- und Rezepturverwalter ausgewählt wurde!

'Extra''Monitoring aktiv'

Mit diesem Befehl wird beim Watch- und Rezepturverwalter im Online Modus die Anzeige ein- bzw. ausgeschaltet. Ist die Anzeige aktiv, erscheint ein Haken vor dem Menüpunkt. Um neue Variablen einzugeben oder einen Wert vorzubelegen (siehe Offline Modus), muss die Anzeige durch den Befehl ausgeschaltet werden. Nachdem die Variablen eingegeben sind, können Sie mit demselben Befehl wieder das Anzeigen der Werte aktivieren.

'Extras' 'Rezeptur schreiben'

Mit diesem Befehl können im Online Modus des Watch- und Rezepturverwalters die vorbelegten Werte (siehe Offline Modus) in die Variablen geschrieben werden.

'Extras' 'Rezeptur lesen'

Mit dem Befehl wird im Online Modus des Watch- und Rezepturverwalters die Vorbelegung der Variablen (siehe Offline Modus) mit dem aktuellen Wert der Variablen ersetzt.

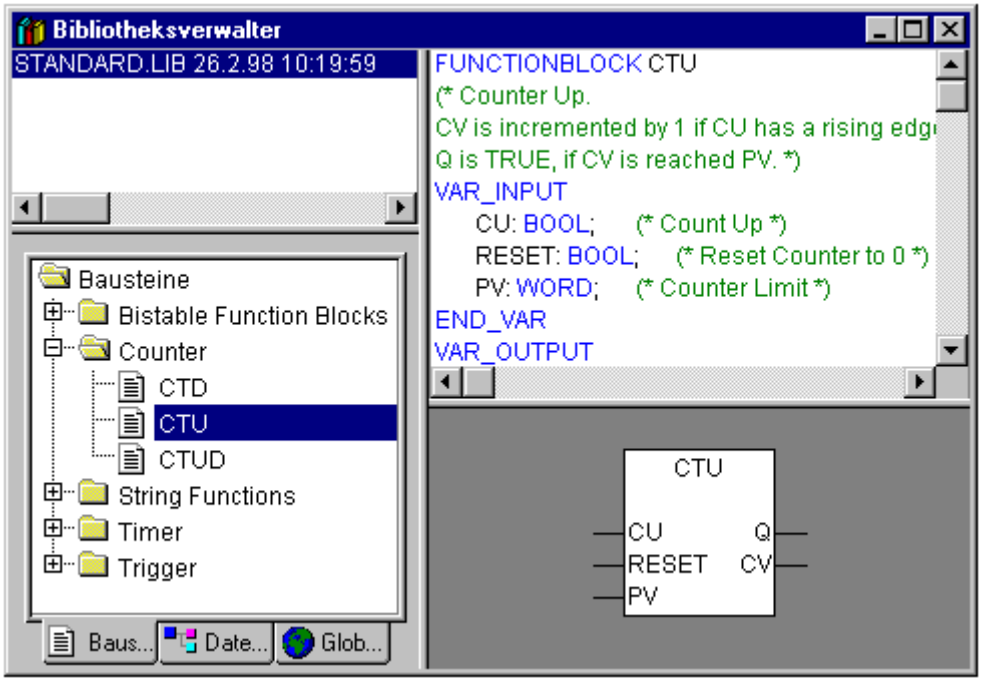
Beispiel:

```
PLC_PRG.Zaehler [:= <aktueller Wert>] =
<aktueller Wert>
```

Werte forcen Sie können im Watch- und Rezepturverwalter auch 'Werte forcen' und 'Werte schreiben'. Wenn Sie auf den jeweiligen Variablenwert klicken, dann öffnet ein Dialog, in dem Sie den neuen Wert der Variablen eingeben können. Geänderte Variablen erscheinen im Watch- und Rezepturverwalter rot.

7 Die Bibliotheksverwaltung

Der Bibliotheksverwalter zeigt alle Bibliotheken, die an das aktuelle Projekt angeschlossen sind. Die Bausteine, Datentypen und globale Variablen der Bibliotheken können wie selbstdefinierte Bausteine, Datentypen und globale Variablen verwendet werden. Der Bibliotheksverwalter wird mit dem Befehl '**Fenster"Bibliotheksverwaltung**' geöffnet. Die Information über die eingebundenen Bibliotheken wird mit dem Projekt gespeichert.



Bibliotheksverwalter

Bibliotheksverwalter nutzen

Das Fenster des Bibliotheksverwalter ist durch Bildschirmteiler in drei bzw. vier Bereiche aufgeteilt. Im linken oberen Bereich sind die dem Projekt angeschlossenen Bibliotheken aufgelistet. In dem darunter liegenden Bereich werden die Bausteine, Datentypen oder Globale Variablen der im oberen Bereich gewählten Bibliothek aufgelistet, je nach gewählter Registerkarte. Ordner werden mit Doppelklick auf die Zeile oder Drücken der <Eingabetaste> auf- und zugeklappt. Vor zugeklappten Ordnern befindet sich ein Pluszeichen, vor aufgeklappten ein Minuszeichen. Wird ein Baustein durch Mausklick oder Auswahl per Pfeiltasten selektiert, so erscheint im rechten Bereich des Bibliotheksverwalters oben die Deklaration des Bausteins und unten die grafische Darstellung als Black Box mit Ein- und Ausgängen. Bei Datentypen und globalen Variablen wird im rechten Bereich des Bibliotheksverwalters die Deklaration angezeigt.

Standardbibliothek

Die Bibliothek 'standard.lib' steht Ihnen standardmäßig zur Verfügung. Sie enthält alle Funktionen und Funktionsbausteine, die von der IEC61131-3 als Standardbausteine für ein IEC-Programmiersystem gefordert werden. Der Unterschied zwischen einer Standardfunktion und einem Operator ist, dass der Operator implizit dem Programmiersystem bekannt ist, während die Standardbausteine als Bibliothek an das Projekt gebunden werden müssen (standard.lib).

Benutzerdefinierte Bibliotheken

Ist ein Projekt fertig und fehlerfrei zu übersetzen, so kann es mit dem Befehl 'Speichern unter' im Menü 'Datei' in einer Bibliothek abgelegt werden. Das Projekt selbst bleibt unverändert. Anschließend steht es wie die Standardbibliothek unter dem eingegebenen Namen zur Verfügung.

'Einfügen' 'weitere Bibliothek'

Mit diesem Befehl können Sie eine weitere Bibliothek an Ihr Projekt binden. Wählen Sie im Dialog zum Öffnen einer Datei die gewünschte Bibliothek mit dem Zusatz "*.lib". Die Bibliothek wird nun im Bibliotheksverwalter mit aufgelistet und Sie können die Objekte der Bibliothek wie selbst definierte Objekte verwenden.

Bibliothek entfernen

Mit dem Befehl 'Bearbeiten' 'Löschen', können Sie eine Bibliothek aus einem Projekt und dem Bibliotheksverwalter entfernen.

Eigenschaften

Dieser Befehl öffnet den Dialog 'Informationen zu interner (bzw. externer) Bibliothek'. Für interne Bibliotheken enthält er einschließlich der Statistik die Daten, die beim Erstellen der Bibliothek als Projektinformationen eingegeben wurden. Für externe Bibliotheken zeigt er den Bibliotheksnamen und –pfad an.

8 Engineering Interface (ENI)

Die Schnittstelle ENI ('ENgineering Interface') ermöglicht den Zugriff aus dem Programmiersystem auf eine externe Projektdatenbank, in der Daten, die während der Erstellung eines Automatisierungsprojektes anfallen, verwaltet werden. Die Verwendung einer externen Datenbank gewährleistet die Konsistenz der Daten, die dann von mehreren Anwendern, Projekten und Programmen gemeinsam genutzt werden können und ermöglicht folgende Erweiterungen in der Funktionalität:

- **Versionsverwaltung** für Projekte und zugehörige Ressourcen (gemeinsam genutzte Objekte): Wurde ein Objekt aus der Datenbank ausgecheckt, modifiziert und wieder eingecheckt, wird in der Datenbank eine neue Version des Objekts erzeugt, die alten Versionen bleiben jedoch erhalten und können bei Bedarf ebenfalls wieder abgerufen werden. Für jedes Objekt und für ein gesamtes Projekt wird die Änderungshistorie aufgezeichnet. Versionen können auf Unterschiede geprüft werden. (Gilt nicht bei Verwendung eines lokalen Dateisystems als Datenbank.)
- **Mehrbenutzerbetrieb**: Die neueste Version einer Bausteinsammlung, z.B. der Bausteine eines Projekts, kann einer Gruppe von Anwendern zugänglich gemacht werden. Die von einem Benutzer ausgecheckten Bausteine sind für die anderen Benutzer als 'in Bearbeitung' markiert und nicht veränderbar. Somit können mehrere Anwender parallel am gleichen Projekt arbeiten, ohne gegenseitig Objektversionen zu überschreiben.
- **Zugriff durch externe Programme**: Neben dem TwinCAT PLC Control können auch andere Tools, die ebenfalls über die ENI-Schnittstelle verfügen, auf die gemeinsame Datenbank zugreifen. Beispielsweise externe Visualisierungen, ECAD-Systeme etc., die die in TwinCAT PLC Control erzeugten Date benötigen oder selbst Daten erzeugen.

Damit die Datenbank, um den Mehrbenutzerbetrieb zu ermöglichen, auch auf einem anderen Rechner liegen kann, setzt sich die ENI-Schnittstelle aus einem Client und einem Serverteil zusammen. Das Programmiersystem ist ebenso ein Client des eigenständigen ENI Server Prozesses wie gegebenenfalls eine andere Applikation, die Zugang zur Datenbank braucht.

Aktuell unterstützt die ENI-Schnittstelle die Datenbanken '**Visual SourceSafe 5.0' und 'Visual SourceSafe 6.0',** und ein lokales Dateisystem. Objekte können dort in verschiedenen '**Ordnern**' (Kategorien mit unterschiedlichen Zugriffseigenschaften) abgelegt werden, für die Bearbeitung ausgecheckt und damit für andere Benutzer gesperrt werden. Der aktuelle Stand der Objekte kann aus der Datenbank abgerufen werden. Gleichzeitig können weiterhin Objekte nur lokal, also im Projekt gehalten werden. Die *.pro Datei ist die lokale Arbeitskopie eines in der Datenbank verwalteten Projekts.

Voraussetzungen für das Arbeiten mit einer ENI Projektdatenbank

Um die ENI-Schnittstelle im TwinCAT PLC Control für das Verwalten der Projektobjekte in einer externen Datenbank nutzen zu können, müssen untenstehende Punkte erfüllt sein:

Zur Installation und Benutzung des Standard ENI Servers sehen Sie bitte die zugehörige Server-Dokumentation bzw. Online Hilfe. Dort finden Sie auch eine Quickstart-Anleitung, die den Aufbau einer einfachen ersten Verbindung zwischen einem Projekt und einer ENI-Datenbank beschreibt. Beachten Sie auch die Möglichkeit, in Zusammenhang mit dem ENI Server den ENI Explorer zu verwenden. Dieser bietet außerhalb der aktuell verwendeten Datenbank eine Oberfläche zum Ausführen und Überwachen der Datenbankfunktionen.

- zur Verbindung zwischen dem TwinCAT PLC Control und ENI Server muss TCP/IP verfügbar sein, da der ENI Server das HTTP-Protokoll verwendet.
- ein ENI-Server muss lokal oder auf einem anderen Rechner installiert und gestartet werden (ENI Server Suite). Eine gültige Lizenz ist erforderlich, um alle Standard-Datenbanktreiber verfügbar zu haben. Lizenzfrei kann nur der lokale Dateisystem-Treiber verwendet werden.
- in der ENI Server Administration (ENIAdmin.exe) muss folgendes konfiguriert sein:
  - der Anwender muss als Benutzer mit Zugangsrechten registriert sein (User Management)
- die Zugriffsrechte zu den Verzeichnissen in der Datenbank müssen korrekt gesetzt sein (Access Rights)
- **Empfehlung**: das Administrator Passwort für den Zugang zu ENIAdmin.exe und ENIControl.exe sollte unmittelbar nach der Installation definiert werden (Admin Password)

- im Service Kontrollprogramm ENI Control muss die Verbindung zur gewünschten Datenbank korrekt konfiguriert sein (Database). Eine erste Einstellung wird bereits während des Setups bei der Installation verlangt, diese kann in ENI Control auch wieder verändert werden.
- eine Projektdatenbank, für die es einen Treiber gibt, den der ENI Server unterstützt; muss installiert sein; sinnvoll ist, dies auf dem Rechner vorzunehmen, auf dem auch der ENI Server läuft. Alternativ kann ein lokales Dateisystem verwendet werden, für das standardmäßig ebenfalls ein Treiber zur Verfügung steht.
- in der Datenbank Administration müssen gegebenenfalls sowohl der Anwender (am Client) als auch der ENI Server als Benutzer mit Zugangsrechten registriert sein. Dies gilt in jedem Fall für die Verwendung von SourceSafe als Datenbank, für andere Datenbanktreiber sehen Sie bitte die zugehörige Dokumentation für die erforderliche Benutzerkonfiguration.
- für das aktuelle Projekt muss die ENI-Schnittstelle aktiviert sein (dies erfolgt im TwinCAT PLC Control-Dialog '**Projekt' 'Optionen' 'Projektdatenbank**').
- für das aktuelle Projekt muss die Konfiguration der Verknüpfung zur Datenbank vorgenommen worden sein; dies erfolgt in den TwinCAT PLC Control unter '**Projekt' 'Optionen' 'Projektdatenbank**'.
- im aktuellen Projekt muss sich der Benutzer mit Benutzername und Passwort beim ENI Server anmelden; dies erfolgt im Login-Dialog, der mit dem Befehl '**Projekt' 'Datenbankverknüpfung' 'Login**' gezielt geöffnet werden kann bzw. beim versuchten Zugang zur Datenbank automatisch geöffnet wird.

Arbeiten mit der Projektdatenbank

Die **Datenbankbefehle** (Abrufen, Auschecken, Einchecken, Versionsgeschichte, Labeln etc.) zum Verwalten der Projektbausteine in der ENI-Projektdatenbank stehen im aktuellen Projekt zur Verfügung, sobald die Verknüpfung zur Datenbank aktiviert und korrekt konfiguriert wurde. Sehen Sie hierzu unter 'Voraussetzungen für das Arbeiten mit einer ENI Projektdatenbank'. Die Befehle sind dann im Menü 'Datenbankverknüpfung' zu finden. Dieses erhält man als Untermenü des Menüs 'Projekt' bzw. im Kontextmenü für ein einzelnes Objekt, das im Object Organizer markiert ist.

Die Zuordnung eines Objekts zu einer Datenbankkategorie wird in den Objekteigenschaften angezeigt und kann dort auch verändert werden.

Die Eigenschaften der Datenbankkategorien (Verbindungsparameter, Zugriffsrecht, Verhalten bei Aus- und Einchecken) können in den Optionsdialogen der Projektdatenbank-Optionen verändert werden.

Kategorien innerhalb der Projektdatenbank

Die Objekte eines Projekts können hinsichtlich der Versionsverwaltung in vier Kategorien gesehen werden:

- Die ENI-Schnittstelle unterscheidet drei Kategorien (ENI-Objektkategorien) von Objekten, die im Datenablagesystem verwaltet werden: Projektobjekte, Gemeinsame Objekte, Übersetzungsobjekte.
- Ein Objekt kann aber auch der Kategorie 'Lokal' angehören, wenn es nicht in der Datenbank abgelegt, sondern wie herkömmlich nur mit dem Projekt gespeichert werden sollen.

Im Programmiersystem kann demzufolge ein Baustein einer der Kategorien Projektobjekte, Gemeinsame Objekte oder Lokal zugeordnet werden; die Übersetzungsdaten existieren im Projekt ja noch nicht als zuordenbare Objekte.

Das Zuordnen eines Objekts zu einer Kategorie erfolgt automatisch beim Erstellen, entsprechend der Voreinstellung in den Projektdatenbank-Optionen, kann aber jederzeit im Objekteigenschaften-Dialog verändert werden.

Jede ENI-Objektkategorie wird im Dialog '**ENI-Einstellungen' (Projektoptionen, Kategorie Projektdatenbank)**, separat konfiguriert. Das heißt, sie erhält eigene Parameter bezüglich der Verbindung zur Datenbank (Verzeichnis, Port, Zugriffsrecht etc.) und bezüglich des Verhaltens beim Abrufen, Aus- und Einchecken. Diese Einstellungen gelten dann für alle Objekte des Projekts, die dieser Kategorie angehören. Auch die Zugangsdaten (Benutzername, Passwort) beim Verbinden zur Datenbank sind dementsprechend für jede Kategorie separat einzugeben. Dazu steht der Login-Dialog zur Verfügung ('**Projekt' 'Datenbankverknüpfung' 'Login**').

Es bietet sich an, in der jeweiligen Datenbank für jede ENI-Objektkategorie ein eigenes Verzeichnis für die Objekte anzulegen, es ist jedoch auch möglich, die Objekte aller Kategorien im selben Verzeichnis zu halten, da die Kategoriezuordnung eine Eigenschaft des Objektes ist und nicht des Verzeichnisses.

Folgende sind die drei möglichen ENI-Objektkategorien:

Voraussetzungen

	Beschreibung
Projekt	für Objekte, die projektspezifische Source-Informationen darstellen, z.B. gemeinsam genutzte Bausteine innerhalb eines Projekts, wichtig für den Mehrbenutzerbetrieb. Beim Befehl 'Alles Abrufen' werden automatisch alle Objekte dieser Kategorie aus dem Projektverzeichnis der Datenbank ins lokale Projekt geholt, auch diejenigen, die dort noch nicht angelegt waren.
Gemeinsame Objekte	für allgemeingültige, projektunabhängige Objekte, etwa Bausteinbibliotheken, die normalerweise von mehreren Anwendern in verschiedenen Projekten benutzt werden. **Achtung**: Beim Befehl 'Alles Abrufen' werden nur diejenigen Objekte dieser Kategorie aus dem Projektverzeichnis der Datenbank ins lokale Projekt kopiert, die dort bereits angelegt sind.
Übersetzungsdateien	für die automatisch vom TwinCAT PLC Control erzeugten projektspezifischen Übersetzungsinformationen (z.B. Symboldateien), die auch von anderen Tools benötigt werden. Beispielsweise benötigt eine Visualisierung die Variablen eines Programmiersystems inklusive der Adressen, die allerdings erst beim Kompilieren vergeben werden

Wahlweise können Projektbausteine auch von der Verwaltung in der Projektdatenbank ausgenommen und ausschließlich lokal, also wie herkömmlich nur mit dem Projekt gespeichert werden.

9 Visualisierung

Um die Daten einer mit TwinCAT PLC Control programmierbaren Steuerung zu visualisieren, also beobachten und bedienen zu können, ist kein zusätzliches Tool erforderlich. Das Programmiersystem beinhaltet einen integrierten Visualisierungs-Editor , so dass der Anwender im Kontext der Applikationsentwicklung bereits Visualisierungsmasken in ein- und derselben Oberfläche erzeugen kann.

Die Integration bietet dabei viele Vorteile:

Die in TwinCAT PLC Control integrierte Visualisierung benötigt keine Tag-Liste und kann direkt auf die Variablen aus der Steuerung zugreifen. Die oft schwer zu konfigurierende OPC Schicht entfällt, weil die Kommunikation über denselben Mechanismus erfolgt, der auch für das Programmiersystem verwendet wird. So wird der Engineering-Aufwand für die Realisierung von Visualisierungen erheblich reduziert.

Ablauf-Varianten

1. Direkt im Programmiersystem

Direkt im Programmiersystem [▶ 198]

Zum Testen der erstellten Visualisierungsmasken, aber auch für Service- und Diagnosezwecke direkt in Verbindung mit Ihrer Steuerung benötigen Sie kein weiteres Tool: Im Online-Modus erhalten Sie sofort die reale Darstellung der Visualisierungen innerhalb des Programmiersystems.

2. Target-Visualisierung (TwinCAT PLC HMI CE)

Target-Visualisierung [▶ 263]

Für Steuerungen mit eingebautem Display können die Visualisierungsinformationen aus dem Programmiersystem mit der Applikation ins Zielsystem geladen werden. Sie werden dort automatisch zur Anzeige gebracht. Diese Lösung kann mit geringem Aufwand auf beliebige mit TwinCAT PLC Control programmierbare Geräte portiert werden.

Der Funktionsumfang im Überblick:

Die unterschiedlichen Visualisierungs-Varianten des TwinCAT Systems haben Differenzen in ihrem Funktionsumfang. Diese Unterschiede sollen in der folgenden Tabelle [▶ 263] dargestellt werden.

• **Elemente**
- Rechteck, Ellipse, Abgerundetes Rechteck
- Linie, Polygon, Linienzug, Kurvenzug
- Bitmap, WMF-Datei
- Schaltfläche, Tabelle, Histogramm, Balkenanzeige, Anzeigeinstrument
- Referenz auf andere Visualisierung

• **Animationen (abhängig vom Element-Typ):**
- Textanzeige
- Farbumschlag
- Sichtbar/Unsichtbar
- Verschiebung
- Rotation
- Skalierung
- Offset auf einzelne Kanten eines Objekts (für Bargraph)
- Button aktiv/inaktiv
- Aktuelle Zeile (nur Textanzeige)

• **Eingabemöglichkeiten:**
- Booleschen Wert toggeln/tasten
- Texteingabe
- Bildwechsel

- Sonderaktion (Visualisierung verlassen, Rezeptur lesen/schreiben, Sprache umschalten, externes EXE aufrufen etc.)
- Zeile anwählen (nur Textanzeige)

• **Sonstige Eigenschaften**
- Sprachumschaltung
- Tooltips für alle Elemente
- ASCII Import/Export
- Hintergrund Bitmap
- Automatische Skalierung
- Zeichenoperationen: Ausrichten, Anordnen, Gruppieren
- Platzhalter-Konzept zur Bausteinbildung von komplexen, grafischen Elementen
- Programmierte Visualisierungs-Ausdrücke

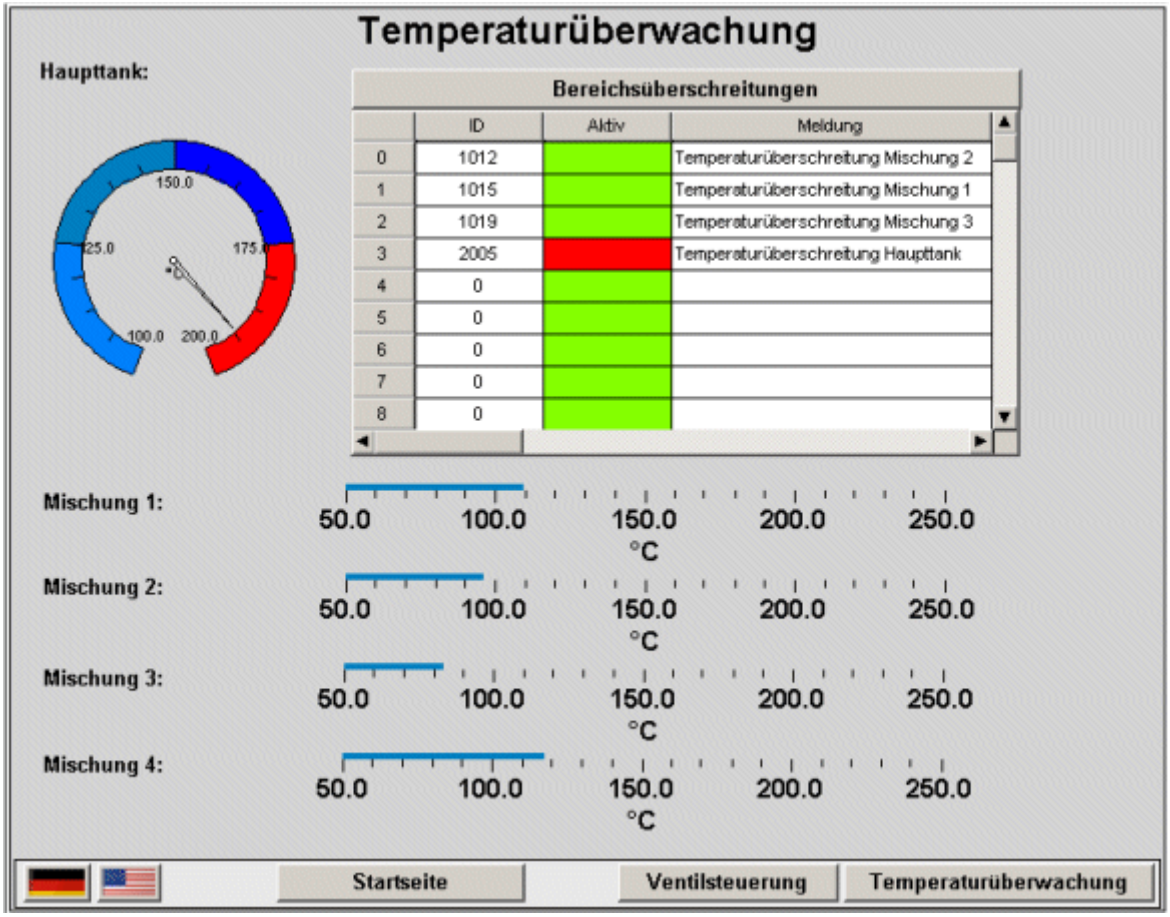
9.1 Visualisierungseditor

Die TwinCAT PLC Control Visualisierung dient der **Beobachtung und Bedienung eines mit TwinCAT PLC Control Visualisierung erstellten Steuerungsprogramms**. Der **Visualisierungseditor** stellt zu diesem Zweck grafische Elemente bereit, die entsprechend angeordnet und mit Projektvariablen verknüpft werden können.

Im **Online Betrieb** verändert sich dann die Darstellung der Elemente in Abhängigkeit von den Variablenwerten.

Einfaches Beispiel:

Sie zeichnen zur Darstellung eines Füllstandes, der im Programm ausgewertet wird, einen Balken, der in Abhängigkeit von der entsprechenden Projektvariable seine Länge oder Farbe verändert, darunter eine Textanzeige des aktuellen Messwerts und eine Schaltfläche zum Starten und Stoppen des Programms.



Beispiel einer Visualisierung

Die Eigenschaften eines Visualisierungselements wie auch einer Visualisierung als Gesamtobjekt werden in entsprechenden **Konfigurationsdialogen** festgelegt. Es können sowohl bestimmte Grundeinstellungen vorgenommen werden, als auch über das Eintragen von Projektvariablen eine dynamische Parametrierung erfolgen.

Besondere zusätzliche Möglichkeiten eröffnet die **Programmierbarkeit** der Elementeigenschaften über Strukturvariablen.

Die Option, bei der Konfiguration **Platzhalter** zu verwenden, erspart Arbeit und Zeit, wenn ein Visualisierungsobjekt mehrfach verwendet werden soll. Dies gilt insbesondere bei der Verwendung von Funktionsbaustein-Instanzen.

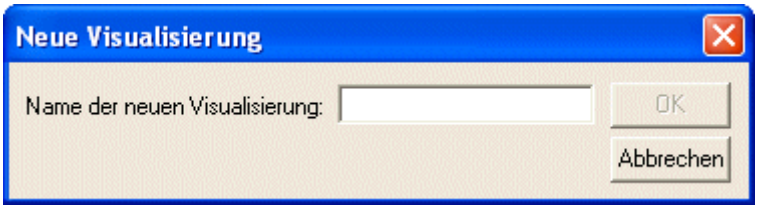
Beachten Sie auch die Möglichkeit, für einzelne Visualisierungen unterschiedliche **Tastenbelegungen** zu definieren.

9.1.1 Visualisierungsobjekt anlegen

Ein Visualisierungsobjekt ist ein TwinCAT PLC Control Objekt, das im Registerblatt Visualisierungen verwaltet wird. Es enthält eine Anordnung von Visualisierungselementen und kann als Gesamtobjekt bestimmte Eigenschaften erhalten. Es können ein oder mehrere Visualisierungsobjekte in einem TwinCAT PLC Control Projekt angelegt und miteinander verknüpft werden. Um ein Visualisierungsobjekt im Object Organizer anzulegen, wählen Sie die Registerkarte

 **Visualisierung**
und den Befehl **'Projekt' 'Objekt einfügen'**.

Der Dialog 'Neue Visualisierung' wird geöffnet,



in dem Sie den Namen der neuen Visualisierung eingeben können. Nach einer gültigen Eingabe, d.h. kein bereits vergebener Name oder Sonderzeichen wurden verwendet, können Sie den Dialog mit OK schließen. Es öffnet sich ein Fenster, in dem Sie die neue Visualisierung editieren können.

HINWEIS

Benennung des Visualisierungsobjekts

Eine Visualisierung sollte nicht den gleichen Namen tragen wie ein anderer Baustein im Projekt. Bei Visualisierungswechseln kann es sonst zu Problemen kommen.

9.1.2 Visualisierungselemente einfügen

Ein Visualisierungselement ist ein grafisches Element, das beim Erstellen eines Visualisierungsobjekts verwendet wird. Die möglichen Elemente werden in der TwinCAT PLC Control Menüleiste angeboten. Jedes Element erhält eine separate Konfiguration.

Sie können verschiedene geometrische Formen sowie Bitmaps, Metafiles, Schaltflächen und bestehende Visualisierungen in Ihre Visualisierung einfügen.

Gehen Sie auf den Menüpunkt **'Einfügen'** und wählen nach Bedarf die Befehle **'Rechteck'**, **'Abgerundetes Rechteck'**, **'Ellipse'**, **'Polygon'**, **'Linienzug'**, **'Kurve'**, **'Kreissektor'**, **'Bitmap'**, **'Visualisierung'**, **'Schaltfläche'**, **'Tabelle'**, **'Zeigerinstrument'**, **'Balkenanzeige'**, **'Histogramm'**, **'Alarmtabelle'**, **'Trend'**, **'WMF-Datei'** aus.
Vor dem gewählten Befehl erscheint ein Haken. Sie können auch die Symbole der Funktionsleiste

verwenden. Das gewählte Element erscheint hier gedrückt (z.B. ).
Wenn Sie nun mit der Maus in das Editierfenster fahren, sehen Sie, dass der Mauszeiger mit dem entsprechende Symbol (z.B. ) gekennzeichnet ist. Klicken Sie an den gewünschten Anfangspunkt Ihres Elements und verschieben bei gedrückter linker Maustaste den Zeiger bis das Element die gewünschten Maße hat.

Wollen Sie ein Polygon oder eine Linie erstellen, klicken Sie zuerst mit der Maus die Position der ersten Ecke des Polygons bzw. des Anfangspunkts der Linie an und danach die weiteren Eckpunkte. Durch einen Doppelklick am letzten Eckpunkt wird das Polygon geschlossen und vollständig gezeichnet bzw. die Linie beendet.

Wollen Sie eine Kurve (Beziers-Kurve) erstellen, geben Sie per Mausklick den Anfangs- und zwei weitere Punkte an, die ein umschreibendes Rechteck definieren. Nach dem dritten Mausklick wird ein Kurvenbogen gezeichnet. Sie können daraufhin durch Verschieben mit der Maus den Kurvenendpunkt noch verändern und durch Doppelklick abschließen oder durch zusätzliche Positionsklicks einen weiteren Bogen direkt anschließen.

Beachten Sie auch die Statusleiste und den Wechsel von Selektions- und Einfügemodus.

'Einfügen' 'Rechteck'

Symbol: 

Mit dem Befehl können Sie ein Rechteck als Element in Ihre aktuelle Visualisierung einfügen.

'Einfügen' 'Abgerundetes Rechteck'

Symbol: 

Mit dem Befehl können Sie ein Rechteck mit abgerundeten Ecken als Element in Ihre aktuelle Visualisierung einfügen.

'Einfügen' 'Ellipse'

Symbol: 

Mit dem Befehl können Sie einen Kreis oder eine Ellipse als Element in Ihre aktuelle Visualisierung einfügen.

'Einfügen' 'Polygon'

Symbol: 

Mit dem Befehl können Sie einen Polygon als Element in Ihre aktuelle Visualisierung einfügen.

'Einfügen' 'Linienzug'

Symbol: 

Mit dem Befehl können Sie eine Linie als Element in Ihre aktuelle Visualisierung einfügen.

'Einfügen' 'Kurve'

Symbol: 

Mit dem Befehl können Sie eine Beziers-Kurve als Element in Ihre aktuelle Visualisierung einfügen.

'Einfügen' 'Kreissektor'

Symbol: 

Mit dem Befehl können Sie einen Kreissektor als Element in Ihre aktuelle Visualisierung einfügen. Ziehen Sie mit gedrückter linker Maustaste einen Bereich in der gewünschten Größe auf. Ein Oval mit einer eingezeichneten Radiuslinie wird angezeigt, das durch Bewegen der Maus Form und Größe verändert. Das kleine schwarze Quadrat außerhalb des Elements gibt den Eckpunkt eines virtuellen Rechtecks, das das Element umgibt, an. Um Anfangs- und Endwinkel des Kreissektors festzulegen, klicken Sie nach Einfügen des Elements auf den Endpunkt der Radiuslinie auf dem Kreisbogen. Wenn Sie den Mauszeiger bei gedrückter Maustaste bewegen, erscheinen zwei kleine schwarze Quadrate, die die beiden Winkelpositionen anzeigen. Diese können ab jetzt auch separat selektiert und verschoben werden. Sollen die Winkel dynamisch definiert werden, öffnen Sie den Konfigurationsdialog der Kategorie 'Winkel' um die entsprechende Variablen einzutragen. Um erneut die Form und Größe des Elements zu verändern, klicken Sie entweder auf den Zentrumspunkt, so dass der Cursor schräg gekreuzte Pfeile anzeigt und bewegen Sie die Maus bei gedrückter rechter Taste oder mit Hilfe der Pfeiltasten. Oder Sie selektieren und verschieben das Eckpositions-Quadrat außerhalb des Elements. Um das Element zu verschieben, klicken Sie in die freie Fläche im Element so dass senkrecht gekreuzte Pfeile angezeigt werden.

'Einfügen' 'Bitmap'

Symbol: 

Mit dem Befehl können Sie ein Bitmap als Element in Ihre aktuelle Visualisierung einfügen. Ziehen Sie mit gedrückter linker Maustaste einen Bereich in der gewünschten Größe auf. Es öffnet sich der Dialog zum Öffnen einer Datei. Nachdem Sie das gewünschte Bitmap ausgewählt haben, wird es in den aufgezogenen Bereich eingefügt. Ob eine Verknüpfung zur Bitmap-Datei gespeichert werden soll, oder das Bitmap als Element fest eingefügt werden soll, kann im Bitmap-Konfigurationsdialog definiert werden.

'Einfügen' 'Visualisierung'

Symbol: 

Mit dem Befehl können Sie eine bestehende Visualisierung in Ihre aktuelle Visualisierung einfügen. Ziehen Sie mit gedrückter linker Maustaste einen Bereich in der gewünschten Größe auf. Es öffnet sich eine Auswahlliste von bestehenden Visualisierungen. Nachdem Sie die gewünschte Visualisierung ausgewählt haben, wird sie in den aufgezogenen Bereich eingefügt. Eine eingefügte Visualisierung wird auch als Referenz bezeichnet.

'Einfügen' 'Schaltfläche'

Symbol: 

Mit dem Befehl können Sie eine Schaltfläche in Ihre aktuelle Visualisierung einfügen. Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Wird für die Schaltfläche eine Toggle-Variable konfiguriert, visualisiert sie den Zustand dieser Variable, indem er optisch als gedrückt bzw. nicht gedrückt dargestellt wird. Umgekehrt wird natürlich die Variable durch "Drücken" der Schaltfläche getoggelt.

'Einfügen' 'WMF-Datei'

Symbol: 

Mit dem Befehl können Sie ein Windows Metafile einfügen. Es öffnet sich der Standarddialog zum Datei Öffnen, in dem nach Dateien mit der Erweiterung *.wmf gesucht werden kann. Nach Auswahl einer Datei und Schließen des Dialogs über OK wird die Datei als Element in die aktuelle Visualisierung eingefügt. Hierbei wird keine Verknüpfung auf eine Datei gespeichert, wie beim Bitmap, sondern die enthaltenen Elemente der wmf-Datei werden als Gruppe eingefügt.

'Einfügen' 'Tabelle'

Symbol: 

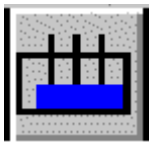
Mit dem Befehl können Sie eine Tabelle in Ihre aktuelle Visualisierung einfügen, die der Darstellung der Elemente eines Arrays dient. Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Noch bevor das Element in seiner endgültigen Aussehen dargestellt wird, erscheint der Konfigurationsdialog 'Tabelle konfigurieren'. Neben den auch bei anderen Elementen verfügbaren Kategorien Tooltip und Zugriffsrechte sind hier die Kategorien 'Tabelle', 'Spalten' und 'Spez.Tabelle' zum Definieren von Inhalt und Darstellungsweise der Tabelle verfügbar.

'Einfügen' 'Zeigerinstrument'

Symbol: 

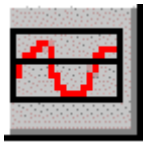
Mit dem Befehl können Sie ein Zeigerinstrument in Ihre aktuelle Visualisierung einfügen. Es bietet eine Skala, die auf einem zu definierenden Kreisbogenausschnitt liegt, und einen Zeiger, der seinen Ursprung in der virtuellen Kreismitte hat.
Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Noch bevor das Element in seiner endgültigen Aussehen dargestellt wird, erscheint der Konfigurationsdialog Zeigerinstrument konfigurieren. Hier können verschiedene Parameter definiert und die Auswirkung auf die Darstellung in einer Vorschau betrachtet werden, bevor das Element nach Bestätigen mit OK endgültig eingefügt wird.

'Einfügen' 'Balkenanzeige'

Symbol: 

Mit dem Befehl können Sie eine Balkenanzeige in Ihre aktuelle Visualisierung einfügen. Es dient der Darstellung eines Variablenwerts in Form eines Balkens, der sich entlang einer horizontalen Skala bewegt. Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Noch bevor das Element in seiner endgültigen Aussehen dargestellt wird, erscheint der Konfigurationsdialog Balkenanzeige konfigurieren. Hier können verschiedene Parameter definiert und die Auswirkung auf die Darstellung in einer Vorschau betrachtet werden, bevor das Element nach Bestätigen mit OK endgültig eingefügt wird:

'Einfügen' 'Histogramm'

Symbol: 

Mit dem Befehl können Sie ein Histogramm in Ihre aktuelle Visualisierung einfügen. Mit diesem Element können die Elemente eines Arrays nebeneinander als Balken dargestellt werden, die entsprechend des Variablenwerts ihre Höhe ändern. Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Noch bevor das Element in seiner endgültigen Aussehen dargestellt wird, erscheint der Konfigurationsdialog Histogramm konfigurieren. Hier können verschiedene Parameter definiert und die Auswirkung auf die Darstellung in einer Vorschau betrachtet werden, bevor das Element nach Bestätigen mit OK endgültig eingefügt wird.

'Einfügen' 'Alarmtabelle'

Symbol: 

Mit dem Befehl können Sie eine Alarmtabelle in Ihre aktuelle Visualisierung einfügen. Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Noch bevor das Element in seiner endgültigen Aussehen dargestellt wird, erscheint der Konfigurationsdialog 'Alarmtabelle konfigurieren'. Neben den auch bei anderen Elementen verfügbaren Kategorien Tooltip und Zugriffsrechte sind hier die Kategorien 'Alarmtabelle', 'Sortiereigenschaften', 'Spalten' und 'Auswahleigenschaften' zum Definieren von Inhalt und Darstellungsweise verfügbar. Sie können die Alarmtabelle verwenden, um die in der Alarmkonfiguration des Projekts definierten Alarme zu visualisieren.

'Einfügen' 'Trend'

Symbol: 

Mit dem Befehl können Sie ein Trend-Element in Ihre aktuelle Visualisierung einfügen. Ziehen Sie das Element mit gedrückter linker Maustaste in der gewünschten Größe auf. Die Konfiguration der Darstellungsart (Achsen, Variablen, Historie) erfolgt im Konfigurationsdialog der Kategorie 'Trend'. Das Trend- oder auch Oszilloskop-Element dient der Darstellung von Variablen über einen gewissen Zeithorizont. Das Trendelement speichert die Daten auf Client-Seite und stellt diese als Graphen dar. Sobald sich ein Wert innerhalb des Graphen ändert, wird ein neuer Eintrag in der Datei vorgenommen, welcher Datum/Zeit und die neuen Variablenwerte enthält. Das Trend-Element wird durchsichtig gezeichnet. Dadurch ist es möglich einen beliebigen Hintergrund (Farbe, Bitmap) auszuwählen.

9.1.3 Visualisierungselemente positionieren

Visualisierungselemente selektieren

Der Selektionsmodus ist standardmäßig aktiviert. Um ein Element zu selektieren, klicken Sie mit der Maus auf das Element. Ebenso können Sie durch Drücken der <Tabulator>-Taste das erste Element der Elementliste und mit jedem weiteren Drücken das jeweils nächste Element anwählen. Wenn Sie die <Tabulator> bei gleichzeitig gedrückter <Umschalt>-Taste anwenden, springen Sie in der Elementliste entsprechend rückwärts.

Um Elemente zu markieren, die unter einem anderen liegen, markieren Sie mit einem Mausklick zunächst das oberste Element. Führen Sie dann bei gedrückter <Strg>-Taste weitere Mausklicks aus, um die darunterliegenden Elemente zu erreichen.

Um mehrere Elemente zu markieren, halten Sie die <Umschalt>-Taste gedrückt und klicken Sie mit der Maus nacheinander auf die entsprechenden Elemente oder ziehen Sie bei gedrückter linker Maustaste ein Fenster über die zu markierenden Elemente auf.

Um alle Elemente zu selektieren, verwenden Sie den Befehl 'Extras' 'Alles Markieren'.

Befinden Sie sich in der Elementliste, können Sie auch von dort durch Anwahl einer Zeile das betreffende Elements in der Visualisierung selektieren.

Selektions- und Einfügemodus wechseln

Nach dem Einfügen eines Visualisierungselements wird automatisch wieder in den Selektionsmodus gewechselt. Will man nun ein weiteres Element derselben Art einfügen, so kann man erneut den entsprechenden Befehl im Menü oder das Symbol in der Funktionsleiste auswählen. Zwischen Selektionsmodus und Einfügemodus kann außer über Menübefehl (‚Extras' ‚Selektieren') oder Symbol

 auch mit gleichzeitig gedrückter -Taste und rechte Maustaste gewechselt werden. Im Einfügemodus erscheint das entsprechende Symbol mit am Mauszeiger und in der Statusleiste wird der Name schwarz angezeigt.

'Extras' 'Selektieren'

Mit diesem Befehl wird der Selektionsmodus ein- bzw. ausgeschaltet. Dies ist ebenso über das Symbol

 oder mittels der rechten Maustaste bei gleichzeitig gedrückter **<Strg>**-Taste möglich.

Kopieren von Visualisierungselementen

Ein oder mehrere ausgewählte Elemente können Sie mit dem Befehl **'Bearbeiten' 'Kopieren',** der Tastenkombination **<Strg>+<C>** oder dem entsprechenden Symbol kopieren und mit **'Bearbeiten' 'Einfügen'** einfügen. Eine weitere Möglichkeit ist, die Elemente zu selektieren und mit gedrückter **<Strg>**-Taste noch einmal in ein Element zu klicken. Mit gedrückter linker Maustaste können Sie die nun kopierten Elemente von den ursprünglichen wegziehen.

Ändern von Visualisierungselementen

Ein bereits eingefügtes Element können Sie mit Mausklick auf das Element bzw. über einen Durchlauf per <Tabulator>-Tastenklicks auswählen. An den Ecken des Elements erscheint jeweils ein schwarzes Quadrat (bei Ellipsen an den Ecken des umschreibenden Rechtecks). Außer bei Polygonen, Linien und Kurven erscheinen weitere Quadrate in der Mitte der Elementkanten zwischen den Eckpunkten.

Bei einem ausgewählten Element wird gleichzeitig der Drehpunkt (Schwerpunkt) mit angezeigt. Um diesen Punkt rotiert dann das Element bei eingestellter Bewegung/Winkel.

Der Drehpunkt wird als kleiner schwarzer Kreis mit einem weißen Kreuz dargestellt (). Mit gedrückter linker Maustaste können Sie den Drehpunkt verschieben. Die Größe des Elements kann verändert werden, indem Sie auf eines der schwarzen Quadrate klicken und mit gedrückt gehaltener linker Maustaste die neuen Umrisse ansteuern. Bei der Auswahl eines Polygons, einer Linie oder Kurve kann so jede einzelne Ecke verschoben werden. Wird dabei die <Strg>-Taste gedrückt, wird an dem Eckpunkt ein zusätzlicher Eckpunkt eingefügt, der durch Ziehen der Maus verschoben werden kann. Mit gedrückter <Umschalt>+<Strg>-Taste kann ein Eckpunkt entfernt werden.

Verschieben von Visualisierungselementen

Ein oder mehrere ausgewählte Elemente können Sie mit gedrückter linker Maustaste oder den Pfeiltasten verschieben.

Gruppierung von Elementen

Elemente werden gruppiert, indem man mehrere Elemente selektiert und den Menüpunkt 'Extras' 'Gruppieren' ausführt. Die entstandene Gruppe verhält sich wie ein einzelnes Element:

• die gruppierten Elemente erhalten einen gemeinsamen Rahmen, mit „Ziehen" an diesem Rahmen werden alle Elemente gemeinsam gestreckt, bzw. gestaucht, Bewegungen sind nur als Gruppe möglich.

• die gruppierten Elemente erhalten gemeinsame Eigenschaften: das heißt Eingaben wirken auf die Gruppe und nicht auf die Einzelelemente. Daher haben die Elemente auch einen gemeinsamen Konfigurationsdialog. Die Eigenschaft „Farbwechsel" kann für eine Gruppe nicht vergeben werden.

Um ein einzelnes Element der Gruppe neu zu konfigurieren, muß die Gruppierung mit dem Befehl 'Extras' 'Gruppierung aufheben' aufgehoben werden, die Gruppierungskonfiguration geht dabei verloren.

'Extras' 'Nach vorn bringen'

Mit diesem Befehl bringen Sie selektierte Visualisierungselemente in den Vordergrund.

'Extras' 'Nach hinten legen'

Mit diesem Befehl legen Sie selektierte Visualisierungselemente in den Hintergrund.

'Extras' 'Ausrichten'

Mit diesem Befehl können Sie mehrere selektierte Visualisierungselemente ausrichten. Es stehen folgende Möglichkeiten der Ausrichtung zur Verfügung:

• **Links:** alle Elemente werden nach dem am weitesten links liegenden Element mit ihrer jeweils linken Kante ausgerichtet

• entsprechend **Rechts / Oben / Unten**

• **Horizontale Mitte**: alle Elemente werden im Schwerpunkt aller Elemente jeweils an deren horizontaler Mitte ausgerichtet

• **Vertikale Mitte**: alle Elemente werden im Schwerpunkt aller Elemente jeweils an deren vertikaler Mitte ausgerichtet

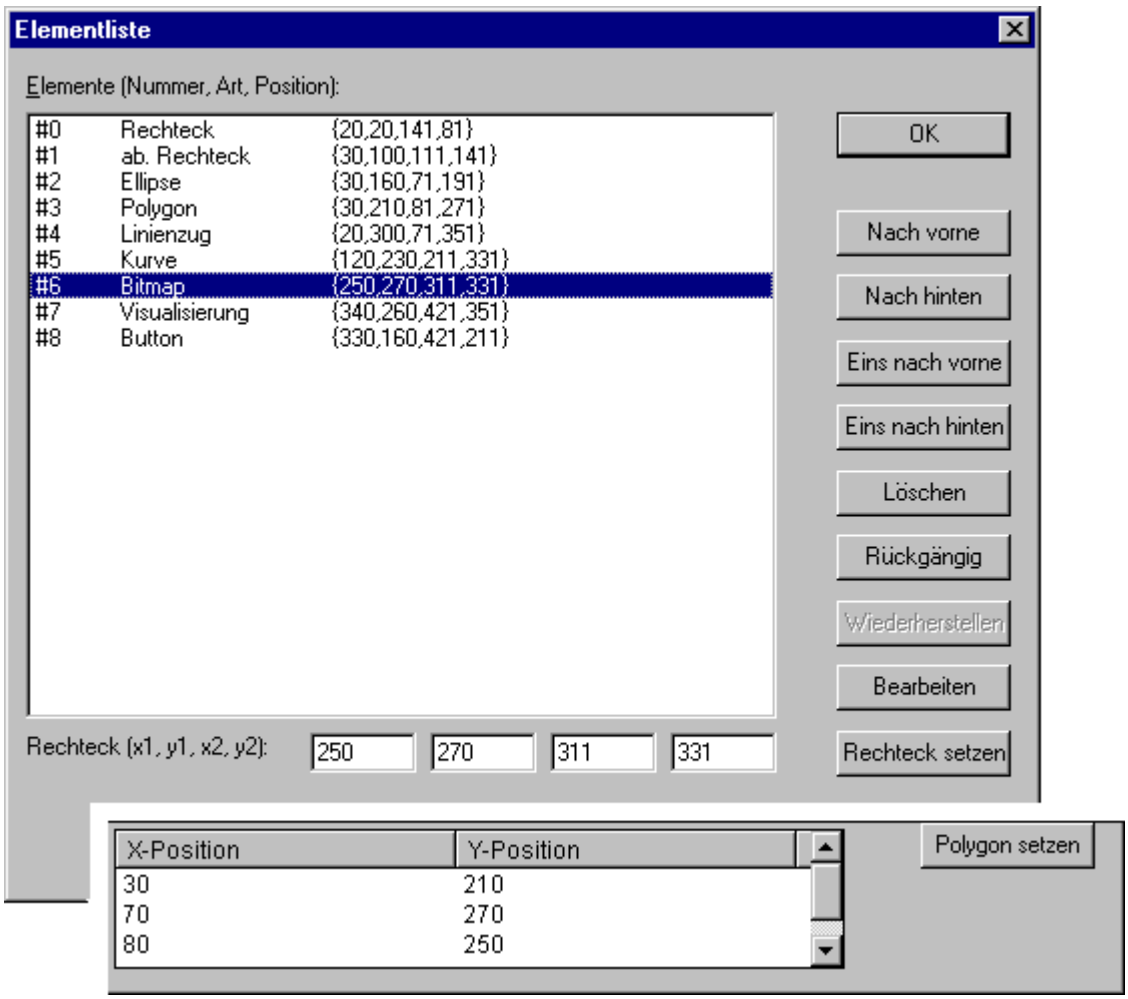
'Extras' 'Elementliste'

Mit diesem Befehl öffnet sich ein Dialog, der alle Visualisierungselemente mit ihrer **Nummer, Art und Position** auflistet. Die Position bezieht sich auf die X- und Y-Position der linken oberen (x1, y1) und rechten unteren (x2, y2) Ecke des Elements.
Wenn Sie einen oder mehrere Einträge selektieren, so werden die entsprechenden Elemente in der Visualisierung zur optischen Kontrolle markiert und ggf. wird die Anzeige zu dem Abschnitt gescrollt, in dem sich die Elemente befinden.
Mit der Schaltfläche **Eins nach vorne** wird das selektierte Visualisierungselement um eine Ebene Richtung Vordergrund verschoben. **Eins nach hinten** verschiebt das Element um eine Ebene Richtung Hintergrund.
Mit der Schaltfläche **Nach vorne** legen Sie selektierte Visualisierungselemente in den Vordergrund.
Mit der Schaltfläche **Nach hinten** legen Sie sie in den Hintergrund.

Unterhalb der Elementliste erhalten Sie abhängig vom markierten Element eine der folgenden Eingabemöglichkeiten zur Veränderung von Größe und Position des Elements:

• Handelt es sich um ein Rechteck, abgerundetes Rechteck, Ellipse, Bitmap, eine Visualisierung, Schaltfläche oder ein Metafile, erscheinen neben dem Text **Rechteck (x1, y1, x2, y2)** vier Eingabefelder, in denen die augenblicklichen x/y-Positionen angegeben sind und editiert werden können.

• Im Falle eines Linienzugs, Polygons oder einer Kurve erscheint eine Tabelle, die für jeden der Positionspunkte (die als kleine schwarze Quadrate angezeigt werden, sobald das Element markiert ist) die **X-Position** und **Y-Position** in jeweils einer Zeile anzeigt. Auch diese können hier editiert werden.
Um die neu gesetzten Positionswerte in die Elementliste und die Visualisierung zu übernehmen, drücken Sie die Schaltfläche **Rechteck setzen** (bei 1.) bzw. **Polygon setzen** (bei 2.).
Mit der Schaltfläche **Löschen** werden alle in der Elementliste selektierten Visualisierungselemente entfernt.
Mit **Rückgängig** und **Wiederherstellen** können Sie die vorgenommenen Änderungen rückgängig machen und wiederherstellen wie bei **'Bearbeiten' 'Rückgängig'** und **'Bearbeiten' 'Wiederherstellen'.** Die Vorgänge können Sie im Dialog beobachten. Zum Verlassen des Dialogs bestätigen Sie Ihre Änderungen mit OK.



Dialog Elementliste

Statusleiste in der Visualisierung

Steht der Fokus auf einer Visualisierung, wird die aktuelle **X-** und **Y-Position** des Mauszeigers in Pixel relativ zur oberen linken Ecke des Bilds in der Statusleiste angegeben. Befindet sich der Mauszeiger auf einem **Element** oder wird ein Element bearbeitet, wird die Nummer desselben angegeben. Haben Sie ein Element zum Einfügen ausgewählt, so erscheint dies ebenfalls (z.B. **Rechteck**).

9.1.4 Visualisierung konfigurieren

Es können sowohl die einzelnen grafischen Elemente einer Visualisierung als auch das Visualisierungsobjekt als ganzes konfiguriert werden. Je nach ausgewählter Einheit stehen verschiedene Konfigurationsdialoge zur Verfügung, die über den Befehl 'Konfigurieren' aus dem Menü 'Extras' bzw. dem Kontextmenü aufgerufen werden können.

Dort werden entweder feste Einstellungen gesetzt oder aber Projektvariablen angegeben, deren Werte dann im Online Modus dynamisch die betreffenden Eigenschaften bestimmen. Außerdem können diese auch über die Komponenten einer Strukturvariablen angesteuert werden, die pro Visualisierungselement gesetzt werden kann.

Beachten Sie die Auswertungsreihenfolge, die später im Online Modus gilt:
- Die dynamisch, also über normale Projektvariablen oder über die Strukturvariablen, gelieferten Werte überschreiben die festen Einstellungen der Elementkonfigurationen.
- Wenn eine Elementeigenschaft sowohl durch eine direkt im Konfigurationsdialog eingetragene Projektvariable als auch über die Komponente einer Strukturvariable angesprochen wird, wird im Online Modus zuerst der Wert der Projektvariablen ausgewertet.

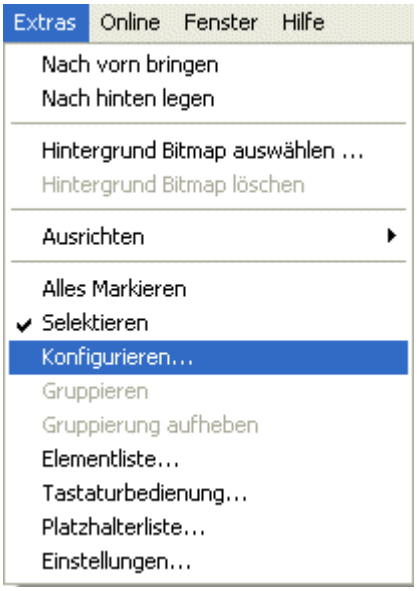
Zu beachten ist auch die Möglichkeit der Verwendung von Platzhaltern wie auch die speziellen Eingabemöglichkeiten im Hinblick auf die Verwendung einer Visualisierung mit der TwinCAT PLC Control Visualisierung, also als ausschließliche Bedienschnittstelle für ein Steuerungsprogramm.

Platzhalter

An jeder Stelle eines Konfigurationsdialogs, an dem Variablen oder Text eingegeben wird, kann anstelle der jeweiligen Variablen oder des Textes auch ein **Platzhalter** eingesetzt werden. Dies ist sinnvoll, wenn das Visualisierungsobjekt nicht unmittelbar im Programm verwendet werden soll, sondern dazu erstellt wird, um in anderen Visualisierungsobjekten als Referenz eingefügt zu werden. Beim Konfigurieren einer solchen **Referenz** können die Platzhalter dann durch Variablennamen oder Texte ersetzt werden.

Sehen Sie dazu das Kapitel zum Platzhalterkonzept [▶ 259].

9.1.4.1 Visualisierungselemente konfigurieren



'Extras' 'Konfigurieren'

Mit diesem Befehl öffnet sich der Dialog 'Element konfigurieren' zum Konfigurieren des selektierten Visualisierungselements. Den Dialog erhalten Sie auch durch Doppelklick auf das Element. Wählen Sie im linken Bereich des Dialogs eine Kategorie aus und füllen Sie im rechten Bereich die gewünschten Angaben aus. Dies geschieht entweder durch Aktivieren bestimmter Optionen oder durch Eintragen einer gültigen Variablen, deren Wert die Eigenschaft bestimmt. Für die verschiedenen Elementtypen sind unterschiedliche Konfigurationskategorien verfügbar.

Auch für eine Gruppe von Elementen stehen Konfigurationsdialoge zur Verfügung. Die Eigenschaften beziehen sich dann jedoch nur auf das "Element" Gruppe. Um die Einzelelemente konfigurieren zu können, muss die Gruppe aufgelöst werden.

Bei Eigenschaften, die sowohl über eine feste Einstellung, wie auch durch eine Variable bestimmt werden, gilt später im Online Modus, dass der Variablenwert den der festen Einstellung überschreibt (Beispiel: "Alarmfarbe innen" kann in Kategorie 'Farben' statisch, in Kategorie 'Farbvariablen' zusätzlich dynamisch über eine Variable definiert werden). Wird die Einstellung zusätzlich noch über eine Strukturvariable angesteuert, wird deren Wert ebenfalls durch die im Konfigurationsdialog eingetragene Projektvariable überschrieben.

Zeigerinstrument, Balkenanzeige und Histogramm müssen vor der Konfiguration neu gruppiert werden.

An den Stellen in der Element-Konfiguration, an denen Variablen wirksam werden, sind folgende **Eingaben** möglich:

• Variablennamen, wobei die Eingabehilfe (<F2>) zur Verfügung steht

• Ausdrücke, die zusammengesetzt sind aus Komponentenzugriffen, Feldzugriffen mit konstantem Index, Variablen und direkten Adressen

• Operatoren und Konstanten, die zusammen mit den bisherigen Ausdrücken beliebig kombiniert werden können

• Platzhalter anstelle von Variablennamen oder Textstrings

Beispiele für zulässige Ausdrücke:

```
 x + y
100*PLC_PRG.a
TRUE
NOT PLC_PRG.b
9*sin(x + 100)+cos(y+100)
```

Nicht möglich sind Funktionsaufrufe. Unzulässige Ausdrücke führen beim Einloggen zu einer Fehlermeldung ("Fehlerhafter Watchausdruck ..").

