

Lenze

Handbuch



Global Drive

Drive PLC Developer Studio



Dieses Handbuch ist gültig für das **Drive PLC Developer Studio V02.00**.

Wichtiger Hinweis:

Die Software wird dem Benutzer in der vorliegenden Form zur Verfügung gestellt. Alle Risiken hinsichtlich der Qualität und der durch ihren Einsatz ermittelten Ergebnisse verbleiben beim Benutzer. Entsprechende Sicherheitsvorkehrungen gegen eventuelle Fehlbedienungen sind vom Benutzer vorzusehen.

Wir übernehmen keine Verantwortung für direkt oder indirekt entstandene Schäden, z. B. Gewinnverluste, Auftragsverluste oder geschäftliche Beeinträchtigungen jeglicher Art.

© 2005 Lenze Drive Systems GmbH

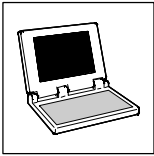
Ohne besondere schriftliche Genehmigung von Lenze Drive Systems GmbH darf kein Teil dieser Dokumentation vervielfältigt oder Dritten zugänglich gemacht werden.

Wir haben alle Angaben in dieser Dokumentation mit größter Sorgfalt zusammengestellt und auf Übereinstimmung mit der beschriebenen Hard- und Software geprüft. Trotzdem können wir Abweichungen nicht ganz ausschließen. Wir übernehmen keine juristische Verantwortung oder Haftung für Schäden, die dadurch eventuell entstehen. Notwendige Korrekturen werden wir in die nachfolgenden Auflagen einarbeiten.

Windows, Windows NT und MS-DOS sind entweder eingetragene Warenzeichen oder Warenzeichen der Microsoft Corporation in den U.S.A und/oder anderen Ländern.

IBM und VGA sind eingetragene Warenzeichen von International Business Machines, Inc.

Alle anderen Markennamen sind Warenzeichen ihrer jeweiligen Besitzer.



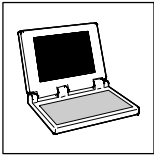
1	Vorwort und Allgemeines	1-1
1.1	Über dieses Handbuch	1-1
1.1.1	Verwendete Begriffe	1-1
1.2	Verwendete Konventionen	1-1
2	Einführung	2-1
2.1	Die Funktionen im Überblick	2-1
2.2	Bestandteile eines Projekts	2-2
2.2.1	Projekt	2-2
2.2.2	Baustein (POE)	2-2
2.2.3	Funktion	2-2
2.2.4	Funktionsblock	2-4
2.2.5	Programm	2-6
2.2.6	PLC_PRG	2-8
2.2.7	System-POE's	2-8
2.2.8	Ressourcen	2-8
2.2.9	Bibliotheken	2-8
2.2.10	Datentypen	2-9
2.2.11	Visualisierung	2-9
2.3	Debugging, Onlinefunktionalitäten	2-10
2.3.1	Debugging	2-10
2.3.2	Breakpoint	2-10
2.3.3	Einzelschritt	2-10
2.3.4	Einzelzyklus	2-10
2.3.5	Werte Online verändern	2-11
2.3.6	Monitoring	2-11
2.3.7	Simulation	2-12
2.3.8	Logbuch	2-12
3	Programmbeispiel "Ampelanlage"	3-1
3.1	Einleitung	3-1
3.2	Programmierung	3-2
3.2.1	DDS starten	3-2
3.2.2	Neues Projekt anlegen	3-2
3.2.3	PLC auswählen	3-2
3.2.4	Bausteine erzeugen	3-2
3.2.5	Der Baustein TRAFFICLIGHT	3-3
3.2.6	Der Baustein WAIT	3-6
3.2.7	Das Hauptprogramm PLC_PRG	3-8
3.2.8	Erweiterung des Programms um eine Alternativverzweigung	3-11
3.3	Simulation	3-14
3.4	Visualisierung	3-15
3.4.1	Erstellen einer neuen Visualisierung	3-15
3.4.2	Elemente in die Visualisierung einfügen und konfigurieren	3-15
3.4.3	Visualisierung im Online-Modus ablaufen lassen	3-16



Drive PLC Developer Studio

Inhalt

4	Programmiersprachen	4-1
4.1	Die Norm IEC 61131-3	4-1
4.2	Anweisungsliste (AWL)	4-2
4.2.1	Operatoren und Modifikatoren	4-2
4.3	Strukturierter Text (ST)	4-4
4.3.1	Ausdrücke	4-4
4.3.2	Auswertung von Ausdrücken	4-4
4.3.3	Anweisungen (Überblick)	4-5
4.3.4	Zuweisungsoperator	4-5
4.3.5	Aufruf eines Funktionsblockes in ST	4-6
4.3.6	RETURN-Anweisung	4-6
4.3.7	IF-Anweisung	4-6
4.3.8	CASE-Anweisung	4-7
4.3.9	FOR-Schleife	4-8
4.3.10	WHILE-Schleife	4-9
4.3.11	REPEAT-Schleife	4-10
4.3.12	EXIT-Anweisung	4-10
4.4	Ablaufsprache (AS)	4-11
4.4.1	Schritt	4-11
4.4.2	Aktion	4-11
4.4.3	Eingangs- bzw. Ausgangsaktion	4-12
4.4.4	Transition/Transitionsbedingung	4-12
4.4.5	Aktiver Schritt	4-13
4.4.6	IEC-Schritt	4-14
4.4.7	Qualifier	4-15
4.4.8	Implizite Variablen in der Ablaufsprache	4-15
4.4.9	Flags in der Ablaufsprache	4-16
4.4.10	Alternativzweig	4-17
4.4.11	Parallelzweig	4-17
4.4.12	Sprung	4-18
4.5	Funktionsplan (FUP)	4-19
4.6	Der Freigraphische Funktionsplaneditor (CFC)	4-20
4.7	Kontaktplan (KOP)	4-21
4.7.1	Kontakt	4-21
4.7.2	Spule	4-21
4.7.3	Set/Reset-Spule	4-22
4.7.4	Funktionsblöcke im Kontaktplan	4-22
4.7.5	KOP als FUP	4-22
5	Arbeitsbereich	5-1
5.1	Bedienoberfläche	5-1
5.1.1	Menüleiste	5-1
5.1.2	Funktionsleiste	5-2
5.1.3	Object Organizer	5-2
5.1.4	Vertikaler Bildschirmteiler	5-3
5.1.5	Arbeitsbereich	5-3
5.1.6	Meldungsfenster	5-3
5.1.7	Statusleiste	5-4
5.1.8	Kontextmenü	5-4



5.2	Fenster anordnen	5-5
5.2.1	Befehle im Menü "Fenster"	5-5
5.3	Grundeinstellungen	5-6
5.3.1	Optionen des DDS	5-6
6	Arbeiten mit Projekten und Objekten	6-1
6.1	Projekte verwalten	6-1
6.1.1	Befehle im Menü "Datei"	6-1
6.1.2	Befehle im Menü "Projekt"	6-8
6.1.3	Arbeitsgruppen	6-21
6.2	Arbeiten mit Objekten	6-24
6.2.1	Objekt	6-24
6.2.2	Ordner	6-25
6.2.3	Befehle im Kontextmenü	6-25
6.2.4	Befehle im Menü "Projekt"	6-26
6.3	Arbeiten im Online-Modus	6-32
6.3.1	Befehle im Menü "Online"	6-32
6.4	Logbuch	6-43
6.4.1	Logbuch-Eigenschaften	6-43
7	Editoren	7-1
7.1	Allgemeine Editierfunktionen	7-1
7.1.1	Befehle im Menü "Bearbeiten"	7-1
7.2	Deklarationseditor	7-9
7.2.1	Deklarationsteil	7-10
7.2.2	Eingangsvariablen	7-10
7.2.3	Ausgangsvariablen	7-10
7.2.4	Ein-/Ausgangsvariablen	7-11
7.2.5	Lokale Variablen	7-11
7.2.6	Remanente Variablen	7-11
7.2.7	Konstanten, Typed Literals	7-12
7.2.8	Remanente Konstanten	7-12
7.2.9	Schlüsselwörter	7-13
7.2.10	Bezeichner	7-13
7.2.11	Variablendeklaration	7-13
7.2.12	AT-Deklaration	7-14
7.2.13	Syntaxcoloring	7-15
7.2.14	Kurzformmodus	7-15
7.2.15	Automatisch deklarieren	7-16
7.2.16	Zeilennummern im Deklarationseditor	7-17
7.2.17	Deklarationen als Tabelle	7-18
7.2.18	Deklarationseditor im Online-Modus	7-19
7.2.19	Kommentar	7-19
7.3	Texteditoren	7-22
7.3.1	Befehle im Menü "Einfügen"	7-22
7.3.2	Die Texteditoren im Online-Modus	7-23
7.3.3	Breakpoint-Positionen	7-25
7.3.4	Was passiert an einem Breakpoint?	7-26
7.3.5	Zeilennummern des Texteditors	7-26



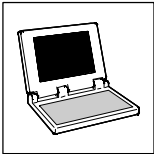
Drive PLC Developer Studio

Inhalt

7.4	Netzwerkeditoren (allgemein)	7-27
7.4.1	Sprungmarken	7-27
7.4.2	Netzwerkcommentare	7-27
7.4.3	Neues Netzwerk einfügen	7-27
7.4.4	Inputs/Outputs on the fly	7-28
7.4.5	Netzwerkeditoren im Online-Modus	7-29
7.5	Funktionsplaneditor	7-30
7.5.1	Cursorpositionen im FUP	7-30
7.5.2	Cursor setzen	7-31
7.5.3	Befehle im Menü "Einfügen"	7-31
7.5.4	Befehle im Menü "Extras"	7-34
7.5.5	Befehle im Menü "Bearbeiten"	7-35
7.5.6	Der Funktionsplan-Editor im Online Modus	7-36
7.6	Kontaktplaneditor	7-37
7.6.1	Cursorpositionen im KOP	7-37
7.6.2	Befehle im Menü "Einfügen"	7-37
7.6.3	Bausteine mit EN-Eingängen	7-39
7.6.4	Befehle im Menü "Extras"	7-40
7.6.5	Der Kontaktplan im Online Modus	7-42
7.7	CFC-Editor	7-43
7.7.1	Cursorpositionen im CFC	7-44
7.7.2	Elemente selektieren	7-44
7.7.3	Elemente kopieren, löschen	7-44
7.7.4	Elemente verschieben	7-45
7.7.5	Befehle im Menü "Einfügen"	7-45
7.7.6	Befehle im Menü "Extras"	7-47
7.7.7	Verbindungen erstellen	7-49
7.7.8	Verbindungen ändern	7-50
7.7.9	Verbindungen löschen	7-50
7.7.10	Rückkopplungen	7-51
7.7.11	Abarbeitungsreihenfolge	7-51
7.7.12	Befehle im Menü "Extras" im Untermenü "Reihenfolge"	7-52
7.7.13	CFC im Online Modus	7-56
7.8	Ablaufspracheneditor	7-57
7.8.1	Blöcke markieren	7-57
7.8.2	Befehle im Menü "Einfügen"	7-58
7.8.3	Befehle im Menü "Extras"	7-60
7.8.4	Befehle im Menü "Projekt"	7-64
7.8.5	AS-Flags	7-65
7.8.6	Ablaufsprache im Online-Modus	7-67
8	Ressourcen	8-1
8.1	Globale Variablen	8-2
8.1.1	Mehrere Variablenlisten	8-2
8.1.2	Dokumentvorlage	8-3
8.2	Codestellen-Initialisierungswerte	8-5
8.3	Parameter-Monitor	8-7
8.3.1	System-Codestellen/User-Codestellen	8-8
8.3.2	Codestellen parametrieren	8-8
8.3.3	Unterscheidung zwischen Online- und Offline-Modus	8-9

Drive PLC Developer Studio

Inhalt



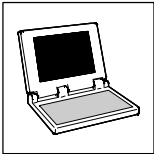
8.4	Parametermanager	8-10
8.4.2	Begriffe des Parametermanagers	8-12
8.4.3	Instanz-Parametermanager	8-13
8.4.4	Typ-Parametermanager	8-20
8.4.5	Skalierroutinen	8-20
8.5	Prozeßabbild	8-23
8.5.1	Erstellung des Prozeßabbildes	8-24
8.6	Steuerungskonfiguration	8-26
8.6.1	Arbeiten in der Steuerungskonfiguration	8-27
8.6.2	Touch-Probe Interface	8-29
8.6.3	Konfiguration eines I/O-Moduls	8-29
8.6.4	Konfiguration eines Kanals	8-30
8.7	Task-Monitor	8-31
8.8	Taskkonfiguration	8-32
8.8.1	Definition einer Task	8-32
8.8.2	Datenkonsistenz	8-33
8.8.3	Einläufige Datenverarbeitung/EVA-Prinzip	8-33
8.8.4	Wenn ein Tasküberlauf zu einem Systemfehler führt	8-34
8.8.5	Deklaration einer Task	8-34
8.8.6	Arbeiten in der Taskkonfiguration	8-35
8.9	Watch- und Rezepturverwalter	8-38
8.9.1	Watch- und Rezepturverwalter im Offline-Modus	8-38
8.9.2	Watch- und Rezepturverwalter im Online-Modus	8-39
8.9.3	Befehlsübersicht	8-39
8.10	Zielsystemeinstellungen	8-41
8.11	Bibliotheksverwaltung	8-44
8.11.1	Fenster des Bibliotheksverwalters	8-44
8.11.2	Mitgelieferte Bibliotheken	8-46
8.11.3	Benutzerdefinierte Bibliotheken	8-46
9	Visualisierung	9-1
9.1	Visualisierungselemente einfügen	9-2
9.1.1	Befehle im Menü "Einfügen"	9-2
9.2	Visualisierungselemente bearbeiten	9-5
9.2.1	Informationen in der Statusleiste	9-5
9.2.2	Ausdrücke	9-5
9.2.3	Visualisierungselemente selektieren	9-6
9.2.4	Visualisierungselemente in Größe und Form verändern	9-7
9.2.5	Visualisierungselemente verschieben	9-7
9.2.6	Visualisierungselemente kopieren, ausschneiden, einfügen	9-7
9.2.7	Visualisierungselemente löschen	9-7
9.2.8	Befehle im Menü "Extras"	9-8
9.3	Visualisierungselemente konfigurieren	9-10
9.3.1	Befehle im Menü "Extras"	9-10
9.3.2	Formatierte Textausgabe	9-14
9.4	Visualisierung in Bibliotheken	9-22



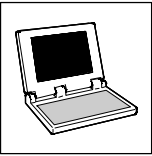
Drive PLC Developer Studio

Inhalt

10 IEC 61131-3 Datentypen	10-1
10.1 Standard-Datentypen	10-1
10.1.1 BOOL	10-1
10.1.2 Ganzzahlige Datentypen	10-1
10.1.3 REAL und LREAL	10-1
10.1.4 STRING	10-2
10.1.5 Zeitdatentypen	10-2
10.2 Definierte Datentypen	10-3
10.2.1 Array	10-3
10.2.2 Pointer	10-5
10.2.3 Aufzählungstyp	10-6
10.2.4 Strukturen	10-6
10.2.5 Referenzen	10-7
10.2.6 Unterbereichstypen	10-8
11 Übersicht der Operatoren	11-1
11.1 DDS integrierte IEC-Operatoren	11-1
11.2 Standard.lib integrierte IEC-Operatoren	11-3
12 IEC 61131-3 Operatoren	12-1
12.1 Arithmetische Operatoren	12-1
12.1.1 ADD	12-1
12.1.2 MUL	12-1
12.1.3 SUB	12-1
12.1.4 DIV	12-1
12.1.5 Bibliothek CheckDiv.lib	12-2
12.1.6 MOD	12-3
12.1.7 INDEXOF	12-3
12.1.8 SIZEOF	12-4
12.2 Bit-String-Operatoren	12-4
12.2.1 AND	12-4
12.2.2 OR	12-5
12.2.3 XOR	12-5
12.2.4 NOT	12-5
12.3 Bit-Shift-Operatoren	12-6
12.3.1 SHL	12-6
12.3.2 SHR	12-6
12.3.3 ROL	12-6
12.3.4 ROR	12-7
12.4 Auswahloperatoren	12-8
12.4.1 SEL	12-8
12.4.2 MAX	12-8
12.4.3 MIN	12-9
12.4.4 LIMIT	12-9
12.4.5 MUX	12-10
12.5 Vergleichsoperatoren	12-10
12.5.1 GT	12-10
12.5.2 LT	12-11



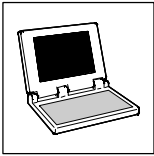
12.5.3	LE	12-11
12.5.4	GE	12-11
12.5.5	EQ	12-11
12.5.6	NE	12-11
12.6	Adreßoperatoren	12-12
12.6.1	ADR	12-12
12.6.2	Inhaltsoperator	12-12
12.7	Aufrufoperator	12-12
12.7.1	CAL	12-12
12.8	Zuweisungsoperator	12-13
12.8.1	MOVE	12-13
13	IEC 61131-3 Operanden	13-1
13.1	Konstanten	13-1
13.1.1	Zahlenkonstanten	13-1
13.1.2	BOOL-Konstanten	13-1
13.1.3	TIME-Konstanten	13-1
13.1.4	DATE-Konstanten	13-2
13.1.5	TIME_OF_DAY-Konstanten	13-2
13.1.6	DATE_AND_TIME-Konstanten	13-2
13.1.7	REAL-und-LREAL-Konstanten	13-3
13.1.8	STRING-Konstanten	13-3
13.1.9	Type Konstanten (Type Literals)	13-3
13.2	Variablen	13-4
13.2.1	Systemvariable	13-4
13.2.2	Zugriff auf Variablen von Arrays, Strukturen und Bausteinen	13-4
13.2.3	Adressierung von Bits in Variablen	13-4
13.2.4	Bezeichner	13-4
13.3	Adressen	13-5
13.3.1	Adresse	13-5
13.3.2	Merker	13-6
13.4	Funktionsaufrufe	13-7
14	IEC 61131-3 Standard-Funktionen	14-1
14.1	Typkonvertierungsfunktionen	14-2
14.1.1	Konvertierungen zwischen ganzzahligen Zahlentypen	14-2
14.1.2	BOOL_TO	14-2
14.1.3	TO_BOOL	14-3
14.1.4	TIME_TO / TIME_OF_DAY	14-3
14.1.5	DATE_TO / DT_TO	14-4
14.1.6	STRING_TO	14-4
14.1.7	TRUNC	14-4
14.2	Numerische Funktionen	14-5
14.2.1	ABS	14-5
14.2.2	SQRT	14-5
14.2.3	LN	14-5
14.2.4	LOG	14-5
14.2.5	EXP	14-5



Drive PLC Developer Studio

Inhalt

14.2.6	SIN	14-6
14.2.7	COS	14-6
14.2.8	TAN	14-6
14.2.9	ASIN	14-6
14.2.10	ACOS	14-6
14.2.11	ATAN	14-7
14.2.12	EXPT	14-7
14.3	STRING-Funktionen	14-7
14.3.1	LEN	14-7
14.3.2	LEFT	14-7
14.3.3	RIGHT	14-8
14.3.4	MID	14-8
14.3.5	CONCAT	14-8
14.3.6	INSERT	14-8
14.3.7	DELETE	14-9
14.3.8	REPLACE	14-9
14.3.9	FIND	14-9
14.4	Bistabile Funktionsblöcke	14-10
14.4.1	SR	14-10
14.4.2	RS	14-10
14.4.3	SEMA	14-11
14.5	Flankenerkennung	14-12
14.5.1	R_TRIG	14-12
14.5.2	F_TRIG	14-13
14.6	Zähler	14-14
14.6.1	CTU	14-14
14.6.2	CTD	14-14
14.6.3	CTUD	14-15
14.7	Timer	14-16
14.7.1	TP	14-16
14.7.2	TON	14-17
14.7.3	TOF	14-18
15	Anhang	15-1
15.1	Kommandozeilen Befehle	15-1
15.2	Kommandodatei (Cmdfile) Befehle	15-2
15.3	IEC-Schlüsselwörter	15-5
15.4	Tastenkombinationen und Funktionstasten	15-7
15.5	Fehlermeldungen	15-10
15.5.1	Warnungen	15-10
15.5.2	Übersetzungsfehler	15-11
15.5.3	Kommunikationsfehler	15-21
15.5.4	Systemnahe Kommunikationsfehler	15-23
15.6	Glossar	15-24
16	Index	16-1



1 Vorwort und Allgemeines

1.1 Über dieses Handbuch

Dieses Handbuch enthält detaillierte Informationen zum **Drive PLC Developer Studio (DDS)**. Das Drive PLC Developer Studio ist die leistungsfähige Entwicklungsumgebung für Ihre SPS-Programme auf Lenze-IEC 61131-Systemen. Mit dem Drive PLC Developer Studio steht Ihnen ein Tool zur Verfügung, mit dem Sie die mächtigen Sprachmittel der IEC ausnutzen können. Neben den einzelnen Editoren für die sechs IEC 61131-3 Sprachen werden Sie durch Monitoring- und Debugging-Funktionen bei der Inbetriebnahme unterstützt. Dabei können Sie den Komfort nutzen, den Sie von ausgereiften Entwicklungsumgebungen höherer Programmiersprachen unter Windows gewohnt sind.

1.1.1 Verwendete Begriffe

Begriff	Im folgenden Text verwendet für
DDS	Drive PLC Developer Studio
GDC	Global Drive Control (Parametrierprogram für Lenze PLC's/Lenze Zielsysteme)
GDO	Global Drive Oscilloscope (für Servo PLC-Geräte)
SB	Systembaustein
FB	Funktionsblock
Parametercodestellen	Codestellen zur Einstellung der Funktionalität von Funktionsblöcken

1.2 Verwendete Konventionen

Dieses Handbuch verwendet folgende Konventionen zur Unterscheidung verschiedener Arten von Information:

Informationsart	Auszeichnung	Beispiel
Namen von Dialogfeldern, Eingabefeldern und Auswahllisten	<i>kursiv</i>	Das Dialogfeld <i>Optionen</i>
Schaltflächen	fett	Klicken Sie auf OK , um...
Menübefehle	fett	Mit dem Befehl Meldungen können Sie... Sind zum Ausführen einer Funktion mehrere Befehle nacheinander erforderlich, sind die einzelnen Befehle durch einen Pfeil voneinander getrennt: Wählen Sie Datei→Öffnen , um...
Tastaturbefehle	<fett>	Mit <F2> öffnen Sie die Eingabehilfe. Ist für einen Befehl eine Tastenkombination erforderlich, ist zwischen den Befehlen ein "+" gesetzt: Mit <Shift>+<ESC> können Sie...
Programmlistings	Courier	IF var1 < var2 THEN...
Schlüsselwörter	Courier fett	...beginnt mit FUNCTION und endet mit END FUNCTION .
Wichtiger Hinweis		Achtung! Verwenden Sie die Befehl Online→Reglersperre nicht für einen Nothalt/Notstop über den PC, da dieser Befehl erst verzögert beim Antriebsregler eintrifft.
Tip		TIP! Wenn Sie den Mauszeiger eine kurze Zeit über ein Symbol in der Funktionsleiste halten, wird Ihnen der entsprechende Befehl in einem "Tooltip" angezeigt.



Drive PLC Developer Studio

Vorwort und Allgemeines



2 Einführung

2.1 Die Funktionen im Überblick

Projektstrukturierung

Ein Projekt wird in einer Datei gespeichert, die den Namen des Projekts trägt.

Der erste Baustein, der in einem neuen Projekt angelegt wird, trägt automatisch den Namen `PLC_PRG`, und von hier aus können andere Bausteine (Programme, Funktionsblöcke und Funktionen) aufgerufen werden.

Das DDS unterscheidet durch den Object Organizer zwischen den verschiedenen Arten von Objekten in einem Projekt:

- Bausteine
- Datentypen
- Darstellungselemente (Visualisierung)
- Ressourcen

Der *Object Organizer* ermöglicht Ihnen einen schnellen Zugriff auf alle Objekte Ihres Projekts. Im Bereich der Ressourcen-Strukturierung erscheinen unter anderen folgende Objekte.

- Bibliotheken
- Codestellen
- Taskkonfigurationen
- usw.

Projekterstellung

Die Verwendung von Adressen, die nicht in der Steuerungskonfiguration sind, ist nicht möglich.

Zunächst sollten Sie Ihre Steuerung konfigurieren, damit Sie während der Programmierung auf die Systembausteine zugreifen können.

Anschließend können Sie die für Ihre Anwendung notwendigen Bausteine anlegen oder aus vorhandenen Projekten kopieren. Programmieren Sie die Bausteine, die Sie benötigen, in der gewünschten Sprache.

Nach Abschluß der Programmierung können Sie das Projekt übersetzen und eventuell angezeigte Fehler beseitigen.

Simulation mit dem DDS

Sind alle Fehler beseitigt, aktivieren Sie die Simulation, loggen sich in der simulierten Steuerung ein und starten das Projekt. Das DDS befindet sich nun im Onlinebetrieb.

Sie können nun das Fenster mit Ihrer Steuerungskonfiguration öffnen und Ihr Projekt auf korrekten Ablauf testen. Belegen Sie hierzu manuell die Eingänge, und beobachten Sie, daß die Ausgänge wie gewünscht gesetzt werden. Des weiteren können Sie in den Bausteinen die aktuellen Werte der lokalen Variablen beobachten. Im Watch- und Rezepturverwalter können Sie die Datensätze konfigurieren, deren Werte Sie betrachten wollen.



Hinweis!

Lenze-Funktionsblöcke werden nicht simuliert.

Simulationseinschränkungen sind allgemein vorhanden. (📖 6-37)



Drive PLC Developer Studio

Einführung

Debugging mit dem DDS

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 Ihr Programm auf logische Korrektheit überprüfen.

Sie können Programmvariable und Ein-/Ausgänge auf bestimmte Werte setzen..

Projektdokumentation mit dem DDS

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

2.2 Bestandteile eines Projekts

2.2.1 Projekt

Ein Projekt beinhaltet alle Objekte eines Steuerungsprogramms. Die Verknüpfungen zu den Bibliotheken werden in einer Datei mit dem Namen des Projekts gespeichert.

Zu einem Projekt gehören folgende Objekte, auf die über den Object Organizer zugegriffen werden kann.

- Bausteine
- Datentypen
- Visualisierungen
- Ressourcen
 - Bibliotheken
 - Codestellen

2.2.2 Baustein (POE)

Funktionen, Funktionsblöcke und Programme sind Bausteine eines Projekts. Sie werden in der IEC 61131 Programmiersprache als Programm Organisations-Einheiten (POE) bezeichnet.

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.

Das DDS unterstützt alle IEC-Standardbausteine sowie Lenze-spezifische Bausteine. Wenn Sie diese Bausteine in Ihrem Projekt benutzen wollen, müssen Sie die entsprechende Funktionsbibliothek mit Hilfe der Bibliotheksverwaltung in Ihr Projekt einbinden.

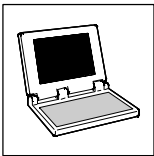
Bausteine können andere Bausteine aufrufen. Rekursionen verursachen einen Compilerfehler und müssen vermieden werden.

2.2.3 Funktion

Eine Funktion ist ein Software-Baustein, der bei der Ausführung genau ein Datenelement (das auch mehrelementig sein kann, wie z. B. Felder oder Strukturen) als Ergebnis 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.

Der Funktionsname und der Name des Ausgangs der Funktion sind gleich.



Beispiel für eine korrekte Funktionsdeklaration:

```
FUNCTION Fct: INT
```

- Eine Funktionsdeklaration beginnt mit dem Schlüsselwort **FUNCTION**.
- Der Funktion muß ein Ergebnis zugewiesen werden, d.h. der Funktionsname wird benutzt wie eine Ausgabevariable.
- Der Aufruf einer Funktion kann in ST als ein Operand in Ausdrücken vorkommen.
- Funktionen können ihre internen Zustände nicht speichern. Aufrufe einer Funktion mit denselben Eingabeparametern liefern immer denselben Wert zurück.
- Im AS können Funktionen nicht programmiert werden.

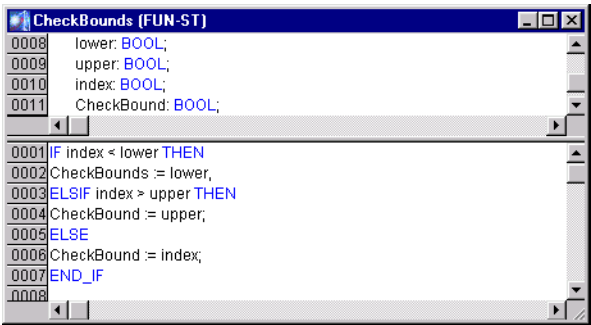
Funktion CheckBounds



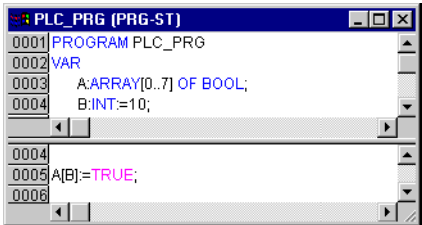
Tip!

Wenn Sie in Ihrem Projekt eine Funktion mit Namen **CheckBounds** definieren, können Sie damit Bereichsüberschreitungen bei dem Zugriff auf ein Array in Ihrem Projekt automatisch abfangen! (Siehe folgendes Beispiel).
Sehen Sie dazu auch die Checkbound-Bibliothek (CheckBounds.lib).

Der Name der Funktion ist festgelegt und darf nur diese Bezeichnung besitzen.



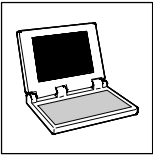
Das folgende Beispielprogramm zum Testen der **CheckBounds**-Funktion greift außerhalb der Grenzen eines definierten Arrays zu:



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 Funktion **CheckBounds** können somit Zugriffe außerhalb von Array-Grenzen korrigiert werden.

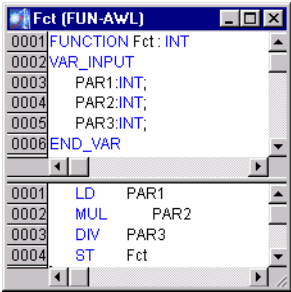
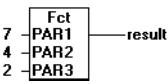
Erkennt die Funktion **CheckBounds** bei einem indirekten Zugriff auf ein Array-Element, daß sich das Array-Element außerhalb der definierten Grenzen befindet wird die Variable **g_bErrorCheck-Bounds** auf **TRUE** gesetzt. Dieser Zustand bleibt erhalten. Damit wird ein fehlerhafter Array-Zugriff innerhalb des PLC Programms erkannt.



Drive PLC Developer Studio

Einführung

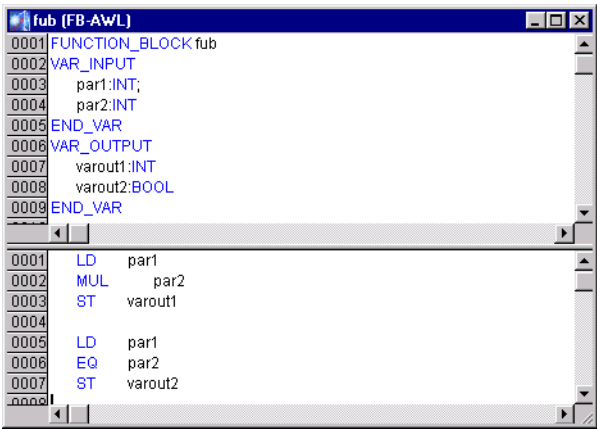
2.2.3.1 Beispiel einer Funktion

Beispiel einer Funktion in AWL:	Beispiele für den Funktionsaufruf der links abgebildeten Funktion:
	AWL <pre>LD 7 Fct 4,2 ST result</pre> ST <pre>result := Fct(7, 4, 2);</pre> FUP 

2.2.4 Funktionsblock

Ein Funktionsblock ist ein Software-Baustein, der bei der Ausführung einen oder mehrere Werte liefert.

- Ein Funktionsblock liefert keinen Rückgabewert im Gegensatz zu einer Funktion.
- Eine Funktionsblockdeklaration beginnt mit dem Schlüsselwort `FUNCTION_BLOCK`.
- Es müssen Instanzen (Datensätze) eines Funktionsblocks geschaffen werden.



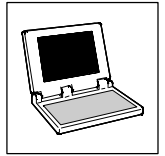
2.2.4.1 Instanzen von Funktionsblöcken

- 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.

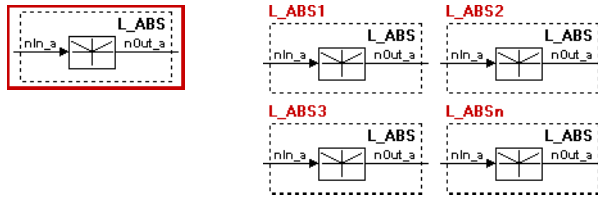
Beispiel für eine Instanz mit Namen INSTANZ des Funktionsblocks FUB

```
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, aber nicht auf dessen internen Variablen.



Beispiel anhand eines Datenmodells:

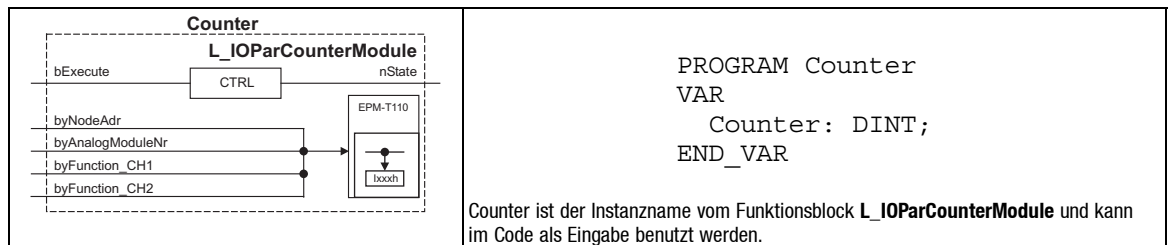


Die Instanzen L_ABS1 ... L_ABSn sind Instanzen vom Funktionsblocktyp L_ABS. Es können so viele Instanzen wie nötig instanziiert werden.

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. Eine Funktion kann keinen Funktionsblock aufrufen.
- Die Zugriff auf die Instanz eines Funktionsblocks ist auf den Baustein beschränkt, in dem sie instanziiert wurde, es sei denn, sie wurde global deklariert. Funktionsblöcke sollten niemals global deklariert werden, das führt zu logischen Fehlern.
- Der Instanzname eines Funktionsblocks kann als Eingabe einer Funktion oder eines Funktionsblocks benutzt werden.



Hinweis!

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

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

2.2.4.2 Aufruf eines Funktionsblocks

Die Eingabe- und Ausgabevariablen eines Funktionsblocks können von einem anderen Baustein angesprochen werden. Dazu muß eine Instanz des Funktionsblocks angelegt und über folgende Syntax die gewünschte Variable angegeben werden.