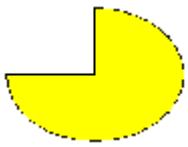
Beispiele für nicht zulässige Ausdrücke:

```
fun(88)
a := 9
RETURN.
```

Für **globale Variablen** sind in diesen Konfigurationsdialogen zwei Schreibweisen möglich: ".globvar" und "globvar" sind gleichbedeutend. Die Schreibweise mit Punkt (entspricht der, die im Watch- und Rezepturverwalter verwendet wird) ist allerdings nicht innerhalb eines zusammengesetzten Ausdrucks möglich.

Winkel

Im Dialog zum 'Kreissektor konfigurieren' können Sie in der Kategorie **Winkel** jeweils einen Wert oder eine Projektvariable eintragen, die **Anfangswinkel** und **Endwinkel** des Kreisssektor-Elements in Winkelgraden definieren. Wenn die Option **Nur Kreisbogen anzeigen** aktiviert ist, wird der nur der Kreisbogen dargestellt. Beispiel: Eingabe für Anfangswinkel: "90", Endwinkel: "180"



Dialog zum Konfigurieren von Kreissektoren

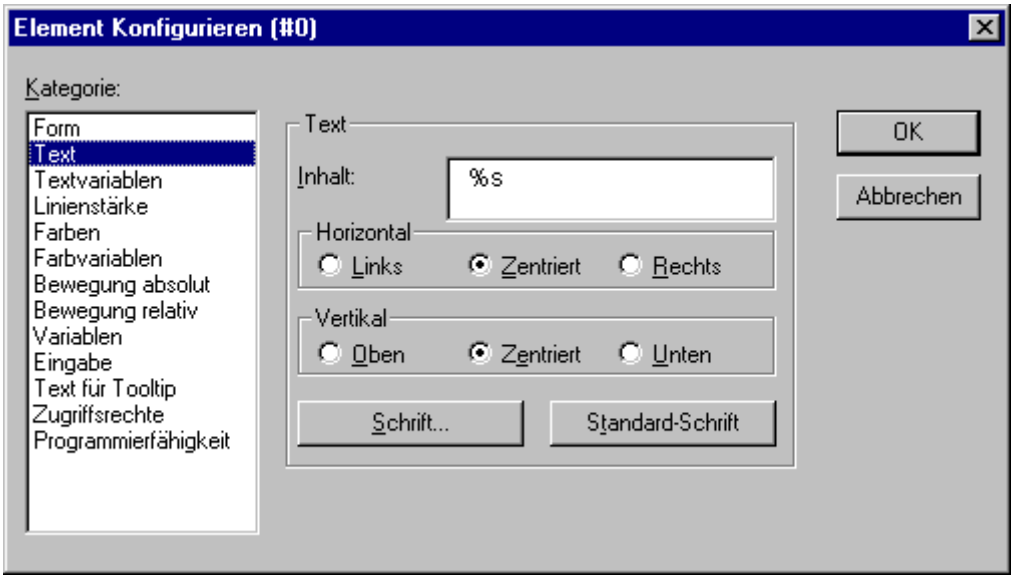
Form

Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie Form zwischen Rechteck, Abgerundetes Rechteck, Ellipse und Linie bzw. Polygon, Linienzug und Kurve wählen. Die Form wechselt in der bereits festgelegten Größe.

Dialog zum Konfigurieren von Visualisierungselementen

Text

Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Text)



Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Text** einen Text für das Element festlegen. Dieser wird direkt eingegeben oder/und es wird eine Variable festgelegt, die ihn liefert. Die Verwendung von Platzhaltern ist dabei möglich. Die Grundeinstellungen für Schrift und Ausrichtung werden ebenfalls definiert.

Wenn Textparameter, die im vorliegenden Dialog statisch definiert sind, zusätzlich dynamisch, d.h. über Variable (s.u. Kategorie 'Textvariablen' bzw. 'Programmierfähigkeit') geliefert werden, werden die statischen Definitionen überschrieben.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Geben Sie den Text direkt in das Feld **Inhalt** ein. Durch die Tastenkombination <Strg>+<Eingabetaste> können Sie Zeilenumbrüche einfügen, mit <Strg>+<Tabulator> Tabstops. Zusätzlich zu der reinen Eingabe von Texten haben Sie folgende **Formatierungsmöglichkeiten:**

• Wenn Sie im Text "%s" eingeben, wird im Online Modus an dieser Stelle der Wert der Variablen aus dem Feld 'Textausgabe' der Kategorie 'Variablen' als String dargestellt. Sie können auch eine Formatangabe verwenden, die der der Funktion sprintf aus der Standard-C-Bibliothek entspricht:

Mögliche Zeichenformatierungen:

Zeichen	Argument / Ausgabe als
d,i	Dezimale Zahl
o	Oktale Zahl ohne Vorzeichen (ohne führende Null)
x	Hexadezimale Zahl ohne Vorzeichen (ohne führendes 0x)
u	Dezimale Zahl ohne Vorzeichen
c	Einzelnes Zeichen
s	Zeichenkette
f	REAL-Werte [-]m.<dddddd>, wobei die Genauigkeit die Anzahl der d festlegt (Voreinstellung ist 6). Das Plus- bzw. Minuszeichen definiert Rechts (Voreinstellung) bzw. Linksbündigkeit, m gibt die Anzahl der gesamten Stellen an (Vorkommastellen, Komma, Nachkommastellen), d definiert die Anzahl der Stellen nach dem Komma.

Der Variablenwert wird im Online Modus entsprechend dargestellt werden. Als Eingabe sind alle IEC-konformen Formatierungen erlaubt, die zum jeweiligen Typ der Variable passen.

Es findet keine Überprüfung statt, ob der in der Formatangabe angegebene Typ zu dem der in 'Textausgabe' eingetragenen Variablen passt.

Beispiel:
Eingabe im Feld Inhalt: Füllstand %2.5f cm
Eingabe im Feld Textausgabe z.B.: fvar1 (REAL Variable)
-> Ausgabe im Online Modus z.B.: Füllstand 32.48999 cm

• Wenn Sie im Text "**%t**", gefolgt von einer bestimmten Folge von speziellen Platzhaltern, eingeben, wird diese Stelle im Online Modus durch die Angabe der Systemzeit ersetzt: Die Platzhalter definieren das Format der Ausgabe; sehen Sie die untenstehende Tabelle.

Vor %t darf kein weiteres Zeichen eingegeben werden (im Gegensatz zu z.B. "%s", s.o.)

Platzhalter	Format
%a	Name des Wochentags, abgekürzt, z.B. "Wed"
%A	Name des Wochentags, volle Länge, z.B. "Wednesday"
%b	Monatsname, abgekürzt, z.B. "Feb"
%B	Monatsname, volle Länge, z.B. "February"
%c	Datum und Uhrzeit im Format <Monat>/<Tag>/<Jahr> <Stunden>:<Minuten>:<Sekunden>, z.B. "08/28/02 16:58:45"
%d	Monatstag als Zahl (01-31), z.B. "24"
%H	Stundenangabe, 24-Stundenformat (01-24), z.B. "16"
%I	Stundenangabe, 12-Stundenformat (01-12), z.B. "05" für 17 Uhr
%j	Tag des Jahres (001 – 366), z.B. "241"
%m	Monat (01 – 12), z.B. "3" für März
%M	Minuten (00 – 59), z.B. "13
%p	Aktueller Anzeiger AM (Stunden <12) bzw. PM (>12) für die Angabe im 12-Stundenformat, z.B. "AM", wenn es gerade 9 Uhr vormittags ist.
%S	Sekunden (00 – 59)
%U	Wochenangabe als Zahl, wobei Sonntag als erster Tag der Woche gerechnet wird) (00 – 53 für 53 mögliche Wochen eines Jahres)
%w	Wochentag als Zahl (0 – 6; Sonntag = 0)
%W	Wochenangabe als Zahl, wobei Montag als erster Tag der Woche gerechnet wird) (00 – 53 für 53 mögliche Wochen eines Jahres)
%x	Datum im Format <Monat>/<Tag>/<Jahr>, z.B. "08/28/02"
%X	Uhrzeit im Format <Stunden>:<Minuten>:<Sekunden>, z.B. "16:58:45"
%y	Jahresangabe ohne Jahrhunderte (00 – 99), z.B. "02"
%Y	Jahresangabe mit Jahrhunderten, z.B. "2002"
%z, %Z	Angabe der Zeitzone (keine Angabe, falls die Zeitzone nicht bekannt ist), z.B. "Westeuropäische Sommerzeit"
%%	Prozentzeichen

Beispiele:

%t%a %b %d.%m.%y %H:%M:%S
-> Ausgabe im Online Modus: Wed Aug 28.08.02 16:32:45

Zwischen den Platzhaltern kann auch ein Text eingegeben werden:

%tHeute ist der %d.%m.%y
-> Ausgabe im Online Modus: Heute ist der 28.08.02

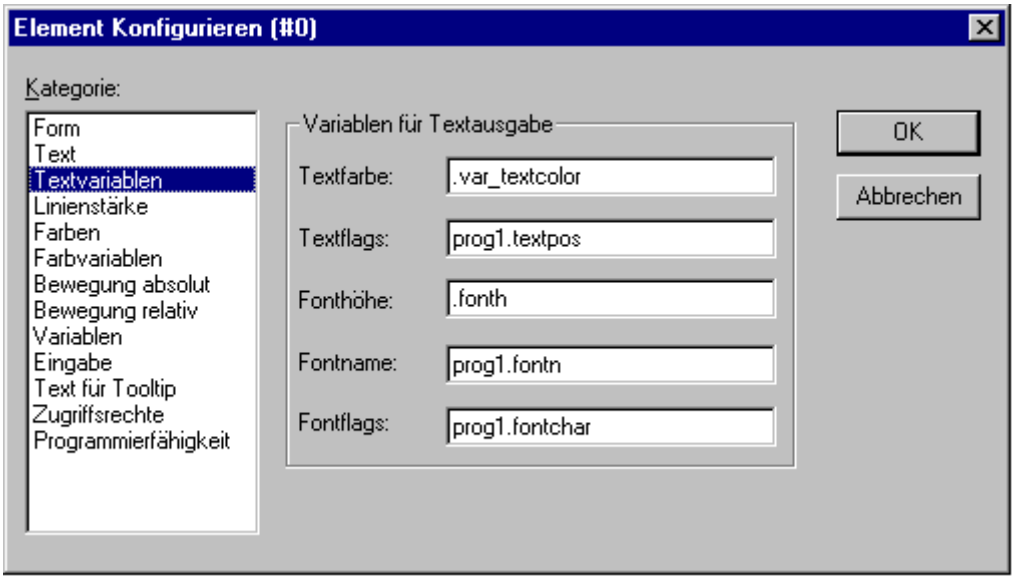
Soll ein Textstring in eine **Übersetzungsdatei** übernommen werden, die dann im Online Modus ein Umschalten in eine andere Landessprache ermöglicht, muss er am Anfang und am Ende mit # begrenzt werden. Beispiele: „#Pumpe 1#" oder aber auch „#Pumpe# 1" Der letztere Fall beispielsweise erspart beim mehrmaligen Vorkommen des Textes Pumpe (Pumpe 1, Pumpe 2 etc.) ein mehrmaliges Auftreten in der Übersetzung.

• Wenn Sie im Text "%**<PREFIX>**" eingeben, können Sie anstelle von "PREFIX" eine beliebige Buchstabenfolge eingeben, die für die Verwendung von dynamischen Texten als Kennzeichner dient. Das Prefix wird dazu in Kombination mit einer ID verwendet, die wiederum über den Eintrag in Kategorie 'Variablen' im Feld 'Textausgabe' festgelegt wird. Die Kombination verweist auf eine bestimmte Textversion, die in einer XML-Datei vorliegt, welche die möglichen dynamischen Texte definiert. Somit wird zur Laufzeit jeweils der zur aktuellen Kombination gehörige Text angezeigt.

Der konfigurierte Text wird online je nach angegebener Ausrichtung **Horizontal Links, Zentriert** oder **Rechts** und **Vertikal Oben, Zentriert** oder **Unten** im Element erscheinen.
Mit der Schaltfläche **Schrift** erscheint der Dialog zu Auswahl der Schriftart. Wählen Sie die gewünschte Schriftart und Bestätigen Sie den Dialog mit **OK**. Mit der Schaltfläche **Standard-Schrift** wird die Schriftart eingestellt, die in den Projektoptionen **('Projekt' 'Optionen' 'Editor')** gewählt ist. Wird sie dort verändert, wird in allen Elementen diese Schriftart angezeigt, außer bei Elementen, bei denen explizit eine andere über die Schaltfläche **Schrift** gewählt wurden.

Textvariablen

Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Textvariablen)



Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Textvariablen** für die in der Kategorie 'Text' eingegebene Zeichenfolge definieren, über welche Projektvariable Farbe und Font-Eigenschaften dynamisch bestimmt werden sollen. Geben Sie dazu den Variablennamen am besten über die Eingabehilfe <F2> ein. Die hier möglichen Konfigurationen sind auch über Komponenten der Struktur VisualObjectType möglich. Sehen Sie dazu die Beschreibung zur Kategorie "Programmierfähigkeit" eines Visualisierungselements; dort finden Sie die gültigen Werte und jeweilige Auswirkung der einzelnen Strukturkomponenten.

Wenn in Kategorie 'Text' entsprechende feste Definitionen zu Textparametern vorliegen, werden diese durch die laufenden Variablenwerte überschrieben.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Die Parameter des Dialogs:

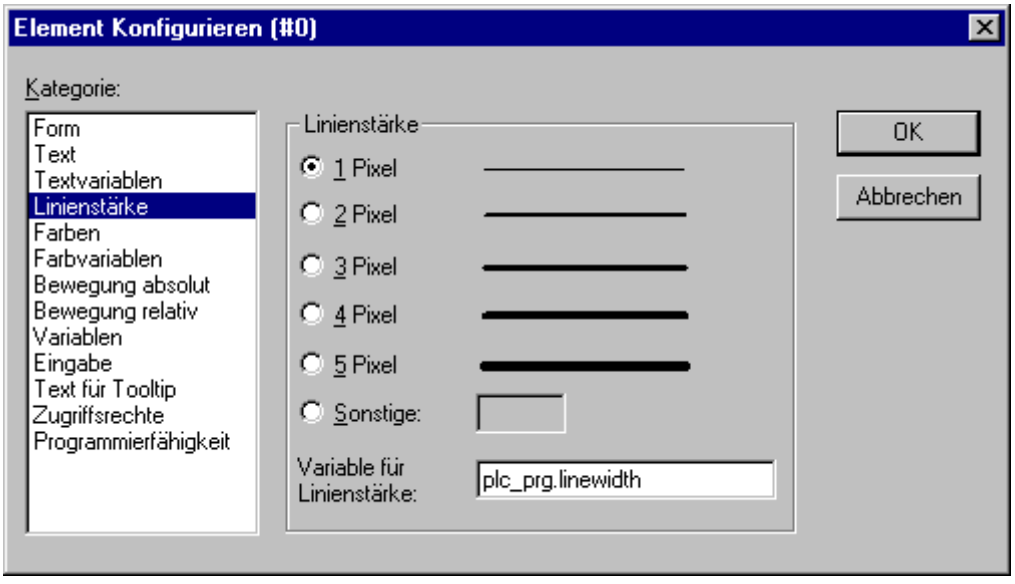
Parameter	Bedeutung:	Beispiel eines Eintrags einer Projektvariable:	Beispiel-Verwendung der Variablen im Programm:	entspricht Komponente von Struktur VisualObjectType:
Textfarbe:	Textfarbe	"plc_prg.var_textcolor"	var_textcolor=16#FF00FF -> Farbe	dwTextColor
Textflags:	Textposition (rechts, links, zentriert...)	"plc_prg.textpos"	textpos:=2 -> Text rechtsbündig platziert	dwTextFlags
Fonthöhe:	Fonthöhe in Pixel	".fonth"	fonth:=16; -> Fonthöhe 16 pt	ntFontHeight
Fontname:	Font-Bezeichnung	"vis1.fontn"	fontn:=arial; -> Arial wird verwendet	stFontName
Fontflags:	Fontdarstellung (Fett, Unterstrichen, Italic...)	"plc_prg.fontchar"	fontchar:=2 -> Text wird fett dargestellt	dwFontFlags

Linienstärke

Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Linienstärke** für ein Element auswählen. Als Optionen sind Stärken von 1 bis 5 Pixel vorgegeben, zusätzlich kann manuell ein anderer Wert **(Sonstige:)**, oder eine Projektvariable **(Variable für Linienstärke:)** eingegeben werden, letzteres auch über die Eingabehilfe (<F2>).

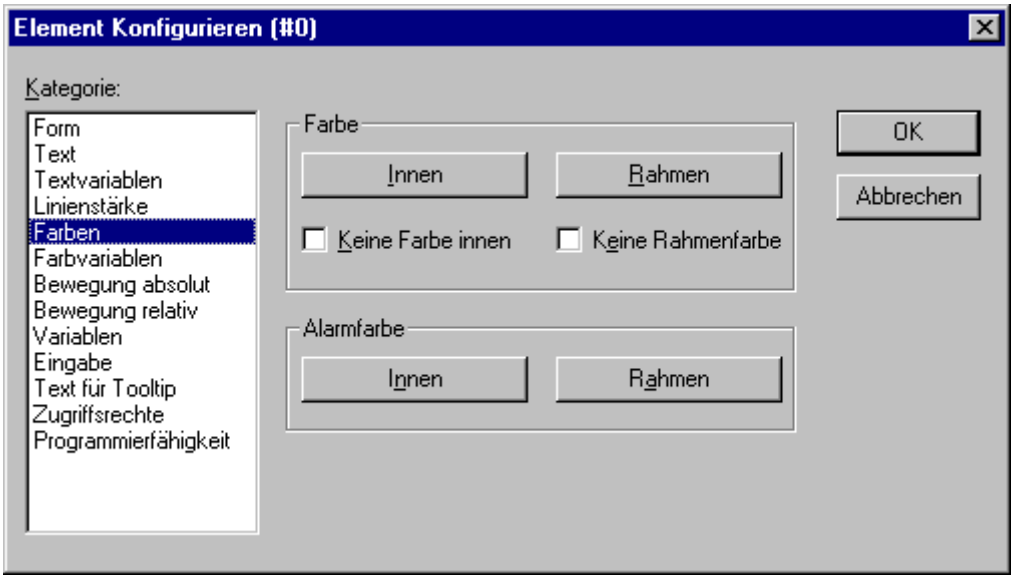
Wenn der Parameter zusätzlich über eine Strukturvariable (s.u. Kategorie 'Programmierfähigkeit') definiert ist, wird im Online Betrieb zunächst die ev. unter 'Variable für Linienstärke' angegebenen Projektvariable ausgewertet !

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.



Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Linienstärke)

Farben



Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Farben)

Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Farben** Grundfarben und Alarmfarben für die Innenfläche und den Rahmen Ihres Elements auswählen. Die Optionen **Keine Farbe innen** und **Keine Rahmenfarbe** ermöglichen die Erstellung transparenter Elemente.

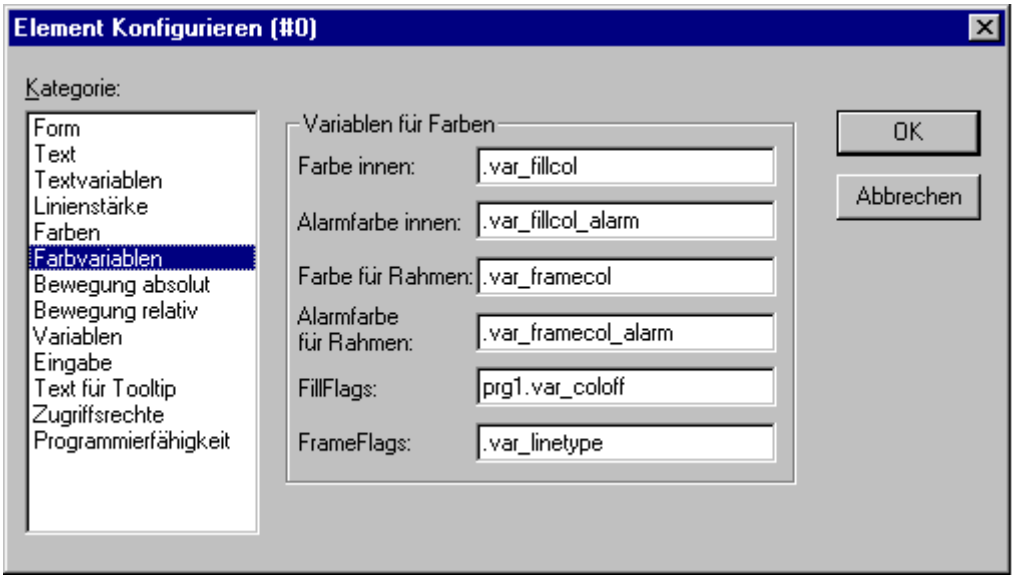
Wenn Farbparameter auch dynamisch, d.h. über eine Systemvariable bzw. Strukturvariable (s.u. Kategorie 'Farbvariablen' bzw. 'Programmierfähigkeit') geliefert werden, werden die hier vorgenommenen statischen Definitionen überschrieben.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Wenn Sie in der Kategorie Variablen im Feld **Farbwechsel** nun eine boolsche Variable eingeben, so wird das Element in der eingestellten **Farbe** dargestellt, solange die Variable FALSE ist. Ist die Variable TRUE, so wird das Element in seiner **Alarmfarbe** dargestellt.

Die Farbwechsel-Funktion ist erst aktiv, wenn sich die Steuerung im Online Modus befindet! Wenn Sie die Farbe für den Rand ändern wollen, dann drücken Sie die Schaltfläche Rahmen, ansonsten **Innen**. In jedem Fall öffnet der Dialog zur Auswahl der Farbe.
Hier können Sie aus den Grundfarben und den selbst definierten Farben den gewünschten Farbton auswählen. Durch Drücken der Schaltfläche **Farben definieren** können Sie die selbst definierten Farben verändern.

Farbvariablen



Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Farbvariablen)

Hier können Sie Projektvariablen eintragen (z.B. PLC_PRG.color_inside), deren Wert dann im Online Modus die jeweilige Eigenschaft bestimmen: Die hier möglichen Konfigurationen sind auch über Komponenten der Struktur VisualObjectType möglich. Sehen Sie deshalb die Beschreibung zur Kategorie "Programmierfähigkeit" eines Visualisierungselements; dort finden Sie die gültigen Werte und jeweilige Auswirkung der einzelnen Parameter.

Wenn in Kategorie 'Farbe' entsprechende feste Definitionen zu Farbparametern vorliegen, werden diese im Online Betrieb durch die Werte der hier definierten Variablen überschrieben; ebenso die eventuell zusätzlich über eine Strukturvariable gelieferten Werte.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Die Parameter des Dialogs:

Parameter	Bedeutung:	Beispiel eines Eintrags einer Projektvariable:	Beispiel-Verwendung der Variablen im Programm:	entspicht Komponente von Struktur VisualObjectType:
Farbe innen:	Füllfarbe	"plc_prg.var_fillcol"	var_fillcol:= 16#FF00FF -> Füllfarbe Pink	dwFillColor
Alarmfarbe innen:	Füllfarbe im Alarmfall	"plc_prg.var_fillcol_a"	var_fillcol_a:= 16#FF00FF -> Alarmfarbe Pink	dwFillColorAlarm
Farbe Rahmen:	Rahmenfarbe	"plc_prg.var_framecol"	var_framecol:= 16#FF00FF -> Rahmenfarbe Pink	dwFrameColor
Alarmfarbe Rahmen:	Rahmenfarbe im Alarmfall	"plc_prg.var_framecol"	var_framecol:= 16#FF00FF -> Alarmfarbe Pink	dwFrameColorAlarm
FillFlags:	Die vorgenommene Farbkonfiguration für 'Innen' kann aktiviert (FALSE) bzw. deaktiviert (TRUE) werden.	"plc_prg.var_col_off"	var_col_off:=1 -> die Farbzuweisungen für die Füllung des Elements werden nicht ausgeführt, der Rahmen bleibt	dwFillFlags
FrameFlags:	Darstellung des Rahmens (voll, gestrichelt, ...)	"plc_prg.var_linetype"	var_linetype:=2; -> Rahmen wird gestrichelt dargestellt	dwFrameFlags

Bewegung absolut

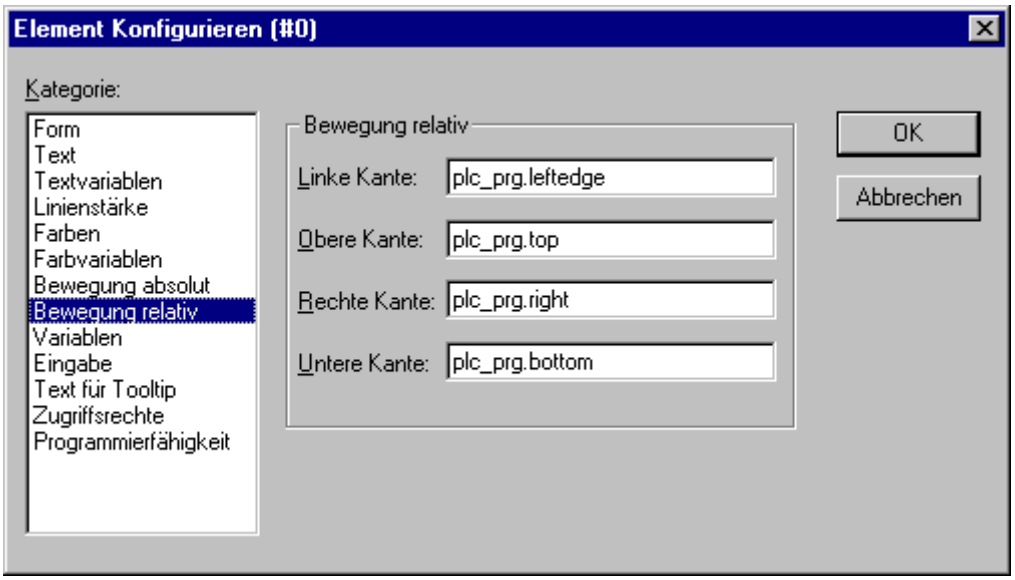
Im Dialog zum Konfigurieren von Visualisierungselementen können in der Kategorie **Bewegungabsolut** in den Feldern **X-** bzw. **Y-Offset** Variablen eingetragen werden, die das Element in X- bzw. Y-Richtung in Abhängigkeit des jeweiligen Variablenwertes verschieben. Eine Variable im Feld **Skalierung** verändert die Größe des Elements linear zum Variablenwert. Der aktuelle Variablenwert, der als Skalierungsfaktor dient, wird implizit durch 1000 dividiert, damit nicht notwendigerweise Real-Variablen verwendet werden müssen, um eine Verkleinerung des Elements zu erreichen. Die Größenänderung des Elements geht immer vom Element-Drehpunkt aus. Eine Variable im Feld **Winkel** bewirkt eine Drehung des Elements um seinen Drehpunkt in Abhängigkeit des Variablenwertes (positiver Wert = mathematisch positiv = Uhrzeigersinn). Der Wert wird in Grad ausgewertet. Bei Polygonen rotiert jeder Punkt, daß heißt das Polygon dreht sich. Bei allen anderen Elementen rotiert das Objekt, wobei immer die obere Kante oben bleibt.
Der Drehpunkt erscheint nach einmaligem Anklicken des Elements und wird als kleiner schwarzer Kreis mit einem weißen Kreuz dargestellt (⊕). Mit gedrückter linker Maustaste können Sie den Drehpunkt verschieben.

Wenn der Parameter zusätzlich über eine Strukturvariable (s.u. Kategorie 'Programmierfähigkeit') geliefert wird, wird im Online Betrieb zunächst die hier angegebene Variable ausgewertet.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Bewegung absolut)

Bewegung relativ



Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Bewegung relativ)

Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Bewegung** relativ den einzelnen Element**kanten** Variablen zuordnen. In Abhängigkeit der Variablenwerte bewegen sich dann die Kanten. Die einfachste Möglichkeit Variablen in die Felder einzugeben ist die Eingabehilfe (<F2>).
Die vier Einträge geben die vier Seiten ihres Elements an. Die Grundposition der Kanten ist stets auf Null, ein neuer Wert der Variablen in der entsprechenden Spalte, verschiebt die Grenze um diesen Wert in Pixel. Die eingegebenen Variablen sollten also vom Typ INT sein.

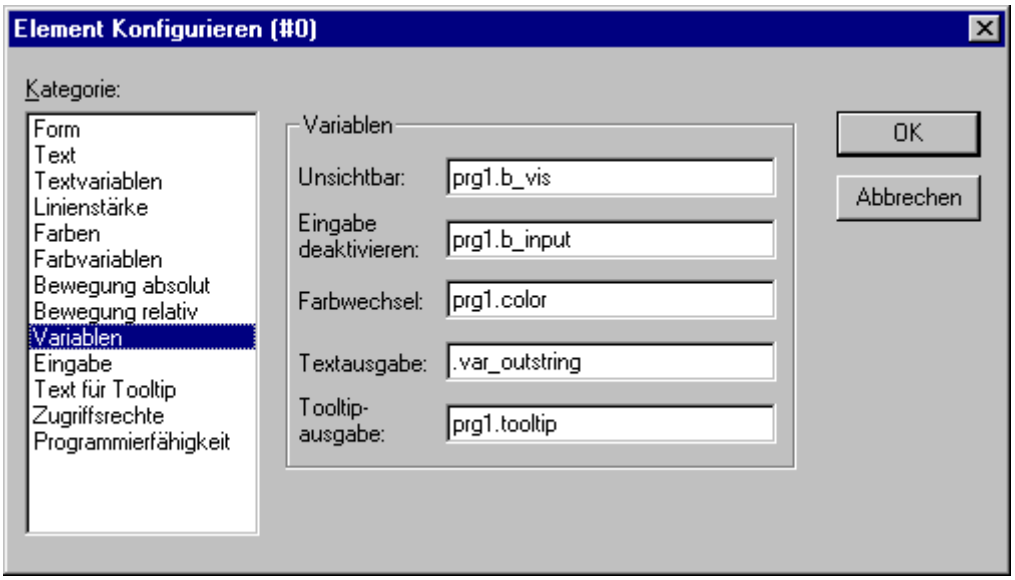
Positive Werte verschieben die horizontalen Kanten nach unten bzw. die vertikalen Kanten nach rechts!

Wenn der Parameter zusätzlich. über eine Strukturvariable (s.u. Kategorie 'Programmierfähigkeit') geliefert wird, wird im Online Betrieb zunächst die hier angegebene Variable ausgewertet.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Variablen

Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Variablen)



Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Variablen** (wie in den Kategorien 'Textvariablen' und 'Farbvariablen') Variablen angeben, die den Zustand des Visualisierungselements beschreiben sollen. Benützen Sie dazu am besten die Eingabehilfe (<F2>).

Wenn die Parameter zusätzlich über eine Strukturvariable (siehe Kategorie 'Programmierfähigkeit') geliefert werden, werden im Online Modus zuerst die hier eingetragenen Projektvariablen ausgewertet.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Die Konfigurationsmöglichkeiten:

Unsichtbar: Wenn die hier eingetragene boolsche Variable den Wert FALSE hat, dann ist das Visualisierungselement sichtbar. Besitzt die Variable den Wert TRUE, dann ist das Element unsichtbar.

Eingabe deaktivieren: Wenn die hier eingetragene boolsche Variable den Wert TRUE hat, dann werden alle Einstellungen der Kategorie 'Eingabe' nicht berücksichtigt.

Farbwechsel: Wenn die hier eingetragene boolsche Variable den Wert FALSE hat, dann wird das Visualisierungselement in seiner Grundfarbe dargestellt. Ist die Variable TRUE, dann wird das Element in seiner Alarmfarbe dargestellt.

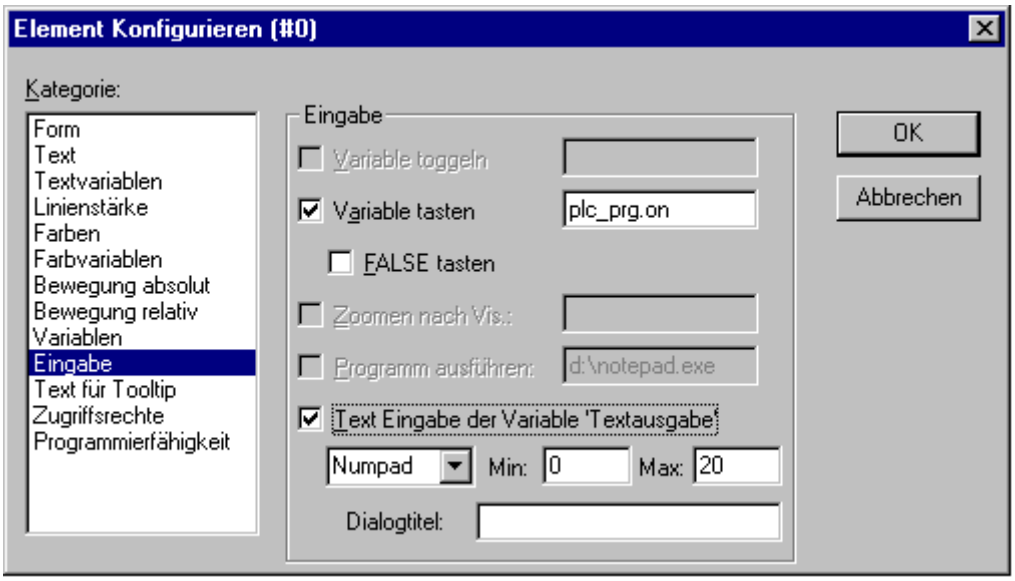
Textausgabe:
• Wenn Sie im Feld **Inhalt** der Kategorie Text zusätzlich zum Text oder ausschließlich "%s" eingegeben haben, wird der Wert der hier bei Textausgabe eingetragenen Variable im Online Modus in der Visualisierung dargestellt. **Das "%s"** wird dann durch den Wert ersetzt.

• Wenn Sie im Feld Inhalt der Kategorie Text zusätzlich zum Text **"%<PREFIX>"** eingegeben haben, wobei "PREFIX" eine bestimmte Zeichenfolge ist (siehe), dann wird die hier bei Textausgabe eingetragene Variable bzw. der hier eingetragene numerische Wert als ID interpretiert, die in Kombination mit dem Präfix als Referenz auf einen Text dient, der in einer XML-Datei beschrieben ist. Dieser Text wird dann im Online Modus anstelle von "%<PREFIX>" angezeigt. Somit ist eine dynamische Veränderung des anzuzeigenden Textes möglich. Sehen Sie hierzu auch die Beschreibung zum Dialog **'Extras' Einstellungen', Kategorie Sprache,** sowie allgemein zur Sprachumschaltung in der Visualisierung.

• Wenn Sie wollen, dass der Wert der Variablen im Online Modus per Tastatur editiert werden kann, aktivieren Sie die Option 'Texteingabe der Variable 'Textausgabe'' in der **Kategorie Eingabe**.

Tooltipausgabe: Hier können Sie eine Variable vom Typ STRING angeben, deren aktueller Wert als Tooltip des Elements in der Visualisierung erscheint.

Eingabe



Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Eingabe)

Variable toggeln: Wenn diese Option aktiviert ist, toggeln Sie im Online Modus mit jedem Mausklick auf das Element den Wert der Variablen, die in dem Eingabefeld dahinter angegeben ist. Zur Eingabe können Sie über <F2> die Eingabehilfe aufrufen. Der Wert der boolschen Variable ändert sich beim ersten Mausklick auf den gewählten Wert TRUE oder FALSE und beim Loslassen zurück auf den jeweils entsprechenden Gegenwert FALSE oder TRUE.

Variable tasten: Wenn diese Option aktiviert ist, können Sie im Online Modus den Wert der boolschen Variablen, die in dem Eingabefeld dahinter angegeben ist, zwischen TRUE und FALSE wechseln lassen. Platzieren Sie den Mauszeiger auf dem Element, drücken Sie die Maustaste und halten Sie sie gedrückt. Wenn die Option **FALSE tasten** aktiviert ist, wird der Wert beim Drücken auf FALSE gesetzt, ansonsten auf TRUE. Sobald Sie die Taste wieder loslassen, springt der Variablenwert auf den Ausgangswert zurück.

Zoomen nach Vis...: Wenn diese Option aktiviert ist, können Sie im nachfolgenden Feld angeben, zu welcher Visualisierung im Online Modus gewechselt werden soll, sobald mit der Maus auf das Element geklickt wird. Dabei wird dann erst das Fenster der Ziel-Visualisierung geöffnet und danach das der aktuellen geschlossen.

Sie haben folgende Eingabemöglichkeiten:
• den Namen eines Visualisierungsobjektes aus dem aktuellen Projekt (siehe Object Organizer)
• Soll zu einer Visualisierungsreferenz gesprungen werden, die Platzhalter enthält, können diese direkt beim Aufruf durch Variablennamen bzw. Texte ersetzt werden. Befolgen Sie hierzu die folgende Syntax: <Visuname>(<Platzhalter1>:=<Text1>, <Platzhalter2>:=<Text2>,..., <Platzhalter n>:=<Textn>) Bei dieser Schreibweise werden die ‚$'-Zeichen weggelassen. Beim Übersetzen der Visualisierung wird überprüft, ob der eingegebene Text der festgelegten Wertemenge entspricht, wenn nicht wird eine entsprechende Warnung ausgegeben.
Beispiel:
Aufruf der Visualisierung visu1, wobei die in visu1 verwendeten Platzhalter var_ref1 und var_ref2 durch die Variablen PLC_PRG.var1 bzw. PROG.var1 ersetzt werden: visu1(var_ref1:=PLC_PRG.var1, var_ref2:=PROG.var1)
• eine Programmvariable vom Typ STRING (z.B. PLC_PRG.xxx), über die der Name des Visualisierungsobjekts (z.B. ‚visu1'), zu dem bei Mausklick gewechselt werden soll, vorgegeben werden kann (z.B. xxx := ‚visu1').
• den Befehl ‚ZOOMTOCALLER': In diesem Fall wird im Online Modus per Mausklick auf das Element ein Zurückspringen in die aufrufende Visualisierung erreicht, falls eine solche Konstellation konfiguriert wurde.

Die Verwendung einer Programmvariable ist für die PLC HMI CE nicht zulässig.

Die implizite Variable **CurrentVisu** (Typ STRING, sehen Sie auch Anhang A zu impliziten (System-) variablen beschreibt den Namen des aktuell geöffneten Visualisierungsobjekts. Sie kann beispielsweise in einer Applikation verwendet werden, um zu steuern, welche Visualisierung geöffnet werden soll, bzw. um festzustellen, welche augenblicklich geöffnet ist.

Dies nur möglich, wenn die Namen der Visualisierungsobjekte in Großbuchstaben definiert sind. (siehe "Anlegen eines Visualisierungsobjekts" [▶ 199]). Beispiel: CurrentVisu:='TC_VISU';

Programm ausführen: Wenn diese Option aktiviert ist, können Sie im Eingabefeld dahinter ein oder mehrere ausführbare Programme, ASSIGN- oder spezielle "INTERN-Befehle" eingeben, die dann bei Mausklick auf das Element im Online Modus ausgeführt werden. Dazu erhalten Sie über die Schaltfläche "..." einen Dialog in dem Befehle ausgewählt (Hinzufügen) und in der gewünschten Reihenfolge (Davor, Danach) angeordnet werden können. Beispielsweise gibt es einen Befehl zur Sprachumschaltung in einer Visualisierung. Sehen Sie zur Erklärung der möglichen Befehle: 'Spezielle Eingabemöglichkeiten für "Bedienversionen".

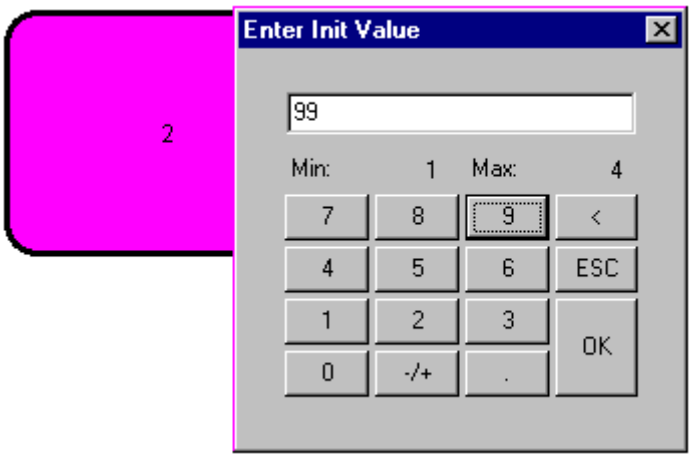
Beispiel: notepad C:/help.txt (das Programm notepad wird gestartet und die Datei help.txt geöffnet)

Text Eingabe der Variable 'Textausgabe': Wenn diese Option aktiviert ist, erhalten Sie im Online Modus in diesem Visualisierungselement die Möglichkeit, einen Wert einzugeben, der nach Drücken der <Eingabetaste> in die Variable geschrieben wird, die im Feld **Textausgabe** der Kategorie Variablen steht. Wählen Sie aus der Auswahlliste, in welcher Weise diese Eingabe dann im Online-Betrieb vorgenommen werden kann:

Text: Ein Editierrahmen wird geöffnet, in den der Wert eingetippt wird.

Numpad bzw. Keypad: Ein Fenster mit der Nachbildung des numerischen bzw. des alphabetischen Tastaturfeldes wird geöffnet, auf dem durch Aktivieren der entsprechenden Tastenelemente ein Wert eingegeben werden kann. Dies ist beispielsweise sinnvoll für Visualisierungen, die über Touch-Screen-Bildschirmen bedient werden sollen.

Numpad zur Eingabe eines Wertes im Online Modus:



Der Wertebereich für die Eingabe von Werten für nicht-STRING Variablen kann durch Angabe von minimal und maximal möglichem Wert in den Feldern **Min:** und **Max:** eingeschränkt werden.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Tooltip

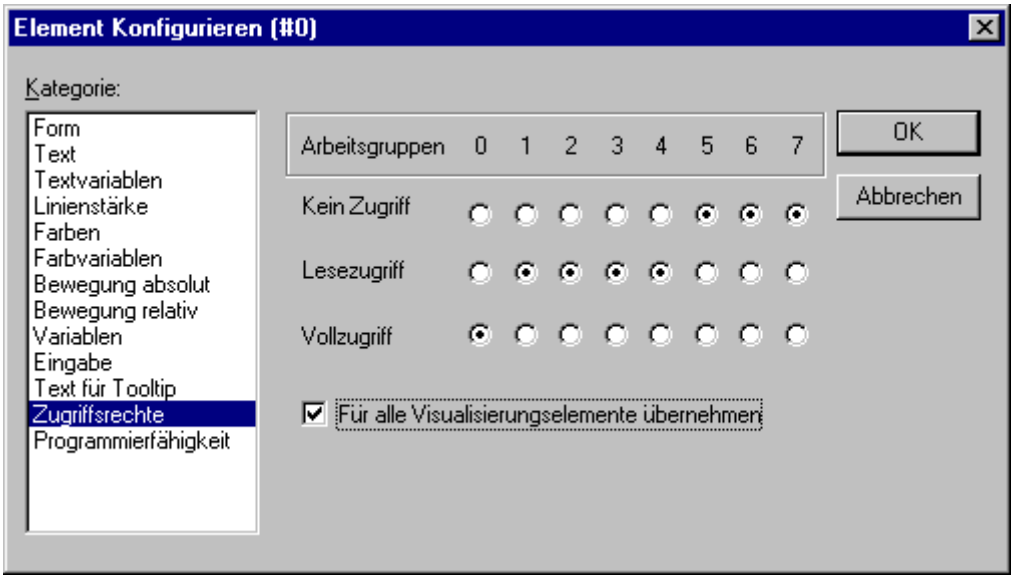
Im Dialog **Text für Tooltip** erhalten Sie ein Eingabefeld für Text, der in einem Textfenster erscheint, sobald der Mauszeiger im Online Modus über das Element geführt wird. Der Text kann mittels der Tastenkombination <Strg>+<Eingabetaste> mit Zeilenumbrüchen versehen werden.

Beachten Sie generell die Auswertungsreihenfolge im Online Modus bei mehrfacher Definition einer Elementeigenschaft.

Zugriffsrechte

Die Bedienmöglichkeiten und die Ansicht einer Visualisierung im Online-Modus können für verschiedene Benutzergruppen unterschiedlich gestaltet werden, indem ihnen pro Element unterschiedliche Zugriffsrechte erteilt werden. Es stehen die in der Benutzerverwaltung des Projekts vorgesehenen acht Arbeitsgruppen zur Verfügung (siehe auch 'Projekt' 'Objekt' 'Eigenschaften' bzw. 'Projekt' 'Passwörter für Arbeitsgruppen'). Die Rechtevergabe erfolgt über Aktivieren des entsprechenden Zugriffsrechts im folgenden Dialog in der Kategorie Zugriffsrechte der Konfiguration eines Visualisierungselements:

Dialog zum Konfigurieren der Zugriffsrechte für ein Visualisierungselement (Kategorie Zugriffsrechte)



Die Zugriffsrechte für Visualisierungselemente haben folgende Bedeutung:

Kein Zugriff Element ist nicht sichtbar
Lesezugriff Element ist sichtbar aber nicht bedienbar (keine Eingaben möglich)
Vollzugriff Element ist sichtbar und bedienbar

Die für ein Visualisierungselement gesetzten Zugriffsrechte werden für alle weiteren Elemente aller im Projekt enthaltenen Visualisierungen sofort übernommen, wenn die Option **Für alle Visualisierungselemente übernehmen** aktiviert wird.

Beachten Sie, dass die für einen kompletten Visualisierungsbaustein (Visualisierungsobjekt) über 'Projekt''Objekt' 'Eigenschaften' gesetzten Zugriffsrechte von denen der einzelnen Visualisierungs-elemente unabhängig sind !

Programmierfähigkeit einer Visualisierung

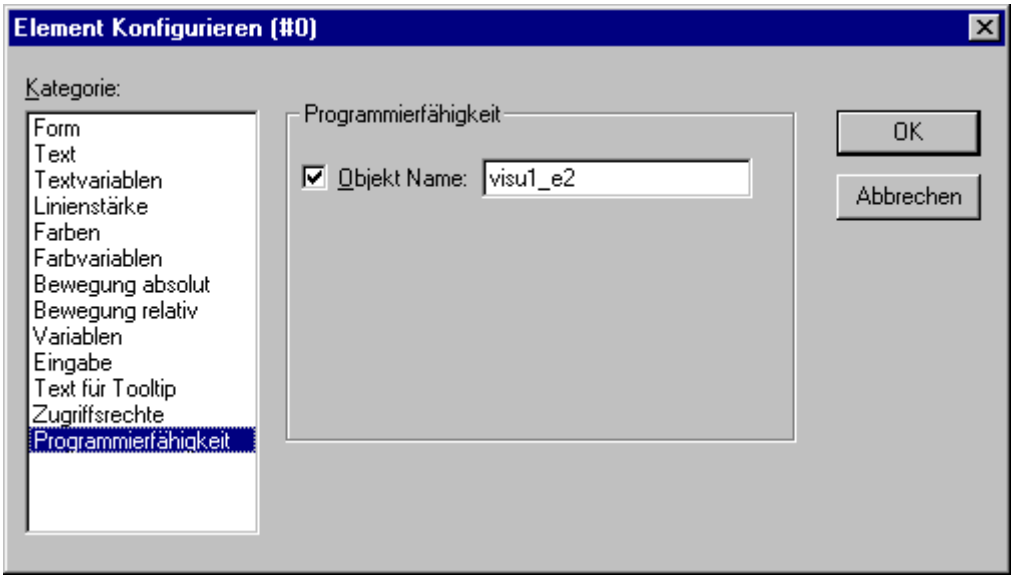
Neben der Möglichkeit, feste Einstellungen für das Aussehen eines Elements in den Konfigurationsdialogen vorzunehmen kann auch eine dynamische Steuerung über Variablen erfolgen. Dazu können die Visualisierungselemente mit normalen Projektvariablen verknüpft werden, es kann dem Element aber auch gezielt eine **Strukturvariable** zugewiesen werden, die ausschließlich zur Programmierung der Eigenschaften verwendet wird:

Beachten Sie die Auswertungsreihenfolge, die später im Online Modus gilt:
- Die dynamisch, also über normale Projektvariablen oder über die Strukturvariablen, gelieferten Werte überschreiben die festen Einstellungen der Elementkonfigurationen.
- Wenn eine Elementeigenschaft sowohl durch eine direkt im Konfigurationsdialog eingetragene Projektvariable als auch über die Komponente einer Strukturvariable angesprochen wird, wird im Online Modus zuerst der Wert der Projektvariablen ausgewertet.

Zur Konfiguration der Elementeigenschaften über eine Strukturvariable gehen Sie folgendermaßen vor:

Tragen Sie im Konfigurationsdialog der Kategorie Programmierfähigkeit im Feld **Objekt Name:** einen neuen, im Projekt **eindeutigen** (!) Variablennamen ein. Dazu müssen Sie die Option durch einen Mausklick in das Kästchen aktivieren. Die Variable erhält automatisch den Typ der Struktur **VisualObjectType**, die Bestandteil einer Bibliothek ist. Die Deklaration erfolgt implizit und ist im Projekt nicht unmittelbar sichtbar.

Nach dem nächsten Übersetzungslauf kann die dem Visualisierungselement zugewiesene Strukturvariable im Projekt verwendet werden. (Tipp hierzu: Aktivieren Sie die Intellisense-Funktion 'Komponenten auflisten' in den Projektoptionen, Kategorie Editor, um nach Eingabe des Variablennamens, gefolgt von einem Punkt, die Strukturelemente in einer Auswahlliste angeboten zu bekommen). Wenn Sie beispielsweise einen Objekt Namen "visu1_line" für ein Visualisierungselement eingegeben haben, können Sie im Programm z.B. mit visu1_line.nLIneWidth:=4 die Liniendicke für dieses Element festlegen.



Dialog zum Konfigurieren der Programmierfähigkeit eines Visualisierungselementes (Kategorie Programmierfähigkeit)

Die Struktur VisualObjectType:

Die folgende Tabelle zeigt alle in Struktur VisualObjectType verfügbaren Elemente und die entsprechende Konfigurationsmöglichkeit in den Dialogen der verschiedenen Kategorien:

Am Beginn des Komponentennamens ist der Datentyp integriert. Dabei bedeutet:
n INT
dw DWORD
b BOOL
st STRING

Komponente (+Datentyp)	Bedeutung	Beispiel (für das Element wurde der Object Name "vis1" definiert.)	entsprechende Einstell-möglichkeit im Konfigu-rationsdialog:
nXOffset:INT;	Verschieben des Elements in X-Richtung	vis1.nXOffset:=val2; (Element wird auf Position X=val2 gesetzt)	- Kat. Bewegung absolut: X-Offset
nYOffset:INT;	Verschieben des Elements in Y-Richtung	vis1.nYOffset:=22; (Element wird auf Position Y=val2 gesetzt)	- Kat. Bewegung absolut: Y-Offset
nScale:INT;	Größenänderung	vis1.nScale:=plc_prg.scale_var; (Größe des Elements ändert sich linear mit der Veränderung von Variable plc_prg.scale_var)	- Kat. Bewegung absolut: Skalierung
nAngle:INT;	Rotieren des Elements um seinen Drehpunkt	vis1.anglevar:=15; (Element rotiert um 15 Grad im Uhrzeigersinn)	- Kat. Bewegung absolut: Winkel
Komponente (+Datentyp)	Bedeutung	Beispiel (für das Element wurde der Object Name "vis1" definiert.)	entsprechende Einstellmöglichkeit im Konfigurationsdialog:
nLeft:INT;	Verschieben der linken Elementkante in X-Richtg.	vis1.nLeft:=val2; (Elementkante rückt auf Position X=val2)	- Kat. Bewegung absolut: X-Offset
nTop:INT;	Verschieben der oberen Elementkante in Y-Richtg. (pos.® n.unten)	vis1.nTop:=val2; (Elementkante rückt auf Position Y=val2)	- Kat. Bewegung absolut: X-Offset
nRight:INT;	Verschieben der rechten Elementkante in X-Richtg.	vis1.nRight:=val2; (Elementkante rückt auf Position X=val2)	- Kat. Bewegung absolut: X-Offset
nBottom:INT;	Verschieben der unteren Elementkante in Y-Richtg. (pos.®n.unten)	vis1.nBottom:=val2; (Element rückt auf Position X=val2)	- Kat. Bewegung absolut: X-Offset
bInvisible:BOOL;	bewirkt Wechsel zwischen Sichtbar und Unsichtbar durch Wechsel von TRUE und FALSE	vis1.visible:=TRUE; (Element ist unsichtbar)	- Kat. Farben: Keine Farbe innen + Keine Rahmenfarbe - Kat. Farbvariablen: FillFlags + FrameFlags
stTextDisplay: STRING;	Text, der im Element erscheint	vis1.TextDisplay:='ON / OFF'; Element wird mit diesem Text beschriftet	- Kat. Text: Eintrag bei 'Inhalt'
bToggleColor: BOOL;	bewirkt Farbwechsel durch Wechsel von TRUE und FALSE	vis1.bToggleColor:=alarm_var; (Wenn Alarmanzeiger alarm_var TRUE wird, wechselt das Element auf die Farbe, die es über die Komponenten dwFillColorAlarm, dwFrameColorAlarm bzw. durch die Einstellungen im Konf.dialog Kat. Farben erhält.	- Kat. Eingabe: Variable toggeln - Kat. Variablen: Farbwechsel

Komponente (+Datentyp)	Bedeutung	Beispiel (für das Element wurde der Object Name "vis1" definiert.)	entsprechende Einstell-möglichkeit im Konfigu-rationsdialog:
bInputDisabled: BOOL;	Eingaben aus Kategorie Eingabe werden bei TRUE beachtet, bei FALSE nicht	vis1.bInputDisabled:=FALSE; (keinerlei Eingabemöglichkeit auf dieses Element)	- Kat. Variablen: Eingabe deaktivieren
stTooltipDisplay:STRING;	Text des Tooltips	vis1.stTooltipDisplay:= 'Schalter für .....';	- Kat. Text für Tooltip: Eintrag bei 'Inhalt'
Komponente (+Datentyp)	Bedeutung	Beispiel (für das Element wurde der Object Name "vis1" definiert.)	entsprechende Einstellmöglichkeit im Konfigurationsdialog:
dwTextFlags: DWORD;	Position des Textes: 1 linksbündig 2 rechtsbündig 4 horizontal zentriert 8 oben 16 unten 32 vertikal zentriert **Achtung**: Es sollte immer sowohl die horizontale als auch die vertikale Positionierung gesetzt sein (Addition der Werte) !	vis1.dwTextFlags:=36; (Text wird in der Mitte des Elementes platziert (4 + 32)	- Kat. Text: Optionen von Horizontal und Vertikal - Kat. Textvariablen: Textflags
dwTextColor : DWORD;	Textfarbe (zur Eingabe der Farbwerte siehe im Anschluss an die Tabelle)	vis1.dwTextColor := 16#00FF0000; (Text wird blau dargestellt)	- Kat. Text: Schrift | Farbe - Kat. Textvariablen: Textfarbe
nFontHeight : INT;	Fonthöhe in Pixel. Sollte im Bereich 10-96 liegen.	vis1.nFontHeight:=16; (Texthöhe ist 16 pt)	- Kat. Text: Schrift | Grad' - Kat. Textvariablen: Fonthöhe
dwFontFlags : DWORD;	Fontdarstellung. Folgende Flags können gesetzt werden: 1 italic 2 fett 4 unterstrichen 8 durchgestrichen + Kombinationen durch Addition der Werte	vis1.dwFontFlags:=10; (Text wird fett und durchgestrichen dargestellt)	- Kat. Text: Schrift | Schriftschnitt - Kat. Textvariablen: Fontflags
stFontName : STRING;	Fontname	vis1.stFontName:='Arial'; (Arial als Schriftart für den Text)	- Kat. Text: Schrift | Schriftart - Kat. Textvariablen: Fontname
nLineWidth : INT;	Linienstärke des Elementrahmens (Anzahl Pixel)	vis1.nLWidth:=3; (Rahmen ist 3 Pixel dick)	- Kat. Linienstärke

Komponente (+Datentyp)	Bedeutung	Beispiel (für das Element wurde der Object Name "vis1" definiert.)	entsprechende Einstell-möglichkeit im Konfigu-rationsdialog:
dwFillColor : DWORD;	Füllfarbe. (zur Eingabe der Farbwerte siehe im Anschluss an die Tabelle)	vis1.dwFillColor":= 16#00FF0000; (Element ist im "Normalzustand" blau)	- Kat. Farben: Farbe | Innen - Kat. Farbvariablen: Farbe innen
Komponente (+Datentyp)	Bedeutung	Beispiel (für das Element wurde der Object Name "vis1" definiert.)	entsprechende Einstellmöglichkeit im Konfigurationsdialog:
dwFillColorAlarm : DWORD;	Füllfarbe im Alarmfall (durch TRUE von Komponente bToggleColor, siehe oben) (zur Eingabe der Farbwerte siehe im Anschluss an die Tabelle)	vis1.dwFillColorAlarm:= 16#00808080; (wenn Variable togglevar auf TRUE gesetzt wird, wird das Element grau)	- Kat. Farben: Alarmfarbe | Innen - Kat. Farbvariablen: Alarmfarbe innen
dwFrameColor: DWORD;	Rahmenfarbe (zur Eingabe der Farbwerte siehe im Anschluss an die Tabelle)	vis1.dwFrameColor:= 16#00FF0000; (Rahmen ist im "Normalzustand" blau)	- Kat. Farben: Farbe | Rahmen - Kat. Farbvariablen: Farbe für Rahmen
dwFrameColorAlarm: DWORD;	Rahmenfarbe im Alarmfall (durch TRUE von Komponente bToggleColor, siehe oben) (zur Eingabe der Farbwerte siehe im Anschluss an die Tabelle)	vis1.dwFrameColorAlarm: =16#00808080; (wenn Variable vis1.bToggleColor auf TRUE gesetzt wird, wird der Elementrahmen grau)	- Kat. Farben: Alarmfarbe Rahmen - Kat. Farbvariablen: Alarmfarbe für Rahmen
dwFillFlags: DWORD;	Farbe, wie mit den Farbvariablen definiert, kann an und abgeschaltet werden. 0 = eingeschaltet, >0 = ausgeschaltet	vis1.dwFillFlags:=1; (das Element wird unsichtbar)	- Kat. Farben: Keine Farbe innen+ Keine Rahmenfarbe - Kat. Farbvariablen: FillFlags
dwFrameFlags: DWORD;	Rahmen-Darstellung 0 Volle Linie 1 gestrichelt (---) 2 gepunktet (...) 3 strich-punkt (_._) 4 strich-punkt-punkt (_..) 8 Linie ausblenden	vis1.FrameFlags:=1; (der Rahmen wird gestrichelt dargestellt)	- Kat. Farbvariablen: FrameFlags

Die Eingabe von Farbwerten:

Beispiel: e1.dwFillColor := 16#00FF00FF;

Eine Farbe wird als Hexadezimalzahl angegeben, die sich aus den Blau/Grün/Rot-Anteilen (RGB) ergibt. Die ersten zwei Nullen nach "16#" sollten gesetzt werden, um die DWORD Größe zu füllen. Für jeden Farbwert stehen 256 (0-255) Farben zur Verfügung

FF Blauanteil
00 Grünanteil
FF Rotanteil

Beispiel für ein blinkendes Visualisierungselement:

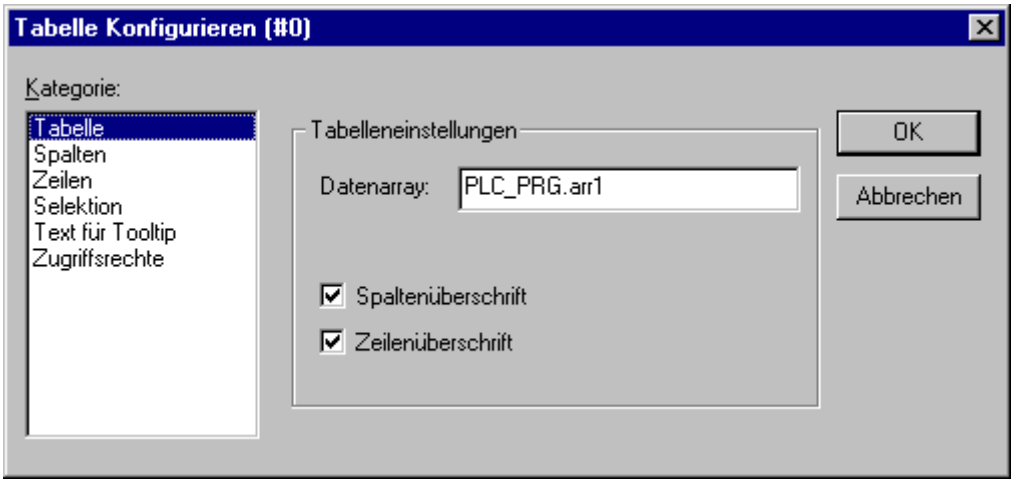
Für ein Rechteck-Element wird eine globale Variable „blinker" vom Typ „VisualObjectType" angelegt. In einem Programm oder Funktionsbaustein kann dann der Wert eines Elements innerhalb der Struktur verändert werden.

```
PROGRAM PLC_PRG
VAR
n:INT:=0;
bMod:BOOL:=TRUE;
END_VAR
(* Blinkendes Element *)
n:=n+1;
bMod:= (n MOD 20) > 10;
IF bMod THEN
    blinker.nFillColor := 16#00808080; (* Grau *)
ELSE
    blinker.nFillColor := 16#00FF0000; (* Blau *)
END_IF
```

9.1.4.2 Tabelle konfigurieren

Sobald eine Tabelle zur Darstellung eines Arrays in die Visualisierung eingefügt wird, öffnet der Dialog 'Tabelle konfigurieren'. Neben den auch für andere Elemente bekannten Kategorien zur Konfiguration von Tooltip und Zugriffsrechten stehen folgende Kategorien zum Definieren von Aussehen und Inhalt der Tabelle bereit:

Kategorie Tabelle:



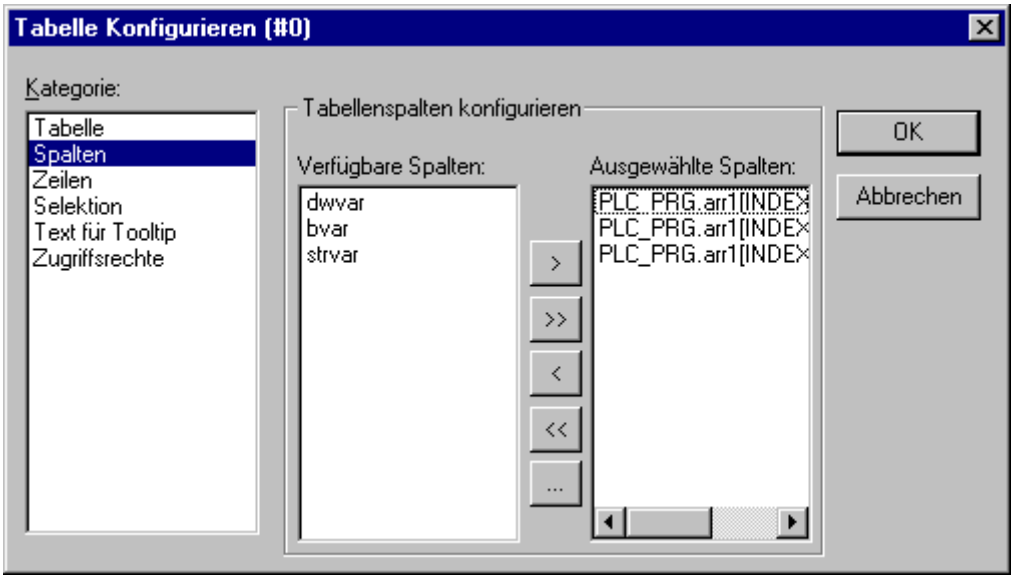
Dialog zum Konfigurieren einer Tabelle, Kategorie Tabelle

Geben Sie hier die folgenden Tabelleneinstellungen an:

Datenarray: Tragen Sie ein Array aus Ihrem Projekt ein, dessen Felder in der Tabelle dargestellt werden sollen. Nehmen Sie die Eingabehilfe (<F2>) bzw. die Intellisense-Funktion zu Hilfe.