```
<Instanzname>.<Variablenname>
```



Drive PLC Developer Studio

Einführung

Schreiben nur auf Eingänge

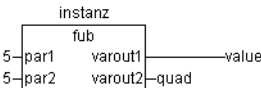
Lesen von Eingängen und Ausgängen

- Wenn man die Eingabeparameter 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 (Zuweisung geschieht durch := wie bei der Initialisierung von Variablen an der Deklarationsstelle).



Tip!

In AS können Aufrufe von Funktionsblöcken nur in Schritten vorkommen.

Deklarationsteil:	Anweisungsteil:
<pre>PROGRAM test VAR quad: BOOL; instanz: fub; value: INT:=0; END_VAR</pre>	AWL CAL instanz (par1:=5,par2:=5) LD instanz.varout2 ST quad LD instanz.varout1 ST value
	ST instanz (par1:=5,par2:=5) ; quad:=instanz.varout2; value:=instanz.varout1;
	FUP 

2.2.5 Programm

Ein Programm ist ein Baustein, der bei der Ausführung einen oder mehrere Werte liefert.

- Eine Programmdeklaration beginnt mit dem Schlüsselwort **PROGRAM**.
- Programme sind global im gesamten Projekt bekannt.
- 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.



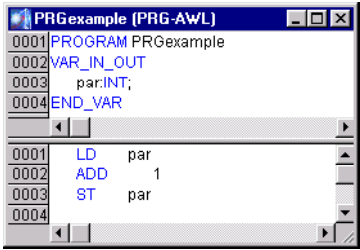
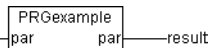
Tip!

Beim Aufruf eines Funktionsblocks werden nur die Werte in der jeweiligen Instanz eines Funktionsblocks geändert.

Diese Veränderungen sind nur wichtig, wenn dieselbe Instanz aufgerufen wird.



2.2.5.1 Beispiel eines Programms

Beispiel eines Programms in AWL:	Beispiele für den Aufruf des links abgebildeten Programms:
	AWL CAL PRGexample LD PRGexample.par ST result ST PRGexample; result:=PRGexample.var; FUP 
Beispiel für eine mögliche Aufrufsequenz von einem Hauptprogramm aus: LD 0 ST PRGexample.par (* par wird mit 0 vorbesetzt *) CAL AWLexample (* result in AWLexample = 1 *) CAL STexample (* result in STexample = 2 *) CAL FUPexample (* result in FUPexample = 3 *) • Wenn von einem Hauptprogramm aus zunächst die Variable par des Programms PRGexample mit 0 initialisiert wird, und dann nacheinander Programme mit den obigen Programmaufrufen aufgerufen werden, dann wird das Ergebnis result 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.	

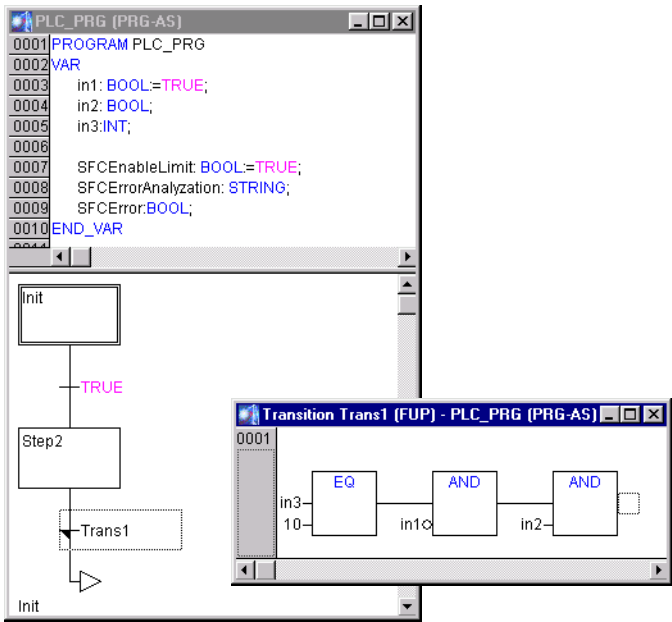


Hinweis!

Die Stringlänge wird durch das benutzte Zielsystem begrenzt.

Die Einschränkungen entstehen durch die Längenbegrenzung in den Stringroutinen. Es können nur 20 Zeichen bearbeitet werden. Der String wird dann abgeschnitten.

Das folgende Beispiel zeigt im Online-Modus die Einschränkung.





Drive PLC Developer Studio

Einführung

2.2.6 PLC_PRG

PLC_PRG ist ein spezieller vordefinierter Baustein für eine zyklische Task. Dieser Baustein wird pro Steuerungszyklus genau einmal aufgerufen.

- Wenn zum ersten Mal, nachdem ein neues Projekt erzeugt wurde, **Projekt→Objekt einfügen** ausgeführt wird, ist die Vorbelegung im Dialogfeld *Baustein* ein Baustein mit dem Namen PLC_PRG vom Typ Programm. Sie sollten diese Vorbelegung nicht ändern!



Vorsicht!

Löschen Sie den Baustein PLC_PRG nicht und benennen Sie ihn auch nicht um, wenn Sie keine Taskkonfiguration verwenden! Hängen Sie das PLC_PRG nicht zusätzlich an eine angelegte Task an. PLC_PRG wird dann mehrfach aufgerufen. Das führt zu logischen Fehlern.

PLC_PRG ist generell das Hauptprogramm in einem Single-Task-Programm.

2.2.7 System-POE's

System-POE's sind hardwareabhängige POE's (Programm-Organisations-Einheiten) mit speziellen Funktionen, die von der entsprechenden PLC (z. B. 9300 Servo PLC, Drive PLC) zur Verfügung gestellt werden. (Siehe Handbuch zur entsprechenden PLC.)

2.2.8 Ressourcen

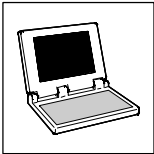
Die Ressourcen benötigen Sie zum Konfigurieren und Organisieren Ihres Projektes und zur Verfolgung von Variablenwerten:

- Globale Variablen, die im ganzen Projekt verwendet werden können
- Steuerungskonfiguration zum Konfigurieren Ihrer Hardware
- Taskkonfiguration zur Steuerung Ihres Programms über Tasks
- Taskmonitoring zur Überwachung der Tasklaufzeiten
- Watch- und Rezepturverwalter zum Anzeigen und Vorbelegen von Variablenwerten
- Zielsystemeinstellungen zur Anwahl und gegebenenfalls zur Endkonfiguration des Zielsystems.

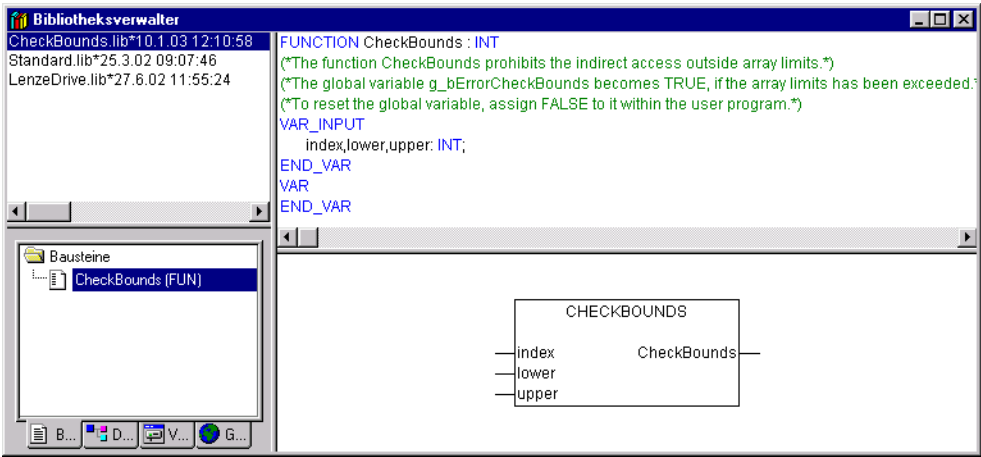
2.2.9 Bibliotheken

Sie können in Ihr Projekt mit Hilfe der Bibliotheksverwaltung eine Reihe von Bibliotheken einbinden, deren Bausteine, Datentypen und globale Variablen Sie genauso benutzen können, wie selbstdefinierte.

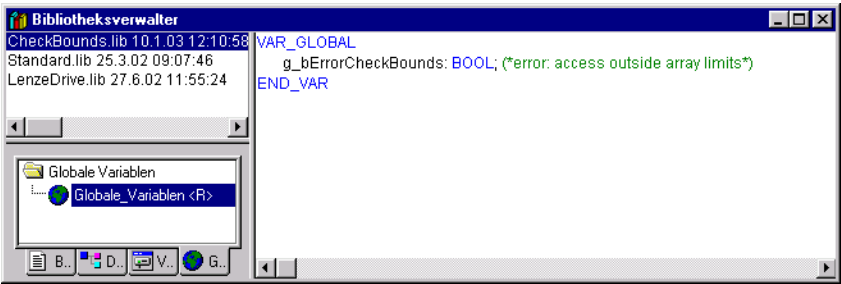
Beim Anlegen eines neuen Projektes werden einige Bibliotheken - abhängig von der ausgewählten PLC - automatisch mit eingebunden (z. B. die Bibliothek "standard.lib").



Bibliothek CheckBounds.lib



In der Registerkarte **Globale Variablen** befindet sich die Variable `g_bErrorCheckBounds`.



2.2.10 Datentypen

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

2.2.11 Visualisierung

Das DDS stellt Ihnen zum Beobachten und Verändern Ihrer Projektvariablen eine Visualisierung zur Verfügung.

Mit Hilfe der Visualisierung können Sie Offline geometrische Elemente zeichnen. Diese können dann Online, in Abhängigkeit von bestimmten Variablenwerten ihre Form/Farbe/Textausgabe verändern.



Drive PLC Developer Studio

Einführung

2.3 Debugging, Onlinefunktionalitäten

2.3.1 Debugging

Mit den Debugging-Funktionen vom DDS wird Ihnen das Auffinden von Fehlern erleichtert.

- Um debuggen zu können, muß im Dialogfeld *Optionen* in der Kategorie *Übersetzungsoptionen* das Kontrollkästchen **Debugging** aktiviert sein.



Hinweis!

Das Kontrollkästchen **Debugging** sollte nur für das Debugging markiert sein.

Breakpoint setzen, Einzelschritt oder Einzelzyklus sind nur bei aktivem Debugging möglich.

2.3.2 Breakpoint

Ein Breakpoint ist eine Stelle im Programm, an der die Abarbeitung angehalten wird.

- Mit Hilfe von Breakpoints 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, im FUP und KOP auf Netzwerknummern, im CFC auf Bausteine und in der AS auf Schritte.
- In der Implementierung des Funktionsblocks der initialisiert wurde können Breakpoints gesetzt werden. In Funktionsblockinstanzen können keine Breakpoints gesetzt werden.

2.3.3 Einzelschritt

Einzelschritt bedeutet in:

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

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

2.3.4 Einzelzyklus

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

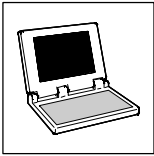


Vorsicht!

Wird ein Breakpoint gesetzt, geht bei dem Einsatz von Task des Echtzeitverhalten verloren. Eine Task mit einem Zyklus von 1ms wird nicht mehr jede Millisekunde gestartet.

Sind Breakpoints gesetzt, werden alle Task nacheinander nach Abarbeitung des Hauptprogramms PLC_PRG gestartet. Ereignisgesteuerte Task werden nur bei gültigen Starterereignis gestartet.

Dieses Verhalten hat u.a. Einfluß auf die Funktionsfähigkeit des erstellten Gesamtprojekts.



2.3.5 Werte Online verändern

Variablen können im laufenden Betrieb einmalig auf einen bestimmten Wert, nachdem das Kommando Werte schreiben zur Steuerung geschickt wurde gesetzt werden. Sie können den Wert einer Variablen durch doppelklicken auch Online verändern. Boolesche Variablen wechseln dadurch von **TRUE** auf **FALSE** bzw. umgekehrt. Für die anderen Variablen erhalten Sie einen Dialogfeld *Variable xy schreiben*. Hier können Sie den Variablenwert editieren.

2.3.6 Monitoring

Im Online Modus werden für alle am Bildschirm sichtbaren Variablen die aktuellen Werte aus der Steuerung gelesen und dargestellt. Diese Darstellung finden sie im Deklarations- und Programmeditor.

Im Watch- und Rezepturverwalter und in einer Visualisierung können aktuelle Variablenwerte ausgegeben werden.

Sollen Variablen aus den Funktionsblock-Instanzen dargestellt und überwacht werden, müssen Sie die entsprechende Instanz öffnen.

Monitoring von VAR_IN_OUT Variablen

Im Deklarationsteil und Programmteil wird beim Monitoring von **VAR_IN_OUT** der dereferenzierte Wert ausgegeben.

Monitoring von Pointern



Hinweis!

Das Monitoring der dereferenzierten Werte von Pointern wird nicht von allen Lenze Zielsystemen unterstützt.

Im Online Betrieb hängt es vom Zielsystem ab, welche dereferenzierten Werte von Pointern (Pointervariable[^]) dargestellt werden.

Bei einigen Lenze Zielsystemen wird der Pointerwert selbst angezeigt.

Im Deklarationsteil wird beim Monitoring der Pointer und der dereferenzierte Wert ausgegeben. Im Programmteil wird nur der Pointer ausgegeben.

Pointervariable= <Pointerwert>

Pt= <Wert>

In den Implementierungen wird der Wert des Pointers angezeigt. Für Dereferenzierungen wird jedoch der dereferenzierte Wert angezeigt.

Monitoring von ARRAY-Komponenten

Angezeigt werden:

- Array-Komponenten, die über eine Konstante indiziert sind. `anarray[1] = 5`

Nicht angezeigt werden:

- Array-Komponenten mit erweitertem Index. `anarray[i+j] = 5` oder `anarray[i+1] = 5`



Drive PLC Developer Studio

Einführung

2.3.7 Simulation

Das erzeugte Steuerungsprogramm wird bei der Simulation auf dem Rechner zusammen mit DDS abgearbeitet. Dabei stehen alle Onlinefunktionen zu Verfügung. Die logische Korrektheit des Programms kann ohne Steuerungshardware aber nur eingeschränkt getestet werden. (📖 6-37)

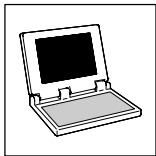


Hinweis!

Sollen bei der Simulation alle Tasks berücksichtigt werden, müssen die Intervallzeiten entsprechend angepaßt werden.

2.3.8 Logbuch

Das Logbuch zeichnet Benutzeraktionen, interne Vorgänge, Statusänderungen und Ausnahmezustände im Online-Modus chronologisch auf. Es dient der Überwachung und der Fehlerrückverfolgung. (📖 6-43)



3 Programmbeispiel “Ampelanlage”

3.1 Einleitung

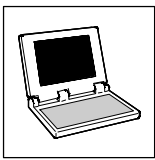
Um Ihnen den Einstieg in das DDS zu erleichtern, gehen wir in diesem Kapitel daran, ein kleines Beispielprogramm zu schreiben.

Es soll eine Mini-Ampelanlage programmiert werden, die zwei Autoampeln an einer Kreuzung steuern soll.

- Beide Ampeln werden sich in ihren rot/grün-Phasen abwechseln.
- Um Unfälle zu vermeiden, werden wir zwischen den Phasen auch noch gelb bzw. gelb/rot-Umschaltphasen vorsehen, wobei die gelb/rot-Umschaltphase kürzer ist als die gelb-Umschaltphase.

Anhand dieses Beispiels lernen Sie,

- wie sich zeitabhängige Programme mit den Sprachmitteln der IEC 61131-3 darstellen lassen,
- wie man mit Hilfe des DDS die verschiedenen Sprachen der Norm editiert,
- wie man die verschiedenen Sprachen miteinander verbindet,
- und wie man ein Programm im DDS simuliert und auf dem Bildschirm visualisiert.



Drive PLC Developer Studio

3.2 Programmierung

3.2.1 DDS starten

1. Wählen Sie im **Startmenü** von Windows im Untermenü **Programme→Lenze→Drive PLC Developer Studio** den Eintrag **Drive PLC Developer Studio**, um das DDS zu starten.



Tip!

Ist unter **Projekt→Optionen** in der Kategorie *Laden & Speichern* das Kontrollkästchen **Automatisch laden** ausgewählt, wird beim Starten des DDS automatisch das zuletzt bearbeitete Projekt geöffnet.

3.2.2 Neues Projekt anlegen

2. Wählen Sie **Datei→Neu**, um ein neues Projekt anzulegen.

3.2.3 PLC auswählen

3. Wählen Sie im Dialogfeld *Zielsystem Einstellungen* aus dem Kombinationsfeld **Konfiguration** eine PLC aus (z. B. die **9300 Servo PLC V1.x**) und bestätigen Sie mit **OK**:



3.2.4 Bausteine erzeugen

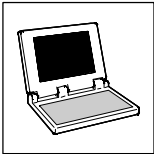
4. Im erscheinenden Dialogfeld *Neuer Baustein* ist der Name des ersten Bausteins mit **PLC_PRG** bereits vorbelegt, behalten Sie diesen Namen und den Typ des Bausteins (Programm) bei.



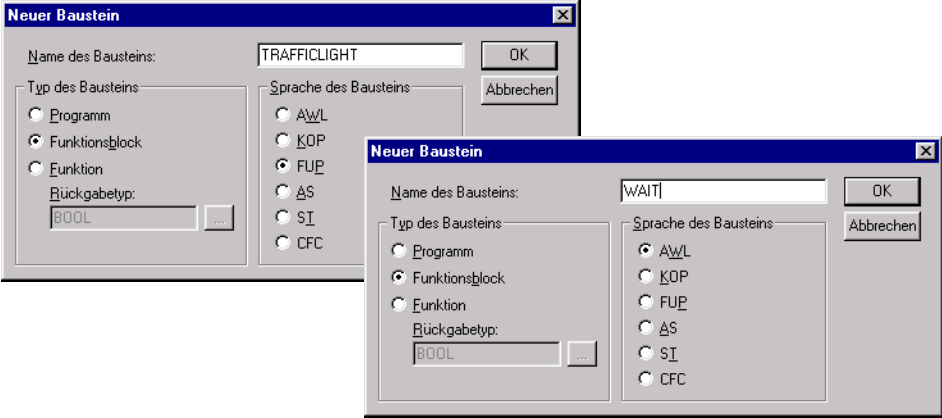
Tip!

Nur der Baustein mit dem Namen **PLC_PRG** vom Typ "Programm" wird in der Zyklischen Task abgearbeitet. Die Zyklische Task braucht nicht explizit angelegt werden.

5. Wählen Sie als Sprache für diesen Baustein **Ablaufsprache (AS)** und bestätigen Sie mit **OK**.



6. Erzeugen Sie nun zwei weitere Objekte im *Object Organizer* in der Registerkarte *Bausteine* mit **Projekt→Objekt→Einfügen**:
- **TRAFFICLIGHT** vom Typ **Funktionsblock** in der Sprache **Funktionsplan (FUP)**
 - **WAIT** vom Typ **Funktionsblock** in der Sprache **Anweisungsliste (AWL)**



TRAFFICLIGHT

Im Baustein **TRAFFICLIGHT** 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.

WAIT

Im Baustein **WAIT** 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

Der Baustein **PLC_PRG** verbindet die Bausteine miteinander, so daß das richtige Ampellicht zur richtigen Zeit und mit der gewünschten Dauer leuchten wird. Das gesamte Projekt wird dadurch in der zyklischen Task abgearbeitet.

3.2.5 Der Baustein TRAFFICLIGHT

7. Aktivieren Sie zum Bearbeiten das Editor-Fenster des Bausteins **TRAFFICLIGHT**, indem Sie im *Object Organizer* in der Registerkarte *Bausteine* auf den Eintrag **TRAFFICLIGHT** doppelklicken.

3.2.5.1 Deklaration

8. Deklarieren Sie im Deklarationseditor
- als Eingabevariable (zwischen den Schlüsselwörtern **VAR_INPUT** und **END_VAR**) eine Variable namens **STATE** vom Typ **INT**.
 - als Ausgabevariablen (zwischen den Schlüsselwörtern **VAR_OUTPUT** und **END_VAR**) die Variablen **RED**, **YELLOW**, **GREEN** und **OFF** vom Typ **BOOL**.

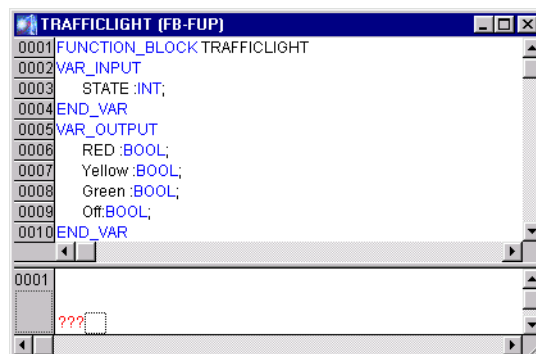
Über den Zustand der Variable **STATE** werden die Ausgabevariablen für die entsprechende Ampelfarbe geschaltet:

Ampelphase	Eingabevariable STATE	Ausgabevariablen			
		RED	YELLOW	GREEN	OFF
grün	1	FALSE	FALSE	TRUE	FALSE
gelb	2	FALSE	TRUE	FALSE	FALSE
rot	3	TRUE	FALSE	FALSE	FALSE
gelb/rot	4	TRUE	TRUE	FALSE	FALSE
aus	5	FALSE	FALSE	FALSE	TRUE



Drive PLC Developer Studio

Der Deklarationsteil des Funktionsblocks **TRAFFICLIGHT** sieht nun folgendermaßen aus:



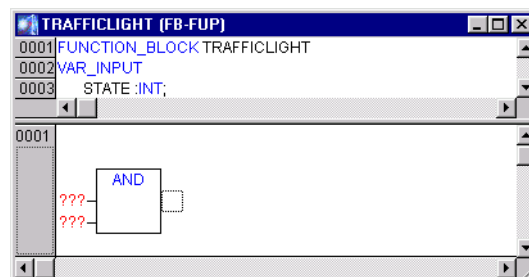
3.2.5.2 Funktionsplan

Nun gilt es, aus der Eingabevariable **STATE** des Bausteins die Werte der Ausgabevariablen zu ermitteln.

9. Klicken Sie in der unteren Fensterhälfte des Editorfensters für den Baustein **TRAFFICLIGHT** in das Feld links neben dem ersten Netzwerk (das graue Feld mit der Nummer 1), um das Netzwerk zu selektieren.

10. Wählen Sie **Einfügen→Baustein**.

Es wird im ersten Netzwerk eine Box mit dem Operator **AND** und zwei Eingängen eingefügt:

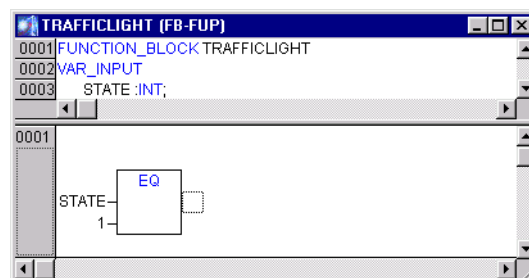


11. Klicken Sie auf den Text "AND" mit dem Mauszeiger und ändern Sie den Text in "EQ".

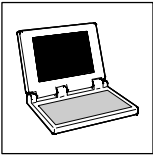
12. Selektieren Sie die Text "???" vom oberen der beiden Eingänge und tragen Sie die Variable **STATE** ein.

13. Selektieren Sie die unteren drei Fragezeichen und benennen den Eingang mit 1.

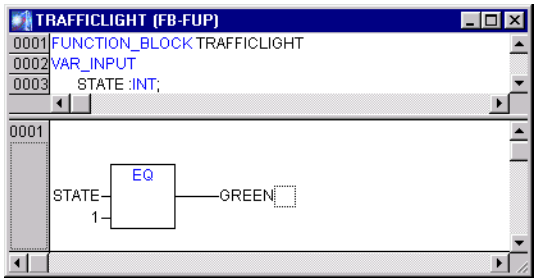
14. Klicken Sie an eine Stelle hinter der EQ-Box, um den Ausgang zu selektieren.



15. Wählen Sie **Einfügen→Zuweisung**.



16. Ändern Sie den Text “???” am Ausgang in “GREEN”.



- STATE wird mit 1 verglichen, ist das Ergebnis wahr (TRUE) wird GREEN zugewiesen. Dieses Netzwerk schaltet also die Ampel auf grün, wenn der vorgegebene Statuswert 1 ist.

Für die anderen Ampelfarben RED, YELLOW bzw. für OFF benötigen wir drei weitere Netzwerke.

17. Wählen Sie **Einfügen→Netzwerk (danach)**, um ein neues Netzwerk zu erzeugen.

18. Wählen Sie **Einfügen→Baustein**.

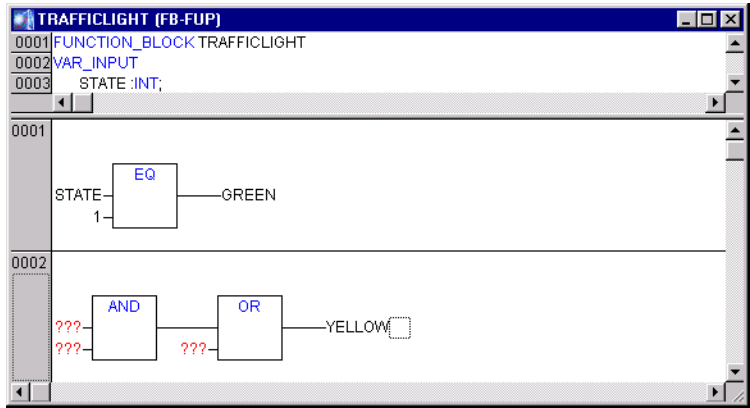
19. Klicken Sie auf den Text “AND” mit dem Mauszeiger und ändern Sie den Text in “OR”.

20. Klicken Sie an eine Stelle hinter der OR-Box, um den Ausgang zu selektieren.

21. Wählen Sie **Einfügen→Zuweisung**.

22. Ändern Sie den Text “???” am Ausgang in “YELLOW”.

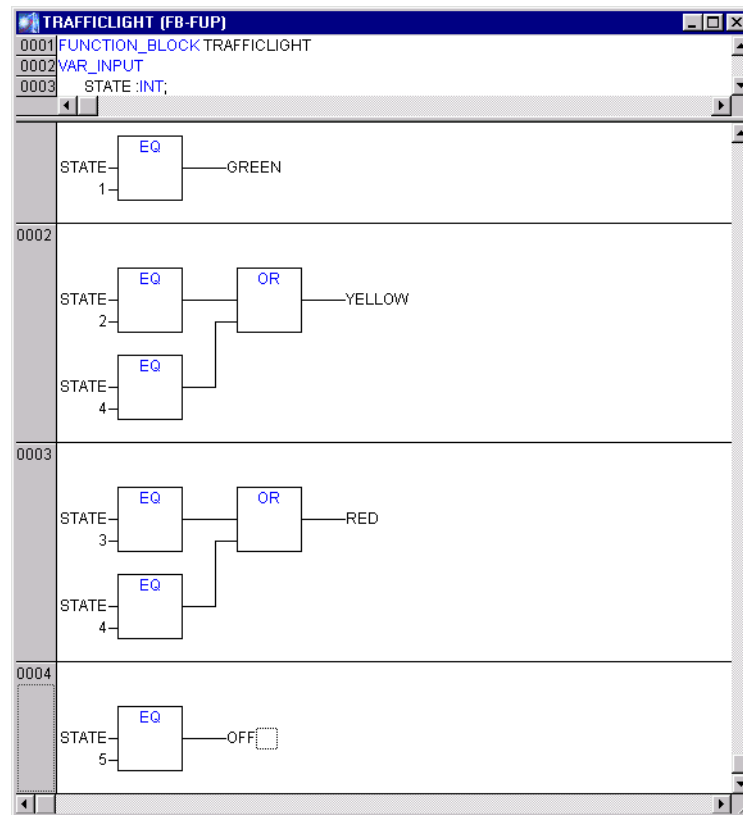
23. Selektieren Sie die den oberen Eingang der OR-Box und wählen Sie **Einfügen→Baustein**, um einen weiteren Operator vor dem selektierten Eingang einzufügen.





Drive PLC Developer Studio

24. Verwenden Sie die zuvor gezeigten Verfahren und Befehle, um den Baustein wie nachfolgend gezeigt zu vervollständigen:



Der erste Baustein ist fertig. **TRAFFICLIGHT** steuert je nach Eingabe des Wertes **STATE** die entsprechenden Ampelfarben an.

3.2.6 Der Baustein WAIT

Der Baustein **WAIT** soll ein Timer werden, mit dem wir die Länge jeder Ampelphase angeben können.

25. Aktivieren Sie zum Bearbeiten das Editor-Fenster des Bausteins **WAIT**, indem Sie im *Object Organizer* in der Registerkarte *Bausteine* auf den Eintrag **WAIT** doppelklicken.

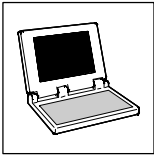
3.2.6.1 Deklaration

26. Deklarieren Sie im Deklarationseditor

- als Eingabevariable (zwischen den Schlüsselwörtern **VAR_INPUT** und **END_VAR**) eine Variable namens **SETTIME** vom Typ **TIME**.
- als Ausgabevariable (zwischen den Schlüsselwörtern **VAR_OUTPUT** und **END_VAR**) eine Variable namens **OK** vom Typ **BOOL**.

27. Belegen Sie die Ausgabevariable **OK** mit **FALSE** vor, indem Sie an das Ende der Deklaration (aber vor dem Semikolon) **:=FALSE** einfügen.

Die Ausgabevariable **OK** soll den Wert **TRUE** ausgeben, sobald die mit **SETTIME** angegebene Zeit abgelaufen ist. Für diese Funktion verwenden wir den Baustein **TP**, einen Pulsgeber.



Pulsgeber TP

Der Pulsgeber **TP** hat zwei Eingänge (IN, PT) und zwei Ausgänge (Q, ET).

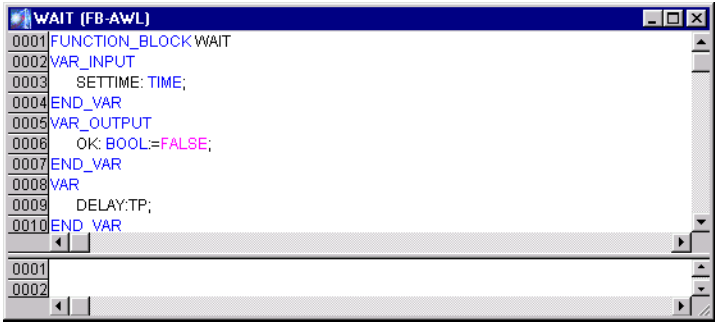
- Wenn am Eingang **IN** **TRUE** anliegt, liefert der Ausgang **Q** für die Zeit **PT** (in Millisekunden) den Wert **TRUE**.
- **ET** gibt die abgelaufene Zeit in Millisekunden aus.

Eingangsvariable	Zustand	Ausgangsvariablen	Zustand/Wert
IN	FALSE	Q	FALSE
		ET	0
IN (ET < PT)		Q	TRUE
		ET	0 ➔ PT*
IN (ET = PT)	TRUE	Q	FALSE
		ET	PT

* im Ausgang ET wird die Zeit in Millisekunden hochgezählt.

Um den Pulsgeber **TP** im Baustein **WAIT** verwenden zu können, müssen wir von **TP** eine lokale Instanz anlegen:

28. Deklarieren Sie im Deklarationseditor als lokale Variable (zwischen den Schlüsselwörtern **VAR** und **END_VAR**) eine Variable namens **DELAY** vom Typ **TP**.



3.2.6.2 Anweisungsliste

Um den gewünschten Timer zu realisieren, muß die Anweisungsliste für den Baustein **WAIT** wie folgt aussehen:



Drive PLC Developer Studio

Arbeitsablauf

Zunächst wird abgefragt, ob **Q** bereits auf **TRUE** gesetzt ist (**TRUE**: der Timer läuft). [Zeile 1]

- Ist **Q TRUE**, ändern wir nichts an der Belegung von **DELAY**, sondern rufen den Funktionsblock **DELAY** ohne Eingabe auf (um zu prüfen, ob die Zeit bereits abgelaufen ist). [Zeile 10]
- Ist **Q FALSE**, setzen wir die Variable **IN** in **DELAY** auf **FALSE**, und damit gleichzeitig **ET** auf 0 und **Q** auf **FALSE**. [Zeile 4]
So sind alle Variablen auf den gewünschten Anfangszustand gesetzt.
 - Nun speichern wir die benötigte Zeit aus der Variablen **SETTIME** in der Variablen **PT** [Zeile 5/6], und rufen **DELAY** mit **IN:=TRUE** auf. [Zeile 7]
 - Im Funktionsblock **DELAY** wird nun die Variable **ET** hochgezählt, bis sie den Wert **SETTIME** erreicht, dann wird **Q** auf **FALSE** gesetzt.
- Der negierte Wert von **Q** wird nach jedem Durchlauf von **WAIT** in **OK** gespeichert. [Zeile 14/15]
- Sobald **Q FALSE** ist, liefert **OK TRUE**.

Der Timer ist hiermit fertig.

3.2.7 Das Hauptprogram PLC_PRG

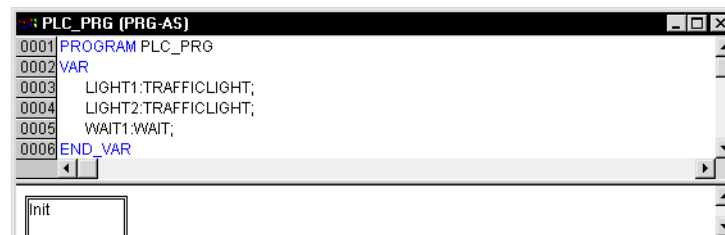
Der Baustein **PLC_PRG** ist das Hauptprogramm, von dem die beiden Funktionsblöcke **WAIT** und **TRAFFICLIGHT** aufgerufen werden.

29. Aktivieren Sie zum Bearbeiten das Editor-Fenster des Bausteins **PLC_PRG**, indem Sie im *Object Organizer* in der Registerkarte *Bausteine* auf den Eintrag **PLC_PRG** doppelklicken.

3.2.7.1 Deklaration

Damit die zuvor erstellten Funktionsblöcke in **PLC_PRG** verwendet werden können, ist es wiederum notwendig, Instanzen dieser Funktionsblöcke zu deklarieren. Für unsere Ampelanlage benötigen wir zwei Instanzen des Funktionsblocks **TRAFFICLIGHT** (**LIGHT1**, **LIGHT2**) und eine Instanz des Funktionsblocks **WAIT** (**WAIT1**).

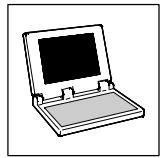
30. Deklarieren Sie im Deklarationseditor als lokale Variable (zwischen den Schlüsselwörtern **VAR** und **END_VAR**) die Variablen für die benötigten Instanzen.



3.2.7.2 Ablaufdiagramm

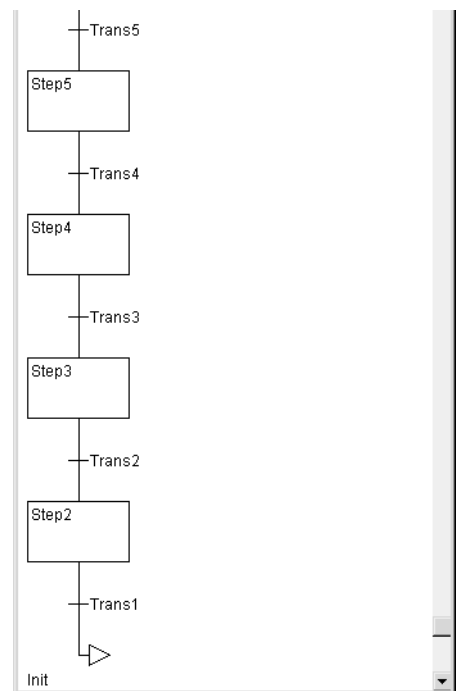
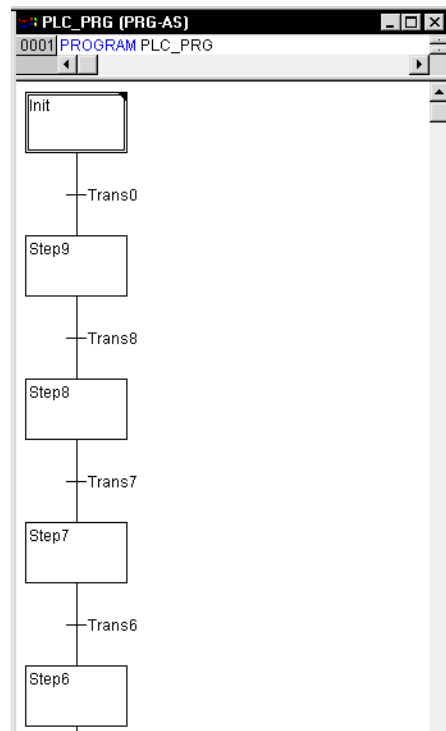
Das Anfangsdiagramm eines Bausteins in AS besteht stets aus einer Aktion **Init**, einer nachfolgenden Transition **Trans0** und einem Sprung zurück zu **Init**.





Für unsere Ampelanlage benötigen wir für jede Ampelphase einen Schritt.

31. Markieren Sie die Transition **Trans0** (auf den Querstrich links von **Trans0** klicken), so daß sie von einer gepunkteten Linie umrahmt ist.
32. Wählen Sie **Einfügen→Schritt-Transition (danach)**, um eine Schritt-Transition nach **Trans0** einzufügen.
33. Wiederholen Sie den vorherigen Schritt noch siebenmal, so daß folgender Ablaufplan vorliegt:
 - Wenn Sie einen Schritt oder Transition wieder löschen wollen, muß der Schritt und die dazugehörige Transition markiert sein. Löschen ist sonst nicht aktiv.
 - Klicken Sie zuerst den Schritt, drücken Sie **<Shift>**, klicken Sie dann auf die Transition.

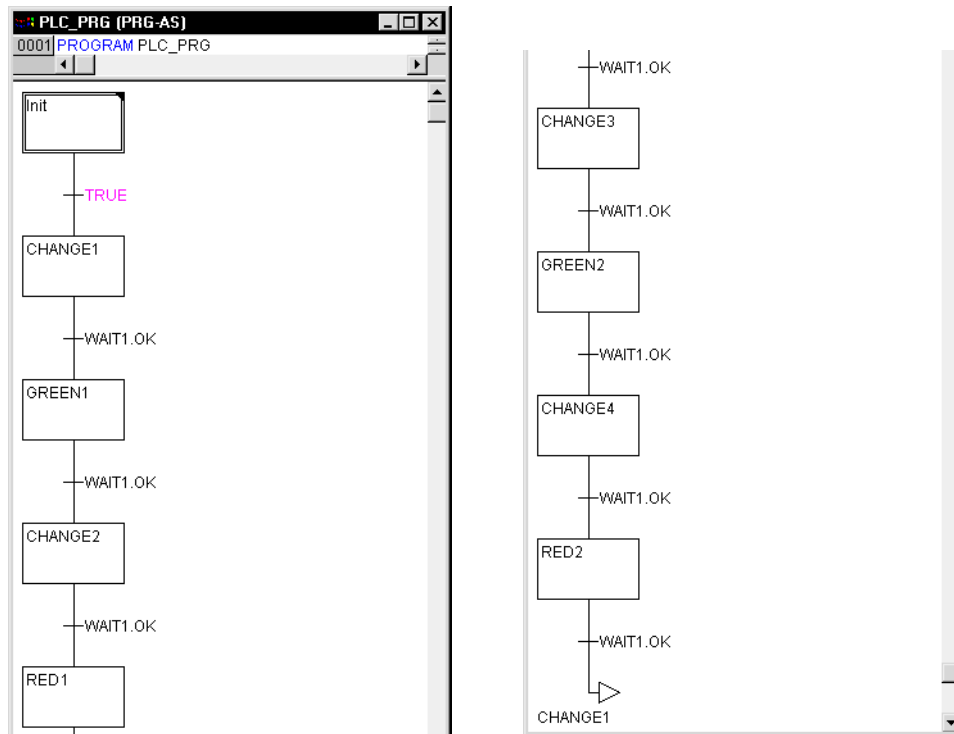


- Wenn Sie direkt auf den Namen einer Transition oder eines Schrittes klicken, dann wird der Text invertiert und Sie können ihn editieren.
34. Ändern Sie den Namen der ersten Transition nach Init in **TRUE**.
 35. Ändern Sie die Namen aller nachfolgenden Transitionen in "WAIT1.OK".
 - Die erste Transition schaltet also immer durch, alle anderen dann, wenn **WAIT1** in **OK TRUE** ausgibt, also wenn die eingegebene Zeit abgelaufen ist.
 36. Ändern Sie die Namen der Schritte folgendermaßen (von oben nach unten):
 - Init (bleibt unverändert)
 - CHANGE1
 - GREEN1
 - CHANGE2
 - RED1
 - CHANGE3
 - GREEN2
 - CHANGE4
 - RED2
 - "CHANGEx" soll jedesmal eine Gelbphase bedeuten, bei "GREEN1" wird Ampel 1 und bei "GREEN2" Ampel 2 grün sein, bei "RED1" wird Ampel 1 und bei "RED2" Ampel 2 rot sein.



Drive PLC Developer Studio

37. Ändern Sie die Rücksprungadresse (unter dem Pfeil) von "Init" nach "CHANGE1", so daß sich folgendes Ablaufdiagramm ergibt:



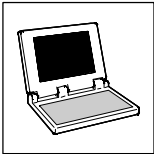
Nun müssen wir die einzelnen Schritte ausprogrammieren.

- Wenn Sie auf dem Feld eines Schrittes einen Doppelklick ausführen, öffnen Sie ein Dialogfeld zum Öffnen einer neuen Aktion.
- In unserem Beispiel werden wir als Sprache jeweils AWL (Anweisungsliste) verwenden.

38. Doppelklicken Sie auf den Schritt "Init", um das Dialogfenster *Neue Aktion* zu öffnen.



39. Wählen Sie als Sprache der Aktion "AWL" und bestätigen Sie mit **OK**.



40. Geben Sie in das Editorfenster die nachfolgenden Aktionen für den Schritt "Init" ein und legen Sie auf die gleiche Art und Weise die Aktionen für die anderen Schritte fest.

Schritt	Aktionen	Status Light1/Light2
Init	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=3)	
CHANGE1	CAL LIGHT1 (STATE:=4) CAL LIGHT2 (STATE:=3) CAL WAIT1 (SETTIME:=t#2s)	
GREEN1	CAL LIGHT1 (STATE:=1) CAL LIGHT2 (STATE:=3) CAL WAIT1 (SETTIME:=t#5s)	
CHANGE2	CAL LIGHT1 (STATE:=2) CAL LIGHT2 (STATE:=3) CAL WAIT1 (SETTIME:=t#2s)	
RED1	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=3) CAL WAIT1 (SETTIME:=t#1s)	
CHANGE3	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=4) CAL WAIT1 (SETTIME:=t#2s)	
GREEN2	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=1) CAL WAIT1 (SETTIME:=t#5s)	
CHANGE4	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=2) CAL WAIT1 (SETTIME:=t#2s)	
RED2	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=3) CAL WAIT1 (SETTIME:=t#1s)	

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

3.2.8 Erweiterung des Programms um eine Alternativverzweigung

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 COUNTER vom Typ INT.

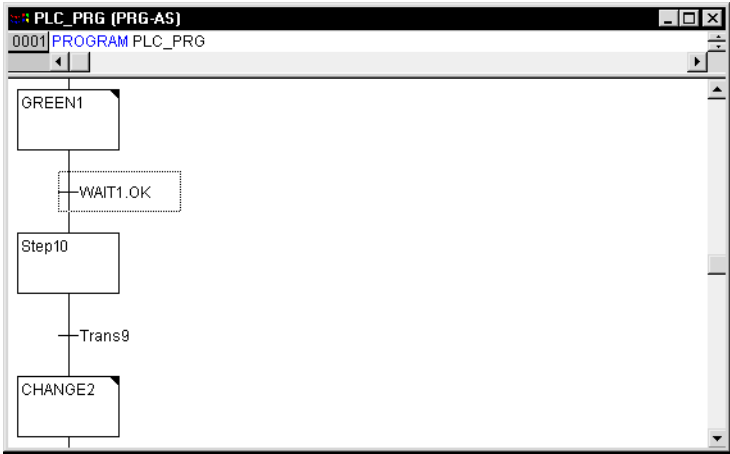
- 41. Aktivieren Sie zum Bearbeiten das Editor-Fenster des Bausteins PLC_PRG, indem Sie im Object Organizer in der Registerkarte Bausteine auf den Eintrag PLC_PRG doppelklicken.
- 42. Deklarieren Sie im Deklarationseditor als lokale Variable (zwischen den Schlüsselwörtern VAR und END_VAR) die Variable COUNTER vom Typ INT.
- 43. Initialisieren Sie die Variable COUNTER im Schritt "Init" mit 0:

Schritt	Aktionen	Status Light1/Light2
Init	CAL LIGHT1 (STATE:=3) CAL LIGHT2 (STATE:=3) LD 0 ST COUNTER	

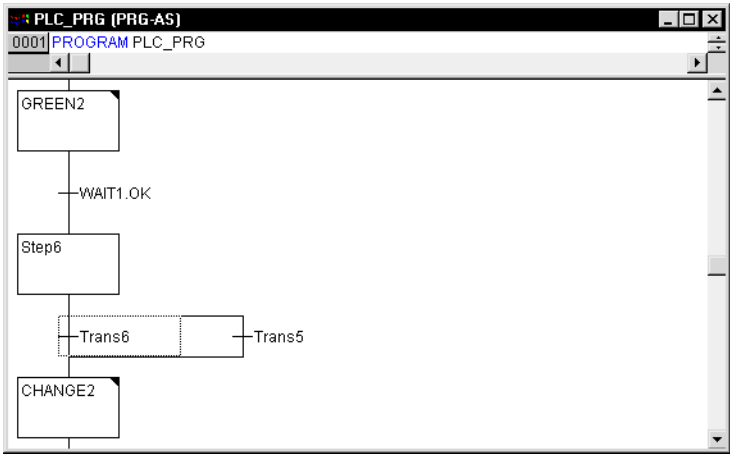


Drive PLC Developer Studio

44. Markieren Sie die Transition nach "CHANGE1" und wählen Sie **Einfügen→Schritt-Transition (danach)**, um eine Schritt-Transition einzufügen.

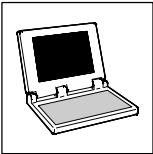


45. Markieren Sie die neu entstandene Transition und wählen Sie **Einfügen→Alternativverzweigung links**, um eine Alternativverzweigung links davon einzufügen.

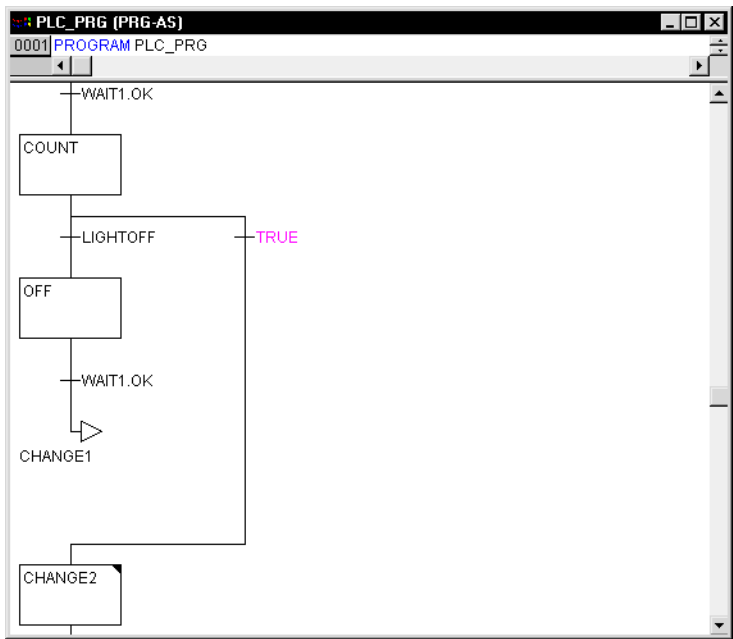


46. Markieren Sie die linke Transition und wählen Sie **Einfügen→Schritt-Transition (danach)**, um eine Schritt-Transition einzufügen.

47. Markieren Sie die neu entstandene Transition und wählen Sie **Einfügen→Sprung**, um einen Sprung einzufügen.



48. Benennen Sie die neu eingefügten Schritte/Transitionen wie nachfolgend abgebildet:



49. Legen Sie für die zwei neuen Aktionen und die neue Transitionsbedingung folgende Aktionen fest:

Schritt	Aktionen	Info
COUNT	Aktion COUNT (AWL) 0001 LD COUNTER 0002 ADD 1 0003 ST COUNTER	Counter wird um 1 erhöht.
OFF	Aktion OFF (AWL) 0001 CAL LIGHT1(STATE:=5) 0002 CAL LIGHT2(STATE:=5) 0003 LD 0 0004 ST COUNTER 0005 CAL WAIT1(SETTIME:=#10s)	Beide Ampeln werden für 10 Sekunden ausgeschaltet und der Counter wird auf 0 zurückgesetzt.

Transition	Aktionen	Info
LIGHTOFF	Transition LIGHTOFF (AWL) 0001 LD COUNTER 0002 GT 7	Transition überprüft, ob COUNTER größer als eine bestimmte Zahl ist, in diesem Beispiel 7.

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.



Tip!

Wählen Sie **Datei→Speichern unter**, um das erstellte Projekt unter einen neuen Namen zu speichern.

- Geben Sie im Dialogfeld *Datei speichern unter* in das Eingabefeld **Dateiname** einen Namen ein und schließen Sie das Dialogfeld mit **Speichern**.



Drive PLC Developer Studio

Programmbeispiel

3.3 Simulation

Testen Sie nun das fertiggestellte Programm.

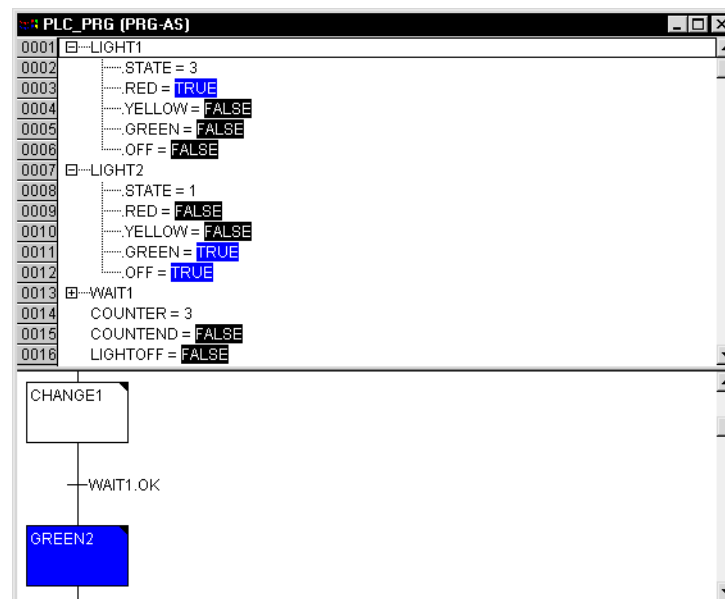
1. Wählen Sie **Projekt→Alles übersetzen**, um das Programm zu übersetzen.
2. Wählen Sie **Online→Simulation**, um den Test des Programms im Simulationsmodus durchzuführen.
3. Wählen Sie **Online→Einloggen**, um sich in die Steuerung einzuloggen.
4. Wählen Sie **Online→Start**, um das Programm auszuführen.

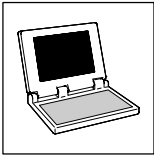
Da der Simulationsmodus aktiviert ist, wird das Programm statt in der Steuerung im DDS ausgeführt.

Monitoring

Sie können die zeitliche Abfolge der einzelnen Schritte ihres Hauptprogramms **PLC_PRG** im Editorfenster verfolgen.

- Ein gerade aktiver Schritt wird blau dargestellt.
- Durch Doppelklick auf das Pluszeichen im Deklarationseditor können Sie die Variablendarstellung aufklappen und die aktuellen Werte der einzelnen Variablen beobachten.





3.4 Visualisierung

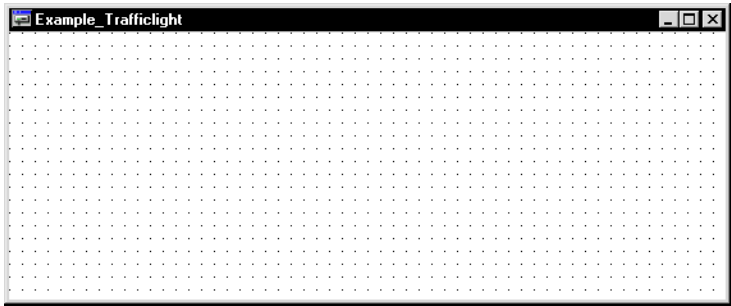
Nachdem wir die Ampelanlage im DDS programmiert und im Simulationsmodus getestet haben, können wir dessen Arbeitsweise mit Hilfe des DDS auf dem Bildschirm visualisieren.

3.4.1 Erstellen einer neuen Visualisierung

Die Visualisierung des DDS befindet sich in der Registerkarte **Visualisierung** im Object Organizer.

1. Wählen Sie im *Object Organizer* die Registerkarte **Visualisierung**.
2. Wählen Sie **Projekt→Objekt→Einfügen**, um eine neue Visualisierung zu erstellen.
3. Geben Sie in das Dialogfeld *Neue Visualisierung* einen Namen für die Visualisierung ein und bestätigen Sie die Eingabe mit **OK**.

Es öffnet sich ein Fenster, in dem Sie Ihre neue Visualisierung erstellen können:



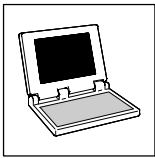
Tip!

Mit dem Befehl **Extras→Einstellungen** öffnen Sie ein Dialogfeld, in dem Sie Einstellungen für Darstellung, Rahmen und Raster der Visualisierung vornehmen können.

3.4.2 Elemente in die Visualisierung einfügen und konfigurieren

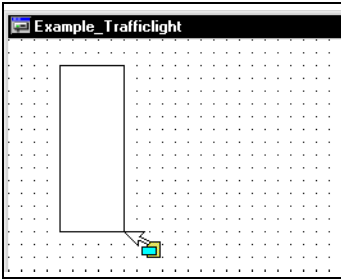
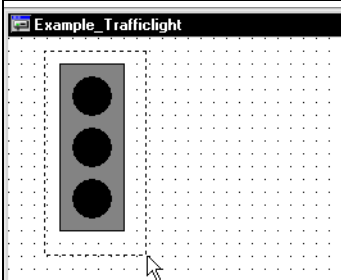
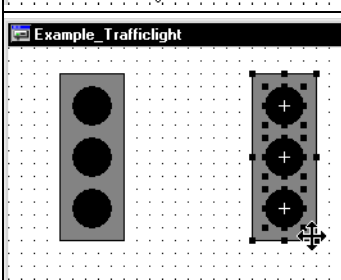
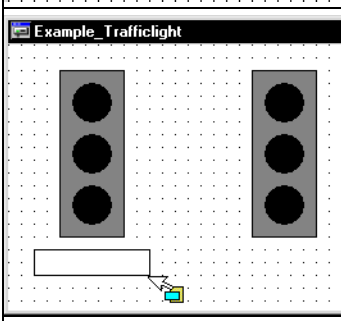
Zur Darstellung einer Ampel benötigen wir die Elemente Kreis (Ellipse) und Rechteck. Nachfolgend zeigen wir Ihnen eine mögliche Vorgehensweise:

	1. Wählen Sie Einfügen→Ellipse, um einen Kreis zu zeichnen. 2. Doppelklicken Sie auf den Kreis, um das Dialogfeld <i>Element konfigurieren</i> zu öffnen. 3. Geben Sie in der Kategorie Variablen in das Eingabefeld <i>Farbwechsel</i> die Variable <code>PLC_PRG.LIGHT1.RED</code> ein. 4. Wählen Sie in der Kategorie Farben für <i>Farbe Innen</i> einen dunklen Farbton aus. 5. Wählen Sie in der Kategorie Farben für <i>Alarmfarbe Innen</i> einen roten Farbton aus. 6. Kopieren Sie den selektierten Kreis mit <Strg>+<C> in die Zwischenablage.
	7. Fügen Sie den kopierten Kreis zweimal mit <Strg>+<V> in die Zeichnung ein und positionieren Sie die eingefügten Kreise mit gedrückter linker Maustaste wie links abgebildet. 8. Konfigurieren Sie die eingefügten Kreise per Doppelklick wie folgt: – Mittlerer Kreis: Kategorie Variablen, Eingabefeld <i>Farbwechsel</i>: <code>PLC_PRG.LIGHT1.YELLOW</code> Kategorie Farben, <i>Farbe Innen</i>: dunklen Farbton auswählen Kategorie Farben, <i>Alarmfarbe Innen</i>: gelben Farbton auswählen – Unterer Kreis: Kategorie Variablen, Eingabefeld <i>Farbwechsel</i>: <code>PLC_PRG.LIGHT1.GREEN</code> Kategorie Farben, <i>Farbe Innen</i>: dunklen Farbton auswählen Kategorie Farben, <i>Alarmfarbe Innen</i>: grünen Farbton auswählen

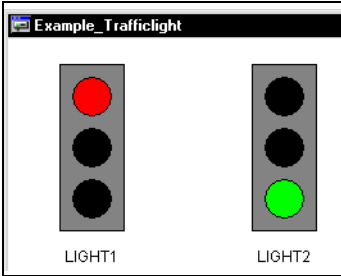


Drive PLC Developer Studio

Programmbeispiel

	9. Wählen Sie Einfügen→Rechteck, um ein Rechteck über die drei Kreise zu zeichnen. 10. Doppelklicken Sie auf das Rechteck, um das Dialogfeld <i>Element konfigurieren</i> zu öffnen. 11. Wählen Sie in der Kategorie Farben für <i>Farbe Innen</i> einen dunkelgrauen Farbton aus. 12. Wählen Sie Extras→Nach hinten legen, um das Rechteck hinter die Kreise zu legen.
	13. Ziehen Sie mit dem Mauszeiger einen Rahmen um die Elemente, um alle Elemente innerhalb des Rahmens zu selektieren. 14. Kopieren Sie die selektierten Elemente mit <Strg>+<C> in die Zwischenablage.
	15. Fügen Sie die kopierten Elemente mit <Strg>+<V> in die Zeichnung ein und positionieren Sie die eingefügten Elemente mit gedrückter linker Maustaste wie links abgebildet. 16. Konfigurieren Sie die drei Kreise der rechten Ampel per Doppelklick wie folgt: – Oberer Kreis: Kategorie Variablen, Eingabefeld <i>Farbwechsel</i>: <code>PLC_PRG.LIGHT2.RED</code> – Mittlerer Kreis: Kategorie Variablen, Eingabefeld <i>Farbwechsel</i>: <code>PLC_PRG.LIGHT2.YELLOW</code> – Unterer Kreis: Kategorie Variablen, Eingabefeld <i>Farbwechsel</i>: <code>PLC_PRG.LIGHT2.GREEN</code>
	17. Wählen Sie Einfügen→Rechteck, um ein Rechteck unter jeder Ampel zu zeichnen. 18. Konfigurieren Sie die Rechtecke per Doppelklick wie folgt: – Wählen Sie in der Kategorie Farben die Option keine Rahmenfarbe. – Geben Sie in der Kategorie Text für jede Ampel einen bezeichnenden Text ein (im Beispiel "LIGHT1" und "LIGHT2").

3.4.3 Visualisierung im Online-Modus ablaufen lassen

	Wenn Sie nun das Programm erneut starten, "leuchten" in der Visualisierung die entsprechenden Ampellichter in der festgelegten Zeitdauer auf und die korrekte Funktion der Ampelanlage kann in einfacher Weise überprüft werden.
---	---



Tip!
Eine ausführliche Beschreibung der Visualisierung finden Sie im Kapitel. (📖 9-1)



4 Programmiersprachen

4.1 Die Norm IEC 61131-3

Die Norm IEC 61131-3 ist ein internationaler Standard für Programmiersprachen von speicherprogrammierbaren Steuerungen.

Die im DDS realisierten Programmiersprachen sind konform zu den Anforderungen der Norm.

Nach diesem Standard besteht ein Programm aus folgenden Elementen:

- Strukturen
- Bausteine
- Globale Variablen
- Lokale Variablen
- Konfiguration
- usw.



Drive PLC Developer Studio

Programmiersprachen

4.2 Anweisungsliste (AWL)

Eine Anweisungsliste (AWL) besteht aus einer Folge von Anweisungen.

- Jede Anweisung beginnt in einer neuen Zeile, beinhaltet einen Operator und, je nach Art der Operation, einen oder mehrere durch Kommata abgetrennte Operanden.
- Vor einer Anweisung kann sich eine Sprung-Marke befinden, gefolgt von einem Doppelpunkt (:).
- Ein Kommentar kann zusätzlich eingetragen werden.
- Leere Zeilen können zwischen Anweisungen eingefügt werden.
- Sämtliche Befehle werden über einen Akku geführt. Dort werden Zwischenergebnisse gespeichert und das Ergebnis des letzten Befehls gespeichert.

Beispiel:

```
LD    17
ST    lint          (* Kommentar *)
GE    5
JMPC  next
LD    idword
EQ    instruct.sdword
STN   test
next:
```

4.2.1 Operatoren und Modifikatoren

In der Sprache AWL können die nachfolgend aufgeführten Operatoren und Modifikatoren verwendet werden.

Modifikatoren

- **C**: conditional 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 Akkus).



In der Tabelle sind einige Operatoren in AWL mit deren möglichen Modifikatoren und der jeweiligen Bedeutung:

Operator	Modifikatoren	Bedeutung	
LD	N (= "NOT")	Load	Zuweisung
ST	N	Store	Speichere aktuelles Ergebnis an die Operandenstelle
S		Set	Setze den Bool-Operand genau dann auf TRUE , wenn das aktuelle Ergebnis TRUE ist
R		Reset	Setze den Bool-Operand genau dann auf FALSE , wenn das aktuelle Ergebnis TRUE ist
AND	N,(		Bitweise AND
OR	N,(		Bitweise OR
XOR	N,(		Bitweise exklusives OR
ADD	(		Addition
SUB	(		Subtraktion
MUL	(		Multiplikation
DIV	(		Division
GT	(	>	Größer
GE	(	>=	Größer Gleich
EQ	(	=	Gleich
NE	(	<>	Ungleich
LE	(	<=	Kleiner Gleich
LT	(	<	Kleiner
JMP/JMPC	N	Jump	Springe zur Marke
CAL/CALC	N	Call	Rufe Programm oder Funktionsblock auf
RET/RETC	N	Return	Kehre vom Aufruf eines Funktionsblocks zurück
)			Werte zurückgestellte Operation aus

Eine umfangreichere Auflistung IEC-Operatoren finden Sie im Kapitel 12. (📖 12-1)

Beispiel

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.

Beispiel 1

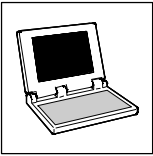
```
LD      2
MUL      2
ADD      3

ST      Result
```

Beispiel 2

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

- Im Beispiel 1 ist der Wert von Result 7.
- Im Beispiel 2 mit Klammern ergibt sich als Wert für Result 10, denn die Operation **MUL** wird erst ausgewertet, wenn man auf **)** trifft; als Operand für **MUL** errechnet sich dann 5.



Drive PLC Developer Studio

Programmiersprachen

4.3 Strukturierte 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. Eine Anweisung wird mit einem **Semikolon**; abgeschlossen.

Beispiel:

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

4.3.1 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.

4.3.2 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 (Rangfolge)
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



4.3.3 Anweisungen (Überblick)

Es gibt folgende Anweisungen in ST:

Anweisungsart	Beispiel
Zuweisung durch Zuweisungsoperator	<pre>A:=B; CV:=CV + 1; C:=SIN(X);</pre>
Aufruf eines Funktionsblocks, Benutzung der FB-Ausgabe	<pre>CMD_TMR(IN:=%IX1.0.1, PT:=T#300ms); A:=CMD_TMR.Q</pre>
RETURN	<pre>RETURN;</pre>
IF-Bedingung	<pre>D:=B*B; IF D<0.0 THEN C:=A; ELSIF D=0.0 THEN C:=B; ELSE C:=D; END_IF;</pre>
CASE-Auswahl	<pre>CASE INT1 OF 1: BOOL1:=TRUE; 2: BOOL2:=TRUE; ELSE BOOL1:=FALSE; BOOL2:=FALSE; END_CASE;</pre>
FOR-Schleife	<pre>J:=101; FOR I:=1 TO 100 BY 2 DO IF ARR[I]=70 THEN J:=I; EXIT; END_IF; END_FOR;</pre>
WHILE-Schleife	<pre>J:=1; WHILE J<=100 AND ARR[J]<>70 DO J:=J+2; END_WHILE;</pre>
REPEAT-Schleife	<pre>J:=-1; REPEAT J:=J+2; UNTIL J= 101 OR ARR[J]=70 END_REPEAT;</pre>
EXIT	<pre>EXIT;</pre>
Leere Anweisung	<pre>;</pre>

4.3.4 Zuweisungsoperator

Auf der linken Seite einer Zuweisung steht ein Operand (Variable, Adresse), dem mit dem Zuweisungsoperator := der Wert des Ausdrucks auf der rechten Seite zugewiesen wird.

Beispiel:

```
Var1 := Var2 * 10;
```

Nach Ausführung dieser Zeile hat Var1 den zehnfachen Wert von Var2.



Drive PLC Developer Studio

Programmiersprachen

4.3.5 Aufruf eines Funktionsblockes in ST

Ein Funktionsblock in ST wird aufgerufen, indem der Name der Instanz des Funktionsblocks verwendet wird, gefolgt von einer Klammer, in der den Parametern die gewünschten Werte zugewiesen werden.

Beispiel

- Ein Timer wird mit Zuweisungen für die Parameter `IN` und `PT` aufgerufen:

```
CMD_TMR(IN := %IX1.0.1, PT := T#300);
```

- Anschließend wird die Ergebnisvariable `Q` an die Variable `A` zugewiesen:

```
A:=CMD_TMR.Q
```

- Die Ergebnisvariable wird wie in AWL mit dem Namen des Funktionsblocks, einem anschließenden Punkt und dem Namen der Variablen angesprochen.

4.3.6 RETURN-Anweisung

Die **RETURN**-Anweisung wird verwendet, um die Abarbeitung des Bausteins zu beenden und in den aufrufenden Baustein zurück zuspringen. Beispielsweise abhängig von einer Bedingung.

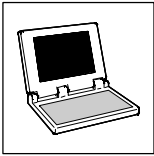
4.3.7 IF-Anweisung

Mit der **IF**-Anweisung kann eine Bedingung abgeprüft werden und abhängig von dieser Bedingung können Anweisungen ausgeführt werden.

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 Teil 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 Booleschen Ausdrücke, beginnend mit `<Boolscher_Ausdruck2>`, der Reihe nach ausgewertet, bis einer der Ausdrücke **TRUE** ergibt. Dann werden nur die Anweisungen nach diesem Booleschen Ausdruck und vor dem nächsten **ELSE** oder **ELSIF** ausgewertet.
- Wenn keiner der Booleschen Ausdrücke **TRUE** ergibt, dann werden ausschließlich die `<ELSE_Anweisungen>` ausgewertet.



Beispiel:

```
IF temp < 17 THEN
    heizung_an:=TRUE;
ELSE
    heizung_an:=FALSE;
END_IF;
```

Im Beispiel wird die Heizung eingeschaltet, wenn die Temperatur unter 17 Grad sinkt, ansonsten bleibt sie ausgeschaltet.

4.3.8 CASE-Anweisung

Mit der **CASE**-Anweisung können mehrere bedingte Anweisungen mit derselben Bedingungsvariable in ein Konstrukt zusammengefasst werden.

Syntax:

```
CASE <Var1> OF
    <Wert 1>:<Anweisung 1>
    <Wert 2>:<Anweisung 2>
    ...
    <Wert n>:<Anweisung n>
ELSE
    <ELSE-Anweisung>
END_CASE;
```

Eine **CASE**-Anweisung wird nach folgendem Schema abgearbeitet:

- Wenn die Variable in <Var1> den Wert <Wert i> hat, dann wird die Anweisung <Anweisung i> ausgeführt.
- Hat <Var 1> keinen der angegebenen Werte, dann wird die <ELSE-Anweisung> ausgeführt.
- Wenn für mehrere Werte der Variablen dieselbe Anweisung auszuführen ist, dann können diese Werte mit Kommata getrennt hintereinander geschrieben werden, und damit die gemeinsame Anweisung bedingen.
- Wenn für einen Wertebereich der Variable dieselbe Anweisung auszuführen ist, dann kann der Anfangs- und Endwert getrennt durch zwei Punkte hintereinander geschrieben werden, 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;
```



Drive PLC Developer Studio

Programmiersprachen

4.3.9 FOR-Schleife

Mit der **FOR**-Schleife können wiederholte Vorgänge programmiert werden.

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.
- Die Überprüfung erfolgt vor Ausführung der <Anweisungen>, so daß die <Anweisungen> niemals ausgeführt werden, wenn <INIT_WERT> größer als <END_WERT> ist.
- Immer, wenn der Abschnitt <Anweisungen> ausgeführt worden ist, wird <INT_Var> um <Schrittgröße> erhöht.
- Die Schrittgröße kann jeden Integerwert haben. Wird keine Schrittgröße angegeben, wird die Schrittgröße 1 verwendet.. Die Schleife muß also terminieren, da <INT_Var> nur größer wird.
- Wenn <INIT_WERT> größer <END_WERT> und <Schrittgröße> negativ ist, dann wird die FOR-Schleife rückwärts gezählt.
- Wenn <INIT_WERT> kleiner <END_WERT> und <Schrittgröße> negativ ist, dann wird die FOR-Schleife nicht ausgeführt.

Beispiel:

```
FOR Zaehler:=1 TO 5 BY 1 DO
    Var1:=Var1*2;
END_FOR;
Erg:=Var1;
```

Nehmen wir an, daß die Variable `Var1` mit dem Wert 1 vorbelegt ist, dann wird sie nach der **FOR**-Schleife den Wert 32 haben.

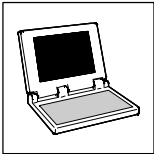


Achtung!

<END_WERT> darf nicht der Grenzwert des Zählers <INT_VAR> sein.

Wenn z. B. die Variable `Zaehler` vom Typ `SINT` ist, darf der <END_WERT> nicht 127 sein, da sonst eine Endlosschleife erfolgt.

Nach Beendigung der FOR-Schleife hat `Zaehler` den Wert 6.



4.3.10 WHILE-Schleife

Die **WHILE**-Schleife kann benutzt werden wie die **FOR**-Schleife, mit dem Unterschied, daß die Abbruchbedingung ein beliebiger boolscher Ausdruck sein kann.

Syntax:

```
WHILE <Boolscher Ausdruck> DO
    <Anweisungen>
END_WHILE;
```

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



Hinweis!

Sie müssen als Programmierer selbst dafür sorgen, daß keine Endlosschleife entsteht, indem Sie im Anweisungsteil der Schleife die Bedingung verändern, also zum Beispiel einen Zähler hoch- oder runterzählen. Ansonsten würde die Task, in der die Endlosschleife abgearbeitet wird, überlaufen.

Beispiel:

```
WHILE Zaehler<>0 DO
    Var1:=Var1*2;
    Zaehler:=Zaehler-1;
END_WHILE
```

Die **WHILE**- und die **REPEAT**-Schleife sind im gewissen 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.

Tasküberläufe sind auch bei **FOR**-Schleifen möglich. (4 4-8)



Drive PLC Developer Studio

Programmiersprachen

4.3.11 REPEAT-Schleife

Die **REPEAT**-Schleife unterscheidet sich von der **WHILE**-Schleife 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 <Boolscher Ausdruck>
END_REPEAT;
```

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

Beispiel:

```
REPEAT
    Var1:=Var1*2;
    Zaehler:=Zaehler-1;
UNTIL
    Zaehler=0
END_REPEAT;
```

4.3.12 EXIT-Anweisung

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

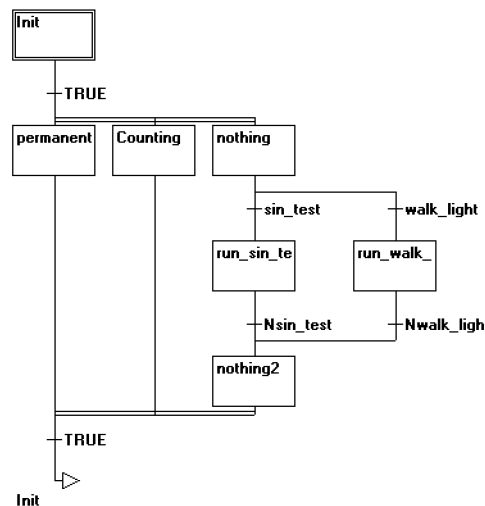
In verschachtelten Schleifen wird mit **EXIT** die innerste Schleife beendet.



4.4 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



4.4.1 Schritt

Ein in der 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, 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.

4.4.2 Aktion

Eine Aktion kann eine Folge von Instruktionen in AWL, CFC oder 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 dem Schritt verbunden.
- Um eine Aktion zu editieren, doppelklicken Sie mit dem Mauszeiger auf den Schritt, zu dem die Aktion gehört, oder markieren Sie den Schritt und führen Sie 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 den Editor geladen. Neue Aktionen können mit **Projekt→Aktion hinzufügen** erzeugt werden.



Drive PLC Developer Studio

Programmiersprachen

4.4.3 Eingangs- bzw. Ausgangsaktion

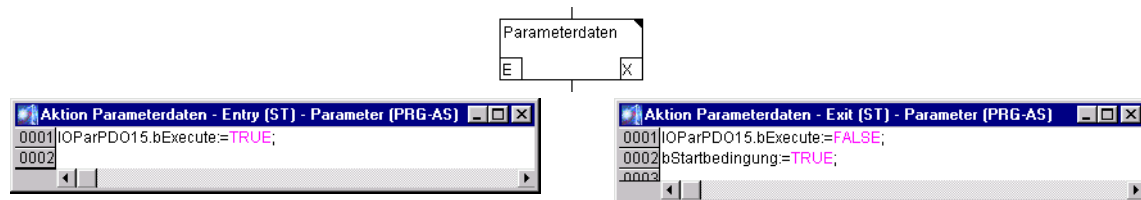
Einem Schritt kann zusätzlich zur Schritt-Aktion eine Eingangsaktion und eine Ausgangsaktion hinzugefügt werden.

- Eine Eingangsaktion wird nur einmal ausgeführt, gleich nachdem der Schritt aktiv geworden ist.
- Eine Ausgangsaktion wird nur einmal ausgeführt, bevor der Schritt deaktiviert wird.
- Die Ein- und Ausgangsaktion kann in einer beliebigen Sprache implementiert werden.

Ein Schritt mit Eingangsaktion wird durch ein 'E' in der linken unteren Ecke gekennzeichnet, die Ausgangsaktion durch ein 'X' in der rechten unteren Ecke.

- Um eine Ein- bzw. Ausgangsaktion zu editieren, doppelklicken Sie mit dem Mauszeiger auf die entsprechende Ecke im Schritt.

Beispiel eines Schrittes mit Ein- und Ausgangsaktion:



Hinweis!

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

4.4.4 Transition/Transitionsbedingung

Zwischen den Schritten liegen sogenannte Transitionen.

Eine Transitionsbedingung muß den Wert **TRUE** oder **FALSE** haben. Somit kann sie bestehen aus

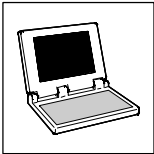
- einer Booleschen Variable
- einer dereferenzierten Adresse auf Boolescher Variable (*poTo_bTest*)
- einer Booleschen Konstante (**TRUE**)
- einer Folge von Instruktionen mit einem Booleschen Ergebnis in ST-Syntax (z.B. $(i \leq 100) \text{ AND } b$)
- einer Folge von Instruktionen programmiert in einer beliebigen Sprache

Eine Transition darf keine Programme, Funktionsblöcke oder Zuweisungen enthalten.



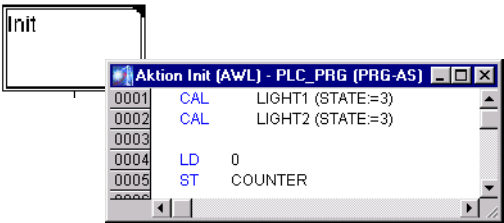
Tip!

Neben Transitionen kann auch der Tip-Modus benutzt werden, um zum nächsten Schritt weiterzuschalten. (SFCtip und SFCtipmode)



4.4.5 Aktiver Schritt

Wird der AS-Editors erstmals aufgerufen, wird immer die zum Initialschritt gehörende Aktion ausgeführt.



- Ein Schritt, dessen Aktion ausgeführt wird, heißt aktiver Schritt.
- Im Online-Modus werden aktive Schritte blau dargestellt.

In einem Steuerungszyklus werden alle Aktionen ausgeführt, die zu aktiven Schritten gehören. Wurde ein aktiver Schritt ausgeführt, dann wird die nachfolgende Transitionsbedingung überprüft. Ist die Transitionsbedingung erfüllt (**TRUE**), wird im nächsten Steuerungszyklus der darauf folgende Schritt ausgeführt.

Schritt-Flag

Zu jedem Schritt gehört ein Flag, welches den Zustand des Schritts speichert.



Hinweis!

Ein Schritt-Flag trägt den Namen des Schrittes.

Ein Schritt und ein boolsche Variable dürfen nicht den gleichen Namen tragen. Es können logische Fehler entstehen.

- Das Flag muß nicht explizit deklariert werden.
- Das Schritt-Flag (aktiver oder inaktiver Zustand des Schritts) wird dargestellt durch den logischen Wert einer boolschen Variablen `<StepName>`.
- Diese boolsche Variable hat den Wert **TRUE**, wenn der zugehörige Schritt aktiv ist, und **FALSE**, wenn er inaktiv ist. Diese Variable wird implizit deklariert, und kann in jeder Aktion und Transition des AS-Bausteins benutzt werden.

Zeitdauer-Flag

Die aktive Zeitdauer eines Schrittes kann über das Flag `_time<StepName>` abgefragt werden.

- Das Flag muß nicht explizit deklariert werden.
- Die Abfrage funktioniert nur dann, wenn für den abzufragenden Schritt in den Schrittattributen eine minimale Zeit eingestellt wurde, z. B. `t#0ms`.



Drive PLC Developer Studio

Programmiersprachen

4.4.6 IEC-Schritt

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 spezielle AS-Bibliothek `iecsfc.lib` einbinden, falls diese noch nicht vorhanden ist.

Einem IEC-Schritt können beliebig viele Aktionen zugewiesen werden.

- IEC-Aktionen sind nicht wie bei den vereinfachten Schritten als Eingangsschritt oder Ausgangsaktion fest einem Schritt zugeordnet, sondern liegen getrennt von den Schritten vor und können innerhalb ihres Bausteins mehrfach verwendet werden. Dazu müssen IEC-Aktionen mit dem Befehl **Extras→Aktion assoziieren** zu den gewünschten Schritten verknüpft werden.

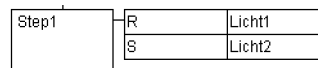
Neben Aktionen können auch boolsche Variablen den Schritten zugewiesen werden.

- 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, auch 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-Bausteins gesetzt oder zurückgesetzt, d. h. ihr wird jedesmal entweder der Wert **TRUE** oder der Wert **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 bzw. boolschen Variablennamen.

Beispiel für einen IEC-Schritt mit zwei Aktionen:



- Zum besserem Verfolgen der Vorgänge werden alle aktiven Aktionen im Onlinebetrieb wie die aktiven Schritte blau dargestellt. Nach jedem Zyklus wird überprüft, welche Aktionen aktiv sind.
- 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.

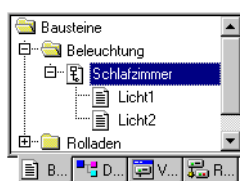


Hinweis!

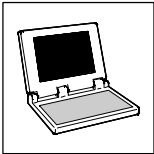
Beachten Sie hierzu auch die Einschränkung bei der Verwendung von Zeit-Qualifier bei mehrmals verwendeten Aktionen im gleichen Zyklus.

Bei einem Aufruf werden zuerst die deaktivierten Aktionen in alphabetischer Reihenfolge abgearbeitet und dann alle aktiven Aktionen wiederum in alphabetischer Reihenfolge.

Im *Object Organizer* hängen die Aktionen direkt unter ihrem jeweiligen AS-Baustein:



- Neue Aktionen können mit **Projekt→Aktion hinzufügen** erzeugt werden.



4.4.7 Qualifier

Zum Assoziieren der Aktionen zu IEC-Schritten 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, maximal solange der Schritt aktiv ist.
D	time Delayed	Die Aktion wird nach einer bestimmten Zeit aktiv, sofern der Schritt noch aktiv ist und dann solange der Schritt 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 für eine bestimmte Zeit aktiviert.

- Die Qualifier **L**, **D**, **SD**, **DS** und **SL** benötigen eine Zeitangabe im **TIME**-Konstantenformat, z.B. **LT#5s**.



Hinweis!

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**).

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

4.4.8 Implizite Variablen in der Ablaufsprache

In der Ablaufsprache gibt es implizit deklarierte Variablen, die verwendet werden können.

- Das Schritt-Flag (aktiver oder inaktiver Zustand des Schritts) heißt **<StepName>.x** bei IEC-Schritten.
- Diese boolsche Variable hat den Wert **TRUE**, wenn der zugehörige Schritt aktiv ist, und **FALSE**, wenn er inaktiv ist. Diese Variable wird implizit deklariert, und kann in jeder Aktion und Transition des AS-Bausteins benutzt werden.
- Ob eine IEC-Aktion aktiv ist oder nicht, kann man mit der Variablen **<AktionsName>.x** abfragen. Beim Deaktivierungslauf der IEC-Aktion hat diese Variable bereits den Wert **FALSE**.
- Bei den IEC-Schritten kann mit der impliziten Variable **<StepName>.t** die aktive Zeitdauer des Schrittes abgefragt werden.
Die Abfrage funktioniert nur dann, wenn für den abzufragenden Schritt in den Schrittattributen eine minimale Zeit eingestellt wurde, z. B. **t#0ms**.



Drive PLC Developer Studio

Programmiersprachen

4.4.9 Flags in der Ablaufsprache

AS-Flags dienen zum Kontrollieren von Schritten. Sie müssen global oder lokal deklariert sein. Flag-Variablen und deren Eigenschaften

SFCEnableLimit

Diese spezielle Variable ist vom Typ **BOOL**. Wenn sie **TRUE** ist, werden Zeitüberschreitungen bei den Schritten in **SFCError** registriert, sonst ignoriert. Die Verwendung kann beispielsweise bei Inbetriebnahme oder Handbetrieb nützlich sein.

SFCInit

Wenn diese Variable **TRUE** ist, wird die Ablaufsprache und die anderen Flags auf den Init-Schritt zurückgesetzt (Initialisierung). Solange die Variable **TRUE** ist, bleibt der Init-Schritt ohne Ausführung gesetzt. Ist der **SFCInit** wieder auf **FALSE** gesetzt, wird der Baustein weiterbearbeitet.

SFCReset

Diese Variable vom Typ **BOOL** verhält sich ähnlich wie die **SFCInit**. Hier wird allerdings nach der Initialisierung der Init-Schritt weiter abgearbeitet. Das kann dazu benutzt werden, um im Init-Schritt das **SFCReset**-Flag sofort wieder auf **FALSE** zu setzen.



Hinweis!

Die gleichzeitige Verwendung von **SFCInit** und **SFCReset** in einem Baustein ist nicht möglich. **SFCInit** hat dann immer Vorrang.

Beispiel-Deklaration