Spaltenüberschrift, Zeilenüberschrift: Aktivieren Sie diese Optionen, wenn in der Tabelle die Spalten- bzw. Zeilenüberschriften angezeigt werden sollen. Die Zeilenüberschrift (am linken Rand des Tabellenfeldes) entspricht dem Array-Index, die Spaltenüberschrift kann in der Kategorie 'Spalten' definiert werden.

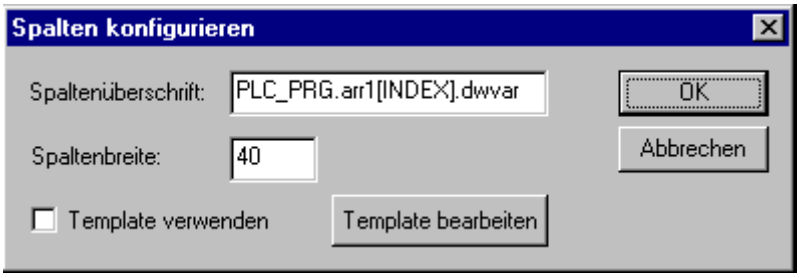
Kategorie Spalten:



Dialog zum Konfigurieren einer Tabelle, Kategorie Spalten

Hier definieren Sie die **Spaltenüberschriften** für die Array-Elemente. Im linken Fensterteil finden Sie die Elemente, die pro Index des Arrays vorliegen. Beispielsweise für das Array einer Struktur alle Strukturelemente.

Über die Pfeiltasten (>, <) zwischen den beiden Fenstern können Sie einzelne Elemente auswählen und ins rechte Fenster übertragen. Mit der Schaltfläche >> werden alle gleichzeitig übertragen. Um nun für eines der Elemente die vordefinierte Darstellung in der Tabellenspalte zu verändern, selektieren Sie den Eintrag und drücken die Schaltfläche **Spalteneigenschaften**. Es öffnet sich der Dialog **Spalten konfigurieren**:



Dialog zum Konfigurieren einer Tabelle, Kategorie Spalten, Spalteneigenschaften

Im Textfeld **Spaltenüberschrift** erscheint zunächst der automatisch vergebene Titel (beim Array einer Struktur z.B. "PLC_PRG.arr1[INDEX].iNo" für die Spalte, die das Strukturelement "iNo" darstellen wird), den Sie editieren können. Außerdem kann die **Spaltenbreite** (Anzahl Zeichen) definiert werden.

Bearbeiten der Konfiguration für alle Felder einer Spalte:

Standardmäßig werden die Tabellenfelder als einfache **Rechtecke** dargestellt, die Einträge sind nicht editierbar. Wenn Sie jedoch die Schaltfläche **Template verwenden** aktivieren, kann zum einen auch eine Darstellung als Bitmap oder Button ausgewählt werden und es können die Parameter des Templates verändert werden. Ein Template bedeutet einen vordefinierten Satz an Einstellungen, z.B. eine bestimmte Definition von Linienstärke, Texteingabemöglichkeit etc. Wenn Sie diese verändern möchten, können Sie dies über die bekannten Konfigurationsdialoge zu Rechteck, Bitmap, Button tun, die sie über die Schaltfläche **Template bearbeiten** erhalten.

Standardmäßig werden die Tabellenfelder als einfache Rechtecke dargestellt, die Einträge sind nicht editierbar. Wenn Sie jedoch für die aktuell markierte Spalte die Schaltfläche Template bearbeiten aktivieren, können die Eigenschaften der Felder dieser Spalte, z.B. eine bestimmte Definition von Linienstärke, Texteingabemöglichkeit etc. verändert werden. Die Vorlage (Template) gilt für alle Felder der aktuell ausgewählten Spalte und kann über den bekannten Konfigurationsdialog für ein Visualisierungselement bearbeitet werden.
Sollen im Template nur für einzelne Zellen innerhalb der Spalte Eigenschaften bzw. Eingaben gesetzt

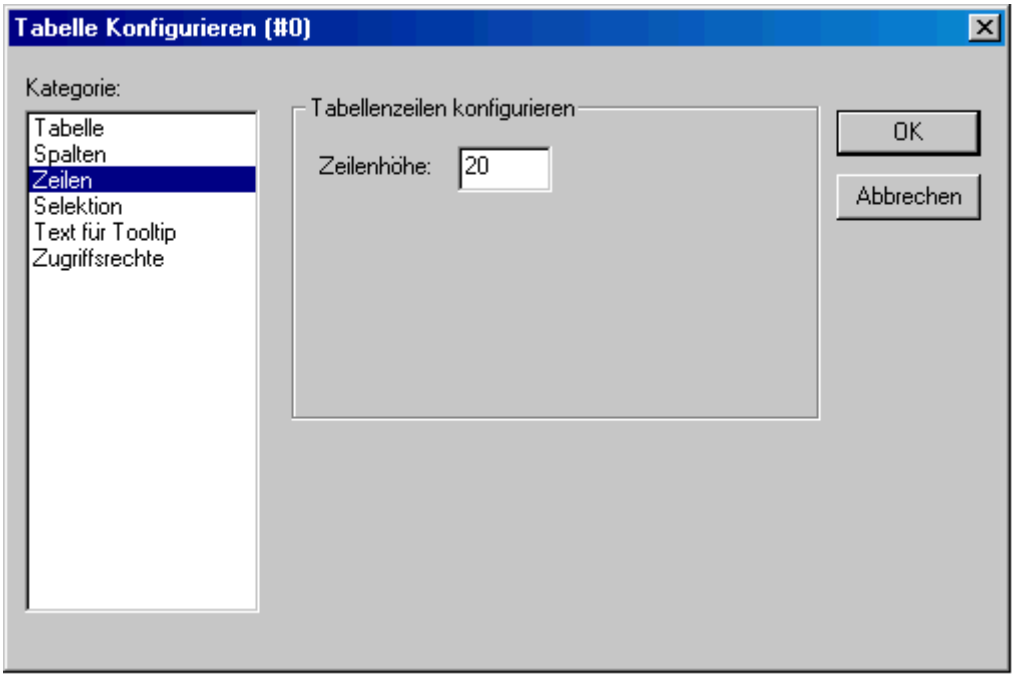
werden, können folgende Platzhalter verwendet werden, die Reihe und Spalte definieren: $ROWCONST$, $COLCONST$, INDEX. (INDEX hat den gleichen Effekt wie $ROWCONST$).

Beispiel für die Verwendung von Platzhaltern in Spalten-Templates

Sie visualisieren ein Array "arr1 [0..2] of BOOL" (-> Tabelle mit 1 Spalte) mit einer Tabelle und im Online-Betrieb soll bei einem Mausklick auf ein Tabellenfeld dieses auf Alarmfarbe rot wechseln und beim nächsten zurück zur Grundfarbe. Gleichzeitig soll der entsprechende Array-Parameter getoggelt werden.
Dazu aktivieren Sie 'Template verwenden' im Konfigurationsdialog für die Tabellenspalte und definieren Sie das Template folgendermaßen:
Kategorie 'Eingabe', Aktion 'Variable toggeln': "PLC_PRG.arr1[INDEX].
Kategorie 'Farben': Alarmfarbe rot.
Kategorie 'Variablen', Aktion 'Farbwechsel': "PLC_PRG.arr1[INDEX].

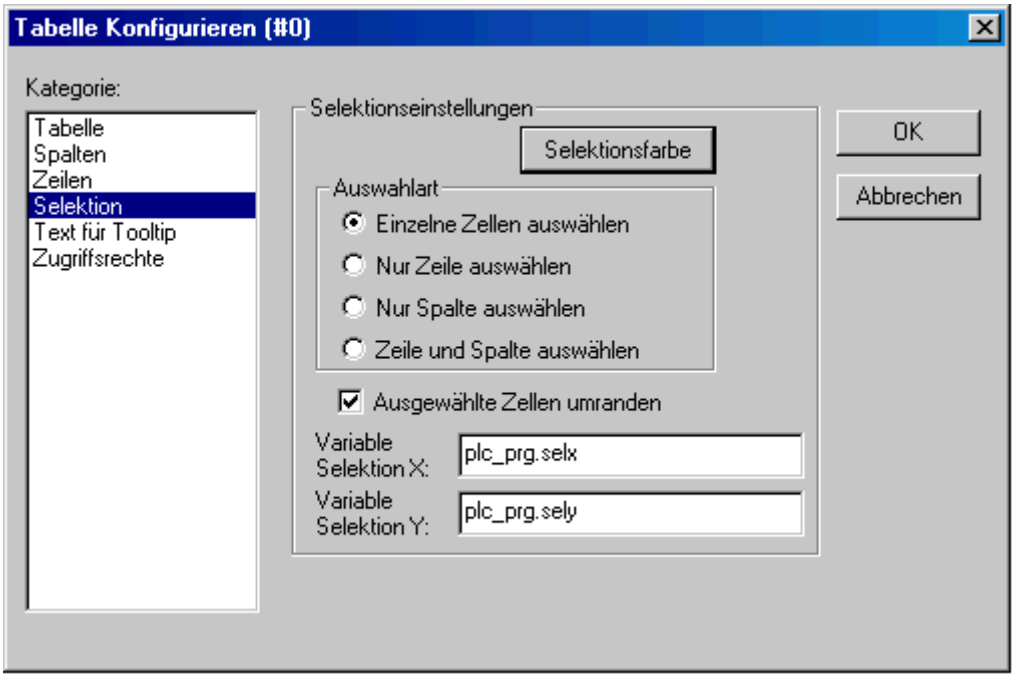
Kategorie Zeilen:

Im Dialog Zeilen können Sie die **Zeilenhöhe** in Pixel definieren.



Dialog zum Konfigurieren einer Tabelle, Kategorie Zeilen

Kategorie Selektion:



Dialog zum Konfigurieren einer Tabelle, Kategorie Selektion

Im Dialog **Selektion** können folgende Einstellungen zu Darstellung und Auswahlverhalten in der Tabelle gemacht werden:

Auswahlart:

Hier bestimmen Sie, welche Auswahl getroffen wird, wenn Sie im Online Modus mit der Maus auf eine Zelle klicken:

Einzelne Zellen auswählen: Nur die angeklickte Zelle wird selektiert.
Nur Zeile auswählen: Die gesamte Zeile wird selektiert.
Nur Spalte auswählen: Die gesamte Spalte wird selektiert.
Zeile und Spalte auswählen: Die gesamte Zeile und Spalte werden selektiert.
Ausgewählte Zellen umranden: Eine selektierte Zelle erhält einen Rahmen.
Variable Selektion X, Variable Selektion Y: Hier können Sie je eine Projektvariable eintragen, die den X- bzw. Y-Index des selektierten Tabellenfeldes anzeigt.

Beispiel:

Tabelle zur Anzeige des Arrays einer Struktur:

Legen Sie in Ihrem Projekt folgende Struktur an:

```
TYPE strucTab :
STRUCT
iNo: INT;
bDigi : BOOL;
sText:STRING;
byDummy: BYTE;
END_STRUCT
END_TYPE
```

Definieren Sie in PLC_PRG folgendes Array:

```
arr1:ARRAY [1..5] OF strucTab;
```

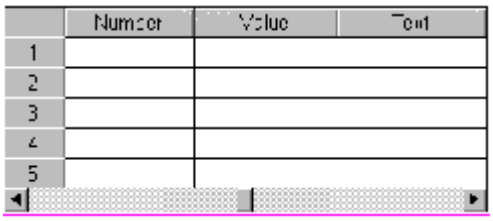
und folgende Variablen:

```
selX:INT;
selY:INT;
```

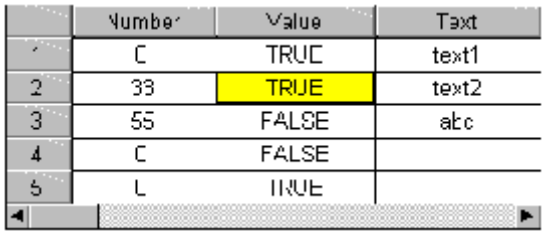
Fügen Sie in der Visualisierung eine Tabelle ein und konfigurieren Sie folgendermaßen:

```
Kat. Tabelle: Datenarray: PLC_PRG.arr1
```

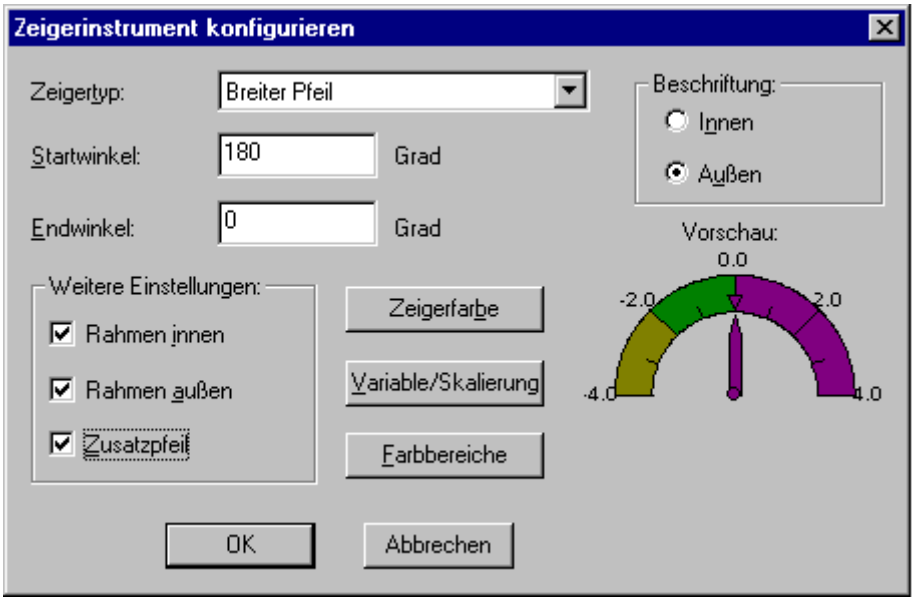
Kat. Spalten: (Beantworten Sie den erscheinenden Frage-Dialog mit JA) - Übertragen Sie die Elemente iNo, bDigi, sText in das rechte Fenster - Selektieren Sie im rechten Fenster den ersten Eintrag (PLC_PRG.arr1[INDEX].iNo), drücken Sie die Schaltfläche Spalteneigenschaften und verändern die Spaltenüberschrift zu "Number". Bestätigen Sie mit OK und definieren Sie auch neue Spaltenüberschriften für die anderen beiden Einträge (z.B. "Value" und "Text"). – In der Kategorie 'Spez.Tabelle' tragen Sie bei Variable Selektion X ein: "PLC_PRG.selX" und bei Variable Selektion "Y: PLC_PRG.selY". Aktivieren Sie die Option 'Rahmen um selektierte Zellen'. Drücken Sie die Schaltfläche 'Selektionsfarbe' und wählen Sie eine Farbe aus. Schließen Sie den Konfigurationsdialog mit OK. Das Tabellenelement sollte nun folgendermaßen aussehen.



Am linken Rand die Nummerierung des Array-Index, oben die Spaltentitel der zur Anzeige gewählten drei Strukturelemente. Die Spaltenbreiten können Sie verändern, indem Sie mit der Maus, wenn der Mauszeiger als waagrechter doppelköpfiger Pfeil erscheint die Trennlinien nach links oder rechts ziehen.
Im Online Modus erscheinen in den Tabellenfeldern die aktuellen Werte der Array-Elemente. Wenn Sie ein Tabellenfeld mit der Maus anklicken, wird es mit einem Rahmen umgeben und in der gewählten Farbe erscheinen. Z.B.:



Zeigerinstrument konfigurieren



Dialog zum Konfigurieren eines Zeigerinstruments

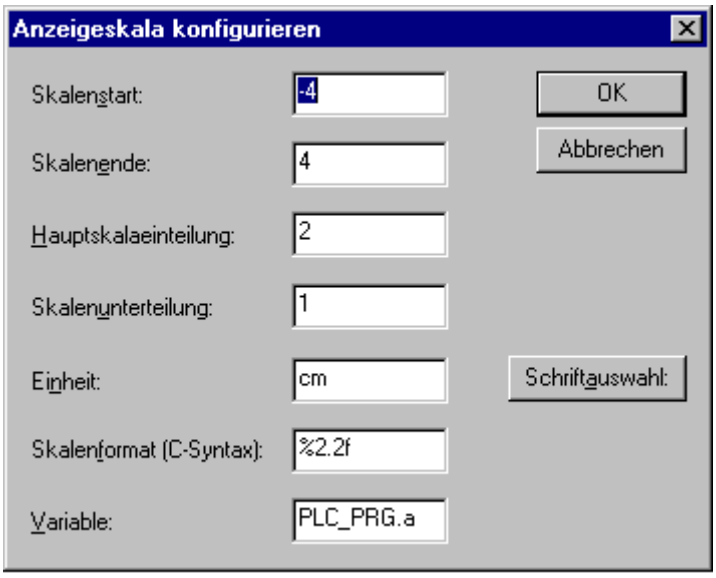
Dieser Dialog öffnet automatisch, sobald ein Anzeigeinstrument in die Visualisierung eingefügt wird. Eine Vorschau zeigt jeweils unmittelbar das Aussehen des Elements entsprechend den eingestellten Parametern:

Zeigertyp: Wählen Sie zwischen 'Normaler Pfeil', 'Dünner Pfeil', "Breiter Pfeil' und 'Dünne Nadel', der auf den jeweils aktuellen Wert auf der Anzeigeskala zeigen soll.

Startwinkel, Endwinkel: Geben Sie hier in ° Grad (Winkelgrad) die Position für Beginn und Ende der Anzeiger-Skala auf einem Kreisbogen an (Beispielsweise definieren ein Startwinkel von 180° und ein Endwinkel von 0° dass die Anzeigeskala im nach oben gebogenen Halbkreis dargestellt wird).

Zeigerfarbe: Diese Schaltfläche öffnet den Standard-Dialog zur Auswahl einer Farbe für den Zeiger.

Variable/Skalierung: Diese Schaltfläche öffnet den Dialog Anzeigeskala konfigurieren:



Dialog zum Konfigurieren der Skaleneinteilung eines Zeigerinstruments

Skalenstart, Skalenende: unterster und oberster Wert auf der Anzeigeskale, z.B. "-4" und "4".

Hauptskalaeinteilung: Angabe, in welchen Schritten die Skalenwerte von Skalenstart bis Skalenende auf der Skala voll angezeigt (d.h. Beschriftung + Skalenstrich) werden sollen. Beispielsweise wird bei Eingabe von "2" jeder zweite ganzzahlige Skalenwert angezeigt.

Skalenunterteilung: Zusätzlich zur Hauptskala (Beschriftung + lange Skalenstriche) kann hier die Anzeige einer Unterskala definiert werden, die nur aus zusätzlichen, kurzen Skalenstrichen ohne Beschriftung besteht.

Einheit: Die Einheit der Skala kann eingegeben werden, z.B. "cm" oder "sec". Die Einheit wird am Zeigerursprung angezeigt.

Skalenformat (C-Syntax): Entsprechend der C-Syntax kann hier das Darstellungsformat der Anzeige der Skalenbeschriftung angegeben werden; siehe hierzu unter Kategorie 'Text'.. Beispielsweise ergibt eine Eingabe von %1.1f eine Anzeige der Skalenwerte als Kommazahl mit einer Stelle vor und einer Stelle nach dem Komma (z.b. "12.0")

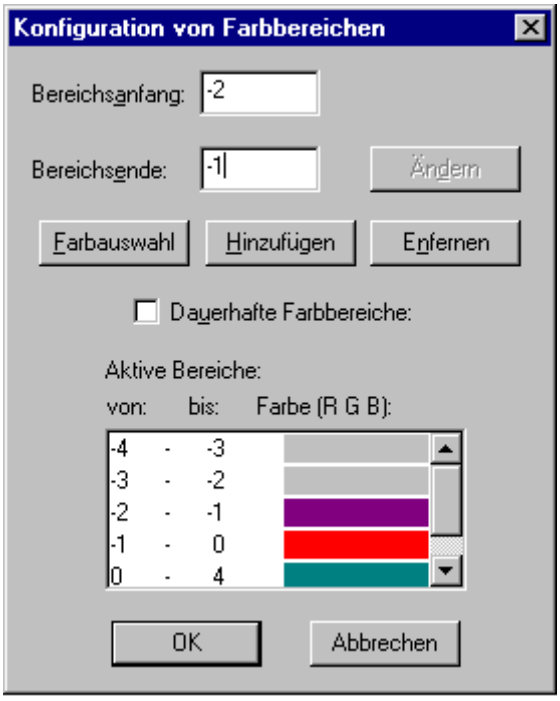
Variable: Die hier eingetragene Variable definiert die Position des Zeigers (z.B. "PLC_PRG.posvar")

Schriftauswahl: Diese Schaltfläche öffnet den Standard-Dialog zur Festlegung der Schriftdarstellung im Anzeigeinstrument

Farbbereiche: Diese Schaltfläche öffnet den Dialog **Konfiguration von Farbbereichen**: Hier können Sie für jeden Bereich der Anzeigeskala eine Farbe festlegen:

Bereichsanfang, Bereichsende: Geben Sie hier Anfangs- und Endwert des Skalenbereichs ein, der die im folgenden zu definierende Farbe erhalten soll:

Farbauswahl: Diese Schaltfläche führt zum Standarddialog zur Auswahl einer Farbe. Bestätigen Sie Ihre Wahl mit OK, worauf der Dialog wieder schließt und drücken Sie die Schaltfläche Hinzufügen, woraufhin die Farbe und der gewählte Skalenbereich im Fenster der Aktiven Bereiche angezeigt wird. Um einen Farbbereich zu entfernen, markieren Sie ihn in diesem Fenster und drücken die Schaltfläche Entfernen.



Dialog zum Konfigurieren eines Farbbereichs

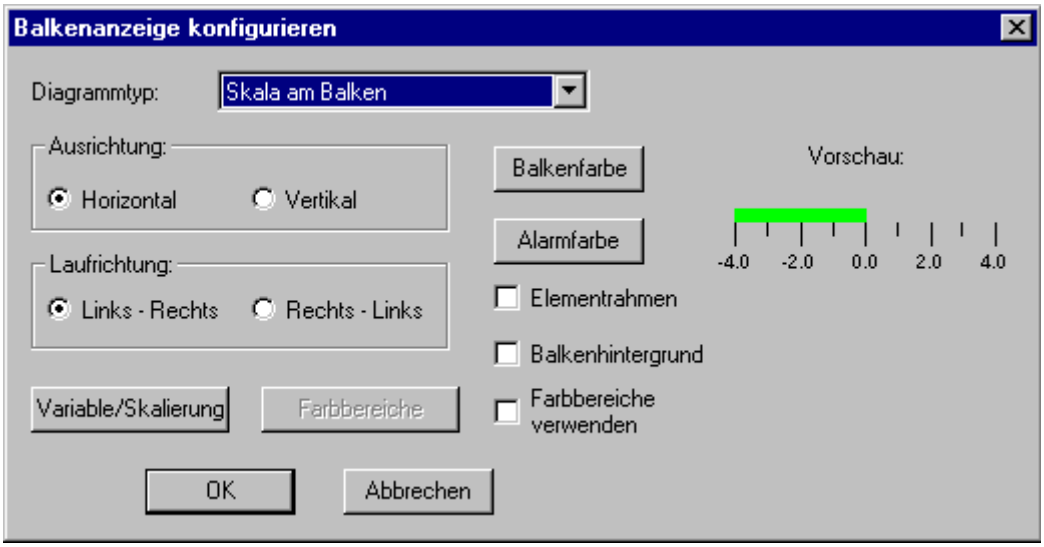
Wenn die Option **Dauerhafte Farbbereiche** aktiviert ist, werden die definierten Farbbereiche immer angezeigt, ansonsten wird im Online Modus nur die Farbe des Bereichs angezeigt, in dem der aktuelle Variablenwert liegt.
Beschriftung: Je nach aktivierter Option Innen oder Außen wird die Beschriftung der Anzeigeskala auf der Innen- oder Außenseite des Skalenbogens dargestellt.

Weitere Einstellungen:

Rahmen innen, Rahmen außen: Wenn die Option aktiviert ist, erhält die Anzeigeskala eine inneren und/ oder äußere Begrenzungslinie.

Zusatzpfeil: Zusätzlich zum Zeiger wird direkt auf der Anzeigeskala ein kleiner Pfeil dargestellt.

Balkenanzeige konfigurieren



Dialog zum Konfigurieren einer Balkenanzeige

Dieser Dialog öffnet automatisch, sobald ein Balkenanzeige-Element in die Visualisierung eingefügt wird, um einen Variablenwert durch die Länge eines Balkens entlang einer horizontalen oder vertikalen Skala darzustellen. Eine Vorschau zeigt jeweils unmittelbar das Aussehen des Elements entsprechend den eingestellten Parametern:

Diagrammtyp: Wählen Sie zwischen 'Skala am Balken', 'Skala im Balken' und 'Balken in der Skala'.

Ausrichtung: Wählen Sie, ob die Anzeige durch einen horizontalen oder vertikalen Balken erfolgen soll.

Laufrichtung: Wählen Sie, ob die Werte der Anzeigeskala von links nach rechts ansteigen sollen oder von rechts nach links.

Balkenfarbe: Diese Schaltfläche öffnet den Standard-Dialog zur Auswahl einer Farbe für den Balken außerhalb des Alarmzustands (siehe Alarmfarbe). Wenn die Option 'Farbbereiche verwenden' (siehe unten) aktiviert ist, sind hier keine Eingaben möglich, bzw. sind diese deaktiviert.

Alarmfarbe: Diese Schaltfläche öffnet den Dialog Alarm konfigurieren, in dem Sie festlegen, bei welchem Wert der Balken in die Alarmfarbe wechselt und welche dies ist. Geben Sie hierzu im Texteingabefeld den gewünschten Grenzwert ein und aktivieren Sie eine der Bedingungen größer oder kleiner, um zu definieren, ob Werte oberhalb oder unterhalb dieses Wertes den Alarmzustand auslösen sollen. Drücken Sie die Schaltfläche Alarmfarbe, um im Standard-Farbauswahldialog die Alarmfarbe auszusuchen. Schließen Sie beide Dialoge mit OK, um die Eingaben zu bestätigen und zum Hauptkonfigurationsdialog für die Balkenanzeige zurückzukehren. Wenn die Option 'Farbbereiche verwenden' (siehe unten) aktiviert ist, sind hier keine Eingaben möglich, bzw. sind diese deaktiviert.

Variable/Skalierung: Diese Schaltfläche öffnet den Dialog Anzeigeskala und Variable konfigurieren, der dem eines Zeigerinstruments-Elements entspricht.

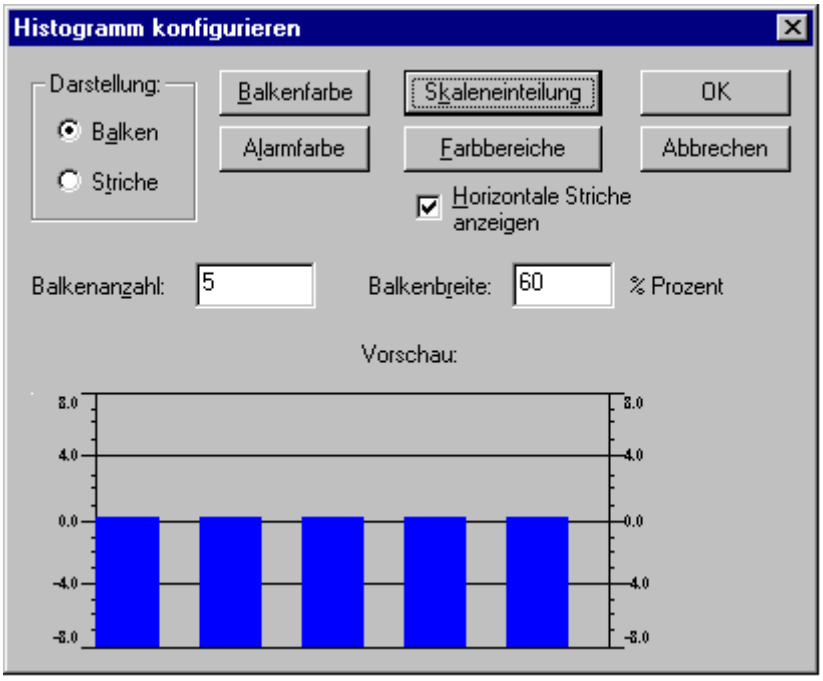
Elementrahmen: Wenn diese Option aktiviert ist, erhält die Balkenanzeige einen Rahmen.

Balkenhintergrund: Wenn diese Option aktiviert ist, wird der gesamte definierte Anzeigebereich als schwarzer Balken hinter dem aktuellen dargestellt, ansonsten wird nur der den aktuellen Wert zeigende Balken angezeigt.

Farbbereiche verwenden: Wenn diese Option aktiviert ist, gelten nicht die unter Balkenfarbe und Alarmfarbe vorgenommenen Einstellungen, sondern die Farbbereichseinstellungen, die über die Schaltfläche Farbbereiche definiert wurden. Entsprechend der Vorgehensweise beim Zeigerinstrument-Element öffnet dazu der Dialog 'Konfiguration von Farbbereichen'.

Histogramm konfigurieren

Ein Histogramm-Element kann verwendet werden, um ein Array zu visualisieren. Die Werte der Array-Elemente können nebeneinander als senkrechte Balken oder Striche dargestellt werden.



Dialog zum Konfigurieren eines Histogramms

Der Konfigurationsdialog öffnet automatisch, sobald das Histogramm-Element in die Visualisierung eingefügt wird. Eine Vorschau zeigt jeweils unmittelbar das Aussehen des Elements entsprechend den eingestellten Parametern:

Darstellung: Wählen Sie eine der Optionen Balken oder Striche.

Horizontale Striche anzeigen: Wenn diese Option aktiviert ist, werden entsprechend der Skaleneinteilung zwischen den links und rechts angeordneten Skalen horizontale Linien gezogen.

Balkenfarbe: Diese Schaltfläche öffnet den Standard-Dialog zur Auswahl einer Farbe für den Balken außerhalb des Alarmzustands (siehe **Alarmfarbe**). Wenn die Option **'Farbbereiche verwenden'** (siehe unten) aktiviert ist, sind hier keine Eingaben möglich, bzw. sind diese deaktiviert.

Alarmfarbe: Diese Schaltfläche öffnet den Dialog **Alarm konfigurieren**, in dem Sie festlegen, bei welchem Wert der Balken in die Alarmfarbe wechselt und welche dies ist. Geben Sie hierzu im **Texteingabefeld** den gewünschten Grenzwert ein und aktivieren Sie eine der Bedingungen **größer** oder **kleiner**, um zu definieren, ob Werte oberhalb oder unterhalb dieses Wertes den Alarmzustand auslösen sollen. Drücken Sie die Schaltfläche **Alarmfarbe**, um im Standard-Farbauswahldialog die Alarmfarbe auszusuchen. Schließen Sie beide Dialoge mit **OK**, um die Eingaben zu bestätigen und zum Hauptkonfigurationsdialog für die Balkenanzeige zurückzukehren. Wenn die Option **'Farbbereiche verwenden'** (siehe unten) aktiviert ist, sind hier keine Eingaben möglich, bzw. sind diese deaktiviert.

Skaleneinteilung: Diese Schaltfläche öffnet den Dialog Anzeigeskala konfigurieren, der dem eines Zeigerinstrument-Elements entspricht.

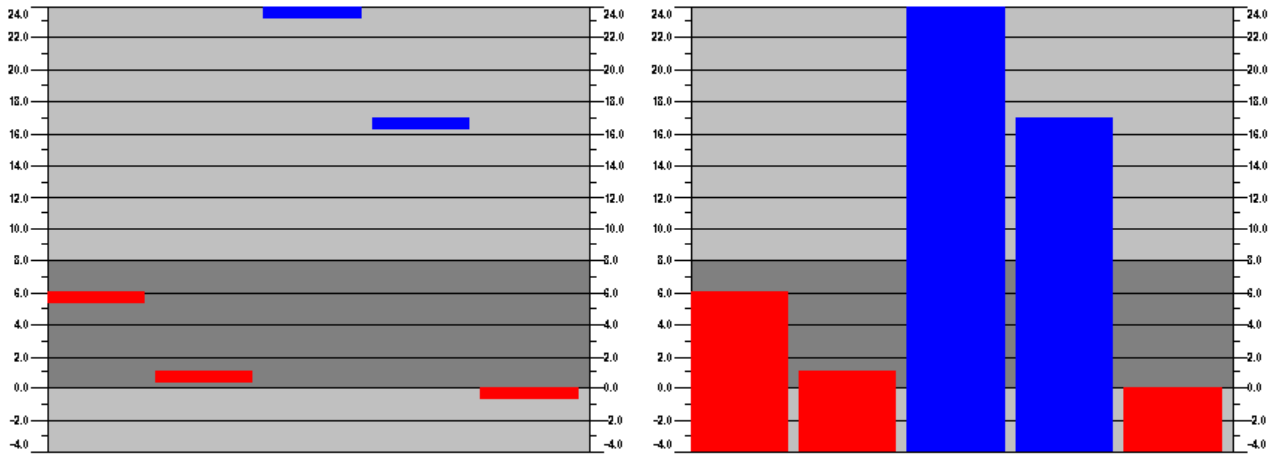
Farbbereiche: Diese Schaltfläche öffnet den Dialog **Konfiguration von Farbbereichen:** Hier können Sie entsprechend der Vorgehensweise beim Zeigerinstrument-Element für jeden Bereich der Anzeigeskala eine Farbe festlegen.

Balkenanzahl: Geben Sie hier die gewünschte Anzahl der Balken an.

Balkenbreite: Geben Sie hier die Breite der Balken in Prozent der für den Balken verfügbaren Gesamtbreite an.

Beispiel

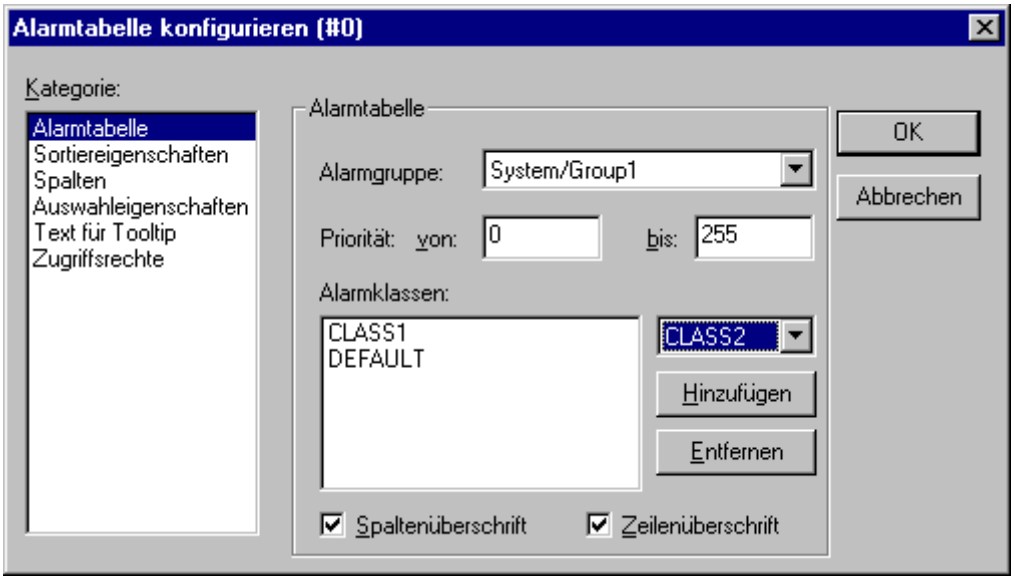
Sehen Sie in der folgenden Abbildung ein Beispiel der Online-Darstellung eines Histogramms in Darstellung als 'Striche' bzw. 'Balken' für beispielsweise ein Array [0..4] of INT. Dafür wurden eine Balkenanzahl von "5", der Skalenstart "–4", das Ende "24" eingestellt, die Haupteinteilung wurde auf "2", die Unterteilung auf "1" gesetzt und der Skalenbereich von 0 – 8 mit einer anderen Farbe hinterlegt als der Rest. Außerdem sollen die Balken ab einem Variablenwert von 8 die Alarmfarbe blau annehmen.



9.1.4.3 Alarmtabelle konfigurieren

Das Element Alarmtabelle dient der Visualisierung von Alarmen, die in der Alarmkonfiguration entsprechend konfiguriert sein müssen.
Sobald das Element in ein Visualisierungsobjekt eingefügt wird, öffnet der Dialog 'Alarmtabelle konfigurieren'. Neben den auch für andere Elemente bekannten Kategorien zur Konfiguration von Tooltip und Zugriffsrechten können folgende Einstellungen zur Darstellung und Auswahlverhalten in der Tabelle gemacht werden:

Kategorie Alarmtabelle:



Dialog zum Konfigurieren einer Alarmtabelle, Kategorie Alarmtabelle

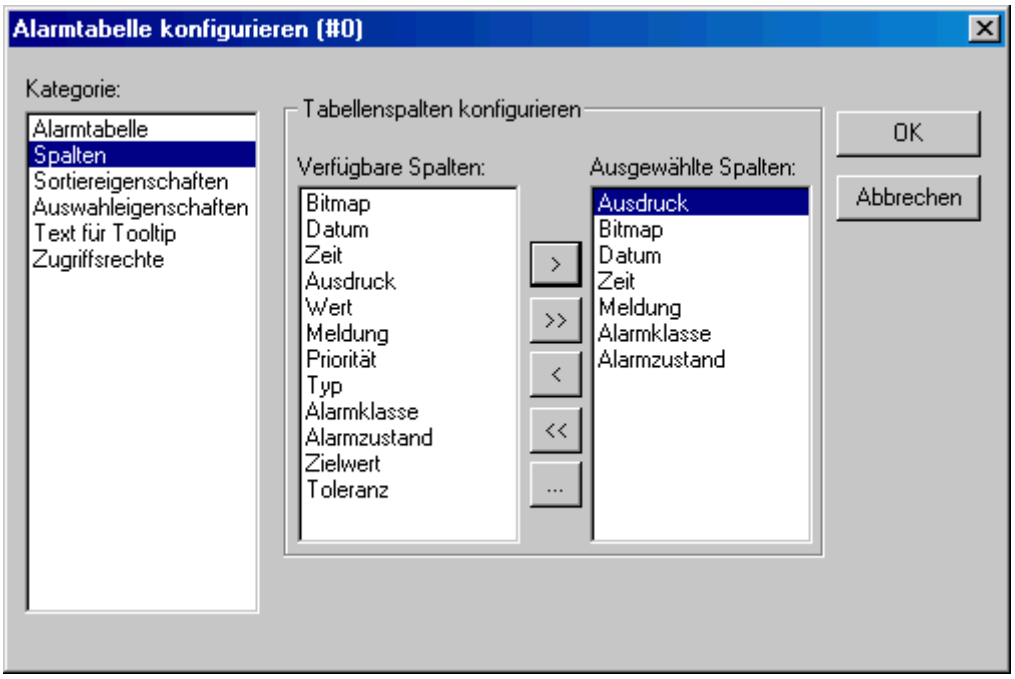
Definieren Sie, was in der Tabelle dargestellt werden soll:

Alarmgruppe ändern: Wählen Sie aus dem Auswahlbaum der Alarmkonfiguration, der über diese Schaltfläche geöffnet wird, die anzuzeigende Gruppe aus (die auch nur einen einzigen Alarm enthalten kann). Angeboten werden alle in der Alarmkonfiguration definierten Gruppen.

Priorität: Definieren Sie, die Alarme welcher Prioritäten angezeigt werden sollen. Die maximal mögliche Bandbreite: von: 0 bis: 255.

Ausgewählte Alarmklassen: Markieren Sie eine zur Anzeige gewünschte Klasse in der Auswahlliste rechts des Felds 'Ausgewählte Alarmklassen' und drücken die Schaltfläche **Hinzufügen**, so dass die Klasse ins Feld übernommen wird. Führen Sie dies für alle benötigten Klassen durch. Über **Entfernen** können Sie einen im Fenster markierten Eintrag wieder entfernen. Aktivieren Sie die Optionen **Spaltenüberschrift** bzw. **Zeilenüberschrift**, wenn die Überschriften in der Tabelle angezeigt werden sollen.

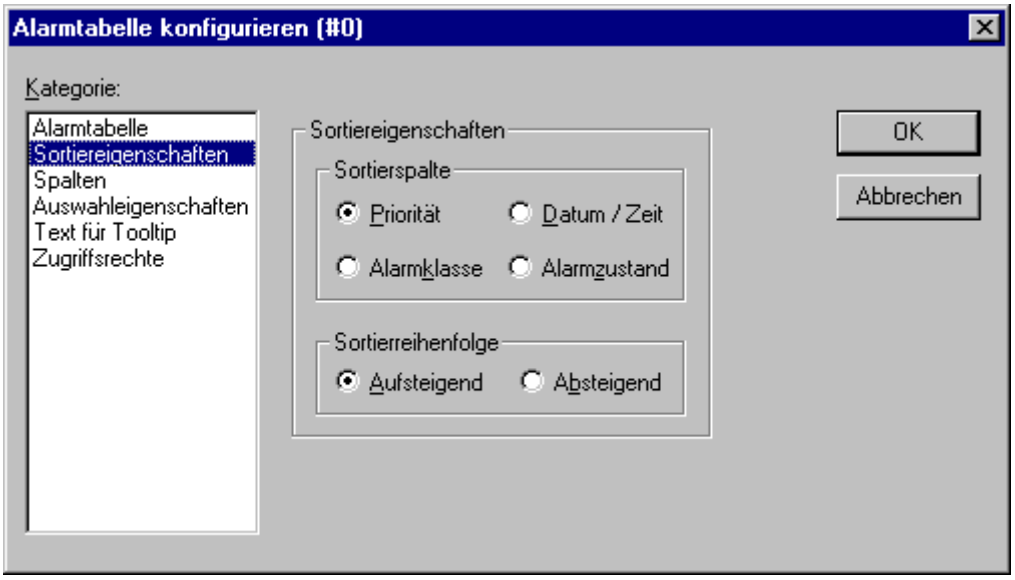
Kategorie Spalten:



Dialog zum Konfigurieren einer Alarmtabelle, Kategorie Spalten

Definieren Sie hier, welche der folgenden Spalten (Alarmparameter) in der Alarmtabelle angezeigt werden sollen: Die Parameter werden mit Ausnahme von Datum und Zeit (Alarmeintrittszeitpunkt) und Alarmzustand in der Alarmgruppenkonfiguration definiert: **Bitmap, Datum, Zeit, Ausdruck, Wert, Meldung, Priorität, Typ, Alarmklasse, Alarmzustand, Zielwert** (für die Alarmtypen DEV+ und DEV-), **Toleranz**. Über die Schaltflächen **">"**, **">>"**, können Sie einzelne bzw. alle Parameter vom linken (**Verfügbare Spalten**) ins rechte Fenster (**Ausgewählte Spalten**) übertragen, die dort angezeigte Auswahl wird dann in der Alarmtabelle angezeigt. Über die Schaltflächen **"<"** bzw. **"<<"** können Sie wieder aus der Auswahl entfernt werden.

Kategorie Sortiereigenschaften:



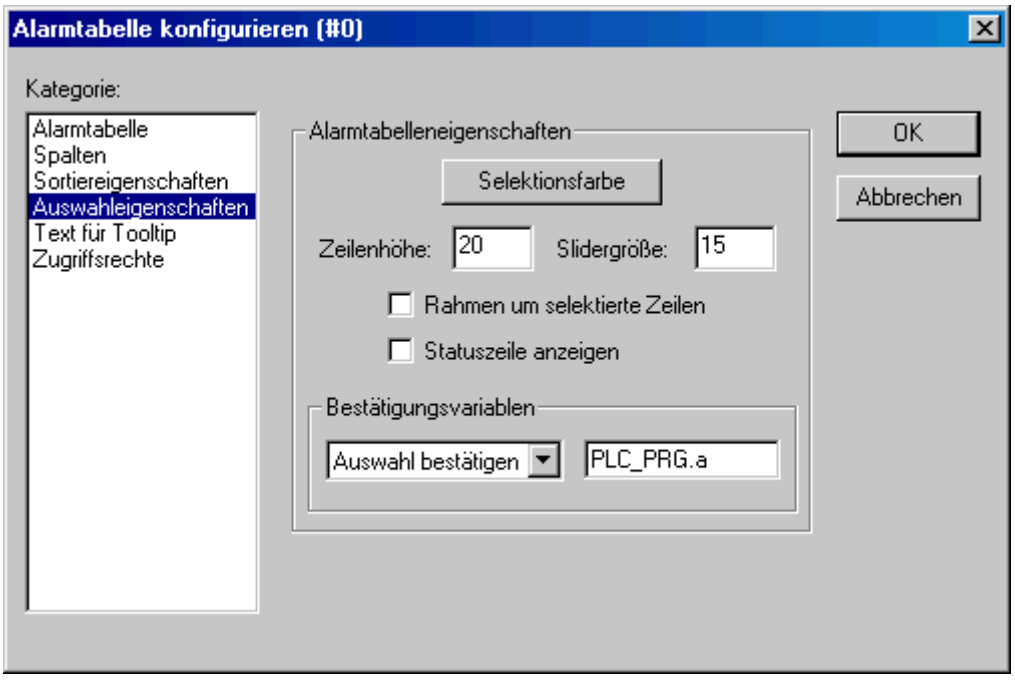
Dialog zum Konfigurieren einer Alarmtabelle, Kategorie Sortiereigenschaften

Definieren Sie hier, nach welchen Kriterien die Alarmtabelle sortiert werden soll:

Sortierspalte: Sortierung nach Priorität, Alarmklasse, Datum/Zeit oder Alarmzustand

Sortierreihenfolge: Aufsteigend oder Absteigend; Beispiel: Aufsteigend nach Priorität bedeutet, dass die Tabelle mit Alarmen der Priorität 0 beginnt (sofern vorhanden), dann die mit höherzahligen Prioritäten.

Kategorie Auswahleigenschaften:



Dialog zum Konfigurieren einer Alarmtabelle, Kategorie Auswahleigenschaften

Definieren Sie hier, was für die Darstellung ausgewählter Tabellenfelder gilt:

Selektionsfarbe: Diese Schaltfläche öffnet den Standarddialog zur Auswahl einer Farbe, in der selektierte Felder angezeigt werden sollen.

Zeilenhöhe: Höhe der Tabellenfelder (Rows) in Pixel.

Sliderhöhe: Höhe der Scrollleiste (in Pixel) am unteren Rand der Tabelle.

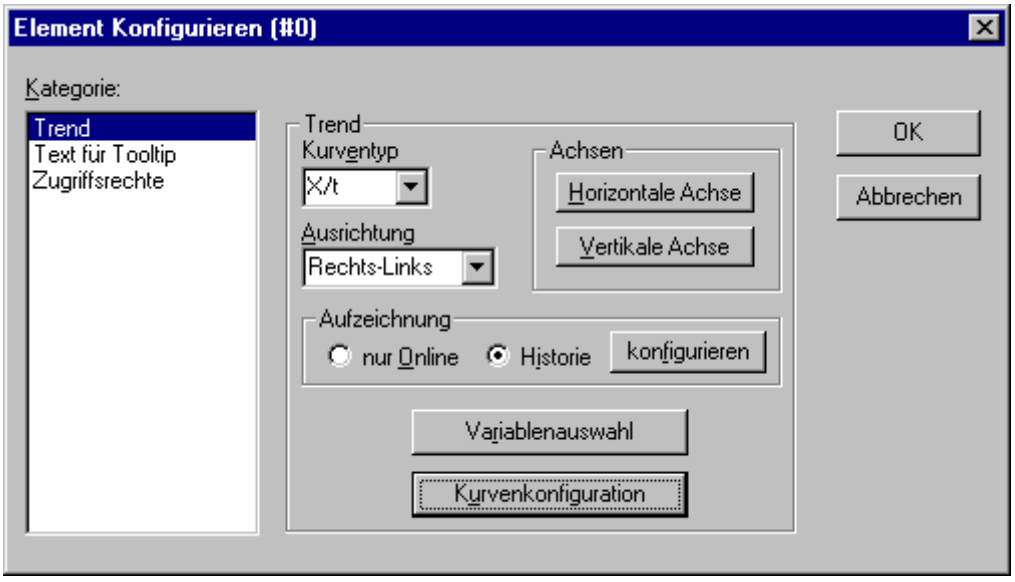
Rahmen um selektierte Zeilen: Wenn diese Option aktiviert ist, wird um die selektierte Tabellenzeile ein Rahmen gezeichnet.

Statuszeile anzeigen: Wenn diese Option aktiviert ist, wird unterhalb der Alarmtabelle eine Statusleiste dargestellt, die folgende Schaltflächen für die Bedienung im Online Modus enthält:

Auswahl bestätigen: Alle in der Alarmtabelle selektierten Alarmeinträge werden bestätigt.
Alle bestätigen: Alle Einträge der Alarmtabelle werden bestätigt.
Geschichte: Wenn diese Schaltfläche gedrückt ist, werden anstelle des momentanen Status aller Alarme alle bereits aufgetretenen Events (Übergänge zwischen Alarmstati) angezeigt. In dieser Liste sind keine Bestätigungen möglich ! Neu hinzukommende Events werden laufend hinzugefügt. Wenn Sie eine Speicherdatei definiert haben, können Sie diese Historie für alle Alarmklassen, denen die Aktion 'Speichern' zugewiesen ist, darin ebenfalls finden.
Start: hebt Stop (siehe unten) auf
Stop: Die laufende Aktualisierung der Liste mit den neu hinzukommenden Events wird gestoppt, bis sie mit 'Start' wieder angestossen wird.

Bestätigungsvariablen: Diese Konfigurationsmöglichkeit besteht nur, wenn nicht die Option 'Statuszeile anzeigen' (siehe oben) aktiviert ist. Die Funktionen der oben für die Statuszeile beschriebenen Schaltflächen können dann mit Variablen gesteuert werden. Um diese festzulegen, wählen Sie aus der Auswahlliste eine Funktion aus und geben im Eingabefeld daneben eine Projektvariable an (<F2>). Im Online Modus kann dann beispielsweise die Bestätigung aller Alarme durch die steigende Flanke der zugeordneten Variable erfolgen.

Trend



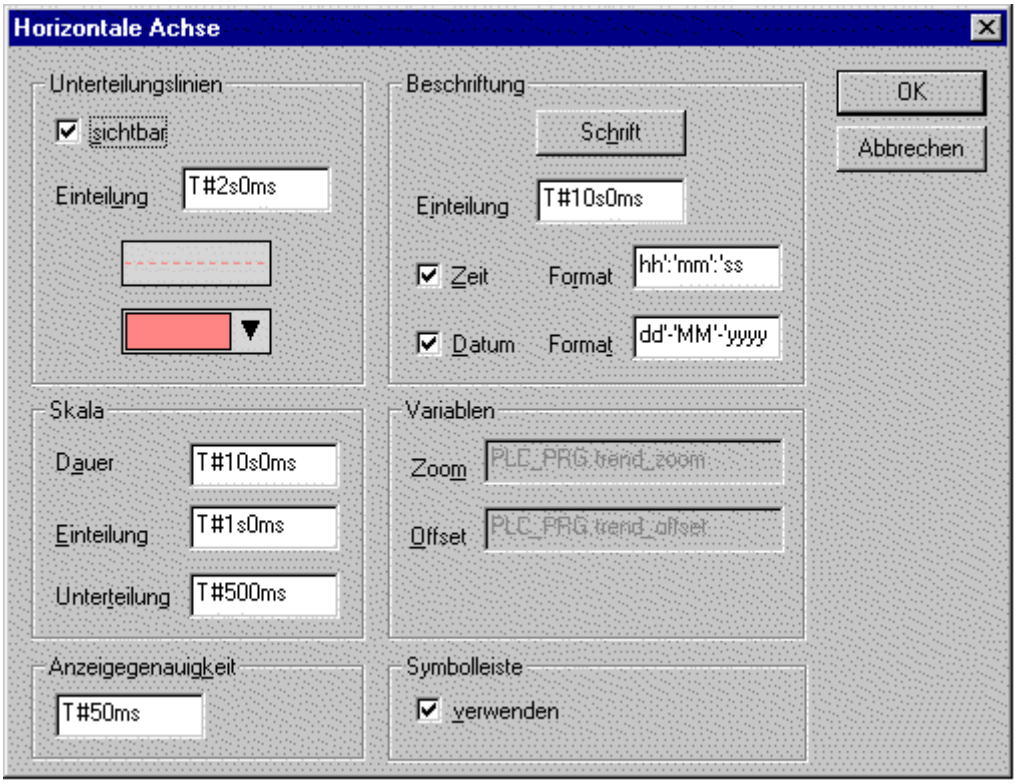
Dialog zum Konfigurieren eines Trend Elements

Die Trend-Funktion dient der Aufzeichnung des zeitlichen Verlaufs von Variablenwerten im Online Modus. Sie ist vergleichbar mit der Trace-Funktionalität. Die Online-Darstellung erfolgt in einem Diagramm, bei der Aufzeichnung in eine Text-Datei werden die einzelnen Messwerte in Zeilen untereinander geschrieben. Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie 'Trend' folgende Einstellungen zur Darstellung vornehmen:

Kurventyp: X/t, horizontale Achse = Zeitachse, vertikale Achse = Werteskala

Ausrichtung: Links-Rechts oder Rechts-Links: Der neueste Wert wird links oder rechts dargestellt;

Achsen:
Horizontale Achse:



Dialog zum Konfigurieren der horizontalen Achse im Trend-Element

Unterteilungslinien: Aktiveren Sie die Option **sichtbar**, wenn senkrechte Unterteilungslinien in Verlängerung der Skaleneinteilungsstriche angezeigt werden sollen. Für diesen Fall können Sie die Einteilung definieren: die eingetragene Zahl gibt an, in welchen Abständen die Unterteilungslinien auf der horizontalen Achse angezeigt werden sollen. Aussehen (normal ___, dashed _ _ _, dotted ....., dashdotted _ . _ .) und Farbe der Linien werden definiert, indem man auf das Rechteck mit der aktuellen Liniendarstellung bzw. auf das unterhalb mit der aktuell eingestellten Farbe klickt, um entsprechende Auswahl-Dialoge zu erhalten.

Skala: Der dargestellte Wertebereich der Skala wird durch die Einträge bei **Dauer** festgelegt. Wird hier z.B. T#20s0ms eingetragen, stellt die Skala einen Bereich von 20 Sekunden dar. Die als längere und kürzere Markierungsstriche darzustellende **Einteilung** und **Untereinteilung** werden in derselben Syntax festgelegt.

Anzeigegenauigkeit: Geben Sie hier im Standardformat für Zeitangaben (z.B. T#50ms) an, in welchen zeitlichen Abständen die aktuellen Werte der Variablen aufgezeichnet werden sollen.

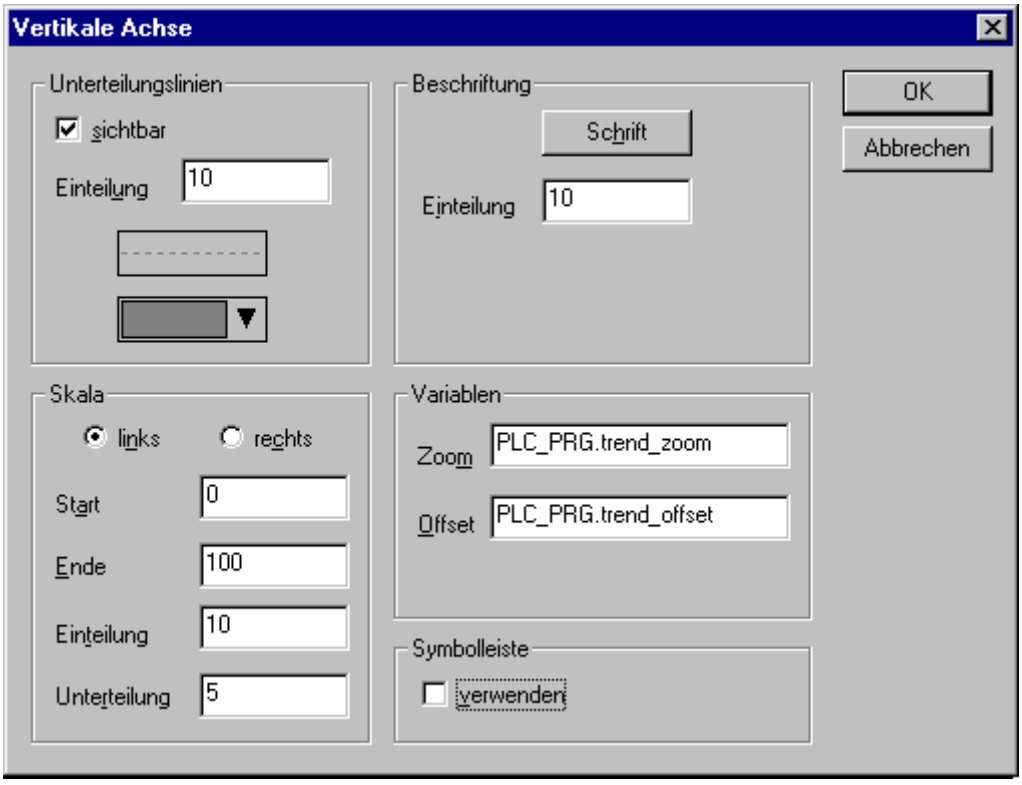
Beschriftung: Hier wird die Darstellung der Beschriftung definiert. Über die Schaltfläche Schrift wird dazu der Standarddialog 'Schriftart' geöffnet. Bei Einteilung geben Sie an, in welchen Abständen auf der Skala beschriftet werden soll (z.B. T#4ms, wenn die Skalenstriche in 4 Millisekunden-Abständen beschriftet werden sollen. Die Beschriftung kann die Uhrzeit (Zeit) und/oder das Datum enthalten, je nachdem, welche Optionen aktiviert sind. Das gewünschte Format kann jeweils im Textfeld in der Standardsyntax für Zeit und Datum angegeben werden.

Variablen: Hier können Projektvariablen eingetragen werden, die die Zoom- bzw. Offset-Werte der horizontalen Skala beinhalten. Beispielsweise wird der Offset des Anzeigebereichs der vertikalen Achse auf "10" gesetzt werden, sobald die entsprechende Variable den Wert "10" annimmt.

Symbolleiste: Wenn die Option verwenden aktiviert ist, wird am unteren Rand des Elements eine horizontale Symbolleiste angezeigt, die Schaltflächen für schrittweises Scrollen im Online Modus nach links oder rechts sowie für die Zoom-Funktion enthält.

Mit den einfachen Pfeilen wird der Darstellungsbereich schrittweise, mit den Doppelpfeilen an Ende bzw. Start des Aufzeichnungszeitraums verschoben. Die Zoom-Tasten erlauben ein schrittweises Zoomen des dargestellten Anzeigebereichs .

Vertikale Achse:



Dialog zum Konfigurieren der vertikalen Achse im Trend-Element

Unterteilungslinien: entsprechend wie bei horizontaler Achse (siehe oben)

Skala: Wählen Sie, ob die Skala am linken (links) oder rechten (rechts) Rand des Trend-Diagramms dargestellt werden soll. Wählen Sie den Start-(unteres Ende) und Endewert (oberes Ende) der Skala sowie die Einteilung und Unterteilung (längere und kürzere Markierungsstriche werden im hier angegebenen Abstand eingezeichnet).

Beschriftung: Schrift und Einteilung entsprechend horizontale Achse (siehe oben)

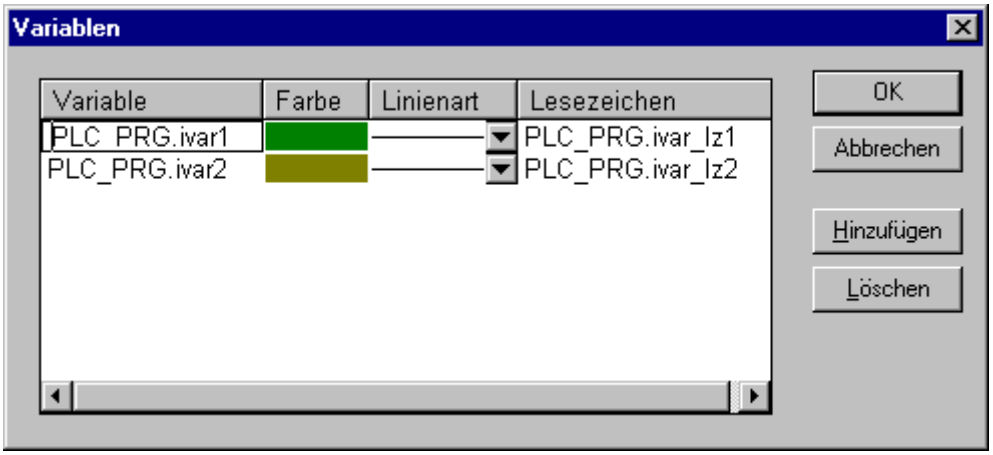
Variablen: entsprechend wie bei horizontaler Achse (siehe oben)

Symbolleiste: entsprechend wie bei der horizontalen Achse (siehe oben), zusätzlich ist hier das Symbol

 enthalten, mit dem die Aufzeichnungsdarstellung stets auf die Standard-Einstellung (0% Zoom, aktueller Zeitbereich) zurückgesetzt werden kann.

Aufzeichnung: Wählen Sie hier, ob die Trend-Aufzeichnung **nur Online** aufgezeichnet werden soll, was bedeutet, dass der zeitliche Verlauf der Variablenwerte im ausgewählten Skalenbereich angezeigt wird, oder ob die Aufzeichnung in einer Historie gespeichert werden soll, die Sie nach Drücken der Schaltfläche **Historie** konfigurieren können. Der Dialog entspricht dem, der bei der Konfiguration der Alarmspeicherung verwendet wird. In der Aufzeichnungsdatei wird pro Messzeitpunkt eine Zeile mit den Namen und Werten aller aufgezeichneten Variablen angelegt. Jede Zeile beginnt mit einer eindeutigen Kennzeichnung im DWORD-Format, die aus dem Datum des Messzeitpunkts gebildet wird.

Variablenauswahl: Nach Drücken dieser Schaltfläche erhalten Sie den Dialog 'Variables', in dem die konfiguriert wird, für welche die Trend-Aufzeichnung durchführt werden soll und wie sie dort dargestellt werden sollen.

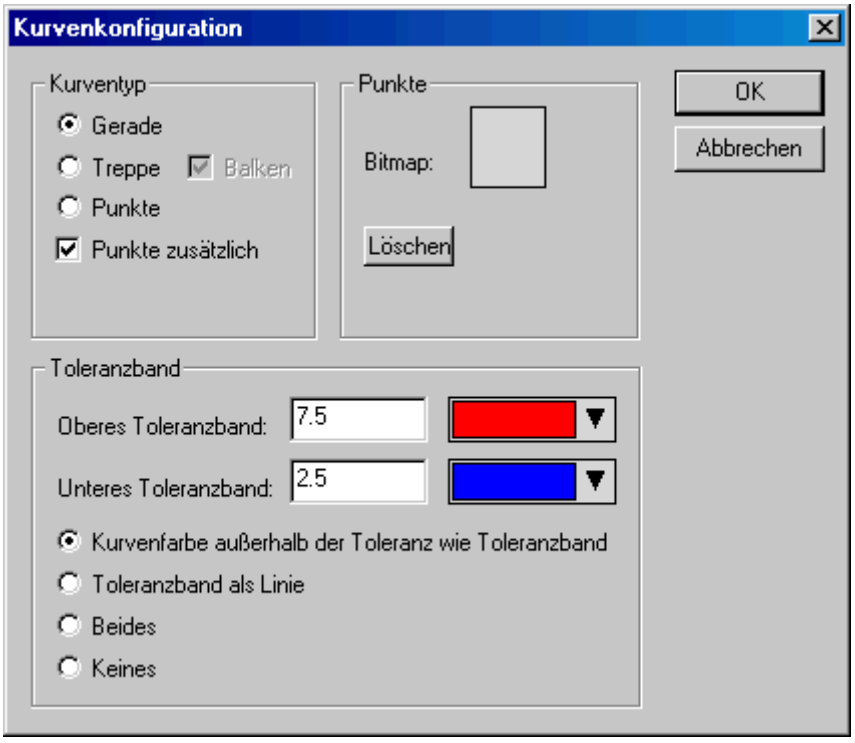


Dialog zur Variablenauswahl für das Trend-Element

Tragen Sie in der Spalte **Variable** (Mausklick auf Feld zum Öffnen eines Editierrahmens) eine Projektvariable ein (Eingabehilfe <F2> oder Intellisense-Funktion empfohlen). Farbe und Linienart für die Darstellung der Variable in der Aufzeichnung definieren Sie durch einen Mausklick auf das entsprechende Feld der Spalte Farbe (Standard-Farbauswahldialog) bzw. durch Auswahl eines Linientyps aus der Liste im entsprechenden Feld der Spalte Linienart (normal ___, dashed _ _ _, dotted ....., dashdotted _ . _ .). In der Spalte Lesezeichen kann eine Variable definiert werden, die dann bei Benutzung der Lesezeichenfunktion im Online Modus den jeweils aus der Aufzeichnung gelesenen Wert wiedergibt. Dazu wird im Online Modus ein kleines grau schattiertes Dreieck in der oberen linken Ecke des Aufzeichnungsdiagramms angezeigt. Wenn Sie dieses mit dem Cursor anwählen, können Sie bei gedrückter Maustaste eine senkrechte Linie horizontal zu den verschiedenen Aufzeichnungszeitpunkten bewegen. Die als 'Lesezeichen' definierte Variable wird dann den entsprechenden Wert aus der zugehörigen Aufzeichnungskurve auslesen.

Führen Sie dies für alle gewünschten Variablen durch. Über die Schaltfläche **Hinzufügen** wird jeweils eine weitere Zeile für einen Variableneintrag am Ende der Liste eingefügt. Eine in der Liste markierte Zeile kann über **Löschen** wieder entfernt werden.

Kurvenkonfiguration: Diese Schaltfläche öffnet den Dialog 'Configure curve'. Hier werden allgemeine Einstellungen zur Darstellung der Trendkurven vorgenommen:



Dialog zur Kurvenkonfiguration für das Trend-Element

Kurventyp: Wählen Sie eine der Optionen Gerade, Treppe oder Punkte. Sie können zu den ersten beiden Typen auch jeweils **Punkte** zusätzlich anzeigen lassen. Für die Darstellung eines Punktes kann ein Bitmap definiert werden, ansonsten wird ein gefülltes Rechteck in der Farbe der Kurve als Punktsymbol verwendet. Zur Auswahl klicken Sie auf das Rechteck neben **Bitmap** um den Standard-Dateiauswahldialog zu erhalten. Über **Löschen** kann das aktuell eingestellte Bitmap wieder aus der Konfiguration entfernt werden.

Toleranzband: Sie können als oberes oder/und unteres Toleranzband je einen Grenzwert auf der Skala definieren. Für jedes Band wird eine Farbe (Auswahldialog nach Drücken der Schaltfläche Farbe) festgelegt. Wenn die Bänder im Online Betrieb dargestellt werden sollen, aktivieren Sie die Option **Toleranzband** als Linie. Wenn Sie wollen, dass die Aufzeichnungskurve, sobald die Toleranzbänder überschritten werden, in der Farbe des jeweiligen Bandes dargestellt wird, aktivieren Sie die Option **Kurvenfarbe außerhalb der Toleranz wie Toleranzband**. Wählen Sie **Beides** oder **Keines**, um beide oder keine dieser genannten Anzeigeoptionen zu aktivieren bzw. zu deaktivieren.

Beispielprogramm zur Anzeige eines Trendelements im Online Modus:

Deklaration in Programm PLC_PRG:

```
VAR
    n: INT;
    rSinus:REAL;
    rValue:REAL;
    rSlider1:REAL; (*für Lesezeichenfunktion*)
    rSlider2:REAL; (*für Lesezeichenfunktion*)
END_VAR
```

Programmteil von PLC_PRG:

```
n:=n+1;
rValue := rValue + 0.01;
rSinus:=SIN(rValue)*50 + 50;
IF n>100 THEN
n:=0;
END_IF
```

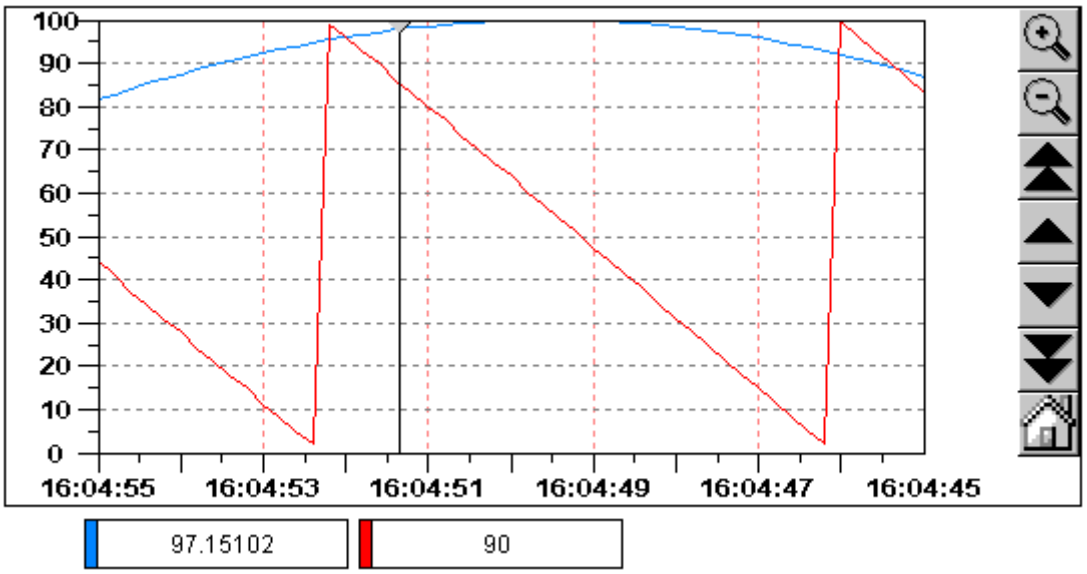
Konfiguration eines Trend-Elements in der Visualisierung wie folgt:

Ausrichtung Links-Rechts, Historie aktiviert
Horizontale Achse: Unterteilungslinien: T#2s, Dauer T#10s, Einteilung: T#1s, Unterteilung: T#500ms, Anzeigegenauigkeit: T#200ms, Beschriftung Zeit ('hh':'mm':'ss') mit der Einteilung von T#2s. Symbolleiste aktiviert.
Vertikale Achse: Unterteilungslinien: 10, gestrichelt, grau; Skala links, Start: 0, Ende: 100, Einteilung: 10, Untereinteilung: 5; Beschriftung Einteilung: 10; Symbolleiste aktiviert.
Variablenauswahl:
1. Variable PLC_PRG.rsinus, blaue Linie, Lesezeichen: PLC_PRG_TRD.rSlider1;
2. Variable PLC_PRG.n, rote Linie, Lesezeichen: PLC_PRG_TRD.rSlider2
Kurvenkonfiguration: Gerade, Kein Toleranzband

Konfiguration zweier Anzeigefelder für die aktuellen Kurvenwerte, die das Lesezeichen ausgibt:

Rechteckelement 1: Kategorie Text: bei Inhalt "%s" eingeben, Kategorie Variablen: bei Textausgabe: PLC_PRG.rSlider1 Rechteckelement 2: Kategorie Text: bei Inhalt "%s" eingeben, Kategorie Variablen: bei Textausgabe: PLC_PRG.rSlider2 (zusätzlich jeweils ein rein optisch konfiguriertes Rechteck-Element vor den Wertefeldern, das die Kurvenfarbe des angezeigten Wertes darstellt)

Ergebnis im Online Modus nach Einloggen und Starten des Programms:



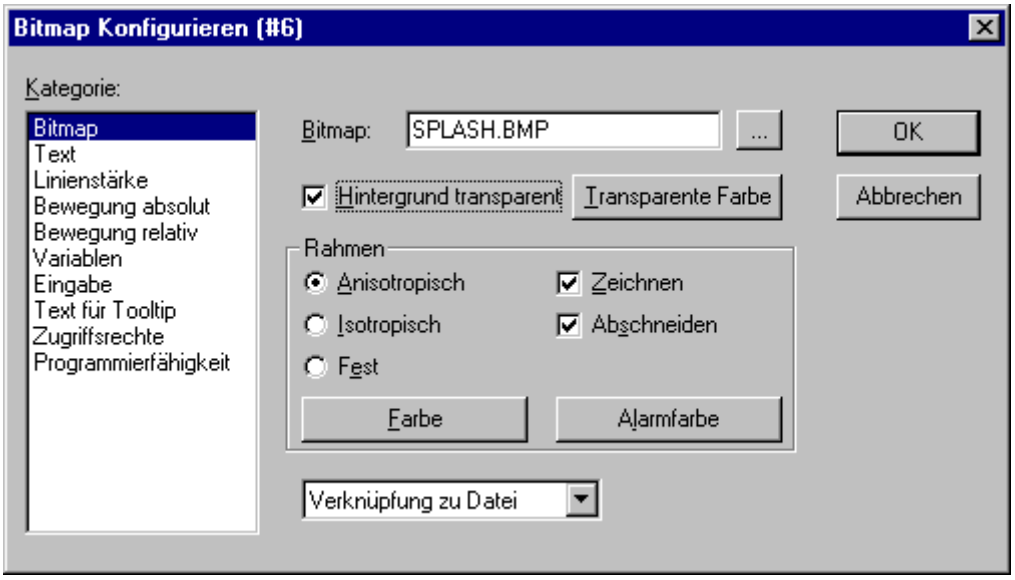
Die Aufzeichnung läuft von links nach rechts der neueste Wert jeweils links; alle 200 Millisekunden wird der aktuelle Wert jeder Kurve hinzugefügt. Die mit den Pfeil-Symbolen gekennzeichneten Schaltflächen erlauben ein Verschieben des dargestellten Wertebereichs. Wenn Sie beispielsweise mit der doppelpfeiligen Schaltfläche ganz an den Anfang des aufgezeichneten Wertebereichs rücken, erhalten Sie ein stehendes Bild der "damaligen" Werte. Wenn Sie dann das Lesezeichen (graues Dreieck links oben) über die Zeitachse bewegen, können Sie für den jeweiligen Zeitpunkt exakt die Werte der jeweiligen Variable in den Rechtecken unterhalb des Diagramms ablesen.

Bitmap

Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Bitmap** die Optionen zu einer Bitmap angeben. Im Feld **Bitmap** geben Sie die Bitmap-Datei mit ihrem Pfad an. Mit der Schaltfläche ... wird der Standarddialog von Windows zum Durchsuchen geöffnet und Sie können dort das gewünschte Bitmap auswählen.
Mit der Option **Hintergrund transparent,** kann eine im Bitmap enthaltene Farbe als transparent definiert werden. Über die Schaltfläche **Transparente Farbe** erhalten Sie dazu den Farbenauswahldialog zur Festlegung dieser Farbe. Wird sie exakt definiert, kann beispielsweise bei Grafiken mit unregelmäßiger Kontur die rechteckige Hintergrundfläche des Bitmaps unsichtbar gemacht werden.
Alle weiteren Angaben beziehen sich auf den **Rahmen** der Bitmap.

Bei der Auswahl **Anisotropisch, Isotropisch** und **Fest** geben Sie an, wie sich das Bitmap auf Größenänderung des Rahmens verhalten soll. **Anisotropisch** bedeutet, daß das Bitmap so groß wie sein Rahmen ist und so durch Größenänderung sich beliebig verzerren lässt. Bei **Isotropisch** bleiben die Proportionen der Bitmap auch bei Größenänderungen immer erhalten, d.h. das Verhältnis von Länge und Breite bleibt gewahrt. Wählen Sie **Fest**, so wird das Bitmap unabhängig von seinem Rahmen in der Originalgröße dargestellt.

Ist die Option **Abschneiden** gewählt, so wird bei der Einstellung **Fest** nur der Ausschnitt des Bitmaps angezeigt, der vom Rahmen umschlossen ist.
Wenn Sie die Option **Zeichnen** wählen, wird der Rahmen dargestellt, in der Farbe, die mit den Schaltflächen **Farbe** und **Alarmfarbe** in den Farbdialogen gewählt wurden. Die Alarmfarbe wird nur angezeigt, wenn die Variable TRUE ist, die in der Kategorie **Variablen** im Feld **Farbwechsel** angegeben ist.
In der Auswahlliste im unteren Teil des Dialogs können Sie festlegen, ob die Bitmap fest ins Projekt eingefügt werden soll (**Einfügen**), oder ob eine Verknüpfung zur Bitmap-Datei angelegt werden soll (**Verknüpfung zu Datei**), die außerhalb des Projekts im oben angegebenen Verzeichnis liegt. Es ist sinnvoll, die Bitmap-Datei im Projektverzeichnis zu halten, da dann der Pfad relativ eingetragen wird und bei einer eventuellen Weiterbearbeitung des Projekts in anderer Umgebung keine Probleme mit dem ansonsten absolut festgelegten Pfad entstehen.



Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Bitmap)

Visualisierung

Im Dialog zum Konfigurieren von Visualisierungselementen können Sie in der Kategorie **Visualisierung** die Eigenschaften einer eingefügten Visualisierung definieren. Nach dem Einfügen wird diese als **'Referenz'** der ursprünglichen bezeichnet.

Im Feld **Visualisierung** wird der Objektname der einzufügenden Visualisierung angegeben. Mit der Schaltfläche ... wird ein Dialog geöffnet, der die neben dem aktuellen zur Verfügung stehenden Visualisierungsbausteinen des Projektes zur Auswahl anbietet.

Dialog zum Konfigurieren von Visualisierungselementen (Kategorie Visualisierung)

Die folgenden Angaben beziehen sich auf den **Rahmen** der Visualisierung:

• Wenn Sie die Option **Zeichnen** wählen, wird der Rahmen dargestellt, in der Farbe, die mit den Schaltflächen **Farbe** und **Alarmfarbe** in den Farbdialogen gewählt wurden. Die Alarmfarbe wird nur angezeigt, wenn die Variable TRUE ist, die in der Kategorie **Variablen** im Feld **Farbwechsel** angegeben ist.
• Wenn Sie **Isotropisch** wählen, bleiben die Proportionen der Visualisierung auch bei Größenänderungen immer erhalten, d.h. das Verhältnis von Länge und Breite zu einander bleibt gewahrt. Andernfalls kann die Visualisierung auch verzerrt werden.
• Ist die Option **Abschneiden** gewählt, so wird im Online Modus nur der Originalausschnitt der Visualisierung angezeigt. Wandert z.B. ein Objekt außerhalb der ursprünglichen Anzeige, so wird dieses abgeschnitten und verschwindet möglicherweise ganz aus dem Blickfeld der Visualisierung.

Die Schaltfläche **Platzhalter** führt zum Dialog 'Platzhalter ersetzen'. Er listet in der Spalte 'Platzhalter' alle Platzhalter auf die bei der ursprünglichen Konfiguration der eingefügten Visualisierung ("Mutter-Visualisierung") eingetragen wurden und bietet in der Spalte 'Ersetzung' die Möglichkeit, diese für die vorliegende Visu-Referenz durch einen bestimmten Wert zu ersetzen. Welche Ersetzungen möglich sind, hängt davon ab, ob bei der "Mutter-Visualisierung" im Dialog 'Extras' 'Platzhalterliste' eine Wertemenge vordefiniert wurde. Ist dies der Fall, wird sie in einer Combo-Box zur Auswahl geboten. Wurde nichts vordefiniert, kann durch Doppelklick auf das entsprechende Feld in der Spalte Ersetzung ein Editierfeld geöffnet und beliebig ausgefüllt werden.

Eine weitere Möglichkeit, Platzhalter in Referenzen zu ersetzen, besteht direkt beim Aufruf einer Visualisierung im Konfigurationsdialog eines Visualisierungselements: Ein solcher Aufruf kann in der Kategorie 'Eingabe' im Optionsfeld **Zoomen nach Vis.** eingetragen werden.

Auf die zeitliche Reihenfolge der Ersetzung kann kein Einfluss genommen werden! Es sollten also keine Platzhalter durch Texte ersetzt werden, die ebenfalls Platzhalter beinhalten!

Durch die Verwendung von Platzhaltern ist es nicht mehr möglich, bereits beim Übersetzen des Projekts ungültige Eingaben in der Konfiguration der Visualisierungselemente zu überprüfen. Daher werden entsprechende Fehlermeldungen erst im Online Modus ausgegeben (... ungültiger Watch-ausdruck..).

HINWEIS

Online-Verhalten einer Referenz

Wenn Sie nach Einfügen einer Visualisierung diese Referenz markieren und konfigurieren, wird sie als 1 Objekt betrachtet und reagiert im Online Betrieb als solches entsprechend seiner Konfiguration auf Eingaben. Werden dagegen für die Referenz an eingefügter Stelle keine Konfigurationseinträge vorgenommen, dann reagieren ihre Einzelelemente online wie diejenigen der Ursprungsvisualisierung.

Beispiel für Einsatz des Platzhalterkonzepts:

Instanzen eines Funktionsblocks können auf einfache Weise mit Hilfe von Referenzen derselben Visualisierung dargestellt werden: Man könnte in der Konfiguration der Visualisierung 'visu', die die Variablen eines Funktionsblocks visualisiert, jeden Variableneintrag beispielsweise mit dem Platzhalter FUB beginnen (z.B: FUB.farbwechsel). Wird dann eine Referenz von 'visu' verwendet (durch Einfügen in einer anderen Visualisierung oder durch Aufruf über 'Zoom to Vis.'), kann in der Konfiguration dieser Referenz nun der Platzhalter FUB mit dem Namen der zu visualisierenden Instanz des Funktionsblocks ersetzt werden. Damit werden dann die entsprechenden Variablen der Funktionsblockinstanzen verwendet (z.B. inst1.farbwechsel etc.)

Dies könnte folgendermaßen aussehen:

```
Definieren Sie im Projekt einen Funktionsblock mit folgender Variablendeklaration:
FUNCTION_BLOCK fu
VAR_INPUT
farbwechsel:BOOL; (* soll Farbwechsel in Visu bewirken *`)
END_VAR
```

Deklarieren Sie im Projekt in PLC_PRG zwei Instanzen von 'fu':

```
inst1_fu:fu;
inst2_fu:fu;
```

Erstellen Sie ein Visualisierungsobjekt 'visu'. Fügen Sie ein Element ein und geben im Konfigurationsdialog der Kategorie 'Variablen' für den 'Farbwechsel' folgendes ein: "FUB.farbwechsel", in Kategorie 'Eingabe' für 'Variable tasten' "FUB.farbwechsel", in Kategorie 'Text' "FUB - Farbwechsel".

Erstellen Sie ein weiteres Visualisierungsobjekt: 'visu1'.
Fügen Sie die Visualisierung 'visu' zweimal in 'visu1' ein.
Markieren Sie die erste eingefügte Referenz von 'visu' und öffnen den Konfigurationsdialog der Kategorie Visualisierung. Drücken Sie die Schaltfläche 'Platzhalter', so dass die Platzhalterliste erscheint. Ersetzen Sie dort den Eintrag 'FUB' durch 'PLC_PRG.inst_1'.
Markieren Sie nun die zweite eingefügte Referenz von 'visu' und ersetzen Sie 'FUB' entsprechend mit 'PLC_PRG.inst_2'. Im Online Modus werden nun die Variablenwerte der beiden Instanzen von 'fu' in je einer

der beiden Referenzen von 'visu' visualisiert.
Der Platzhalter FUB kann natürlich an allen Stellen der Konfiguration von 'visu' verwendet werden, wo Variablen oder Textstrings eingetragen werden.

9.1.4.4 Spezielle Eingabemöglichkeiten für "Bedienversionen"

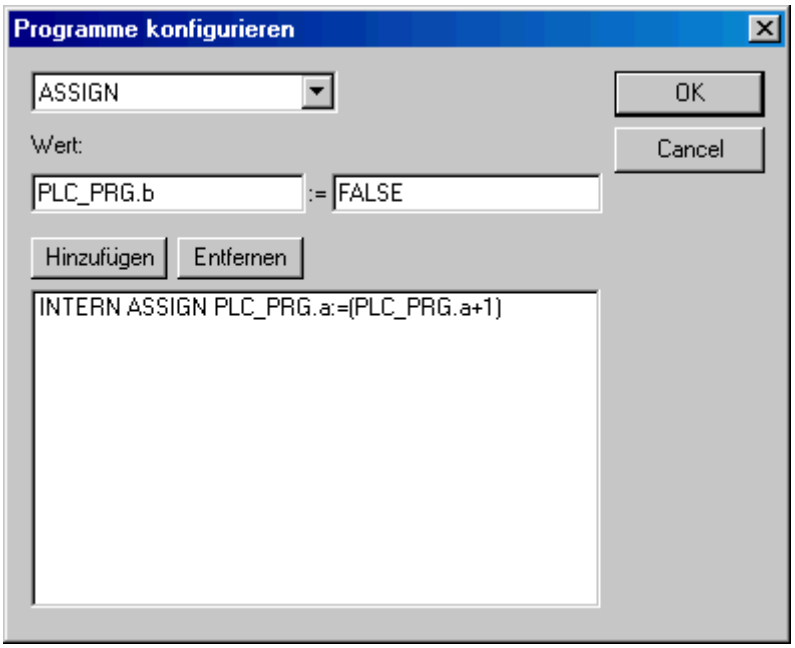
Die TwinCAT Visualisierung kann targetspezifisch über die TwinCAT PLC Visualisierung oder als Target-Visualisierung als reine Bedienoberfläche verwendet werden. In diesem Fall hat der Anwender keine Möglichkeit, das Steuerungsprogramm zu editieren, Menüs und Funktionsleisten stehen nicht zur Verfügung.

Die wesentlichen Steuer- und Überwachungsfunktionen in einem Projekt müssen also beim Erstellen eines für eine "Bedienversion" vorgesehenen Projektes auf Visualisierungselemente gelegt werden und damit im Online Modus bedient werden. Hierzu gibt es folgende spezielle Eingabemöglichkeiten im Konfigurationsdialog eines Visualisierungselements:

Geben Sie im Feld **Programm ausführen** in der Kategorie **Eingabe** interne Befehle nach folgender Syntax ein.

INTERN <BEFEHL>[PARAMETER]*

Wenn Sie die Schaltfläche ... drücken, erscheint dazu der Dialog **Programme konfigurieren** mit einer Auswahlliste:



Die nachfolgende Tabelle zeigt die verfügbaren internen Befehle. Sie erwarten zum Teil mehrere Parameter, die dann durch Leertasten getrennt eingegeben werden. Optionale Parameter sind hier durch eckige Klammern gekennzeichnet. Bei den Befehlen, die die Angabe einer Watchliste erfordern, kann anstelle des direkten Namens auch ein Platzhalter verwendet werden. Werden für ein Element mehrere Befehle eingetragen, werden diese durch Kommata getrennt.

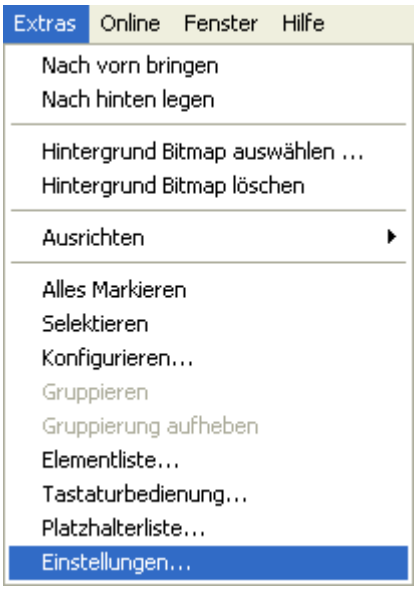
Voraussetzungen

INTERN Befehl	Entspricht in TwinCAT PLC Control	Erklärung
ASSIGN <Variable>:=<Expression>	Zuweisung	Einer Variable wird der Wert einer anderen bzw. ein Ausdruck zugewiesen. Beispiele: INTERN ASSIGN PLC_PRG.ivar1:=PROG1.ivar+12; INTERN ASSIGN PLC_PRG.bvar:=FALSE;
PROGRAM <Pfad ausführbares Programm> [Pfad aufzurufender Datei]	Programmaufruf	Das Programm wird ausgeführt. Beispiel: INTERN PROGRAM C:\programms\notepad.exe text.txt
LANGUAGEDIALOG	Einstellungen Visualisierung	Der Konfigurationsdialog für eine Visualisierung, der auch die Kategorie Sprache enthält, wird aufgerufen.
LANGUAGE <Sprachenangabe, wie in eingestellter Sprachdatei *.xml, *.vis, *.tlt oder *.txt verwendet>	Einstellungen Visualisierung, Sprache	Die gewünschte Sprache wird eingestellt, ohne dass sich der Einstellungsdialog für die Visualisierung öffnet. Sehen Sie Sprachumschaltung in der Visualisierung [▶ 251]!
LANGUAGE DEFAULT	Einstellungen Visualisierung, Sprache	Für dynamische Texte wird die Default-Sprache, die in der aktuell angezogenen xml-Datei [▶ 256] definiert ist, verwendet.
CHANGEUSERLEVEL		Ein Dialog zum Einstellen des Benutzerlevels (Arbeitsgruppe) erscheint. Es stehen die in TwinCAT PLC Control vorgesehenen acht Gruppen zur Verfügung
CHANGEPASSWORD	vgl. 'Projekt' 'Passwörter für Arbeitsgruppen'	Ein Dialog zum Ändern des Arbeitsgruppen-Passworts erscheint.
TRACE[1]	Ressourcen, Traceaufzeichnung	Das Fenster zur Traceaufzeichnung wird geöffnet (siehe Ressourcen, Traceaufzeichnung). Die in TwinCAT PLC Control zugehörigen Menübefehle Trace **Starten**, **Lesen**, **Stoppen**, **Speichern**, **Laden** sind in diesem Fenster verfügbar.
SAVEPROJECT[1]	‚Datei' 'Speichern'	Das Projekt wird gespeichert.
EXITPROGRAM[1]	‚Datei' 'Schließen'	Das Programm wird beendet.
PRINT[1]	‚Datei' 'Drucken'	Die aktuelle Visualisierung wird online ausgedruckt.

[1] Wird nicht von der Target-Visualisierung unterstützt.

9.1.4.5 Visualisierungsobjekt konfigurieren

Neben der Konfiguration der einzelnen grafischen Elemente einer Visualisierung kann diese auch als ganzes Objekt mit bestimmten Parametern versehen werden. Dies betrifft die Darstellung von Rahmen, Sprache, Raster, Hintergrund etc. wie auch die Zuordnung expliziter Tastenbelegungen, die für genau ein Visualisierungsobjekt gelten sollen.



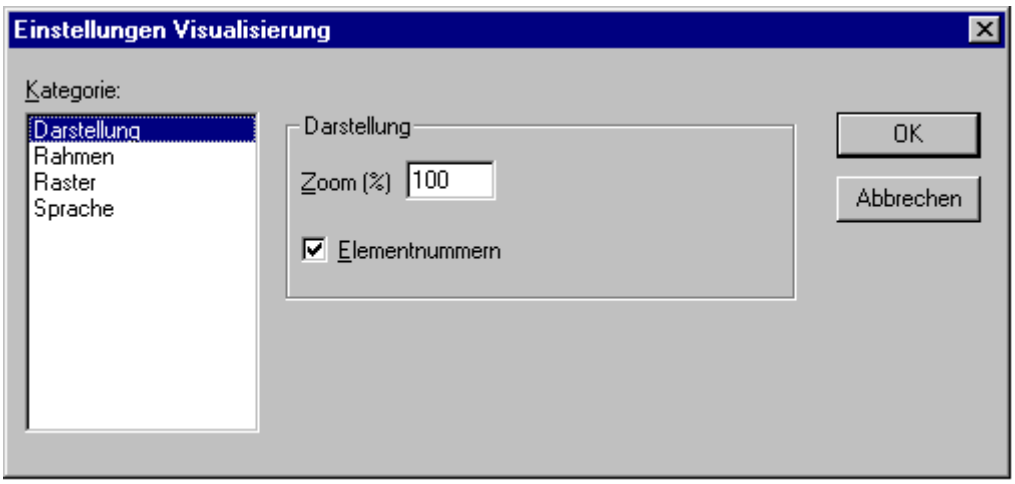
'Extras' 'Einstellungen' / Darstellung, Rahmen, Raster, Sprache

Nach diesem Befehl können Sie im Dialog 'Einstellungen Visualisierung' die optische und sprachliche Darstellung der Visualisierung beeinflussen:

Die Kategorien **Darstellung, Rahmen** und **Sprache** können auch im Online Modus bearbeitet werden.

Kategorie Darstellung:

Geben Sie im Feld Zoom eine Vergrößerung zwischen 10 und 500 % an, um die Anzeige der Visualisierung zu vergrößern bzw. zu verkleinern.



Einstellungsdialog von Visualisierungen (Kategorie Darstellung)

Kategorie Rahmen:

Mit **Auto-Scrolling** erreicht man, dass beim Zeichnen oder Verschieben eines Visualisierungselements der sichtbaren Bereich des Visualisierungsfensters automatisch verschoben wird, wenn man an den Fensterrahmen stößt. Die gesamte Visualisierung mit allen Elementen wird im Online Modus im Fenster dargestellt, egal wie groß das Fenster ist, wenn **Online automatisch anpassen** gewählt ist. Ist **Hintergrund Bitmap einbeziehen** angewählt, so wird beim Anpassen das Bitmap in die Berechnung einbezogen, ansonsten werden nur die Elemente bedacht.

Kategorie Raster:

Hier können Sie festlegen, ob die Rasterpunkte, die als Zeichenhilfe dienen, im Offline Modus **Sichtbar** sind, wobei der Abstand der sichtbaren Punkte mindestens 10 beträgt, auch wenn die angegebene Größe kleiner ist. In diesem Fall erscheinen nur die Rasterpunkte im Abstand eines Vielfachen der angegebenen Größe. Ist der Punkt Aktiv gewählt, so werden die Elemente beim Zeichnen und Verschieben auf die Rasterpunkte gelegt. Im Feld **Größe** wird der Abstand der Rasterpunkte angegeben.

Kategorie Sprache:

Hier können Sie festlegen, in welcher Landessprache der Text, den Sie im Konfigurationsdialog bei den Optionen Text und Text für Tooltip einem Element zugeordnet haben, angezeigt werden soll. Statisch kann dies über die Verwendung einer **Sprachdatei** erfolgen. Die Funktion **'Dynamische Texte'** bietet alternativ die Möglichkeit, den Inhalt einer Textanzeige abhängig von einer Projektvariable wechseln zu lassen. Siehe Kapitel Sprachumschaltung

Die Textanzeige wechselt nur im Online Modus !

'Extras' 'Hintergrund Bitmap auswählen'

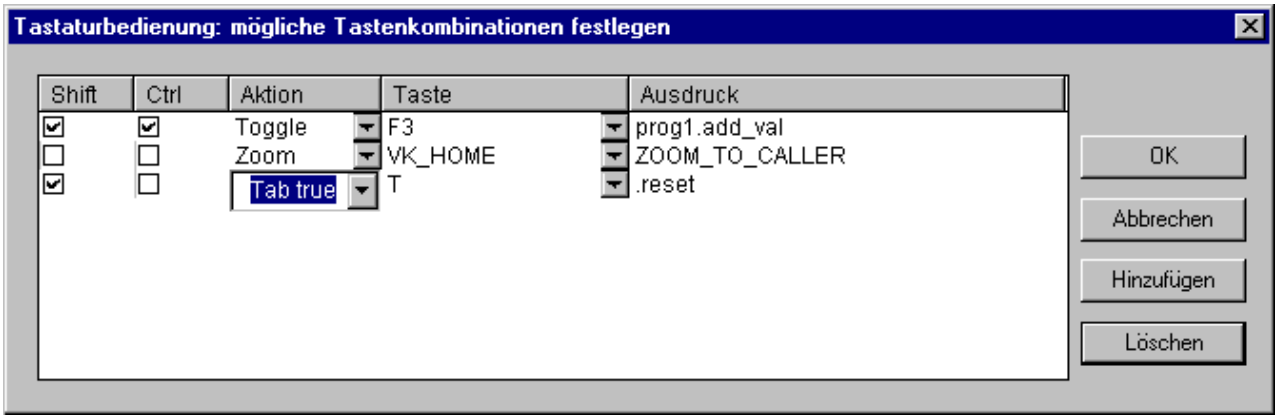
Mit diesem Befehl öffnen Sie den Dialog zur Auswahl von Dateien. Wählen Sie eine Datei mit dem Zusatz "*.bmp". Die ausgewählte Bitmap erscheint im Hintergrund Ihrer Visualisierung.
Mit dem Befehl **'Extras' 'Hintergrund Bitmap löschen'** können Sie die Bitmap wieder löschen.

'Extras' 'Hintergrund Bitmap löschen'

Mit diesem Befehl löschen Sie die Bitmap im Hintergrund der aktuellen Visualisierung.
Mit dem Befehl **'Extras' 'Hintergrund Bitmap auswählen'** können Sie ein Bitmap für die aktuell Visualisierung wählen.

'Extras' 'Tastaturbedienung'

Es ist möglich, während der Konfiguration für einzelne Visualisierungen Tasten(kombinationen) festzulegen, die genauso mit Aktionen verknüpft werden können wie einzelne Visualisierungselemente. Beispielsweise könnte in einer Visualisierung konfiguriert werden, daß die Tastenkombination <Strg><F2> im Online Modus dieselbe Aktion auslöst wie das Drücken (per Maus oder über Touchscreen) auf ein bestimmtes Visualisierungselement.
Solche Tastenbelegungen können die reine Tastaturbedienung optimieren.
Bereits standardmäßig sind die Tasten **<Tabulator>**, **<Leertaste>**, und **<Eingabetaste>** so implementiert, dass über sie im Online Modus jedes Element innerhalb einer Visualisierung erreicht und aktiviert werden kann. Der Dialog 'Tastaturbedienung' wird aus dem Menü 'Extras' oder dem Kontextmenü aufgerufen:



Dialog 'Tastaturbedienung: mögliche Tastenkombinationen festlegen'

In der Spalte Taste bietet eine Auswahlliste die folgenden Tasten zur Belegung:

Tastenkürzel	Bedeutung
VK_TAB	Tabulatortaste
VK_RETURN	Eingabetaste
VK_SPACE	Leertaste
VK_ESCAPE	Esc-Taste
VK_INSERT	Einfg.-Taste
VK_DELETE	Entf.-Taste
VK_HOME	Pos1-Taste
VK_END	Ende-Taste
VK_PRIOR	Bild ()-Taste
VK_NEXT	Bild ()-Taste
VK_LEFT	Pfeiltaste (¬)
VK_RIGHT	Pfeiltaste (®)
VK_UP	Pfeiltaste ()
VK_DOWN	Pfeiltaste ()
VK_F1-VK_F12	Funktionstasten F1 bis F12
0-9	Tasten 0 bis 9
A-Z	Tasten A bis Z
VK_NUMPAD0 - VK_NUMPAD9	Tasten 0 bis 9 des numerischen Tastaturfeldes
VK_MULTIPLY	Taste ´ des numerischen Tastaturfeldes
VK_ADD	Taste + des numerischen Tastaturfeldes
VK_SUBTRACT	Taste – des numerischen Tastaturfeldes
VK_DIVIDE	Taste ¸ des numerischen Tastaturfeldes

In den Spalten **Shift** und **Ctrl** kann über Anklicken der Kontrollkästchen die <Umschalt>- und/oder die <Strg>-Taste zur Tastenkombination mit der gewählten Taste hinzugefügt werden.
In der Spalte **Aktion** wird eingetragen, was beim Drücken der Taste(nkombination) ausgelöst werden soll. Möglich sind die in der untenstehenden Tabelle genannten Funktionen, die in einer Auswahlliste angeboten werden. Sie entsprechen denen, die im Konfigurationsdialog der Kategorie Eingabe verfügbar sind.
In die Spalte **Ausdruck** muss abhängig von der ausgewählten Aktion entweder ein Variablenname, ein INTERN-Befehl, Visualisierungsname oder eine Elementnummer eingetragen werden; genauso wie dies im Konfigurationsdialog der Kategorie 'Eingabe' für das Visualisierungselement getan würde. Sehen Sie im folgenden die Bedeutung der möglichen Aktionen und die jeweils einzugebenden Ausdrücke:

Aktion	Bedeutung	Ausdruck
Toggln	Variable Toggeln	Variable, z.B. "plc_prg.tvar"
Tap true	Variable Tasten (auf TRUE setzen)	Programmvariable, z.B. "plc_prg.svar"
Tap false	Variable Tasten (auf FALSE setzen)	Programmvariable, z.B. "plc_prg.xvar"
Zoom	Zoomen nach Vis.	Name des Visualisierungsbausteins, zu dem gesprungen werden soll, z.B. "Visu1"
Exec	Programm ausführen	Name der ausführbaren Datei, z.B. "notepad C:\help.txt" (Notepad startet und öffnet Datei help.txt)
Text	Texteingabemöglichkeit für die Variable, die in Kategorie 'Variable' bei Textausgabe angegeben ist	Nummer des Elements, für das die Texteingabe konfiguriert werden soll, z.B. "#2" (Elementnummeranzeige kann im Dialog 'Extras' 'Einstellungen' eingeschaltet werden, außerdem siehe Elementliste)

Über die Schaltfläche **Hinzufügen** wird eine weitere, leere Zeile am Tabellenende hinzugefügt. Über **Löschen** wird die Zeile, in der der Cursor steht, gelöscht. Mit **OK** bzw. **Abbrechen** werden die Eingaben gespeichert bzw. nicht gespeichert und der Dialog geschlossen.

Die Tastaturbedienung kann getrennt für jede Visualisierung konfiguriert werden. Somit kann dieselbe Taste(nkombination) in unterschiedlichen Visualisierungen unterschiedliche Aktionen auslösen.

Beispiel:

Die folgenden Tasten-Konfigurationen wurden für die Visualisierungen VIS_1 bzw. VIS_2 vorgenommen:

VIS_1:				
Shift	Ctrl	Aktion	Taste	Ausdruck
ja		Toggle	A	PLC_PRG.automatic
	ja	Zoom	Z	VIS_2

VIS_2:				
Shift	Ctrl	Aktion	Taste	Ausdruck
		Exec	E	INTERN LANGUAGE DEUTSCH
	ja	Zoom	Z	TC_VISU

Wenn nun VIS_1 im Online Modus den Eingabefokus hat, bewirkt ein Drücken der Tastenkombination <Shift><A>, daß die Variable PLC_PRG.automatic getoggelt wird. <Ctrl><Z> bewirkt, dass aus Visu1 nach VIS_2 gewechselt wird.

Wenn VIS_2 das aktive Fenster ist und die Taste <E> gedrückt wird, wird die Sprache in der Visualisierung auf Deutsch gestellt. <Ctrl><Z> bewirkt hier, dass zur Visualisierung TC_VISU gewechselt wird.

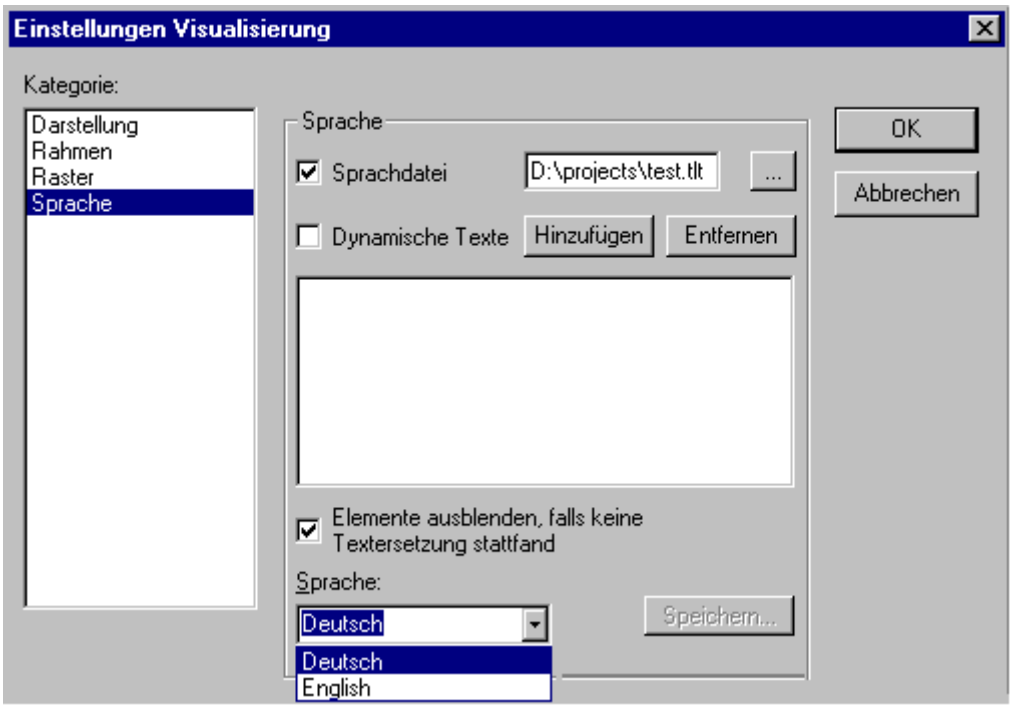
9.2 Sprachumschaltung

Die Sprachumschaltung für Texte in der Visualisierung kann über statische oder dynamische Texte erfolgen, die in Sprachdateien bereitgestellt werden müssen. Unicode-Format ist nur innerhalb der dynamischen Texte möglich!

Wie wird eine Sprachumschaltung erreicht:

Im Konfigurationsdialog ‚Einstellungen Visualisierung'

im Auswahlfenster unter **Sprache** können Sie aus den in der aktuell eingestellten Sprachdatei definierten Sprachen diejenige wählen, die im Online-Betrieb als **Startsprache** verwendet werden soll, für das unten gezeigte Beispiel Deutsch und English.



Eine **Sprachumschaltung** im **Online-Betrieb** wird über ein Visualisierungselement vorgenommen. Zur entsprechenden Konfiguration eines Elements stehen Ihnen u.a. die internen Kommandos "INTERN LANGUAGE <Sprache>" und "INTERN LANGUAGEDIALOG" zur Verfügung, die im Konfigurationsdialog in Kategorie ‚Eingabe' verwendet werden können (siehe Spezielle Eingabemöglichkeiten für "Bedienversionen" [▶ 245]).

Beispiel:

Sie fügen ein Schaltflächen-Element ein, mit dem die Visualisierungstexte auf Deutsch geschaltet werden sollen. Beschriften Sie das Element mit 'German', aktivieren Sie im Konfigurationsdialog bei 'Eingabe' die Option 'Programm ausführen' und definieren einen Befehl "INTERN LANGUAGE <sprache>". Die Sprache geben Sie dabei mit dem in der Sprachdatei verwendeten Kürzel an, also im Falle des nachfolgend gezeigten Beispiels der vis-Datei: "INTERN LANGUAGE german". Wenn die Schaltfläche nun im Online Modus bedient wird, werden die Visualisierungstexte gemäß der für „german" vorliegenden Einträge in der Sprachdatei in Deutsch dargestellt.

9.2.1 Statisch

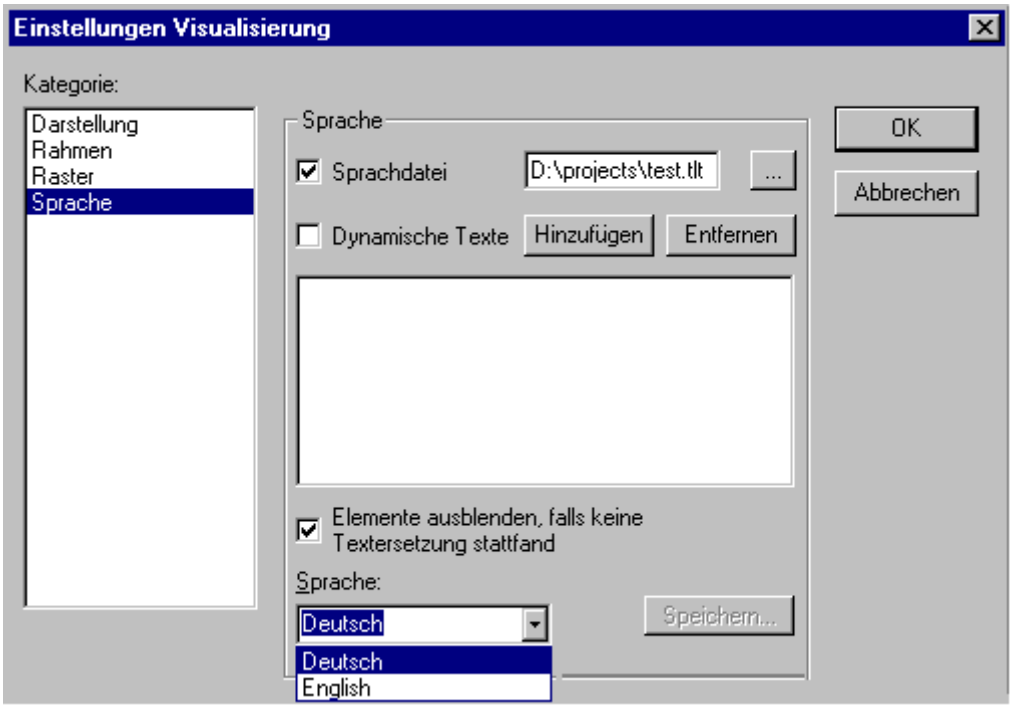
9.2.1.1 Statische Sprachumschaltung

Zum statischen Umschalten zwischen verschiedenen Sprachen wird eine Sprachdatei *.vis, *.tlt oder *.txt verwendet (zur Erstellung siehe weiter unten). Der Unterschied zur dynamischen Sprachumschaltung besteht darin, dass die Sprache nicht über eine Projektvariable zur Laufzeit vorgegeben werden kann.

HINWEIS
Für Visualisierungen empfiehlt sich generell die Verwendung einer ***.vis**-Sprachdatei, da *.tlt- bzw. *.txt.-Übersetzungsdateien nur für Visualisierungen in TwinCAT PLC Control und auch dort nicht für die Visualisierungselemente Zeigerinstrument, Balkenanzeige und Histogramm funktionieren.

Im Dialog ‚Einstellungen Visualisierung', wird konfiguriert, welche Sprachdatei verwendet werden soll: Um eine Übersetzungs-(*.tlt, *.txt) oder eine reine Visualisierungs-Sprachdatei (*.vis) auszuwählen, die die Texte in den verschiedenen Sprachen enthält, aktivieren Sie im die Option **Sprachdatei** und geben Sie im Eingabefeld daneben den entsprechenden Dateipfad ein. Sie können auch über die Schaltfläche den Standarddialog  zum Öffnen einer Datei zur Hilfe nehmen.

Auswahl Sprachdatei für eine Visualisierung (Kategorie Sprache)



Zur Erstellung einer **Übersetzungsdatei *.tlt** oder ***.txt** siehe ' Abschnitt 'Projekt' [▶ 67] 'In andere Sprache übersetzen [▶ 78]'.

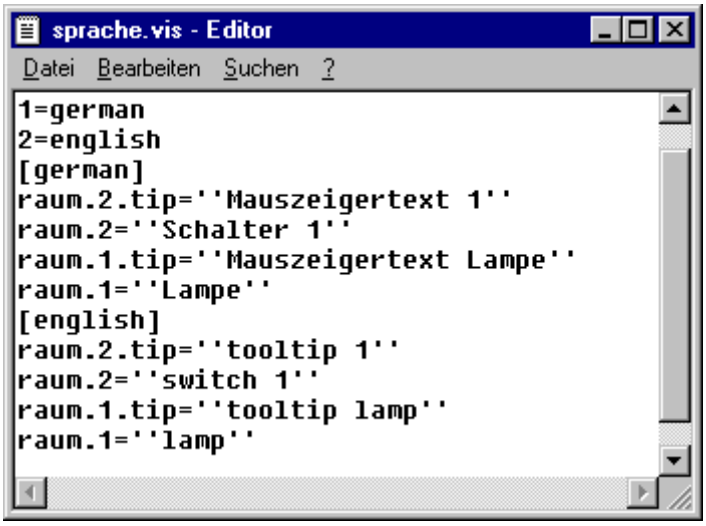
Zur Erstellung einer speziellen **Sprachdatei *.vis** gehen Sie folgendermaßen vor:

Öffnen Sie ebenfalls den Dialog Einstellungen Visualisierung, Kategorie Sprache. Wählen Sie die Option **Sprachdatei**. Im zugehörigen Eingabefeld geben Sie an, wo die zu erstellende Datei gespeichert werden soll. Sie erhält den Zusatz ‚.vis'. Sie können dazu auch über die Schaltfläche  den Dialog **Öffnen** zu Hilfe nehmen. Falls bereits eine Sprachdatei mit der Erweiterung .vis vorliegt, wird sie Ihnen hier angeboten.