```
PROGRAM Flags
VAR
    SFCEnableLimit:BOOL;
    SFCError:BOOL;
    SFCErrorStep:STRING;
    SFCReset AT %IX1.0.2: BOOL;
    SFCInit AT %IX1.0.3:BOOL;
END_VAR
```

SFCQuitError

Solange diese boolsche Variable **TRUE** ist, wird die Abarbeitung des Ablaufsprachen-Diagramms angehalten. Zeitüberschreitungen in den Variablen **SFCError** werden dabei zurückgesetzt. Wird die Variable wieder auf **FALSE** gesetzt, werden alle bisherigen Zeiten in den aktiven Schritten zurückgesetzt.

SFCPause

Solange diese boolsche Variable **TRUE** ist, wird die Abarbeitung des AS-Diagramms angehalten.

SFCError

Diese boolsche Variable wird **TRUE**, wenn in einem Ablaufsprachen-Diagramm eine Zeitüberschreitung aufgetreten ist. Wird die Variable **SFCError** nicht wieder zurückgesetzt und tritt anschließend im Programm nach der ersten Zeitüberschreitung eine weitere auf, wird diese nicht mehr registriert.

SFCTrans

Diese boolsche Variable wird **TRUE**, wenn eine Transition schaltet.

SFCErrorStep

Diese Variable ist vom Typ **STRING**. Wird durch **SFCError** eine Zeitüberschreitung registriert, speichert diese Variable den Namen des verursachenden Schritts.

SFCErrorPOU

Diese Variable vom Typ **STRING** enthält nach einer Zeitüberschreitung den Namen des Bausteins, in dem die Überschreitung statt fand.



SFCCurrentStep

Diese Variable ist vom Typ **STRING**. In dieser Variablen wird der Name des aktiven Schrittes unabhängig von der Zeitüberwachung gespeichert. Bei der Parallelverzweigung wird der Schritt im ganz rechten Zweig gespeichert.

SFCTip, SFCTipMode

Diese Variablen vom Typ **BOOL** erlauben den Tip-Bereich 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 **TRUE** gesetzt ist, kann zusätzlich auch über die Transitionen weitergeschaltet werden.

SFCErrAnalyzation

Diese Variable ist vom Typ **STRING**. Wird durch das AS-Flag **SFCErr** eine Zeitüberschreitung registriert, gibt **SFCErrAnalyzation** die verantwortlichen Variablen bzw. Transitionsausdrücke aus.

Für diese Funktion muß die Bibliothek Analyzation.lib in das DDS-Projekt eingebunden sein.



Hinweis!

Auf einigen Zielsystemen kann die Ausgabe der Variablen bzw. Transitionsausdrücke längenbegrenzt sein.

Teilweise können die Stringroutinen nur 20 Zeichen verarbeiten.

4.4.10 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 Alternativanfangsline 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.

4.4.11 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 mit einem Sprung. Die Parallelverzweigung kann mit einer Sprungmarke versehen werden.
- 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. 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.



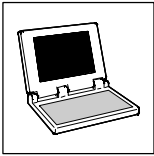
Drive PLC Developer Studio

Programmiersprachen

4.4.12 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.

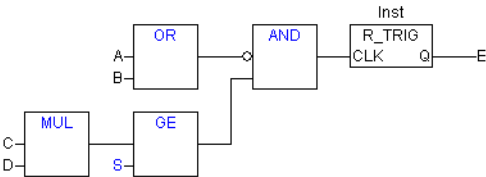


4.5 Funktionsplan (FUP)

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.

Beispiel für ein Netzwerk im Funktionsplan:





Drive PLC Developer Studio

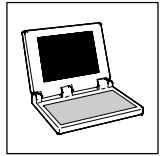
Programmiersprachen

4.6 Der Freigraphische Funktionsplaneditor (CFC)

Der Freigraphische Funktionsplaneditor arbeitet mit frei platzierbaren Elementen, die z.B. Rückkopplungen erlauben.

Ein Beispiel für ein Netzwerk im Freigraphischen Funktionsplaneditor.





4.7 Kontaktplan (KOP)

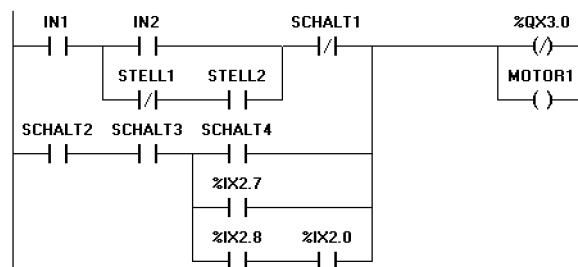
Der Kontaktplan ist eine graphisch orientierte Programmiersprache, die dem Prinzip einer elektrischen Schaltung angenähert ist.

Einerseits eignet sich der Kontaktplan dazu, logische Schaltwerke zu konstruieren, andererseits können Sie aber auch Netzwerke wie im FUP erstellen. Daher kann der Kontaktplan 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 vertikalen Stromleitung begrenzt. Dazwischen befindet sich ein Schaltplan aus Kontakten, Spulen und Verbindungslinien, die von links nach rechts den Zustand "AN" oder "AUS" (**TRUE** oder **FALSE**) weitergeben.

Beispiel für ein Netzwerk im Kontaktplan:



4.7.1 Kontakt

Jedes Netzwerk im KOP besteht auf der linken Seite aus einem Netzwerk von Kontakten, die von links nach rechts den Zustand "AN" oder "AUS" weitergeben, was den Variablenwerten **TRUE** bzw. **FALSE** entspricht.

Hat die zu einem Kontakt gehörige boolsche Variable den Wert **TRUE**, dann wird der Zustand "AN" über die Verbindungslinie von links nach rechts weitergegeben, sonst erhält die rechte Verbindung den Wert "AUS".

Parallel- bzw. Reihenschaltung

- Kontakte können parallel geschaltet sein, dann muß einer der Parallelzweige den Wert "AN" übergeben, damit die Parallelverzweigung den Wert "AN" übergibt.
- Kontakte können in Reihe geschaltet sein, dann müssen alle Kontakte den Zustand "AN" übergeben, damit der letzte Kontakt den Zustand "AN" weitergibt.

Negation

Ein Kontakt kann auch negiert sein, erkennbar am Schrägstrich im Kontaktsymbol. Er gibt dann den Eingangszustand weiter, wenn sein Zustand "AUS" (**FALSE**) ist.

4.7.2 Spule

Auf der rechten Seite eines Netzwerks im KOP befindet sich eine beliebige Anzahl sogenannter Spulen, dargestellt durch Klammern.

- Eine Spule gibt den Wert der Verbindungen von links nach rechts weiter, und kopiert ihn in eine zugehörige boolsche Variable.
- An der Eingangsline kann der Wert "AN" oder "AUS" anliegen (entsprechend den boolschen Werten **TRUE** bzw. **FALSE**).
- Spulen können nur parallel geschaltet werden.

Negation

Eine Spule kann auch negiert sein, erkennbar am Schrägstrich im Spulensymbol. Sie kopiert dann den negierten Wert in die zugehörige boolsche Variable.



Drive PLC Developer Studio

Programmiersprachen

4.7.3 Set/Reset-Spule

Eine Spule kann auch als Set oder Reset-Spule definiert sein.

- Eine Variable einer Set-Spule wird **TRUE**, wenn das Netzwerkergebnis **TRUE** ist.
- Eine Variable einer Reset-Spule wird **FALSE**, wenn das Netzwerkergebnis **FALSE** ist.
- Sind die Bedingungen nicht gegeben, bleibt die entsprechende Variable unverändert.

Eine Set-Spule (erkennbar am "S" im Spulensymbol) kann den Zustand "AN" annehmen, daraufhin aber nicht mehr den Zustand "AUS".

Eine Reset-Spule (erkennbar am "R" im Spulensymbol) kann den Zustand "AUS" annehmen, daraufhin aber nicht mehr den Zustand "AN".

4.7.4 Funktionsblöcke im Kontaktplan

Neben Kontakten und Spulen können Sie im Kontaktplan auch Funktionsblöcke und Programme eingeben.

Funktionsblöcke bzw. Programme 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.

4.7.5 KOP als FUP

Wenn Sie das Ergebnis der Schaltung im Kontaktplan zur Steuerung anderer Bausteine nutzen möchten, können Sie dies über globale Variable oder über einen direkten Aufruf tun.

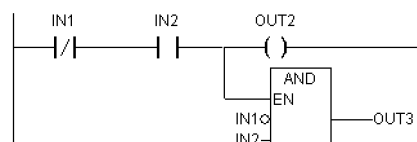
Globale Variable

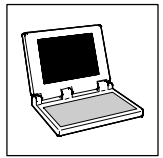
Sie können das Ergebnis der Schaltung mit Hilfe der Spulen in einer globalen Variable ablegen, die an anderer Stelle weiter benutzt wird.

Direkter Aufruf

Sie können den Aufruf direkt in Ihr KOP-Netzwerk einbauen. Dazu führen Sie einen EN-Baustein ein ("EN" steht für "Enable").

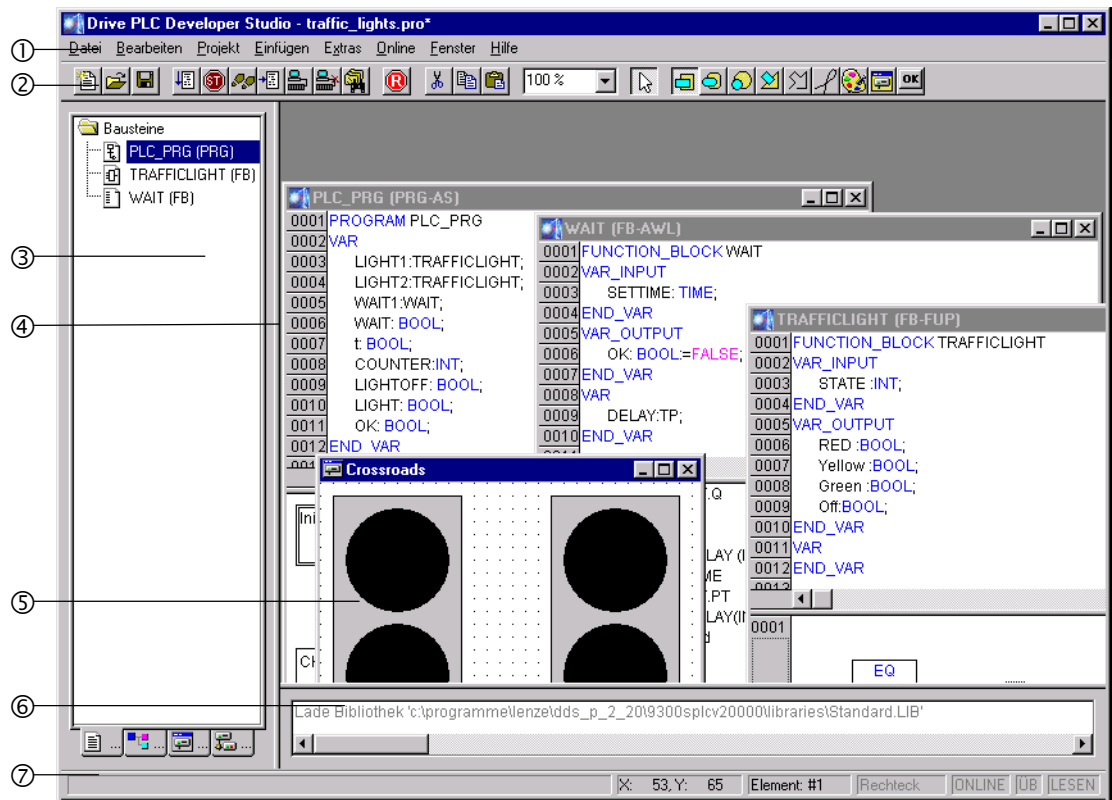
- EN-Bausteine sind ganz normale Operatoren, Funktionen, Programme oder Funktionsblöcke, die einen zusätzlichen Eingang haben, der mit EN beschriftet und vom Typ **BOOL** ist.
- Ein EN-Baustein wird erst dann ausgewertet, wenn sein EN-Eingang den Wert **TRUE** hat.
- Ein EN-Baustein wird parallel zu 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.





5 Arbeitsbereich

5.1 Bedienoberfläche



Die Bedienoberfläche des DDS setzt sich aus folgenden Elementen zusammen:

- ① Menüleiste
- ② Funktionsleiste (optional)
- ③ *Object Organizer*
- ④ Vertikaler Bildschirmteiler zwischen dem *Object Organizer* und dem Arbeitsbereich des DDS.
- ⑤ Arbeitsbereich, in dem sich die Editorfenster befinden.
- ⑥ Meldungsfenster (optional)
- ⑦ Statusleiste (optional)

5.1.1 Menüleiste

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



Drive PLC Developer Studio

Arbeitsbereich

5.1.2 Funktionsleiste

Die Funktionsleiste ermöglicht Ihnen einen schnellen Zugriff auf häufig benötigte Menübefehle.

- Klicken Sie auf ein Symbol, um den entsprechenden Befehl auszuführen.



Tip!

Wenn Sie den Mauszeiger eine kurze Zeit über einem Symbol in der Funktionsleiste halten, wird der Name des Symbols in einem Tooltip angezeigt.

Über die "Suchen"-Funktion der Online-Hilfe können sich dann ausführlichere Informationen zum entsprechenden Symbol und der damit verbundenen Funktion anzeigen lassen.

- Die Auswahl der zur Verfügung gestellten Funktionen ist abhängig vom aktiven Fenster.
- Die Anzeige der Funktionsleiste ist optional.
Projekt→Optionen Kategorie *Arbeitsbereich*

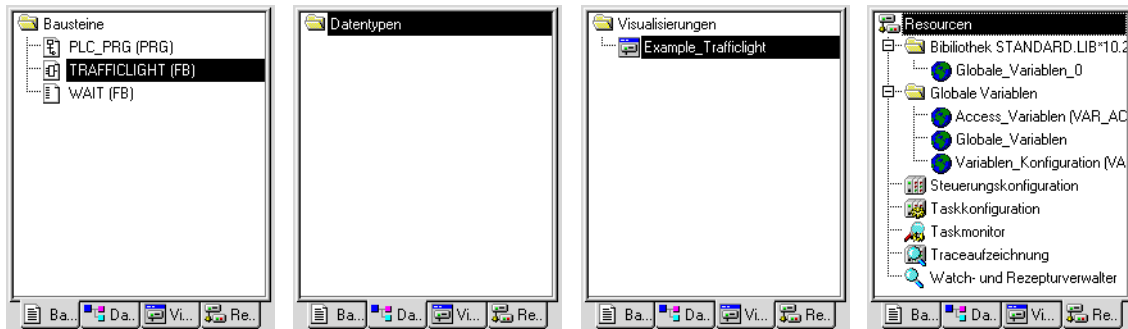
5.1.2.1 Zoom

200 %

Mit dieser Zoomfunktion in der Funktionsleiste können alle grafischen Editoren und die Visualisierung gezoomt werden. Es wird nur das aktive Fenster gezoomt.

Die Bausteine der Editoren und der Visualisierung können besser bearbeitet werden.

5.1.3 Object Organizer



Der *Object Organizer* befindet sich links im Hauptfenster des DDS. Er ermöglicht Ihnen einen schnellen Zugriff auf die vier Objektarten:

- Bausteine
- Datentypen
- Visualisierungen
- Ressourcen

Zum Wechseln zwischen den jeweiligen Objektarten klicken sie auf die entsprechende Registerkarte, die sich unter dem *Object Organizer* befindet oder benutzen Sie die linke bzw. rechte Pfeiltaste.



5.1.4 Vertikaler Bildschirmteiler

Der Bildschirmteiler ist die Grenze zwischen zwei nicht überlappenden Fenstern.

Im DDS gibt es Bildschirmteiler zwischen

- dem *Object Organizer* und dem Arbeitsbereich des Hauptfensters
- der Schnittstelle (Deklarationsteil) und der Implementierung (Anweisungsteil) von Bausteinen
- 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.



Tip!

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.

5.1.5 Arbeitsbereich

Der Arbeitsbereich befindet sich rechts im Hauptfenster des DDS. Alle Editoren für Objekte und der Bibliotheksverwaltung werden in diesem Bereich geöffnet. In der Titelseite der Fenster erscheint der jeweilige Objektname. Bausteine werden mit einem Kürzel des Bausteintyps und der verwendeten Programmiersprache gekennzeichnet.



Tip!

Unter dem Menübefehl **Fenster** im Hauptmenü befinden sich alle Befehle zur Fensterverwaltung.

5.1.6 Meldungsfenster

Das Meldungsfenster befindet sich getrennt durch einen horizontalen Bildschirmteiler unterhalb des Arbeitsbereiches im Hauptfenster.

Es enthält alle Meldungen aus dem letzten Übersetzungs-, Überprüfungs- oder Vergleichsvorgang. Auch Suchergebnisse und Querverweise können hier ausgegeben werden.

- Wenn Sie im Meldungsfenster mit der Maus einen Doppelklick auf eine Meldung ausführen oder die **<Eingabetaste>** drücken, öffnet sich der Editor mit dem entsprechenden 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.
- Sie können das Meldungsfenster mit **<Shift>+<ESC>** ein- bzw. ausblenden.



Drive PLC Developer Studio

Arbeitsbereich

5.1.7 Statusleiste

Die Statusleiste, die sich ganz unten im Hauptfenster des DDS befindet, zeigt Ihnen Informationen über das aktuelle Projekt und über Menübefehle an.

- Trifft ein Zustand zu, erscheint der zugehörige Begriff rechts in der Statusleiste in schwarzer Schrift, trifft ein Zustand nicht zu in grauer Schrift.
- Die Anzeige der Statusleiste ist optional.
Projekt→Optionen Kategorie *Arbeitsbereich*

Online-Modus

Wenn Sie im Online-Modus arbeiten, erscheint der Begriff **Online** in schwarzer Schrift, arbeiten Sie dagegen im Offline-Modus, erscheint der Begriff in grauer Schrift.

Im Online-Modus zeigt Ihnen die Statusleiste folgende Zustände an:

- **SIM**: das DDS befindet sich in der Simulation
- **LÄUFT**: das Programm wird abgearbeitet
- **BP**: ein Breakpoint ist gesetzt

Weitere Anzeigen in der Statuszeile

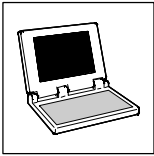
- Bei Texteditoren wird die Zeilen- und Spaltennummer der aktuellen Cursorposition angegeben. Im Überschreibmodus wird in der Statuszeile **ÜB** angezeigt.
- Befindet sich der Mauszeiger in einer Visualisierung, wird die aktuelle X- und Y-Position des Cursors in Pixel relativ zur oberen linken Ecke des Bildes angegeben.
- Befindet sich der Mauszeiger auf einem Element oder wird ein Element bearbeitet, wird die Nummer des Elements angegeben.
- Haben Sie ein Element zum Einfügen ausgewählt, so erscheint dies ebenfalls (z. B. Rechteck).
- Wenn Sie einen Menübefehl angewählt, aber noch nicht betätigt haben, dann erscheint eine kurze Beschreibung in der Statusleiste.
- Wird ein Projekt nur mit Lesezugriff geöffnet, ist das Wort **LESEN** in der Statuszeile sichtbar.

5.1.8 Kontextmenü

Kurzform: <Umschalt>+<F10>

Sie können die rechte Maustaste verwenden, um sich ein Kontextmenü anzeigen zu lassen.

- Das Kontextmenü enthält die am häufigsten verwendeten Befehle für ein markiertes Objekt oder für den aktiven Editor.
- Die Auswahl der zur Verfügung gestellten Befehle ist abhängig vom aktiven Fenster.



5.2 Fenster anordnen

Im Menü **Fenster** finden Sie alle Befehle zur Fensterverwaltung.
Dies sind sowohl Befehle zum automatischen Anordnen der Fenster, als auch zum Öffnen des Bibliotheksverwalters, zum Öffnen des Logbuchs und zum Wechseln zwischen geöffneten Fenstern.
Ganz unten im Menü **Fenster** finden Sie eine Auflistung aller geöffneten Fenster in der Reihenfolge, in der sie geöffnet wurden.

- Klicken Sie auf einen Eintrag, um zum entsprechenden Fenster zu wechseln. Das aktive Fenster ist durch einen Haken (✓) vor dem Menüeintrag gekennzeichnet.

5.2.1 Befehle im Menü “Fenster”

5.2.1.1 Nebeneinander

Symbol: -	Menübefehl: Fenster→Nebeneinander	Tastaturbefehl: -
-----------	--	-------------------

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

5.2.1.2 Untereinander

Symbol: -	Menübefehl: Fenster→Untereinander	Tastaturbefehl: -
-----------	--	-------------------

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

5.2.1.3 Überlappend

Symbol: -	Menübefehl: Fenster→Überlappend	Tastaturbefehl: -
-----------	--	-------------------

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

5.2.1.4 Symbole anordnen

Symbol: -	Menübefehl: Fenster→Symbole anordnen	Tastaturbefehl: -
-----------	---	-------------------

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

5.2.1.5 Alle Schließen

Symbol: -	Menübefehl: Fenster→Alle Schließen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl schließen Sie alle geöffneten Fenster im Arbeitsbereich.

5.2.1.6 Meldungen

Symbol: -	Menübefehl: Fenster→Meldungen	Tastaturbefehl: <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, erscheint im Menü ein Haken (✓) vor dem Befehl.

5.2.1.7 Bibliotheksverwaltung

Symbol: -	Menübefehl: Fenster→Bibliotheksverwaltung	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie das Dialogfeld *Bibliotheksverwalter*. (☐ 8-44)

5.2.1.8 Logbuch

Symbol: -	Menübefehl: Fenster→Logbuch	Tastaturbefehl: -
-----------	------------------------------------	-------------------

Mit diesem Befehl öffnen Sie das Logbuch-Fenster. Zu dem geöffneten Logbuch-Fenster ist der Menübefehl **Logbuch** vorhanden. (☐ 6-43)



Drive PLC Developer Studio

Arbeitsbereich

5.3 Grundeinstellungen

5.3.1 Optionen des DDS

Über den Menübefehl **Projekt→Optionen** im Hauptmenü können Sie die Ansicht Ihres Hauptfensters konfigurieren. Darüber hinaus können Sie noch weitere Einstellungen vornehmen, um das DDS an Ihre individuellen Bedürfnisse anzupassen.

Die Einstellungen, die Sie hierbei vornehmen, werden, soweit nicht anders vermerkt, in der ini-Datei des DDS gespeichert und beim nächsten Start wiederhergestellt.

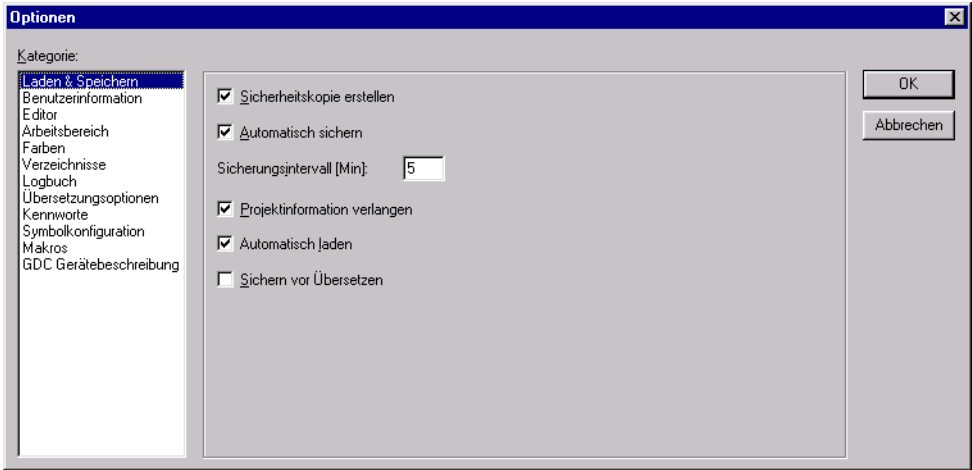
5.3.1.1 Optionen

Symbol: -	Menübefehl: Projekt→Optionen...	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie das Dialogfeld *Optionen*.

Die Optionen sind in verschiedenen Kategorien eingeteilt. Wählen Sie im Feld *Kategorie* auf der linken Seite die gewünschte Kategorie durch einen Mausklick oder mit Hilfe der Pfeiltasten aus und nehmen Sie auf der rechten Seite die gewünschten Einstellungen vor.

Optionen→Laden & Speichern



Ist eine Option aktiviert, erscheint vor der Option ein Haken (✓).

Sicherheitskopie erstellen

Ist das Kontrollkästchen **Sicherheitskopie erstellen** aktiviert, speichert das DDS die alte Datei bei jeder Sicherung in eine Sicherungsdatei mit dem Suffix ".bak". So können Sie also stets die Version vor der letzten Speicherung wiederherstellen.

Automatisch sichern

Ist das Kontrollkästchen **Automatisch sichern** aktiviert, wird Ihr Projekt, während Sie daran arbeiten, ständig nach dem von Ihnen eingestellten Zeitintervall (Sicherungsintervall) in eine temporäre Datei mit dem Präfix ".asd" gespeichert. Diese Datei wird beim normalen Beenden des Programms wieder gelöscht.

Sollte das DDS aus irgendeinem Grund nicht "normal" beendet werden (z. B. bei einem Stromausfall), wird die Datei nicht gelöscht und es erscheint beim erneuten Öffnen des Projektes eine Meldung, daß eine Sicherungsdatei existiert. Sie können nun entscheiden, ob Sie die Originaldatei oder die Sicherungsdatei öffnen wollen.

Sicherungsintervall (Min)

Eingabe eines Zeitintervalls in Minuten zum temporären Sichern.

Projektinformation verlangen

Ist das Kontrollkästchen **Projektinformation verlangen** aktiviert, wird beim Abspeichern eines neuen Projekts oder beim Abspeichern eines Projekts unter einem neuen Namen automatisch die Projektinformation aufgerufen.

Die Projektinformation können Sie mit dem Menübefehl **Projekt→Projektinformation** einsehen und bearbeiten.

Automatisch laden

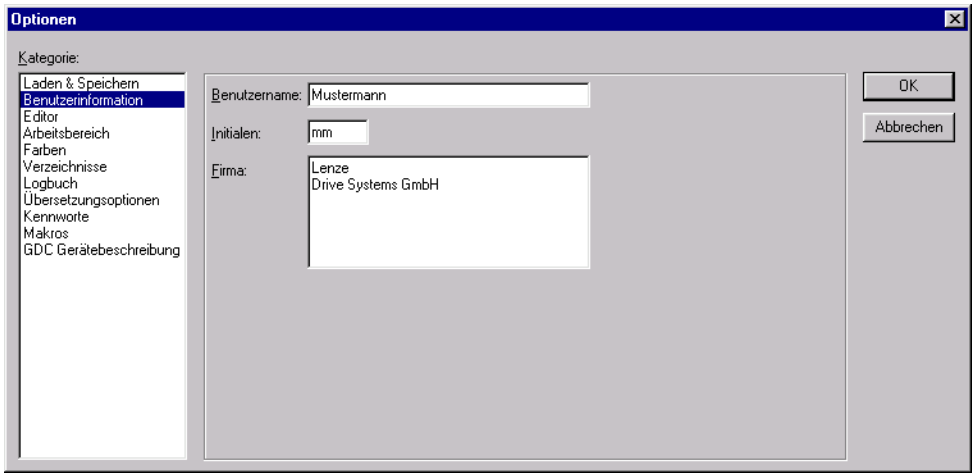
Ist die Option **Automatisch laden** aktiviert, dann wird beim nächsten Start des DDS das zuletzt geöffnete Projekt automatisch geladen.

Das Laden eines Projekts kann beim Start des DDS auch durch Angabe eines Projektes in der Befehlszeile erfolgen.

Sicherung vor dem Übersetzen

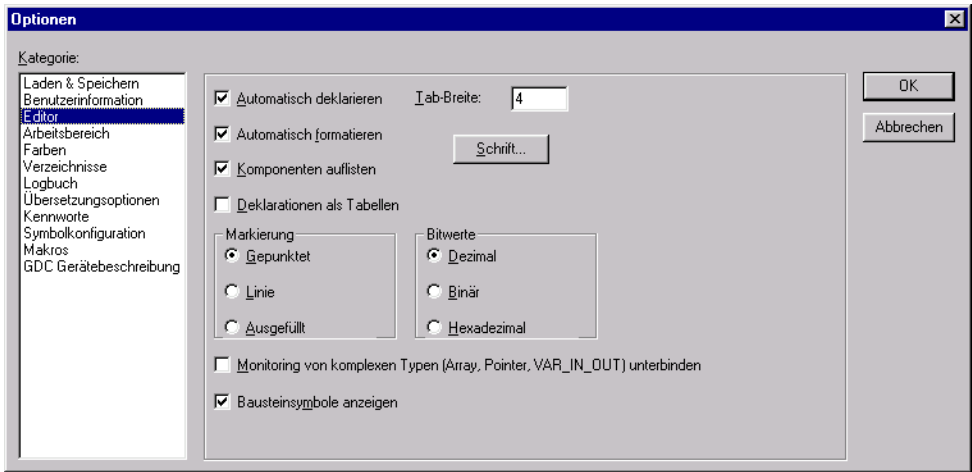
Das Projekt wird vor jedem Übersetzungslauf gesichert.

Optionen→Benutzerinformation



Jeder der Einträge kann verändert werden und wird mit dem Projekt gespeichert.

Optionen→Editor



Ist eine Option aktiviert, erscheint vor der Option ein Haken (✓).



Drive PLC Developer Studio

Arbeitsbereich

Automatisch deklarieren

Ist die Option **Automatisch deklarieren** aktiviert, erscheint in allen Editoren nach Eingabe einer noch nicht deklarierten Variable das Dialogfeld *Variablendeklaration*, über das diese Variable deklariert werden kann.

Automatisch formatieren

Ist die Option **Automatisch formatieren** aktiviert, führt das DDS 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.
- Tabulatoren werden eingefügt, so daß sich eine einheitliche Spaltenaufteilung ergibt.

Deklarationen als Tabelle

Ist die Option **Deklarationen als Tabelle** aktiviert, können Sie Variablen statt mit dem üblichen Deklarationseditor als Tabelle editieren.

PLC_PRG (PRG-AS)					
	VAR	VAR_INPUT	VAR_OUTPUT	VAR_IN_OUT	CONSTANT
	Name	Adresse	Typ	Initial	Kommentar
0001	LIGHT1		TRAFFICLIGHT		
0002	LIGHT2		TRAFFICLIGHT		
0003	WAIT1		WAIT		
0004	WAIT		BOOL		
0005	t		BOOL		
0006	COUNTER		INT		
0007	LIGHTOFF		BOOL		
0008	LIGHT		BOOL		

- Diese Tabelle ist wie ein Karteikasten geordnet, in dem Registerkarten für Eingabe-, Ausgabe-, lokale und Ein-/Ausgabevariablen 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 ist eine Tab-Breite von vier Zeichen, wobei die Zeichenbreite wiederum von der eingestellten Schriftart abhängt.

Schrift

Klicken Sie auf **Schrift**, um im Dialogfeld *Schriftart* die Schriftart für die Editoren auszuwählen.

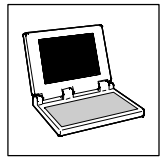
- Die Größe der Schrift ist die Grundeinheit für alle Zeichnungsoperationen. Die Wahl einer größeren Schriftgröße vergrößert somit die Ausgabe und sogar den Ausdruck bei jedem Editor des DDS.

Markierung

In dem Gruppenfeld **Markierung** können Sie zwischen drei Darstellungsformen für eine Markierung in den graphischen Editoren auswählen:

- **Gepunktet:** die Markierung wird durch ein gepunktetes Rechteck dargestellt.
- **Linie:** die Markierung wird durch ein Rechteck mit durchgezogener Linie dargestellt.
- **Ausgefüllt:** die Markierung wird durch ein ausgefülltes Rechteck (invers) angezeigt.

Aktiviert ist die Auswahl, vor der ein Punkt erscheint.



Bitwerte

In dem Gruppenfeld *Bitwerte* können Sie zwischen drei Darstellungsformen für binäre Daten (Typ **BYTE**, **WORD**, **DWORD**) beim Monitoring auswählen:

- **Dezimal**
- **Hexadezimal**
- **Binär**

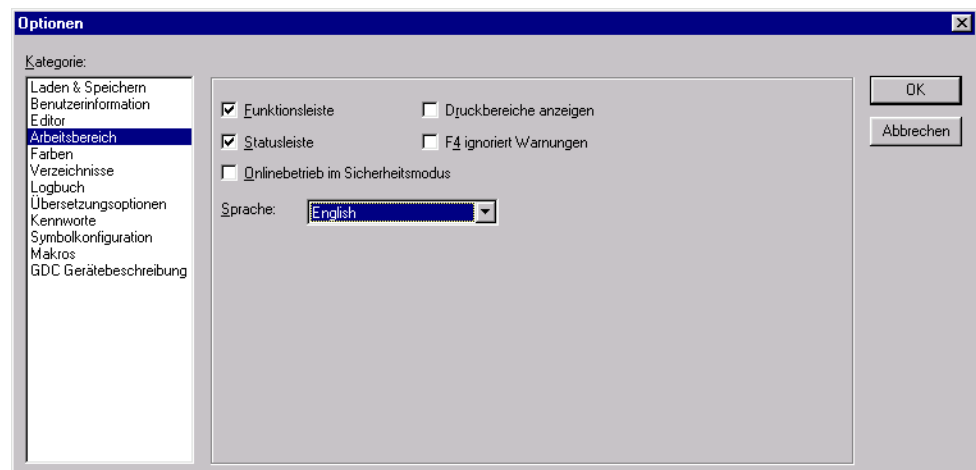
Aktiviert ist die Auswahl, vor der ein Punkt erscheint.

Monitoring von komplexen Typen

Bausteinsymbole anzeigen

Ist das Kontrollkästchen **Bausteinsymbole anzeigen** markiert, werden bei den Lenze-Bausteinen Signallaufpläne in Form von Bitmaps angezeigt.

Optionen→Arbeitsbereich



Ist eine Option aktiviert, erscheint vor der Option ein Haken (✓).

Funktionsleiste

Ist die Option **Funktionsleiste** aktiviert, wird die Funktionsleiste mit den Schaltflächen zur schnelleren Auswahl von Menübefehlen unterhalb der Menüleiste angezeigt.

Statusleiste

Ist die Option **Statusleiste** aktiviert, wird die Statusleiste am unteren Rand des DDS-Hauptfensters angezeigt.

Onlinebetrieb im Sicherheitsmodus

Ist die Option **Onlinebetrieb im Sicherheitsmodus** aktiviert, erscheint im Online-Modus bei den Befehlen **Start**, **Stop**, **Reset**, **Breakpoint an**, **Einzelzyklus** und **Werte schreiben** ein Dialogfeld mit zusätzlicher Sicherheitsabfrage, ob der Befehl wirklich ausgeführt werden soll. Diese Option wird mit dem Projekt gespeichert.

Sprache

Sie können als Sprache für die Dialoge und Menüs des DDS zwischen Deutsch und Englisch auswählen.



Hinweis!

Die Sprachauswahl ist nur unter Windows NT und Windows 2000 möglich.



Drive PLC Developer Studio

Arbeitsbereich

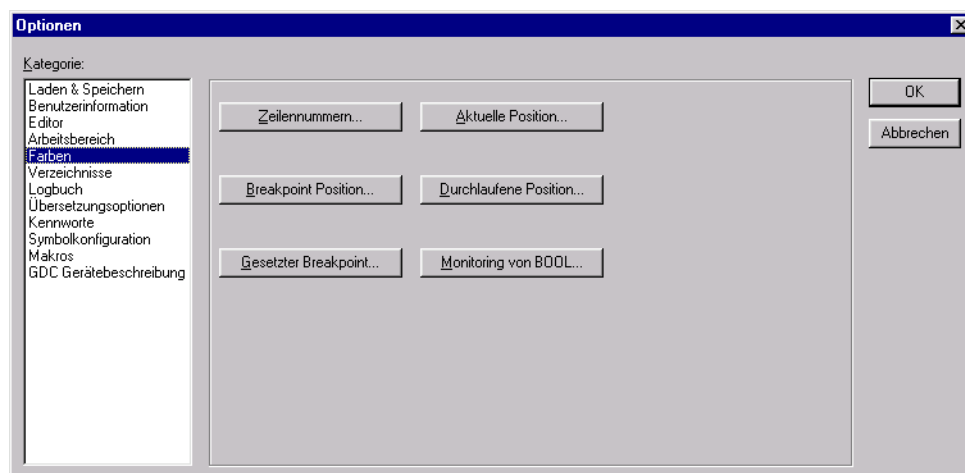
Druckbereiche anzeigen

In jedem Editorfenster wird der Druckbereich durch rot gestrichelte Linien dargestellt. Der Druckbereich hängt von den Druckereigenschaften und der Größe des Content-Bereichs der eingestellten Druckvorlage **Datei→Einstellung Dokumentationen** ab.

Warnungen ignorieren mit F4

Nach dem Übersetzungslauf springt der Fokus beim drücken von **F4** im Meldungsfenster nur in den Zeilen mit Fehlermeldungen. Die Warnungsausgaben werden dabei ignoriert.

Optionen→Farben

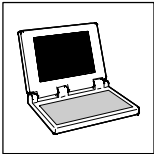


Sie können die Farbeinstellungen des DDS für folgende Darstellungen ändern:

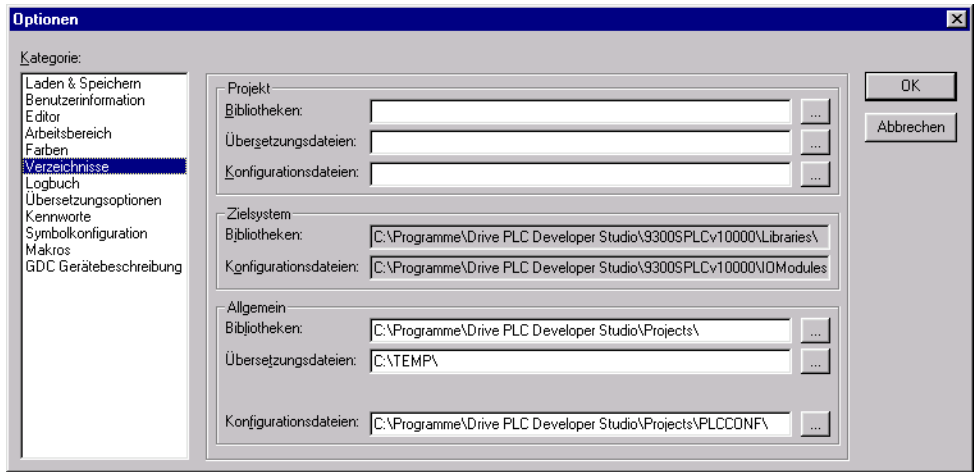
- **Zeilennummern** (hellgrau*)
- **Breakpoint Position** (dunkelgrau*)
- **Gesetzter Breakpoint** (hellblau*)
- **Aktuelle Position** (rot*)
- **Durchlaufenen Positionen** (grün*)
- **Monitoring von BOOL** (blau*)

Klicken Sie auf eine der Schaltflächen, um das Dialogfeld *Farben* zum Auswählen der Farbe für die entsprechende Darstellung zu öffnen.

*DDS-Voreinstellung



Optionen→Verzeichnisse



Projekt

Das DDS sucht nach Bibliotheken und Konfigurationsdateien in Verzeichnissen, die in einem Projekt eingetragen sind. Es können auch Verzeichnisse eingetragen werden, die als Ablage von Übersetzungsdateien verwendet werden sollen.

Wenn Sie die Schaltfläche (...) hinter einem Eingabefeld aktivieren, öffnet sich ein Dialogfeld zum Auswählen des Verzeichnisses innerhalb Ihrer Verzeichnisstruktur ("Browse"). Für Bibliotheks- und Konfigurationsdateien können jeweils mehrere Pfade getrennt durch ein Semikolon ";" eingegeben werden. Die Angaben werden mit dem Projekt gespeichert.

Allgemein

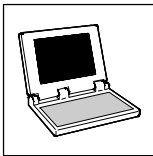
Hier können Verzeichnisse eingetragen werden, die DDS nach Bibliotheken und Konfigurationsdateien durchsuchen. Es können auch Verzeichnisse eingetragen werden, die als Ablage von Übersetzungsdateien verwendet werden sollen.

Wenn Sie die Schaltfläche (...) hinter einem Eingabefeld aktivieren, öffnet sich ein Dialogfeld zum Auswählen des Verzeichnisses innerhalb Ihrer Verzeichnisstruktur ("Browse"). Für Bibliotheks- und Konfigurationsdateien können jeweils mehrere Pfade getrennt durch ein Semikolon ";" eingegeben werden. Die Angaben werden in die ini-Datei des Programmsystems geschrieben und gelten für alle Projekte.

Zielsystem

Hier werden die Verzeichnisse für Bibliotheken und Konfigurationsdateien dargestellt, die im Zielsystem, z.B. durch Angaben in der Target-Datei, eingestellt sind. Diese Fenster sind nicht editierbar. Ein Eintrag kann selektiert und kopiert werden.

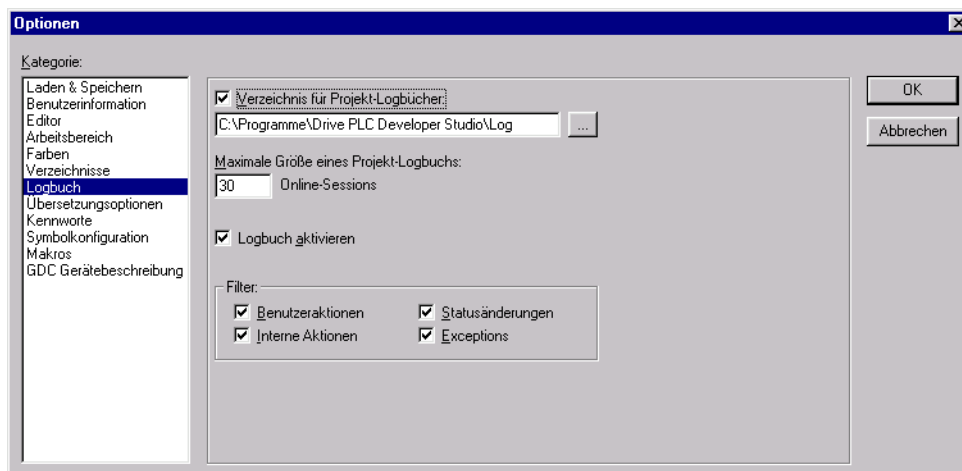
DDS sucht in der Reihenfolge Projekt, Zielsystem, Allgemein. Liegen gleichnamige Dateien vor, wird die im vorher durchsuchten Verzeichnis verwendet.



Drive PLC Developer Studio

Arbeitsbereich

Optionen→Logbuch



In diesem Dialog können Sie eine Datei konfigurieren, die als Projekt-Logbuch alle Benutzeraktionen und interne Vorgänge im Online-Modus chronologisch aufzeichnet.

Wird ein bestehendes Projekt geöffnet, für das noch kein Logbuch generiert wurde, öffnet sich ein entsprechender Dialog. Dieser zeigt an, daß ein Logbuch angelegt wird. Bei dem nächsten Login-Vorgang werden erstmals Einträge geschrieben.

Das Logbuch wird beim Speichern des Projekts automatisch als Binärdatei mitgespeichert. Mit der Option **Verzeichnis für Projekt-Logbücher** haben Sie die Möglichkeit das Logbuch in einem anderen Verzeichnis zu speichern.

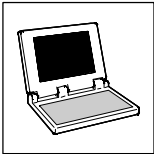
Die Logbuch-Datei erhält automatisch den Namen des Projekts mit der Erweiterung .log. Unter **Maximale Größe eines Projekt-Logbuchs** wird die Höchstanzahl der aufzuzeichnenden **Online-Sessions** festgelegt. Wird während der Aufzeichnung diese Anzahl überschritten, wird der jeweils älteste Eintrag zugunsten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

Die Funktion Logbuch kann im Kontrollkästchen **Logbuch aktivieren** ein- und ausgeschaltet werden.

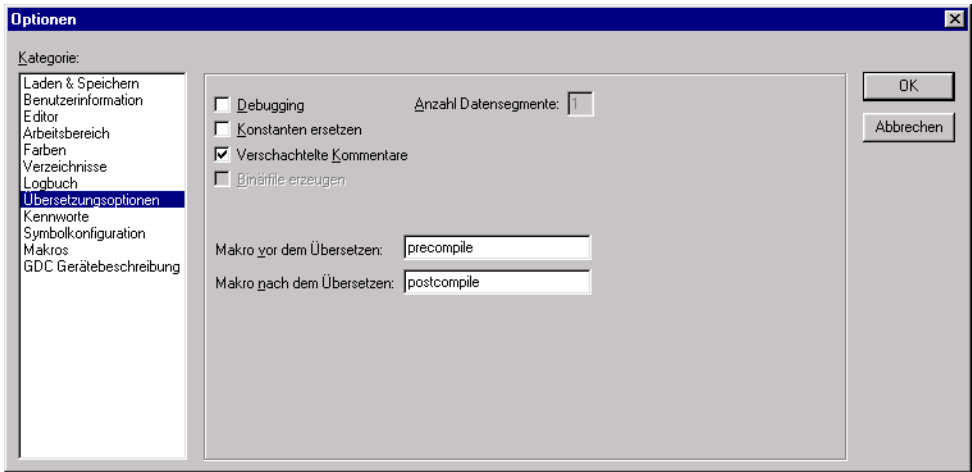
Im Gruppenfeld **Filter** können Sie wählen, welche Aktionen aufgezeichnet werden sollen.

- Benutzeraktion
- Interne Aktion
- Statusänderung
- Exception

Nur Aktionen der markierten Kategorien werden im Logbuch-Fenster angezeigt, bzw. in die Logbuch-Datei geschrieben. (6-43)



Optionen→Übersetzungsoptionen



Debugging

Ist die Option **Debugging** aktiviert, kann der Code deutlich umfangreicher werden, da zusätzlicher Debugging-Code generiert wird. Dieser ist notwendig, um die Debugging-Funktionen des DDS zu nutzen. Die Option wird mit dem Projekt gespeichert.

Nur wenn das Kontrollkästchen **Debugging** markiert ist, können Breakpoints gesetzt werden und das Durchgehen in Einzelschritten ist möglich.

- Wenn Sie die Option **Debugging** deaktivieren, ermöglichen Sie dadurch eine schnellere Ausführung und einen geringeren Code-Umfang.

Konstanten ersetzen

Mit der Option wird für jede Konstante direkt der Wert geladen. Im Online-Modus werden die Konstanten 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. Das ermöglicht zwar das Schreiben des Variablenwertes, bedeutet aber längere Bearbeitungszeit.

Verschachtelte Kommentare

Diese Option ermöglicht das Eingeben von verschachtelten Kommentaren.

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

Der Kommentar, der mit der ersten Klammer beginnt, ist der äußere. Er wird mit der letzten Klammer geschlossen.

Binärfile erzeugen

Ist diese Option gewählt, wird beim Übersetzen ein binäres Abbild des erzeugten Codes (Bootprojekt) im Projektverzeichnis angelegt. Dateiname: projektname.bin.

Makro vor dem Übersetzen

Mit der Option kann auf dem Übersetzungsvorgang Einfluß genommen werden. Das Makro wird vor dem Übersetzungslauf ausgeführt.



Drive PLC Developer Studio

Arbeitsbereich

Makro nach dem Übersetzen

Mit der Option kann auf dem Übersetzungsvorgang Einfluß genommen werden. Das Makro wird nach dem Übersetzungsvorgang ausgeführt.

Makrobefehle

Im Anhang finden Sie eine Liste über die gesamten Makros.

Kommandozeilen Befehle (15-1)

Kommandodatei Befehle (15-2)

Folgende Makrobefehle können nicht ausgeführt werden.

file new, file open, file close, file save as, file quit, online,
project compile, project check, project build, debug, watchlist

Eine ausführlichere Beschreibung zum Erstellen von Makros finden Sie unter
Optionen→Makros

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

Optionen→Kennworte

Zur Sicherung Ihrer Dateien vor unerwünschten Zugriffen bietet das DDS Ihnen die Möglichkeit, das Öffnen oder Verändern Ihrer Datei durch ein Kennwort zu schützen.

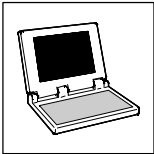
Kennwort zum Öffnen einer Datei festlegen:

1. Geben Sie das gewünschte Kennwort im Textfeld **Kennwort** ein.
Für jeden getippten Buchstaben erscheint im Feld ein Stern (*).
2. Wiederholen Sie die Eingabe des Kennwortes im Textfeld **Kennwort bestätigen**.
3. Klicken Sie auf **OK**, um das Dialogfeld zu schließen.

Wenn die Meldung "Das Kennwort und seine Bestätigung stimmen nicht überein." erscheint, haben Sie sich bei einem der beiden Einträge vertippt. Wiederholen Sie in diesem Fall am besten beide Einträge, so daß das Dialogfeld ohne Fehlermeldung schließt.

Wenn Sie nun die Datei abspeichern und erneut öffnen, erscheint ein Dialogfeld, in dem Sie aufgefordert werden, das Kennwort einzugeben.

Das Projekt wird nur dann geöffnet, wenn Sie das richtige Kennwort eingetippt haben, ansonsten erscheint die Meldung "Das Kennwort ist nicht korrekt."



Kennwort zum Verändern einer Datei festlegen:

Neben dem Öffnen der Datei können Sie auch das Verändern der Datei mit einem Kennwort schützen:

1. Geben Sie das gewünschte Kennwort im Textfeld **Schreibschutz-Kennwort** ein.
Für jeden getippten Buchstaben erscheint im Feld ein Stern (*).
2. Wiederholen Sie die Eingabe des Kennwortes im Textfeld **Schreibschutz-Kennwort bestätigen**.
3. Klicken Sie auf **OK**, um das Dialogfeld zu schließen.

Wenn die Meldung "Das Kennwort und seine Bestätigung stimmen nicht überein." erscheint, haben Sie sich bei einem der beiden Einträge vertippt. Wiederholen Sie in diesem Fall am besten beide Einträge, so daß das Dialogfeld ohne Fehlermeldung schließt.

Ein schreibgeschütztes Projekt kann auch ohne Kennwort geöffnet werden.

- Klicken Sie hierzu einfach auf **Abbrechen**, wenn das DDS Sie beim Öffnen der Datei auffordert, das Schreibschutz-Kennwort einzugeben. Nun können Sie das Projekt zwar übersetzen, in die Steuerung laden, simulieren etc., Sie können es aber nicht verändern.
- In der Statuszeile ist **LESEN** aktiv.



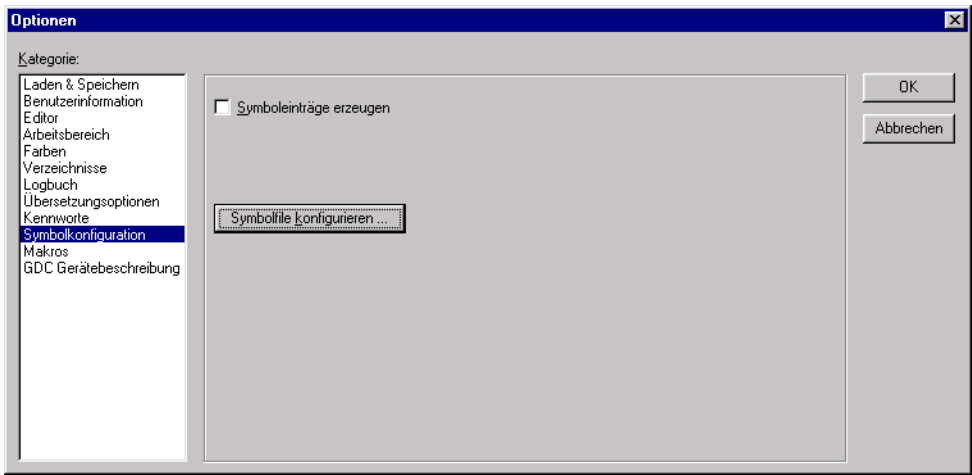
Hinweis!

Merken Sie sich die Kennwörter gut. Wenn Sie ein Kennwort tatsächlich mal vergessen haben sollten, wenden Sie sich bitte direkt an Lenze.

- Die Kennwörter werden mit dem Projekt gespeichert.
- Um differenziertere Zugriffsrechte zu schaffen, können Sie Arbeitsgruppen festlegen.

Optionen→Symbolkonfiguration

Wenn Sie die Kategorie *Symbolkonfiguration* wählen, wird folgender Dialog geöffnet.



Der Dialog dient der Konfiguration der Symboldatei. (Textdatei*.sym bzw. Binärdatei*.sdb) Diese sind für den Datenaustausch mit der Steuerung über die Symbolschnittstelle nötig und werden dazu beispielsweise vom Global Drive Oscilloscope verwendet.

Symboleinträge erzeugen

Ist das Kontrollkästchen markiert, werden automatisch bei jedem Übersetzen des Projekts Symboleinträge für die Projektvariablen in der Datei angelegt.

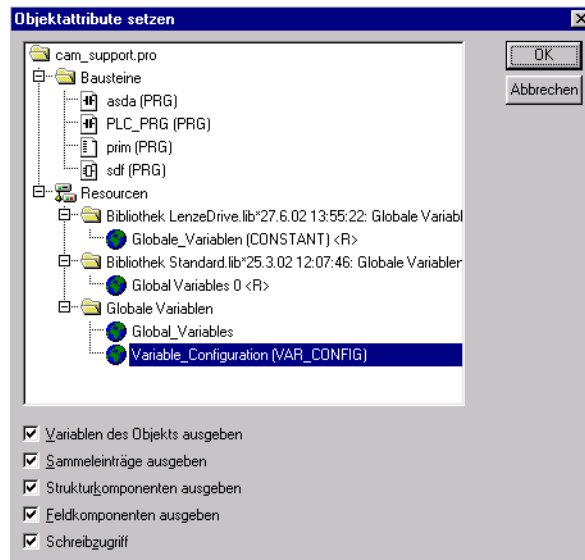


Drive PLC Developer Studio

Arbeitsbereich

Symbolfile konfigurieren

Mit dieser Schaltfläche öffnen Sie das Dialogfeld *Objektattribute setzen*.



Wählen Sie aus der Baumstruktur die gewünschten Projektbausteine aus und markieren Sie das gewünschte Kontrollkästchen.

Variablen des Objekts angeben

Die Variablen des gewählten Objekts werden in das Symbolfile ausgegeben. Die weiteren Optionen werden nur wirksam, wenn dieses Kontrollkästchen markiert ist.

Sammeleinträge ausgeben

Für Strukturen und Arrays des Objekts werden Einträge zum Zugriff auf die Gesamtvariablen erzeugt.

Strukturkomponenten ausgeben

Für Strukturen des Objekts wird für jede Variablenkomponente ein eigener Eintrag erzeugt.

Feldkomponenten ausgeben

Für Arrays des Objekts wird für jede Variablenkomponente ein eigenen Eintrag erzeugt.

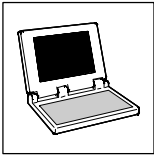
Schreibzugriff

Die Variablen des Objekts dürfen vom OPC-Server verändert werden.

Wenn die Einstellung für die aktuelle Bausteinauswahl vorgenommen wurde, kann ein anderer Baustein ausgewählt und mit einer anderen Optionen versehen werden.

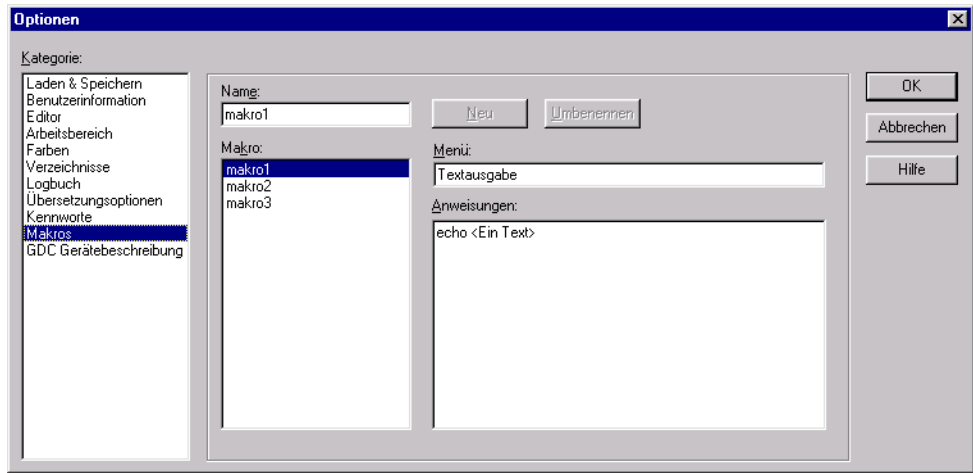
OK

Wenn Sie mit der Schaltfläche das Dialogfeld schließen, werden alle Änderungen übernommen.



Optionen→Makros

Wenn Sie die Kategorie *Makros* wählen, wird folgender Dialog geöffnet.



Name

In diesem Textfeld geben Sie den Namen für das Makro ein.

Neu

Drücken Sie die Schaltfläche **Neu** um das erstellte Makro aufzunehmen.

Makro

In diesem Feld sind alle erstellten Makros aufgelistet. Markierte Makros können mit der Taste **<Entf>** gelöscht werden.

Umbenennen

Ist ein bestehendes Makro in der Makro-Liste selektiert, können Sie im Dialogfeld *Name* einen anderen Namen eingeben. Drücken Sie die Schaltfläche **Umbenennen**, erhält das Makro den neuen Namen.

Anweisungen

In diesem Dialogfeld werden Kommandos definiert bzw. editiert, die das Makro umfassen sollen. Eine neue Anweisungszeile wird mit **<Strg>+<Enter>** eingefügt. Über die rechte Maustaste erhalten Sie das Kontextmenü mit den üblichen Editorfunktionen. Zusammengehörige Bestandteile eines Kommandos können mit Ausführungszeichen zusammengefügt werden.

Makrobefehle - Kommandos

Im Anhang finden Sie eine Liste über die gesamten Makros.

Kommandozeilen Befehle (📖 15-1)

Kommandodatei Befehle (📖 15-2)

Menü

In diesem Dialogfeld wird definiert, mit welchem Menüeintrag das Makro unter **Bearbeiten→Makros** eingehängt werden soll. Um einen Buchstaben als Short-Cut zu erhalten, muß diesem das Zeichen & vorangestellt werden. Der Name **Ma&kro 1** erzeugt den Menüeintrag **Makro 1**.

OK

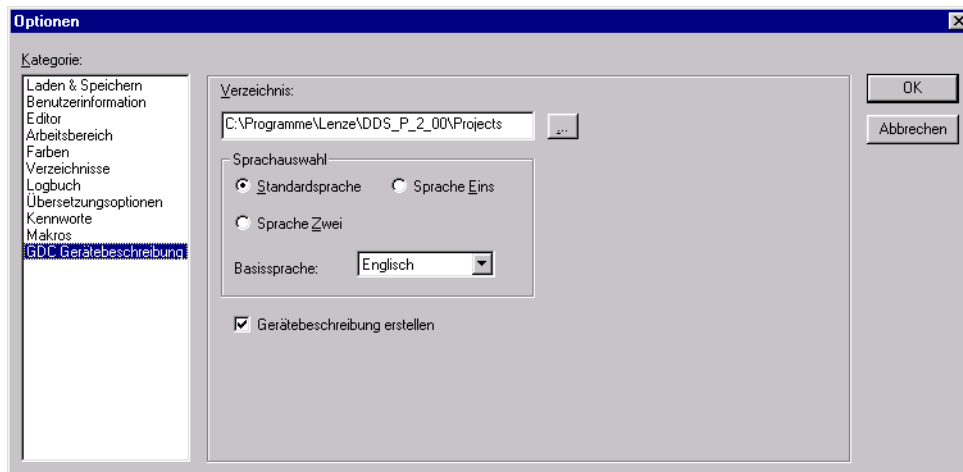
Verläßt den Dialog und speichert die Einträge im Projekt.
Eine Prüfung der Makros erfolgt erst beim Ausführen des Menübefehls.



Drive PLC Developer Studio

Arbeitsbereich

Optionen → GDC Gerätebeschreibung



In diesem Dialogfeld können Sie festlegen, ob und für welche Sprachen eine projektbezogene Gerätebeschreibungsdatei (PDB) für Global Drive Control (GDC) erstellt werden soll.



Tip!

Eine PDB wird nur erstellt, wenn im Textfeld **Verzeichnis** ein Verzeichnis angegeben ist und das Kontrollkästchen **Gerätebeschreibung erstellen** aktiviert ist! Die Erstellung/Aktualisierung der PDB erfolgt dann bei dem Übersetzen des Projektes, falls sich PDB relevante Daten geändert haben.

Der Dateiname der PDB setzt sich zusammen aus dem Namen des Projektes gefolgt von einem Index ("_S", "_1", "_2") für die entsprechende Sprache und der Dateiendung ".pdb".

Beispiel:

- Name des Projekts: "Example"
- Sprachauswahl: *Standardsprache* und *Sprache Zwei*
- ⇒ Dateinamen der PDBs: "Example_S.pdb" und "Example_2.pdb"

Verzeichnis

Im Eingabefeld *Verzeichnis* legen Sie den Pfad zu dem Verzeichnis fest, in dem die vom DDS zu erstellenden Gerätebeschreibungsdateien (PDBs) für Global Drive Control gespeichert werden sollen.

- Wenn Sie die Schaltfläche ... hinter dem Eingabefeld aktivieren, öffnet sich ein Dialogfeld zum Auswählen des Verzeichnisses innerhalb Ihrer Verzeichnisstruktur ("Browse").

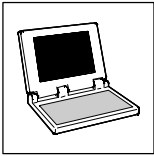
Sprachauswahl

Hier können Sie die Sprachen auswählen, für die jeweils eine PDB erstellt werden soll.

- Bis zu drei länderspezifische PDBs können für ein Projekt erstellt werden.
- Die entsprechenden GDC-Anzeigetexte für die jeweilige Codestelle legen Sie im Instanz-Parametermanager im Dialogfeld *Weitere Eingaben und Informationen* (Aufruf über Schaltfläche **erweitert** im Parametermanager) fest.
- Ist eine Sprache ausgewählt, erscheint vor der Sprache ein Haken.

Basissprache

In der ausgewählten Sprache wird die pdb-Datei erstellt.



Gerätebeschreibung erstellen

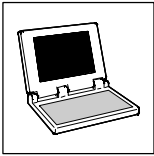
Ist das Kontrollkästchen **Gerätebeschreibung erstellen** aktiviert, wird bei jedem Übersetzen des Projektes für jede unter Sprachauswahl ausgewählte Sprache eine PDB für Global Drive Control im unter *Verzeichnis* angegebenen Verzeichnis erstellt/aktualisiert.

- Diese projektbezogene PDB enthält die Basiscodestellen der entsprechenden PLC (basierend auf dessen Basis-PDB) sowie die projektspezifischen Codestellen, die im Parametermanager definiert wurden.
- Erstellung erfolgt nur, wenn sich PDB-relevante Daten geändert haben.
- Die Gerätebeschreibung wird zusätzlich in das Projektverzeichnis abgelegt, um das Handling der Software Global Drive Loader zu erleichtern.
- Falls der Lenze OPC-Server installiert ist, wird ebenfalls die Gerätebeschreibung in dessen PDB-Verzeichnis kopiert.



Drive PLC Developer Studio

Arbeitsbereich



6 Arbeiten mit Projekten und Objekten

6.1 Projekte verwalten

Die Befehle des DDS, die sich auf ein ganzes Projekt beziehen, finden Sie im Hauptmenü unter **Datei** und **Projekt**.

Im Menü **Projekt** befinden sich auch Befehle, die sich auf Objekte beziehen, eine Beschreibung dieser Befehle finden Sie im Kapitel "Arbeiten mit Objekten". (📖 6-24)



Hinweis!

Schreibschutz in der bin-Datei

Bei jedem geänderten und übersetzten Projekt wird die zugehörige *.bin überschrieben. Diese Datei wird mit dem Projekt im selben Ordner abgelegt. Ist diese Datei schreibgeschützt, kann das Projekt nicht übersetzt werden. Sie müssen den Schreibschutz entfernen.

6.1.1 Befehle im Menü "Datei"

6.1.1.1 Neu

Symbol:		Menübefehl: Datei→Neu	Tastaturbefehl: -
---------	--	------------------------------	-------------------

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

6.1.1.2 Neu, mit Vorlage

Symbol: -	Menübefehl: Datei→Neu, mit Vorlage	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl legen Sie ein neues Projekt mit Namen dem Namen "Unbenannt" samt einer optionalen Vorlage an.

Die Vorlage ist vom Typ *.lpc und beinhaltet eine feste Lenze-Konfiguration.

Die Lenze-Konfiguration beinhaltet:

- Programm-POE
Bausteine, in denen die Technologiefunktion realisiert ist.
- benötigte Bibliotheken
- globale Variablen
- Einträge in der Steuerungskonfiguration
- Einträge in der Taskkonfiguration
- vordefinierte Anwender-Codestellen
- ggf. Visualisierung und Rezepturverwaltung

Wählen Sie im Dialogfeld *Öffnen* eine Vorlagedatei mit dem Zusatz "*.lpc" aus.

Sind für die Vorlage **Kennworte** oder **Arbeitsgruppen** definiert worden, erscheint ein Dialog zur Eingabe des Paßworts.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.1.1.3 Öffnen

Symbol:	Menübefehl: Datei→Öffnen	Tastaturbefehl: <Strg>+<O>
---------	---------------------------------	---

Mit diesem Befehl öffnen Sie ein bereits bestehendes Projekt.

- Wenn bereits ein Projekt geöffnet war und verändert wurde, dann erfolgt eine Nachfrage, ob dieses Projekt gespeichert werden soll.

Wählen Sie im Dialogfeld *Öffnen* eine Projektdatei mit dem Zusatz **.pro* oder eine Bibliotheksdatei mit dem Zusatz **.lib* aus. Diese Datei muß existieren; es ist nicht möglich, mit dem Befehl **Öffnen** ein neues Projekt zu erzeugen.

- Ganz unten im Menü **Datei** sind die zuletzt geöffneten Projekte aufgelistet. Wenn Sie eines davon auswählen, wird dieses Projekt geöffnet.
- Sind für das Projekt Kennworte oder Arbeitsgruppen definiert worden, erscheint beim Öffnen das Dialogfeld zur Eingabe des Kennwortes.

6.1.1.4 Schließen

Symbol: -	Menübefehl: Datei→Schließen	Tastaturbefehl: -
-----------	------------------------------------	-------------------

Mit diesem Befehl schließen Sie das aktuell geöffnete Projekt.

- Wenn das Projekt verändert wurde, erscheint ein Dialogfeld mit der Nachfrage, ob diese Veränderungen gespeichert werden sollen.
- Wenn das zu speichernde Projekt den Namen "Unbenannt" trägt, muß ein neuer Name dafür eingegeben werden.

6.1.1.5 Speichern

Symbol:	Menübefehl: Datei→Speichern	Tastaturbefehl: <Strg>+<S>
---------	------------------------------------	---

Mit diesem Befehl speichern Sie das Projekt, sofern es verändert wurde.

- Wenn das zu speichernde Projekt den Namen "Unbenannt" trägt, muß ein neuer Name dafür eingegeben werden.

6.1.1.6 Speichern unter

Symbol: -	Menübefehl: Datei→Speichern unter	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl kann das aktuelle Projekt unter einen anderen Namen oder als Bibliothek gespeichert werden. Die ursprüngliche Projektdatei bleibt dabei unverändert.

Nachdem der Befehl gewählt wurde, erscheint das Dialogfeld *Datei speichern unter*.

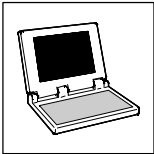
- Wählen Sie entweder einen existierenden Dateinamen, um ein existierendes Projekt zu überschreiben oder geben Sie einen neuen Dateinamen ein.
- Wählen Sie den gewünschten Dateityp aus.

Projekt unter neuen Namen speichern

Soll das Projekt nur unter einem neuen Namen abgespeichert werden, wählen Sie den Dateityp "DDS Projekt (*.pro)".

Klicken Sie anschließend auf **OK**.

- Das aktuelle Projekt wird in der angegebenen Datei gespeichert. Wenn der neue Dateiname bereits existiert, dann wird gefragt, ob diese Datei überschrieben werden soll.



Projekt als Bibliothek speichern

Soll das Projekt als Bibliothek abgespeichert werden, um es in anderen Projekten benutzen zu können, wählen Sie den Dateityp:

- "Interne Bibliothek (*.lib)", wenn Sie Ihre Bausteine im DDS programmiert haben.

Klicken Sie anschließend auf **OK**.

- Das aktuelle Projekt wird in der angegebenen Datei gespeichert. Wenn der neue Dateiname bereits existiert, dann wird gefragt, ob diese Datei überschrieben werden soll.
- Wenn Sie das Projekt als Bibliothek speichern, wird das ganze Projekt kompiliert. Wenn dabei ein Fehler auftritt, wird mitgeteilt, daß ein korrektes Projekt notwendig ist, um eine Bibliothek zu erzeugen. Das Projekt wird dann nicht als Bibliothek gespeichert.

6.1.1.7 Archiv speichern/versenden

Symbol: -	Menübefehl: Datei→Archiv speichern/versenden	Tastaturbefehl: -
-----------	--	-------------------

Das DDS besitzt eine Archivfunktion, mit der alle zu einem Projekt gehörenden Dateien (Bibliotheken, Bitmaps usw.) in einer Archivdatei gespeichert werden können.



Hinweis!

Um ein Projekt so zu archivieren, daß bei einem späteren Einloggen kein Übersetzen und kein Download erforderlich ist, muß vor der Archivierung ein Download durchgeführt werden und im Dialogfeld *Archiv speichern* das Kontrollkästchen **Compile-Information** aktiviert sein.

Mit dem Befehl kann eine ZIP-Archiv-Datei erstellt werden. Sie enthält alle für ein DDS-Projekt wichtigen Dateien. Die ZIP-Datei kann im Dateisystem abgespeichert oder mit E-Mail gesendet werden.

Nach dem Entpacken der ZIP-Datei auf einem beliebigen PC ist ein Login in einem Antriebsregler ohne vorherigen Download möglich.



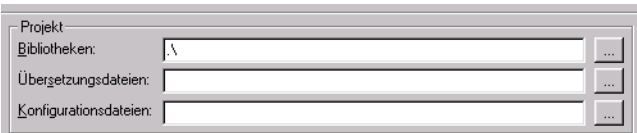
Hinweis!

Um die Möglichkeit des ZIP-Archivs auf diese Weise nutzen zu können, gehen Sie folgendermaßen vor.

1. Die zugehörigen Dateien (Bibliotheken, Bitmaps usw.) eines Projekts müssen relativ zum Projekt angelegt werden.



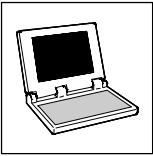
2. Öffnen Sie mit dem Menübefehl **Projekt→Optionen** Kategorie *Verzeichnisse* den folgenden Dialog und geben die in dem Eingabefeld **Bibliotheken** nur ".\" ein. Die Eingabefelder **Übersetzungsdateien** und **Konfigurationsdateien** bleiben leer.



Hinweis!

Pfade, denen ein Laufwerksbuchstabe zugeordnet wurde (SUBST[Laufwerk1: Laufwerk2:]Pfad) werden nicht unterstützt.

Verwenden Sie nicht die SUBST Funktion von Windows.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

3. Übersetzen Sie das Projekt.



Hinweis!

Nach dem Übersetzen muß das Projekt einmal gespeichert werden. Während des Speichervorgangs werden vom DDS Dateien erzeugt, die Bestandteil des ZIP-Archivs sind.

4. Mit dem Menübefehl **Datei→Archiv speichern/versenden** öffnen Sie den folgenden Dialog. Markieren Sie nur (Projektdatei, Referenzierte Bibliotheken, Compile-Information und Bitmap-Dateien). Nur wenn ein Bitmap im Projekt enthalten ist, läßt sich dieser Menüpunkt markieren. Sind keine Bitmaps enthalten kann diese Markierung unbeachtet bleiben.

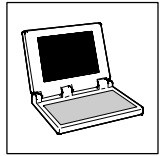


5. Sind die entsprechenden Markierungen vorhanden, speichern Sie das Archiv.

Folgende Informationen in das Archiv aufnehmen

Neben der beschriebenen Verwendung, kann das ZIP-Archiv auch generell benutzt werden. Eine Kategorie muß mit (✓) ausgewählt werden.

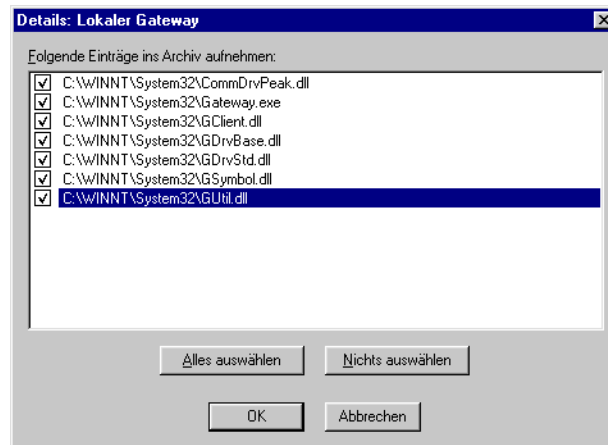
Kategorie	Zugehörige Dateien
Projektdatei	<Projektname>.pro (die DDS Projektdatei)
Referenzierte Bibliotheken	*.lib, *.obj, *.hex (Bibliotheken und ggf. die zugehörigen obj- und hex-Dateien)
Compile-Informationen	*.ci (Informationen des letzten Übersetzungslaufs) *.ri (Informationen des letzten Downloads) <temp>.* (temporäre Übersetzungs und Download-Dateien) auch für die Simulation
INI-Datei	DDS.ini
Logbuch	*.log (Projekt-Logbuch)
Bitmap-Dateien	*.bmp (Bitmaps, die in Projektbausteinen und Visualisierungen verwendet werden)
Registrierungseinträge	Registry.reg (Einträge für Automation Alliance, Gateway und SPS)
Symbol-Dateien	*.sdb, *.sym (Aus dem Projekt erzeugte Symbolinformation)
Konfigurationsdateien	Dateien für die Steuerungskonfiguration (Konfigurationsdateien, Gerätestammdateien, Icons etc.) e.g. *.cfg *.con *.eds *.dib *.ico...
Target-Dateien	*.trg (Target-Dateien im Binärformat für alle installierten Targets) *.txt (Target-Dateien im Binärformat für alle installierten Targets, wenn verfügbar)
Lokaler Gateway	Gateway-Dateien: Gateway.exe, GatewayDDE.exe, GClient.dll, GDrvBase.dll, GDrvStd.dll, Ghandle.dll, GSymbol.dll, GUtil.dll und weitere im Gateway-Verzeichnis vorhandene DLLs



Details

Mit der Schaltfläche **Details** wird ein Dialogfeld aufgerufen, in dem Informationen zu der jeweiligen Kategorie gezielt ausgewählt werden können.

Das Dialogfeld *Details* wird anhand *Details: Lokaler Gateway* erläutert.



Das Dialogfeld zeigt die Details Lokaler Gateway. Mit (✓) können Sie gezielt auswählen. Mit der Schaltfläche **Alles Auswählen** wird alles markiert, **Nichts Auswählen** entfernt die Marken. Bestätigen Sie den Dialog mit **OK**.

Andere Dateien

Mit der Schaltfläche **Andere Dateien** öffnen Sie ein Dialogfeld zum aufnehmen benutzerdefinierter Dateien in das Archiv.

Kommentar

Mit der Schaltfläche öffnet sich ein Texteditor zum Erstellen einer README.TXT Datei. Sie enthält den eingegebenen Text, dem automatisch das Erstellungsdatum und Versionsnummer der aktuell verwendeten DDS-Version hinzugefügt wird.

Generieren des ZIP-Archivs

Wenn alle Einstellungen vorgenommen und geprüft wurden, kann das ZIP-Archiv erstellt werden.

Speichern

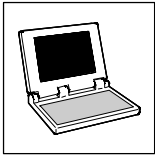
Eine ZIP-Datei wird generiert und gespeichert. Mit dem Windows Dialog kann die Datei an den gewünschten Ort abgelegt werden. Der Default-Name der Datei ist projektname.zip. Wird die Schaltfläche **Speichern** gedrückt, startet die Generierung des Archivs. Neben einem Fortschrittkatalog wird der Ablauf in dem Meldungsfenster protokolliert.

Senden

Eine leere E-Mail mit dem Projekt als Zip-Datei im Anhang wird erstellt. Diese Funktion setzt einen korrekte Installation des MAPI (Messaging Application Programming Interface) voraus.

Abbrechen

Der Vorgang wird abgebrochen. Alle Einstellungen werden nicht gespeichert.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.1.1.8 Drucken

Symbol: -	Menübefehl: Datei→Drucken	Tastaturbefehl: <Strg>+<P>
-----------	----------------------------------	---

Mit diesem Befehl wird der Inhalt des aktiven Fensters gedruckt.

Nach Auswahl des Befehls erscheint das Dialogfeld *Drucken*.

- Wählen Sie die gewünschte Option und klicken Sie anschließend auf **OK**. Das aktive Fenster wird ausgedruckt. Jeder Editor kann farbig ausgedruckt werden.
- Mit der Schaltfläche **Eigenschaften** öffnen Sie das Dialogfeld *Druckereinrichtung*.
- Das Layout Ihres Ausdruckes können Sie mit dem Befehl **Datei→Einstellungen Dokumentation** festlegen.
- Während des Druckens wird Ihnen in einem Dialogfeld die Anzahl der bereits gedruckten Seiten mitgeteilt. Wenn Sie dieses Dialogfeld schließen, stoppt der Druckvorgang nach der nächsten Seite.
- Um Ihr gesamtes Projekt zu dokumentieren, verwenden Sie den Befehl **Projekt→Dokumentieren**.
- Wenn Sie eine Dokumentvorlage zu Ihrem Projekt erstellen wollen, in der Sie die Kommentare zu allen im Projekt verwendeten Variablen vorgeben können, dann öffnen Sie eine globale Variablenliste und benutzen Sie den Befehl **Extras→Doku-Vorlage erstellen**. (☞ 8-3)

6.1.1.9 Einstellungen Dokumentation

Symbol: -	Menübefehl: Datei→Einstellungen Dokumentation	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl können Sie das Layout der ausgedruckten Seiten festlegen.



Datei

Im Textfeld **Datei** können Sie den Namen der Datei mit Suffix ".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, klicken Sie auf **Durchsuchen**, um die gewünschte Datei in Ihrer Verzeichnisstruktur zu suchen.

Bearbeiten

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

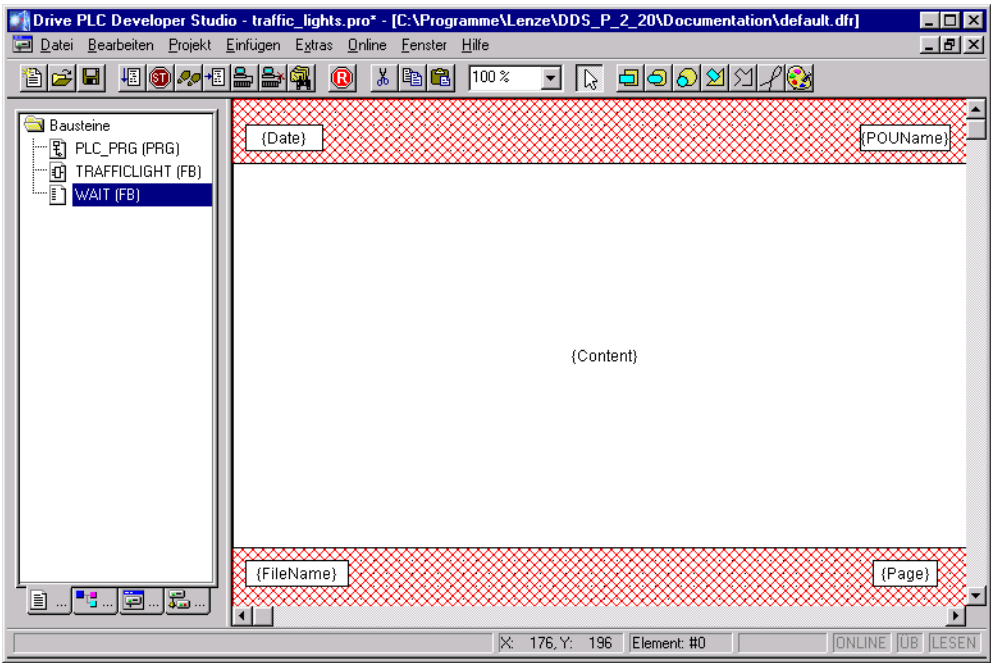


Tip!

Mit der Schaltfläche **Einrichtung** öffnen Sie das Dialogfeld *Druckereinrichtung*. Dort können Sie Einstellungen für die Dokumentation vornehmen.

Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten



Mit dem Befehl **Einfügen→Platzhalter** und anschließender Auswahl eines der fünf Platzhalter können Sie diesen durch Aufziehen eines Rechteckes im Layout einfügen und falls erforderlich an die gewünschte Position verschieben.

Platzhalter werden im Ausdruck wie folgt ersetzt:

Befehl	Platzhalter	Wirkung
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.

Desweiteren können Sie mit **Einfügen→Bitmap** eine Bitmap-Grafik (z. B. ein Firmenlogo) in die Seite einfügen. Dabei müssen Sie nach Auswahl der Grafik ebenfalls ein Rechteck mit der Maus in dem Layout aufziehen. Zusätzlich können weitere Visualisierungselemente eingefügt werden.

Neue Seite für Objekt/Neue Seite je Unterobjekt

Sie können ebenfalls auswählen, ob eine neue Seite für jedes Objekt und für jedes Unterobjekt begonnen wird.

Einrichtung

Mit der Schaltfläche **Einrichtung** öffnen Sie das Dialogfeld *Druckereinrichtung*. Dort können Sie Drucker, Papierformat usw. einstellen.

OK/Abbrechen

Klicken Sie auf **OK**, um die Veränderungen zu übernehmen bzw. auf **Abbrechen**, um die Änderungen zu verwerfen.

Wenn die Vorlage verändert wurde, erfolgt eine Nachfrage beim Schließen des Fensters, ob diese Veränderungen gespeichert werden sollen oder nicht.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.1.1.10 Beenden

Symbol: -	Menübefehl: Datei→Beenden	Tastaturbefehl: <Alt>+<F4>
-----------	----------------------------------	---

Mit diesem Befehl beenden Sie das DDS.

- Wenn ein Projekt geöffnet ist und verändert wurde, dann erfolgt eine Nachfrage, ob dieses Projekt gespeichert werden soll.

6.1.2 Befehle im Menü “Projekt”

6.1.2.1 Übersetzen

Symbol: -	Menübefehl: Projekt→übersetzen	Tastaturbefehl: <F11>
-----------	---------------------------------------	------------------------------------

Mit diesem Befehl wird das Projekt kompiliert. Sie können die syntaktische Korrektheit Ihres Programms überprüfen. Der Übersetzungsvorgang ist inkrementell, es werden nur die veränderten Bausteine neu übersetzt. Für einen nicht inkrementellen Übersetzungsvorgang muß vorher der Befehl **Projekt→Alles bereinigen** ausgeführt werden.

Für Zielsysteme, die Online Change unterstützen, sind nach dem Übersetzungslauf alle Bausteine im Objekt-Manager mit einem blauen Pfeil gekennzeichnet. Sie werden beim nächsten Download auf die Steuerung geladen.

Der mit **Projekt→Übersetzen** durchgeführte Übersetzungslauf erfolgt automatisch bei **Online→Einloggen**

Fehler und Warnungen sind mit Nummern gekennzeichnet.

Schnittstelle des Bausteins 'ABLAUF'
Fehler 4024: ABLAUF (8): Erwarte ';' oder '=' vor 'END_VAR'
Fehler 3781: ABLAUF (9): 'END_VAR' oder Bezeichner erwartet
Fehler 4357: ABLAUF (1): Die IEC-Bibliothek wurde nicht gefunden
Schnittstelle des Bausteins 'AMPEL'
Fehler 4024: AMPEL (3): Erwarte ';' vor '
Schnittstelle des Bausteins 'WARTEN'
Fehler 3740: WARTEN (9): Unbekannter Typ 'TB'
5 Fehler, 0 Warnung(en)

Mit dem Befehl **Projekt→Optionen** in der Kategorie **Laden & Speichern** Kontrollkästchen **Sichern vor Übersetzen** wird das Projekt vor dem Übersetzen gespeichert.



Tip!

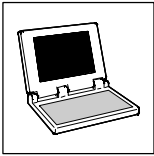
Die Querverweise entstehen während der Kompilierung und werden nicht in der Übersetzungsinformation gespeichert!

Um die Befehle **Aufrufbaum ausgeben**, **Querverweisliste ausgeben** und die Befehle **Unbenutzte Variablen ausgeben**, **Konkurrierender Zugriff** und **Mehrfaches Schreiben auf Output** des Menüs **Projekt→Überprüfen** anzuwenden, muß das Projekt nach einer Veränderung neu übersetzt werden.

6.1.2.2 Alles Übersetzen

Mit **Projekt→Alles Übersetzen** wird im Gegensatz zu **Projekt→Übersetzen** das Projekt komplett neu kompiliert. Die Download-Information wird dabei **nicht** verworfen.

Mit dem Menübefehl **Projekt→Alles bereinigen** werden dagegen die Download-Informationen gelöscht.



6.1.2.3 Alles bereinigen



Hinweis!

Ein Login ohne einen erneuten Download des Projekts ist nur dann möglich, wenn vorher die Datei *.ri mit den Projektinformationen des letzten Downloads explizit außerhalb des Projektverzeichnisses abgelegt wurde und vor dem Einloggen wieder geladen werden kann.

Führen Sie den Menübefehl **Projekt→Download-Information laden** aus.

Symbol: -	Menübefehl: Projekt→Alles bereinigen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl werden die Informationen des letzten Downloads und des letzten Übersetzungsvorgangs gelöscht. Nachdem der Befehl ausgeführt wurde, erscheint ein Dialog, der darauf hinweist, daß ein Login ohne erneuten Download nicht möglich ist. Sie können hier den Vorgang fortsetzen oder abbrechen.

6.1.2.4 Download-Information laden

Symbol: -	Menübefehl: Projekt→Download-Information laden...	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl kann die entsprechende Download-Information direkt wieder geladen werden. Falls die Information nicht im Projektverzeichnis gespeichert wurde. Der Standarddialog **Datei→Öffnen** ermöglicht das gezielte Laden von gespeicherten Informationen.

- Die Download-Information wird bei jedem Download automatisch in eine Datei **Projektname Targetidentifizier.ri**. Bsp: LD_FillingPlants00000000r.ri und ins Projektverzeichnis gelegt.
- Sie wird automatisch bei jedem Öffnen des Projekts wieder neu geladen. Sie dient der Steuerung zu einem ID-Check, der folgendes prüft:
Entspricht das Projekt auf die Steuerung dem geöffneten Projekt.
- Es 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 über den Menübefehl **Projekt→Alles bereinigen** aus dem Projektverzeichnis gelöscht, kann die Download-Information mit **Projekt→Download-Information laden...** explizit von einem anderen Verzeichnis geladen werden.

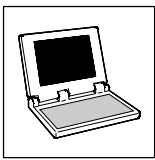
6.1.2.5 In andere Sprache übersetzen

Symbol: -	Menübefehl: Projekt→In andere Sprach übersetzen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Menüpunkt dient dazu die aktuelle Projektdatei in eine andere linguistische Sprache zu übersetzen. Das geschieht durch das Einlesen einer Übersetzungsdatei, die aus dem Projekt heraus erzeugt wurde. Anschließend wird die Übersetzungsdatei mit Hilfe eines Texteditors mit Übersetzungstexten in der gewünschten Landessprache ergänzt.

Dazu hat das Menü zwei Unterpunkte.

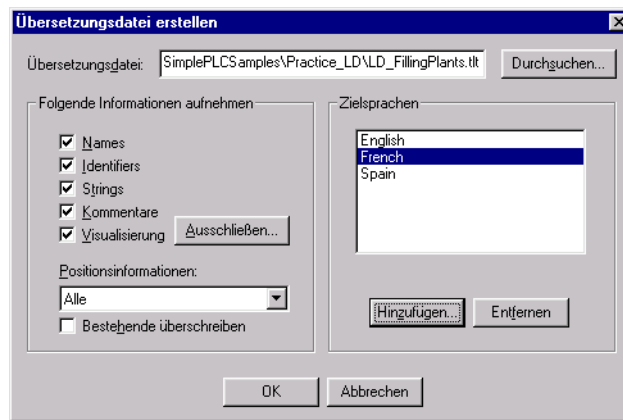
- **Projekt→In andere Sprache übersetzen→Übersetzungsdatei erstellen**
- **Projekt→In andere Sprache übersetzen→Projekt übersetzen**



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Übersetzungsdatei erstellen



- Geben Sie in das Feld **Übersetzungsdatei** einen Pfad ein, um die Datei an der gewünschten Stelle zu speichern. Die Standard-Dateierweiterung ist *.tlt (Textdatei).
- Um eine vorhandene Übersetzungsdatei zu bearbeiten, können Sie diese mit dem Windows-Dialog **Durchsuchen** auswählen und öffnen.
- Sie können folgende Informationen der erstellten oder modifizierten Übersetzungsdatei optional mitgeben.
 - Name, Titel im Objekt Organizer
 - Identifiers, Bezeichner
 - Strings, Kommentare, Visualisierung
 - Positionsinformationen

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

Als Visualisierungstexte gelten hier die Elemente **Text** und **Tooltip-Text** der Visualisierungselemente.



Tip!

Für Visualisierungstexte **Text** und **Tooltip-Text** der Visualisierungselemente ist zu beachten, daß sie im Konfigurationsdialog des Visualisierungselements zwischen zwei Zeichen # eingegeben werden müssen. #text#, um in die Übersetzungsdatei übernommen zu werden. (📖 9-1)

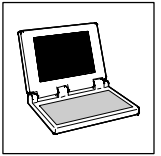
Positionsinformationen

Diese beschreiben 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 Positionsformatierungen generiert.
- **Erstmaliges Auftreten** Es wird die Position in die Übersetzungsdatei aufgenommen, an der das zu übersetzende Elemente erstmalig auftritt.
- **Alle** Es werden alle Positionen angegeben, an denen das betreffende Element im Projekt auftritt.

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



Tip!

Für jedes Element (Sprachsymbol) werden maximal 64 Positionsinformationen generiert. Auch wenn im Dialogfeld *Übersetzungsdatei erstellen* im Kombinationsfeld **Positionsinformationen** Alle ausgewählt wurde.

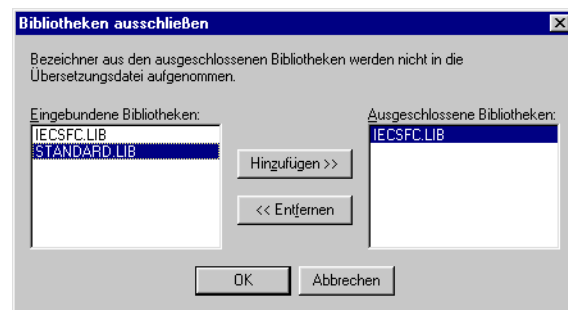
Positionsinformationen überschreiben

Alle bestehenden Positionsinformationen in der Übersetzungsdatei, die in Bearbeitung sind, werden überschrieben. Unabhängig von welchem Projekt sie generiert wurden.

Diese Liste enthält Bezeichner für alle Sprachen, die in der Übersetzungsdatei enthalten sind und die nach Beenden des Dialogs *Übersetzungsdatei erstellen* aufgenommen werden sollen.

Ausschließen

Die Schaltfläche **Ausschließen** öffnet das Dialogfeld *Bibliotheken ausschließen*.



Hier können die im Projekt vorhandenen Bibliotheken ausgewählt werden, deren Benutzerinformation nicht in die Übersetzungsdatei übernommen werden sollen.

Über die Schaltflächen **Hinzufügen** und **Entfernen** können Sie bestimmen welche Bibliotheken ausgeschlossen und welche eingebunden werden. Bestätigen Sie anschließend mit **OK**. Der Dialog schließt sich.

Zielsprachen

Die Schaltfläche **Hinzufügen** im Gruppenfeld **Zielsprachen** öffnet das Dialogfeld *Zielsprachen hinzufügen..*

Dort können Sie eine weitere Sprache hinzufügen. Geben Sie z. B. English(USA) ein und bestätigen mit **OK**. Die eingegebene Sprache darf am Anfang und Ende kein Leerzeichen oder Umlaut haben. Sind die Eingaben nicht in Ordnung oder vollständig ist die Schaltfläche **OK** nicht aktiv.

Die Schaltfläche **Entfernen** im Gruppenfeld **Zielsprachen** löscht eine Sprache aus der Liste. Dabei wird nur die markierte Sprache entfernt.

Übersetzungsdatei erstellen

Drücken Sie **OK** im Dialogfeld *Übersetzungsdatei erstellen* wird eine Übersetzungsdatei generiert. Besteht bereits eine gleichnamige Übersetzungsdatei wird erst vom System eine Sicherheitsabfrage gestellt.

Wählen Sie **Nein** kehren Sie zurück zum Dialogfeld *Übersetzungsdatei erstellen*.

Wählen Sie **Ja** wird eine Kopie der bestehenden Übersetzungsdatei mit den Dateinamen Backup_of_Übersetzungsdatei.tlt erstellt.

Beim Erzeugen einer Übersetzungsdatei geschieht folgendes.

- Für jede neue Zielsprache wird für jedes auszugebene Sprachsymbol ein Platzhalter **##TODO** generiert.
- Wird eine bereits existierende Übersetzungsdatei bearbeitet, werden Dateneinträge von Sprachen, die in der Übersetzungsdatei, aber nicht in der Zielsprachenliste stehen, entfernt. Unabhängig aus welchem Projekt sie generiert wurden.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Übersetzungsdatei bearbeiten

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. Für jedes 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 Zielsprache Englisch(USA) und Französisch sind vorgesehen. In diesem Beispiel wurde auch die Positionsinformation für das zu übersetzende Projektelelement mitgegeben.

Vor der Übersetzung