Im Eingabefeld unter **Sprache** geben Sie ein Kennwort für die aktuell in der Visualisierung verwendete Sprache ein, z.B. „german" (oder „D" oder „deutsch"). Drücken Sie dann auf **Speichern**. Sie erzeugen eine Datei mit der Erweiterung .vis, die Sie nun mit einem normalen Texteditor bearbeiten können. Öffnen Sie dazu die Datei beispielsweise mit notepad:

Sie erhalten eine Liste der Textvariablen für die aktuelle Sprache, die unter dem Titel [Sprachen] mit z.B. „1=german" eine Referenz auf Ihren Titel [german] enthält. Sie können nun die Liste durch eine Kopie der Variablenzeilen erweitern, in der Sie dann die deutschen durch englische Texte ersetzen und darüber ein „[english]" setzen. Unter der vorhandenen Zeile „1=german" ergänzen Sie entsprechend mit „2=english".

Beispiel einer Sprachdatei für eine Visualisierung (Kategorie Sprache)



9.2.2 Dynamisch

9.2.2.1 Dynamische Sprachumschaltung

Dynamische Texte erlauben ein Umschalten zwischen unterschiedlichen, jeweils einer Landessprache zugeordneten Texten für ein Visualisierungselement. Der Unterschied zu statischen Texten besteht darin, dass die konkrete Textauswahl auch über eine Programmvariable erfolgen kann.

In der Konfiguration des Elements wird eine Prefix-ID-Kombination eingegeben, der in einer XML-Datei ein Text zugeordnet ist. Dabei ist die ID über eine Projektvariable veränderbar.

Anwendungsbeispiel: Die ID repräsentiert eine Fehlernummer, als Prefix wird beispielsweise „Error“ verwendet. Die Sprachdatei liefert über die entsprechende Prefix-ID-Kombination eine zugehörige Fehlermeldung, die – abhängig von der eingestellten Landessprache - in dieser Sprache in der Visualisierung angezeigt wird.

Zu beachten:

- Die Sprachdateien für dynamische Texte können in Unicode (UTF-16) oder ANSI (ISO-8859-1) erstellt werden. Dies ist über die encoding-Syntax vor dem Header der xml-Datei anzugeben.
- Für die Target-Visualisierung können die Start-Sprache, das Verzeichnis der zu verwendenden XML-Textlisten und eine Liste von Textlisten über die Konfigurationsdatei des Zielsystems definiert sein, was erlaubt, diese Parameter nachträglich zu ändern, ohne ein neues Bootprojekt erzeugen zu müssen. Somit können bestehende Textlisten einfach nachträglich geändert (Startsprache, Texte) werden bzw. neue Landessprachen ergänzt werden. Wenn das Zielsystem eine solche Konfiguration bereitstellt, werden die Textlisten, die in TwinCAT PLC Control für die Visualisierung definiert sind, im Online-Betrieb nicht berücksichtigt! Gibt es keine zielsystemspezifische Konfiguration für die Sprachumschaltung, ist nach einer Änderung der in TwinCAT PLC Control definierten Textlisten ein erneuter Download des Projekts erforderlich.

9.2.2.2 Konfiguration

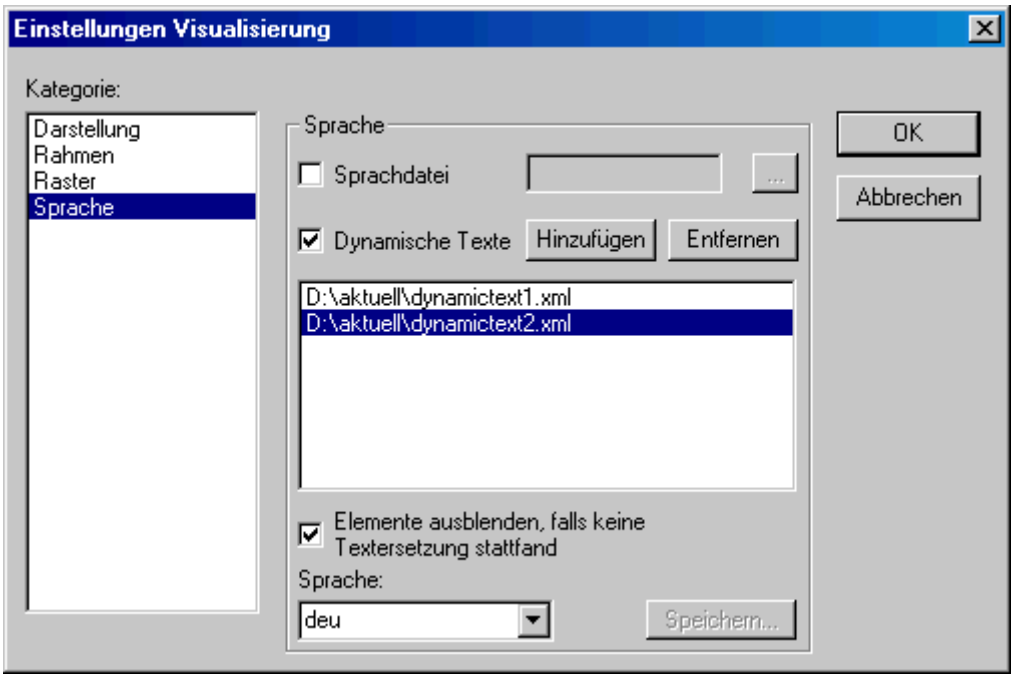
Welcher Text in einem Visualisierungselement im Online Modus erscheinen soll, kann dynamisch mit Hilfe einer Prefix-ID-Kombination gesteuert werden, der in einer XML-Datei (siehe "XML Datei für dynamische Texte) [▶ 256] ein Text zugeordnet ist. Prefix und ID werden in der Konfiguration des Visualisierungselementes definiert, wobei die ID dynamisch über eine Projektvariable gegeben werden kann. Eine Default-Sprache kann über einen INTERN Befehl definiert werden.
Die XML-Datei, die die Textzuordnung beschreibt, wird - ebenfalls in der Konfiguration des Visualisierungselements - mit dem Projekt verknüpft. Eine solche XML-Datei muss nach einem vorgegebenen Schema erstellt worden sein. In dieser Datei werden die Textversionen mit

Landessprachenkürzeln markiert, so dass nicht nur zwischen unterschiedlichen Inhalten für einen Visualisierungstext umgeschaltet werden kann, sondern auch eine Sprachumschaltung möglich ist. Diese erfolgt wie oben für die Sprachdatei beschrieben über den Auswahldialog im Dialog 'Einstellungen' / Sprache.

Folgende Einträge in den Konfigurationsdialogen sind also nötig:

1. Eintragen der XML-Textdateien: Dialog 'Einstellungen' Kategorie Sprache:

Aktivieren Sie die Option **Dynamische Texte** und drücken Sie die Schaltfläche **Hinzufügen**, um eine oder mehrere vorliegende XML-Datei ins Projekt einzubinden. Die ausgewählten Dateien werden im Fenster unterhalb der Schaltfläche aufgelistet. Über **Entfernen** können Sie eine aktuell markierte Datei wieder löschen. Eventuell ist es wünschenswert, nur die Visualisierungselemente anzuzeigen, für die eine Textersetzung vorgenommen wird, dann aktivieren Sie die Option **Elemente ausblenden, falls keine Textersetzung stattfand.**
Die Sprachauswahl im Feld bei **Sprache** wirkt sich wie beim Verwenden einer Sprachdatei auf die dynamischen Texte aus, die mit der entsprechenden Sprachbezeichnung in der XML-Datei definiert sind.



Konfigurationsdialog Einstellungen / Sprache für dynamische Texte

2. Definieren der ID (wie in der XML-Datei verwendet) im Konfigurationsdialog der Kategorie 'Variablen' im Feld 'Textausgabe':

Eintrag eines Wertes bzw. einer Projektvariable.

3. Definieren des Textformats im Konfigurationsdialog, Kategorie Text:

Geben Sie im Feld 'Inhalt' im Text an der Stelle, an der sie die Ersetzung durch einen dynamischen Text wünschen, folgenden Platzhalter ein: "%<PREFIX>". Anstelle von "PREFIX" kann eine Buchstabenfolge eingetragen werden, die mit einer in der XML-Datei vorliegenden PREFIX-Definition übereinstimmen muss.

Für jede ID-Prefix-Kombination, die in einer eingetragenen XML-Textdatei gefunden wird, wird dann der entsprechende Text im Online Modus im Visualisierungselement dargestellt. Wird kein Eintrag gefunden, so findet keine Ersetzung statt.

9.2.2.3 XML Datei für dynamische Texte

Zur Verwendung von dynamischen Texten sehen Sie die Beschreibung für den Dialog 'Einstellungen', Kategorie Sprache. Die zugrunde liegende Datei muss im XML-Format vorliegen (<dateiname>.xml). Sie enthält eine Zuordnung von Texten zu Prefix-ID-Kombinationen, die im Visualisierungselement angegeben werden können. Außerdem können in einer Header-Section eine Default-Sprache und ein einer Sprache zugeordneter Default-Font definiert werden. Die Beschreibungen in der XML-Datei werden von den Tags <dynamic_text> und <\dynamic_text> am Beginn und Ende der Datei eingeschlossen.

Die Sprachdateien für dynamische Texte können in **Unicode** (UTF-16) oder **ANSI** (ISO-8859-1) erstellt werden. Dies ist über die encoding-Syntax vor dem Header der xml-Datei anzugeben (siehe unten, Datei-Beispiel).

Ursprüngliche Formate der XML-Datei, die noch kein <dynamic_text>-Tag und keine Header-Sektion verwenden, werden auch weiterhin unterstützt.

Die Target-Visualisierung bietet eine Schnittstelle zum Abfragen der Einträge dynamischer Textlisten. Somit können diese dann direkt im Programm verwendet werden.

Die **Header-Section** beginnt mit **<header>** und wird mit **<\header>** abgeschlossen. Soll eine **Default-Sprache** festgelegt werden, fügen Sie hier den Eintrag **<default-language>** ein. Ein einer Sprache zugehöriger Default-Font wird über den Eintrag <default-font> festgelegt. Die Einträge sind optional, wenn sie fehlen, wird der dynamische Text in der Visualisierung in der/dem lokal konfigurierten Sprache bzw. Font angezeigt.

<header>	
<default-language><Sprachkürzel> </default-language>	Angabe der Default-Sprache; das heißt, wenn für einen Texteintrag (s.u.) kein Text in der eingestellten Sprache vorhanden ist, wird der Text verwendet, der im gleichen Texteintrag unter dem Kürzel der Default-Sprache definiert ist. Wenn auch ein solcher nicht vorliegt, wird "<PREFIX> <ID>“ ausgegeben. Werden mehrere XML-Dateien verwendet, die jeweils eine Header-Section enthalten, wird diejenige berücksichtigt, die zuletzt gelesen wird. Sinnvoll ist es, nur 1 Header-Section zu verwenden ! Das Sprachkürzel sollte einem der beiden Texteinträgen verwendeten entsprechen (s.u.). Anmerkung: Die Default-Sprache kann zur Laufzeit gezielt mit Hilfe eines Visualisierungselements eingestellt werden, das mit dem Befehl INTERN LANGUAGE DEFAULT in Kategorie Eingabe, Programm ausführen konfiguriert wurde (siehe Kap. 2.4.1, Spez. Eingabemöglichkeiten für Bedienversionen).
<default-font>	Angabe des Default-Fonts für eine bei <language> angegebene Sprache.: Der mit der
<language><Sprachkürzel> </language>	
<font-name><Font-Bezeichnung></font-name>	Font-Bezeichnung angegebene Font (z.B. "Arial" wird automatisch für alle Elemente
<default-font>	verwendet, welche dynamische Texte in dieser Sprache ausgeben. Das Sprachkürzel sollte einem der beiden Texteinträgen verwendeten entsprechen (s.u.)
</default-font>	weitere Default-Fonts für andere Sprachen
<language>...	
...	
<default-font>	
...	
</header>	

Die **Liste der Zuordnungen** zwischen Prefix-ID und Texten wird mit **<text-list>** eröffnet und mit **<\text-list>** abgeschlossen. Die einzelnen Zuordnungen beginnen jeweils mit **<text prefix>** und enden mit **<\text>**.

Ein Texteintrag für eine **Prefix-ID-Kombination** enthält folgende Zeilen:

<text prefix>= "<PREFIX> id="<ID>"	" Das hier verwendete "PREFIX" entspricht dem im Visualisierungselement (Konfigurationskategorie Text) verwendeten; Die "ID" entspricht dem Eintrag in Konfigurationskategorie 'Variablen', Textausgabe
<Sprachkürzel> <!CDATA[<TEXT>] </Sprachkürzel>	Das frei wählbare Sprachkürzel wird im Dialog 'Einstellungen', Kategorie Sprache des Visualisierungselements dann in der Auswahlliste bei 'Sprache' wiedergegeben; der "TEXT" ist der, der im Online Modus anstelle der oben definierten ID-PREFIX-Kombination im Visualisierungselement angezeigt wird.
</text>	

Zumindest in 1 Landessprache muss pro Prefix-ID-Kombination ein Texteintrag vorliegen. Sehen Sie z.B. im unten gezeigten Dateibeispiel: <deutsch> startet den Text in deutscher Sprache, </deutsch> beendet ihn.

Die dynamischen Texte können zum einen zur Darstellung eines Textes in verschiedenen Sprachen dienen, aber natürlich auch, um verschiedene Inhalte dynamisch anzuzeigen.

Beispiel:

Sie verwenden zwei Visualisierungselemente, eins für die Anzeige einer Maschinenbezeichnung, das andere für die Anzeige eines Fehlertextes zu einer bestimmten Fehlernummer:
(1) Definieren Sie in PLC_PRG die Variablen *ivar* vom Typ INT, die die Maschinenbezeichnung festlegt und *errnum* vom Typ INT, die die Fehlernummer liefert.
(2) Konfigurieren Sie ein Visualisierungselement zur Anzeige der aktuellen Maschinenbezeichnung:
a. Tragen Sie in Kategorie Text im Textfeld ein: "%<Maschine>"
b. Tragen Sie in Kategorie Variablen bei Textausgabe ein: PLC_PRG.ivar
(3) Konfigurieren Sie ein weiteres Visualisierungselement zur Ausgabe des aktuellen Fehlertextes:
a. Tragen Sie in Kategorie Text im Textfeld ein: "%<Error>"
b. Tragen Sie in Kategorie Variablen bei Textausgabe ein: "PLC_PRG.errnum“
(4) Erstellen Sie eine xml-Datei, z.B. mit Namen dynamictextsample.xml, gemäß der oben beschriebenen Syntax, die für dieses Beispiel folgendermaßen aussieht:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<dynamic-text>
<header>
    <default-language>deutsch</default-language>
    <default-font>
        <language>deutsch</language>
        <font-name> Arial </font-name>
        <font-color>0,0,0</font-color>
        <font-height>-13</font-height>
        <font-weight>700</font-weight>
        <font-italic>false</font-italic>
        <font-underline>false</font-underline>
        <font-strike-out>false</font-strike-out>
        <font-char-set>0</font-char-set>
    </default-font>
    <default-font>
        <language>english</language>
        <font-name> Arial </font-name>
        <font-color>0,0,0</font-color>
        <font-height>-13</font-height>
        <font-weight>700</font-weight>
        <font-italic>false</font-italic>
        <font-underline>false</font-underline>
        <font-strike-out>false</font-strike-out>
        <font-char-set>0</font-char-set>
    </default-font>
</header>
<text-list>
    <text prefix="ERROR" id="4711">
        <deutsch> Fehler an Position 4711 </deutsch>
        <english> Error at position 4711 </english>
    </text>
    <text prefix="ERROR" id="815">
        <deutsch> Fehler an Position 815 </deutsch>
        <english> Error at position 815 </english>
    </text>
    <text prefix="ERROR" id="2000">
        <deutsch> <![CDATA[Das ist ein Fehlertext über
        mehrere Zeilen]]> </deutsch>
        <english> <![CDATA[This is a error text over more than
        one line]]> </english>
    </text>
    <text prefix="MASCHINE" id="1">
        <deutsch> <![CDATA[Vorschub]]> </deutsch>
        <english> <![CDATA[Feed rate]]> </english>
    </text>
    <text prefix="MASCHINE" id="2">
        <deutsch> <![CDATA[Beschleunigung]]> </deutsch>
        <english> <![CDATA[Acceleration]]> </english>
    </text>
</text-list>
</dynamic-text>
```

(5) Öffnen Sie in der Visualisierung den Dialog ‚Einstellungen’, Kategorie Sprache: Aktivieren Sie die Option ‚Dynamische Texte’; Fügen Sie die Datei dynamictextsample.xml, die nun auf Ihrem Rechner vorliegt, der Dateienliste hinzu.

(6) Gehen Sie mit dem Projekt in den Online Modus.
(7) Stellen Sie in den Einstellungen der Visualisierung die Sprache auf "deutsch". Setzen Sie PLC_PRG.ivar auf "1" und PLC_PRG.errnum auf "4711". In den Visualisierungselementen sollten nun folgende Texte erscheinen: "Vorschub" bzw. "Fehler an Position 4711".
(8) Setzen Sie PLC_PRG.ivar auf 2 und PLC_PRG.errnum auf "2000". Die Texte wechseln zu "Beschleunigung" und "Das ist ein Fehlertext über mehrere Zeilen. Der Text wird jeweils in Arial 13 dargestellt.
(9) Stellen Sie in den Einstellungen der Visualisierung die Sprache auf "english". Nun werden die folgenden Texte angezeigt: "Acceleration" und "This is a error text over more than one line".

9.2.3 Sprachabhängige Online Hilfe

9.2.3.1 Aufruf sprachabhängiger Online Hilfe

Je nach der Sprache, die für die Visualisierung aktuell eingestellt ist, kann der Aufruf einer sprachspezifischen Hilfedatei mit einem Visualisierungselement verknüpft werden. Dazu muss für dieses Element der Befehl INTERN HELP im Dialog 'Element konfigurieren' bei 'Programm ausführen' eingetragen sein und in der PlcControl.ini-Datei muss eine Sektion [Visu-Helpfiles] vorhanden sein. Unter dieser müssen die entsprechenden Hilfedateien den in der Visualisierung einstellbaren Sprachen zugeordnet werden: z.B.:

```
[Visu-Helpfiles]
German=C:\PROGRAMME\HELP_D.HLP
English=C:\PROGRAMME\HELP_E.HLP
```

9.3 Platzhalterkonzept

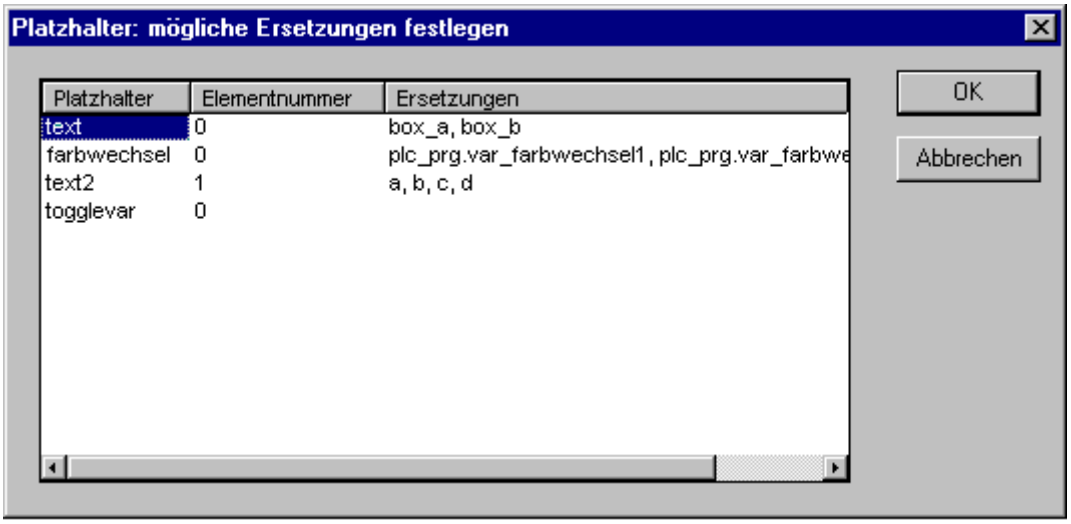
An jeder Stelle eines Konfigurationsdialogs, an dem Variablen oder Text eingegeben wird, kann anstelle der jeweiligen Variablen oder des Textes auch ein **Platzhalter** eingesetzt werden. Dies ist sinnvoll, wenn das Visualisierungsobjekt nicht unmittelbar im Programm verwendet werden soll, sondern dazu erstellt wird, um in anderen Visualisierungsobjekten als Referenz eingefügt zu werden. Beim Konfigurieren einer solchen **Referenz** können die Platzhalter dann durch Variablennamen oder Texte ersetzt werden.

Als Platzhalter gilt jede Zeichenfolge, die von zwei Dollarzeichen ($) eingeschlossen ist (z.B. $variable1$, variable$x$). Für jeden Platzhalter kann im Dialog 'Platzhalterliste' (Aufruf über **'Extras' 'Platzhalterliste')** bereits vordefiniert werden, mit welchen Werten er in der Referenz ersetzt werden kann. Das Ersetzen der Platzhalter erfolgt dort ebenso über eine Platzhalterliste.

'Extras' 'Platzhalterliste'

Die Platzhalterliste wird an zwei verschiedenen Stellen, zur Verwaltung und Konfiguration der Platzhalter, verwendet:

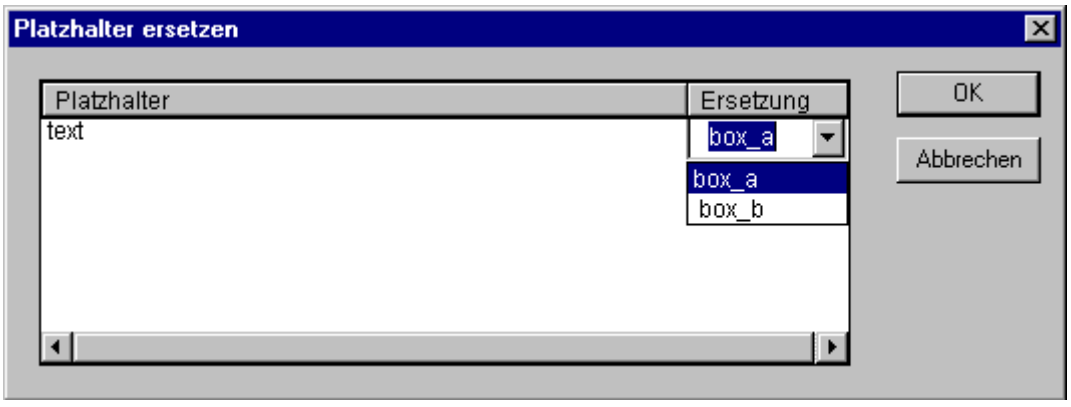
• zunächst bei der **Konfiguration eines Visualisierungsobjekts**, von dem später Referenzen in anderen Visualisierungsbausteinen angelegt werden sollen und in dem Sie deswegen Platzhalter anstellen von Variablen und Strings eintragen. In diesem Fall erhalten Sie den Dialog über den Befehl 'Extras' 'Platzhalterliste' oder das Kontextmenü. Er enthält drei Spalten:



Platzhalterliste zur Eingabe von möglichen Ersetzungswerten für Platzhalter

• In der Spalte **Platzhalter** werden alle Platzhalter aufgelistet, die bei der Konfiguration dieser Visualisierung verwendet werden. Die Spalte **Elementnummer** zeigt an, in welchem Element ein Platzhalter konfiguriert wurde. In der Spalte **Ersetzungen** kann nun für diese Platzhalter eine Auswahl von Strings (Text, Ausdrücke, Variablennamen) eingegeben werden, die dann später beim Konfigurieren einer Referenz des Visu-Bausteins als Eingabe anstelle des Platzhalters möglich sind. Die Elemente der Auswahl müssen durch Kommata getrennt eingegeben werden. Wird hier kein oder ein nicht gültiger Ersetzungsstring vorgegeben, dann kann der Platzhalter bei der Konfiguration der Referenz, die ihn beinhaltet, durch eine beliebige Zeichenfolge/Variablenname ersetzt werden.

• Dann beim **Konfigurieren einer Referenz** des oben genannten Visualisierungsbausteins, also nachdem dieser in einer anderen Visualisierung mit 'Einfügen' 'Visualisierung' eingefügt wurde. Rufen Sie zu diesem Zweck den Dialog folgendermaßen auf: Markieren Sie die eingefügte Visualisierung, wählen Sie den Befehl 'Konfigurieren' aus dem Kontextmenü oder dem Menü 'Extras' und drücken die Schaltfläche Platzhalter in der Kategorie 'Visualisierung': Der Dialog enthält in diesem Fall nur zwei Spalten:



• In der Spalte **Platzhalter** erscheinen wieder die im Baustein verwendeten Platzhalter, wie sie in der ursprünglichen Visualisierung vergeben wurden. Wenn für diese (wie oben beschrieben) eine mögliche Ersetzungs-Wertemenge vordefiniert ist, wird diese nun in der Spalte **Ersetzung** in einer Auswahlliste angeboten. Wählen Sie einen der Werte, der nun in der vorliegenden Referenz anstelle des Platzhalters verwendet werden soll. Wenn keine Wertemenge vordefiniert wurde, können Sie durch einen Mausklick in das entsprechende Feld in der Spalte 'Ersetzung' ein Eingabefeld öffnen und einen gewünschten Ersetzungswert eintippen.

9.4 Online Mode

Beachten Sie folgende Punkte zur Visualisierung im Online Modus:

Auswertungsreihenfolge:

- Die dynamisch, also über normale Projektvariablen oder über die Strukturvariablen gelieferten Werte zur Definition von Visualisierungselementen überschreiben die festen (statischen) Einstellungen der Elementkonfigurationen.
- Wenn eine Elementeigenschaft sowohl durch eine direkt im Konfigurationsdialog eingetragene Projektvariable als auch über die Komponente einer Strukturvariable angesprochen wird, wird zuerst der Wert der Projektvariablen ausgewertet.

• Sie haben die Möglichkeit, eine reine Tastaturbedienung für die Visualisierung zu konfigurieren,

• Die Konfigurations-Kategorien Darstellung, Rahmen und Sprache können auch im Online Modus bearbeitet werden.

• Die einzelnen Elemente von Visualisierungsreferenzen verhalten sich im Online Modus identisch wie die entsprechenden der Visualisierung, die referenziert wird

• Bei Umstellen der verwendeten Landessprache wechselt diese nur im Online Modus.

• Die Visualisierung kann im Online Modus ausgedruckt werden

Erklärungen zur Online-Bedienung bestimmter Visualisierungselemente wie z.B. Trend und Alarmtabelle sehen Sie bitte im entsprechenden Kapitel zur Konfiguration des Elements.

Tastaturbedienung - im Online Modus

Um für die Online-Bedienung einer Visualisierung von Maus und Touch-Screen unabhängig zu sein, empfiehlt es sich, die Visualisierung so zu konfigurieren, dass eine reine Tastaturbedienung der Elemente möglich wird:
Ohne es explizit konfigurieren zu müssen, funktionieren im Online Modus per Default bereits folgende Tasten(kombinationen):

• Durch Drücken der <**Tabulator**>-Taste wird das erste Element der Elementliste markiert, für das eine Eingabe konfiguriert ist. Mit jeder weiteren Betätigung der Taste wechselt man zum jeweils nächsten Element der Elementliste. Bei gleichzeitigem Drücken von <**Umschalt**> wechselt die Markierung zum vorhergehenden Element.

• Mit den Pfeiltasten können Sie von einem selektierten Element in jede Richtung auf das jeweils nächstliegende wechseln.

• Mit der <**Leertaste**> können Sie eine Betätigung auf das selektierte Visualisierungselements durchführen. Handelt es sich um ein Element mit einer Textausgabe-Variablen, wird dadurch ein Texteingabefeld geöffnet, das den Textinhalt dieser Variablen zeigt. Durch Drücken der <**Eingabetaste**> wird dieser Wert geschrieben.

Zusätzliche Tasten(kombinationen für die Online-Bedienung können in der Konfiguration der Visualisierung im Dialog 'Tastaturbedienung...' definiert werden. Dabei können auch die Tasten <Tabulator>, <Leertaste> und <Eingabetaste> mit einer anderen als der oben beschriebenen Funktion versehen werden.

Die einzelnen Elemente von Referenzen verhalten sich im Online Modus identisch wie die entsprechenden der Visualisierung, die referenziert wird. Sie werden also genauso als einzelne Elemente auf Eingaben und Bedienung durch Maus und Tastatur reagieren, auch die Anzeige der Tooltips bei Referenzen ist elementbezogen Bei einer Abarbeitung der Elementliste, wie sie beispielsweise beim Springen von einem Eingabelement zum nächsten per Tabulator erfolgt, erfolgt die Abarbeitung aller Einzelelemente einer Referenz ab der Stelle, an der die Referenz in der Elementliste steht, bevor zum nächsten Element der Liste gesprungen wird.

Visualisierung Drucken im Online Modus

Über 'Datei' 'Drucken' können Sie den Inhalt des Visualisierungsfensters im Online Modus drucken. Visualisierungen, die über die Seitenränder hinausgehen, können dabei Inkonsistenzen zeigen, vor allem, wenn sich bewegte Elemente in der Visualisierung befinden.

9.5 Bibliotheken

Visualisierungen können auch in Bibliotheken mit abgelegt werden und somit als Bibliotheksbausteine in Projekten zur Verfügung gestellt werden. Diese können wie die im Projekt direkt vorliegenden Visualisierungen als Referenzen eingefügt werden oder über den Befehl "Zoomen nach Vis." in der Eingabe-Konfiguration einer anderen Visualisierung aufgerufen werden.

HINWEIS

Eindeutige Namen

Visualisierungen, die in einem Projekt verwendet werden, sollten eindeutige Namen tragen. Es kann zu Problemen führen, wenn beispielsweise eine Visualisierung aus einer Bibliothek referenziert oder aufgerufen wird, die den gleichen Namen hat, wie eine im Projekt vorliegende. Denn bei der Abarbeitung von Referenzen oder Visualisierungsaufrufen im Programm werden zunächst die Visualisierungen im Projekt, erst danach die der geladenen Bibliotheken berücksichtigt.

9.6 TwinCAT PLC HMI Visualisierung

Das **TwinCAT PLC HMI** ist ein System zur Ausführung von Visualisierungen, die mit dem TwinCAT Programmiersystem erstellt wurden. Wenn ein Steuerungsprogramm entsprechende Visualisierungen enthält, werden diese nach dem Start von **TwinCAT PLC HMI** im Vollbildmodus dargestellt und der Benutzer kann darüber per Mausklick oder Tastatur die im zugrunde liegenden Projekt enthaltenen Steuer- und Überwachungsfunktionen bedienen. Dies ist auch möglich, wenn die TwinCAT PLC Projektdatei mit einem Leseschutz versehen ist. Der Anwender hat keine Möglichkeit, das Steuerungsprogramm zu editieren, Menüs und Funktionsleisten stehen nicht zur Verfügung, es handelt sich um eine reine 'Bedienung' der enthaltenen Visualisierungselemente.

Die wesentlichen Steuer- und Überwachungsfunktionen in einem Projekt müssen also beim Erstellen eines für die Bedienversion vorgesehenen Projektes auf Visualisierungselemente gelegt werden und damit im Online Modus bedient werden. Hierzu gibt es spezielle Eingabemöglichkeiten im Konfigurationsdialog eines Visualisierungselements.

Durch die nahtlose Integration in TwinCAT PLC Control bietet die Visualisierung mit **TwinCAT PLC HMI** folgende Vorteile:

• Es kann direkt mit den TwinCAT PLC Control Variablen des Steuerungsprogramms gearbeitet werden.

• Die Verwendung von Ausdrücken in der Visualisierungskonfiguration ist möglich (z.B. „Variable1+ Variable2" "12 + 5")

• Ein Platzhalterkonzept ermöglicht objektorientiertes Arbeiten.

• Die TwinCAT PLC Control Funktionen Trace und Rezepturen lesen/schreiben stehen auch in **TwinCAT PLC HMI** zur Verfügung

9.6.1 Installation, Start und Bedienung

Installation:

TwinCAT PLC HMI steht als Supplement zur Verfügung und kann mit dem Setup nachinstalliert werden. Eine Lizenz ist kostenpflichtig und eine limitierte Demoversion steht nicht zur Verfügung.

Start:

TwinCAT PLC HMI (TCatPlcCtrlHmi.exe) wird über eine Verknüpfung oder die Kommandozeile gestartet.

In jedem Fall muss mindestens das gewünschte TwinCAT PLC Control Projekt angegeben werden. Wenn der Aufruf keine weiteren Parameter enthält, startet TwinCAT PLC HMI automatisch mit dem Visualisierungsbaustein **TC_VISU**, vorausgesetzt, dass ein solcher im Projekt vorhanden ist, und auf dem Zielsystem wie es beim letzten Speichern des Projekts eingestellt war.

Es stehen jedoch zusätzlich sowohl die üblichen in TwinCAT verfügbaren Kommandozeilen- und Kommandodateibefehle (siehe TwinCAT Handbuch) als auch der folgende spezielle Parameter zur Verfügung:

/visu <Visualisierungsbaustein>

Wenn das Projekt einen Visualisierungsbaustein mit dem Namen TC_VISU enthält, wird automatisch mit diesem gestartet; soll ein anderer als Einsprungbaustein funktionieren, muss dieser mit einem Zusatz "/visu <Visualisierungsbaustein>" im Aufrufkommando genannt werden

Beispiel für einen Kommandozeilenaufruf:

C:\TwinCAT\Plc\TCatPlcCtrlHmi.exe D:\PROJECTS\PROJECT.PRO /visu v_firstvisupage

Das Projekt project.pro startet mit dem Visualisierungsbaustein 'v_firstvisupage'. Soll die Visualisierung mit TwinCAT automatisch gestartet werden, kann dieses über eine Verknüpfung im TwinCAT StartUp (All Users) auf die 'TCatPlcCtrlHmi.exe' und genannte Parameter erfolgen.

Wenn im Pfad Leerzeichen enthalten sind, muss er in Hochkommas (") gefasst werden.

Bedienung:

Das Projekt startet im **Vollbildmodus** mit dem Start-Visualisierungsbaustein.

Die weitere Bedienung des Projekts erfolgt per Tastatur und Maus über die Visualisierungselemente.

Steht kein Visualisierungselement mit der entsprechenden Funktion zur Verfügung, kann TwinCAT PLC HMI jederzeit mit **<Alt><F4>** beendet werden.

9.7 Target-Visualisierung

Die Target-Visualisierung ist eine der möglichen Anwendungen einer in TwinCAT erzeugten Visualisierung. Das TwinCAT Programmiersystem kann für die Visualisierungsobjekte eines Projekts ST-Code (Strukturierter Text) erzeugen, der mit den anderen Programmbausteinen auf die Steuerung geladen wird.

Wenn das Laufzeitsystem die Funktionalität unterstützt und entsprechende Monitorausstattung vorhanden ist, kann die Visualisierung dann direkt auf der Steuerung gestartet werden. Das Programmiersystem muss nicht mehr laufen, um die Visualisierung nutzen zu können, was erheblich verringerten Hauptspeicherbedarf bedeutet.

Funktionen der Bibliothek **SysLibTargetVisu.lib** erlauben unter anderem, Informationen über die letzten Benutzereingaben [▶ 270] (Position des letzten Mausklicks, bisherige Anzahl von Klicks) im Programm zu erhalten bzw. Texte aus dynamischen Textlisten abzufragen um diese Werte direkt im Programm zu verwenden.

9.7.1 Feature Übersicht der Visualisierung mit TwinCAT

Die folgende Tabelle gibt einen Überblick über die einzelnen Möglichkeiten der Visualisierung mit TwinCAT

Voraussetzungen

	PLC Control	PLC HMI	PLC HMI Web	PLC HMI CE	Comment
Current test version:	**Build 1010**	**2.10.0.900**	**1.0.0.7**	**1.0.9.10**	**Ab TwinCAT 2.10 Build 1334**
Rectangle	✓	✓	✓	✓	
Rounded rectangle	✓	✓	✓	✓	
Ellipse	✓	✓	✓	✓	
Polygon	✓	✓	✓	✓	
Curve	✓	✓	✓	✓	
Pie	✓	✓	✓	✓	
Bitmap	✓	✓	✓	✓	
Visualisation	✓	✓	✓	✓	
Button	✓	✓	✓	✓	
WMF/JPG File	✓	✓	✓	✓	
Table	✓	✓	✓	✓	
ActiveX element	✓	✓	✗	✗	
Trend	✓	✓	✗	✓ / ✗	HMI CE: Only online trend is available
Alarm table	✓	✓	✗	✗	
Meter	✓	✓	✓	✓	
Bar display	✓	✓	✓	✓	
Histogram	✓	✓	✗	✗	
Invisible elements	✓	✓	✓	✓	
Change color	✓	✓	✓	✓	
Background bitmap	✓	✓	✓	✓	
Button background	✓	✓	✓	✓	
Tooltip	✓	✓	✓	✓	HMI CE: Elements not too near to the edge or tooltip is outside the window

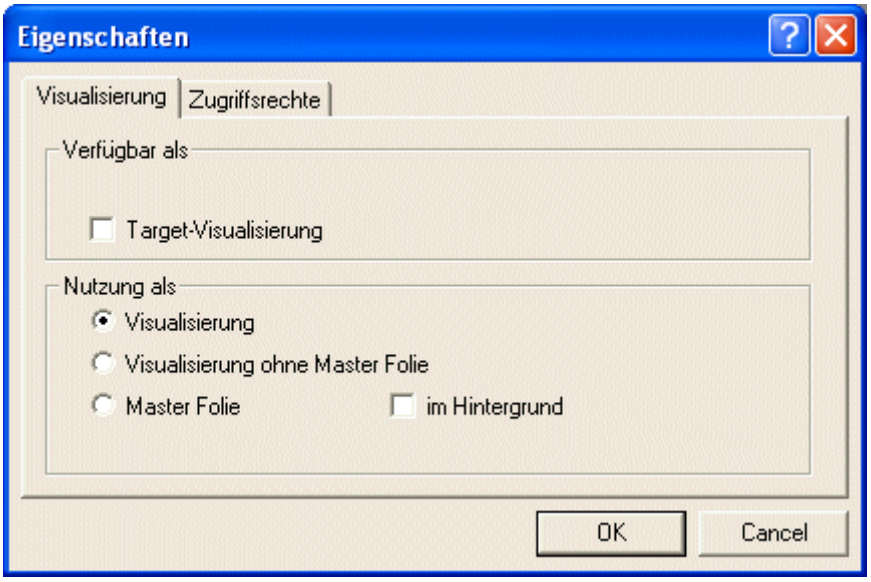
	PLC Control	PLC HMI	PLC HMI Web	PLC HMI CE	Comment
Security	✓	✓	✓	✓	
Placeholder	✓	✓	✓ / ✗	✓ / ✗	Web/HMI CE: Zoom to visu with placeholders in the zoom command is not possible
Print function	✓	✓	✗	✗	
Password change	✓	✓	✓	✓	
Change user level	✓	✓	✓	✓	
Language dialog	✓	✓	✓	✓	
Language automatic change	✓	✓	✓	✓	
Exit	✓	✓	✗	✗	
Trace	✓	✓	✗	✗	
Text input 'Text'	✓	✓	✓	✓	Web: Please don't exceed the maximum string length of your variable HMI CE: The hidden function is not supported
Text input 'Numpad'	✓	✓	✓	✓	HMI CE: The hidden function is not supported
Text input 'Keypad'	✓	✓	✓	✓	Web: Please don't exceed the maximum string length of your variable HMI CE: The hidden function is not supported

Restrictions

Restrictions [▶ 272]

9.7.2 Voraussetzungen

• Das Zielsystem muss die Funktionalität unterstützen, d.h. in den Zielsystemeinstellungen (Projekt->Optionen->TwinCAT) muss die Option 'Enable CE Target-Visualization' aktiviert sein. Wenn es in der Target-Datei entsprechend definiert ist, kann diese Option auch vom Anwender ein- oder ausgeschaltet werden.



• Die Bibliothek **SysLibTargetVisu.lib** wird benötigt, um die Visualisierungsfunktionen im Laufzeitsystem zu implementieren. Sie wird automatisch im Bibliotheksverwalter eingefügt, wenn die Option "Enable CE Target-Visualization" in den Zielsystemeinstellungen aktiviert ist. Im Laufzeitsystem muss SysLibTargetVisu.lib ebenfalls implementiert sein.

• Bitte beachten Sie, das Sie bei der Verwendung der Trend-Funktion unter Windows CE die SysLibAlarmTrend.lib manuell in Ihr Projekt einbinden müssen. Bei Geräten mit ARM-Prozessoren wird die Trend-Funktion nicht unterstützt.

• Das Betriebssystem des Steuerungsrechners: Windows CE

• Der Steuerungsrechner benötigt entsprechende Monitor- und Bedienausrüstung, um die Visualisierung darstellen bzw. bedienen zu können.

9.7.3 Bereitstellen einer Target Visualisierung

1. Erstellen Sie eine Visualisierung in TwinCAT PLC.

Um die Leistung zur Laufzeit zu verbessern, sollten Sie möglichst viele Elemente, welche in der Darstellung statisch sind (keine Bewegungen, keine Textveränderung, keine Farbveränderung, ...) **nach hinten** legen. Tipp: Mit Hilfe von 'Extras' 'Elementliste' können sämtliche Elemente „Nach vorn“ und „Nach hinten“ verschoben werden. Der Hintergrund ist folgender. Alle statischen Elemente werden nur einmal in eine Hintergrundbitmap gezeichnet. Dadurch wird der zyklische Zeichenaufwand stark verringert. Dies macht sich vor allem für komplexe Polygone oder auch Bitmaps bezahlt.

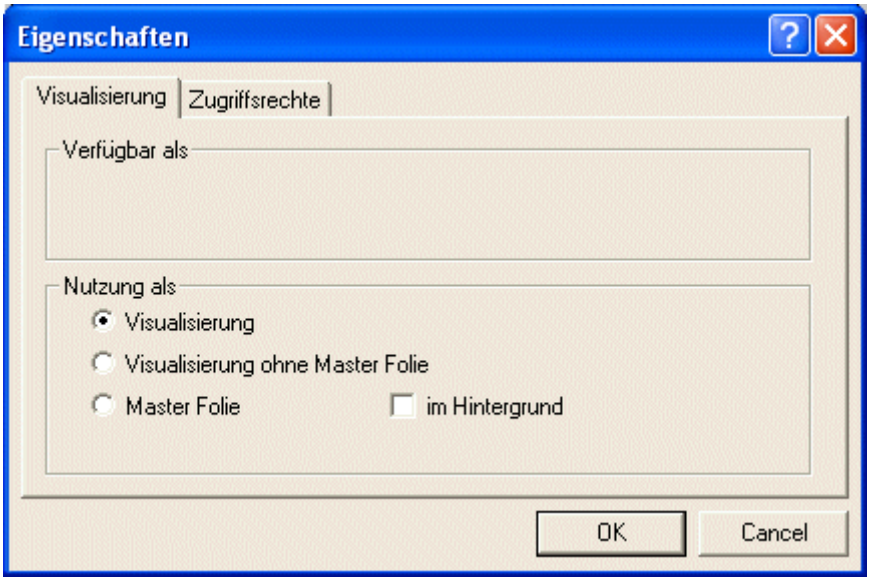
Wenn ein Visualisierungsbaustein mit dem Namen **TC_VISU** vorhanden ist, wird die Target-Visualisierung später beim Aufruf mit diesem Baustein gestartet. Ansonsten wird der Baustein geladen werden, der an erster Stelle der Objektliste im Register 'Visualisierungen' in TwinCAT PLC Control steht.

Falls in Ihrem Projekt implizite Visualisierungsvariablen als **remanente** Variablen behandelt werden sollen, nehmen Sie die entsprechenden Deklarationen bei den Globalen Variablen [▶ 273] vor.

Die Bitmaps der Visualisierung werden als Dateien übertragen.

2. Deaktivieren Sie

für jeden Visualisierungsbaustein, der nicht in der Target-Visualisierung enthalten sein soll, in den **Objekt-Eigenschaften** ('Projekt' 'Objekt' 'Eigenschaften') im Registerblatt 'Visualisierung' die Option 'Visualisierung':



3. Konfigurieren Sie

in den Zielsystemeinstellungen in der Kategorie "TwinCAT" (die jeweilige Verfügbarkeit der Optionen ist abhängig vom Zielsystem!) dass das Projekt für die Target-Visualisierung vorgesehen ist: Option "Enable CE Target Visualization" aktivieren.

Zusätzlich kann hier gewählt werden, ob die Benutzereingaben und das Neuzeichnen der Target-Visualisierungselemente ...
... mit Hilfe von automatisch angelegten VISU-Tasks oder über individuelle Programmierung gesteuert werden sollen: Option **Taskerzeugung deaktivieren**
... von einem oder aufgeteilt von zwei Bausteinen bzw. Tasks abgearbeitet werden sollen:
Option **VISU_INPUT_TASK** verwenden. (Lassen Sie sich in diesem Fall nicht durch den Begriff "..._TASK" irritieren, die Option hat auch Bedeutung, wenn die automatische Task-Erzeugung deaktiviert ist!)

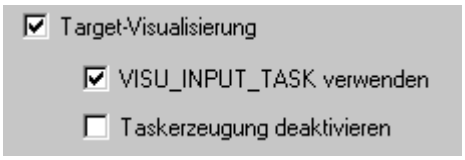
Folgende Konfigurationen sind also, abgesehen von der Option der Deaktivierung der Tastaturbedienung für Tabellen, möglich:

Steuerung über automatisch erzeugte VISU-Tasks, die die Programme MAINTARGETVISU_PAINT _CODE und MAINTTARGETVISU_INPUT_CODE aufrufen:

Fall A:

Taskerzeugung deaktivieren ist ausgeschaltet:

VISU_INPUT_TASK verwenden ist aktiviert:



Es werden automatisch zwei Tasks mit je einem Programmaufruf angelegt:
- **VISU_TASK** ruft den implizit verfügbaren Programmbaustein **MAINTARGETVISU_PAINT_CODE** auf, der das Neuzeichnen der Visualisierungselemente übernimmt.
- **VISU_INPUT_TASK** ruft den implizit verfügbaren Programmbaustein **MAINTARGETVISU_INPUT_CODE** auf, der die Abarbeitung der Benutzereingaben übernimmt.

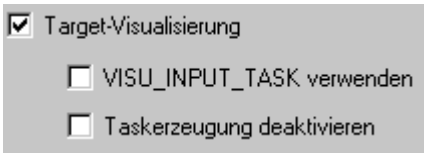
Die Default-Einstellungen der Tasks:
VISU_INPUT_TASK: zyklisch, Priorität 2, Intervall t#50ms;
VISU_TASK: zyklisch, Priorität 3, Intervall t#200ms

Diese Parameter können natürlich modifiziert werden, aber: VISU_INPUT_TASK sollte immer vorrangig vor VISU_TASK abgearbeitet werden, um ein sinnvolles Zusammenspiel von Eingaben und Aktualisieren der Darstellung zu gewährleisten.
Die Task, die das Haupt-Programm (z.B. PLC_PRG) aufruft, sollte mindestens so oft wie VISU_INPUT_TASK, idealerweise sogar noch vorrangiger abgearbeitet werden, kann allerdings aber auch direkt an VISU_INPUT_TASK angehängt sein.

Fall B:

Taskerzeugung deaktivieren ist ausgeschaltet:

VISU_INPUT_TASK verwenden ist deaktiviert:



Nur die Task **VISU_TASK** wird automatisch angelegt, beinhaltet jedoch zusätzlich die Funktion der VISU_INPUT_TASK.
Das implizite Programm **MAINTARGETVISU_ PAINT_CODE** enthält in diesem Fall zusätzlich die Funktionalität des Programms MAINTTARGET VISU_ INPUT_CODE.

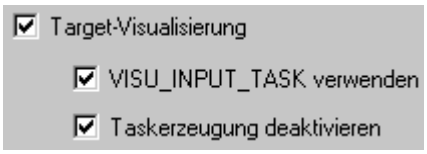
Diese Konfiguration ist für Systeme geeignet, die kein Multi-Tasking erlauben. Nachteilig ist, dass keine differenzierte Abarbeitungszeiten für die Verarbeitung von Benutzereingaben und dem Neuzeichnen der Elemente eingestellt werden können, s.o.

Steuerung ohne automatisch festgelegte Tasks;
die impliziten Programmbausteine MAINTARGETVISU_PAINT_CODE und MAINTTARGETVISU_INPUT_CODE können individuell im Applikationsprogramm aufgerufen bzw. an Tasks gebunden werden:

Fall C:

Taskerzeugung deaktivieren ist aktiviert:

VISU_INPUT_TASK verwenden ist aktiviert:



Die beiden impliziten Programmbausteine stehen zur Verfügung und können einzeln aufgerufen werden bzw. an eine beliebige Task geknüpft werden. (Beachten Sie hierzu die Tipps in (Fall A).

Beispiel für das Steuern der Target-Visu-Programmbausteine im Applikationsprogramm:

```
PROGRAM visu_control
VAR
    n: INT;
END_VAR

n:=n+1;
IF (n MOD 4) =0 THEN
   MAINTARGETVISU_PAINT_CODE();
END_IF;
MAINTARGETVISU_INPUT_CODE();
```

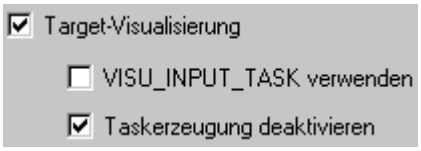
Hier wird im Programm visu_control der Eingabeverarbeitende Baustein nur nach jedem vierten Aufruf des zeichnenden Bausteins aufgerufen, um die Gefahr zu vermindern, dass das Neuzeichnen durch eine Benutzereingabe unterbrochen wird.

Achten Sie beim Anlegen Ihres Programms unbedingt darauf, diese Gefahr der unkorrekten Anzeige zu umgehen!

Fall D:

Taskerzeugung deaktivieren ist aktiviert:

VISU_INPUT_TASK verwenden ist deaktiviert:



Es steht nur der implizite Programmbaustein MAINTARGET VISU_PAINT_CODE zur Verfügung, beinhaltet jedoch zusätzlich die Funktionalität von MAINTARGETVISU_ INPUT_ CODE.
Er kann im Applikationsprogramm aufgerufen bzw. an eine beliebige Task gebunden werden.

4. Laden Sie das Projekt

auf die Steuerung ('Online' 'Einloggen').

9.7.4 Aufruf aus dem Zielsystem

Starten Sie das geladene Projekt auf der Steuerung. Die Visualisierung startet daraufhin mit dem Baustein TC_VISU bzw. – falls ein solcher nicht vorhanden ist – mit dem Visualisierungsobjekt, das an erster Stelle der Objektliste im Registerblatt 'Visualisierungen' in TwinCAT PLC Control steht.

9.7.5 Schnittstelle zur Abfrage von Benutzeraktionen und dynamischen Texten

Die beim Aktivieren der Zielsystem-Einstellung ‚Target-Visualisierung' automatisch eingebundene Bibliothek SysLibTargetVisu.lib bietet die folgenden Funktionen zur Abfrage von Benutzereingaben durch Einträge in dynamischen Textlisten:

Funktion GetText : BOOL

Diese Funktion liefert einen sprachabhängigen Text aus der aktuellen dynamischen Textliste.

Übergabe-Parameter:

stResult: STRING(256);	Dient als IN_OUT Parameter und erhält den Text zugewiesen, der mittels des Prefix strings „stPrefix“ und der ID „dwID“ gefunden wurde.
nResultLength:INT;	Hier sollte die maximale Länge des strings „stResult“ übergeben werden, falls diese kleiner als 256 Zeichen ist.
stPrefix: STRING;	Präfix des Texteintrags in der aktuellen dynamischen Textliste.
dwID: DWORD;	ID des Texteintrags in der aktuellen dynamischen Textliste.

Rückgabewert:
FALSE - Es wurde kein passender Text zu „stPrefix“ und „dwID“ gefunden.
TRUE - Es wurde ein passender Text zu „stPrefix“ und „dwID“ gefunden.

Funktion GetTextByStringId: BOOL

Diese Funktion kann wie die Funktion GetText einen sprachabhängigen Text der dynamischen Textlisten zurückliefern. Der Unterschied besteht im Parameter „stID“ über den die ID des Texteintrags als Text-String anstatt als numerischer Wert übergeben werden kann. Dadurch ist es möglich, auch IDs, die in der xml-Textdatei als Strings definiert sind, wie z.B. „Text123“, verwenden zu können.

Übergabe-Parameter:

stResult: STRING(256);	Dient als IN_OUT Parameter und erhält den Text zugewiesen, der mittels des Prefix strings „stPrefix“ und der ID „dwID“ gefunden wurde.
nResultLength:INT;	Hier sollte die maximale Länge des strings „stResult“ übergeben werden, falls diese kleiner als 256 Zeichen ist.
stPrefix: STRING;	Präfix des Texteintrags in der aktuellen dynamischen Textliste.
stID: STRING;	ID des Texteintrags in der aktuellen dynamischen Textliste.

Rückgabewert:
FALSE - Es wurde kein passender Text zu „stPrefix“ und „stID“ gefunden.
TRUE - Es wurde ein passender Text zu „stPrefix“ und „stID“ gefunden.

Des Weiteren können Benutzereingaben via Mausklicks analysiert werden. Dazu ist die Bibliothek TcMouseEvents.lib mit folgenden Funktionen manuell einzubinden:

Funktion GetLastLeftMouseDownEvent : BOOL bzw. Funktion GetLastRightMouseDownEvent : BOOL

Diese Funktion bietet Information zum zuletzt ausgelösten "Left" bzw. "Right" MouseDown Event. Sie enthält einen Zeiger (pMouseEvent : POINTER TO MOUSEEVENT;) auf die Struktur MouseEvent, die folgende Parameter umfaßt:

dwCounter : DWORD;	Anzahl der MouseDownEvents seit Systemstart. Mit diesem Parameter kann ausgewertet werden ob kein, ein oder mehrere Events seit der letzten Anfrage ausgelöst wurden.
nXPos : INT;	Letzte Maus-Position in X/Y-Koordinaten
nYPos : INT;	

Rückgabewert:
Es wird kein Wert zurückgeliefert.

Funktion GetLastMouseMoveEvent : BOOL

Diese Funktion bietet Information zum zuletzt ausgelösten MouseMove Event. Sie enthält einen Zeiger (pMouseEvent : POINTER TO MOUSEEVENT;) auf die Struktur MouseEvent. Siehe oben, Funktion GetLastMouseDownEvent.

Rückgabewert:
Es wird kein Wert zurückgeliefert.

Funktion GetLastLeftMouseUpEvent : BOOL bzw. Funktion GetLastRightMouseUpEvent : BOOL

Diese Funktion bietet Information zum zuletzt ausgelösten "Left" bzw. "Right" MouseUp Event. Sie enthält einen Zeiger (pMouseEvent : POINTER TO MOUSEEVENT;) auf die Struktur MouseEvent.

Siehe oben, Funktion GetLastLeftMouseDownEvent bzw. Funktion GetLastRightMouseDownEvent .

Rückgabewert:
Es wird kein Wert zurückgeliefert.

9.7.6 Einschränkungen

Interne Befehle

PRINT
Dieser Befehl zum Ausdrucken der aktuellen Visualisierung kann für die Target-Visualisierung nicht verwendet werden.

EXITPROGRAM
Dieser Befehl zum Beenden des Programms kann für die Target-Visualisierung nicht verwendet werden.

TRACE
Dieser Befehl zum Öffnen des Fensters zur Traceaufzeichnung kann für die Target-Visualisierung nicht verwendet werden.

SAVEPROJECT
Dieser Befehl zum Speichern des Projekts kann für die Target-Visualisierung nicht verwendet werden.

Grafikformate

Innerhalb der Target-Visualisierung werden zur Zeit nur einfache Bitmaps unterstützt. Nicht unterstützt werden die Formate .jpg, .tif, .ico. Unterstützung für das Format .jpg ist ab der Version 1.0.9 der TargetVisu DLL verfügbar.

Sonstiges

Slider in Tabelle
Der Slider für das Scrollen kann gegenwärtig nicht angezeigt werden.

Texte
Clipping Texte, die über die Grenzen des Elementes hinaus reichen, werden gegenwärtig noch nicht abgeschnitten

Alarmbehandlung
Das Alarming wird derzeit von der Target-Visualisierung nicht unterstützt.

Trend
Die Target-Visualisierung unterstützt ab der Version 1.0.8 den Online-Trend (ohne History).

Platzhalter
Die Übergabe von Parametern zur Ersetzung von Platzhaltern beim Aufruf wird von der Target-Visualisierung nicht unterstützt.
Beispiel:
<Visuname>(<Platzhalter1>:=<Text1>, <Platzhalter2>:=<Text2>,..., <Platzhalter n>:=<Textn>)

VAR_IN_OUT
VAR_IN_OUT Variablen können in der TwinCAT HMI CE nicht verwendet werden.

Visu-Seite
Wenn eine Visu-Seite mit der HMI CE angezeigt werden soll, sollte sie speziell für diese Form der Visualisierung entwickelt werden. Dies ist nötig da Scroll Bars zur Zeit in der HMI CE nicht unterstützt werden.

9.8 Systemvariablen

Folgende implizit erzeugte Systemvariablen stehen für die Programmierung von Visualisierungen zur Verfügung:

Voraussetzungen

Implizit generierte Variable	Datentyp	Funktion
CurrentVisu	String[40]	Enthält den Namen der aktuellen Visualisierung. Wird der Name verändert, so wird ein Visualisierungswechsel durchgeführt. Zu beachten ist, das der String für den Visualisierungsnamen immer in Grossbuchstaben anzugeben ist.Zielsystemabhängig kann diese Variable In den Zielsystemeinstellungen, Kategorie Visualisierung aktiviert/deaktiviert werden.
CurrentCaller	String[40]	Enthält den Namen der vorherigen Visualisierung. Wird für die Funktionalität ZOOMTOCALLER verwendet.
CurrentLanguage	String[40]	Enthält die aktuell angewählte Sprache, welche innerhalb der Sprachdatei zur Verfügung steht. Diese ist in Grossbuchstaben anzugeben.
CurrentUserLevel	INT	Enthält den aktuell angewählten Benutzerlevel 0..7
CurrentPasswords[0 .. 7]	ARRAY [0..7] OF STRING[20]	Enthält alle Passwörter welche innerhalb von TwinCAT PLC Control unter „Optionen“ „Passwörter für Arbeitsgruppe“ eingetragen wurden.

10 Anhang

10.1 Tastaturbedienung

Um TwinCAT PLC Control nur mit der Tastatur bedienen zu können, müssen Sie einige Befehle benutzen, die Sie nicht im Menü finden können.

- Mit der Funktionstaste <F6> wechseln Sie in einem geöffneten Baustein zwischen Deklarationsteil und Anweisungsteil hin und her.
- Mit <Alt>+<F6> wechseln Sie von einem geöffneten Objekt zum Object Organizer und von dort zum Meldungsfenster, falls es geöffnet ist. Ist ein Suche-Dialog geöffnet, dann wechseln Sie mit <Alt>+<F6> vom Object Organizer zum Suche-Dialog und von dort zum Objekt.
- Mit dem <Tabulator> springen Sie in den Dialogen zwischen den Eingabefeldern und Schaltflächen weiter.
- Mit den Pfeiltasten bewegen Sie sich innerhalb des Object Organizers und des Bibliotheksverwalters durch die Registerkarten und die Objekte. Alle anderen Aktionen können über die Menübefehle oder über die Kurzformen, die sich hinter den Menübefehlen befinden ausgelöst werden.
- Mit <Umschalt<+<F10> erhalten Sie das Kontextmenü mit den am häufigsten verwendeten Befehlen für ein markiertes Objekt oder für den aktiven Editor.

Tastenkombinationen

Hier finden Sie eine Übersicht aller Tastenkombinationen und Funktionstasten:

Allgemeine Bedienung	
Wechsel zwischen Deklarationsteil und Anweisungsteil eines Bausteins	<F6>
Wechsel zwischen Object Organizer, Objekt und Meldungsfenster	<Alt>+<F6>
Kontextmenü	<Umschalt>+<F10>
Kurzformmodus für Deklarationen	<Strg>+<Eingabetaste>
Wechsel von Meldung im Meldungsfenster zu der Stelle im Editor	<Eingabetaste>
Auf- und Zuklappen mehrstufiger Variablen	<Eingabetaste>
Auf- und Zuklappen von Ordnern	<Eingabetaste>
Registerkartenwechsel im Object Organizer und Bibliotheksverwalter	<Pfeiltasten>
Weiterspringen in Dialogen	<Tabulator>
Kontextsensitive Hilfe	<F1>

Allgemeine Befehle	
'Datei"Speichern'	<Strg>+<S>
'Datei"Drucken'	<Strg>+<P>
'Datei"Beenden'	<Alt>+<F4>
'Projekt"Objekt löschen'	<Entf>
'Projekt"Alles Übersetzen'	<Umschalt>+<F8>
'Projekt"Objekt einfügen'	<Einfg>
'Projekt"Objekt umbenennen'	<Leertaste>
'Projekt"Objekt bearbeiten'	<Eingabetaste>
'Bearbeiten"Rückgängig'	<Strg>+<Z>
'Bearbeiten"Wiederherstellen'	<Strg>+<Y>
'Bearbeiten"Ausschneiden'	<Strg>+<X> oder <Umschalt>+<Entf>
'Bearbeiten"Kopieren'	<Strg>+<C>
'Bearbeiten"Einfügen'	<Strg>+<V>
'Bearbeiten"Löschen'	<Entf>
'Bearbeiten"Weitersuchen'	<F3>
'Bearbeiten"Eingabehilfe'	<F2>
'Bearbeiten"Nächster Fehler'	<F4>
'Bearbeiten"Vorheriger Fehler'	<Umschalt>+<F4>
'Online"Einloggen'	<F11>
'Online"Ausloggen'	<F12>
'Online"Start'	<F5>
'Online"Breakpoint an/aus'	<F9>
'Online"Einzelschritt über'	<F10>
'Online"Einzelschritt in'	<F8>
'Online"Einzelzyklus'	<Strg>+<F5>
'Online"Werte schreiben'	<Strg>+<F7>
'Online"Werte forcen'	<F7>
'Online"Forcen aufheben'	<Strg>+<Umschalt>+<F7>
‘Online’‘Schreiben/Forcen Dialog’	<Umschalt>+<F7>
‘Online’‘Ablaufkontrolle’	<Strg>+<F11>
'Fenster"Meldungen'	<Umschalt>+<Esc>

Befehle des FUP-Editors	
'Einfügen"Netzwerk (danach)'	<Umschalt>+<T>
'Einfügen"Zuweisung'	<Strg>+<A>
'Einfügen"Sprung'	<Strg>+<L>
'Einfügen"Return'	<Strg>+<R>
'Einfügen"Operator'	<Strg>+<O>
'Einfügen"Funktion'	<Strg>+<F>
'Einfügen"Funktionsblock'	<Strg>+<B>
'Einfügen"Eingang'	<Strg>+<U>
'Extras"Negation'	<Strg>+<N>
'Extras"Zoom'	<Alt>+<Eingabetaste>

Befehle des CFC-Editors	
'Einfügen"Baustein'	<Strg>+<B>
'Einfügen"Eingang'	<Strg>+<E>
'Einfügen"Ausgang'	<Strg>+<A>
'Einfügen"Sprung'	<Strg>+<G>
'Einfügen"Label'	<Strg>+<L>
'Einfügen"Return'	<Strg>+<R>
'Einfügen"Kommentar'	<Strg>+<K>
'Einfügen"Bausteineingang'	<Strg>+<U>
'Extras"Negation'	<Strg>+<N>
'Extras"Set/Reset'	<Strg>+<T>
'Extras"Verbindung'	<Strg>+<M>
'Extras"EN/ENO'	<Strg>+<O>
'Extras"Zoom'	<Alt>+<Eingabetaste>

Befehle des KOP-Editors	
'Einfügen"Netzwerk (danach)'	<Umschalt>+<T>
'Einfügen"Kontakt'	<Strg>+<O>
'Einfügen"Paralleler Kontakt'	<Strg>+<R>
'Einfügen' 'Funktionsblock'	<Strg>+<B>
'Einfügen"Spule'	<Strg>+<L>
'Extras"Darunter Einfügen'	<Strg>+<U>
'Extras"Negation'	<Strg>+<N>

Befehle des AS-Editors	
'Einfügen"Schritt-Transition (davor)'	<Strg>+<T>
'Einfügen"Schritt-Transition (danach)'	<Strg>+<E>
'Einfügen"Alternativzweig (rechts)'	<Strg>+<A>
'Einfügen"Paralellzweig (rechts)'	<Strg>+<L>
'Einfügen"Sprung' (AS)	<Strg>+<U>
'Extras"Zoom Aktion/Transition'	<Alt>+<Eingabetaste>
Wechsel aus AS-Übersicht zurück in Editor	<Eingabetaste>

Bedienung der Steuerungskonfiguration	
Auf- und Zuklappen von Organisationselementen	<Eingabetaste>
Editierrahmen um den Namen setzen	<Leertaste>
'Extras"Eintrag bearbeiten'	<Eingabetaste>

Bedienung der Taskkonfiguration	
Editierrahmen um den Task- oder Programmnamen setzen	<Leertaste>

10.2 Übersetzungsfehler

Hier finden Sie die Fehlermeldungen die der Parser anzeigt (kursiv), und deren mögliche Ursachen:

Warnungen

Nummer	Fehlermeldung	mögliche Ursache
1100	Unbekannte Funktion '<Name>' in Bibliothek.	Sie verwenden eine externe Bibliothek. Überprüfen Sie, ob alle Funktionen, die in der .hex-Datei angegeben sind, auch in der .lib-Datei definiert sind
1101	Nicht aufgelöstes Symbol '<Symbol>	Der Codegenerator erwartet einen Baustein mit dem Namen <Symbol>. Dieser ist im Projekt nicht definiert. Definieren Sie eine Funktion/ein Programm mit dem entsprechenden Namen.
1102	Ungültige Schnittstelle für Symbol '<Symbol>	Der Codegenerator erwartet eine Funktion mit dem Namen <Symbol> und genau einem skalaren Eingang oder ein Programm mit dem Namen <Symbol> und keinem Ein- oder Ausgang.
1103	Die Konstante '%s' an Code-Adresse <%04X %04X> liegt über einer 16K Seitengrenze!	Eine Stringkonstante liegt über der 16K Seitengrenze. Das System kann dies nicht handhaben. Abhängig vom Laufzeitsystem besteht eventuell die Möglichkeit, dies über einen Eintrag in der Targetdatei zu umgehen. Bitte wenden Sie sich diesbezüglich an Ihren Steuerungshersteller.
1200	Task '%s', Aufruf von '%s' Accessvariablen in der Parameterliste werden nicht aktualisiert	Variablen, die nur bei einem Funktionsbaustein-Aufruf in der Taskkonfiguration verwendet werden, werden nicht in die Querverweisliste eingetragen.
1300	Die Datei '<Name>' wurde nicht gefunden	Die Datei, auf die das globale Variablenobjekt verweist, existiert nicht. Prüfen Sie den Pfad.
1301	Analyse-Bibliothek wird nicht gefunden. Code für Analyse wird nicht erzeugt.	Sie verwenden die Analyze-Funktion, die Bibliothek analyzation.lib fehlt jedoch. Fügen Sie die Bibliothek im Bibliotheksverwalter ein.
1302	Neue extern referenzierte Funktionen eingefügt. Online Change ist damit nicht mehr möglich!	Sie haben seit dem letzten Download eine Bibliothek eingebunden, die Funktionen enthält, die im Laufzeitsystem noch nicht referenziert sind. Deshalb ist ein Download des gesamten Projekts nötig.
1400	Unbekannte Compilerdirektive '<Name>' wird ignoriert!	Dieses Pragma wird vom Compiler nicht unterstützt. Siehe Stichwort 'Pragma' für unterstützte Direktiven.
1401	Die Struktur '<Name>' enthält keine Elemente.	Die Struktur enthält keine Elemente, Variablen dieses Typs belegen jedoch 1 Byte im Speicher.
	Diese Expression enthält keine Zuweisung. Es wird kein Code generiert.	Das Ergebnis dieses Ausdrucks wird nicht verwendet. Somit wird für den gesamten Ausdruck kein Code generiert.

Nummer	Fehlermeldung	mögliche Ursache
1501	String Konstante wird als VAR_IN_OUT übergeben: '<Name>' darf nicht überschrieben werden!	Die Konstante darf im Rumpf des Bausteins nicht beschrieben werden, da dort keine Größenprüfung möglich ist.
1502	Variable '<Name>' hat den gleichen Namen wie ein Baustein. Der Baustein wird nicht aufgerufen!	Sie verwenden eine Variable, die den gleichen Namen wie ein Baustein trägt. Beispiel: PROGRAM a ... VAR_GLOBAL a: INT; END_VAR ... a; (* Es wird nicht der Baustein a aufgerufen, sondern die Variable a geladen. *)
1503	Der Baustein hat keine Ausgänge, Verknüpfung wird mit TRUE fortgesetzt.	Sie verknüpfen den Ausgangs-Pin eines Bausteins ohne Ausgänge in FUP oder KOP weiter. Die Verknüpfung bekommt automatisch den Wert TRUE zugewiesen.
1504	Anweisung wird möglicherweise nicht ausgeführt, abhängig vom logischen Ausdruck	Unter Umständen werden nicht alle Zweige des logischen Ausdrucks ausgeführt. Beispiel: IF a AND funct(TRUE) THEN .... Wenn a FALSE ist, wird funct nicht mehr aufgerufen.
1505	Seiteneffekt in '<Name>'! Zweig wird möglicherweise nicht gerechnet	Der erste Eingang des Bausteins ist FALSE, deshalb wird der Seitenzweig, der am zweiten Eingang einmündet eventuell nicht mehr berechnet.
1506	Variable '%s' hat den gleichen Namen wie eine lokale Aktion. Die Aktion wird nicht aufgerufen!	Benennen Sie die Variable oder die Aktion um, so dass sichergestellt ist, dass keine gleichen Namen verwendet werden.
1600	Offene DB unklar (Generierter Code kann fehlerhaft sein).	Aus dem Original Siemens Programm geht nicht hervor, welcher Datenbaustein geöffnet ist.
1700	Eingang nicht verbunden.	Sie verwenden im CFC eine Eingangsbox, die nicht weiterverbunden ist. Es wird dafür kein Code erzeugt.
1800	<Name>(Element #<Elementnummer>): Ungültiger Watchausdruck '<Name>'	Das Visualisierungselement enthält einen Ausdruck, der nicht gemonitored werden kann. Prüfen Sie Variablennamen und Platzhalterersetzungen.
1801	Eingabe auf Ausdruck nicht möglich.	Sie verwenden in der Konfiguration des Visualisierungsobjekts einen zusammengesetzten Ausdruck als Ziel einer Eingabeaktion. Ersetzen Sie diesen durch eine einzelne Variable.

Nummer	Fehlermeldung	mögliche Ursache
1900	Die POU '<Name>' (Einsprungfunktion) steht in der Bibliothek nicht zur Verfügung	Der Einsprungsbaustein (z.B. PLC_PRG) wird beim Verwenden der Bibliothek nicht zur Verfügung stehen.
1901	Access Variablen und Konfigurationsvariablen werden nicht in einer Bibliothek abgespeichert!	Access Variablen und Variablenkonfiguration werden nicht in der Bibliothek gespeichert
1902	'<Name>': Bibliothek für den aktuellen Maschinentyp nicht geeignet!	Die .obj-Datei der Bibliothek wurde für einen anderen Maschinentyp erzeugt.
1903	<Name>: Ungültige Bibliothek	Die Datei entspricht nicht dem benötigten Dateiformat des Zielsystems.

Übersetzungsfehler

Nummer	Fehlermeldung	mögliche Ursache
	Programm zu groß. Maximale Größe: '<Anzahl>' Byte (<Anzahl> K)	Die maximale Programmgröße ist überschritten. Verkleinern Sie das Programm.
3101	Datenbereich zu groß. Maximale Größe: '<Anzahl>' Byte (<Anzahl> K)	Der Datenspeicher ist aufgebraucht. Verringern Sie den Dateienbedarf Ihrer Applikation.
3110	Fehler in Bibliotheks-Datei '<Name>'.	Die .hex-Datei entspricht nicht dem INTEL Hex-Format.
3111	Bibliothek '<Name>' ist zu groß. Maximale Größe: 64K	Die .hex-Datei überschreitet die maximal mögliche Größe.
3112	Nicht verschiebbare Anweisung in Bibliothek.	Die .hex-Datei enthält eine nicht relozierbare Anweisung. Der Bibliotheks-Code kann nicht gelinkt werden.
3113	Bibliotheks-Code überschreibt Funktionstabellen.	Die Bereiche für Code und Informationstabellen überlappen sich.
3114	Bibliothek verwendet mehr als ein Segment.	Die in der .hex-Datei enthaltenen Tabellen und Code verwenden mehr als ein Segment.
3115	Konstante kann nicht an VAR_IN_OUT zugewiesen werden. Inkompatible Datentypen.	Das interne Zeigerformat für Stringkonstanten kann nicht in das interne Zeigerformat von VAR_IN_OUT konvertiert werden, weil die Daten als "near" und die Stringkonstanten als "huge" oder "far" definiert definiert sind. Wenn möglich, verändern Sie diese Zielsystemeinstellungen
3120	Aktuelles Code-Segment ist größer als 64K.	Der soeben generierte System-Code ist größer als 64K. Eventuell wird zuviel Initialisierungs-Code benötigt.
3121	Baustein zu groß.	Ein Baustein darf die Größe von 64K nicht überschreiten.
3122	Initialisierung zu groß. Maximale Größe: 64K	Der Initialisierungscode für einen Funktionsbaustein oder eine Struktur darf 64K nicht überschreiten.
3130	Anwendungs-Stack zu klein: '<Anzahl>' DWORD benötigt, '<Anzahl>' DWORD verfügbar.	Die Schachtelungstiefe der Bausteinaufrufe ist zu groß. Vergrößern Sie die Stackgröße in den Zielsystemeinstellungen oder übersetzen Sie das Programm ohne die Projektübersetzungsoption 'Debug'.
3131	Benutzer-Stack zu klein: '<Anzahl>' WORD benötigt, '<Anzahl>' WORD verfügbar.	Wenden Sie sich an Ihren Steuerungshersteller.
3132	System-Stack zu klein: '<Anzahl>' WORD benötigt, '<Anzahl>' WORD verfügbar.	Wenden Sie sich an Ihren Steuerungshersteller.
3150	Parameter %d der Funktion '<Name>' : Das Ergebnis einer IEC-Funktion kann nicht als Stringparameter einer C-Funktion übergeben werden.	Verwenden Sie eine Zwischenvariable, auf die das Ergebnis der IEC-Funktion gelegt wird.
3160	Kann Bibliotheks-Datei '<Name>' nicht öffnen.	Die für eine Bibliothek benötigte Datei '<Name>' kann nicht gefunden werden.
3161	Bibliothek '<Name>' enthält kein Codesegment	Eine .obj-Datei einer Bibliothek muss mindestens eine C-Funktion enthalten. Fügen Sie in die .obj-Datei eine Dummy-Funktion ein, die nicht in der .lib-Datei definiert ist.
3162	Kann Referenz in Bibliothek '<Name>' nicht auflösen (Symbol '<Name>' , Class '<Name>' , Type '<Name>')	Die .obj-Datei enthält eine nicht auflösbare Referenz auf ein anderes Symbol. Prüfen Sie die Einstellungen des C-Compilers.
3163	Unbekannter Referenztyp in Bibliothek '<Name>' (Symbol '<Name>' , Class '<Name>' , Type '<Name>')	Die .obj-Datei enthält einen vom Codegenerator nicht auflösbaren Referenztypen. Prüfen Sie die Einstellungen des C-Compilers.

Nummer	Fehlermeldung	mögliche Ursache
	<Name> (%d): Logischer Ausdruck zu komplex.	Der temporäre Speicher des Zielsystems reicht für die Größe des Ausdrucks nicht aus. Teilen Sie den Ausdruck in mehrere Teilausdrücke mit Zuweisungen auf Zwischenvariablen auf.
3201	<Name> (<Netzwerk>): Ein Netzwerk darf maximal 512 Bytes Code ergeben	Interne Sprünge können nicht aufgelöst werden. Aktivieren Sie die Option "16 bit Sprungoffsets verwenden" in den 68k target settings.
3202	Stacküberlauf bei geschachtelten String/Array/ Struktur Funktionsaufrufen	Sie benutzen einen geschachtelten Funktionsaufruf der Form CONCAT(x, f(i)). Dies kann zu Datenverlust führen. Teilen Sie den Aufruf in zwei Ausdrücke auf.
3203	Zuweisung zu komplex (zu viele benötigte Adressregister)	Teilen Sie die Zuweisung in mehrere auf.
3204	Ein Sprung ist über 32k Bytes lang	Sprungdistanzen dürfen nicht größer als 32767 bytes sein.
3205	Interner Fehler: Zu viele constant strings	In einem Baustein dürfen maximal 3000 Stringkonstanten verwendet werden.
3206	Funktionsblock zu groß	Ein Funktionsblock darf maximal 32767 Bytes Code ergeben.
3207	Array Optimierung	Die Optimierung der Array-Zugriffe schlug fehl, weil innerhalb der Indexberechnung eine Funktion aufgerufen wurde.
3208	Umwandlung ist nicht implementiert	Sie verwenden eine Konvertierungsfunktion, die für den aktuellen Codegenerator nicht implementiert ist.
3209	Operator nicht implementiert	Sie verwenden einen Operator, der für diesen Datentyp beim aktuellen Codegenerator nicht implementiert ist: MIN(string1,string2).
3210	Funktion '<Name>' nicht gefunden	Sie rufen eine Funktion auf, die im Projekt nicht vorhanden ist.
3211	Stringvariable zu oft verwendet	Eine Variable vom Typ String darf beim 68K Codegenerator in einem Ausdruck nur 10 mal verwendet werden.
3250	Real wird auf 8 Bit Controller nicht unterstützt	Das Zielsystem wird derzeit nicht unterstützt.
3251	'date of day' Typen werden auf 8 Bit Controller nicht unterstützt	Das Zielsystem wird derzeit nicht unterstützt.
3252	Stackgröße übersteigt %ld Bytes	Das Zielsystem wird derzeit nicht unterstützt.
3253	Hexfile nicht gefunden: '<Name>'	Das Zielsystem wird derzeit nicht unterstützt.
3254	Aufruf einer externen Bibliotheksfunktion konnte nicht aufgelöst werden.	Das Zielsystem wird derzeit nicht unterstützt.
	Fehler beim Importieren der Access Variablen	Die .exp-Datei enthält einen fehlerhaften Access-Variablen-Abschnitt.
3401	Fehler beim Importieren der Variablen Konfiguration	Die .exp-Datei enthält einen fehlerhaften Konfigurations-Variablen-Abschnitt.
3402	Fehler beim Importieren der globalen Variablen	Die .exp-Datei enthält einen fehlerhaften Abschnitt der globalen Variablen.
3403	<Name> konnte nicht importiert werden	Der Abschnitt in der .exp-Datei für das angegebene Objekt ist fehlerhaft.
3404	Fehler beim Importieren der Task Konfiguration	Der Abschnitt in der .exp-Datei für die Task Konfiguration ist fehlerhaft.
3405	Fehler beim Importieren der Steuerungskonfiguration	Der Abschnitt in der .exp-Datei für die Steuerungskonfiguration ist fehlerhaft.

Nummer	Fehlermeldung	mögliche Ursache
3406	Zwei Schritte mit dem Namen '<Name>' . Der zweite Schritt wurde nicht importiert	Der in der .exp-Datei enthaltene Abschnitt des AS-Bausteins enthält zwei Schritte mit gleichem Namen. Benennen Sie einen der Schritte in der Export-Datei um.
3407	Eingangsschritt '<Name>' nicht gefunden	In der .exp-Datei fehlt der genannte Schritt.
3408	Nachfolgeschritt '<Name>' nicht gefunden	In der .exp-Datei fehlt der genannte Schritt.
3409	Keine nachfolgende Transition für Schritt '<Name>'	In der .exp-Datei fehlt eine Transition, die den genannten Schritt als Eingangsschritt benötigt.
3410	Kein nachfolgender Schritt für Transition '<Name>'	In der .exp-Datei fehlt ein Schritt, der die genannte Transition benötigt.
3411	Schritt '<Name>' nicht erreichbar von Init-Step	In der .exp-Datei fehlt die Verknüpfung zwischen dem genannten Schritt und dem Init-Step.
3450	PDO<Name>: <Modulname> <Konfigurationsdialogname> -<PDO Name> COB-Id fehlt!	Klicken Sie auf die Schaltfläche Eigenschaften im Konfigurationsdialog <Konfigurationsdialogname> des Moduls <Modulname> und geben Sie für die PDO <PDO Name> eine COB ID ein.
3451	Fehler beim Laden: EDS-Datei '<Name>' konnte nicht gefunden werden, wird aber in der Konfiguration verwendet!	Die für die CAN-Konfiguration benötigte Gerätedatei liegt eventuell nicht im richtigen Verzeichnis vor. Prüfen Sie dies anhand des Verzeichniseintrags für Konfigurationsdateien in 'Projekt' 'Optionen' 'Verzeichnisse'.
3452	Das Modul '<Name>' konnte nicht erstellt werden!	Die Gerätedatei für Modul <Name> passt nicht mehr zur vorliegenden Konfiguration. Eventuell wurde sie seit Erstellung der Konfiguration verändert oder ist korrupt.
3453	Der Kanal '<Name>' konnte nicht erstellt werden!	Die Gerätedatei für Kanal <Name> passt nicht mehr zur vorliegenden Konfiguration. Eventuell wurde sie seit Erstellung der Konfiguration verändert oder ist korrupt.
3454	Die Adresse '<Name>' verweist auf einen belegten Speicherbereich!	Sie haben die Option 'Adressüberschneidungen prüfen' im Dialog Einstellungen der Steuerungskonfiguration aktiviert und es wurde eine Überschneidung festgestellt. Beachten Sie, dass die Grundlage der Bereichsprüfung die Größe ist, die sich aufgrund des Datentyps der Module ergibt und nicht der Wert im Eintrag 'size' in der Konfigurationsdatei !
3455	Fehler beim Laden: GSD-Datei '<Name>' konnte nicht gefunden werden, wird aber in der Konfiguration verwendet!	Die für die Profibus-Konfiguration nötige Gerätedatei liegt eventuell nicht im richtigen Verzeichnis vor. Siehe hierzu Eintrag für Konfigurationsdateien in 'Projekt' 'Optionen' 'Verzeichnisse'.
3456	Das Profibus-Gerät '<Name>' konnte nicht erstellt werden!	Die Gerätedatei für das Gerät <Name> paßt nicht mehr zur vorliegenden Konfiguration. Eventuell wurde sie seit Erstellung der Konfiguration verändert oder ist korrupt.
3457	Modulbeschreibung fehlerhaft: '<Name>'!	Prüfen Sie die zum Modul gehörige Gerätedatei.
	Kein VAR_CONFIG für '<Name>'	Fügen Sie für die genannte Variable in der globalen Variablenliste, die die 'Variablen_Konfiguration' enthält, eine Deklaration ein.

Nummer	Fehlermeldung	mögliche Ursache
3501	Keine Adresse in VAR_CONFIG für '<Name>'	Fügen Sie für die genannte Variable in der globalen Variablenliste, die die Variablenkonfiguration enthält, eine Adresse ein.
3502	Falscher Datentyp von '<Name>' in VAR_CONFIG	Die genannte Variable ist in der globalen Variablenliste, die die Variablenkonfiguration enthält, mit einem anderen Datentypen deklariert als im Funktionsbaustein.
3503	Falscher Adresstyp von '<Name>' in VAR_CONFIG	Die genannte Variable ist in der globalen Variablenliste, die die Variablenkonfiguration enthält, mit einem anderen Adresstypen deklariert als im Funktionsbaustein.
3504	"Initialwerte für VAR_CONFIG Variablen werden nicht unterstützt	Eine Variable der Variablen_Konfiguration ist mit Adresse und Initialwert deklariert. Ein Initialwert kann jedoch nur bei Eingangsvariablen ohne Adresszuweisung definiert werden.
3505	<Name> ist kein gültiger Instanzpfad	In der Variablenkonfiguration wurde eine Variable angegeben, die nicht existiert.
3506	Zugriffspfad erwartet	In der globalen Variablenliste für die Access Variablen fehlt für eine Variable ein korrekter Zugriffspfad: <Bezeichner>:'<Zugriffspfad>':<Typ> <Zugriffsart>
3507	Keine Adressangabe für VAR_ACCESS erlaubt	In der globalen Variablenliste für die Access Variablen ist für eine Variable eine Adresszuweisung vorhanden. Gültige Definition: <Bezeichner>:'<Zugriffspfad>':<Typ> <Zugriffsart>
3550	Der Taskname '<Name>' wurde doppelt verwendet	Sie haben zwei Tasks mit demselben Namen definiert. Benennen Sie eine davon um.
3551	Die Task '<Name>' muss mindestens einen Programmaufruf enthalten	Fügen Sie einen Programmaufruf ein oder löschen Sie die Task.
3552	Ereignis-Variable '<Name>' in Task '<Name>' nicht definiert	Sie haben in der Konfiguration der genannten Task eine Ereignisvariable verwendet, die im Projekt nicht global deklariert ist. Verwenden Sie eine andere Variable bzw. definieren Sie die eingetragene Variable global.
3553	Ereignis-Variable '<Name>' in Task '<Name>' muss vom Typ BOOL sein	Verwenden Sie eine Variable vom Typ BOOL als Ereignisvariable.
3554	Taskeintrag '<Name>' muss ein Programm oder eine globale Funktionsblockinstanz sein	Sie haben im Feld Programmaufruf eine Funktion oder einen nicht definierten Baustein eingetragen.
3555	Der Taskeintrag '<Name>' ist falsch parametriert	Sie haben im Feld Programmaufruf Parameter angegeben, die nicht der Deklaration des Bausteins entsprechen.
	Implizite Variable nicht gefunden.	Wenden Sie zunächst den Befehl 'Alles übersetzen' an. Falls die Fehlermeldung erneut erscheint, wenden Sie sich bitte an Ihren Steuerungshersteller.
3601	<Name> ist ein reservierter Variablenname	Sie haben im Projekt eine Variable deklariert, die bereits für den Codegenerator reserviert ist. Benennen Sie diese Variable um.
3610	'<Name>' wird nicht unterstützt	Das angegebene Feature wird von dieser Version nicht unterstützt.

Nummer	Fehlermeldung	mögliche Ursache
3611	Das Übersetzungsverzeichnis '<Name>' ist ungültig	Sie haben in den Projektoptionen/Verzeichnisse ein ungültiges Verzeichnis für die Übersetzungsdateien eingetragen.
3612	Maximale Anzahl der Bausteine (<Anzahl>) überschritten ! Übersetzung wird abgebrochen.	Sie verwenden zu viele Bausteine und Datentypen im Projekt. Verändern Sie die max. Anzahl der Bausteine in den Zielsystemeinstellungen/ Speicheraufteilung.
3613	Übersetzung abgebrochen	Die Übersetzung wurde durch den Benutzer abgebrochen.
3614	Das Projekt enthält keinen Baustein '<Name>' (Einsprungfunktion) oder eine Taskkonfiguration	Ein Projekt benötigt eine Einsprungsfunktion vom Typ Programm (z.B. PLC_PRG) oder eine Taskkonfiguration.
3615	<Name> (Einsprungfunktion) muss vom Typ Programm sein	Sie verwenden eine Einsprungsfunktion (z.B. PLC_PRG), die nicht vom Typ Programm ist.
3616	Programme in externen Bibliotheken werden nicht unterstützt	Die zu speichernde Bibliothek enthält ein Programm. Dieses wird bei der Verwendung der Bibliothek nicht zur Verfügung stehen.
3617	Zu wenig Speicher	Erhöhen Sie die virtuelle Speicherkapazität Ihres Rechners.
3618	Bitzugriffe werden vom aktuellen Codegenerator nicht unterstützt	Der Codegenerator für das aktuell eingestellte Zielsystem unterstützt keine Bitzugriffe auf Variablen.
	Ein Baustein mit Namen '<Name>' ist bereits in Bibliothek '<Name>'	Sie verwenden einen Bausteinnamen, der bereits für einen Bibliotheksbaustein vergeben ist. Benennen Sie den Baustein um.
3701	Der Bausteinname im Deklarationsnamen stimmt nicht mit dem in der Objektliste überein	Benennen Sie Ihren Baustein mittels des Menübefehls 'Projekt' 'Objekt umbenennen' neu, oder ändern Sie den Namen des Bausteins in dessen Deklarationsteil. Der Name muss direkt nach den Schlüsselwörtern PROGRAM, FUNCTION oder FUNCTIONBLOCK stehen.
3702	Zu viele Bezeichner	Pro Variablendeklaration können maximal 100 Bezeichner angegeben werden.
3703	Mehrere Deklarationen mit dem gleichen Bezeichner '<Name>'	Im Deklarationsteil des Objekts existieren mehrere Bezeichner mit dem gleichen Namen.
3704	Datenrekursion: <Baustein 0> -> <Baustein 1> -> .. -> <Baustein 0>	Eine FB-Instanz wurde verwendet, die sich selbst wieder benötigt.
3720	Nach 'AT' muss eine Adresse stehen	Fügen Sie eine gültige Adresse nach dem Schlüsselwort AT ein, oder ändern Sie das Schlüsselwort AT.
3721	Nur VAR und VAR_GLOBAL dürfen auf Adressen gelegt werden	Kopieren Sie die Deklaration in einen VAR oder VAR_GLOBAL-Bereich.
3722	Auf der angegebenen Adresse dürfen nur einfache boolsche Variablen stehen	Ändern Sie die Adresse oder den in der Deklaration angegebenen Typ der Variablen.
3729	Unzulässiger Typ '<Name>' auf Adresse: '<Name>'	Der Typ dieser Variable kann auf der angegebenen Adresse nicht platziert werden. Beispiel: Für ein Zielsystem, das mit Alignment 2 arbeitet, ist folgende Deklaration ungültig: var1 AT %IB1:WORD;
3740	Unbekannter Typ: '<Name>'	Sie verwenden zur Variablendeklaration einen ungültigen Typen.
3741	Typbezeichner erwartet	Sie verwenden ein Schlüsselwort oder einen Operator anstelle eines gültigen Typbezeichners.

Nummer	Fehlermeldung	mögliche Ursache
3742	Aufzählungswert erwartet	In der Definition des Enumerationstyps fehlt ein Bezeichner nach der öffnenden Klammer oder nach einem Komma innerhalb des Klammerbereichs.
3743	Ganze Zahl erwartet	Enumerationswerte können nur mit ganzen Zahlen vom Typ INT initialisiert werden.
3744	Enum-Konstante '<Name>' bereits definiert	Prüfen Sie, ob Sie folgende Regeln bei der Vergabe von Enumerationswerten beachtet haben: - Innerhalb einer Enum-Definition müssen alle Werte eindeutig sein. - Innerhalb aller globalen Enum-Definitionen müssen alle Werte eindeutig sein. - Innerhalb aller lokalen Enum-Definitionen eines Bausteins müssen alle Werte eindeutig sein.
3745	Bereichsgrenzen sind nur für Integer-Datentypen erlaubt!	Unterbereichstypen können nur auf Basis von Integer-Datentypen definiert werden.
3746	Bereichsgrenze '<Name>' nicht kompatibel zu Datentyp '<Name>'	Eine Grenze des für den Unterbereichstypen angegebenen Bereichs liegt außerhalb der für den Basistypen erlaubten Grenzen.
3747	Unbekannte Stringlänge: '<Name>'	Sie verwenden eine unbekannte Konstante zur Definition der Stringlänge.
3748	Mehr als 3 Dimensionen sind für ein Array unzulässig	Sie verwenden mehr als die zulässigen 3 Dimensionen für ein Array. Verwenden Sie gegebenenfalls ein ARRAY OF ARRAY.
3749	Untergrenze '<Name>' unbekannt	Sie verwenden eine nicht definierte Konstante als Untergrenze eines Unterbereichs- oder Array-Typen.
3750	Obergrenze '<Name>' unbekannt	Sie verwenden eine nicht definierte Konstante als Obergrenze eines Unterbereichs- oder Array-Typen.
3760	Fehlerhafter Initialwert	Verwenden Sie einen Initialwert, welcher der Typdefinition entspricht. Sie können den Variablendeklarationsdialog zu Hilfe nehmen (Shift/F2 oder, 'Bearbeiten''Variablendeklaration').
3761	VAR_IN_OUT Variablen dürfen keinen Initialwert haben.	Entfernen Sie die Initialisierung bei der Deklaration der Variablen.
3780	VAR, VAR_INPUT, VAR_OUTPUT oder VAR_IN_OUT erwartet	Die erste Zeile nach dem Namen eines Bausteines muss eines dieser Schlüsselwörter beinhalten.
3781	END_VAR oder Bezeichner erwartet	Schreiben Sie einen gültigen Bezeichner oder END_VAR an den Anfang der Deklarationszeile.
3782	Unerwartetes Ende	Im Deklarationsteil: Fügen Sie am Ende des Deklarationsteils das Schlüsselwort END_VAR ein. Im Texteditor: Fügen Sie Anweisungen ein, die die letzte Anweisungssequenz beenden (z.B. END_IF).
3783	END_STRUCT oder Bezeichner erwartet	Stellen Sie sicher, dass die Typdeklaration richtig abgeschlossen ist.
	Die globalen Variablen brauchen zu viel Speicher. Erhöhen Sie den verfügbaren Speicher in den Projektoptionen	Vergrößern Sie die in den Projektoptionen im Bereich Übersetzungsoptionen eingestellte Anzahl der Segmente.

Nummer	Fehlermeldung	mögliche Ursache
3801	Die Variable '<Name>' ist zu groß. (<Größe> Byte)	Die Variable verwendet einen Typen, der größer als 1 Datensegment ist. Die Segmentgröße lässt sich, abhängig vom Zielsystem, in den Zielsystemeinstellungen/ Speicheraufteilung einstellen. Sollten Sie dort keine Eingabemöglichkeit finden, wenden Sie sich bitte an Ihren Steuerungshersteller.
3802	Speicher für Retainvariable aufgebraucht. Variable '<Name>', %u Byte.	Der verfügbare Speicherplatz für Retain-Variablen ist erschöpft. Er lässt sich, abhängig vom Zielsystem, in den Zielsystemeinstellungen/ Speicheraufteilung einstellen. Sollten Sie dort keine Eingabemöglichkeit finden, wenden Sie sich bitte an Ihren Steuerungshersteller. (Beachten Sie hierzu auch, dass bei Instanzen von Funktionsblöcken, in denen eine Retain-Variable verwendet wird, die gesamte Instanz im Retain-Speicher verwaltet wird!)
3803	Speicher für globale Variablen aufgebraucht. Variable '<Name>', '<Anzahl>' Byte.	Der verfügbare Speicherplatz für globale Variablen ist erschöpft. Er lässt sich, abhängig vom Zielsystem, in den Zielsystemeinstellungen/ Speicheraufteilung einstellen. Sollten Sie dort keine Eingabemöglichkeit finden, wenden Sie sich bitte an Ihren Steuerungshersteller.
3820	VAR_OUTPUT und VAR_IN_OUT ist in Funktionen nicht erlaubt.	Sie dürfen in einer Funktion keine Ausgangs-/ Referenzparameter definieren.
3821	Zumindest ein Input bei einer Funktion erforderlich	Sie dürfen in einer Funktion keine Ausgangs-/ Referenzparameter definieren.
3840	Unbekannte globale Variable '<Name>'!	Sie verwenden im Baustein eine VAR_EXTERNAL Variable, für die keine entsprechende globale Variable deklariert ist.
3841	Deklaration von '<Name>' stimmt nicht mit globaler Deklaration überein!	Die Typangabe in der Deklaration der VAR_EXTERNAL Variable stimmt nicht mit derjenigen in der globalen Deklaration überein.
	Mehrfache Unterstriche im Bezeichner	Entfernen Sie mehrfache Unterstriche im Bezeichnernamen.
3901	Es sind maximal 4 Adressfelder zulässig	Sie verwenden eine direkte Adresszuweisung auf eine Adresse, die mehr als vier Stufen enthält (z.B. %QB0.1.1.0.1).
3902	Schlüsselwörter müssen groß geschrieben werden	Schreiben Sie das Schlüsselwort in Großbuchstaben bzw. aktivieren Sie die Option 'Automatisch formatieren'.
3903	Ungültige Zeitkonstante	Die Konstante ist nicht entsprechend dem IEC61131-3 Format angegeben.
3904	Überlauf in Zeitkonstante	Sie verwenden einen Wert für die Zeitkonstante, der im internen Format nicht mehr darstellbar ist. Der maximal darstellbare Wert ist t#49d17h2m47s295ms.
3905	Ungültige Datumskonstante	Die Konstante ist nicht entsprechend dem IEC61131-3 Format angegeben.
3906	Ungültige Tageszeitkonstante	Die Konstante ist nicht entsprechend dem IEC61131-3 Format angegeben.
3907	Ungültige Datum/Zeit-Konstante	Die Konstante ist nicht entsprechend dem IEC61131-3 Format angegeben.
3908	Ungültige Stringkonstante	Die Stringkonstante enthält ein ungültiges Zeichen.

Nummer	Fehlermeldung	mögliche Ursache
	Bezeichner erwartet	Geben Sie an dieser Stelle einen gültigen Bezeichner an.
4001	Variable '<Name>' nicht deklariert	Deklarieren Sie die Variable lokal oder global.
4010	Unverträgliche Typen: Kann '<Name>' nicht in '<Name>' konvertieren.	Prüfen Sie die erforderlichen Typen des Operators (Suchen Sie dafür den Operator in Ihrer Hilfe-Datei), und ändern Sie den Typ der Variable, die den Fehler produziert hat, in einen erlaubten Typ, oder wählen Sie eine andere Variable.
4011	Unzulässiger Typ in Parameter <Parameter> von '<Name>' : Kann '<Name>' nicht in '<Name>' konvertieren.	Der Typ des Aktual-Parameters kann nicht in den des Formal-Parameters überführt werden. Verwenden Sie eine Typkonvertierung oder verwenden Sie einen entsprechenden Variablentypen.
4012	Unzulässiger Typ für Eingang '<Name>' von '<Name>' : Kann '<Name>' nicht in '<Name>' konvertieren.	Der Variablen '<Name>' wird ein Wert mit dem unzulässigen Typ <Typ2> zugewiesen. Ändern Sie die Variable oder die Konstante zu einer Variablen oder Konstanten mit dem Typ <Typ1> oder verwenden Sie eine Typkonvertierung bzw. eine Konstante mit Typ-Präfix.
4013	Unzulässiger Typ für Ausgang '<Name>' von '<Name>' : Kann '<Name>' nicht in '<Name>' konvertieren.	Der Variablen '<Name>' wird ein Wert mit dem unzulässigen Typ <Typ2> zugewiesen. Ändern Sie die Variable oder die Konstante zu einer Variablen oder Konstanten mit dem Typ <Typ1> oder verwenden Sie eine Typkonvertierung bzw. eine Konstante mit Typ-Präfix.
4014	Konstante mit Typ-Präfix: '<Name>' kann nicht nach '<Name>' konvertiert werden	Der Typ der Konstanten ist nicht kompatibel mit dem Typen des Präfix. Beispiel: SINT#255
4015	Unzulässiger Datentyp '<Name>' für direkten Bitzugriff	Direkte Bitadressierung ist nur für Integer- und Bitstring-Datentypen zulässig. Sie verwenden im Bitzugriff <var1>.<bit> eine Variable var1 vom Typ REAL/LREAL oder eine Konstante.
4016	Bitindex '<%d>' außerhalb des gültigen Bereichs für Variable des Typs '<Name>'	Sie versuchen, auf ein Bit zuzugreifen, das für den Datentyp der Variable nicht definiert ist.
4017	MOD ist für REAL nicht definiert	Der Operator MOD kann nur für Integer- und Bitstring-Datentypen verwendet werden.
4020	Operanden von ST, STN, S, R müssen Variable mit Schreibzugriff sein	Ersetzen Sie den ersten Operanden durch eine Variable, auf die geschrieben werden kann.
4021	Kein Schreibzugriff auf '<Name>'	Ersetzen Sie die Variable durch eine mit Schreibzugriff.
4022	Operand erwartet	Ergänzen Sie einen Operanden hinter dem bestehenden Befehl.
4023	Nach '+' bzw. '-' wird eine Zahl erwartet	Geben Sie eine Zahl ein.
4024	Erwarte <Operator 0> oder <Operator 1> oder ... vor '<Name>'	Geben Sie an der genannten Stelle einen gültigen Operator ein.
4025	Erwarte ':=' oder '=>' vor '<Name>'	Geben Sie an der genannten Stelle einen der beiden Operatoren ein.
4026	BITADR erwartet eine Bitadresse oder eine Variable auf einer Bitadresse	Verwenden Sie eine gültige Bitadresse (z.B. %IX0.1).
4027	Ganze Zahl oder symbolische Konstante erwartet	Fügen Sie eine ganze Zahl oder den Bezeichner einer gültigen Konstante ein.
4028	INI Operator benötigt eine Funktionsblockinstanz oder eine Strukturvariable	Prüfen Sie den Typen der Variablen, auf den Sie den INI Operator anwenden.

Nummer	Fehlermeldung	mögliche Ursache
4029	Ineinander verschachtelte Aufrufe derselben Funktion sind nicht möglich.	Bei nicht reentranten Zielsystemen und im Simulationsmodus darf ein Funktionsaufruf keinen Aufruf auf sich selbst als Parameter enthalten. Beispiel: fun1(a,fun1(b,c,d),e); Verwenden Sie eine Zwischenvariable.
4030	Als Operanden zu ADR sind keine Konstanten und Ausdrücke erlaubt	Ersetzen Sie die Konstante oder den Ausdruck durch eine Variable oder eine direkte Adresse.
4031	Der Adressoperator ist auf Bits nicht erlaubt! Verwenden Sie stattdessen BITADR	Verwenden Sie BITADR. Beachten Sie: Der BITADR liefert keine physikalische Speicheradresse
4032	'<Anzahl>' Operanden sind zu wenige für '<Name>'. Es werden mindestens '<Anzahl>' benötigt	Überprüfen Sie, wie viele Operanden der Operator '<Name>' benötigt, und fügen Sie die fehlenden ein.
4033	'<Anzahl>' Operanden sind zu viele für '<Name>'. Es werden genau '<Anzahl>' benötigt	Überprüfen Sie, wie viele Operanden der Operator '<Name>' benötigt, und entfernen Sie die überzähligen.
4034	Division durch 0	Sie verwenden eine Division durch 0 in einem konstanten Ausdruck. Verwenden Sie gegebenenfalls eine Variable mit dem Wert 0 um einen Laufzeitfehler zu erzwingen.
4035	ADR darf nicht auf 'VAR CONSTANT' angewendet werden, wenn 'Konstanten ersetzen' aktiviert ist	Ein Adresszugriff auf Konstanten, für die die direkten Werte verwendet werden, ist nicht möglich. Deaktivieren Sie gegebenenfalls die Option 'Konstanten ersetzen' in den Projektoptionen, Kategorie Übersetzungsoptionen.
4040	Sprungmarke <LabelName> ist nicht definiert	Definieren Sie eine Marke mit dem Namen <LabelName> oder ändern Sie <LabelName> in eine definierte Marke.
4041	Mehrfache Definition der Sprungmarke '<Name>'	Die Sprungmarke '<Name>' ist im Baustein mehrfach definiert. Benennen Sie entsprechend um oder entfernen Sie eine Definition.
4042	Es dürfen höchstens '<Anzahl>' Sprungmarken in Folge stehen	Die Anzahl der Sprungmarken pro Anweisung ist auf '<Anzahl>' begrenzt. Fügen Sie eine Dummy-Anweisung ein.
4043	Labelformat ungültig. Ein Label muss ein Bezeichner sein, dem ein Doppelpunkt folgen kann.	Der für das Label verwendete Name ist entweder kein gültiger Bezeichner oder es fehlt der Doppelpunkt bei der Definition.
4050	Baustein '<Name>' existiert nicht im Projekt	Definieren Sie einen Baustein mit dem Namen '<Name>' durch die Menübefehle 'Projekt' 'Objekt anfügen' oder ändern Sie '<Name>' in den Namen eines definierten Bausteins
4051	'<Name>' ist keine Funktion	Verwenden Sie für '<Name>' einen der im Projekt oder den Bibliotheken definierten Funktionsnamen.
4052	'<Instanzname>' muss eine deklarierte Instanz des Funktionsblocks '<Name>' sein	Verwenden Sie für <Instanzname> eine im Projekt definierte Instanz des Typs '<Name>' oder ändern Sie den Typen von <Instanzname> auf '<Name>' .
4053	<Name> ist kein gültiger Baustein oder Operator	Ersetzen Sie '<Name>' durch den Namen eines im Projekt definierten Bausteins oder eines Operators.
4054	Bausteinname als Parameter von 'INDEXOF' erwartet	Der angegebene Parameter ist kein gültiger Bausteinname.

Nummer	Fehlermeldung	mögliche Ursache
4060	VAR_IN_OUT Parameter '<Name>' von '<Name>' benötigt Variable mit Schreibzugriff als Eingabe.	An VAR_IN_OUT Parameter müssen Variable mit Schreibzugriff übergeben werden, da diese innerhalb des Bausteins modifiziert werden können.
4061	VAR_IN_OUT Parameter '<Name>' von '<Name>' muss belegt werden.	VAR_IN_OUT Parameter müssen mit Variablen mit Schreibzugriff belegt werden, da diese innerhalb des Bausteins modifiziert werden können.
4062	Kein Zugriff auf VAR_IN_OUT Parameter '<Name>' von '<Name>' von außen.	VAR_IN_OUT Parameter dürfen nur innerhalb des Bausteins beschrieben oder gelesen werden, da es sich um eine Übergabe über Referenz handelt.
4063	VAR_IN_OUT Parameter '<Name>' von '<Name>' kann nicht mit Bitadressen belegt werden.	Eine Bitadresse ist keine gültige physikalische Adresse. Übergeben Sie eine Variable oder eine direkte Nicht-Bitadresse.
4064	VAR_IN_OUT darf in lokalem Aktionsaufruf nicht überschrieben werden!	Löschen Sie die Belegung der VAR_IN_OUT Variablen für den lokalen Aktionsaufruf.
4070	Ein Baustein enthält einen zu tief geschachtelten Ausdruck.	Verkleinern Sie die Schachtelungstiefe, indem Sie mit Hilfe von Zuweisungen auf Zwischenvariablen den Ausdruck auf mehrere Ausdrücke umverteilen.
4071	Netzwerk ist zu groß	Teilen Sie das Netzwerk in mehrere Netzwerke auf.
	'^' benötigt einen Pointertyp	Sie versuchen, eine Variable zu dereferenzieren, die nicht als POINTER TO deklariert ist.
4110	'[<index>]' ist nur für Arrayvariablen zulässig	Sie verwenden [<index>] für eine Variable, die nicht als ARRAY OF deklariert ist.
4111	Der Ausdruck im Index eines Arrays muss ein Ergebnis vom Typ INT haben	Verwenden Sie einen Ausdruck des entsprechenden Typs oder eine Typkonvertierung.
4112	Zu viele Array-Indizes	Überprüfen Sie die Zahl der Indizes (1, 2, oder 3), für die das Array deklariert ist und entfernen Sie die überzähligen.
4113	Zu wenig Array-Indizes	Überprüfen Sie die Zahl der Indizes (1, 2, oder 3), für die das Array deklariert ist und ergänzen Sie die fehlenden.
4114	Ein konstanter Index liegt nicht im Array-Bereich	Stellen Sie sicher, dass die verwendeten Indizes innerhalb der Grenzen des Arrays liegen.
4120	Vor dem '.' muss eine Strukturvariable stehen	Der Bezeichner links vom Punkt muss eine Variable vom Typ STRUCT oder FUNCTION_BLOCK sein oder der Name einer FUNCTION oder eines PROGRAM sein.
4121	'<Name>' ist keine Komponente von <Objektname>	Die Komponente '<Name>' ist in der Definition des Objekts <Objektname> nicht enthalten.
4122	<Name> ist kein Eingabeparameter des aufgerufenen Funktionsblocks	Überprüfen Sie die Eingabevariablen des aufgerufenen Funktionsblocks und ändern Sie '<Name>' in eine dieser Variablen.
	'LD' erwartet	Fügen Sie im Editorfenster des AWL-Bausteins bzw. nach der Sprungmarke zumindest eine LD-Anweisung ein.
4201	AWL Operator erwartet	Jede AWL-Anweisung muss mit einem Operator oder einer Sprungmarke beginnen.
4202	Unerwartetes Ende des Klammerausdrucks	Fügen Sie die schließende Klammer ein.

Nummer	Fehlermeldung	mögliche Ursache
4203	<Name> in Klammern ist nicht zulässig	Der angegebene Operator ist innerhalb eines AWL-Klammerausdrucks nicht zulässig. (nicht zulässig sind: 'JMP', 'RET', 'CAL', 'LDN', 'LD', 'TIME')
4204	Schließende Klammer ohne zugehörige öffnende Klammer	Fügen Sie die öffnende Klammer ein oder löschen Sie die schließende.
4205	Nach ')' ist kein Komma zulässig	Entfernen Sie das Komma nach der schließenden Klammer.
4206	Keine Sprungmarken innerhalb von Klammerausdrücken	Verschieben Sie die Sprungmarke so, dass sie außerhalb des Klammerausdrucks liegt.
4207	'N' Modifikator verlangt einen Operanden vom Typ BOOL,BYTE,WORD or DWORD	Der N-Modifikator benötigt einen Datentypen, für den eine boolsche Negation ausgeführt werden kann.
4208	Der Ausdruck vor einem bedingten Befehl muss ein Ergebnis vom Typ BOOL liefern	Stellen Sie sicher, dass der Ausdruck ein boolsches Ergebnis liefert oder verwenden Sie eine Typkonvertierung.
4209	An dieser Stelle ist kein Funktionsname zulässig	Tauschen Sie den Funktionsaufruf gegen eine Variable oder eine Konstante aus.
4210	'CAL', 'CALC' und 'CALN' benötigen eine Funktionsblockinstanz als Operanden	Deklarieren Sie eine Instanz des Funktionsblocks, den Sie aufrufen möchten.
4211	Kommentar ist in AWL nur am Zeilenende zulässig	Verschieben Sie den Kommentar ans Zeilenende oder in eine eigene Zeile.
4212	Akkumulator ist ungültig vor bedingter Anweisung	Der Inhalt des Akkumulators ist nicht definiert. Dies ist der Fall nach Anweisungen, die kein Ergebnis liefern (z.B. 'CAL').
4213	'S' und 'R' verlangen einen Operanden vom Typ BOOL	Verwenden Sie an dieser Stelle eine boolsche Variable.
4250	Kein korrekter Anfang für eine ST Anweisung	Die Zeile beginnt nicht mit einer gültigen ST-Anweisung.
4251	Funktion '<Name>' hat zu viele Parameter	Es wurden mehr Parameter angegeben, als in der Funktionsdefinition deklariert sind.
4252	Funktion '<Name>' hat zu wenige Parameter	Es wurden weniger Parameter angegeben, als in der Funktionsdefinition deklariert sind.
4253	'IF' und 'ELSIF' benötigen als Bedingung einen Boolschen Ausdruck	Stellen Sie sicher, dass die Bedingung, die einem 'IF' folgt, ein boolescher Ausdruck ist.
4254	'WHILE' benötigt als Bedingung einen Boolschen Ausdruck	Stellen Sie sicher, dass die Bedingung, die einem 'WHILE' folgt, ein boolescher Ausdruck ist.
4255	'UNTIL' benötigt als Bedingung einen Boolschen Ausdruck	Stellen Sie sicher, dass die Bedingung, die einem 'UNTIL' folgt, ein boolescher Ausdruck ist.
4256	'NOT' verlangt einen boolschen Operanden	Stellen Sie sicher, dass die Bedingung, die einem 'NOT' folgt, ein boolescher Ausdruck ist.
4257	Der Zähler der 'FOR' Anweisung muss vom Typ INT sein	Stellen Sie sicher, dass die Zählvariable ein Integer- oder Bitstring Datentyp ist (z.B. DINT, DWORD).
4258	Der Zähler in der 'FOR' Anweisung ist keine Variable mit Schreibzugriff	Ersetzen Sie die Zählvariable durch eine Variable mit Schreibzugriff.
4259	Der Startwert der 'FOR' Anweisung muss vom Typ INT sein	Der Startwert der 'FOR' Anweisung muss kompatibel zum Typen der Zählvariable sein.
4260	Der Endwert der 'FOR' Anweisung muss vom Typ INT sein	Der Endwert der 'FOR' Anweisung muss kompatibel zum Typen der Zählvariable sein.
4261	Der Inkrementationswert der 'FOR' Anweisung muss vom Typ INT sein	Der Inkrementationswert der 'FOR' Anweisung muss kompatibel zum Typen der Zählvariable sein.

Nummer	Fehlermeldung	mögliche Ursache
4262	'EXIT' ist nur innerhalb einer Schleife erlaubt	Verwenden Sie 'EXIT' nur innerhalb von 'FOR', 'WHILE' oder 'UNTIL' Anweisungen.
4263	Zahl, 'ELSE' oder 'END_CASE' erwartet	Innerhalb eines 'CASE' können nur eine Zahl oder eine 'ELSE' Anweisung angegeben werden oder die Endanweisung 'END_CASE'.
4264	Der Selector der CASE-Anweisung muss vom Typ INT sein	Stellen Sie sicher, dass der Selektor ein Integer- oder Bitstring Datentyp ist (z.B. DINT, DWORD).
4265	Nach ',' wird eine Zahl erwartet	In der Aufzählung der CASE Selektoren muss nach einem Komma ein weiterer Selektor angegeben werden.
4266	Mindestens eine Anweisung ist erforderlich	Geben Sie eine Anweisung ein, mindestens einen Strichpunkt.
4267	Ein Funktionsbausteinaufruf muss mit dem Namen einer Instanz beginnen	Der Kennzeichner im Funktionsbausteinaufruf ist keine Instanz. Deklarieren Sie eine Instanz des gewünschten Funktionsbausteins bzw. verwenden Sie den Namen einer bereits deklarierten Instanz.
4268	Es wird ein Ausdruck erwartet	An dieser Stelle muss ein Ausdruck eingegeben werden.
4269	Nach 'ELSE'-Zweig wird 'END_CASE' erwartet	Schließen Sie die 'CASE' Anweisung nach dem 'ELSE' Zweig mit einem 'END_CASE' ab.
4270	'CASE'-Konstante '%ld' wird bereits verwendet	Ein 'CASE' Selektor darf innerhalb einer 'CASE'-Anweisung nur einmal verwendet werden.
4271	Die Untergrenze des angegebenen Bereichs ist größer als die Obergrenze.	Korrigieren Sie die Selektoren-Bereichsgrenzen so, dass die Untergrenze nicht größer als die Obergrenze ist.
4272	Erwarte Parameter '<Name>' an Stelle <Position> im Aufruf von '<Name>'	Wenn Sie die Funktionsparameter im Funktionsaufruf mit Angabe der Parameternamen vornehmen, muss dennoch zusätzlich die Position der Parameter (Reihenfolge) mit der in der Funktionsdefinition vorzufindenden übereinstimmen.
4273	CASE-Bereich '<Bereichsgrenzen>' überschneidet sich mit bereits verwendetem Bereich '<Bereichsgrenzen>'	Stellen Sie sicher, dass sich die in der CASE-Anweisung angegebenen Selektoren-Bereiche nicht überschneiden.
4274	Mehrfacher 'ELSE'-Zweig in CASE-Anweisung	Eine CASE-Anweisung darf nicht mehr als einen 'ELSE' Zweig enthalten.
	Sprung bzw. Return benötigen eine boolsche Eingabe	Stellen Sie sicher, dass der Eingang für den Sprung bzw. die Return-Anweisung ein boolescher Ausdruck ist.
4301	Baustein '<Name>' verlangt genau '<Anzahl>' Eingänge	Die Anzahl der Eingänge entspricht nicht der Anzahl der in der Bausteindefinition angegebenen VAR_INPUT und VAR_IN_OUT Variablen.
4302	Baustein '<Name>' verlangt genau '<Anzahl>' Ausgänge	Die Anzahl der Eingänge entspricht nicht der Anzahl der in der Bausteindefinition angegebenen VAR_OUTPUT Variablen.
4303	<Name> ist kein Operator	Ersetzen Sie '<Name>' durch einen gültigen Operator.
4320	Nicht boolescher Ausdruck '<Name>' bei Kontakt benutzt	Das Schaltsignal für einen Kontakt muss ein boolescher Ausdruck sein.
4321	Nicht boolescher Ausdruck '<Name>' bei Spule benutzt	Die Ausgangsvariable einer Spule muss vom Typ BOOL sein.

Nummer	Fehlermeldung	mögliche Ursache
4330	Es wird ein Ausdruck erwartet bei Eingang 'EN' des Bausteins '<Name>'	Beschalten Sie den Eingang EN des Bausteins '<Name>' mit einem Eingang oder einem Ausdruck.
4331	Es wird ein Ausdruck erwartet bei Eingang '<Anzahl>'%d des Bausteins '<Name>'	Der Eingang des Operatorbausteins ist nicht beschaltet.
4332	Es wird ein Ausdruck erwartet bei Eingang '<Name>' des Bausteins '<Name>'	Der Eingang des Bausteins ist vom Typ VAR_IN_OUT und ist nicht beschaltet.
4333	Bezeichner in Sprung erwartet	Das angegebene Sprungziel ist kein gültiger Bezeichner.
4334	Es wird ein Ausdruck erwartet beim Eingang des Sprungs	Beschalten Sie den Eingang des Sprungs mit einem boolschen Ausdruck. Wenn dieser TRUE ist, wird der Sprung ausgeführt.
4335	Es wird ein Ausdruck erwartet beim Eingang von Return	Beschalten Sie den Eingang der Return-Anweisung mit einem boolschen Ausdruck. Wenn dieser TRUE ist, wird der Sprung ausgeführt.
4336	Es wird ein Ausdruck erwartet beim Eingang des Ausgangs	Verknüpfen Sie den Ausgang mit Ausdruck, der diesem Ausgang zugewiesen werden kann.
4337	Bezeichner für Eingang erwartet	Fügen Sie in der Eingangsbox einen gültigen Ausdruck oder Bezeichner ein.
4338	Baustein '<Name>' hat keine echten Eingänge	Keiner der Eingänge des Operatorbausteins '<Name>' ist mit einem gültigen Ausdruck beschaltet.
4339	Unverträgliche Typen bei Ausgang: Kann '<Name>' nicht in '<Name>' konvertieren.	Der Ausdruck in der Ausgangsbox ist nicht typkompatibel mit dem Ausdruck, der ihm zugewiesen werden soll.
4340	Sprung benötigt eine boolesche Eingabe	Stellen Sie sicher, dass der Eingang für den Sprung ein boolescher Ausdruck ist.
4341	Return benötigt eine boolesche Eingabe	Stellen Sie sicher, dass der Eingang für die Return-Anweisung ein boolescher Ausdruck ist.
4342	Eingang 'EN' der Box benötigt eine boolsche Eingabe	Verknüpfen Sie den EN-Eingang des Bausteins mit einem gültigen boolschen Ausdruck.
4343	Konstantenbelegung: Unzulässiger Typ für Parameter '<Name>' von '<Name>': Kann '<Typ>' nicht in '<Typ>' konvertieren.	Sie haben Eingang '<Name>' von Baustein '<Name>' als VAR_INPUT CONSTANT deklariert. Sie haben diesem im Dialog 'Parameter bearbeiten' jedoch einen Ausdruck zugewiesen, der nicht typkompatibel ist.
4344	'S' und 'R' benötigen boolsche Operanden	Setzen Sie hinter der Set- bzw. Reset-Anweisung einen gültigen boolschen Ausdruck ein.
4345	Unzulässiger Typ für Parameter '<Name>' von '<Name>': Kann '<Typ>' nicht in '<Typ>' konvertieren.	Sie haben Eingang '<Name>' von Baustein '<Name>' einen Ausdruck zugewiesen, der nicht typkompatibel ist.
4346	Ein Ausgang darf keine Konstante sein	Das Ziel einer Zuweisung muss eine Variable oder direkte Adresse mit Schreibzugriff sein.
4347	VAR_IN_OUT Parameter benötigt Variable mit Schreibzugriff	An VAR_IN_OUT Parameter müssen Variable mit Schreibzugriff übergeben werden, da diese innerhalb des Bausteins modifiziert werden können.
4350	Eine AS-Aktion kann nicht von außerhalb aufgerufen werden	AS-Aktionen können nur innerhalb des AS-Bausteins aufgerufen werden, in dem sie definiert sind.
4351	Der Schrittname ist kein zulässiger Bezeichner: '<Name>'	Benennen Sie den Schritt um und wählen Sie für den Namen einen gültigen Bezeichner.
4352	Unzulässige Zeichen folgen dem zulässigen Schrittnamen:'<Name>'	Löschen Sie die unzulässigen Zeichen im Schrittnamen.

Nummer	Fehlermeldung	mögliche Ursache
4353	Schrittnamen sind doppelt: '<Name>'	Benennen Sie einen der Schritte um.
4354	Sprung auf nicht definierten Schritt: '<Name>'	Wählen Sie als Sprungziel einen vorhandenen Schrittnamen bzw. fügen Sie einen Schritt mit dem noch nicht definierten Namen ein.
4355	Eine Transition darf keine Seiteneffekte (Zuweisungen, FB-Aufrufe etc.) haben	Eine Transition darf nur einen boolschen Ausdruck enthalten.
4356	Sprung ohne gültige Schrittnamen: '<Name>'	Verwenden Sie einen gültigen Bezeichner als Sprungziel.
4357	Die IEC-Bibliothek wurde nicht gefunden	Prüfen Sie, ob im Bibliotheksverwalter die Bibliothek iecsfc.lib eingebunden wurde und ob die in den Projektoptionen eingetragenen Bibliothekspfade korrekt sind.
4358	Nicht deklarierte Aktion: '<Name>'	Sorgen Sie dafür, dass die Aktion des IEC-Schritts im Object Organizer unterhalb des AS-Bausteins eingefügt ist und der Aktionsname im Kästchen rechts vom Qualifizierer eingetragen ist.
4359	Ungültiger Qualifizierer: '<Name>'	Tragen Sie für die IEC-Aktion im Kästchen links neben dem Aktionsnamen einen Qualifizierer ein.
4360	Erwarte Zeitkonstante nach Qualifizierer: '<Name>'	Tragen Sie für die IEC-Aktion im Kästchen links neben dem Aktionsnamen hinter dem Qualifizierer eine Zeitkonstante ein.
4361	Bezeichner '<Name>' bezeichnet keine Aktion	Tragen Sie für die IEC-Aktion im Kästchen rechts neben dem Qualifizierer den Namen einer im Projekt definierten Aktion oder boolsche Variable ein.
4362	Nicht boolescher Ausdruck in Aktion: '<Name>'	Geben Sie eine boolsche Variable oder einen gültigen Aktionsnamen ein.
4363	IEC-Schrittname bereits für Variable verwendet: '<Name>'	Benennen Sie entweder den Schritt oder die Variable um.
4364	Eine Transition muss ein boolescher Ausdruck sein	Das Ergebnis des Transitionsausdrucks muss vom Typ BOOL sein.
4365	Schritt '<Name>' hat fehlerhaften Zeitgrenzenwert	Öffnen Sie den Dialog Schrittattribute für den Schritt '<Name>' und tragen Sie gültige Zeitvariablen oder -konstanten ein.
4366	Die Marke für den Parallelschritt ist kein zulässiger Bezeichner: '<Name>'	Tragen Sie neben dem Dreieck, das die Sprungmarke anzeigt, einen zulässigen Bezeichner ein.
4367	Die Marke '<Name>' ist bereits vorhanden	Sie haben bereits eine Sprungmarke oder einen Schritt mit diesem Namen bezeichnet. Benennen Sie dementsprechend um.
4368	Aktion '<Name>' wird in mehreren übereinanderliegenden SFC-Ebenen verwendet!	Sie verwenden die Aktion '<Name>' sowohl im Baustein als auch in einer oder mehreren Aktionen dieses Bausteins.
4369	Genau ein Netzwerk für Transitionen nötig	Sie haben für die Transition mehrere FUP bzw. KOP-Netzwerke verwendet. Reduzieren Sie auf genau ein Netzwerk.
4370	Überflüssige Zeilen nach korrekter AWL-Transition gefunden	Löschen Sie die nicht benötigten Zeilen am Ende der Transition.
4371	Überflüssige Zeichen nach gültigem Ausdruck: '<Name>'	Löschen Sie die nicht benötigten Zeichen am Ende der Transition.
	Baustein '<Name>' unvollständig / mit Fehlern importiert bzw. konvertiert.	Der Baustein kann nicht vollständig nach IEC 61131-3 konvertiert werden.
4401	S5-Zeitkonstante '<Anzahl>' Sekunden zu groß (max. 9990s).	Im Akku steht keine gültige BCD-kodierte Zeit.

Nummer	Fehlermeldung	mögliche Ursache
4402	Direkter Zugriff nur auf E/As erlaubt.	Stellen Sie sicher, dass Sie nur auf eine als Ein- oder Ausgang definierte Variable zugreifen.
4403	Ungültiger oder nicht nach IEC 61131-3 konvertierbarer STEP5/7-Befehl.	Nicht jeder STEP5/7-Befehl ist nach IEC 61131-3 konvertierbar, z.B. CPU-Befehle wie MAS.
4404	Ungültiger oder nicht nach IEC 61131-3 konvertierbarer STEP5/7-Operand.	Nicht jeder STEP5/7-Operand ist nach IEC 61131-3 konvertierbar bzw. ein Operand fehlt.
4405	Reset eines STEP5/7-Timers kann nicht nach IEC 61131-3 konvertiert werden.	Die entsprechenden IEC-Timer haben keinen Reset-Eingang.
4406	STEP5/7-Zählerkonstante zu groß (max. 999)	Im Akku steht keine gültige BCD-kodierte Zählerkonstante.
4407	STEP5/7-Anweisung ist nicht nach IEC 61131-3 konvertierbar	Nicht jede STEP5/7-Anweisung ist nach IEC 61131-3 konvertierbar, z.B. DUF.
4408	Bitzugriff auf Timer-/Zähler-Worte nicht IEC 61131-3 konvertierbar.	Spezielle Timer-/Zählerbefehle sind nicht nach IEC 61131-3 konvertierbar.
4409	Inhalt von Akku1 oder Akku2 undefiniert, nicht nach IEC 61131-3 konvertierbar.	Ein Befehl, der die beiden Akkus verknüpft, kann nicht konvertiert werden, weil die Akku-Inhalte nicht bekannt sind.
4410	Aufgerufener Baustein nicht im Projekt.	Importieren Sie zuerst den aufgerufenen Baustein.
4411	Fehler in globaler Variablen-Liste.	Überprüfen Sie bitte die SEQ-Datei.
4412	Interner Fehler Nr.11	Wenden Sie sich bitte an Ihren Steuerungshersteller.
4413	Fehlerhaftes Format einer Zeile in Datenbaustein	Im zu importierenden Code ist ein fehlerhaftes Datum enthalten.
4414	FB/FX-Name fehlt	In der Ausgangs S5D-Datei fehlt der symbolische Name eines (erweiterten) Funktionsbausteins.
4415	Befehl nach Bausteinende nicht erlaubt	Ein geschützter Baustein kann nicht importiert werden.
4416	Ungültiger Befehl	Der S5/S7-Befehl kann nicht disassembliert werden.
4417	Kommentar nicht abgeschlossen	Schließen Sie den Kommentar mit "*)".
4418	FB/FX-Name zu lang (max. 8 Zeichen)	Der symbolische Name eines (erweiterten) Funktionsbausteins ist zu lang.
4419	Erwartetes Zeilenformat ""(* Name: <FB/FX-Name> *)""	Korrigieren Sie die Zeile entsprechend.
4420	FB/FX-Parametername fehlt	Überprüfen Sie die Funktionsbausteine.
4421	FB/FX-Parameterartname ungültig	Überprüfen Sie die Funktionsbausteine.
4422	FB/FX-Parameterart nicht angegeben	Überprüfen Sie die Funktionsbausteine.
4423	Ungültiger Aktualoperand	Überprüfen Sie die Schnittstelle des Funktionsbausteins.
4424	Warnung: Aufgerufener Baustein nicht vorhanden oder Kopf fehlerhaft oder hat keine Parameter	Der aufgerufene Funktionsbaustein wurde entweder noch nicht importiert oder ist fehlerhaft oder hat keine Parameter (im letzteren Fall können Sie die Meldung ignorieren).
4425	Sprungmarke nicht definiert	Das Ziel eines Sprungs ist nicht angegeben.
4426	Baustein hat keinen gültigen STEP5-Namen wie z.B. PB10	Ändern Sie den Bausteinnamen.
4427	Timertyp nicht angegeben	Fügen Sie eine Deklaration des Timers in die globale Variablenliste ein.
4428	Maximale STEP5/7-Klammertiefe überschritten	Es dürfen nicht mehr als sieben öffnende Klammern verwendet werden.

Nummer	Fehlermeldung	mögliche Ursache
4429	Fehler in Formal-Parameter-Name	Der Parametername darf nicht länger als vier Zeichen sein.
4430	Typ von Formal-Parameter nicht IEC-konvertierbar	Timer, Zähler und Bausteine können nicht als Formal-Parameter in IEC 61131-3 konvertiert werden.
4431	Zu viele VAR_OUTPUT-Parameter für einen Aufruf in STEP5/7-AWL	Ein Baustein darf nicht mehr als sechzehn Formal-Parameter als Ausgänge enthalten.
4432	Sprungmarken mitten in einem Ausdruck sind verboten	In IEC 61131-3 dürfen Sprungmarken nicht an beliebiger Stelle stehen.
4434	Zu viele Labels	Ein Baustein darf nicht mehr als 100 Labels enthalten.
4435	Nach Sprung / Aufruf kann nicht weiterverknüpft werden	Nach einem Sprung oder Aufruf muss ein Ladebefehl stehen.
4436	Inhalt von VKE undefiniert, nicht nach IEC 61131-3 konvertierbar.	Ein Befehl, der das VKE verwendet, kann nicht konvertiert werden, weil der Wert des VKE nicht bekannt ist.
4437	Typ von Befehl und Operand passen nicht zusammen	Ein Bit-Befehl wurde auf einen Word-Operanden angewendet oder umgekehrt.
4438	Kein Datenbaustein aufgeschlagen (fügen Sie ein A DB ein)	Fügen Sie ein A DB ein.
	Unbekannte Variable oder Adresse	Diese Watch-Variable ist im Projekt nicht deklariert. Durch Drücken von <F2> erhalten Sie die Eingabehilfe zu deklarierten Variablen.
4501	Einem gültigen Watchausdruck folgen unzulässige Zeichen	Entfernen Sie die überzähligen Zeichen.
4520	Fehler in Compilerdirektive: Flag erwartet vor '<Name>'!	Das Pragma ist nicht korrekt eingegeben. Überprüfen Sie, ob '<Name>' ein gültiges Flag ist.
4521	Fehler in Compilerdirektive: Unerwartetes Element '<Name>'!	Überprüfen Sie das Pragma auf korrekte Zusammensetzung.
4522	'flag off' Direktive erwartet!	Das Ausschalten des Pragmas fehlt, fügen Sie eine 'flag off' Anweisung ein.
4550	Index nicht im erlaubten Bereich : Variablen OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, dass der Index in dem in den Zielsystemeinstellungen/ Netzfunktionen festgelegten Bereich liegt.
4551	Subindex nicht in erlaubten Bereich : Variablen OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, dass der Subindex in dem in den Zielsystemeinstellungen/ Netzfunktionen festgelegten Bereich liegt.
4552	Index nicht in erlaubtem Bereich : Parameter OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, dass der Index in dem in den Zielsystemeinstellungen/ Netzfunktionen festgelegten Bereich liegt.
4553	Subindex nicht in erlaubtem Bereich : Parameter OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, dass der Subindex in dem in den Zielsystemeinstellungen/ Netzfunktionen festgelegten Bereich liegt.
4554	Variablenname ungültig: Variablen OD <Nummer>, Zeile <Zeilennummer>.	Geben Sie im Feld Variable eine gültige Projektvariable ein. Verwenden Sie die Schreibweise <Bausteinname> .<Variablenname> bzw. für globale Variablen .<Variablenname>
4555	Leeres Tabellenfeld, Eingabe nicht optional: Parameter OD %d, Zeile %d	Für dieses Feld muss eine Eingabe vorgenommen werden.
4556	Leeres Tabellenfeld, Eingabe nicht optional: Variablen OD %d, Zeile %d	Für dieses Feld muss eine Eingabe vorgenommen werden.

10.3 Kommandozeilen-/Kommandodatei-Befehle

Kommandozeilen-Befehle

Sie haben die Möglichkeit, TwinCAT PLC Control beim Start bestimmte Kommandos, die dann beim Ausführen geltend werden, mitzugeben. Diese Kommandozeilen-Befehle beginnen mit "/". Groß-/Kleinschreibung wird nicht berücksichtigt. Die Abarbeitung erfolgt sequentiell von links nach rechts.

Befehl	Beschreibung
/debug	
/online	
/run	
/show ...	Die Darstellung des TwinCAT PLC Control -Frame-Windows kann gesetzt werden.
/show hide	Das Fenster wird nicht angezeigt und erscheint auch nicht in der Task-Leiste.
/show icon	Das Fenster wird minimiert angezeigt.
/show max	Das Fenster wird maximiert angezeigt.
/show normal	Das Fenster wird in dem Zustand angezeigt, den es beim letzten Schließen hatte.
/out <outfile>	Alle Meldungen werden außer in das Meldungsfenster auch in die Datei <outfile> ausgegeben.
/cmd <cmdfile>	Nach dem Start werden die Befehle in der Kommandodatei <cmdfile> ausgeführt.

Die Eingabe einer Kommandozeile ist folgendermaßen aufgebaut:
"<Pfad der TwinCAT PLC Control-Exe-Datei>" "<Pfad des Projekts>" /<Befehl1> /<Befehl2> ....

Beispiel für eine Kommandozeile:
"D:\dir1 TwinCAT PLC Control" "C:\projects\ampel.pro" /show hide /cmd command.cmd

Datei ampel.pro wird geöffnet, das Fenster wird allerdings nicht angezeigt. Der Inhalt der Kommandodatei (cmdfile) command.cmd wird abgearbeitet.

Kommandodatei-Befehle

Im Folgenden finden Sie eine Auflistung der Befehle, die in einer Kommandodatei (<cmdfile>) verwendet werden können, das Sie dann wiederum über die Kommandozeile (siehe oben) aufrufen können. Groß/Kleinschreibung wird nicht berücksichtigt. Die Befehlszeile wird als Meldung im Meldungsfenster und ggf. in der Meldungsdatei (siehe unten) ausgegeben, außer dem Befehl wird ein "@" vorangestellt. Alle Zeichen nach einem Semikolon (;) werden ignoriert (Kommentar).

Befehle des Online Menüs:

Befehle	Beschreibung
online login	Einloggen mit dem geladenen Projekt ('Online Einloggen')
online logout	Ausloggen ('Online' 'Ausloggen')
online run	Starten des Anwenderprogramms ('Online' 'Start')
online sim	Einschalten der Simulation ((*) 'Online' 'Simulation')
online sim off	Ausschalten der Simulation. ('Online' 'Simulation')

Befehle des Datei Menüs:

Befehle	Beschreibung
file new	Es wird ein neues Projekt angelegt ('Datei' 'Neu').
file open <projectfile>	Es wird das angegebene Projekt geladen ('Datei' 'Öffnen').
file close	Das geladene Projekt wird geschlossen ('Datei' 'Schließen').
file save	Das geladene Projekt wird gespeichert ('Datei' 'Speichern').
file saveas <projectfile>	Das geladene Projekt wird unter dem angegeben Namen gespeichert ('Datei' 'Speichern unter').
file quit	TwinCAT PLC Control wird beendet ('Datei' 'Beenden').

Befehle des Projekt Menüs:

Befehle	Beschreibung
project compile	Das geladene Projekt wird mit "Alles übersetzen" übersetzt ('Projekt' 'Alles übersetzen').
project check	Das geladene Projekt wird überprüft ('Projekt' 'Alles überprüfen').
project build	Das geladene Projekt wird übersetzt ('Projekt' 'Übersetzen').
project import <file1> ... <fileN>	Die angegebenen Dateien <file1> ... <fileN> werden in das geladene Projekt importiert ('Projekt' 'Importieren').
project export <expfile>	Das geladene Projekt wird in die angegebene Datei <expfile> exportiert ('Projekt' 'Exportieren').
project expmul <expfile>	Jedes Objekt des geladenen Projekts wird in eine separate Datei exportiert, die jeweils den Namen des Objekts trägt.

Befehle zur Steuerung der Meldungsdatei

Befehle	Beschreibung
out open <msgfile>	Öffnet die angegebene Datei als Meldungsausgabe. Neue Meldungen werden angehängt.
out close	Schließt die derzeit geöffnete Meldungsdatei.
out clear	Löscht alle Meldungen aus der derzeit geöffneten Meldungsdatei.

Befehle zur Steuerung der Meldungs-Ausgaben

Befehle	Beschreibung
echo on	Die Befehlszeilen werden auch als Meldung ausgegeben.
echo off	Die Befehlszeilen werden nicht als Meldung ausgegeben.
echo <text>	Der <text> wird als Meldung ausgegeben.

Befehle zur Steuerung von Ersetzen von Objekten bzw. Dateien bei Import, Export, Ersetzen:

Befehle	Beschreibung
replace ok replace yes	Ersetzen
replace no	Nicht ersetzen.
replace noall	Nichts ersetzen
replace yesall	Alle ersetzen.

Befehle zur Steuerung des Default-Verhaltens von Dialogen:

Befehle	Beschreibung
query on	Dialoge werden angezeigt und erwarten Benutzereingaben.
query off ok	Alle Dialoge verhalten sich so, wie wenn der Benutzer OK angeklickt hat.
query off no	Alle Dialoge verhalten sich so, wie wenn der Benutzer Nein anklickt.
query off cancel	Alle Dialoge verhalten sich so, wie wenn der Benutzer Abbruch anklickt.

Befehle debugs

Befehl	Beschreibung
debug	entspricht /debug in der Kommandozeile.

Befehle zum Aufruf von Kommando-Dateien als Unterprogramme:

Befehl	Beschreibung
call <parameter1> ... <parameter10>	Kommandodateien werden als Unterprogramme aufgerufen. Es können bis zu 10 Parameter übergeben werden. In der aufgerufenen Datei kann mit $0 - $9 auf die Parameter zugegriffen werden.

Befehle zum Setzen der verwendeten Verzeichnisse:

Befehl	Beschreibung
dir lib <libdir>	Setzt <libdir> als Bibliotheksverzeichnis.
dir compile <compiledir>	Setzt <compiledir> als Verzeichnis für die Übersetzungsdateien.

Befehle zur Verzögerung der Abarbeitung des CMDFILEs:

Befehl	Beschreibung
delay 5000	Wartet 5 Sekunden.

Befehle zur Steuerung des Watch- und Rezepturverwalters:

Befehle	Beschreibung
watchlist load <file>	Lädt die unter <file> gespeicherte Watchliste und öffnet das zugehörige Fenster ('Extras' 'Watchliste laden').
watchlist save <file>	Speichert die aktuelle Watchliste unter <file> ('Extras' 'Watchliste speichern').
watchlist set <text>	Gibt einer zuvor geladenen Watchliste den Namen <text> ('Extras' 'Watchliste umbenennen').
watchlist read	Aktualisiert die Werte der Watchvariablen ('Extras' 'Rezeptur lesen').
watchlist write	Belegt die Watchvariablen mit den sich in der Watchliste befindenden Werten > ('Extras' 'Rezeptur schreiben').

Befehle zum Einbinden von Bibliotheken:

Befehle	Beschreibung
library add <Bibliotheksdatei1> <Bibliotheksdatei2> .. <BibliotheksdateiN>	Hängt die angegebenen Bibliotheksdateien an die Bibliotheksliste des aktuell geöffneten Projekts an. Handelt es sich beim Pfad der Datei um einen relativen Pfad, so wird das im Projekt eingestellte Bibliotheksverzeichnis als Wurzel des Pfads eingesetzt.
library delete [<Bibliothek1> <Bibliothek2> .. <BibliothekN>]	Löscht die angegebenen, bzw. wenn kein Bibliotheksname angegeben ist, alle Bibliotheken aus der Bibliotheksliste des aktuell geöffneten Projekts

Befehle zum Kopieren von Objekten:

Befehle	Berschreibung
object copy <Quellprojektdatei> <Quellpfad> <Zielpfad>	Kopiert Objekte aus dem angegebenen Pfad der Quellprojektdatei in den Zielpfad des gerade geöffneten Projekts. Ist der Quellpfad der Name eines Objektes, so wird dieses kopiert. Handelt es sich um einen Ordner, so werden alle Objekte unterhalb dieses Ordners kopiert. In diesem Fall wird die Ordnerstruktur unterhalb des Quellordners mit übernommen. Existiert der Zielpfad noch nicht, so wird er erstellt.

Befehle zum Systemaufruf:

Befehle	
system <Befehl>	Führt den angegeben Betriebssystembefehl aus.

Befehle zum Onlinebetrieb:

Befehle	
Reset	Führt einen Reset der Steuerung aus.
ResetAll	Führt einen Reset All der Steuerung aus. Hier werden auch die persistenten Daten gelöscht.
CreateBootproject	Ein Bootprojekt wird erzeugt.
ChooseRuntime <text>	Ein bestimmtes Laufzeitsystem kann ausgewählt werden. Als <text> muss die Net-Id und mit ':' getrennt die Portnummer stehen: Beispiel: ChooseRuntime 172.16.77.23.1.1:811 oder ChooseRuntime 5.2.122.255.1.1:801

Beispiel eines cmdfile:

Folgende Kommandodatei öffnet die Projektdatei ampel.pro, lädt eine unter w.wtc geladene Watchliste, startet das Anwenderprogramm, schreibt nach 1 Sekunde die Variablenwerte in die Watchliste watch.wtc, die gespeichert wird und schließt das Projekt danach wieder.