```
##NAME_ITEM
[D:\DDS\projects\Bspdt_22.pro::ST_Visualisierung::0]
ST_Visualisierung
##English :: ##TODO
##French :: ##TODO
##END_NAME_ITEM
```

Nach der Übersetzung

##TODO wurde durch die englischen und französischen Ausdrücke ersetzt.

```
##NAME_ITEM
[D:\DDS\projects\Bspdt_22.pro::ST_Visualisierung::0]
ST_Visualisierung
##English :: ST_Visualization
##French :: ST_Visu
##END_NAME_ITEM
```

Achten Sie darauf, daß übersetzte Bezeichner und Namen gemäß der Norm gültig bleiben. String und Kommentare in die entsprechenden Klammerzeichen eingeschlossen werden.



Hinweis!

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

6.1.2.6

Projekt übersetzen

Symbol: -	Menübefehl: Projekt→In anderer Sprache übersetzen	Tastaturbefehl: -
-----------	--	-------------------

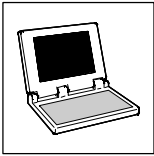
Dieser Menü-Befehl hat die Unterpunkte

- Übersetzungsdatei erstellen
- Projekt übersetzen



Tip!

Der Übersetzungslauf kann nicht rückgängig gemacht werden. Speichern Sie eine Projektkopie vor dem Übersetzen unter anderen Namen.



Übersetzungsdatei erstellen

Projekt übersetzen



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

Übersetzungsdatei

In diesem Textfeld den Pfad der Übersetzungsdatei an.

Durchsuchen

Mit der Schaltfläche erhalten Sie den Windows-Dialog zum Auswählen einer Datei.

Zielsprache

In diesem Kombinationsfeld sind die Sprachen-Bezeichner zur Auswahl der Zielsprache enthalten.

OK

Die Schaltfläche startet den Übersetzungslauf des aktuellen Projekts mit Hilfe der angegebenen Übersetzungsdatei in die ausgewählte Zielsprache. Während der Übersetzung wird der Fortschritt und gegebenenfalls Fehlermeldungen angezeigt. Nach der Übersetzung wird das Dialogfeld und alle anderen geöffneten Dialogfenster geschlossen.

Abbrechen

Die Schaltfläche schließt das Dialogfeld ohne Änderungen.

Falls die Übersetzungsdatei fehlerhafte Eingaben enthält, wird nach **OK** eine Fehlermeldung mit Dateipfad und fehlerhafte Zeile, ausgegeben.

[C:\Programme\DDS\projects\visu.tlt(78)]; Übersetzungstext erwartet.

6.1.2.7

Dokumentieren

Symbol: -	Menübefehl: Projekt→Dokumentieren...	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl ermöglicht es Ihnen, eine Dokumentation Ihres gesamten Projekts zu drucken.

Zu einer vollständigen Dokumentation gehören:

- Inhalt der Dokumentation
- Bausteine
- Datentypen
- Aufrufbäume von Bausteinen und Datentypen
- Visualisierungen
- Ressourcen (Globale Variablen, Variablenkonfiguration, Steuerungskonfiguration, Taskkonfiguration, Watch- und Rezepturverwalter)
- Querverweisliste
- Tabelle mit den über den Parametermanager vergebenen Codestellen

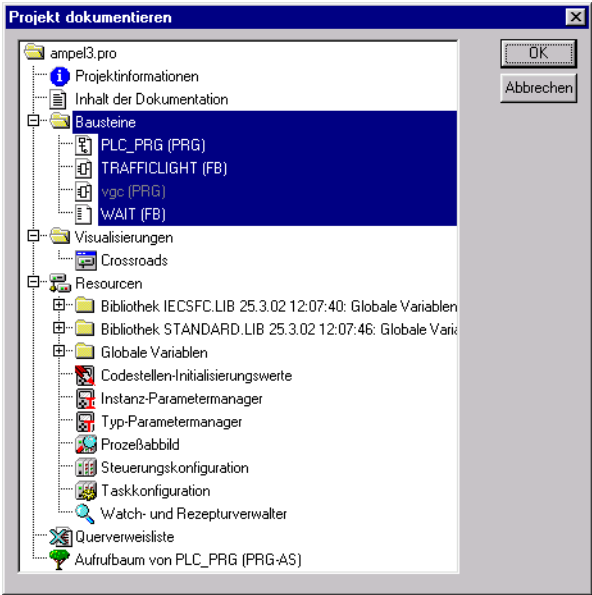
Für die Dokumentation der "Aufrufbäume" und der "Querverweisliste" muß das Projekt fehlerfrei übersetzt worden sein.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Nach Auswahl des Befehls öffnet sich ein Dialogfeld, in dem Sie die zu druckenden Objekte auswählen können:



- Wenn Sie Ihre Auswahl getroffen haben, klicken Sie auf **OK**.
- Es erscheint das Dialogfeld *Drucken*.
- Das Layout der zu druckenden Seiten können Sie mit **Datei→Einstellungen Dokumentation** festlegen.

6.1.2.8 Exportieren

Symbol: -	Menübefehl: Projekt→Exportieren...	Tastaturbefehl: -
-----------	---	-------------------

Das DDS bietet Ihnen 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 das DDS 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, Visualisierungen und die Ressourcen exportiert werden.
- Zusätzlich können die Einträge im Bibliotheksverwalter, d. h. die Verknüpfungsinformationen zu den Bibliotheken mit exportiert werden. Die Bibliotheken werden nicht exportiert.

Nach Auswahl des Befehls öffnet sich ein Dialogfeld, in dem Sie die zu exportierenden Objekte auswählen können.

- Wenn Sie Ihre Auswahl getroffen haben, klicken Sie auf **OK**.
- Es erscheint das Dialogfeld *Speichern*. Geben Sie einen Dateinamen mit dem Suffix ".exp" an.



6.1.2.9 Importieren

Symbol: -	Menübefehl: Projekt→Importieren...	Tastaturbefehl: -
-----------	---	-------------------

Wählen Sie in dem erscheinenden Dialogfeld 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 ein Dialogfeld mit der Frage "Wollen Sie es ersetzen?".

- Bestätigen Sie mit **Ja**, um das Objekt im Projekt durch das Objekt aus der Importdatei zu ersetzen
- Bestätigen Sie mit **Nein**, um das Objekt aus der Importdatei mit einer Namensergänzung (Unterstrich und fortlaufende Zähler "_0", "_1", ...) zu importieren.
- Bestätigen Sie mit **Ja, alle**, bzw. **Nein, alle**, um die zuvor beschriebenen Aktionen auf **alle** Objekte anzuwenden.

Im Meldungsfenster wird der Import protokolliert.



Hinweis!

Folgende Ressourcen-Elemente werden nicht oder unvollständig exportiert.

- Codestellen-Initialisierungswerte
- Instanz-Parametermanager
- Typ-Parametermanager
- Taskkonfiguration

6.1.2.10 Vergleichen

Symbol: -	Menübefehl: Projekt→Vergleichen...	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl dient zum vergleichen von

- zwei Projekten
- einem geöffneten Projekt mit dem zuletzt gespeicherten. Damit die Änderungen angezeigt werden dürfen Sie nicht zwischenspeichern.



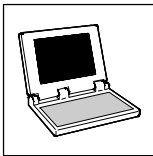
Tip!

Ist der Vergleichsmodus aktiv (Statusleiste: Compare), kann das Projekt nicht editiert werden.

Vereinbarungen

Bezeichnung	Bedeutung
aktuelles Projekt	das Projekt, welches im Moment bearbeitet wird
Vergleichsprojekt	das Projekt, welches zum Vergleich aufgerufen wird
Vergleichsmodus	wenn Projekt→Vergleichen gewählt wurde, ist das Projekt im Vergleichsmodus
Einheit	die kleinste Vergleichseinheit

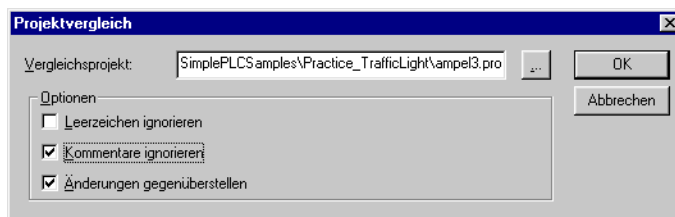
Im Vergleichsmodus werden aktuelles und Vergleichsobjekt in einem geteilten Fenster dargestellt. Bei den Editorbausteinen besteht die Möglichkeit der direkten Gegenüberstellung der Inhalte. Vor dem Vergleichslauf können Filter aktiviert werden.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Projektvergleich durchführen



Vergleichsprojekt

In diesem Textfeld wird standardmäßig der Pfad des Vergleichsprojektes angezeigt.

Leerzeichen ignorieren

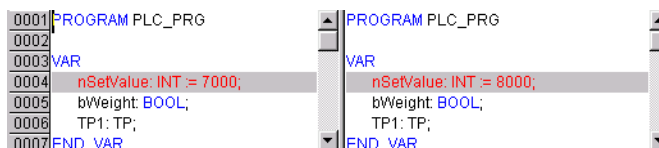
Ist das Kontrollkästchen aktiv, werden Unterschiede in den Leerzeichen nicht angezeigt.

Kommentare ignorieren

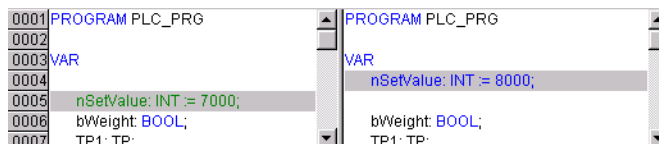
Ist das Kontrollkästchen aktiv, werden Unterschiede in den Kommentaren nicht angezeigt.

Änderungen gegenüberstellen

Ist das Kontrollkästchen aktiv, werden Einheiten angezeigt, die geändert aber nicht gelöscht wurden. In der Zeile 4 wurde der Zahlenwert geändert. 7000 ist die Änderung und 8000 der ursprüngliche Wert.

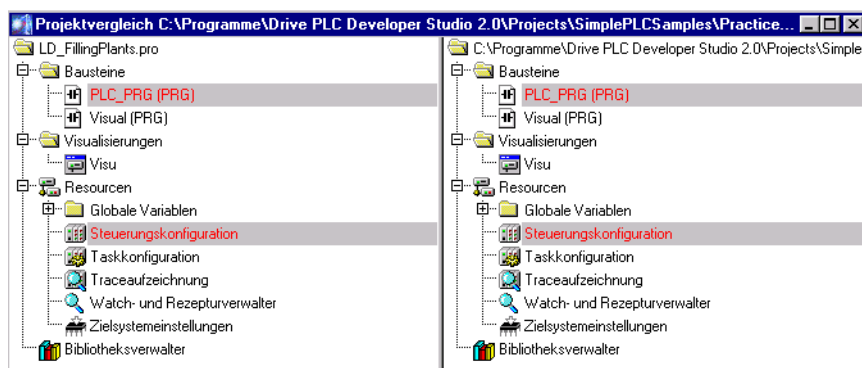


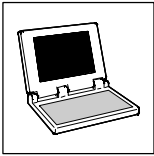
Ist das Kontrollkästchen nicht aktiv, werden die Werte nicht direkt gegenüber gestellt.



Ergebnisdarstellung

Die Ergebnisse werden zuerst in einem Verzeichnisbaum dargestellt. Mit Doppelklick kann der jeweilige Baustein geöffnet werden, um die markierten Änderungen zu sehen.





Farben

Unterschiede werden durch farbigen Text gekennzeichnet.

Rot

Einheit wurde modifiziert und wird in beiden Fensterhälften rot dargestellt.

Blau

Einheit ist nur im Vergleichsprojekt vorhanden. Im aktuellen Projekt ist eine Lücke.

Grün

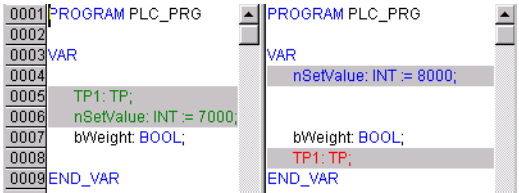
Einheit ist nur im aktuellen Projekt vorhanden. Im Vergleichsobjekt wird eine Lücke eingefügt.

Schwarz

Einheit ohne Unterschiede.

Hinter den Bausteinen des aktuellen Projekts kann folgender Text erscheinen.

- (Eigenschaften geändert)
Dieser Text erscheint hinter einen Bausteinamen, wenn Unterschiede in den Bausteineigenschaften gefunden wurden.
- (Zugriffsrechte geändert)
Dieser Text erscheint hinter einen Bausteinamen, wenn Unterschiede in den Zugriffsrechten gefunden wurden.



6.1.2.11 Kopieren

Symbol: -	Menübefehl: Projekt→Kopieren	Tastaturbefehl: -
-----------	-------------------------------------	-------------------

Mit diesem Befehl können Sie Objekte (Bausteine, Datentypen, Visualisierungen und Ressourcen) und Verknüpfungen zu Bibliotheken aus anderen Projekten in Ihr aktuelles Projekt kopieren.

Nach Auswahl des Befehls öffnet sich das Dialogfeld *Projekt kopieren*.

1. Suchen Sie sich das Projekt aus, aus dem Sie Objekte in das aktuelle Projekt kopieren wollen.
2. Klicken Sie auf **Öffnen**, um das Projekt zu öffnen.

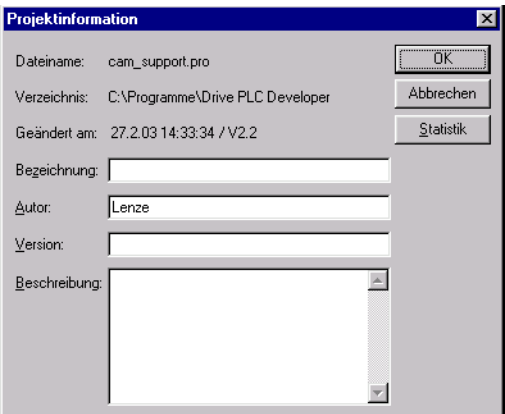
Daraufhin öffnet sich ein Dialogfeld, in dem Sie die gewünschten Objekte auswählen können.

- Wenn ein gleichnamiges Objekt im Projekt bereits besteht, dann erhält der Name des neuen Objekts als letztes Zeichen einen Unterstrich und eine Nummer ("_1", "_2", ...).

6.1.2.12 Projektinformation

Symbol: -	Menübefehl: Projekt→Projektinformation	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl können Sie Informationen zu Ihrem Projekt abspeichern.



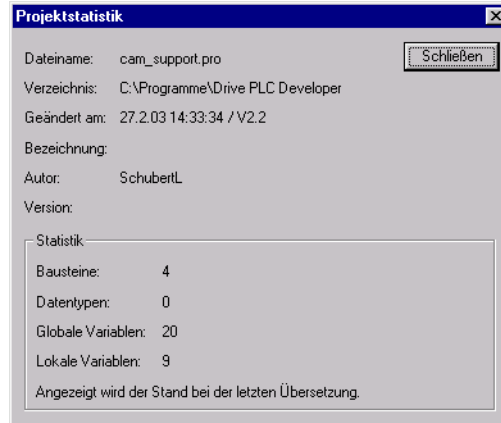


Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Statistik

Durch Klicken auf die Schaltfläche **Statistik** erhalten Sie eine statistische Auskunft über das Projekt.



Die Statistik enthält die Angaben aus der Projektinformation, sowie die Anzahl der Bausteine, der Datentypen, der lokalen und globalen Variablen, wie sie bei der letzten Übersetzung aufgezeichnet wurden.

- Ist im Dialogfeld *Optionen* in der Kategorie *Laden & Speichern* die Option **Projektinformation verlangen** aktiviert, so wird beim Abspeichern eines neuen Projekts oder beim Abspeichern eines Projekts unter einem neuen Namen automatisch die Projektinformation aufgerufen.

6.1.2.13 Global Suchen

Symbol:	Menübefehl: Projekt→Global Suchen	Tastaturbefehl: -
---------	--	-------------------

Mit diesem Befehl können Sie nach dem Vorkommen eines Textes in Bausteinen, Datentypen oder den Objekten der globalen Variablen suchen.

Nach Auswahl des Befehls öffnet sich ein Dialogfeld, in dem Sie die gewünschten Objekte auswählen können.

Wenn Sie die Auswahl mit **OK** bestätigt haben, erscheint das Dialogfeld *Suchen*.

- Wird ein Text in einem Objekt gefunden, so wird das Objekt in den zugehörigen Editor geladen und die Fundstelle angezeigt.
- Das Anzeigen des gefundenen Textes sowie das Suchen und Weitersuchen verhalten sich analog zum Menübefehl **Bearbeiten→Suchen**.

Suchen nach

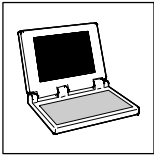
Hier kann die gesuchte Zeichenfolge eingegeben werden. Wird ein Text in einem Objekt gefunden, dann wird das Objekt in den zugehörigen Editor bzw. Bibliotheksverwalter geladen und die Fundstelle angezeigt.

Meldungsfenster

Wird diese Schaltfläche gedrückt, werden alle Verwendungsstellen der gesuchten Zeichenfolge in den ausgewählten Objekten zeilenweise im Meldungsfenster aufgelistet und eine Anzahl der gefundenen Stellen angegeben.

Folgende Informationen werden dabei angezeigt.

- Objektname
- Fundstelle im Deklarationsteil oder Implementationsteil eines Bausteins
- Zeilen bzw. Netzwerknummer
- komplette Zeile bei den Text-Editoren
- komplette Texteinheit bei den grafischen Editoren



Das Meldungsfenster kann folgendes anzeigen. Gesucht wird die Zeichenfolge bottom.

Global suchen
ST-EXAMPLE (PRG-ST)(Deklaration) #5 bottom: INT:= -250;
ST-EXAMPLE (PRG-ST)(Implementation) #14 bottom :=yVal + offset;
ST-EXAMPLE (PRG-ST)(Implementation) #19 IF (bottom >-250) THEN
ST-EXAMPLE (PRG-ST)(Implementation) #20 bottom := Bottom -offset;
Die Zeichenfolge 'bottom' wurde 4 mal gefunden

Mit einem Doppelklick auf eine der Zeilen, wird der entsprechende Editor geöffnet und die Zeile mit der Zeichenfolge markiert. Mit den Funktionstasten <F4> und <Umschalt>+<F4> kann zwischen den Ausgabzeilen gesprungen werden.



Hinweis!

Nicht implementiert für Parametermanager und Codestelleninitialisierung.

6.1.2.14 Global Ersetzen

Symbol: -	Menübefehl: Projekt→Global Ersetzen	Tastaturbefehl: -
-----------	-------------------------------------	-------------------

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.

Die Bibliotheken werden nicht zur Auswahl angeboten und es ist keine Ausgabe ins Meldungsfenster möglich.

Nach Auswahl des Befehls öffnet sich ein Dialogfeld, in dem Sie die gewünschten Objekte auswählen können.

Wenn Sie die Auswahl mit **OK** bestätigt haben, erscheint das Dialogfeld *Ersetzen*.

Weitersuchen

- Wird der Text im Kombinationsfeld **Suchen nach** in einem der zu durchsuchenden Objekte gefunden, wird die Fundstelle angezeigt.

Ersetzen

- Die aktuelle Fundstelle des Textes im Kombinationsfeld **Suchen nach** in den zu durchsuchenden Objekten wird durch den Text im Kombinationsfeld **Ersetzen durch** ersetzt.
- Die Markierung springt nach dem Ersetzungsvorgang zur nächsten Fundstelle.

Alles Ersetzen

- Alle Fundstellen des Textes im Kombinationsfeld **Suchen nach** in den zu durchsuchenden Objekten werden durch den Text im Kombinationsfeld **Ersetzen durch** ersetzt.

Abbrechen

- Beendet die "Suchen und Ersetzen"-Funktion.



Hinweis!

Nicht implementiert für Parametermanager und Codestelleninitialisierung.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.1.2.15 Projekt überprüfen

Symbol: -	Menübefehl: Projekt→überprüfen	Tastaturbefehl: -
-----------	---------------------------------------	-------------------

Dieser Menübefehl beinhaltet vier separate Unterpunkte.

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

Die Ergebnisse werden im Meldungsfenster ausgegeben.

Jede dieser vier Funktionen prüft den Stand des letzten Übersetzungslaufs. ein Projekt muß also mindestens einmal korrekt übersetzt sein. Ist das nicht der Fall sind die Menüpunkte nicht aktiv.

Unbenutzte Variablen

Diese Funktion sucht nach deklarierten Variablen, die nicht im Programm verwendet werden. Sie werden mit Bausteinname und Bausteinzeile ausgegeben.

Beispiel: PLC_PRG (4) - var1

Die Variablen in Bibliotheken werden nicht berücksichtigt.

Überlappende Speicherbereiche

Diese Funktion prüft während der Zuweisung von Variablen durch die AT-Deklaration das Überschneiden auf bestimmte Speicherbereiche.

Beispiel:

Durch die Zuweisung der Variablen