```
file open C:\work\projects\ampel.pro

query off ok

watchlist load c:\work\w.wtc

online login

online run

delay 1000

CreateBootproject

watchlist read
```

```
watchlist save c:\work\watch.wtc

online logout

file close
```

10.4 Datentypen

Der Benutzer kann Standarddatentypen und selbstdefinierten Datentypen beim Programmieren verwenden. Jedem Bezeichner wird ein Datentyp zugeordnet, der festlegt, wie viel Speicherplatz reserviert wird und welche Werte dem Speicherinhalt entsprechen.

Datenverlust

Die unterschiedlichen Datentypen decken unterschiedliche Zahlenbereich ab. Es kann passieren, dass bei der Typkonvertierung von größere auf kleinere Typen Information verloren geht.

Voraussetzungen

Standard Datentyp	Benutzerdefinierter Datentyp
BOOL [▶ 302]	ARRAY [▶ 306] (Felder, Arrays)
BYTE [▶ 303]	POINTER [▶ 307] (Zeiger)
WORD [▶ 303]	ENUM [▶ 308] (Aufzählungstyp)
DWORD [▶ 303]	STRUCT [▶ 308] (Strukturen)
SINT [▶ 304]	ALIAS [▶ 310] (Referenzen)
USINT [▶ 304]	Unterbereichstypen [▶ 310]
INT [▶ 304]	
UINT [▶ 304]	
DINT [▶ 304]	
UDINT [▶ 304]	
LINT (64 Bit Integer, wird aktuell von TwinCAT nicht unterstützt)	
ULINT (Unsigned 64 Bit Integer, wird aktuell von TwinCAT nicht unterstützt)	
REAL [▶ 305]	
LREAL [▶ 305]	
STRING [▶ 305]	
TIME [▶ 305]	
TIME_OF_DAY [▶ 305] (TOD)	
DATE [▶ 306]	
DATE_AND_TIME [▶ 306] (DT)	

10.4.1 Standard-Datentypen

10.4.1.1 BOOL

Variablen vom Typ BOOL können die Wahrheitswerte TRUE und FALSE annehmen.

Typ	Speicherplatz
BOOL	8 Bit

Eine Variable vom Typ BOOL hat den Wahrheitswert TRUE wenn das niederwertigste Bit im Speicher gesetzt ist (z.B. 2#00000001). Wenn kein Bit im Speicher gesetzt ist, besitzt die Variable den Wahrheitswert FALSE (2#00000000). Alle anderen Werte können im Online-Mode nicht richtig interpretiert und angezeigt werden (***INVALID: 16#xy *** in der Online-Anzeige). Solche Probleme können auftreten, wenn im SPS-Programm z.B. mit überlappenden Speicherbereichen gearbeitet wird.

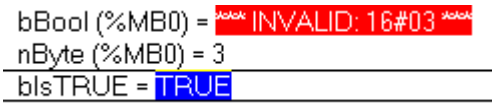
Beispiel:

Die boolsche Variable liegt im gleichen Speicherbereich wie die Byte-Variable.

```
PROGRAM MAIN
VAR
    bBool AT%MB0    : BOOL;
    nByte AT%MB0    : BYTE := 3;
    bIsTRUE         : BOOL;
END_VAR
```

```
IF bBool THEN
    bIsTRUE := TRUE;
ELSE
    bIsTRUE := FALSE;
END_IF
```

Online-Anzeige nach Programm-Start:



10.4.1.2 BYTE

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
BYTE	0	255	8 Bit

10.4.1.3 WORD

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
WORD	0	65535	16 Bit

10.4.1.4 DWORD

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
DWORD	0	4294967295	32 Bit

10.4.1.5 SINT

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
SINT	-128	127	8 Bit

10.4.1.6 USINT

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
USINT	0	255	8 Bit

10.4.1.7 INT

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
INT	-32768	32767	16 Bit

10.4.1.8 UINT

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
UINT	0	65535	16 Bit

10.4.1.9 DINT

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
DINT	-2147483648	2147483647	32 Bit

10.4.1.10 UDINT

Ganzzahliger Datentyp.

Typ	Untergrenze	Obergrenze	Speicherplatz
UDINT	0	4294967295	32 Bit

10.4.1.11 REAL

Gleitpunkttyp mit einer einfachen Genauigkeit, der benötigt wird bei Verwendung von rationalen Zahlen.

Typ	Untergrenze	Obergrenze	Speicherplatz
REAL	~ -3.402823 x 10^{38}	~ 3.402823 x 10^{38}	32 Bit

10.4.1.12 LREAL

Gleitpunkttyp mit einer doppelten Genauigkeit, der benötigt wird bei Verwendung von rationalen Zahlen.

Typ	Untergrenze	Obergrenze	Speicherplatz
LREAL	~ -1.79769313486231E308	~ 1.79769313486232E308	64 Bit

10.4.1.13 STRING

Eine Variable vom Typ STRING kann eine beliebige Zeichenkette aufnehmen. Die Größenangabe zur Speicherplatzreservierung bei der Deklaration bezieht sich auf Zeichen und kann in runden oder eckigen Klammern erfolgen. Ist keine Größe (1 bis 255) angegeben, so werden standardmäßig 80 Zeichen angenommen. Strings sind alle nullterminiert. D.h. das letzte Zeichen eines Strings ist immer Null.

Beispiel einer Stringdeklaration mit 35 Zeichen:

```
str:STRING(35):='Dies ist ein String';
```

Typ	Speicherplatz
STRING	1 Byte pro Zeichen (Größe bei der Deklaration) + 1 Byte für die Nullterminierung

10.4.1.14 TIME

Zeitdauer. Die kleinste Zeiteinheit ist eine Millisekunde. Der Datentyp wird intern wie DWORD behandelt.

Typ	Untergrenze	Obergrenze	Speicherplatz
TIME	T#0ms	T#71582m47s295ms	32 Bit

10.4.1.15 TOD

Tageszeit. Die kleinste Zeiteinheit ist eine Millisekunde. Der Datentyp wird intern wie DWORD behandelt.

Typ	Untergrenze	Obergrenze	Speicherplatz
TIME_OF_DAY TOD	TOD#00:00	TOD#1193:02:47.295	32 Bit

10.4.1.16 DATE

Datum. Die kleinste Zeiteinheit ist eine Sekunde. Der Datentyp wird intern wie DWORD behandelt.

Typ	Untergrenze	Obergrenze	Speicherplatz
DATE	D#1970-01-01	D#2106-02-06	32 Bit

10.4.1.17 DT

Datum und Tageszeit. Die kleinste Zeiteinheit ist eine Sekunde. Der Datentyp wird intern wie DWORD behandelt.

Typ	Untergrenze	Obergrenze	Speicherplatz
DATE_AND_TIME DT	DT#1970-01-01-00:00	DT#2106-02-06-06:28:15	32 Bit

10.4.2 Anwender-Datentypen

10.4.2.1 Arrays

Es werden ein-, zwei-, und dreidimensionale Felder (Arrays) von elementaren Datentypen unterstützt. Arrays können im Deklarationsteil eines Bausteins und in den globalen Variablenlisten definiert werden.

Syntax:

```
<Feld_Name>:ARRAY
[<ug1>..<og1>,<ug2>..<og2>] OF <elem.
Typ>
```

ug1, *ug2* geben die untere Grenze des Feldbereichs an, *og1*, *og2* die obere Grenze. Die Grenzwerte müssen ganzzahlig sein.

Beispiel:

```
Kartenspiel : ARRAY [1..13, 1..4] OF INT;
```

Initialisierung von Arrays

Entweder werden alle Elemente eines Array initialisiert oder keines.

Beispiele für Initialisierungen von Arrays:

```
arr1 : ARRAY [1..5] OF INT := 1,2,3,4,5;

arr2 : ARRAY [1..2,3..4] OF INT := 1,3(7); (* kurz für 1,7,7,7
*)

arr3 : ARRAY [1..2,2..3,3..4] OF INT := 2(0),4(4),2,3; (* kurz für
0,0,4,4,4,4,2,3 *)
```

Beispiel für die Initialisierung eines Arrays einer Struktur:

```
TYPE STRUCT1

STRUCT

    p1:int;

    p2:int;

    p3:dword;

END_STRUCT
```

```
arr1 : ARRAY[1..3] OF STRUCT1:=[(p1:=1,p2:=10,p3:=4723),(p1:=2,p2=0,p3:=299),
(p1:=14,p2:=5,p3:=112)];
```

Beispiel für eine teilweise Initialisierung eines Arrays:

```
arr1 : ARRAY [1..10] OF INT := 1,2;
```

Elemente, für die kein Wert vorgegeben wird, werden mit dem Default-Initialwert des Basistypen initialisiert. Im obigen Beispiel werden also die Elemente *arr1*[3] bis *arr1*[10] mit 0 initialisiert.

Auf Komponenten von Arrays greift man bei einem zweidimensionalen Feld mit folgender Syntax zu:

```
<Feld_Name>[Index1,Index2]
```

Beispiel:

```
Kartenspiel[9,2]
```

Wenn Sie in Ihrem Projekt eine Funktion mit Namen CheckBounds [▶ 341] definieren, können Sie damit Bereichsüberschreitungen bei Arrays automatisch überprüfen! Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

10.4.2.2 Pointer

In Pointern speichert man die Adresse von Variablen oder Funktionsblöcken zur Laufzeit eines Programms. Pointerdeklarationen haben folgende Syntax:

```
<Bezeichner>: POINTER TO
<Datentyp/Funktionsblock>;
```

Ein Pointer kann auf jeden beliebigen Datentyp und Funktionsblock zeigen, auch selbstdefinierte.

Mit dem Adressoperator ADR [▶ 331] wird dem Pointer die Adresse einer Variablen oder Funktionsblocks zugewiesen.

Die Dereferenzierung eines Pointers erfolgt über den Inhaltsoperator "^" nach dem Pointerbezeichner. Mit der Hilfe des Operators SIZEOF [▶ 331] wird ein Pointer z.B. inkrementiert.

Ein Pointer wird byte-weise hochgezählt! Über die Anweisung p=p+SIZEOF(p^); kann ein Hochzählen wie im C-Compiler erreicht werden.

HINWEIS

Adressenverschiebung

Wenn Online Change angewendet wird, können sich Inhalte von Adressen verschieben. Beachten Sie dies bei der Verwendung von Pointern auf Adressen.

Beispiel:

```
pt:POINTER TO INT;

var_int1:INT := 5;

var_int2:INT;
```

```
pt := ADR(var_int1);

var_int2:= pt^; (* var_int2 ist nun 5 *)
```

Beispiel 2 (Pointer Inkrementierung):

```
ptByCurrDataOffs : POINTER TO BYTE;

udiAddress       : UDINT;
```

```
(*--- pointer increment ---*)

udiAddress := ptByCurrDataOffs;
```

```
udiAddress := udiAddress + SIZEOF(ptByCurrDataOffs^);

ptByCurrDataOffs := udiAddress;

(* -- end of pointer increment ---*)
```

10.4.2.3 Aufzählungstyp (ENUM)

Ein Aufzählungstyp ist ein selbstdefinierter Datentyp, der aus einer Menge von Stringkonstanten besteht. Diese Konstanten bezeichnet man als Enumerationswerte. Die Enumerationswerte sind immer im ganzen Projekt bekannt. Legen Sie ihre Aufzählungstypen am besten als Objekte im Object Organizer unter der Registerkarte Datentypen an. Sie beginnen mit dem Schlüsselwort TYPE und enden mit END_TYPE.

Syntax:

```
TYPE <Bezeichner>:(<Enum_0>
,<Enum_1>, ...,<Enum_n>);

END_TYPE
```

Eine Variable vom Typ <Bezeichner> kann einen der Enumerationswerte annehmen und wird mit dem ersten initialisiert. Die Werte sind zu ganzen Zahlen kompatibel, d.h. man kann damit Operationen wie mit INT durchführen. Der Variablen kann eine Zahl x zugewiesen werden. Sind die Enumerationswerte nicht initialisiert, beginnt die Zählung bei 0. Achten Sie beim Initialisieren darauf, dass die Initialwerte aufsteigend sind. Die Gültigkeit der Zahl wird zur Laufzeit überprüft.

Beispiel:

```
TYPE AMPEL: (Rot, Gelb, Gruen:=10); (*Rot hat
den Initialwert 0, Gelb 1, Gruen 10 *)

END_TYPE


AMPEL1 : AMPEL;


AMPEL1:=0; (* Ampel1 hat den Wert Rot*)

FOR i:= Rot TO Gruen DO

    i := i + 1;

END_FOR;
```

Der gleiche Enumerationswert darf nicht zweimal verwendet werden.

Beispiel:

```
AMPEL: (rot, gelb, gruen);

FARBE: (blau, weiss, rot);
```

Fehler: rot darf nicht für AMPEL und FARBE verwendet werden.

10.4.2.4 Strukturen

Strukturen werden als Objekte im Object Organizer unter der Registerkarte Datentypen abgelegt. Sie beginnen mit dem Schlüsselwort TYPE und STRUCT und enden mit END_STRUCT und END_TYPE. Strukturdeklarationen haben folgende Syntax:

```
TYPE <Strukturname>:

    STRUCT <Variablendeklaration 1> . .
<Variablendeklaration n>
```

```
    END_STRUCT
END_TYPE
```

<Strukturname> ist nun ein Typ, der im gesamten Projekt bekannt ist, und der wie ein Standard Datentyp benutzt werden kann. Verschachtelte Strukturen sind erlaubt. Die einzige Einschränkung ist, dass Variablen nicht auf Adressen gesetzt werden können (AT-Deklaration ist nicht erlaubt!).

Beispiel für eine Strukturdefinition mit Namen Polygonzug:

```
TYPE Polygonzug:
    STRUCT
    Start:ARRAY [1..2] OF INT;
    Punkt1:ARRAY [1..2] OF INT;
    Punkt2:ARRAY [1..2] OF INT;
    Punkt3:ARRAY [1..2] OF INT;
    Punkt4:ARRAY [1..2] OF INT;
    Ende:ARRAY [1..2] OF INT;
    END_STRUCT
END_TYPE
```

Beispiel für die Initialisierung einer Struktur:
Poly_1:polygonzug := (Start:=3,3, Punkt1 =5,2, Punkt2:=7,3, Punkt3:=8,5, Punkt4:=5,7, Ende := 3,5);

Auf Komponenten von Strukturen greift man mit folgender Syntax zu:

```
<Struktur_Name>.<Komponentenname>
```

Wenn wir zum Beispiel eine Struktur mit Namen "Woche" haben, die eine Komponente mit Namen "Montag" beinhaltet, dann können wir darauf folgendermaßen zugreifen: Woche.Montag

HINWEIS

Unterschiedliche Strukturen

Strukturen und Arrays können auf verschiedenen Hardwareplattformen (beispielsweise CX1000 und CX90xx) aufgrund unterschiedlicher Alignments in Aufbau und Größe unterschiedlich sein.
Beim Datenaustausch ist auf die identische Größe und gleich ausgerichtete Strukturen zu achten!

Beispiel für eine Strukturdefinition mit Namen ST_ALIGN_SAMPLE:

```
TYPE ST_ALIGN_SAMPLE:
    STRUCT
     _diField1   : DINT;
     _byField1   : BYTE;
     _iField     : INT;
     _byField2   : BYTE;
     _diField2   : DINT;
     _pField     : POINTER TO BYTE;
   END_STRUCT
END_TYPE
```

Somit ergeben sich für die Memberkomponenten der Struktur ST_ALIGN_SAMPLE auf CX90xx(RISC) folgende Größen und Offsets:

_diField1 (DINT), Offset = 0 (16#0), Size = 4
_byField1 (BYTE), Offset = 4 (16#4), Size = 1
_iField (INT), Offset = 6 (16#6), Size = 2

_byField2 (BYTE), Offset = 8 (16#8), Size = 1
_diField2 (DINT), Offset = 12 (16#C), Size = 4
_pField (POINTER TO BYTE), Offset = 16 (16#10), Size = 4

Gesamtgröße durch natürliches Alignment mit Pack(4) und mit sogenannten Paddingbytes: 20

Somit ergeben sich für die Memberkomponenten der Struktur ST_ALIGN_SAMPLE auf CX10xx folgende Größen und Offsets:

_diField1 (DINT), Offset = 0 (16#0), Size = 4
_byField1 (BYTE), Offset = 4 (16#4), Size = 1
_iField (INT), Offset = 5 (16#5), Size = 2
_byField2 (BYTE), Offset = 7 (16#7), Size = 1
_diField2 (DINT), Offset = 8 (16#8), Size = 4
_pField (POINTER TO BYTE), Offset = 12 (16#C), Size = 4

Gesamtgröße: 16

Ansicht der Struktur ST_ALIGN_SAMPLE bei CX90xx Plattformen (RISC) **mit Darstellung der Paddingbytes**:

```
TYPE ST_ALIGN_SAMPLE:

    STRUCT

     _diField1    : DINT;

     _byField1    : BYTE;
     _byPadding   : BYTE;
     _iField      :
INT;

     _byField2    : BYTE;
     _a_byPadding : ARRAY[0..2] OF BYTE;

_diField2    : DINT;

     _pField      : POINTER TO BYTE;

   END_STRUCT

END_TYPE
```

10.4.2.5 Referenzen (Alias Typen)

Der selbstdefinierte Datentyp Referenz dient dazu, um einen alternativen Namen für eine Variable, Konstante oder einen Funktionsblock zu erzeugen. Legen Sie ihre Referenzen als Objekte im Object Organizer unter der Registerkarte Datentypen an. Sie beginnen mit dem Schlüsselwort TYPE und enden mit END_TYPE.

Syntax:

```
TYPE <Bezeichner>:
<Zuweisungsausdruck>;

END_TYPE
```

Beispiel:

```
TYPE message:STRING[50];

END_TYPE;
```

10.4.2.6 Unterbereichstypen

Ein Unterbereichstyp ist ein Typ, dessen Wertebereich nur eine Untermenge eines Basistypen umfasst. Die Deklaration kann im Register Datentypen erfolgen, eine Variablen kann aber auch direkt mit einem Unterbereichstypen deklariert werden:
Syntax für die Deklaration im Register '**Datentypen**':

```
TYPE <Name> : <Inttype>
(<ug>..<og>) END_TYPE;
```

Typ	Beschreibung
<Name>	muss ein gültiger IEC Bezeichner sein.
<Inttype>	ist einer der Datentypen SINT, USINT, INT, UINT, DINT, UDINT, BYTE, WORD, DWORD.
<ug>	ist eine Konstante, die kompatibel sein muss zum Basistypen, und die die Untergrenze des Bereichstypen festlegt. Die Untergrenze selbst gehört zu diesem Bereich
<og>	ist eine Konstante, die kompatibel sein muss zum Basistypen, und die die Obergrenze des Bereichstypen festlegt. Die Obergrenze selbst gehört zu diesem Basistypen.

Beispiel:

```
TYPE

SubInt : INT (-4095..4095);

END_TYPE
```

Direkte Deklaration einer Variablen mit einem Unterbereichstypen (Beachten Sie die korrekte Angabe eines Initialwerts, wenn der Unterbereich nicht die '0' enthält):

```
VAR

    i1 : INT (-4095..4095);

    i2: INT (5..10):=5;

    ui : UINT (0..10000);

END_VAR
```

Wird einem Unterbereichstypen eine Konstante zugewiesen (in der Deklaration oder in der Implementation), die nicht in diesen Bereich fällt (z.B. i:=5000), wird eine Fehlermeldung ausgegeben.

Um die Einhaltung der Bereichsgrenzen zur Laufzeit zu überprüfen, müssen die Funktionen CheckRangeSigned [▶ 344] bzw. CheckRangeUnsigned [▶ 346] eingefügt werden.

10.5 Operatoren

TwinCAT PLC Control unterstützt alle IEC-Operatoren. Diese sind, im Gegensatz zu den Standardfunktionen implizit im ganzen Projekt bekannt. In den Bausteinimplementationen werden Operatoren wie Funktionen benutzt.

10.5.1 Übersicht IEC Operatoren

Die folgende Tabelle zeigt die Operatoren in ST und AWL mit den verfügbaren Modifikatoren in AWL.

Die Spalte 'wo?' gibt an, in welcher Bibliothek der Operator enthalten ist bzw. ob er als IEC-Operator im Programmiersystem integriert ist.

Beachten Sie für die Spalte 'Operator AWL': Es wird nur die Zeile dargestellt, in der der Operator verwendet wird. Vorausgesetzt wird ein in der vorangehenden Zeile erfolgtes Laden des (ersten) benötigten Operanden (z.b. LD in).

Die Spalte 'Mod.AWL' zeigt die möglichen Modifikatoren in AWL:

C	Die Anweisung wird nur ausgeführt, wenn das Ergebnis des vorhergehenden Ausdrucks TRUE ist
N	bei JMPC, CALC, RETC: Die Anweisung wird nur ausgeführt, wenn das Ergebnis des vorhergehenden Ausdrucks FALSE ist
N	sonst: Negation des Operanden (nicht des Akku)
(	Klammerung begrenzt Operator, erst nach Erreichen der abschließenden Klammer wird die der Klammer vorangestellte Operation ausgeführt

Die detaillierte Beschreibung zur Anwendung entnehmen Sie bitte den entsprechenden Anhängen zu den integrierten IEC-Operatoren bzw. den Bibliotheken.

Operator ST	Operator AWL	Mod. AWL	Bedeutung
'			Stringbegrenzung (z.B. 'string')
[..]			Array: Darstellung des Array-Bereiches (z.B. Array[0..3] OF INT)
:			Trennzeichen zwischen Operand und Typ bei der Deklaration (z.B. var1 : INT;)
;			Abschluss Anweisung (z.B. a:=var1;)
^			Dereferenziere Pointer (z.B. pointer1^)
	LD var1	N	Lade Wert von var1 in Akku
:=	ST var1	N	Speichere aktuelles Ergebnis an die Operandenstelle var1
	S boolvar		Setze den Bool-Operanden boolvar genau dann auf TRUE, wenn das aktuelle Ergebnis TRUE ist
	R boolvar		Setze den Bool-Operand genau dann auf FALSE, wenn das aktuelle Ergebnis TRUE ist
	JMP marke	CN	Springe zur Marke
<Programmname>	CAL prog1	CN	Rufe Programm prog1 auf
<Instanzname>	CAL inst1	CN	Rufe FB Instanz inst1 auf
<Fktname>(vx,vy,..)	<Fktname> vx,vy,..	CN	Rufe Funktion auf und übergebe Parameter vx, vy
RETURN	RET	CN	Verlasse den Baustein und kehre ggf. zurück zum Aufrufer
	(		Wert, der der Klammer folgt, wird als Operand gesehen, vorherige Operation wird zurückgestellt, bis Klammer geschlossen wird
	)		Werte zurückgestellte Operation aus
AND	AND	N, (	Bitweises AND
OR	OR	N, (	Bitweises OR
XOR	XOR	N, (	Bitweises exklusives OR
NOT	NOT		Bitweises NOT
+	ADD	(	Addition
-	SUB	(	Subtraktion
*	MUL	(	Multiplikation
/	DIV	(	Division
>	GT	(	größer
>=	GE	(	größer/gleich

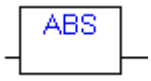
Operator ST	Operator AWL	Mod. AWL	Bedeutung
=	EQ	(	gleich
<	LT	(	kleiner
<>	NE	(	nicht gleich
<=	LE	(	kleiner/gleich
MOD(in)	MOD		Modulo Division
INDEXOF(in)	INDEXOF		interner Index eines Bausteins in1; [INT]
SIZEOF(in)	SIZEOF		benötigte Byte-Anzahl für geg. Datentyp von in
SHL(in,K)	SHL		bitweises Links-Shift eines Operanden um K
SHR(in,K)	SHR		bitweises Rechts-Shift eines Operanden um K
ROL(in,K)	ROL		bitweise Links-Rotation eines Operanden um K
ROR(in,K)	ROR		bitweise Rechts-Rotation eines Operanden um K
SEL(G,in0,in1)	SEL		binäre Selektion zw. 2 Operanden in0 (G ist FALSE) und in1 (G ist TRUE)
MAX(in0,in1)	MAX		liefert von zwei Werten den größten
MIN(in0,in1)	MIN		liefert von zwei Werten den kleinsten
LIMIT(Min,in,Max)	LIMIT		Limitierung des Wertebereichs (in wird bei Überschreitung auf Min bzw. Max zurückgesetzt)
MUX(K, in0,.. in_n)	MUX		Auswahl des K-ten Wertes aus einer Anzahl von Werten (in0 bis In_n)
ADR(in)	ADR		Adresse des Operanden in [DWORD]
BOOL_TO_<type>(in)	BOOL_TO_<type>		Typkonvertierung des boolschen Operanden in anderen elementaren Typ
<type>_TO_BOOL(in)	<type>_TO_BOOL		Typkonvertierung des Operanden nach BOOL
INT_TO_<type>(in)	INT_TO_<type>		Typkonvertierung des INT Operanden in anderen elementaren Typ
REAL_TO_<type>(in)	REAL_TO_<type>		Typkonvertierung des REAL Operanden in anderen elementaren Typ
LREAL_TO_<type>(in)	LREAL_TO_<type>		Typkonvertierung des LREAL Operanden in anderen elementaren Typ
TIME_TO_<type>(in)	TIME_TO_<type>		Typkonvertierung des TIME Operanden in anderen elementaren Typ
TOD_TO_<type>(in)	TOD_TO_<type>		Typkonvertierung des TOD Operanden in anderen elementaren Typ

Operator ST	Operator AWL	Mod. AWL	Bedeutung
DATE_TO_<type>(in)	DATE_TO_<type>		Typkonvertierung des DATE Operanden in anderen elementaren Typ
DT_TO_<type>(in)	DT_TO_<type>		Typkonvertierung des DT Operanden in anderen elementaren Typ
STRING_TO_<type>(in)	STRING_TO_<type>		Typkonvertierung des STRING Operanden in anderen elementaren Typ
TRUNC(in)	TRUNC		Konvertierung von REAL nach INT
ABS(in)	ABS		Absolutwert des Operanden
SQRT(in)	SQRT		Quadratwurzel des Operanden
LN(in)	LN		natürlicher Logarithmus des Operanden
LOG(in)	LOG		Logarithmus des Operanden zur Basis 10
EXP(in)	EXP		Exponentialfunktion des Operanden
SIN(in)	SIN		Sinus des Operanden
COS(IN)	COS		Cosinus des Operanden
TAN(in)	TAN		Tangens des Operanden
ASIN(in)	ASIN		Arcussinus des Operanden
ACOS(in)	ACOS		Arcuscosinus des Operanden
ATAN(in)	ATAN		Arcustangens des Operanden
EXPT(in,expt)	EXPT expt		Potenzierung des Operanden um expt
LEN(in)	LEN		Stringlänge des Operanden
LEFT(str, size)	LEFT		linker Anfangsstring (Größe size) von String str
RIGHT(str, size)	RIGHT		rechter Anfangsstring (Größe size) von String str
MID(str, len, pos)	MID		Teilstring der Größe len aus String str
CONCAT(str1, str2)	CONCAT		Aneinanderhängen zweier Strings
INSERT(str1, str2, pos)	INSERT		Einfügen String str1 in String str2 an Position pos
DELETE(str1, len, pos)	DELETE		Lösche Teilstring mit Länge len, beginnend an Position pos von str1
REPLACE(str1, str2, len, pos)	REPLACE		Ersetze Teilstring von Länge len, beginnend an Position pos von str1 durch str2
FIND(str1, str2)	FIND		Suchen eines Teilstrings str2 in str1
SR	SR		Bistabiler FB wird dominant gesetzt

Operator ST	Operator AWL	Mod. AWL	Bedeutung
RS	RS		Bistabiler FB wird zurückgesetzt
SEMA	SEMA		FB: Software Semaphor
R_TRIG	R_TRIG		FB: ansteigende Flanke wird erkannt
F_TRIG	F_TRIG		FB: fallende Flanke wird erkannt
CTU	CTU		FB: Aufwärtszähler
CTD	CTD		FB: Abwärtszähler
CTUD	CTUD		FB: Auf- und Abwärtszähler
TP	TP		FB: Pulsgeber
TON	TON		FB: Einschaltverzögerung
TOF	TOF		FB: Ausschaltverzögerung

10.5.2 Numerische Operatoren

10.5.2.1 ABS



Liefert den Absolutwert einer Zahl. ABS(-2) ergibt 2.

Folgende Typkombinationen für IN und OUT sind möglich:

IN	OUT
INT	INT, REAL, WORD, DWORD, DINT
REAL	REAL
BYTE	INT, REAL, BYTE, WORD, DWORD, DINT
WORD	INT, REAL, WORD, DWORD, DINT
DWORD	REAL, DWORD, DINT
SINT	REAL
USINT	REAL
UINT	INT, REAL, WORD, DWORD, DINT, UDINT, UINT
DINT	REAL, DWORD, DINT
UDINT	REAL, DWORD, DINT, UDINT

10.5.2.2 ACOS

Liefert den Arcuscosinus (Umkehrfunktion von Cosinus) einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.3 ASIN

Liefert den Arcussinus (Umkehrfunktion von Sinus) einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.4 ATAN

Liefert den Arcustangens (Umkehrfunktion von Tangens) einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

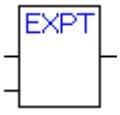
10.5.2.5 COS

Liefert den Cosinus einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.6 EXP

Liefert die Exponentialfunktion. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.7 EXPT



Potenzierung einer Variablen mit einer anderen: OUT = $IN1^{IN2}$;

IN1 und IN2 können vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

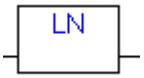
Beispiel in AWL:

```
LD 7

EXPT 2

ST var1 (* Ergebnis ist 49 *)
```

Beispiel in ST:

```
var1 := EXPT(7,2);
```

10.5.2.8 LN

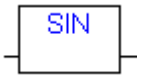


Liefert den natürlichen Logarithmus einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.9 LOG

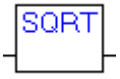
Liefert den Logarithmus zur Basis 10 einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.10 SIN



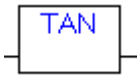
Liefert den Sinus einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.11 SQRT



Liefert die Quadratwurzel einer Zahl. IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.2.12 TAN



Liefert den Tangens einer Zahl. Der Wert wird in Bogenmaß errechnet.IN kann vom Typ BYTE, WORD, DWORD, INT, DINT, REAL, SINT, USINT, UINT, UDINT sein, OUT muss vom Typ REAL sein.

10.5.3 Arithmetische Operatoren

10.5.3.1 ADD

Addition von Variablen vom Typ BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL und LREAL.Es können auch zwei TIME-Variablen addiert werden, die Summe ist dann wieder eine Zeit (z.B. gilt t#45s + t#50s = t#1m35s)

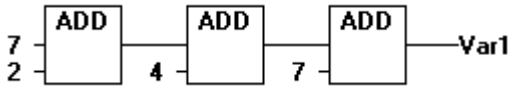
Beispiel in AWL:

```
LD 7

ADD 2,4,7

ST var1
```

Beispiel in ST:

```
var1 := 7+2+4+7;
```

Beispiel in FUP:



10.5.3.2 MUL

Multiplikation von Variablen vom Typ BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL und LREAL.

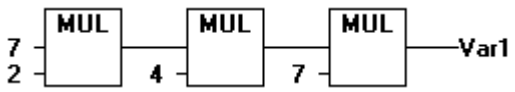
Beispiel in AWL:

```
LD 7

MUL 2,4,7

ST var1
```

Beispiel in ST:

```
var1 := 7*2*4*7;
```

Beispiel in FUP:



10.5.3.3 SUB

Subtraktion einer Variablen vom Typ BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL und LREAL von einer anderen Variablen von einem dieser Typen. Eine TIME-Variable kann auch von einer anderen TIME-Variablen subtrahiert werden, das Ergebnis ist dann wieder vom Typ TIME. Beachten Sie, dass negative TIME-Werte nicht definiert sind.

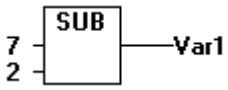
Beispiel in AWL:

```
LD 7

SUB 8

ST var1
```

Beispiel in ST:

```
var1 := 7-2;
```

Beispiel in FUP:



10.5.3.4 DIV

Division einer Variablen vom Typ BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL und LREAL durch eine andere Variable von einem dieser Typen.

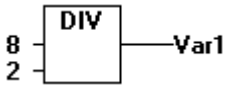
Beispiel in AWL:

```
LD 8

DIV 2

ST var1
```

Beispiel in ST:

```
var1 := 8/2;
```

Beispiel in FUP:



Wenn Sie in Ihrem Projekt Funktionen mit Namen CheckDivByte [▶ 342], CheckDivWord [▶ 343], CheckDivDWord [▶ 344] und CheckDivReal [▶ 343] definieren, können Sie damit bei Verwendung des Operators DIV den Wert des Divisors überprüfen, beispielsweise um eine Division durch 0 zu verhindern. Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

10.5.3.5 MOD

Modulo Division einer Variablen vom Typ BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT und UDINT durch eine andere Variable von einem dieser Typen. Als Ergebnis liefert diese Funktion den ganzzahligen Rest der Division.

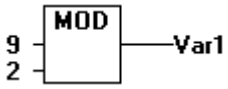
Beispiel in AWL:

```
LD 9
MOD 2
ST var1 (* Ergebnis ist 1 *)
```

Beispiel in ST:

```
var1 := 9 MOD 2;
```

Beispiel in FUP:



Ähnliche Funktionen

LMOD

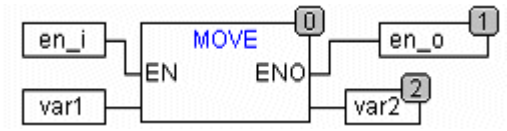
10.5.3.6 MOVE

Zuweisung einer Variablen auf eine andere Variable eines entsprechenden Typs. Dadurch, dass MOVE in den CFC- und KOP-Editoren als Baustein verfügbar ist, kann dort die EN/EN0-Funktionalität auch auf eine Variablenzuweisung angewendet werden. Im FUP-Editor ist dies leider nicht möglich.

Beispiel in CFC in Verbindung mit der EN/EN0 Funktion:

Nur wenn en_i TRUE ist, wird der Wert der Variablen var1 Variable var2 zugewiesen.

Bitstring Operatoren...



Beispiel in AWL:

```
LD ivar1
MOVE
ST ivar2 (* Ergebnis: var2 erhält Wert von var1 *)
```

(! entspricht:

```
LD ivar1
ST ivar2
```

)

Beispiel in ST:

```
ivar2 := MOVE(ivar1);
```

(! entspricht:

```
ivar2 := ivar1;
```

)

10.5.4 Bitstring Operatoren

10.5.4.1 AND

Bitweise AND von Bit-Operanden. Die Operanden sollten vom Typ BOOL, BYTE, WORD oder DWORD sein.

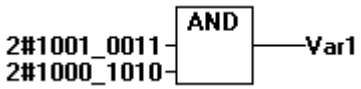
Beispiel in AWL:

```
var1 :BYTE;

LD 2#1001_0011

AND 2#1000_1010

ST var1 (* Ergebnis ist 2#1000_0010 *)
```

Beispiel in ST:

```
var1 := 2#1001_0011 AND 2#1000_1010
```

Beispiel in FUP:



10.5.4.2 OR

Bitweise OR von Bit-Operanden. Die Operanden sollten vom Typ BOOL, BYTE, WORD oder DWORD sein.

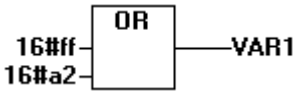
Beispiel in AWL:

```
var1 :BYTE;

LD 2#1001_0011

OR 2#1000_1010

ST var1 (* Ergebnis ist 2#1001_1011 *)
```

Beispiel in ST:

```
Var1 := 2#1001_0011 OR 2#1000_1010
```

Beispiel in FUP:



10.5.4.3 XOR

Bitweise XOR von Bit-Operanden. Die Operanden sollten vom Typ BOOL, BYTE, WORD oder DWORD sein.

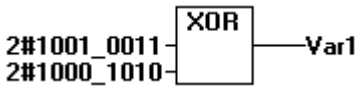
Beispiel in AWL:

```
Var1 :BYTE;

LD 2#1001_0011

XOR 2#1000_1010

ST Var1 (* Ergebnis ist 2#0001_1001 *)
```

Beispiel in ST:

```
Var1 := 2#1001_0011 XOR 2#1000_1010
```

Beispiel in FUP:



Beachten Sie, dass das Verhalten des XOR-Bausteins in erweiterter Form (mehr als 2 Eingänge) nicht normkonform implementiert ist. Die Eingänge werden paarweise geprüft und die jeweiligen Ergebnisse dann wiederum gegeneinander verglichen

10.5.4.4 NOT

Bitweise NOT eines Bit Operanden. Der Operand sollte vom Typ BOOL, BYTE, WORD oder DWORD sein

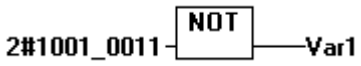
Beispiel in AWL:

```
Var1 :BYTE;

LD 2#1001_0011

NOT

ST Var1 (* Ergebnis ist 2#0110_1100 *)
```

Beispiel in ST:

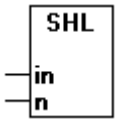
```
Var1 := NOT 2#1001_0011;
```

Beispiel in FUP:



10.5.5 Bitshift Operatoren

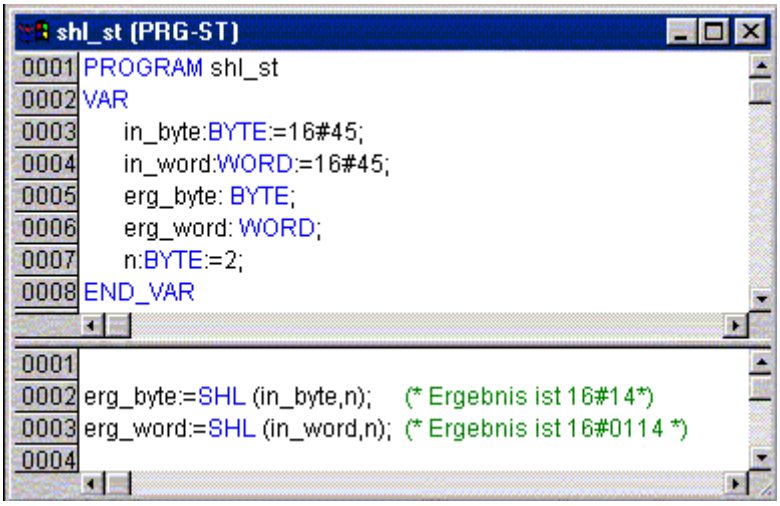
10.5.5.1 SHL



Bitweises Links-Shift eines Operanden: A:= SHL (IN, N)A, IN und N sollten vom Typ BYTE, WORD, DWORD sein. IN wird um N Bits nach links geschoben, und von rechts mit Nullen aufgefüllt.

Beachten Sie, dass die Anzahl der Bits, die für die Rechenoperation berücksichtigt wird, durch den Datentyp der Eingangsvariable in vorgegeben wird. Handelt es sich hierbei um eine Konstante, wird der kleinstmögliche Datentyp berücksichtigt. Der Datentyp der Ausgangsvariable bleibt ohne Auswirkung auf die Rechenoperation.

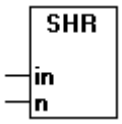
Sehen Sie im nachfolgenden Beispiel in hexadezimaler Darstellung, wie sich bei gleichem Wert der Eingangsvariablen in_byte und in_word die Ergebnisse erg_byte und erg_word der Operation unterscheiden, je nachdem, ob in vom Typ BYTE oder WORD ist.



Beispiel:

```
LD 1

SHL 1

ST Var1 (* Ergebnis ist 2 *)
```

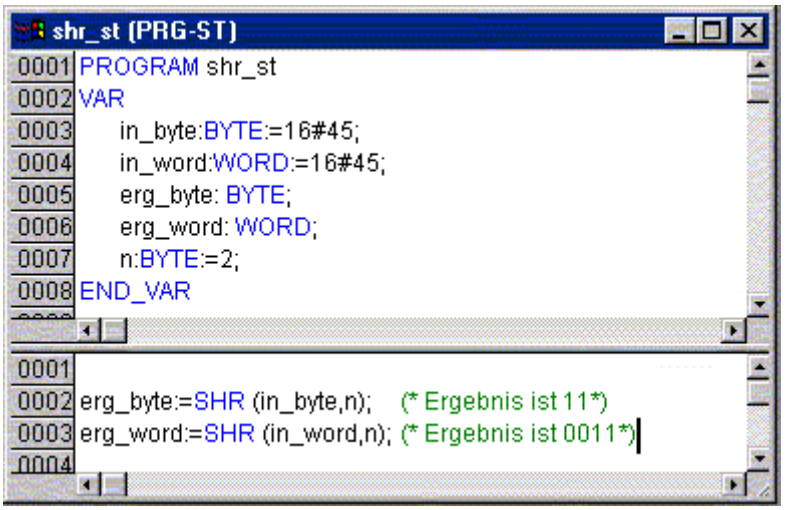
10.5.5.2 SHR



Bitweises Rechts-Shift eines Operanden: A:= SHR (IN, N)A, IN und N sollten vom Typ BYTE, WORD oder DWORD sein. IN wird um N Bits nach rechts geschoben, und von links mit Nullen aufgefüllt.

Beachten Sie, dass die Anzahl der Bits, die für die Rechenoperation berücksichtigt wird, durch den Datentyp der Eingangsvariable in vorgegeben wird. Handelt es sich hierbei um eine Konstante, wird der kleinstmögliche Datentyp berücksichtigt. Der Datentyp der Ausgangsvariable bleibt ohne Auswirkung auf die Rechenoperation.

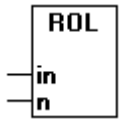
Beispiel in ST:



Beispiel:

```
LD 32

SHR 2

ST Var1 (* Ergebnis ist 8 *)
```

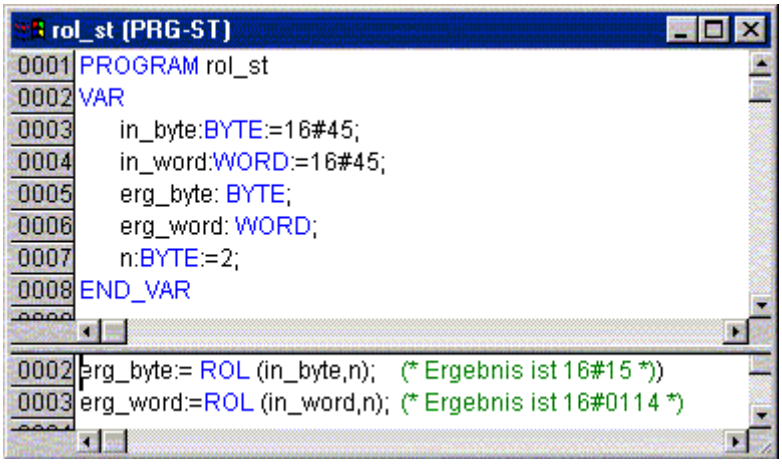
10.5.5.3 ROL



Bitweise Linksrotation eines Operanden: A:= ROL (IN, N)A, IN und N sollten vom Typ BYTE, WORD oder DWORD sein. IN wird N mal um eine Bitstelle nach links geschoben, wobei das linke Bit von rechts wieder eingeschoben wird.

> Beachten Sie, dass die Anzahl der Bits, die für die Rechenoperation berücksichtigt wird, durch den Datentyp der Eingangsvariable in vorgegeben wird. Handelt es sich hierbei um eine Konstante, wird der kleinstmögliche Datentyp berücksichtigt. Der Datentyp der Ausgangsvariable bleibt ohne Auswirkung auf die Rechenoperation.

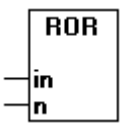
Beispiel in ST:



Beispiel:

```
Var1 :BYTE;

LD 2#1001_0011

ROL 3

ST Var1 (* Ergebnis ist 2#1001_1100 *)
```

10.5.5.4 ROR

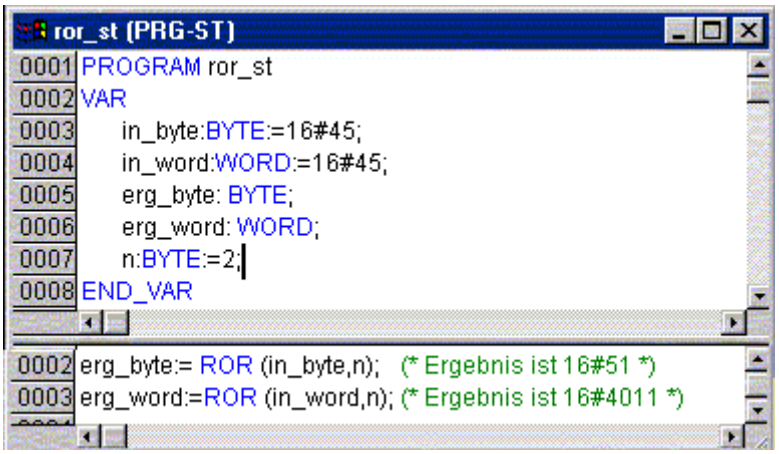


Bitweise Rechtsrotation eines Operanden: A:= ROR (IN, N)A, IN und N sollten vom Typ BYTE, WORD oder DWORD sein. IN wird N mal um eine Bitstelle nach rechts geschoben, wobei das rechte Bit von links wieder eingeschoben wird.

Beachten Sie, dass die Anzahl der Bits, die für die Rechenoperation berücksichtigt wird, durch den Datentyp der Eingangsvariable in vorgegeben wird. Handelt es sich hierbei um eine Konstante, wird der kleinstmögliche Datentyp berücksichtigt. Der Datentyp der Ausgangsvariable bleibt ohne Auswirkung auf die Rechenoperation.

Sehen Sie im nachfolgenden Beispiel in hexadezimaler Darstellung, wie sich bei gleichem Wert der Eingangsvariablen in_byte und in_word die Ergebnisse erg_byte und erg_word der Operation unterscheiden, je nachdem, ob in vom Typ BYTE oder WORD ist.

Beispiel in ST:



Beispiel in AWL:

```
Var1 :BYTE;

LD 2#1001_0011

ROR 3

ST Var1 (* Ergebnis ist 2#0111_0010 *)
```

10.5.6 Auswahloperatoren

10.5.6.1 SEL

Binäre Selektion.

OUT := SEL(G, IN0, IN1)

bedeutet:

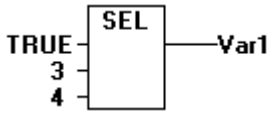
OUT := IN0 if G=FALSE;OUT := IN1 if G=TRUE.

IN0, IN1 und OUT können jeden Typ haben, G muss vom Typ BOOL sein. Das Ergebnis der Selektion ist IN0, wenn G FALSE ist, IN1, wenn G TRUE ist.

Beispiel in AWL:

```
LD TRUE

SEL 3,4

ST Var1 (* Ergebnis ist 4 *)


LD FALSE

SEL 3,4

ST Var1 (* Ergebnis ist 3 *)
```

Beispiel in FUP:



Zum Zweck der Laufzeitoptimierung wird folgendermaßen abgearbeitet: Ein Ausdruck, der IN0 vorgeschaltet ist, wird nur dann berechnet, wenn G FALSE ist. Ein Ausdruck der IN1 vorgeschaltet ist, wird nur dann berechnet, wenn G TRUE ist !

10.5.6.2 MAX

Maximumsfunktion. Liefert von zwei Werten den größten.

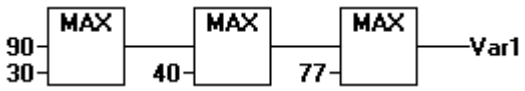
OUT := MAX(IN0, IN1)

IN0, IN1 und OUT können von beliebigem Typ sein.

Beispiel in AWL:

```
LD 90

MAX 30

MAX 40

MAX 77

ST Var1 (* Ergebnis ist 90 *)
```

Beispiel in FUP:



10.5.6.3 MIN

Minimumsfunktion. Liefert von zwei Werten den kleinsten.

OUT := MIN(IN0, IN1)

IN0, IN1 und OUT können von beliebigem Typ sein.

Beispiel in AWL:

```
LD 90

MIN 30

MIN 40

MIN 77

ST Var1 (* Ergebnis ist 30 *)
```

Beispiel in FUP:

10.5.6.4 LIMIT

Limitierung

OUT := LIMIT(Min, IN, Max)

bedeutet:

OUT := MIN (MAX (IN, Min), Max)

Max ist die obere, Min die untere Schranke für das Ergebnis. Wenn der Wert IN die obere Grenze Max überschreitet, dann liefert LIMIT Max. Wenn IN Min unterschreitet, dann ist das Ergebnis Min.IN und OUT können von beliebigem Typ sein.

Beispiel in AWL:

```
LD 90

LIMIT 30,80

ST Var1 (* Ergebnis ist 80 *)
```

10.5.6.5 MUX

Multiplexer

OUT := MUX(K, IN0,...,INn)

bedeutet:

OUT := INK.

IN0, ...,INn und OUT können von beliebigem Typ sein. K muss von den Typen BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT oder UDINT sein. MUX wählt aus einer Menge von Werten den K-ten aus. Ist K größer als die Anzahl der weiteren Eingänge (n) , so wird der letzte Wert weiter gegeben (INn).

Beispiel in AWL:

```
LD 0

MUX 30,40,50,60,70,80

ST Var1 (* Ergebnis ist 30 *)
```

Zum Zweck der Laufzeitoptimierung wird nur der Ausdruck, der INK vorgeschaltet ist, berechnet !

10.5.7 Vergleichsoperatoren

10.5.7.1 GT

Größer als

Ein Boolescher Operator mit dem Ergebnis TRUE, wenn der erste Operand größer als der zweite ist. Die Operanden können vom Typ BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME und STRING sein.

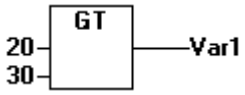
Beispiel in AWL:

```
LD 20

GT 30

ST Var1 (* Ergebnis ist FALSE *)
```

Beispiel in ST:

```
VAR1 := 20 > 30 > 40 > 50 > 60
> 70;
```

Beispiel in FUP:



10.5.7.2 LT

Kleiner als

Ein Boolescher Operator mit dem Ergebnis TRUE, wenn der erste Operand kleiner als der zweite ist. Die Operanden können vom Typ BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME und STRING sein.

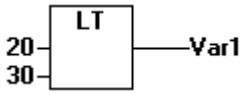
Beispiel in AWL:

```
LD 20

LT 30

ST Var1 (* Ergebnis ist TRUE *)
```

Beispiel in ST:

VAR1 := 20 < 30;

Beispiel in FUP:



10.5.7.3 LE

Kleiner oder gleich.

Ein Boolscher Operator mit Ergebnis TRUE, wenn der erste Operand kleiner als der zweite Operand oder gleich groß wie der zweite Operand ist. Die Operanden können vom Typ BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME und STRING sein.

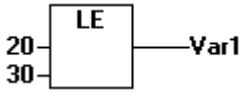
Beispiel in AWL:

```
LD 20

LE 30

ST Var1 (* Ergebnis ist TRUE *)
```

Beispiel in ST:

```
VAR1 := 20 <= 30;
```

Beispiel in FUP



10.5.7.4 GE

Größer oder gleich

Ein Boolescher Operator mit Ergebnis TRUE, wenn der erste Operand größer als der zweite Operand oder gleich groß wie der zweite Operand ist. Die Operanden können vom Typ BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME und STRING sein.

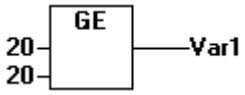
Beispiel in AWL:

```
LD 60

GE 40

ST Var1 (* Ergebnis ist TRUE *)
```

Beispiel in ST:

```
VAR1 := 60 >= 40;
```

Beispiel in FUP:



10.5.7.5 EQ

Gleichheit

Ein Boolescher Operator mit Ergebnis TRUE, wenn die Operanden gleich sind. Die Operanden können vom Typ BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME und STRING sein.

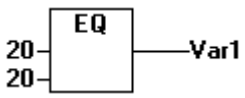
Beispiel in AWL:

```
LD 40

EQ 40

ST Var1 (* Ergebnis ist TRUE *)
```

Beispiel in ST:

```
VAR1 := 40 = 40;
```

Beispiel in FUP:



10.5.7.6 NE

Ungleichheit

Ein Boolescher Operator mit Ergebnis TRUE, wenn die Operanden ungleich sind. Die Operanden können vom Typ BOOL, BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL, LREAL, TIME, DATE, TIME_OF_DAY, DATE_AND_TIME und STRING sein.

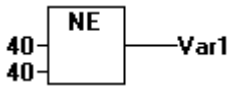
Beispiel in AWL:

```
LD 40

NE 40

ST Var1 (* Ergebnis ist FALSE *)
```

Beispiel in ST:

```
VAR1 := 40 <> 40;
```

Beispiel in FUP:



10.5.8 Verschiedene Operatoren

10.5.8.1 INDEXOF

Als Ergebnis liefert diese Funktion den internen Index eines Bausteins.

Beispiel in ST:

```
var1 := INDEXOF(baustein2);
```

10.5.8.2 SIZEOF

Als Ergebnis liefert diese Funktion die Anzahl der Bytes, die der angegebene Datentyp benötigt.

Beispiel in AWL:

```
arr1:ARRAY[0..4] OF INT;

var1:INT;

LD arr1

SIZEOF

ST var1 (* Ergebnis ist 10 *)
```

10.5.8.3 ADR (Adressoperator)

Adressfunktion, nicht von der Norm IEC61131-3 vorgeschrieben.

ADR liefert die Adresse seines Arguments in einem DWORD. Diese Adresse kann an Herstellerfunktionen geschickt und dort wie ein Pointer behandelt werden oder innerhalb des Projektes an einen Pointer zugewiesen werden.

HINWEIS

Wenn Online Change angewendet wird, können sich Inhalte von Adressen verschieben. Beachten Sie dies bei der Verwendung von Pointern auf Adressen.

Beispiel in ST:

```
dwVar :=
ADR(bVar);
```

Beispiel in AWL:

```
LD var1

ADR

ST var2
```

10.5.8.4 ADRINST(Instanzadressoperator)

Der ADRINST-Operator liefert die Adresse einer Programm- oder Funktionsbaustein-Instanz (ADRINST entspricht etwa dem C++ **this**-Operator).

10.5.8.5 ^ (Inhaltsoperator)

Die Dereferenzierung eines Pointers erfolgt über den Inhaltsoperator "^" nach dem Pointerbezeichner.

Beispiel in ST:

```
pt:POINTER TO INT;

var_int1:INT;

var_int2:INT;

pt := ADR(var_int1);

var_int2:=pt^;
```

10.5.8.6 CAL (Aufrufoperator)

Aufruf eines Funktionsblocks

Mit CAL ruft man in AWL die Instanz eines Funktionsblock auf. Nach dem Namen der Instanz eines Funktionsblocks folgt, in runde Klammern gesetzt, die Belegung der Eingabevariablen des Funktionsblocks.Beispiel: Aufruf der Instanz Inst eines Funktionsblocks mit Belegung der Eingabevariablen Par1, Par2 auf 0 bzw. TRUE.

Beispiel:

```
CAL INST(PAR1 := 0, PAR2 := TRUE)
```

10.5.8.7 BITADR

BITADR liefert die Bitadresse einer auf Einzeladresse allozierten SPS-Variablen.

Beispiel in ST:

```
bOFF AT%QX10.1 : BOOL;

iBitAdr : BYTE;
```

```
iBitAdr := BITADR( bOFF ); (* returns 81
*)
```

10.5.9 Typkonvertierung

10.5.9.1 BOOL_TO-Konvertierungen

Konvertierung vom Typ BOOL zu einem anderen Typ: Bei Zahlentypen ist das Ergebnis 1, wenn der Operand TRUE ist, und 0, wenn der Operand FALSE ist. Beim Typ STRING ist das Ergebnis 'TRUE' bzw. 'FALSE'.

Beispiele in ST:

```
i:=BOOL_TO_INT(TRUE); (* Ergebnis ist 1
*)

str:=BOOL_TO_STRING(TRUE); (* Ergebnis ist 'TRUE' *)

t:=BOOL_TO_TIME(TRUE); (* Ergebnis ist T#1ms *)

tof:=BOOL_TO_TOD(TRUE); (* Ergebnis ist TOD#00:00:00.001 *)

dat:=BOOL_TO_DATE(FALSE); (* Ergebnis ist D#1970-01-01 *)

dandt:=BOOL_TO_DT(TRUE); (* Ergebnis ist DT#1970-01-01-00:00:01
*)
```

10.5.9.2 TO_BOOL-Konvertierungen

Konvertierung von einem Typ zum Typ BOOL: Das Ergebnis ist TRUE, wenn der Operand ungleich 0 ist. Das Ergebnis ist FALSE, wenn der Operand gleich 0 ist. Beim Typ STRING ist das Ergebnis TRUE, wenn der Operand 'TRUE' ist, ansonsten ist das Ergebnis FALSE.

Beispiele in ST:

```
b := BYTE_TO_BOOL(2#11010101); (* Ergebnis ist
TRUE *)

b := INT_TO_BOOL(0); (* Ergebnis ist FALSE *)

b := TIME_TO_BOOL(T#5ms); (* Ergebnis ist TRUE *)

b := STRING_TO_BOOL('TRUE'); (* Ergebnis ist TRUE *)
```

10.5.9.3 STRING_TO-Konvertierungen

Konvertierung vom Typ STRING zu einem anderen Typ: Der Operand vom Typ STRING muss einen gültigen Wert des Zieltyps haben, sonst ist das Ergebnis 0.

Beispiele in ST:

```
b :=STRING_TO_BOOL('TRUE'); (*
Ergebnis ist TRUE *)

w :=STRING_TO_WORD('abc34'); (* Ergebnis ist 0 *)

t :=STRING_TO_TIME('T#127ms'); (* Ergebnis ist T#127ms
*)
```

10.5.9.4 TO_STRING-Konvertierungen

Konvertierung vom beliebeigen Typ zum STRING.

Beispiele in ST:

```
str:=BOOL_TO_STRING(TRUE); (* Ergebnis ist
'TRUE' *)

str :=TIME_TO_STRING(T#12ms); (* Ergebnis ist 'T#12ms'
*)
```

```
str :=DATE_TO_STRING(D#2002-08-18); (* Ergebnis ist
'D#2002-08-18' *)

str:=TOD_TO_STRING(TOD#14:01:05.123); (* Ergebnis ist
'TOD#14:01:05.123' *)

str:=DT_TO_STRING(DT#1998-02-13-14:20); (* Ergebnis ist
'DT#1998-02-13-14:20' *)

k := LREAL_TO_STRING(); (* Ergebnis ist '1.4' *)
```

10.5.9.5 TIME_TO-Konvertierungen

Konvertierung vom Typ TIME zu einem anderen Typ. Intern wird die Zeit in einem DWORD in Millisekunden abgespeichert. Dieser Wert wird konvertiert. Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen. Beim Typ STRING ist das Ergebnis die Zeitkonstante.

Beispiele in ST:

```
str :=TIME_TO_STRING(T#12ms); (* Ergebnis ist
'T#12ms' *)

dw:=TIME_TO_DWORD(T#5m); (* Ergebnis ist 300000 *)
```

10.5.9.6 DATE_TO-Konvertierungen

Konvertierung vom Typ DATE zu einem anderen Typ: Intern wird das Datum in einem DWORD in Sekunden seit dem 1.Januar 1970 abgespeichert. Dieser Wert wird konvertiert. Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen. Beim Typ STRING ist das Ergebnis die Datumskonstante.

Beispiele in ST:

```
b :=DATE_TO_BOOL(D#1970-01-01); (* Ergebnis
ist FALSE *)

i :=DATE_TO_INT(D#1970-01-15); (* Ergebnis ist 29952 *)

str :=DATE_TO_STRING(D#2002-08-18); (* Ergebnis ist
'D#2002-08-18' *)

vdt:=DATE_TO_DT(D#2002-08-18); (* Ergebnis ist DT#2002-08-18-00:00
*)

udw:=DATE_TO_DWORD(D#2002-08-18); (* Ergebniss ist 16#3D5EE380
*)
```

10.5.9.7 TOD_TO-Konvertierungen

Konvertierung vom Typ TIME_OF_DAY zu einem anderen Typ. Intern wird die Zeit in einem DWORD in Millisekunden abgespeichert (bei TIME_OF_DAY seit 00:00 Uhr). Dieser Wert wird konvertiert. Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen. Beim Typ STRING ist das Ergebnis die Zeitkonstante.

Beispiele in ST:

```
si:=TOD_TO_SINT(TOD#00:00:00.012); (* Ergebnis
ist 12 *)

str:=TOD_TO_STRING(TOD#14:01:05.123); (* Ergebnis ist
'TOD#14:01:05.123' *)

tm:= TOD_TO_TIME(TOD#14:01:05.123); (* Ergebnis ist T#841m5s123ms
*)

udi:= TOD_TO_UDINT(TOD#14:01:05.123); (* Ergebnis ist 16#03020963
*)
```

10.5.9.8 DT_TO-Konvertierungen

Konvertierung vom Typ DATE_AND_TIME zu einem anderen Typ: Intern wird das Datum in einem DWORD in Sekunden seit dem 1.Januar 1970 abgespeichert. Dieser Wert wird konvertiert. Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen. Beim Typ STRING ist das Ergebnis die Datumskonstante.

Beispiele in ST:

```
byt :=DT_TO_BYTE(DT#1970-01-15-05:05:05); (*
Ergebnis ist 129 *)

str:=DT_TO_STRING(DT#1998-02-13-14:20); (* Ergebnis ist
'DT#1998-02-13-14:20' *)

vtod:=DT_TO_TOD(DT#1998-02-13-14:20); (* Ergebnis ist TOD#14:20
*)

vdate:=DT_TO_DATE(DT#1998-02-13-14:20); (* Ergebnis ist
D#1998-02-13 *)

vdw:=DT_TO_DWORD(DT#1998-02-13-14:20); (* Ergebnis ist 16#34E45690
*)
```

10.5.9.9 REAL_TO-/LREAL_TO-Konvertierungen

Konvertierung vom Typ REAL bzw. LREAL zu einem anderen Typ: Es wird nach oben oder unten auf einen ganzzahligen Wert gerundet und in den entsprechenden Typen gewandelt. Ausgenommen davon sind die Typen STRING, BOOL, REAL und LREAL.Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen.

Beachten Sie bei der Konvertierung in den Typ STRING, dass die Gesamtkommastellenzahl auf 16 begrenzt ist. Enthält die (L)REAL-Zahl mehr Stellen, wird die sechzehnte Stelle gerundet und so im string dargestellt. Wenn der STRING für die Zahl zu kurz definiert ist, wird von rechts her entsprechend abgeschnitten.

Beispiel in ST:

```
i := REAL_TO_INT(1.5); (* Ergebnis ist 2
*)

j := REAL_TO_INT(1.4); (* Ergebnis ist 1 *)

k := LREAL_TO_STRING(); (* Ergebnis ist '1.4' *)
```

Beispiel in AWL:

```
LD 2.7

REAL_TO_INT

GE %MW8
```

10.5.9.10 Konvertierung ganzzahliger Werte

Konvertierung von einem ganzzahligen Zahlentyp zu einem anderen Zahlentyp: Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen. Wenn die zu konvertierende Zahl die Bereichsgrenze überschreitet, dann werden die ersten Bytes der Zahl nicht berücksichtigt.

Beispiel in ST:

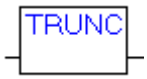
```
si := INT_TO_SINT(4223); (* Ergebnis ist 127
*)
```

Wenn sie die Integerzahl 4223 (16#107f in Hexadezimaldarstellung) in eine SINT-Variable speichern, dann enthält diese die Zahl 127 (16#7f in Hexadezimaldarstellung).

Beispiel in AWL:

```
LD 2

INT_TO_REAL

MUL 3.5
```

10.5.9.11 TRUNC



Konvertierung vom Typ REAL zum Typ INT. Es wird nur der Betrag des ganzzahligen Anteils der Zahl genommen. Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen.

Beispiele in ST:

```
i:=TRUNC(1.9); (* Ergebnis ist 1 *).
i:=TRUNC(-1.4); (* Ergebnis ist -1 *).
```

Beispiel in AWL:

```
LD 2.7
TRUNC
GE %MW8
```

Ähnliche Funktionen

LTRUNC

FLOOR

10.6 Operanden

10.6.1 Konstanten

10.6.1.1 BOOL-Konstanten

BOOL-Konstanten sind die Wahrheitswerte TRUE und FALSE.

10.6.1.2 TIME-Konstanten

In TwinCAT PLC Control können TIME-Konstanten deklariert werden. Insbesondere werden diese benutzt, um die Timer aus der Standardbibliothek zu bedienen. Eine TIME-Konstante besteht stets aus einem anführenden "t" oder "T" (bzw. "time" oder "TIME" in der ausführlichen Form) und einem Doppelkreuz "#". Danach kommt die eigentliche Zeitdeklaration, diese kann bestehen aus Tagen (bezeichnet mit "d"), Stunden (bezeichnet mit "h"), Minuten (bezeichnet mit "m"), Sekunden (bezeichnet mit "s") und Millisekunden (bezeichnet mit "ms"). Es ist zu beachten, dass die Zeitangaben der Größe nach geordnet sein müssen (d vor h vor m vor s vor m vor ms), wobei nicht alle Zeiten vorkommen müssen.

Beispiele für korrekte TIME-Konstanten in einer ST-Zuweisung:

```
TIME1 := T#14ms;

TIME1 := T#100S12ms; (*Überlauf in der höchsten Komponente ist
erlaubt*)

TIME1 := t#12h34m15s;
```

nicht korrekt wäre:

```
TIME1 := t#5m68s; (*Überlauf bei einer
niedrigeren Stelle*)

TIME1 := 15ms; (*Es fehlt T#*)

TIME1 := t#4ms13d; (*falsche Reihenfolge der Zeitangaben*)
```

10.6.1.3 DATE-Konstanten

Mit diesem Typ kann man Datumsangaben machen. Eine DATE-Konstante wird deklariert durch ein anführendes "d", "D", "DATE" oder "date" und ein nachfolgendes "#". Anschließend können Sie ein beliebiges Datum in der Reihenfolge Jahr-Monat-Tag eingeben.

Beispiele:

```
DATE#1996-05-06

d#1972-03-29
```

10.6.1.4 TIME_OF_DAY-Konstanten

Mit diesem Typ können Sie Uhrzeiten speichern. Eine TIME_OF_DAY-Deklaration beginnt mit "tod#", "TOD#", "TIME_OF_DAY#" oder "time_of_day#", anschließend können Sie eine Uhrzeit angeben in der Schreibweise: Stunde:Minute:Sekunde. Sekunden können dabei als reelle Zahlen angegeben werden, es können also auch Sekundenbruchteile angegeben werden.

Beispiele:

```
TIME_OF_DAY#15:36:30.123

tod#00:00:00
```

10.6.1.5 DATE_AND_TIME-Konstanten

Datumskonstanten und Uhrzeiten können auch kombiniert werden zu sogenannten DATE_AND_TIME-Konstanten. DATE_AND_TIME-Konstanten beginnen mit "dt#", "DT#", "DATE_AND_TIME#" oder "date_and_time#". Nach der Datumsangabe folgt ein Bindestrich und danach die Uhrzeit.

Beispiele:

```
DATE_AND_TIME#1996-05-06-15:36:30

dt#1972-03-29-00:00:00
```

10.6.1.6 Numerische Konstanten

Zahlenwerte können als Dualzahlen, Oktalzahlen, Dezimalzahlen und Hexadezimalzahlen auftreten. Wenn ein Integerwert keine Dezimalzahl ist, dann muss seine Basis gefolgt von einem Doppelkreuz (#) vor die Integerkonstante geschrieben werden. Die Ziffernwerte für die Zahlen 10 bis 15 bei Hexadezimalzahlen werden wie üblich durch die Buchstaben A-F angegeben. Unterstriche innerhalb eines Zahlenwertes sind erlaubt.

Beispiele:

```
14 (Dezimalzahl)

2#1001_0011 (Dualzahl)

8#67 (Oktalzahl)

16#A (Hexadezimalzahl)
```

Der Typ dieser Zahlenwerte kann dabei BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT, REAL oder LREAL sein. Implizite Konvertierungen von "größere" auf "kleinere" Typen sind nicht erlaubt. D.h. eine DINT- Variable kann nicht ohne weiteres als INT-Variable benutzt werden.

10.6.1.7 REAL-/LREAL-Konstanten

REAL- und LREAL-Konstanten können als Dezimalbrüche und in Exponentialdarstellung angegeben werden. Man verwendet hierbei die amerikanische Schreibweise mit Punkt.

Beispiel:

```
7.4 statt 7,4

1.64e+009 statt 1,64e+009
```

10.6.1.8 STRING-Konstanten

Ein String ist eine beliebige Zeichenreihe. STRING-Konstanten werden mit einfachen Hochkommas vorn und hinten begrenzt. Es können auch Leerzeichen und Umlaute eingegeben werden. Sie werden genauso wie alle anderen Zeichen behandelt. In Zeichenfolgen wird die Kombination des Dollarzeichens ($) gefolgt von zwei hexadezimalen Ziffern als hexadezimale Darstellung des acht Bit Zeichencodes interpretiert. Außerdem werden, wenn sie in einer Zeichenfolge auftauchen, Kombinationen von zwei Zeichen, die mit dem Dollarzeichen beginnen, wie folgt interpretiert:

Sonderzeichen	Beschreibung
$$	Dollarzeichen
$'	Hochkomma
$L or $l	Zeilenvorschub
$N or $n	Neue Zeile
$P or $p	Seitenvorschub
$R or $r	Zeilenumbruch
$T or $t	Tabulator

Beispiele:

```
'w1Wüß?'

'Susi und Claus'

':-)'
```

10.6.1.9 Typisierte Konstanten

Grundsätzlich wird bei der Verwendung von IEC-Konstanten der kleinstmögliche Datentyp verwendet. Soll ein anderer Datentyp verwendet werden, kann dies mithilfe von Typed Literals erreicht werden, ohne dass die Konstante explizit deklariert werden muss. Die Konstante wird hierbei mit einem Präfix versehen, welches den Typ festlegt:

Die Schreibweise ist: <Type>#<Literal>

<Type> gibt den gewünschten Datentyp an, mögliche Eingaben: BOOL, SINT, USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD, REAL, LREAL. Der Typ muss in Großbuchstaben geschrieben werden.

<Literal> gibt die Konstante an. Die Eingabe muss zum unter <Type> angegebenen Datentypen passen.

Beispiel:

```
var1:=DINT#34;
```

Kann die Konstante nicht ohne Datenverlust in den Zieltyp überführt werden, so wird eine Fehlermeldung ausgegeben:

Typed literals können überall dort verwendet werden, wo normale Konstanten verwendet werden können.

10.6.2 Variablen

Variablen werden entweder lokal im Deklarationsteil eines Bausteins deklariert oder in den globalen Variablenlisten.

Für den Bezeichner von Variablen ist zu beachten, dass sie keine Leerstellen und Umlaute enthalten dürfen, sie dürfen nicht doppelt deklariert werden und nicht identisch mit Schlüsselwörtern sein. Groß-/Kleinschreibung bei Variablen wird nicht beachtet, das heißt VAR1, Var1 und var1 sind keine unterschiedlichen Variablen. Unterstriche sind in Bezeichner signifikant, z.B. werden "A_BCD" und "AB_CD" als unterschiedliche Bezeichner interpretiert. Mehrfach aufeinanderfolgende Unterstriche am Anfang eines Bezeichners oder in einem Bezeichner sind nicht erlaubt. Die Bezeichnerlänge sowie der signifikante Bereich sind unbegrenzt. Variablen können überall verwendet werden, wo der deklarierte Typ es erlaubt.

10.6.2.1 Adressen

Die direkte Darstellung einzelner Speicherzellen erfolgt mittels spezieller Zeichenreihen. Diese entstehen aus der Konkatenation des Prozentzeichens "%", einem Bereichspräfix, einem Präfix für die Größe und einem oder mehreren natürlichen Zahlen, die durch Leerzeichen voneinander getrennt sind. Folgende Bereichspräfixe werden unterstützt:

Bestimmungszeichen	Beschreibung
I	Eingang
Q	Ausgang
M	Merker

Folgende Präfixe für die Größe werden unterstützt:

Präfix	Beschreibung
X	Einzelbit
B	Byte (8 Bit)
W	Word (16 Bit)
D	Doppelwort (32 Bit)
*	Konfigurationsvariablen (VAR_CONFIG)

Beispiele:

```
%QX75.1 (*Bit 1 des Ausgangsbytes 75*)
%IW215 (*Eingangswort 215*)
%QB7 (*Ausgangsbyte 7*)
%MD48 (*Doppelwort an der Speicherstelle 48 im Merkerbereich*)
```

Boolsche Werte werden byteweise alloziert, wenn die nicht explizit eine Einzelbitadresse angegeben wird.

Beispiel: Eine Wertänderung von varbool1 AT %QW0 betrifft den Bereich von QX0.0 bis QX0.7.

Merker

Man kann alle unterstützten Größen für den Zugriff auf den Merker benutzen. Zum Beispiel würde die Adresse %MD48 die Bytes Nr. 48, 49, 50 und 51 im Merkerbereich adressieren. Das erste Byte ist das Byte Nr. 0. Ebenso kann man auf Worte und Bytes und sogar auf Bits zugreifen: Mit %MX5.0 etwa greift man auf das erste Bit im fünften Wort zu.

HINWEIS

Bei Verwendung anderer Plattformen:

Bei einem CX9xxx und CP mit ARM Plattform muss zwingend ein 4-Byte-Alignment eingehalten werden.

Beispiel:

```
rMyVar1      AT %MW0       : REAL;
rMyVar2      AT %MW4       : REAL;
```

10.6.2.2 Array-/Struktur-/FB-/Programm-Variablen

Auf Komponenten von zweidimensionalen Arrays greift man mit folgender Syntax zu:

```
<Feldname>[Index1, Index2]
```

Auf Variablen von Strukturen greift man mit folgender Syntax zu:

```
<Strukturname>.<Variablenname>
```

Auf Variablen von Funktionsblöcken und Programmen greift man mit folgender Syntax zu:

```
<Bausteinname>.<Variablenname>
```

10.6.2.3 Adressierung von Bits in Variablen

In ganzzahligen Variablen können einzelne Bits angesprochen werden. Dazu wird an die Variable mit einem Punkt abgetrennt der Index des zu adressierenden Bits angehängt. Der Bit-Index kann durch eine beliebige Konstante gegeben werden. Die Indizierung ist 0-basiert.
Beispiel:

```
a : INT;

b : BOOL;

...

a.2 := b;
```

Das dritte Bit der Variablen a wird auf den Wert der Variable b gesetzt.

Ist der Index größer als die Bit-Breite der Variablen so wird folgender Fehler ausgegeben:
Index '<n>' außerhalb des gültigen Bereichs für Variable '<var>'!

Die Bitadressierung ist bei folgenden Variablentypen möglich:
SINT, INT, DINT, USINT, UINT, UDINT, BYTE, WORD, DWORD.

Ist der Typ der Variablen nicht zulässig, so wird folgende Fehlermeldung ausgegeben:
Unzulässiger Datentyp '<Typ>' für direkte Indizierung".

Ein Bit-Zugriff darf nicht einer VAR_IN_OUT-Variablen zugewiesen werden !

10.6.2.4 Funktionen

Im ST kann auch ein Funktionsaufruf als Operand auftreten.

Beispiel:

```
Ergebnis := Fct(7) + 3;
```

10.6.2.5 System Flags

10.6.2.5.1 Systemflags

Systemflags sind implizit deklarierte Variablen, die von Ihrer speziellen Steuerung abhängig sind. Um herauszufinden, welche Systemflags Ihr System besitzt, wählen Sie den Befehl '**Einfügen"Operand**', es erscheint der Eingabehilfedialog, hier wählen Sie die Kategorie **System Variable**.

10.6.2.5.2 SYSTEMINFO

```
VAR_GLOBAL
    SystemInfo     AT%MB32768(*The real address may differ!*) : SYSTEMINFOTYPE;
END_VAR
```

Systemflags sind implizit deklarierte Variablen. Mit der Eingabehilfe finden Sie unter Systemvariablen eine Variable Systeminfo. Der Typ SYSTEMINFOTYPE ist in der System-Bibliothek deklariert. Um auf die Variable zugreifen zu können muss die System-Bibliothek in das Projekt eingebunden werden.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PLCSystem.Lib
TwinCAT v2.8.0	PC (i386)	TcSystem.Lib

10.6.2.5.3 SYSTEMTASKINFOARR

```
VAR_GLOBAL
    SystemTaskInfoArr       AT%MB32832(*The real address may differ!
*) : ARRAY[1..4] OF SYSTEMTASKINFOTYPE;
END_VAR
```

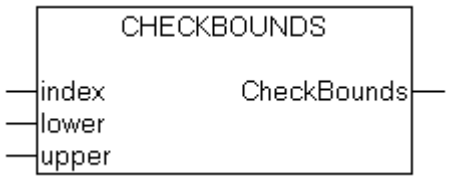
Systemflags sind implizit deklarierte Variablen. Mit der Eingabehilfe finden Sie unter Systemvariablen eine Variable SystemTaskInfoArr. Diese Variable ist ein Feld von vier Strukturen des Types SYTEMTASKINFOTYPE. Die Strukturdefinition ist in der System-Bibliothek zu finden. Der Index in dieses Feld ist die Task-Id. Um in der SPS zur Laufzeit die Task-Id feststellen zu können, gibt es den Funktionsbaustein GETCURTASKINDEX. Dieser Funktionsbaustein liefert die Task-Id der Task aus der er aufgerufen wurde zurück. Um auf die Variable zugreifen zu können muss die System-Bibliothek in das Projekt eingebunden werden.

Voraussetzungen

Entwicklungsumgebung	Zielplattform	Einzubindende SPS Bibliotheken
TwinCAT v2.7.0	PC (i386)	PLCSystem.Lib
TwinCAT v2.8.0	PC (i386)	TcSystem.Lib

10.7 System Funktionen

10.7.1 FUNCTION CheckBounds



Wenn Sie in Ihrem Projekt eine Funktion mit Namen CheckBounds definieren, können Sie damit Bereichsüberschreitungen beim Zugriff auf Arrays in Ihrem Projekt automatisch überprüfen! Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen. Ein Beispiel für eine Implementierung dieser Funktion ist nachfolgend abgedruckt.

HINWEIS

Systemauslastung

Die Funktion kann eine erhebliche Erhöhung der Systemauslastung verursachen, deshalb sollte sie nur für Testzwecke eingesetzt werden.

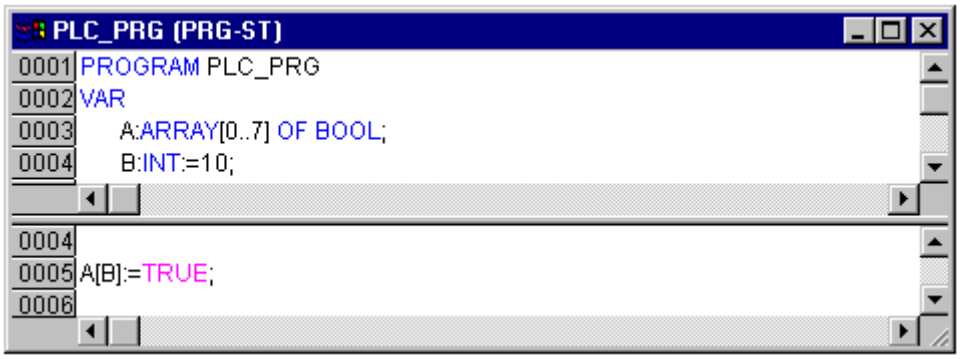
FUNCTION CheckBounds : DINT

```
VAR_INPUT
    index, lower, upper : DINT;
END_VAR
```

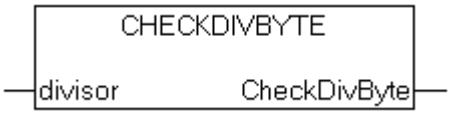
Beispiel für eine Implementierung der CheckBounds-Funktion:

```
IF index<lower THEN
    CheckBounds := lower;
ELSIF index>upper THEN
    CheckBounds := upper;
ELSE
    CheckBounds := index;
END_IF
```

Das folgende Beispielprogramm zum Testen der CheckBounds-Funktion greift außerhalb der Grenzen eines definierten Arrays zu. Die Funktion CheckBounds gewährleistet, dass der Wert TRUE nicht an die Stelle A[10], sondern an der oberen noch gültigen Bereichsgrenze A[7] zugewiesen wird. Mit der CheckBounds-Funktion können somit Zugriffe außerhalb von Array-Grenzen korrigiert werden.



10.7.2 FUNCTION CheckDivByte : BYTE



Wenn Sie in Ihrem Projekt eine Funktion mit dem Namen CheckDivByte definieren, können Sie damit bei Verwendung des Operators DIV [▶ 320] den Wert des Divisors überprüfen, beispielsweise um eine Division durch 0 zu verhindern. Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

HINWEIS

Systemauslastung

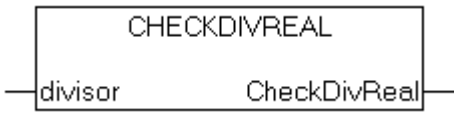
Die Funktion kann eine erhebliche Erhöhung der Systemauslastung verursachen, deshalb sollte sie nur für Testzwecke eingesetzt werden.

```
VAR_INPUT
    divisor : BYTE;
END_VAR
```

Beispiel für die Implementierung der Funktion CheckDivByte:

```
IF divisor = 0 THEN
    CheckDivByte := 1;
ELSE
    CheckDivByte := divisor;
END_IF
```

10.7.3 FUNCTION CheckDivReal : REAL



Wenn Sie in Ihrem Projekt eine Funktion mit dem Namen CheckDivReal definieren, können Sie damit bei Verwendung des Operators DIV [▶ 320] den Wert des Divisors überprüfen, beispielsweise um eine Division durch 0 zu verhindern. Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

HINWEIS

Systemauslastung

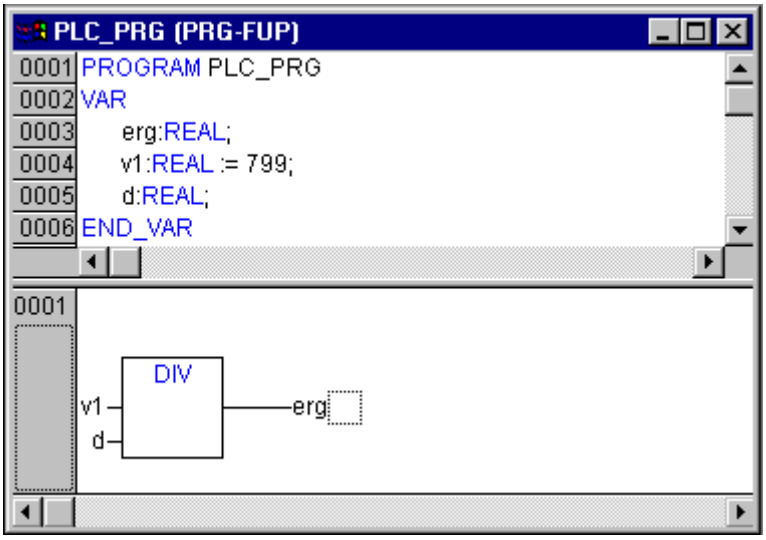
Die Funktion kann eine erhebliche Erhöhung der Systemauslastung verursachen, deshalb sollte sie nur für Testzwecke eingesetzt werden.

```
VAR_INPUT
    divisor : REAL;
END_VAR
```

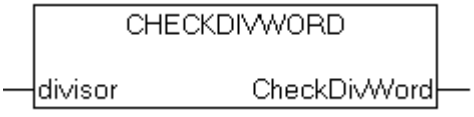
Beispiel für die Implementierung der Funktion CheckDivReal:

```
IF divisor = 0 THEN
    CheckDivReal := 1;
ELSE
    CheckDivReal := divisor;
END_IF
```

Das Ergebnis der Funktion CheckDivReal wird vom Operator DIV als Divisor eingesetzt. Im nachfolgend dargestellten Beispielprogramm wird dadurch verhindert, dass durch 0 geteilt wird, der Divisor (d) wird von 0 auf 1 gesetzt. Das Ergebnis erg der Division ist dementsprechend 799.



10.7.4 FUNCTION CheckDivWord : WORD



Wenn Sie in Ihrem Projekt eine Funktion mit dem Namen CheckDivWord definieren, können Sie damit bei Verwendung des Operators DIV [▶ 320] den Wert des Divisors überprüfen, beispielsweise um eine Division durch 0 zu verhindern. Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

HINWEIS

Systemauslastung

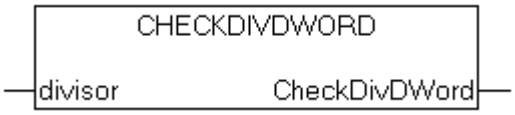
Die Funktion kann eine erhebliche Erhöhung der Systemauslastung verursachen, deshalb sollte sie nur für Testzwecke eingesetzt werden.

```
VAR_INPUT
    divisor : WORD;
END_VAR
```

Beispiel für die Implementierung der Funktion CheckDivWord:

```
IF divisor = 0 THEN
    CheckDivWord := 1;
ELSE
    CheckDivWord := divisor;
END_IF
```

10.7.5 FUNCTION CheckDivDWord : DWORD



Wenn Sie in Ihrem Projekt eine Funktion mit dem Namen CheckDivDWord definieren, können Sie damit bei Verwendung des Operators DIV [▶ 320] den Wert des Divisors überprüfen, beispielsweise um eine Division durch 0 zu verhindern. Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.

HINWEIS

Systemauslastung

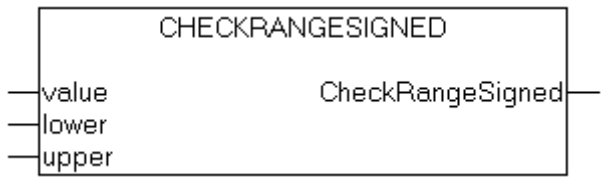
Die Funktion kann eine erhebliche Erhöhung der Systemauslastung verursachen, deshalb sollte sie nur für Testzwecke eingesetzt werden.

```
VAR_INPUT
    divisor : DWORD;
END_VAR
```

Beispiel für die Implementierung der Funktion CheckDivDWord:

```
IF divisor = 0 THEN
    CheckDivDWord := 1;
ELSE
    CheckDivDWord := divisor;
END_IF
```

10.7.6 FUNCTION CheckRangeSigned : DINT



Um die Einhaltung der Bereichsgrenzen der Unterbereichstypen zur Laufzeit überprüfen zu können, kann die Funktion CheckRangeSigned eingefügt werden. In der Funktion können Bereichsverletzungen in geeigneter Art und Weise abgefangen werden (z.B. kann der Wert abgeschnitten werden oder ein Fehlerflag gesetzt werden). Die Funktion wird implizit aufgerufen, sobald auf eine Variable geschrieben wird, die von einem Unterbereichstyp ist, der aus einem vorzeichenbehafteten Typ gebildet wurde. Um eine Überprüfung vorzeichenloser Unterbereichstypen durchführen zu können muss die Funktion CheckRangeUnsigned [▶ 346] benutzt werden.

```
VAR_INPUT
    value, lower, upper: DINT;
END_VAR
```

Beispiel:

Im Falle einer Variable eines vorzeichenbehafteten Unterbereichstyps (also wie i von oben) wird die Funktion CheckRangeSigned aufgerufen, die folgendermaßen programmiert sein könnte, um einen Wert auf den erlaubten Bereich zurückzuschneiden:

```
FUNCTION CheckRangeSigned : DINT

VAR_INPUT

    value, lower, upper: DINT;

END_VAR


IF (value < lower) THEN

CheckRangeSigned := lower;

ELSIF(value > upper) THEN

CheckRangeSigned := upper;

ELSE

CheckRangeSigned := value;

END_IF
```

Zwingend für einen automatischen Aufruf ist der Funktionsname CheckRangeSigned und die Gestaltung der Schnittstelle: Rückgabewert und drei Parameter vom Typ DINT.
Die Funktion wird beim Aufruf folgendermaßen parametriert:

value	bekommt den Wert, der dem Bereichstypen zugewiesen werden soll
lower	die Untergrenze des Bereichs
upper	die Obergrenze des Bereichs
Rückgabewert	der Wert, der tatsächlich dem Bereichstypen zugewiesen wird

Aus einer Zuweisung i := 10*y; wird in diesem Beispiel implizit folgende erzeugt:
i := CheckRangeSigned(10*y, -4095, 4095);

Wenn y beispielsweise den Wert 1000 hat, dann hat i nach dieser Zuweisung trotzdem nur den Wert 4095.

HINWEIS

Wert der Variablen/Endlosschleife

Sind die beiden Funktionen CheckRangeSigned und CheckRangeUnsigned [▶ 346] nicht vorhanden, findet zur Laufzeit keine Typüberprüfung der Unterbereichstypen statt! Die Variable i könnte dann also durchaus beliebige Werte zwischen -32768 und 32767 annehmen!

Wenn eine Funktion CheckRangeSigned bzw. CheckRangeUnsigned wie oben gezeigt implementiert ist, kann bei der Verwendung des Unterbereichstypen in einer FOR-Schleife eine Endlosschleife entstehen. Dies geschieht genau dann, wenn der für die **FOR-Schleife** angegebenen Bereich genauso groß oder größer ist als der des Unterbereichstypen !

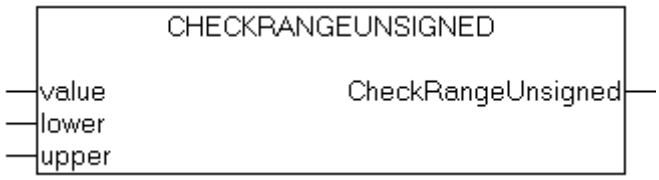
Beispiel:

```
VAR

    ui : UINT (0..10000);

END_VAR


FOR ui:=0 TO 10000 DO
```

```
...
END_FOR
```

Die FOR-Schleife wird nicht verlassen, da ui nicht größer als 10000 werden kann.
Ebenso ist der Inhalt der CheckRange-Funktionen bei der Verwendung von Inkrementationswerten in der FOR-Schleife zu beachten !

10.7.7 FUNCTION CheckRangeUnsigned : UDINT



Um die Einhaltung der Bereichsgrenzen der Unterbereichstypen zur Laufzeit überprüfen zu können, kann die Funktion CheckRangeUnsigned eingefügt werden. In der Funktion können Bereichsverletzungen in geeigneter Art und Weise abgefangen werden (z.B. kann der Wert abgeschnitten werden oder ein Fehlerflag gesetzt werden). Die Funktion wird implizit aufgerufen, sobald auf eine Variable geschrieben wird, die von einem Unterbereichstyp ist, der aus einem nicht vorzeichenbehafteten Typ gebildet wurde. Um eine Überprüfung vorzeichenbehafteten Unterbereichstypen durchführen zu können muss die Funktion CheckRangeSigned [▶ 344] benutzt werden.

```
VAR_INPUT
    value, lower, upper: UDINT;
END_VAR
```

Beispiel für eine Implementierung: siehe CheckRangeSigned [▶ 344]

HINWEIS

Wert der Variablen/Endlosschleife

Sind die beiden Funktionen CheckRangeSigned und CheckRangeUnsigned nicht vorhanden, findet zur Laufzeit keine Typüberprüfung der Unterbereichstypen statt! Die Variable i könnte dann also durchaus beliebige Werte zwischen -32768 und 32767 annehmen!

Wenn eine Funktion CheckRangeSigned bzw. CheckRangeUnsigned wie oben gezeigt implementiert ist, kann bei der Verwendung des Unterbereichstypen in einer FOR-Schleife eine Endlosschleife entstehen. Dies geschieht genau dann, wenn der für die **FOR-Schleife** angegebenen Bereich genauso groß oder größer ist als der des Unterbereichstypen !

Beispiel:

```
VAR

    ui : UINT (0..10000);

END_VAR


FOR ui:=0 TO 10000 DO

...

END_FOR
```

Die FOR-Schleife wird nicht verlassen, da ui nicht größer als 10000 werden kann.
Ebenso ist der Inhalt der CheckRange-Funktionen bei der Verwendung von Inkrementationswerten in der FOR-Schleife zu beachten !

Mehr Informationen:
www.beckhoff.de/tx1200

Beckhoff Automation GmbH & Co. KG
Hülshorstweg 20
33415 Verl
Deutschland
Telefon: +49 5246 9630
info@beckhoff.de
www.beckhoff.de