```
var1 AT %QB21: INT
var2 AT %QD5: DWORD
```

entsteht eine Überschneidung. Sie belegen das Byte 21 gleichzeitig. Die Ausgabe sieht dann wie folgt aus.

%QB21 wird durch folgende Variablen referenziert.

```
PLC_PRG (3): var1 AT %QB21
PLC_PRG (7): var2 AT %QD5
```

Konkurrierender Zugriff

Diese Funktion sucht nach Speicherbereichen, die in mehr als einer Task referenziert werden. Kein Unterschied zwischen Lese- und Schreibzugriff. Die Ausgabe sieht dann wie folgt aus.

%MB28 wird in folgenden Tasks referenziert:

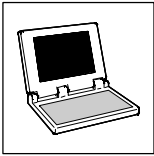
```
Task1 - PLC_PRG (6): %MB28 [Nur-Lese-Zugriff]
Task2 - POU1.ACTION (1) %MB28 [Schreibzugriff]
```

Mehrfaches Speichern auf Output

Diese Funktion sucht nach Speicherbereichen, auf die in einem Projekt an mehreren Stellen zugegriffen wird. Die Ausgabe sieht dann wie folgt aus.

%QB24 wird an folgenden Stellen beschrieben:

```
PLC_PRG (3): %QB24
PLC_PRG.POU1 (8): %QB24
```



6.1.3 Arbeitsgruppen

Es können im DDS bis zu acht Gruppen mit unterschiedlichen Zugriffsrechten auf Bausteine, Datentypen, Visualisierungen 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 ein 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, ist zunächst kein Passwort festgelegt.
- Solange kein Passwort für die Gruppe 0 festgelegt wurde, betritt man das Projekt automatisch als Mitglied der Gruppe 0.
- Wurde für ein Projekt ein Passwort für die Arbeitsgruppe 0 festgelegt, wird beim Öffnen des Projekts für alle Gruppen die Eingabe eines Passworts verlangt.

1. Wählen Sie im Kombinationsfeld **Arbeitsgruppe** die Gruppe aus, zu der Sie gehören.
2. Geben Sie in das Textfeld **Passwort** das Passwort ein.
3. Aktivieren Sie die Schaltfläche **OK**, um die Eingaben zu übernehmen.

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.



Achtung!

Ein Projekt kann über eine Arbeitsgruppe geöffnet werden, falls an der Arbeitsgruppe kein Passwort vergeben wurde.

- Mit dem Menübefehl **Projekt→Passwörter für Arbeitsgruppe** können Passwörter vergeben werden.
- Mit dem Menübefehl **Projekt→Objekte→Zugriffsrechte** werden die Rechte für einzelne oder alle Objekte vergeben.

6.1.3.1 Passwörter für Arbeitsgruppen

Symbol: -	Menübefehl: Projekt→Passwörter für Arbeitsgruppen	Tastaturbefehl: -
-----------	--	-------------------

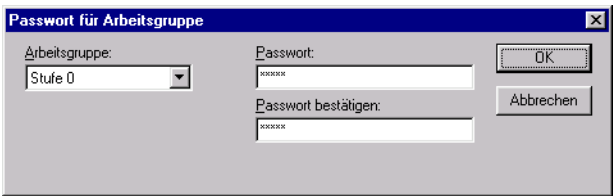
Mit diesem Befehl öffnen Sie das Dialogfeld *Passwort für Arbeitsgruppe*.



Hinweis!

Dieser Befehl kann nur von Mitgliedern der Gruppe 0 (Administratoren) ausgeführt werden.

Dialogfeld "Passwort für Arbeitsgruppe"





Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

1. Wählen Sie im Kombinationsfeld **Arbeitsgruppe** die Gruppe aus, für die Sie ein Passwort festlegen möchten.
2. Geben Sie in das Textfeld **Passwort** das Passwort ein. Für jeden getippten Buchstaben erscheint im Feld ein Stern (*).
3. Wiederholen Sie die Eingabe im Textfeld **Passwort bestätigen**.
4. Aktivieren Sie die Schaltfläche **OK**, um die Eingaben zu übernehmen.

Wenn die Meldung “Das Kennwort und seine Bestätigung stimmen nicht überein.” erscheint, haben Sie sich bei einem der beiden Einträge vertippt.

- Wiederholen Sie in diesem Fall am besten beide Einträge, so daß das Dialogfeld ohne Fehlermeldung schließt.
- Vergeben Sie gegebenenfalls erst danach ein Passwort für die nächste Gruppe durch erneuten Aufruf des Befehls.



Tip!
Mit dem Befehl **Objekt→Zugriffsrechte** können Sie die Rechte für einzelne oder alle Objekte vergeben.

6.1.3.2 Ausnahmeverhalten

Symbol: -	Menübefehl: Projekt→Ausnahmeverhalten	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie das Dialogfeld *Ausnahmeverhalten*. In diesem Dialogfeld können Sie die Überwachungszeit für die Zyklische Task sowie das generelle Verhalten des ausgewählten Zielsystems bei einem Tasküberlauf festlegen. (📖 8-34)

Das Aussehen des Dialogfeldes *Ausnahmeverhalten* hängt von den Zielsystemeinstellungen ab. (📖 8-41)

Servo PLC

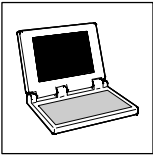
Zielsystem mit Antriebsfunktion



Hinweis!
Die Konfiguration der digitalen Ausgänge
– geht bei den Erweiterungsmodulen verloren.
– bleibt bei der Servo PLC erhalten.

Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten



	QSP	Trip/RSP	Warnung
Ereignis während des Programmablaufs	Reaktion des Zielsystems		
SPS Stopp, Breakpoint	- Fail QSP wird aktiviert - Codestelle C168, Subcode 1 - Fehlereintrag: kein	- RSP wird aktiviert - Codestelle C168, Subcode 1 - Fehlereintrag: kein	- Motor läuft weiter - Codestelle C168, Subcode 1 - Fehlereintrag: kein
Tasküberlauf	- Fail QSP wird aktiviert - Codestelle C168, Subcode 1 - Fehlereintrag: Tasküberlauf	- RSP wird aktiviert - Codestelle C168, Subcode 1 - Fehlereintrag: Tasküberlauf	- Motor läuft weiter - Codestelle C168, Subcode 1 - Fehlereintrag: kein

Drive PLC

Zielsystem ohne Antriebsfunktion

Ausnahmeverhalten

Verhalten bei Breakpoint oder Tasküberlauf

Reaktion des Zielsystems:

☒ Trip/RSP ☐ Warnung

Digitale Ausgänge werden

☐ auf LOW gesetzt ☒ nicht verändert

Überwachungszeit für die zyklische Task

Überwachungszeit (T#1ms - T#10000ms):

T#1s0ms

OK

Abbrechen



Hinweis!

Die Konfiguration der digitalen Ausgänge

- bleibt bei den Erweiterungsmodulen erhalten.
- bleibt bei der Drive PLC erhalten.

	QSP	Trip/RSP	Warnung
Ereignis während des Programmablaufs	Reaktion des Zielsystems		
SPS Stopp, Breakpoint		- RSP wird aktiviert - Codestelle C168, Subcode 1 - Fehlereintrag: kein	- Motor läuft weiter - Codestelle C168, Subcode 1 - Fehlereintrag: kein
Tasküberlauf		- RSP wird aktiviert - Codestelle C168, Subcode 1 - Fehlereintrag: Tasküberlauf	- Motor läuft weiter - Codestelle C168, Subcode 1 - Fehlereintrag: kein



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.2 Arbeiten mit Objekten

In diesem Kapitel wird beschrieben, wie Sie mit Objekten arbeiten und welche Hilfen Ihnen zur Verfügung stehen, um den Überblick über ein Projekt zu behalten (Ordner, Aufrufbaum, Querverweisliste, usw.).

6.2.1 Objekt

Als "Objekt" werden Bausteine, Datentypen, Visualisierungen und die Ressourcen, globale Variablen, Steuerungskonfiguration, Taskkonfiguration und Watch- und Rezepturverwalter, etc. bezeichnet.

- Die zur Strukturierung des Projektes eingefügten Ordner sind zum Teil im Projekt enthalten. Zum Beispiel der Ordner Bausteine mit der Aktion PLC_PRG.
- Sämtliche Objekte eines Projekts stehen im *Object Organizer*.



Tip!

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.

Mit Drag&Drop können Sie Objekte und Ordner innerhalb ihrer Objektart verschieben. Mit dieser Windows-Funktion können Sie Objekte markieren und bei gedrückter linker Maustaste verschieben. Kommt es beim Verschieben zu einer Namenskollision, erscheint eine Fehlermeldung und der Vorgang wird ignoriert. Es ist aber möglich in einem Ordner mit einem gleichnamigen Unterordner zu erzeugen.

6.2.1.1 Objekte auswählen

Ausgewählte Objekte sind blau unterlegt.

Auswahl des gesamten Projektes

Wenn Sie das gesamte Projekt selektieren wollen, selektieren Sie den Namen Ihres Projektes in der ersten Zeile.

Auswahl einzelner Objekte

Wollen Sie nur ein einzelnes Objekt selektieren, klicken Sie auf das entsprechende Objekt bzw. bewegen Sie 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.

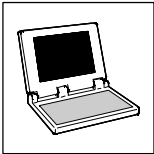
Wenn Sie ein Organisationsobjekt selektieren, sind alle zugehörigen Objekte ebenfalls selektiert.

Auswahl eines Bereiches von Objekten

Bei gedrückter <Umschalt>-Taste können Sie einen Bereich von Objekten auswählen.

Auswahl mehrerer einzelner Objekte

Bei gedrückter <Strg>-Taste können Sie mehrere einzelne Objekte auswählen.



6.2.2 Ordner

Um den Überblick bei größeren Projekten zu behalten, sollten Sie Ihre Bausteine, Datentypen, Visualisierungen und globale Variablen im *Object Organizer* sinnvoll in Ordnern gruppieren.

- Es ist eine beliebige Schachtelungstiefe von Ordnern möglich.
- Weitere Ordner können Sie mit dem Befehl **Neuer Ordner** einfügen.

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.
	Anstelle des Pluszeichens wird nun ein Minuszeichen angezeigt. Mit einem Klick auf das Minuszeichen kann der Ordner wieder zugeklappt werden.

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

Drag & Drop

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.



Tip!

Ordner haben keinen Einfluß auf das Programm, sondern dienen der übersichtlichen Strukturierung Ihres Projektes.

6.2.3 Befehle im Kontextmenü

6.2.3.1 Neuer Ordner

Symbol: -	Menübefehl: Kontextmenü→Neuer Ordner	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl im Kontextmenü des *Object Organizers* wird ein neuer Ordner als Ordnungsobjekt eingefügt.

- Ist ein Ordner selektiert, wird der neue Ordner unter den selektierten Ordner angelegt, ansonsten auf gleicher Ebene.
- Ist eine Aktion ausgewählt, wird der neue Ordner auf der Ebene des Bausteins eingefügt, in der die Aktion eingebunden ist.
- Das Kontextmenü erscheint, wenn ein Objekt oder die Objektart selektiert ist und Sie die rechte Maustaste oder <Umschalt>+<F10> drücken.

Namenskonvention für Ordner

- Neu eingefügte Ordner bekommen die Bezeichnung **Neuer Ordner**.
- Auf der gleichen Ebene können Ordner und Objekte nicht den gleichen Namen haben.

6.2.3.2 Knoten Expandieren

Symbol: -	Menübefehl: Kontextmenü→Knoten Expandieren	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl im Kontextmenü des *Object Organizers* werden die Objekte eingeblendet, die sich unter dem ausgewählten Objekt befinden.

- Bei Ordnern können Sie auch mit Doppelklick oder drücken der <Eingabetaste> die Ordner auf- und zuklappen.
- Das Kontextmenü erscheint, wenn ein Objekt oder die Objektart selektiert ist und Sie die rechte Maustaste oder <Umschalt>+<F10> drücken.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.2.3.3 Knoten Kollabieren

Symbol: -	Menübefehl: Kontextmenü→Knoten Kollabieren	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl im Kontextmenü des *Object Organizers* werden die dem ausgewählten Objekt untergeordneten Objekte wieder ausgeblendet.

- Bei Ordnern können Sie auch mit Doppelklick oder drücken der **<Eingabetaste>** die Ordner auf- und zuklappen.
- Das Kontextmenü erscheint, wenn ein Objekt oder die Objektart selektiert ist und Sie die rechte Maustaste oder **<Umschalt>+<F10>** drücken.

6.2.4 Befehle im Menü “Projekt”

6.2.4.1 Objekt löschen

Symbol: -	Menübefehl: Projekt→Objekt löschen	Tastaturbefehl: <Entf>
-----------	---	-------------------------------------

Mit diesem Befehl wird das aktuell markierte Objekt oder ein Ordner mit den darunterliegenden Objekten aus dem *Object Organizer* entfernt, und ist somit im Projekt gelöscht.

- Vor dem Löschvorgang erfolgt eine Sicherheitsabfrage, ob das Objekt wirklich gelöscht werden soll.
- Wenn das Editorfenster des Objekts geöffnet war, wird es automatisch geschlossen.
- Wenn Sie zum Löschen den Menübefehl **Bearbeiten→Ausschneiden** verwenden, wird das Objekt zusätzlich in die Zwischenablage geschoben.

6.2.4.2 Objekt einfügen

Symbol: -	Menübefehl: Projekt→Objekt einfügen	Tastaturbefehl: <Einf>
-----------	--	-------------------------------------

Mit diesem Befehl erstellen Sie ein neues Objekt. Die Art des Objektes hängt von der gewählten Registerkarte im *Object Organizer* ab.

In dem erscheinenden Dialogfeld geben Sie den Namen des neuen Objektes an.

Beachten Sie folgende Einschränkungen

- Der Bausteinname dürfen keine Leerzeichen haben.
- Bausteine und Datentypen dürfen nicht den gleichen Namen haben.
- Globale Variablenlisten dürfen nicht den gleichen Namen haben.
- Aktionen im selben Baustein dürfen nicht den gleichen Namen haben.
- Visualisierungen dürfen nicht den gleichen Namen haben.

In allen anderen Fällen sind Namensübereinstimmungen erlaubt. Aktionen verschiedener Bausteine können den gleichen Namen haben. Visualisierungen können den gleichen Namen wie Bausteine haben.

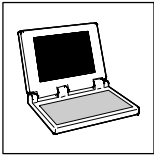
Dialog zum Anlegen eines neuen Bausteins

Der Dialog muß vollständig ausgefüllt sein. Wenn nicht gegen die genannte Namensvergabe verstoßen wurde, kann mit **OK** das neue Objekt im Objekt Organizer angelegt werden und das passende Eingabefenster erscheint.

Mit dem Menübefehl **Bearbeiten→Einfügen** wird das Objekt aus der Zwischenablage eingefügt. Dabei erscheint kein Dialog. Bei einem Verstoß gegen die Namensvergabe wird der Name mit einer fortlaufenden Nummer ergänzt. lade_1, lade_2...

- Beachten Sie, daß der Name des Objekts noch nicht verwendet ist.
- Wenn es sich um einen Baustein handelt, muß zusätzlich der Typ des Bausteins (Programm, Funktion oder Funktionsblock) und die Sprache, in der er programmiert werden soll, gewählt werden.

Nach dem Bestätigen der Eingabe erscheint das zum Objekt passende Eingabefenster.



6.2.4.3 Objekt umbenennen

Symbol: -	Menübefehl: Projekt→Objekt umbenennen	Tastaturbefehl: <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.
- Ist das Bearbeitungsfenster des Objekts geöffnet, dann ändert sich der Titel bei der Neubenennung des Objekts automatisch.

6.2.4.4 Objekt konvertieren

Symbol: -	Menübefehl: Projekt→Objekt konvertieren	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl kann nur auf Bausteine angewendet werden. Mit diesem Befehl können Sie Bausteine in den Sprachen CFC, ST, FUP, KOP und AWL in eine der drei Sprachen AWL, FUP und KOP konvertieren.



Hinweis!

Der Befehl kann nur auf Bausteine angewandt werden. Zum Ausführen des Befehls muß das Projekt übersetzt sein.

1. Geben Sie in dem erscheinenden Dialogfeld den Namen des neuen Bausteins an. Beachten Sie, daß der Name des Bausteins noch nicht für einen anderen Baustein verwendet worden sein darf.
2. Wählen Sie die Sprache aus, in die Sie konvertieren wollen.
3. Klicken Sie auf **OK**, um den neuen Baustein Ihrer Bausteinliste hinzuzufügen.

6.2.4.5 Objekt kopieren

Symbol: -	Menübefehl: Projekt→Objekt kopieren	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl wird ein ausgewähltes Objekt kopiert und unter einem neuem Namen abgespeichert.

Geben Sie in dem erscheinenden Dialogfeld den Namen des neuen Objektes an. Beachten Sie, daß der Name des Objekts noch nicht verwendet worden sein darf.

Klicken Sie auf **OK**, um das ausgewählte Objekt zu kopieren.

- Wenn Sie den Menübefehl **Bearbeiten→Kopieren** verwenden, wird das Objekt in die Zwischenablage kopiert, ohne daß ein Dialogfeld erscheint.

6.2.4.6 Objekt bearbeiten

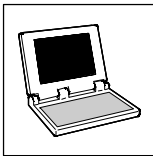
Symbol: -	Menübefehl: Projekt→Objekt bearbeiten	Tastaturbefehl: <Eingabetaste>
-----------	--	---

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.

Sie können ein Objekt auch bearbeiten, indem Sie

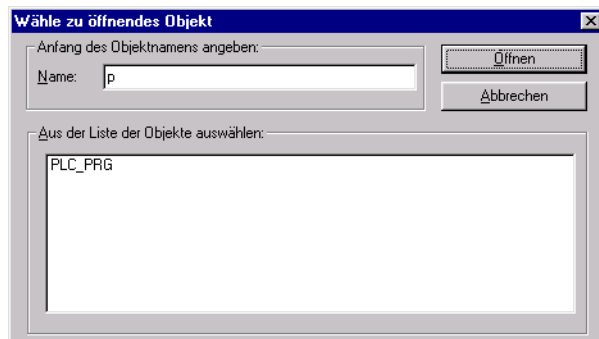
- mit der Maus auf das entsprechende Objekt doppelklicken oder
- in den *Object Organizer* klicken und den ersten Buchstaben des Objektnamens im *Object Organizer* eintippen.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Wenn Sie den ersten Buchstaben des Objektnamens im *Object Organizer* eintippen, öffnet sich ein Dialogfeld, in dem alle Objekte der eingestellten Objektart mit diesen Anfangsbuchstaben zur Auswahl stehen.



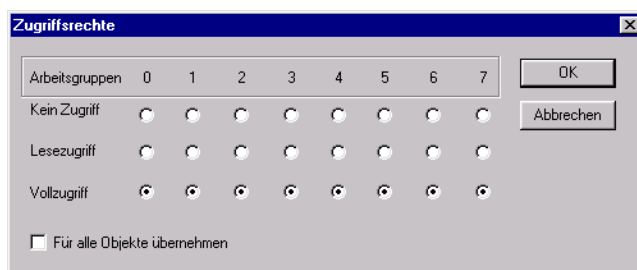
- Selektieren Sie das gewünschte Objekt und klicken Sie auf **Öffnen**, um das Objekt in sein Bearbeitungsfenster zu laden.
- Diese Möglichkeit wird bei der Objektart Ressourcen nur für globale Variablen unterstützt.
- Vor allem in Projekten mit vielen Objekten kann diese Funktion sehr hilfreich sein.

6.2.4.7

Zugriffsrechte

Symbol: -	Menübefehl: Projekt→Objekt Zugriffsrechte	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie das Dialogfeld zur Vergabe der Zugriffsrechte der verschiedenen Arbeitsgruppen.



Mitglieder der Arbeitsgruppe 0 (Administratoren) können nun für jede Arbeitsgruppe individuell Zugriffsrechte vergeben.

Folgende Einstellungen sind 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 das Kontrollkästchen **Für alle Objekte übernehmen** aktiviert ist, auf sämtliche Bausteine, Datentypen, Visualisierungen und Ressourcen des Projekts.

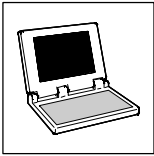
Die Zuordnung zu einer Arbeitsgruppe erfolgt beim Öffnen des Projektes durch eine Passwortabfrage, sofern ein Passwort für die Arbeitsgruppe 0 vergeben wurde.

6.2.4.8

Eigenschaften

Symbol: -	Menübefehl: Projekt→Objekt Eigenschaften	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl steht nur zur Verfügung, wenn eine globale Variablenliste im Objekt Organizer markiert ist. Bei der Neuerstellung einer globalen Variablenliste wird dieser Dialog mit dem Menübefehl **Objekt→Einfügen** geöffnet. Im Objekt Organizer muß der Ordner Globale Variablen markiert sein.



6.2.4.9 Instanz öffnen

Symbol: -	Menübefehl: Projekt→Instanz öffnen	Tastaturbefehl: -
-----------	---	-------------------

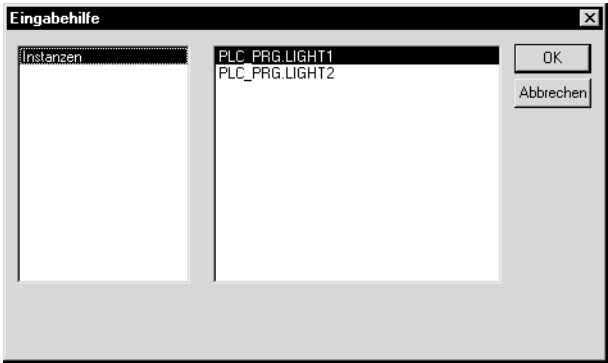
Mit diesem Befehl können Sie im Online-Modus Instanzen von Funktionsblöcken öffnen und anzeigen lassen. Mit einem Doppelklick auf den Funktionsblock im *Object Organizer* zu einem Auswahldialog, der die Instanzen des Funktionsblocks und die Implementation auflistet. Wählen Sie die gewünschte Instanz oder Implementation und bestätigen Sie mit **OK**, um sich die Liste anzeigen zu lassen.



Hinweis!

Der Funktionsblock, dessen Instanz geöffnet werden soll, muß zuerst im *Object Organizer* ausgewählt werden, bevor Sie diesen Befehl ausführen!

Im Dialogfeld *Eingabehilfe* können Sie dann die gewünschte Instanz dieses Funktionsblockes auswählen.



Tip!

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

6.2.4.10 Aufrufbaum ausgeben

Symbol: -	Menübefehl: Projekt→Aufrufbaum ausgeben	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie ein Fenster, in dem der Aufrufbaum des im *Object Organizer* ausgewählten Objekts dargestellt wird.

Beispiel: Aufrufbaum für Objekt **PLC_PRG** der Ampelanlage.



Das Projekt muß hierfür übersetzt sein. Siehe **Projekt→Alles Übersetzen..**(📄 6-8)

- Der Aufrufbaum beinhaltet sowohl Aufrufe von Bausteinen, als auch Verweise auf verwendete Datentypen.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.2.4.11 Querverweisliste ausgeben

Symbol: -	Menübefehl: Projekt→Querverweisliste ausgeben	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie ein Dialogfeld, das die Ausgabe aller Verwendungsstellen zu einer Variable, einer Adresse oder einem Bausteins ermöglicht.

Auswahl

Kategorie: Variable

Name: gnVar01

Querverweise

Gehe zu

Abbrechen

Ins Meldungsfenster

Baustein	Variable	Adresse	Bereich	Zugriff
POE_01 (1)	gnVar01		Global	Lesen
POE_01 (1)	gnVar01		Global	Schreiben
POE_02 (1)	gnVar01		Global	Lesen
POE_02 (1)	gnVar01		Global	Schreiben
POE_03 (1)	gnVar01		Global	Lesen
POE_03 (1)	gnVar01		Global	Schreiben
PLC_PRG (1)	gnVar01		Global	Lesen
PLC_PRG (1)	gnVar01		Global	Schreiben

Das Projekt muß hierfür übersetzt sein. Siehe **Projekt→Alles Übersetzen**

1. Wählen Sie zuerst die Kategorie **Variable**, **Adresse** oder **Baustein**.
2. Geben Sie dann den Namen des gewünschten Elements ein.
Um alle Elemente der eingestellten Kategorie zu erhalten, geben Sie bei *Name* * ein.
3. Klicken Sie auf **Querverweise**, um eine Liste aller Verwendungsstellen zu erhalten.

Es wird neben dem Baustein und der Zeilen- bzw. Netzwerknummer, der Variablenname und die eventuelle Adressenanbindung angegeben. Die Spalte *Bereich* gibt die globale oder lokale Variable an. Die Spalte *Zugriff* zeigt wie auf die jeweilige Variable zugegriffen wird.

- Wenn Sie eine Zeile der Querverweisliste markieren und auf **Gehe zu** klicken 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 Ausführen zu erleichtern, können Sie durch Klicken auf **Ins Meldungsfenster** die aktuelle Querverweisliste ins Meldungsfenster übernehmen und von dort zum jeweiligen Baustein wechseln.



6.2.4.12 **Unbenutzte Variablen ausgeben**

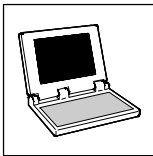
Symbol: -	Menübefehl: Projekt→Unbenutzte Variablen ausgeben	Tastaturbefehl: -
--------------	--	----------------------

Mit diesem Befehl wird eine Liste im Meldungsfenster mit allen Variablen ausgegeben, die zwar im Projekt deklariert wurden, aber nirgendwo angewendet werden.

```
-----Überprüfen: Unbenutzte Variablen-----
WARNUNG: Seit der letzten Übersetzung sind Änderungen am Projekt vorgenommen
count (10): summe
[Bibliotheksojekt] (2): g_fScaleFunctionReal
[Bibliotheksojekt] (3): g_byScaleFunctionByte
[Bibliotheksojekt] (4): g_siScaleFunctionSint
[Bibliotheksojekt] (5): g_nScaleFunctionInt
[Bibliotheksojekt] (6): g_wScaleFunctionWord
[Bibliotheksojekt] (7): g_dnScaleFunctionDint
[Bibliotheksojekt] (8): g_dwScaleFunctionDword
[Bibliotheksojekt] (9): g_bScaleFunctionBool
[Bibliotheksojekt] (10): g_bScaleFunctionError
```

Das Projekt muß hierfür übersetzt sein. Siehe **Projekt→Alles Übersetzen**

- Wenn Sie eine Variable markieren und auf **Gehe zu** klicken oder einen Doppelklick auf die Variable ausführen, wechseln Sie in das jeweilige Objekt, in dem die Variable deklariert wurde.
- Gibt es in dem Projekt keine unbenutzten Variablen, wird Ihnen dies in einer Meldung angezeigt.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.3 Arbeiten im Online-Modus

Die in diesem Kapitel beschriebenen Befehle sind über das Menü **Online** nach dem Einloggen verfügbar. Die Ausführung einiger Befehle ist abhängig vom aktiven Editor und von dem ausgewählten Zielsystem.

6.3.1 Befehle im Menü "Online"

6.3.1.1 Einloggen

Symbol: 	Menübefehl: Online→Einloggen	Tastaturbefehl: <Alt>+<F8>
---	-------------------------------------	---

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, wird es jetzt übersetzt (entsprechend **Projekt→Übersetzen**). Treten beim Übersetzen Fehler auf, wechselt das DDS nicht in den Online-Modus.

Fehlermeldungen beim Einloggen

Wenn die Fehlermeldung "Das gewählte Steuerungsprofil entspricht nicht dem des Zielsystems..."

- überprüfen Sie, ob das in den Zielsystemeinstellungen (Ressourcen) eingestellte Zielsystem mit dem in den **Online→Kommunikationsparameter** eingestellten Parameter, übereinstimmt.
- Überprüfen Sie die Verbindung zwischen PC und Zielsystem.
- Nach korrektem Einloggen steht in der Statuszeile **Online**.

Wenn die Fehlermeldung "Das Programm wurde geändert! Soll das neue Programm geladen werden?" erscheint, paßt das aktuelle Projekt im Editor nicht zu dem derzeit in der Steuerung geladenen Programm (bzw. zu dem gestarteten Simulationsprogramm).

- Monitoring und Debugging sind in diesem Fall nicht möglich.
- Bestätigen Sie mit **Nein**, um in die Steuerung einzuloggen. Das Projekt wird aber in der Steuerung nicht geladen.
- Bestätigen mit **Ja**, um das aktuelle Projekt in die Steuerung laden.

6.3.1.2 Ausloggen

Symbol: 	Menübefehl: Online→Ausloggen	Tastaturbefehl: <Strg>+<F8>
---	-------------------------------------	--

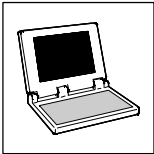
Die Verbindung zur Steuerung bzw. das Simulationsprogramm wird beendet und in den Offline-Modus gewechselt.

6.3.1.3 Laden

Symbol: -	Menübefehl: Online→Laden	Tastaturbefehl: -
-----------	---------------------------------	-------------------

Dieser Befehl führt einen Download des Programms in die PLC durch.

- Beachten Sie, daß beim Einloggen kein Download des Programms erfolgt, wenn das Projekt im DDS identisch mit dem Projekt in der PLC ist.



6.3.1.4 Start

Symbol: 	Menübefehl: Online→Start	Tastaturbefehl: <F5>
---	---------------------------------	-----------------------------------

Dieser Befehl startet die Abarbeitung des Anwenderprogramms in der PLC bzw. in der Simulation. In der Statuszeile steht **LÄUFT**, wenn korrekt eingeloggt und gestartet wurde.

Der Befehl kann in den folgenden Fällen ausgeführt werden:

- Nachdem das Anwenderprogramm in der Steuerung mit dem Befehl **Online→Stop** gestoppt wurde.
- Wenn das Anwenderprogramm auf einem Breakpoint steht.
- Wenn der Menübefehl **Online→Einzelzyklus** ausgeführt wurde.

6.3.1.5 Stop

Symbol: 	Menübefehl: Online→Stop	Tastaturbefehl: <Umschalt>+<F8>
---	--------------------------------	--

Dieser Befehl stoppt die Abarbeitung des Anwenderprogramms in der PLC bzw. in der Simulation zwischen zwei Zyklen.

- Wird das Programm gestoppt, wird in der PLC Quickstop (QSP) gesetzt.
- Benutzen Sie den Befehl **Online→Start**, um die Programmabarbeitung fortzusetzen.

6.3.1.6 Reset

Symbol: 	Menübefehl: Online→Reset	Tastaturbefehl: -
---	---------------------------------	-------------------

Dieser Befehl setzt Variablen, die mit einem bestimmten Wert initialisiert wurden, auf diesen Wert zurück.

- Alle anderen Variablen werden auf einen Standardwert zurückgesetzt (Integer-Zahlen beispielsweise auf 0).
- Es erfolgt eine Sicherheitsabfrage, bevor alle Variablen zurückgesetzt werden.

Folgende Unterschiede bestehen zwischen dem Befehl **Online→Reset** gegenüber dem Netzabschalten bzw. Aus- Einschalten der Steuerung.

- Variablen auf Codestellen werden nur bei einem **Reset** auf ihren Initialisierungswert zurückgesetzt. Sie behalten beim Netzschalten den Wert, der mit C0003 abgespeichert wurde.
- RETAIN-Variablen behalten ab den Betriebssystemen Servo PLC 6.0 und Drive PLC 6.0 bei einem **Reset** und bei einem Netzschalten ihren Wert.
In den Vorgängerversionen werden sie bei **Reset** neu initialisiert.



Achtung!

Ab den Betriebssystemen Servo PLC 6.0 und Drive PLC 6.0 werden RETAIN-Variablen nicht initialisiert.

Benutzen Sie den Befehl **Online→Start**, um die Programmabarbeitung erneut zu starten.

6.3.1.7 Reset (Kalt)

Symbol: -	Menübefehl: Online→Reset (Kalt)	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl setzt alle Variablen, inklusive der RETAIN-Variablen, auf den Initialisierungswert zurück. Nur die persistenten Variablen behalten Ihren Wert, den sie vor dem Reset (Kalt) hatten.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.3.1.8 Reset (Ursprung)

Symbol: -	Menübefehl: Online→Reset (Ursprung)	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl setzt alle Variablen, inklusive der RETAIN- und PERSISTANT, auf den Initialisierungswert zurück. Das Anwenderprogramm auf die Steuerung wird gelöscht und die Steuerung wird in den Urzustand zurückversetzt. Ab den Betriebssystemen Servo PLC 6.0 und Drive PLC 6.0.

6.3.1.9 Breakpoint an/aus

Symbol:	Menübefehl: Online→Breakpoint an/aus	Tastaturbefehl: <F9>
---------	---	----------------------

Dieser Befehl setzt einen Breakpoint an der aktuellen Position im aktiven Fenster.

- Sie können auch einen Breakpoint setzen, indem Sie mit der Maus auf das entsprechende Zeilennummernfeld klicken.
- Ist an der aktuellen Position bereits ein Breakpoint gesetzt, wird dieser entfernt.
- Ein gesetzter Breakpoint wird durch ein Zeilennummern-/Netzwerknummernfeld bzw. Schritt mit hellblauer Hintergrundfarbe dargestellt.
- Wenn bei der Programmabarbeitung ein Breakpoint erreicht ist, stoppt das Programm und das entsprechende Feld wird mit einer roten Hintergrundfarbe dargestellt.
- Um das Programm fortzusetzen, verwenden Sie die Menübefehle **Online→Start**, **Online→Einzelschritt in** oder **Online→Einzelschritt über**.
- Zum Setzen bzw. Entfernen von Breakpoints können Sie auch den Menübefehl **Online→Breakpoint-Dialog** verwenden.



Warnung!

Wird die Programmabarbeitung aufgrund eines Breakpoints angehalten, tritt für die Steuerung das unter **Projekt→Ausnahmeverhalten** definierte Verhalten in Kraft.

Die Position, an der ein Breakpoint gesetzt werden kann, hängt vom jeweiligen Editor ab:

AWL, ST

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 können Sie auch auf das Zeilennummernfeld klicken.

FUP, KOP

Im FUP- und KOP-Editor wird der Breakpoint auf das aktuell markierte Netzwerk gesetzt.

- Zum Setzen bzw. Entfernen eines Breakpoints können Sie auch auf das Netzwerknummernfeld klicken.

AS

Im AS-Editor wird der Breakpoint auf den aktuell markierten Schritt gesetzt.

- Zum Setzen bzw. Entfernen eines Breakpoints können Sie Doppelklicken, während Sie die <Umschalt>-Taste gedrückt halten.



Hinweis!

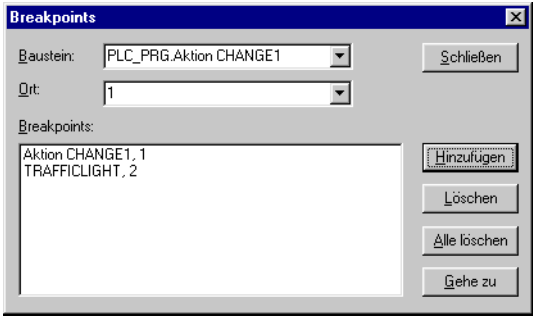
Werden Breakpoints gesetzt, ändert sich das Zeitverhalten der Abarbeitung.



6.3.1.10 Breakpoint-Dialog

Symbol: -	Menübefehl: Online→Breakpoint-Dialog	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl öffnet ein Dialogfeld zum Editieren von Breakpoints im gesamten Projekt.



- Das Dialogfeld zeigt zudem alle aktuell gesetzten Breakpoints an.

Breakpoint setzen

1. Wählen Sie im Kombinationsfeld **Baustein** den Baustein aus, in dem Sie den Breakpoint setzen möchten.
2. Wählen Sie im Kombinationsfeld **Ort** die Zeile bzw. das Netzwerk aus, wo Sie den Breakpoint setzen möchten.
3. Klicken Sie auf **Hinzufügen**, um den Breakpoint an die gewünschte Stelle zu setzen. Der Breakpoint wird in die Liste aufgenommen.

Breakpoint löschen

Um einen Breakpoint zu löschen, markieren Sie in der Liste der gesetzten Breakpoints den zu löschenden Breakpoint und klicken Sie auf **Löschen**.

- Klicken Sie auf **Alle löschen**, um alle Breakpoints zu löschen.

Zur Breakpoint-Position gehen

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 Breakpoint und klicken Sie auf **Gehe zu**.

6.3.1.11 Einzelschritt über

Symbol:	Menübefehl: Online→Einzelschritt über	Tastaturbefehl: <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, 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, geht das Programm zur nächsten Anweisung des aufrufenden Bausteins weiter.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

6.3.1.12 Einzelschritt in

Symbol: -	Menübefehl: Online→Einzelschritt in	Tastaturbefehl: <F8>
-----------	--	----------------------

Mit diesem Befehl 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, geht der Befehl zur ersten Anweisung des aufgerufenen Bausteins weiter.

In allen anderen Situationen verhält sich der Befehl genau wie **Online→Einzelschritt über**.

6.3.1.13 Einzelzyklus

Symbol: -	Menübefehl: Online→Einzelzyklus	Tastaturbefehl: <Strg>+<F5>
-----------	--	-----------------------------

Dieser Befehl führt einen einzelnen Steuerungszyklus aus und stoppt nach diesem Zyklus.

- **Online→Einzelzyklus** funktioniert nur bei der zyklischen Task. Die EVENT/INTERVALL-Task wird nicht berücksichtigt.
- **Online→Einzelzyklus** kann kontinuierlich wiederholt werden, um in einzelnen Zyklen fortzufahren.
- Einzelzyklus endet, wenn der Menübefehl **Online→Start** ausgeführt wird.

6.3.1.14 Werte schreiben

Symbol: -	Menübefehl: Online→Werte schreiben	Tastaturbefehl: <Strg>+<F7>
-----------	---	-----------------------------

Dieser Befehl verändert den Wert von einelementigen Variablen.

Doppelklicken Sie auf die Zeile, in der die Variable deklariert ist oder markieren Sie die Zeile und bestätigen Sie die <Eingabetaste>.

- Es öffnet sich ein Dialogfeld, in dem Sie für die Variable einen neuen Wert eingeben können. (Bei Booleschen Variablen wird der Wert getoggelt, ohne daß ein Dialogfeld erscheint.)

Der neue Wert (jetzt hellblau dargestellt) wird noch nicht in die Steuerung geschrieben.

- Sie können nun die Werte beliebig vieler Variablen verändern und anschließend auf einmal (Zykluskonsistent) in die Steuerung schreiben.

1. Definieren der Werte

- Bei nicht booleschen Variablen wird mit Doppelklick auf die Zeile, in der die Variable deklariert ist, ein Dialogfeld **Variable 'Zaehler' schreiben** (Zaehler als Beispiel) geöffnet. Dort kann der Wert eingegeben werden, der auf die Variable geschrieben werden soll.
- Bei booleschen Variablen wird durch Doppelklick auf die Zeile, in der die Variable deklariert ist, der Wert getoggelt (wechsel zwischen **TRUE**, **FALSE** und keinem neuen Wert) ohne das ein Dialog erscheint.

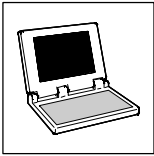
Der zum Schreiben vorgesehene neue Wert wird hellblau in spitzen Klammern hinter dem bisherigen Deklarationswert angezeigt.



Tip!

Die Anzeige der zu schreibenden Werte hat folgende Ausnahme.
Im FUP- und KOP- Editor steht ein Wert ohne spitze Klammern türkisfarben neben dem Variablennamen.

- Das Definieren der Werte kann für beliebig viele Variablen durchgeführt werden.
- Die Werte können zum Schreiben eingetragen, geändert und gelöscht werden.
- Die zum Schreiben vorgemerkten Werte werden in einer Schreibliste (Watchlist) gespeichert. Bis sie geschrieben oder gelöscht werden.



- 2. Schreiben der Werte
 - Menübefehl **Online→Werte schreiben**
 - Dialogfeld *Werte schreiben* Schaltfläche **Werte schreiben**

Wird **Werte schreiben** ausgeführt, werden alle in der Schreibliste enthaltenen Werte einmalig am Zyklusbeginn auf die entsprechenden Variablen der Steuerung geschrieben. Damit sind die Variablen aus der Schreibliste gelöscht.



Tip!

In der Ablaufsprache können die Einzelwerte, aus denen sich ein Transitions-Ausdruck zusammensetzt, nicht mit **Werte schreiben** verändert werden.

Beim Monitoring wird der Gesamtwert des Ausdrucks und nicht die Werte der Einzelvariablen dargestellt. Beispiel:

a AND b wird nur dann als TRUE dargestellt, wenn beide Variablen den Wert True haben.

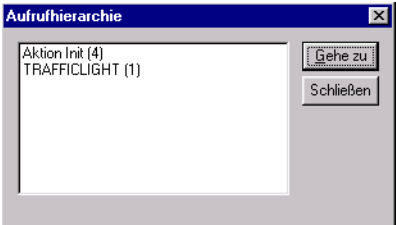
Im FUP wird in einem Ausdruck, der beispielsweise als Eingang eines Funktionsblocks verwendet wird, nur die erste Variable sichtbar (Monitor). Ein **Werte schreiben** ist nur für diese Variable möglich.

Mit **Werte schreiben** werden die veränderten Werte einmalig in die Steuerung geschrieben und können sofort wieder überschrieben werden.

6.3.1.15 Aufrufhierarchie

Symbol: -	Menübefehl: Online→Aufrufhierarchie	Tastaturbefehl: -
-----------	--	-------------------

Wenn die Simulation an einem Breakpoint stoppt, können Sie mit diesem Befehl ein Dialogfeld öffnen, in dem eine Liste der Bausteine, die sich momentan im Aufruf-Stack befinden, ausgegeben wird.



- Der erste Baustein ist stets **PLC_PRG**, denn hier beginnt die Abarbeitung.
- Der letzte Baustein ist stets der Baustein in dem die Abarbeitung momentan steht.

Wählen Sie einen Baustein aus und klicken Sie auf **Gehe zu**, um den ausgewählten Baustein in ein Fenster zu laden und die Zeile bzw. das Netzwerk, in dem sich die Abarbeitung befindet, anzuzeigen.

6.3.1.16 Simulation

Symbol: -	Menübefehl: Online→Simulation	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

Ist die **Simulation** ausgewählt, erscheint ein Haken ✓ vor dem Menüpunkt.

Im Simulationsmodus läuft das Benutzerprogramm unter Windows auf dem selben Rechner. Mit der Simulation testen Sie das Projekt. Die Kommunikation zwischen dem Rechner und der Simulation benutzt den Windows Message Mechanismus.

Wenn das Programm nicht im Simulationsmodus ist, läuft es auf der Steuerung. Die Kommunikation zwischen dem PC und der Steuerung läuft über einem Gateway. Der Status dieses Flags wird mit dem Projekt gespeichert.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Einschränkungen bei der Simulation

Sollen bei der Simulation alle Tasks berücksichtigt werden, müssen die Intervallzeiten entsprechend angepaßt werden.
Folgende Projekte sind bei der Simulation eingeschränkt bzw. können nicht simuliert werden.

- Lenze FB's
- Systembausteine

6.3.1.17 Kommunikationsparameter

Symbol: -	Menübefehl: Online→Kommunikationsparameter	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl können Sie Kommunikationskanäle zur Zielhardware aufbauen und administrieren.

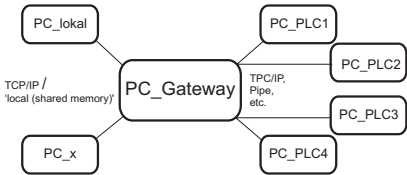


Hinweis!

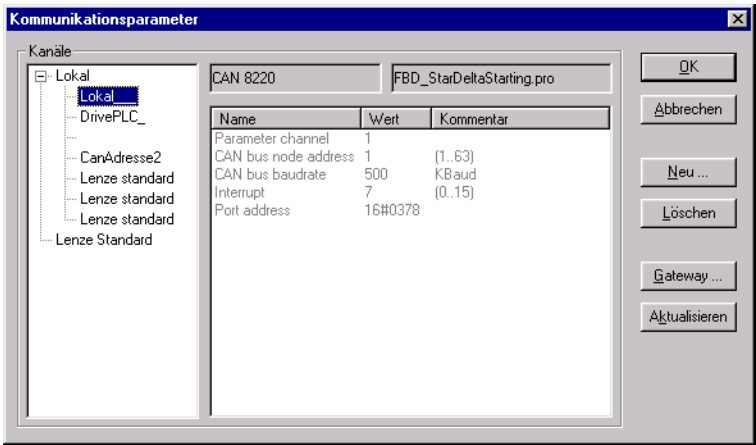
Bei der Verwendung des OPC- oder DDE-Servers müssen in dessen Konfiguration die selben Kommunikationsparameter eingestellt sein.

Für die praktische Nutzung werden zur Zeit folgende Kanäle zum Zielsystem angeboten:

- **Local-Gateway**
- **Remote-Gateway**
Über ein TCP/IP-Netzwerk auf einen Remote-Gateway-PC mit paralleler Schnittstelle und Dongle.



Das Dialogfeld *Kommunikationsparameter* ist in vier Bereiche untergliedert:

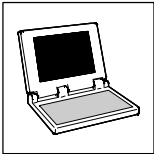


1. Kanäle

Hier werden die vom lokalen Host ausgehenden Kommunikationskanäle angezeigt sowie die zu bearbeitenden selektiert.

2. Kommunikationstreiber-Anzeige

Der aktuell geladene Kommunikationstreiber (z.B. "CAN 8220 | JumpingLEDs.pro").



3. Kommunikations-Parameter

Die für den aktuell geladenen Kommunikationstreiber eingestellten Parameter.

4. Schaltflächen

OK	Verlassen mit Parameterübernahme
Abbruch	Verlassen ohne Parameterübernahme
Neu...	Anlegen eines neuen Kommunikationskanals, z. B. über die Parallele Schnittstelle/Systembus-Dongle
Löschen	Entfernen eines im Bereich Kanäle selektierten Kanals
Gateway	Anlegen eines neuen Kommunikationskanals zu einem Remote-Gateway-PC
Aktualisieren	Überprüfen der Parameter bzw. Aktualisieren der Anzeige

Anlegen eines Kommunikationskanals über den Systembus-Dongle

Die praktisch am häufigsten benutzte Kommunikationsvariante ist die über den an der Parallelen Schnittstelle angeschlossenen Systembus-Dongle am lokalen PC.

Um eine Verbindung zu einem bestimmten Zielsystem (z. B. zur **9300 Servo PLC**) innerhalb eines CAN-Busnetzwerkes herzustellen, müssen Ihnen zwei Parameter bekannt sein:

- Die Geräteadresse dieses Gerätes am Bus (CAN bus node adress). Die Geräteadresse ist in Codestelle C0350 des Zielsystems gespeichert und muß mit der Einstellung des Kommunikationskanals übereinstimmen.
- Die am Bus eingestellte Baudrate (CAN bus baudrate). Die Geräteadresse ist in Codestelle C0351 des Zielsystems gespeichert und muß mit der Einstellung des Kommunikationskanals übereinstimmen.



Hinweis!

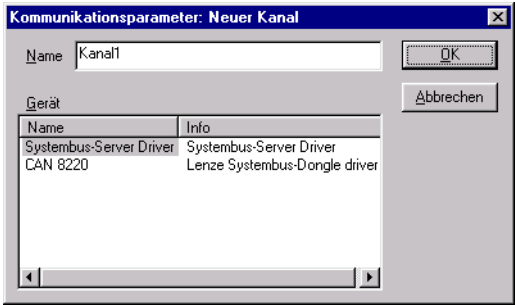
Standardmäßig ist von Lenze sowohl im DDS als auch im Zielsystem die Baudrate 500 KBAud eingestellt.

Kanalerstellung mit Default-Parametern:

Selektieren Sie im Dialogfeld *Kommunikationsparameter* den Eintrag **localhost→Lenze standard** und bestätigen Sie mit **OK**.

Anlegen eines Kommunikationkanals mit dem Lenze CAN-Treiber:

1. Klicken Sie auf **Neu**, um einen neuen localhost-Kanal zu konfigurieren.
2. Selektieren Sie im Dialogfeld *Kommunikationsparameter: Neuer Kanal* den Eintrag **CAN 8220**:



3. Bestätigen Sie die Auswahl mit **OK**.
4. Selektieren Sie den neu erstellten Kanal und tragen Sie im Bereich **Kommunikationsparameter** die ermittelten/gewünschten Parameter durch Anklicken des entsprechenden Eintrags und Auswahl der möglichen Parameter mit Hilfe der Pfeiltasten ein.
5. Übernehmen Sie die Einstellungen mit **OK**.
(Dadurch wird der Kommunikationstreiber GATEWAY.EXE parametrier und gestartet.)

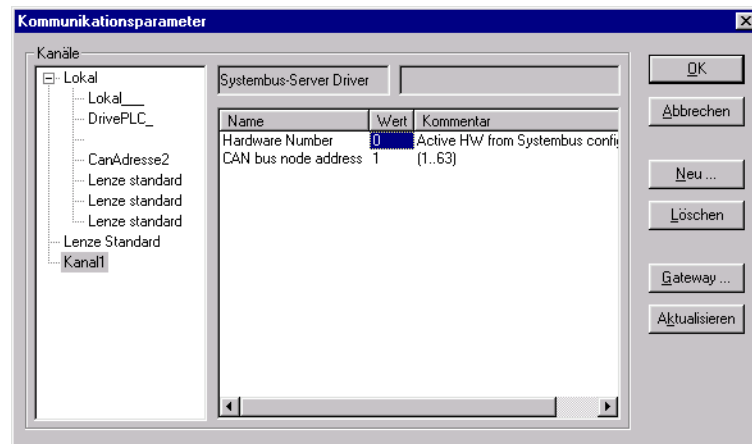


Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Anlegen eines Kommunikationskanals mit dem OPC Systembus-Treiber

1. Klicken Sie auf **Neu**, um einen neuen Kanal zu konfigurieren.
2. Selektieren Sie im Dialogfeld *Kommunikationsparameter: Neuer Kanal*, im Listenfeld **Gerät** den Eintrag **Systembus-Server Driver**.
3. Bestätigen Sie die Auswahl mit **OK**.



4. Mit dem Eintrag **Hardware Number** können Sie einen Wert eingeben, um damit einen bestimmten Port zu selektieren. Ist der Wert 0 wird der Standard-Port verwendet. Mit dem Kommunikationstool **Systembuskonfigurator** können Sie folgendes ermitteln.
 - Anzahl der vorhandenen aktiven Ports
 - die jeweilige Nummer der Ports
 - den Standard-Port
5. Mit dem Eintrag **Can bus node address** können Sie eine Geräteadresse einstellen. Die Geräteadresse ist in der Codestelle C0530 des Zielsystems gespeichert. Sie muß mit der Geräteadresse des Kommunikationskanals übereinstimmen.
6. Übernehmen Sie die Einstellungen mit **OK**. Der Kommunikationstreiber GATEWAY.EXE wird parametrisiert und gestartet.

Anlegen eines Kommunikationskanals über ein TCP/IP-Netzwerk zu einem Remote-Gateway-PC

Es ist möglich, vom DDS aus über ein TCP/IP-Netzwerk auf einen entfernten Rechner (dem Remote-Gateway-PC) mit Systembus-Dongle zuzugreifen.

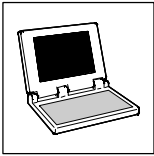
Sie können dadurch alle Funktionen, die Ihnen mit einem direkten CAN-Bus-Zugang zur Verfügung stehen würden (Programm-Download, Monitoring, usw.), über das Netzwerk ausführen.

Voraussetzung hierfür ist, daß der Kommunikationskanal vom Gateway-Rechner zum Zielsystem am Gateway-Rechner richtig eingerichtet ist und der Kommunikationstreiber GATEWAY.EXE auf dem Remote-Gateway-PC läuft (siehe "Anlegen eines Kommunikationskanals über den Systembus-Dongle").

- Eine Remote-Veränderung der Kommunikationsparameter auf dem Gateway-PC über das TCP/IP-Netz ist z. Z. nicht möglich.
- Um eine Verbindung zu einem Gateway-PC herzustellen, muß die IP-Adresse des Remote-PCs im Netzwerk bekannt sein:

Ermittlung der IP-Adresse des Gateway-PC

Die IP-Adresse erfahren Sie von Ihrem Netzwerk-Administrator (oder am Gateway-PC, indem Sie in einem DOS-Fenster das Programm "IPCONFIG" aufrufen).



Kanalerstellung zum Gateway-PC über ein TCP/IP-Netzwerk

- 1. Selektieren Sie im Dialogfeld *Kommunikationsparameter* den Eintrag **lokal**.
- 2. Klicken Sie auf **Gateway**, um das Dialogfeld *Kommunikationsparameter: Gateway* zu öffnen:



- 3. Tragen Sie die IP-Adresse des Gateway-PC ein und bestätigen Sie mit **OK**.

Im Bereich **Kanäle** wird die IP-Adresse des Gateway-PC angezeigt und im Bereich **Kommunikations-Parameter** die auf dem Rechner eingestellten Parameter.

- Sollte hinter der IP-Adresse “(Nicht verbunden)” angezeigt werden, so läuft der Kommunikationstreiber GATEWAY.EXE nicht auf dem Gateway-PC. Starten Sie ihn dann vom DDS auf dem Gateway-PC.

Aufheben eines Kommunikationskanals über ein TCP/IP-Netzwerk zu einem Remote-Gateway-PC

Um die Verbindung zu einem Remote- Gateway-PC aufzuheben und auf das lokale Gateway zurück-zukehren, gehen sie folgendermaßen vor:

- 1. Selektieren Sie im Dialogfeld *Kommunikationsparameter* im Bereich **Kommunikations-Parameter** die IP-Adresse.
- 2. Klicken Sie auf **Gateway**, um das Dialogfeld *Kommunikationsparameter: Gateway* zu öffnen.
- 3. Tragen Sie im Eingabefeld **Adresse** "localhost" ein und bestätigen Sie mit **OK**.

Kommunikationsaufbau fehlgeschlagen

Bei einem fehlgeschlagenen Versuch des Kommunikationsaufbaus zum Gateway-Server auf einem anderen Rechner. Folgende Punkte müssen bei einem korrekten Kommunikationsaufbau erfüllt sein.

- Das dreifarbige Symbol in der unteren Menüleiste zeigt den aktiven Gateway-Server an.
- Wurde die richtige IP-Adresse im Dialogfeld *Gateway: Kommunikationsparameter* eingetragen.
- Die lokale TCP/IP- Verbindung muß funktionieren.

6.3.1.18 Reglerfreigabe

Symbol: -	Menübefehl: Online→Reglerfreigabe	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl können Sie Reglerfreigabe im Antriebsregler durchführen.
Der Menüpunkt wird für Zielsysteme, die nicht über Reglersperre verfügen, deaktiviert.

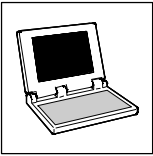
- Diese Funktion entspricht beim GDC der Taste **Start**.



Warnung!

Verwenden Sie die Befehle **Reglerfreigabe/Reglersperre** nicht für einen Not Aus/Not Stop über den PC, da diese Befehle nur verzögert zum Antriebsregler gelangen.

Im DDS werden diese Befehle am Ende einer Nachrichtenschlange angefügt, so daß sie im WorstCase-Fall erst einige Sekunden später im Antriebsregler eintreffen.



Drive PLC Developer Studio

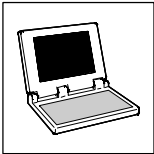
Arbeiten mit Projekten und Objekten

6.3.1.19 Reglersperre

Symbol: -	Menübefehl: Online→Reglersperre	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl können Sie Reglersperre im Antriebsregler durchführen.
Der Menüpunkt wird für Zielsysteme, die nicht über Reglersperre verfügen, deaktiviert.

- Diese Funktion entspricht beim GDC der Taste "Stop".



6.4 Logbuch

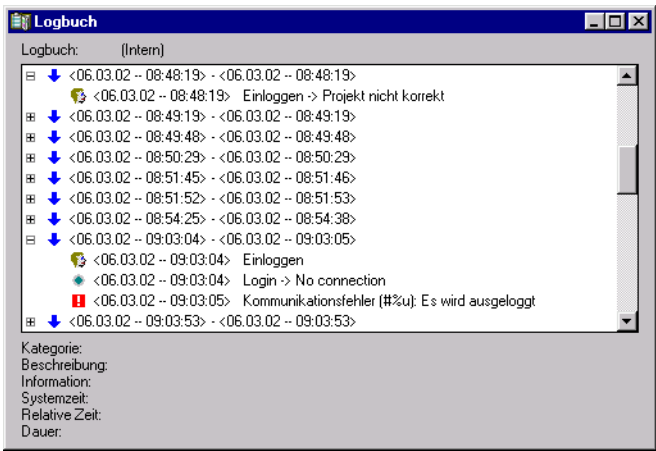
Das Logbuch speichert in chronologischer Reihenfolge die Aktionen, die während einer Online-Session passieren. Dazu wird zu jedem Projekt eine binäre Logdatei (*.log) angelegt. Dazu kann der Anwender Auszüge aus dem jeweiligen Projekt-Logbuch in einem externen Logbuch abspeichern.

6.4.1 Logbuch-Eigenschaften

6.4.1.1 Logbuch-Fenster

Das Logbuch-Fenster kann im Offline- und Online-Modus geöffnet werden. Im Online-Modus kann es als unmittelbarer Monitor dienen.

Symbol: -	Menübefehl: Fenster→Logbuch	Tastaturbefehl: -
-----------	-----------------------------	-------------------



Angaben zum Logbuch-Fenster

- Hinter Logbuch steht der Dateiname des momentan angezeigten Logbuchs. Falls es sich um das Logbuch des aktuellen Projekts handelt, wird im oberen Teil des Dialogfeldes [Intern] angezeigt.
- Die Einträge werden im Fenster dargestellt. Der neueste Eintrag wird unten angehängt.
- Es werden nur die Aktionen der Kategorien dargestellt, die im Menü: **Projekt→Optionen** Kategorie *Logbuch* Gruppenfeld **Filter** mit (✓) aktiviert sind.
- Unterhalb des Protokollfensters wird jeweils die zum selektierten Eintrag verfügbare Information angezeigt.
- Das Logbuch kann unter **Projekt→Optionen** Kategorie *Logbuch* mit dem Kontrollkästchen **Logbuch aktivieren** aktiviert bzw. deaktiviert werden.

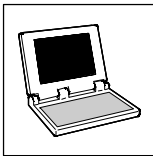
Kategorie

Der Logbuch-Eintrag besitzt vier Kategorien.

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

Beschreibung

Benutzeraktionen heißen wie ihre korrespondierenden Menübefehle. Alle anderen Aktionen sind englischsprachig und heißen ähnlich der zugehörigen `OnlineXXX()`-Funktion.



Drive PLC Developer Studio

Arbeiten mit Projekten und Objekten

Information

In diesem Feld werden Fehler beschrieben, die während einer Aktion auftreten. Treten keine Fehler auf, bleibt das Feld leer.

Systemzeit

Die momentane Systemzeit bei Beginn der Aktion.

Relative Zeit

Die relative Zeit zum Beginn der Online-Session.

Dauer

Die Dauer der Aktion in Millisekunden.

6.4.1.2 Logbuch-Menü

Wenn das Logbuch-Fenster den Eingabefokus besitzt, wird in der Menüleiste der Menübefehl **Logbuch** anstelle von **Einfügen** eingeblendet. **Extras** bleibt erhalten, hat aber keinen Inhalt.

6.4.1.3 Laden

Symbol: -	Menübefehl: Logbuch→Laden	Tastaturbefehl: -
-----------	----------------------------------	-------------------

Über den Windows-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.
- Die geladene Version wird durch das Projekt-Logbuch ersetzt, wenn
 - das Logbuch-Fenster geschlossen und danach wieder geöffnet wird.
 - eine neue Online-Session gestartet wird.

6.4.1.4 Speichern

Symbol: -	Menübefehl: Logbuch→Speichern	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

Dieser Befehl ist nur anwählbar, wenn das Projekt-Logbuch angezeigt wird. Ein Auszug des Projekt-Logbuchs kann mit diesem Befehl in eine externe Datei gespeichert werden. Die Sessions müssen ausgewählt werden um sie zu speichern.

Wird **OK** gedrückt erscheint wieder der Windows-Standarddialog zum Speichern der Datei.

6.4.1.5 Projekt-Logbuch anzeigen

Symbol: -	Menübefehl: Logbuch→Projekt-Logbuch anzeigen	Tastaturbefehl: -
-----------	---	-------------------

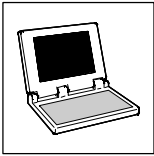
Dieser Befehl ist nur anwählbar, wenn aktuell ein externes Logbuch angezeigt wird. Er schaltet die Darstellung wieder auf das Projekt-Logbuch zurück.

6.4.1.6 Projekt-Logbuch speichern

Das Projekt-Logbuch wird automatisch in einer binären Datei `projectname.log` gespeichert. Unabhängig von einer Sicherung des Logbuchs in einer externen Datei.

Mit dem Menübefehl **Projekt→Optionen** Kategorie *Logbuch* kann ein Pfad zum speichern angegeben werden. Wird kein Pfad angegeben erfolgt die Ablage im gleichen Verzeichnis, in dem das Projekt abgespeichert wurde.

Unter dem Menübefehl **Projekt→Optionen** Kategorie *Logbuch* kann im Textfeld **Maximal Größe eines Projekt-Logbuchs** die Anzahl der Online-Sessions eingegeben werden. Wird die Anzahl überschritten, dann werden nach dem First in First out Prinzip die Älteren zugunsten der Neueren entfernt.



7 Editoren

7.1 Allgemeine Editierfunktionen

Die in diesem Kapitel beschriebenen Befehle sind in allen Editoren über das Menü **Bearbeiten** und zum Teil auch über das Kontextmenü im *Object Organizer* verfügbar.

Druckgrenzen

Im Menübefehl **Projekt→Optionen**, in der Kategorie *Arbeitsbereich* befindet sich das Kontrollkästchen **Druckbereiche anzeigen**. Dadurch kann der Druckbereich der einzelnen Editoren rot umrandet angezeigt werden. Dabei gelten die Vorgaben des eingestellten Druckers und die im Menübefehl **Datei→Einstellung Dokumentationen** ausgewählte Größe der Druckvorlage.

Ist kein Drucker oder Druckvorlage eingestellt, greifen Default-Einstellungen (Default.DFR und Standard Drucker).

Die horizontalen Druckgrenzen werden eingezeichnet, als wäre bei dem Dialogfeld *Einstellungen Dokumentation* das Kontrollkästchen **Neue Seite je Objekt** bzw. **Neue Seite je Unterobjekt** gewählt. Die untere Grenze wird nicht angezeigt.



Tip!

Um die exakte Anzeige des Druckbereiches zu gewährleisten, muß der Zoomfaktor auf 100% eingestellt werden.

7.1.1 Befehle im Menü “Bearbeiten”

7.1.1.1 Rückgängig

Symbol: -	Menübefehl: Bearbeiten→Rückgängig	Tastaturbefehl: <Strg>+<Z>
-----------	--	---

Dieser Befehl macht im aktuell geöffneten Editorfenster bzw. im *Object Organizer* die zuletzt 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, Visualisierungen und globalen Variablen und im Object Organizer.

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



Tip!

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 Rückgängig oder Wiederherstellen im *Object Organizer* muß dieser das aktive Fenster sein.

7.1.1.2 Wiederherstellen

Symbol: -	Menübefehl: Bearbeiten→Wiederherstellen	Tastaturbefehl: <Strg>+<Y>
-----------	--	---

Mit dem Befehl können Sie eine über **Bearbeiten→Rückgängig** rückgängig gemachte Aktion 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.



Drive PLC Developer Studio

Editoren

7.1.1.3 Ausschneiden

Symbol: 	Menübefehl: Bearbeiten→Ausschneiden	Tastaturbefehl: <Strg>+<X> <Umschalt>+<Entf>
---	--	---

Dieser Befehl entfernt die aktuelle Markierung aus dem Editor und legt sie in der Zwischenablage ab.

- Beim *Object Organizer* gilt dies analog für markierte Objekte, wobei nicht alle Objekte ausgeschnitten werden können, z. B. die Steuerungskonfiguration.



Hinweis!

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 Folge von Zeichen.
- Im FUP-, KOP- und CFC-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.

7.1.1.4 Kopieren

Symbol: 	Menübefehl: Bearbeiten→Kopieren	Tastaturbefehl: <Strg>+<C> <Strg>+<Einf>
---	--	---

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 für markierte Objekte, wobei nicht alle Objekte kopiert werden können, z. B. die Steuerungskonfiguration.



Tip!

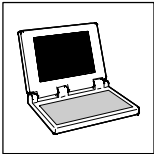
Beachten Sie, daß nicht alle Editoren das Kopieren unterstützen, und daß es in einigen Editoren eingeschränkt sein kann.

Die Form der Auswahl hängt vom jeweiligen Editor ab:

- In den Texteditoren (AWL, ST, Deklarationen) ist die Markierung eine Folge von Zeichen.
- Im FUP-, KOP- und CFC-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.



7.1.1.5 Einfügen

Symbol: 	Menübefehl: Bearbeiten→Einfügen	Tastaturbefehl: <Strg>+<V> <Umschalt>+<Einf>
---	--	---

- Dieser Befehl 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.



Tip!
Beachten Sie, daß nicht alle Editoren das Einfügen unterstützen, und daß es in einigen Editoren eingeschränkt sein kann.

- Inhalte aus der Zwischenablage können an verschiedenen Positionen eingefügt werden. Das Einfügen ist abhängig vom jeweiligen Editor:
- Bei den Texteditoren (AWL, ST, Deklarationen) ist die aktuelle Position die Position des blinkenden Cursors (eine kleine senkrechte Linie, die durch Mausklick positioniert werden kann).
 - Im FUP-, KOP- und CFC-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, 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-Editor können Sie ebenso die Befehle **Extras→Parallelzweig einfügen (rechts)** bzw. **Extras→Einfügen danach** benutzen.

7.1.1.6 Löschen

Symbol: -	Menübefehl: Bearbeiten→Löschen	Tastaturbefehl: <Entf>
-----------	---------------------------------------	-------------------------------------

- Löscht den markierten Bereich aus dem Editorfenster. Der Inhalt der Zwischenablage wird dabei nicht verändert.
- Beim *Object Organizer* gilt dies analog für markierte Objekte, wobei nicht alle Objekte gelöscht werden können, z. B. die Steuerungskonfiguration.
- Die Form der Auswahl hängt vom jeweiligen Editor ab:
- In den Texteditoren (AWL, ST, Deklarationen) ist die Markierung eine Folge von Zeichen.
 - Im FUP-, KOP- und CFC-Editor ist die Auswahl eine Menge von Netzwerken, die jeweils durch eine gepunktetes Rechteck im Netzwerk-Zahlenfeld markiert sind.
 - Im AS-Editor ist die Auswahl ein Teil einer Schrittfolge, umgeben von einem gepunkteten Rechteck.
 - Im Bibliotheksverwalter ist die Auswahl der aktuell gewählte Bibliotheksname.



Drive PLC Developer Studio

Editoren

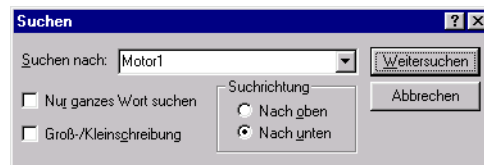
7.1.1.7

Suchen

Symbol: 	Menübefehl: Bearbeiten→Suchen	Tastaturbefehl: <Strg>+<F>
---	--------------------------------------	---

Mit diesem Befehl können Sie nach einer Zeichenfolge im aktuellen Editorfenster suchen.

Nach Auswahl des Befehls öffnet sich das Dialogfeld *Suchen*.



- Das Dialogfeld *Suchen* bleibt geöffnet, bis Sie auf **Abbrechen** klicken.

Suchen nach

Geben Sie im Eingabefeld **Suche nach** die zu suchende Zeichenfolge ein.

Nur ganzes Wort suchen

Wählen Sie aus, ob Sie den zu suchenden Text als ganzes Wort suchen wollen, oder auch als Teil eines Wortes.

Groß-/Kleinschreibung

Wählen Sie aus, ob bei der Suche die Groß-/Kleinschreibung beachtet werden soll.

Suchrichtung

Wählen Sie aus, ob die Suche ausgehend von der aktuellen Cursorposition nach oben oder nach unten erfolgen soll.

Suche starten

Klicken Sie auf **Weitersuchen**, um die Suche zu starten.

Die Suche 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 Ihnen das gemeldet.



Tip!

Die Suche kann mehrmals hintereinander durch erneutes Klicken auf **Weitersuchen** durchgeführt werden, bis der Anfang, bzw. das Ende des Inhalts des Editorfensters erreicht ist.

Beachten Sie, daß der gefundene Text vom Dialogfeld *Suchen* verdeckt sein kann.

7.1.1.8

Weitersuchen

Symbol: 	Menübefehl: Bearbeiten→Weitersuchen	Tastaturbefehl: <F3>
---	--	-----------------------------------

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

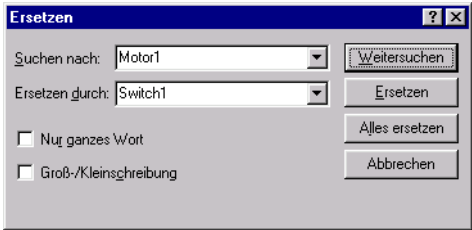


7.1.1.9 Ersetzen

Symbol: -	Menübefehl: Bearbeiten→Ersetzen	Tastaturbefehl: <Strg>+<H>
-----------	--	---

Mit diesem Befehl können Sie nach einer Zeichenfolge im aktuellen Editorfenster suchen und diese durch eine andere ersetzen.

Nach Auswahl des Befehls öffnet sich das Dialogfeld *Ersetzen*.



- Das Dialogfeld *Ersetzen* bleibt geöffnet, bis Sie auf **Abbrechen** klicken.

Suchen nach

Geben Sie im Eingabefeld **Suchen nach** die zu suchende Zeichenfolge ein.

Ersetzen durch

Geben Sie im Eingabefeld **Ersetzen durch** die Zeichenfolge ein, durch die der zu suchende Text ersetzt werden soll.

Nur ganzes Wort suchen

Wählen Sie aus, ob Sie den zu suchenden Text als ganzes Wort suchen wollen, oder auch als Teil eines Wortes.

Groß-/Kleinschreibung

Wählen Sie aus, ob bei der Suche die Groß-/Kleinschreibung beachtet werden soll.

Suche starten

Klicken Sie auf **Weitersuchen**, um die Suche zu starten.

Die Suchen beginnt an der aktuellen Position.

- Wenn die Textstelle gefunden wurde, dann wird diese markiert.
- Wenn die Textstelle nicht gefunden wurde, wird Ihnen das gemeldet.

Ersetzen

Klicken Sie auf **Ersetzen**, um die aktuelle Markierung durch den Text im Feld **Ersetzen durch** zu ersetzen.

Alles Ersetzen

Klicken Sie auf **Alles Ersetzen**, um alle Fundstellen des Suchtextes nach der aktuellen Position durch den Text im Feld **Ersetzen durch** zu ersetzen.



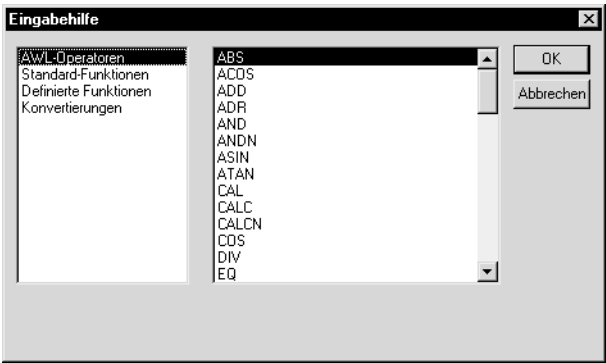
Drive PLC Developer Studio

Editoren

7.1.1.10 Eingabehilfe

Symbol: -	Menübefehl: Bearbeiten→Eingabehilfe	Tastaturbefehl: <F2>
-----------	-------------------------------------	----------------------

Mit diesem Befehl öffnen Sie das Dialogfeld *Eingabehilfe* zur Auswahl von möglichen Eingaben an der aktuellen Cursorposition im Editorfenster.



1. Wählen Sie in der linken Spalte die gewünschte Kategorie der Eingabe aus.
2. Markieren Sie in der rechten Spalte den gewünschten Eintrag.
3. Klicken Sie auf **OK**, um die Auswahl zu bestätigen.

Damit wird Ihre Auswahl an der aktuellen Cursorposition eingefügt.

- Die Eingabehilfe ist kontextsensitiv, d.h. die jeweils angebotenen Kategorien sind abhängig vom aktuellen Editorfenster und der aktuellen Cursorposition.
- Mit Argumenten
- Bei der Option wird beim Einfügen des gewählten Elements die zu übergebenden Argumente mit angegeben.

Beispiele:

Auswahl von Funktionsblock fu1, der die Eingangsvariable var_in definiert hat:

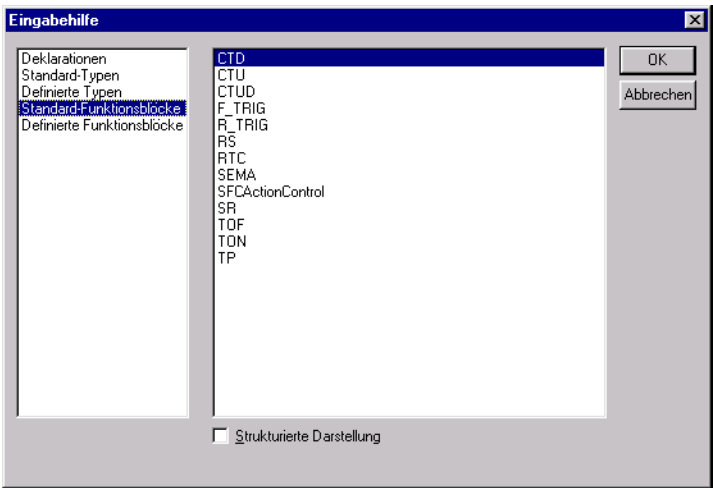
fu1(var_in:=);

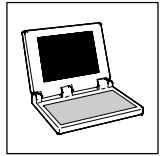
Einfügen von Funktion func1, die als Übergabeparameter var1 und var2 braucht:

func1(var1, var2);

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

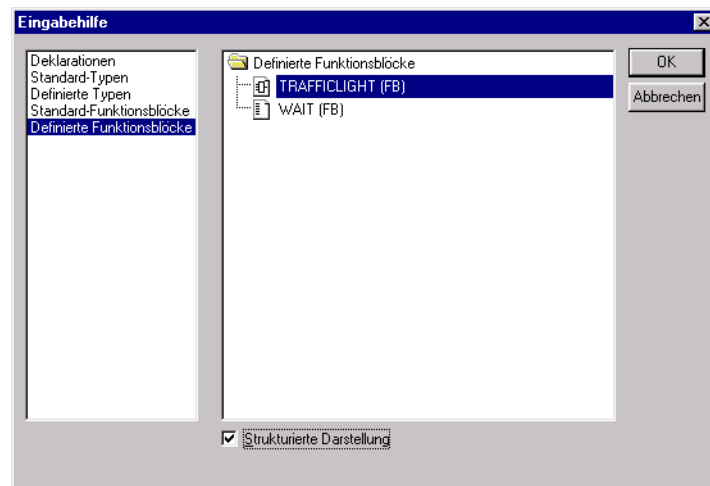
Nicht-Strukturierte Darstellung





Die Bausteine, Variablen oder Datentypen in jeder Kategorie sind linear alphabetisch sortiert. An manchen Positionen (z. B. in der Watchliste) werden mehrstufige Variablennamen benötigt. Der Dialog zur Eingabehilfe zeigt dann eine Liste aller Bausteine sowie einen einzelnen Punkt für die globalen Variablen. Bausteinnamen werden durch ein Punkt abgeschlossen. Wird ein Baustein markiert, öffnet sich die Liste der zugehörigen Variablen. Liegen Instanzen und Datentypen vor, kann weiter aufgeklappt werden. Durch **OK** wird die zuletzt markierte Variable übernommen.

Strukturierte Darstellung



Hier werden die Bausteine, Variablen oder Datentypen hierarchisch geordnet. Das ist möglich für

- Standard-Programme
- Standard-Funktionen
- Standard-Funktionsblöcke
- Standard-Typen
- 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 der obersten Stelle eingefügt und die jeweilige Hierarchie wie im Bibliotheksverwalter dargestellt.

Die Ein- und Ausgangsvariablen von Funktionsblöcken, die als lokale oder globale Variable deklariert sind, befinden sich in der Kategorie Lokale Variable bzw. Globale Variable unterhalb des Instanznamens als Liste.

- Inst_TP.ET
- Inst_TP.IN

Inst_TP kann wie ein Explorer-Verzeichnis expandiert werden.

Mit Argumenten

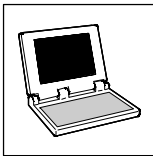
Ist das Kontrollkästchen markiert, werden bei den Textsprachen ST und AWL und bei der Taskkonfiguration der Instanzname und die Eingangsparameter des Funktionsblocks eingefügt.

Bei der Auswahl Inst (DeklarationInst:TON;) wird

Inst(IN:=, PT:=, Q=>, ET=>); eingefügt.

: = Funktionsblock-Eingänge beschalten

=> Funktionsblock-Ausgänge beschalten



Drive PLC Developer Studio

Editoren

Ist das Kontrollkästchen nicht markiert, 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 den Funktionsblockinstanzen dargestellt.

Für Aufzählungen werden die einzelnen Werte unter dem entsprechenden Typ aufgelistet. Folgende Reihenfolge wird eingehalten:

Aufzählungen aus Bibliotheken, aus Datentypen, lokale Aufzählungen aus Bausteinen.



Hinweis!

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

7.1.1.11 Nächster Fehler

Symbol: -	Menübefehl: Bearbeiten → Nächster Fehler	Tastaturbefehl: <F4>
-----------	--	-----------------------------------

Nach dem fehlerhaften Übersetzen eines Projektes kann mit diesem Befehl der nächste Fehler angezeigt werden.

- Es wird das entsprechende Editorfenster aktiviert und die fehlerhafte Stelle markiert.
- Gleichzeitig wird im Meldungsfenster die zugehörige Fehlermeldung markiert.
- Beim schrittweisen Durchgehen mit **F4**, können Warnungen ignoriert werden werden.
Projekt→**Optionen** Kategorie *Arbeitsbereich* Kontrollkästchen **F4 ignoriert Warnungen**.

7.1.1.12 Vorheriger Fehler

Symbol: -	Menübefehl: Bearbeiten → Vorheriger Fehler	Tastaturbefehl: <Umschalt>+<F4>
-----------	--	--

Nach dem fehlerhaften Übersetzen eines Projektes kann mit diesem Befehl der vorherige Fehler angezeigt werden.

- Es wird das entsprechende Editorfenster aktiviert und die fehlerhafte Stelle markiert.
- Gleichzeitig wird im Meldungsfenster die zugehörige Fehlermeldung markiert.
- Beim schrittweisen Durchgehen mit **<Umschalt>+<F4>** können Warnungen ignoriert werden.
Projekt→**Optionen** Kategorie *Arbeitsbereich* Kontrollkästchen **F4 ignoriert Warnungen**.

7.1.1.13 Makros

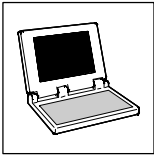
Symbol: -	Menübefehl: Bearbeiten → Makros	Tastaturbefehl: -
-----------	---	-------------------

Unter diesem Menüpunkt erscheinen alle Makros, die für das aktuelle Projekt vereinbart wurden. Sie können unter **Projekt**→**Optionen** Kategorie *Makros* neue Makros erstellen. Wird ein ausführbares Makro ausgewählt, öffnet sich das Dialogfeld *Makro Ausführen*. Es erscheint der Makroname und die aktuelle Befehlszeile. Mit **Abbrechen** kann der Makro gestoppt werden. Die aktuelle Befehlszeile wird aber noch abgearbeitet.

Eine entsprechende Meldung wird ins Message-Fenster und während des Online-Betriebs in das Logbuch ausgegeben.

Makro Ausführung durch Anwender abgebrochen.

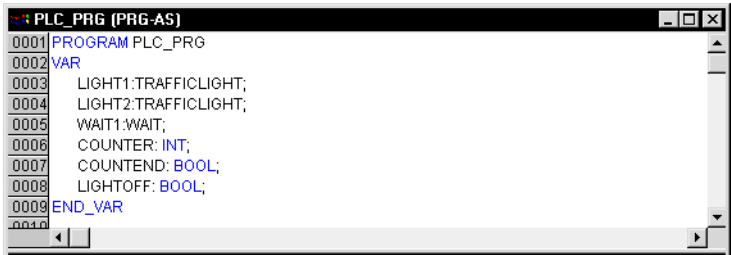
Makros können Online und Offline ausgeführt werden. Es werden aber nur die im jeweiligen Mode verfügbaren Befehle ausgeführt.



7.2 Deklarationseditor

Der Deklarationseditor wird verwendet für die Variablendeklaration von Bausteinen und globalen Variablen, zur Datentypendeklaration und im Watch- und Rezepturverwalter. Er verfügt über die Windows-Funktionalitäten und über die Funktionalitäten der IntelliMouse. Dazu muß aber der entsprechende Treiber installiert sein.

Der Überschreibemodus wird in der Statusleiste mit **ÜB** angezeigt. Mit der Taste **<Einf>** kann zwischen Überschreibe- und Einfügemodus gewechselt werden.



- Die Variablendeklaration wird durch das Verwendung von unterschiedlichen Farben unterstützt. Schlüsselwörter werden mit blauen Buchstaben geschrieben.
- Alle Editoren für Bausteine bestehen aus einem Deklarationsteil und einem Rumpf. Diese sind durch einen horizontalen Bildschirmteiler getrennt, den Sie bei Bedarf mit gedrückter linker Maustaste verschieben können.



Tip!

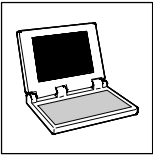
Die wichtigsten Befehle zum Deklarationseditor finden Sie auch im Kontextmenü (rechte Maustaste oder **<Umschalt>+<F10>**).

Deklarationen als Tabelle

Mit dem Menübefehl **Projekt→Optionen** in der Kategorie *Editork* können Sie mit markierten Kontrollkästchen **Deklarationen als Tabellen** Variablen statt mit dem Deklarationseditor als Tabelle editieren. (7-18)

	VAR	VAR_INPUT	VAR_OUTPUT	VAR_IN_OUT	CONSTANT	RETAIN	INFO
	Name	Adresse	Typ	Initial	Kommentar		
0001	TP2		TP				
0002	TP1		TP				
0003	btogg		BOOL				
0004	bNOTtogg		BOOL				
0005	bFillIn		BOOL				

- Diese Tabelle ist wie ein Karteikasten geordnet, in dem Registerkarten für Eingangsvariablen (VAR_INPUT), Ausgangsvariablen VAR_OUTPUT, lokale Variablen VAR und Ein-Ausgangsvariablen VAR_IN_OUT sind.
- Für jede Variable stehen Ihnen die Felder Name, Adresse, Typ, Initial und Kommentar zur Verfügung.



Drive PLC Developer Studio

Editoren

7.2.1 Deklarationsteil

Im Deklarationsteil eines Bausteins werden alle Variablen lokal deklariert.

Dies können

- Eingangsvariablen,
- Ausgangsvariablen,
- Ein-/Ausgangsvariablen,
- lokale Variablen,
- remanente Variablen,
- Konstanten
- und remanente Konstanten sein.

Die Deklarationssyntax orientiert sich am Standard der IEC 61131-3.

7.2.1.1 Deklarations Schlüsselworte

Symbol: -	Menübefehl: Einfügen→Deklarations Schlüsselworte	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie eine Liste aller Schlüsselwörter, die im Deklarationsteil eines Bausteins benutzt werden können.

- Nachdem ein Schlüsselwort ausgewählt und die Auswahl 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.

7.2.2 Eingangsvariablen

Zwischen den Schlüsselwörtern **VAR_INPUT** und **END_VAR** werden alle Variablen deklariert, die als Eingangsvariablen eines Bausteins dienen.

- Die Werte dieser Variablen können von der Aufrufstelle übergeben werden.

Beispiel:

```
VAR_INPUT
  in1:INT;      (* 1. Eingangsvariable *)
END_VAR
```

Mit Eingangsvariablen können Daten vom aufrufenden an den aufgerufenen Baustein übergeben werden.

7.2.3 Ausgangsvariablen

Zwischen den Schlüsselwörtern **VAR_OUTPUT** und **END_VAR** werden alle Variablen deklariert, die als Ausgangsvariablen eines Bausteins dienen.

- Die Werte dieser Variablen werden dem aufrufenden Baustein zurückgeliefert, dort können diese abgefragt und weiterverwendet werden.

Beispiel:

```
VAR_OUTPUT
  out1:INT;      (* 1. Ausgangsvariable *)
END_VAR
```




7.2.4 Ein-/Ausgangsvariablen

Zwischen den Schlüsselwörtern **VAR_IN_OUT** und **END_VAR** werden alle Variablen deklariert, die als Ein- und Ausgangsvariablen eines Bausteins dienen.



Hinweis!

Bei diesen Variablen wird direkt der Wert der übergebenen Variablen verändert ("Übergabe als Pointer", Call-by-Reference). Deshalb darf der Eingabewert für eine solche Variable keine Konstante sein. Aus diesem Grund können auch **VAR_IN_OUT** Variablen eines Funktionsblocks nicht von außen direkt über *Funktionsblockinstanz*, *Ein-/Ausgangsvariable* gelesen werden.

Beispiel:

```
VAR_IN_OUT
  inout1:INT;      (* 1. EinAusgangsvariable *)
END_VAR
```

7.2.5 Lokale Variablen

Zwischen den Schlüsselwörtern **VAR** und **END_VAR** werden alle lokalen Variablen eines Bausteins deklariert.

- Lokale Variable haben keine Verbindung nach außen, d. h., es kann nicht von außen auf sie geschrieben werden.

Beispiel:

```
VAR
  loc1:INT;      (* 1. Lokale Variable *)
END_VAR
```

7.2.6 Remanente Variablen

Remanente Variablen können ihren Wert über die übliche Programmlaufzeit hinaus behalten. Zu den remanenten Variablen gehören Retain-Variablen und persistente Variablen.

Retain-Variablen

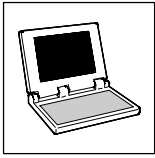
Zwischen den Schlüsselwörtern **VAR RETAIN** und **END_VAR** werden alle Retain-Variablen eines Bausteins deklariert.

- Retain-Variablen behalten nach einem Ausschalten der Steuerung bzw. nach Reset ihren Wert. Bei erneutem Start des Programms wird mit den gespeicherten Werten weitergearbeitet.
- In zwei Fällen werden die Retain-Variablen auf ihren Initialisierungswert zurückgesetzt.
 - Mit einem Programm-Download.
 - Im Online-Modus mit den Menübefehlen **Online→Reset (Kalt)** und **Online→Reset (Ursprung)**.

Persistente Variablen

Persistente Variablen liegen im Persistent-Speicher und behalten ihren Wert. Der Persistent-Speicher kann nur im Online-Modus mit dem Menübefehl **Online→Reset (Ursprung)** gelöscht werden.

Der Zugriff auf den Persistent-Speicher erfolgt über den Systembaustein **VAR_Persistent**. Im Object Organizer, Registerkarte **Ressourcen** können Sie die Steuerungskonfiguration öffnen. Mit dem Menübefehl **Einfügen→Unterelement anhängen→Var_Persistent...** hängen Sie den Systembaustein als Unterelement an.



Drive PLC Developer Studio

Editoren



Hinweis!

Der Systembaustein **VAR_Persistent** steht für die 9300 Servo PLC V6.3 zu Verfügung.

Anwendungsbeispiel:

Ein Betriebsstundenzähler, der nach einem Stromausfall weiterzählen soll.

- Alle anderen Variablen werden neu initialisiert, entweder mit den standard Initialwerten (0 bzw. FALSE) oder mit den definierten Initialwerten.

Beispiel:

```
VAR RETAIN
    rem1:INT;      (* 1. Remanente Variable *)
END_VAR
```

7.2.7 Konstanten, Typed Literals

Zwischen den Schlüsselwörtern **VAR CONSTANT** und **END_VAR** werden alle Konstanten deklariert.

- Konstanten 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 Liste möglicher Konstanten finden Sie auch im Kapitel Operanden. (📖 13-1)

7.2.8 Remanente Konstanten

Zwischen den Schlüsselwörtern **VAR CONSTANT RETAIN** und **END_VAR** werden alle remanenten Konstanten deklariert.

- Remanente Konstanten können lokal oder global deklariert werden.

Syntax:

```
VAR CONSTANT RETAIN
    <Bezeichner>:<Typ> := <Initialisierung>;
END_VAR
```

Beispiel:

```
VAR CONSTANT RETAIN
    Hochlauf:DINT:=10000;      (* Hochlaufzeit *)
END_VAR
```



7.2.9 Schlüsselwörter

- Schlüsselwörter sind in allen Editoren mit Großbuchstaben zu schreiben.
- Schlüsselwörter dürfen nicht als Variablennamen verwendet werden.

7.2.10 Bezeichner



Hinweis!

Nur die ersten 32 Zeichen sind signifikant!

Ein Bezeichner ist eine Folge von Buchstaben, Ziffern und Unterstrichen, die mit einem Buchstaben oder einem Unterstrich beginnen.

Ein Bezeichner einer Variablen darf

- keine Leerstellen und Umlaute enthalten,
- nicht doppelt deklariert werden
- und nicht identisch mit Schlüsselwörtern sein.

Des Weiteren gilt:

- Groß-/Kleinschreibung bei Variablen wird **nicht** beachtet.
(Beispiel: VAR1, Var1 und var1 sind keine unterschiedlichen Variablen)
- Unterstriche im Bezeichner sind signifikant.
(Beispiel: A_BCD und AB_CD werden als unterschiedliche Bezeichner interpretiert.)
- Mehrfach aufeinanderfolgende Unterstriche am Anfang eines Bezeichners oder in einem Bezeichner sind **nicht** erlaubt.

7.2.11 Variablendeklaration

Syntax:

`<Bezeichner> { AT <Adresse> } : <Typ> { := <Initialisierung> } ;`

Die Teile in geschweiften Klammern {} sind optional.

Alle Variablendeklarationen und Datentypen können Initialisierungen enthalten.

- Initialisierungen erfolgen mit dem Zuweisungsoperator ":=".
- Für Variablen von elementaren Typen sind diese Initialisierungen Konstanten.
- Die Default-Initialisierung ist für alle Deklarationen ist 0 bzw. FALSE.

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 können Sie den Kurzformmodus verwenden.
- 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.



Drive PLC Developer Studio

Editoren

Beachten Sie die Möglichkeit des automatischen Deklarierens.



Tip!

Beachten Sie für die Bezeichner von Variablen die Hinweise im Kapitel IEC 61131-3 Operanden. (13-4)

7.2.11.1 Typen für Variablendeklaration

Symbol: -	Menübefehl: Einfügen→Typen	Tastaturbefehl: -
-----------	----------------------------	-------------------

Mit diesem Befehl erhalten Sie eine Auswahl der möglichen Typen für eine Variablendeklaration.

- Die Liste erhalten Sie auch, wenn Sie die Eingabehilfe aufrufen.

Die Typen sind eingeteilt in folgende Kategorien:

- Standard-Typen (BOOL, BYTE, etc.)
- Definierte Typen (Strukturen, Aufzählungstypen, etc.)
- Standard-Funktionsblöcke (für Instanzdeklarationen)
- Definierte Funktionsblöcke (für Instanzdeklarationen)

Das DDS unterstützt sämtliche Standard-Typen der Norm IEC 61131-3.

7.2.12 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 Sie einer Adresse einen aussagekräftigeren Namen geben können, und daß Sie eine eventuelle Veränderung eines Ein- oder Ausgangssignals nur an einer Stelle (nämlich in der Deklaration) zu ändern brauchen.



Tip!

Beachten Sie, daß auf Variablen, die auf einen Eingang gelegt sind nicht schreibend zugegriffen werden kann.

Als weitere Einschränkung gilt, daß AT-Deklarationen nur für lokale und globale Variablen gemacht werden können, also nicht für Ein- und Ausgangsvariablen von Bausteinen.

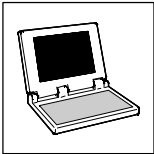
Beispiele:

```
bTrip_b AT %QX1.0.0: BOOL;  
bRechts_b AT %IX1.0.1: BOOL;  
bReversieren_b AT %MX2.2: BOOL;
```



Tip!

Wenn boolsche Variablen auf eine Byte-, Word- oder DWORD-Adresse gelegt werden, belegen sie ein Byte mit TRUE bzw. FALSE. Aber nicht nur das erste Bit nach dem Offset.



7.2.13 Syntaxcoloring

In allen Editoren werden Sie bei der Implementierung und der Variablendeklaration optisch unterstützt.

- Die farbige Darstellung des Textes hilft Ihnen, Fehler zu vermeiden bzw. schneller zu entdecken. (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	Boolsche Werte (TRUE , FALSE)
Rot	Fehlerhafte Eingabe (z. B. ungültige Zeitkonstante, Schlüsselwort kleingeschrieben, ...)
Schwarz	Variablen, Konstanten, Zuweisungsoperatoren, ...

7.2.14 Kurzformmodus

Der Deklarationseditor bietet Ihnen die Möglichkeit des Kurzformmodus.

- der Kurzformmodus 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 (①).
- Jede Konstante wird, je nach Typ der Deklaration, zu einer Initialisierung oder einer Stringlänge (②, ③).
- Eine Adresse (wie in **%MD12**) wird um das **AT**-Attribut erweitert (④).
- Ein Text nach einem Strichpunkt (;) wird ein Kommentar (③).
- Alle anderen Zeichen in der Zeile werden ignoriert (wie z. B. das Ausrufezeichen in ⑤).

Beispiele:

	Kurzform	Deklaration
①	A	A: BOOL ;
②	A B I 2	A, B: INT := 2;
③	ST S 2; String	ST: STRING (2); (* String *)
④	X %MD12 R 5; Number	X AT %MD12:REAL := 5.0; (* Number *)
⑤	B !	B: BOOL ;



Drive PLC Developer Studio

Editoren

7.2.15 Automatisch deklarieren

Wenn die Option **Automatisch deklarieren** im Dialogfeld *Optionen*, Kategorie *Editor*, aktiviert ist, erscheint in allen Editoren nach Eingabe einer noch nicht deklarierten Variablen ein Dialogfeld, in dem diese Variable deklariert werden kann.

- Im Kombinationsfeld **Klasse** wählen Sie die Klasse der Variablen aus.
- Im Textfeld **Name** geben Sie den Namen der Variablen ein. (Eingabefeld ist mit dem im Editor eingegebenen Variablennamen vorbelegt.)
- Im Textfeld **Typ** geben Sie den Variablentyp ein. (Eingabefeld ist mit dem Typ **BOOL** vorbelegt.)
- Mit der Schaltfläche ... rufen Sie die Eingabehilfe zur Auswahl aller möglichen Typen auf.
- Im Textfeld **Initialwert** können Sie der Variablen einen Wert zuweisen, ansonsten wird der Standardinitialwert verwendet.
- Mit den Kontrollkästchen **CONSTANT** (Konstante) und **RETAIN** (Batteriepufferung) können Sie definieren, ob es sich um eine Konstante und/oder eine remanente Retain-Variable handelt.

Wird im Dialogfeld *Typ* die Variable **ARRAY** ausgewählt, erscheint ein Dialog mit dem die Array-Grenzen bestimmt werden. Im folgenden Bild wurde ein zwei-dimensionales Array vom Typ UDINT (unsigned double integer) erzeugt.

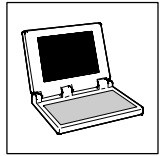
Dim.	Start	Ende
1	0	2
2	0	5
3		

Typ: UDINT

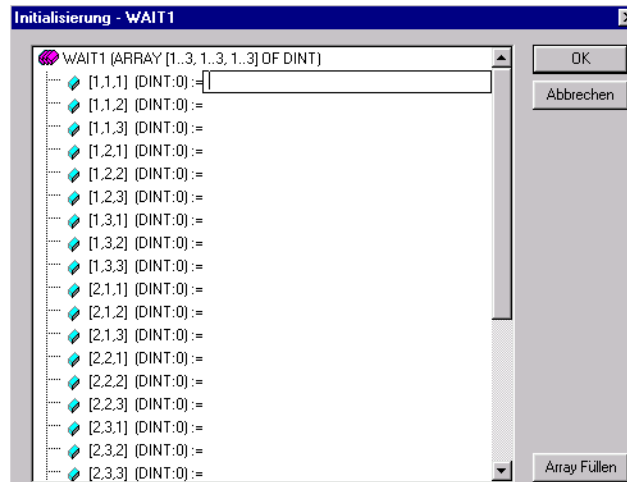
Mit diesem Baustein kann ein drei dimensionales Array erzeugt werden. Unter *Start und Ende* können die Grenzen von jeder Dimension eingegeben werden.

Mit der Schaltfläche ... wird der Datentyp des Arrays vereinbart. Wird der Dialog mit **OK** verlassen, erstellt das System eine Variablendeklaration im IEC-Format.

Beispiel: ARRAY[0..5, 0..2] OF INT



Der Dialog der Variablendeklaration ermöglicht im Dialogfeld *Initialwert* das Eintragen eines Initialwertes der zu deklarierenden Variablen. Ist diese ein Array oder eine Struktur, kann über die Schaltfläche ... oder <F2> (Cursor muß im Dialogfeld *Initialwert* stehen) ein spezieller Initialisierungsdialog geöffnet werden. Bei anderen Typen (INT, BYTE) kann der Eingabehilfedialog geöffnet werden.



Im Initialisierungsdialog für ein Array sind die Array-Elemente aufgelistet. Mit einem Mausklick hinter dem := öffnet sich ein Editierfeld zum Eintragen des Initialwertes.

Im Initialisierungsdialog für eine Struktur werden die einzelnen Komponenten in einer Baumstruktur dargestellt. Hinter dem Variablenname stehen in Klammern Typ und Default-Initialwert der Komponente. Danach folgt ein :=. Wenn Sie dahinter klicken, öffnet sich ein Editierfeld. Hier können Sie den gewünschten Initialwert eingeben. Ist die Komponente ein ARRAY, können Sie den Initialisierungsdialog durch Mausklick auf das Pluszeichen aufklappen. Initialwerte können eingetragen werden.

Wird der Initialisierungsdialog mit **OK** verlassen, erscheint im Feld *Initialwert* die Initialisierung des Arrays oder 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). Im Eingabefeld *Kommentar* können Informationen, z.B. Änderungen eingetragen werden. Mit <Strg> + <Eingabe> werden hier Zeilenumbrüche erzeugt.

Mit **OK** wird der Deklarationsdialog geschlossen und auf die Variable gemäß IEC-Syntax in den entsprechenden Deklarationseditor.



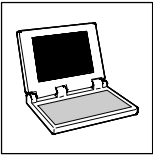
Tip!

Den Dialog zur Variablendeklaration erhalten Sie auch über **Bearbeiten→Variablendeklaration**.

Das Dialogfeld *Variablendeklaration* kann, wenn der Cursor auf einer Variablen steht, im Offline-Modus mit den aktuellen variablenbezogenen Einstellungen durch <Shift>+<F2> geöffnet werden.

7.2.16 Zeilennummern im Deklarationseditor

- Im Offline-Modus markieren Sie durch Klicken auf eine Zeilennummer die gesamte Textzeile.
- Im Online-Modus klappen Sie durch Klicken auf eine Zeilennummer die Variable in dieser Zeile auf bzw. zu, falls es sich um eine strukturierte Variable handelt.



Drive PLC Developer Studio

Editoren

7.2.17 Deklarationen als Tabelle

Ist das Kontrollkästchen **Deklarationen als Tabelle** im Dialogfeld *Optionen*, Kategorie *Editor* markiert, können Sie Variablen in der Tabelle statt mit dem Deklarationseditor als Tabelle editieren:

PLC_PRG (PRG-AS)								
	VAR	VAR_INPUT	VAR_OUTPUT	VAR_IN_OUT	CONSTANT	RETAIN	INFO	
	Name	Adresse	Typ	Initial	Kommentar			
0001	LIGHT1		TRAFFICLIGHT					
0002	LIGHT2		TRAFFICLIGHT					
0003	WAIT1		WAIT					
0004	COUNTER		INT					
0005	COUNTEND		BOOL					
0006	LIGHTOFF		BOOL					

- Diese Tabelle ist wie ein Karteikasten geordnet, in dem Registerkarten für Eingangs-, Ausgangs-, lokale- und Ein-/Ausgangsvariablen sind. Sie können durch Klicken auf ein Register die Registerkarte der jeweiligen Variablenklasse auswählen und die Variablen editieren.

Für jede Variable stehen Ihnen folgende Eingabefelder zur Verfügung:

Name	Bezeichner der Variablen
Adresse	Mit der Variablen verbundene Adresse (AT-Deklaration)
Typ	Variablentyp (Bei dem Instanzieren eines Funktionsblocks den Funktionsblocknamen eingeben.)
Initial	Initialisierungswert der Variablen (entsprechend dem Zuweisungsoperator ":=")
Kommentar	Kommentar

- Zwischen den beiden Darstellungsarten Deklarationseditor/Deklarationen als Tabelle können Sie jederzeit wechseln.
- Um eine neue Variable einzufügen, verwenden Sie den Befehl **Einfügen→Neue Deklaration**.

7.2.17.1 Neue Variable in Deklarationstabelle einfügen

Symbol: -	Menübefehl: Einfügen→Neue Deklaration	Tastaturbefehl: -
-----------	--	-------------------

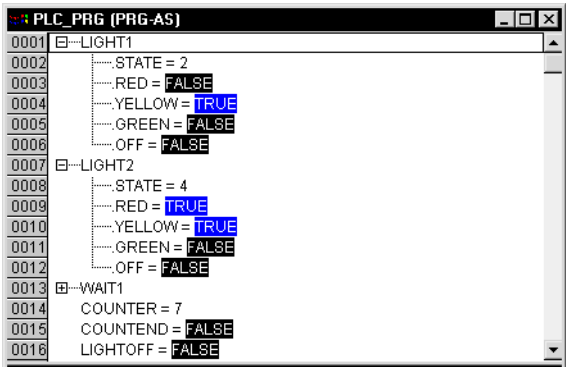
Mit diesem Befehl tragen Sie eine neue Variable in die Deklarationstabelle des Deklarationseditors ein.

- Befindet sich die aktuelle Cursorposition in einem Feld der Tabelle, wird die neue Variable vor dieser Zeile, ansonsten am Anfang der Tabelle eingefügt.
- Sie können auch am Ende der Tabelle eine neue Deklaration einfügen, indem Sie im letzten Feld der Tabelle die **<Pfeil rechts>**-Taste oder die **<Tab>**-Taste betätigen.
- Nach Ausführen des Befehls erhalten Sie eine Variable, die als Vorbelegung im Feld *Name* "Name", und im Feld *Typ* BOOL stehen hat. Diese Werte können Sie in die gewünschten Werte ändern. Name und Typ genügen für eine vollständige Variablendeklaration.



7.2.18 Deklarationseditor 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 (???).

Mehrelementige Variablen

Vor jeder mehrelementigen Variablen steht eine Pluszeichen.

- Durch Drücken der **<Eingabetaste>** oder durch Doppelklicken auf die Variable klappt diese auf, und alle ihre Komponenten werden nachfolgend aufgelistet. Vor der Variablen erscheint ein Minuszeichen.
- Mit einem weiteren Doppelklick bzw. durch Drücken der **<Eingabetaste>** werden die Komponenten wieder ausgeblendet und es erscheint wieder das Pluszeichen.

Einelementige Variablen

Durch Drücken der **<Eingabetaste>** oder durch Doppelklicken auf eine einelementige Variable öffnet sich das Dialogfeld zum Schreiben einer Variablen. Durch Eingabe eines neuen Wertes können Sie den aktuellen Wert der Variablen ändern. Bei boolschen Variablen wird der Wert getoggelt, ohne daß das Dialogfeld angezeigt wird.

Der neue Wert (jetzt mit anderer Farbe dargestellt) wird noch nicht in die Steuerung geschrieben.

- Sie können auf diese Weise die Werte beliebig vieler Variablen verändern und anschließend auf einmal (Zyklus konsistent) in die Steuerung schreiben.

7.2.19 Kommentar

Benutzerkommentare müssen in die speziellen Zeichenfolgen (* und *) eingeschlossen werden. Kommentare sind in allen Texteditoren und dort an beliebiger Stelle erlaubt. Das sind alle Deklarationen, die Sprachen AWL und ST und selbstdefinierte Datentypen.

Unter **Projekt→Optionen** Kategorie *Übersetzungsoptionen* können verschachtelte Kommentare ein- und ausgeschaltet werden.

In FUP, KOP und CFC können zu jedem Netzwerk Kommentare eingegeben werden.

- Suchen Sie hierzu das Netzwerk aus, das Sie kommentieren möchten und wählen Sie den Befehl **Einfügen→Kommentar**.

In AS können Sie Kommentare zum Schritt im Dialogfeld *Editieren von Schrittattributen* eingeben.



Tip!

Wenn Sie den Mauszeiger im Online-Modus eine kurze Zeit über einer Variablen halten, wird der Typ und gegebenenfalls der Kommentar der Variablen in einem Tooltip angezeigt.

Im Offline-Modus wird nur der Typ der Variablen angezeigt.



Drive PLC Developer Studio

Editoren

7.2.19.1 Pragma-Anweisung

- Die Pragma-Anweisung dient zum Steuern des Übersetzungsvorganges. Sie wird immer mit zusätzlichem Text in einer Programmzeile oder in einer eigenen Zeile des Deklarationseditors geschrieben.
- Sie steht in geschweiften Klammern. Groß- und Kleinschreibung wird nicht berücksichtigt.
{<Anweisung>}
- Kann der Compiler den Anweisungstext nicht interpretieren, wird das gesamte Pragma wie ein Kommentar behandelt und überlesen. Bei diesem Vorgang wird eine Warnung ausgegeben.
Ignore Compilerdirektive <Anweisungstext>
- Die Prag-Anweisung wirkt auf die Zeile in der sie steht, bzw. auf die folgenden, bis es mit einer entsprechenden Pragma-Anweisung wieder aufgehoben wird oder dieselbe Pragma-Anweisung mit anderen Parametern ausgeführt wird. Die Pragma-Anweisung endet auch am Ende eine Implementation, einer globalen Variablenliste oder Typendeklaration.
- Die öffnende Klammer darf unmittelbar auf einen Variablennamen folgen. Klammern, die zusammen gehören müssen in der selben Zeile stehen.

Pragma-Anweisungen sind mit folgender Syntax zu schreiben:

```
{flag[<flags>] [off | on]}
```

Mit diesem Pragma können die Eigenschaften einer Variablendeklaration beeinflusst werden. <flags> kann eine Kombination der folgenden Flags sein.

Folgendes Pragma ist zur Zeit verfügbar:

noinit	Die Variable wird nicht initialisiert
nowatch	Die Variable wird nicht gezeigt (kein Monitoring)
nowrite	Die Variable wird ohne Schreibrecht in die Symboldatei exportiert
noread	Die Variable wird ohne Leserecht 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. Das Pragma kann auch von einem anderen {flag<flags>on} Pragma überschrieben werden.

Ohne die Modifikation mit **on** oder **off** wirkt das Pragma nur auf die aktuelle Variablendeklaration.

Beispiele

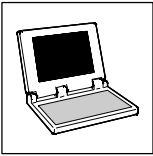
Die Variable a wird nicht initialisiert und nicht gezeigt. (Monitor)
Die Variable b wird nicht initialisiert

VAR a:INT{flag noinit, nowatch}; b:INT{flag noinit}; END VAR	VAR {flag noinit, 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
--	--

Mit den Flags noread und nowrite können einzelne Variablen mit einem eingeschränkten Zugriffsrecht versehen werden. Als Default hat die Variable die gleiche Einstellung wie der Baustein. Hat Sie keine Lese- und Schreibrechte, wird sie nicht in die Symboldatei exportiert.



Wird der Baustein mit Lese- und Schreibrecht versehen, dann kann mit folgendem Pragma die Variable a nur mit Schreibrecht und Variable b nicht exportiert werden.

<pre>VAR a:INT {flag noread}; b:INT {flag noread,nowrite}; END_VAR</pre>	<pre>VAR {flag noread on} a:INT; {flag noread,nowrite on} b:INT; {flag off} END_VAR</pre>
--	---

Die Variablen a und b werden nicht in die Symboldatei exportiert.

<pre>{flag noread, nowrite on} VAR a:INT; b:INT; END_VAR {flag off}</pre>	<pre>VAR {flag noread, nowrite on} a:INT; b:INT; {flag off} END_VAR</pre>
---	---

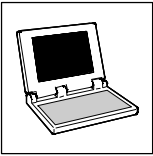
Das Pragma wird auf die untergeordneten Variablendeklarationen vererbt.

```
a:afb;
.
.
.
FUNKTION_BLOCK afB
VAR
    b:bfb{flag nowrite};
    c:INT;

END_VAR
.
.
.
FUNKTION_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.

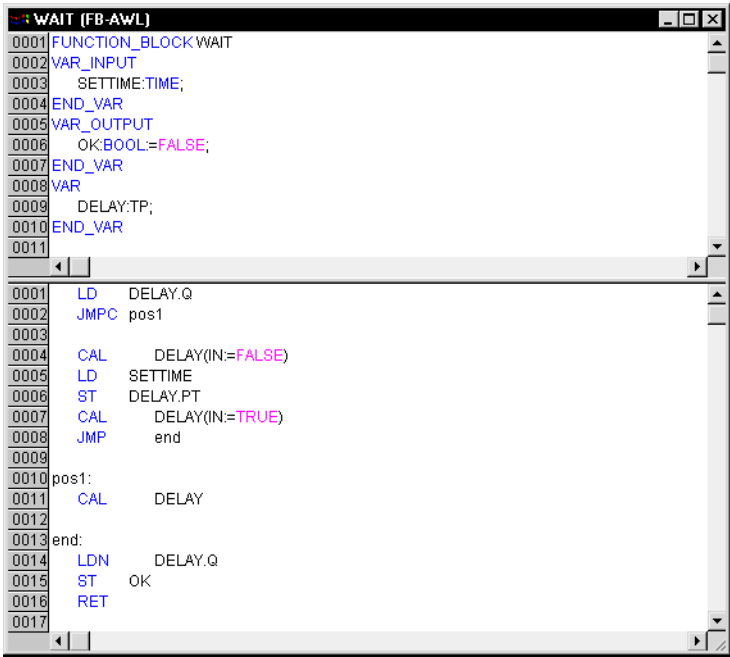


Drive PLC Developer Studio

Editoren

7.3 Texteditoren

Die für den Implementierungsteil verwendeten Texteditoren des DDS (**Anweisungslisteneditor** und **Editor für Strukturierten Text**) verfügen über die üblichen Funktionen von Windows-Texteditoren, desweiteren unterstützen sie das Programmtext farblich unterschieden wird. Diese Eigenschaft wird als Syntaxcoloring bezeichnet.



7.3.1 Befehle im Menü “Einfügen”

7.3.1.1 Operator

Symbol: -	Menübefehl: Einfügen→Operator	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

Mit diesem Befehl werden in einem Dialogfeld alle Operatoren angezeigt, die in der aktuellen Sprache verfügbar sind.

- Wenn Sie einen der Operatoren auswählen und auf **OK** klicken, wird der markierte Operator an die aktuelle Cursorposition eingefügt.

7.3.1.2 Operand

Symbol: -	Menübefehl: Einfügen→Operand	Tastaturbefehl: -
-----------	-------------------------------------	-------------------

Mit diesem Befehl werden in einem Dialogfeld alle Variablen angezeigt.

Sie können wählen, ob eine Liste der globalen, der lokalen oder der Systemvariablen dargestellt werden soll.

- Wenn Sie einen der Operanden auswählen und auf **OK** klicken, wird der markierte Operand an der aktuellen Cursorposition eingefügt.



7.3.1.3 Funktion

Symbol: -	Menübefehl: Einfügen→Funktion	Tastaturbefehl: -
-----------	-------------------------------	-------------------

- Mit diesem Befehl werden in einem Dialogfeld alle Funktionen angezeigt.
- Sie können wählen, ob eine Liste der benutzerdefinierten oder der Standardfunktionen dargestellt werden soll.
- Wenn Sie eine der Funktionen auswählen und auf **OK** klicken, wird die markierte Funktion an die aktuelle Cursorposition eingefügt.
 - Wurde im Dialogfeld das Kontrollkästchen **Mit Argumenten** aktiviert, werden die erforderlichen Eingabevariablen der Funktion mit eingefügt.

7.3.1.4 Funktionsblock

Symbol: -	Menübefehl: Einfügen→Funktionsblock	Tastaturbefehl: -
-----------	-------------------------------------	-------------------

- Mit diesem Befehl werden in einem Dialogfeld alle Funktionsblöcke angezeigt.
- Sie können wählen, ob eine Liste der benutzerdefinierten oder der Standardfunktionsblöcke dargestellt werden soll.
- Wenn Sie einen der Funktionsblöcke auswählen und auf **OK** klicken, wird der markierte Funktionsblock an die aktuelle Cursorposition eingefügt.
 - Wurde im Dialogfeld die Option **Mit Argumenten** aktiviert, werden die erforderlichen Eingangs-Variablen 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);

Ein mehrzeiliger Bausteinaufruf ist möglich.

```
CAL CTU_inst(  
  CU:=%IX1.0.0,  
  PV:=(  
    LD A  
    ADD 5  
  )  
)
```

7.3.2 Die Texteditoren im Online-Modus

- Im Online-Modus können Sie in den Texteditoren Breakpoints setzen sowie Einzelschrittabarbeitung (Steppen) durchführen. Zusammen mit dem Monitoring haben Sie die Debugging-Funktion eines modernen Windows-Hochsprachendebuggers.
- Im Online-Modus ist das Texteditor-Fenster vertikal zweigeteilt:
- Links befindet sich der normale Programmtext.
 - Rechts sind die Variablen mit ihrem jeweiligen Wert dargestellt.



Drive PLC Developer Studio

Editoren



Tip!

Wenn die Steuerung läuft, werden die aktuellen Werte der jeweiligen Variablen dargestellt (Monitoring).

Beim Monitoring von Ausdrücken oder Bit-adressierten Variablen ist folgendes zu beachten.

Bei den Bit-adressierten Variablen wird immer der angesprochene Bit-Wert gezeigt. (Monitor)

Wenn Sie den Mauszeiger im Online-Modus eine kurze Zeit über einer Variablen halten, wird der Typ und gegebenenfalls der Kommentar der Variablen in einem Tooltip angezeigt.

Im Offline-Modus wird nur der Typ der Variablen angezeigt.

Mehrelementige Variablen

Mehrelementige Variable (Arrays, Strukturen oder Instanzen von Funktionsblöcken) sind durch ein Pluszeichen vor dem Bezeichner gekennzeichnet.

- Durch Drücken der **<Eingabetaste>** oder durch Doppelklicken auf die Variable klappt diese auf, und alle ihre Komponenten werden nachfolgend aufgelistet. Vor der Variablen erscheint ein Minuszeichen.
- Mit einem weiteren Doppelklick bzw. durch Drücken der **<Eingabetaste>** werden die Komponenten wieder ausgeblendet und es erscheint wieder das Pluszeichen.

7.3.2.1

Monitoring-Fenster konfigurieren

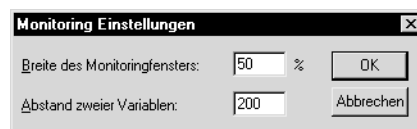
Symbol: -	Menübefehl: Extras→Monitoring Einstellungen	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl können Sie Ihr Monitoring-Fenster konfigurieren.

Im Online-Modus ist das Texteditor-Fenster vertikal zweigeteilt:

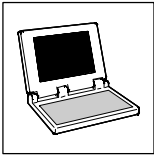
- Links befindet sich der normale Programmtext.
- Rechts sind die Variablen mit ihrem jeweiligen Wert dargestellt.

Im Dialogfeld *Monitoring Einstellungen* können Sie folgende Einstellungen vornehmen:



- Die Breite des Monitoring-Bereichs (in Prozent).
- Der Abstand zweier Monitoring-Variablen in einer Zeile.

Die Abstandsangabe 1 entspricht dabei ein Pixel in der gewählten Auflösung.

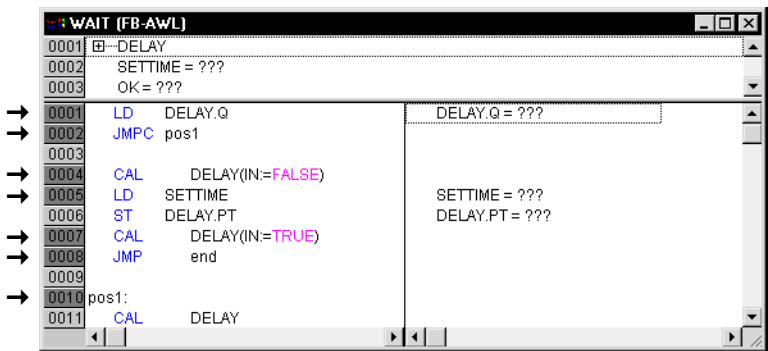


7.3.3 Breakpoint-Positionen

Breakpoint-Positionen sind alle Stellen im Programm, an denen sich Variablenwerte ändern können oder an denen der Programmfluß verzweigt.

- Da intern im DDS mehrere AWL-Zeilen zu einer C-Code-Zeile zusammengefaßt werden, können nicht in jeder Zeile Breakpoints gesetzt werden.
- Besonderheit Funktionsaufrufe:
Hier muß gegebenenfalls ein Breakpoint in der Funktion gesetzt werden

Mögliche Breakpoint-Positionen sind durch ein dunkelgraues Zeilennummernfeld gekennzeichnet:



Anweisungslisteneditor

Folgende Breakpoint-Positionen sind im Anweisungslisteneditor möglich:

- 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

Editor für Strukturierten Text

Folgende Breakpoint-Positionen sind im Editor für Strukturierten Text möglich:

- Bei jeder Zuweisung
- Bei jeder RETURN und EXIT-Anweisung
- in Zeilen, in denen Bedingungen ausgewertet werden (WHILE, IF, REPEAT)
- Am Ende des Bausteins



Drive PLC Developer Studio

Editoren

7.3.4 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, wird rot dargestellt.
- In der Steuerung steht die Bearbeitung des Anwenderprogramms.

Mit **Online→Start** kann das Programm nach einem Breakpoint-bedingten Halt fortgesetzt werden.

Mit **Online→Einzelschritt über** bzw. **Online→Einzelschritt in** kann das Programm bis zur nächsten Breakpoint-Position fortgesetzt werden.

- Ist die Anweisung, auf der angehalten wurde, ein **CALL**-Befehl oder steht in den Zeilen bis zur nächsten Breakpoint-Position ein Funktionsaufruf, so wird dieser mit **Online→Einzelschritt über** übersprungen, mit **Online→Einzelschritt in** wird in den aufgerufenen Baustein verzweigt.
- Wird das Programm an einen Breakpoint gestoppt, dann werden alle Tasks eingefroren. Nach dem Aufheben eines Breakpoints läuft das Programm an der Stelle weiter, an der es gestoppt wurde.



Hinweis!

Sie sollten genau überlegen, an welche Stelle Sie einen Breakpoint setzen, damit es im laufenden Prozess nicht zu unerwarteten Störungen kommt.

Durch das Setzen eines Breakpoints wird das Zeitverhalten des Zielsystems geändert.

Wird das Programm an einem Breakpoint gestoppt, dann verhält sich der Antrieb wie in **Projekt→Ausnahmeverhalten** eingestellt. (☐ 6-22)

7.3.5 Zeilennummern des Texteditors

Die Zeilennummern des Texteditors geben die Nummer jeder Textzeile einer Implementierung eines Bausteins an.

Offline-Modus

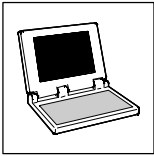
Im Offline-Modus können Sie durch Klicken auf eine Zeilennummer die gesamte Textzeile markieren.

Online-Modus

Im Online-Modus zeigt die Hintergrundfarbe der Zeilennummer den Breakpointzustand 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.

Ein einfacher Mausklick wechselt im Online-Modus den Breakpointzustand dieser Zeile.



7.4 **Netzwerkeditoren (allgemein)**

Mit den Netzwerkeditoren des DDS können Sie in den graphisch orientierten Sprachen FUP, KOP und CFC programmieren.

Bei den Netzwerkeditoren gemeinsam sind Sprungmarken, Netzwerkkommentare und die Befehle **Einfügen→Netzwerk (danach)** / **Einfügen→Netzwerk (davor)**.

7.4.1 **Sprungmarken**

Zu jedem Netzwerk gehört eine Sprungmarke (Label), die wahlweise auch leer sein kann.

Diese Marke können Sie editieren, indem Sie in die erste Zeile des Netzwerks unmittelbar neben die Netzwerknummer klicken. Jetzt können Sie eine Marke gefolgt von einem Doppelpunkt eingeben.

7.4.2 **Netzwerkkommentare**

Zu jedem Netzwerk kann ein mehrzeiliger Kommentar vergeben werden. Kommentare werden im Unterschied zum Programmtext in grauer Schrift dargestellt.

7.4.2.1 **Optionen für Netzwerkkommentare**

Symbol: -	Menübefehl: Extras→Optionen	Tastaturbefehl: -
-----------	-----------------------------	-------------------

Mit diesem Befehl können Sie folgende Optionen für Netzwerkkommentare festlegen:

Maximale Kommentargröße

Im Feld *Maximale Kommentargröße* können Sie die Anzahl der Zeilen festlegen, die maximal für einen Netzwerkkommentar zur Verfügung stehen sollen (Voreinstellung: 4).

Minimale Kommentargröße

Im Feld *Minimale Kommentargröße* können Sie die Anzahl der Zeilen festlegen, die generell für Kommentare freigelassen werden sollen (Voreinstellung: 0).

- Ist hier z. B. der Wert 2 eingestellt, stehen an jedem Netzwerkanfang nach der Labelzeile zwei leere Kommentarzeilen. Beachten Sie, daß ein kleiner Wert den Vorteil hat, daß mehr Netzwerke in den Bildschirmbereich passen.
- Ist die minimale Kommentargröße größer als 0, können Sie einfach in die entsprechende Kommentarzeile klicken, um den Kommentar einzugeben.
- Ist die minimale Kommentargröße gleich 0, müssen Sie zunächst das Netzwerk, zu dem ein Kommentar eingegeben werden soll, auswählen und dann mit **Einfügen→Kommentar** eine Kommentarzeile einfügen.

7.4.3 **Neues Netzwerk einfügen**

7.4.3.1 **Netzwerk (danach) / Netzwerk (davor)**

Symbol: -	Menübefehl: Einfügen→Netzwerk (danach) Einfügen→Netzwerk (davor)	Tastaturbefehl: <Strg>+<T> -
-----------	---	---------------------------------

Mit diesen Befehlen fügen Sie ein neues Netzwerk im FUP-/KOP-Editor vor bzw. nach dem aktuellen Netzwerk ein.

- Das aktuelle Netzwerk (mit gepunktetem Rechteck und der Netzwerknummer) ändern Sie durch Klicken auf die Netzwerknummer.
- Durch Klicken bei gedrückter <Umschalt>-Taste wählen Sie alle Netzwerke zwischen dem aktuellen und dem angeklickten Netzwerk aus.



Drive PLC Developer Studio

Editoren

7.4.4 Inputs/Outputs on the fly

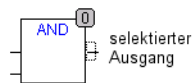
Diese Funktion unterstützt das Erstellen von Ein- und Ausgängen durch direkte Tastatureingabe oder mit Hilfe eines Auswahlfeldes.



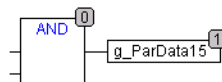
Tip!

Diese Funktion erspart beim CFC-Editor das Ziehen und Verbinden von Ein- Ausgängen und Bausteinen

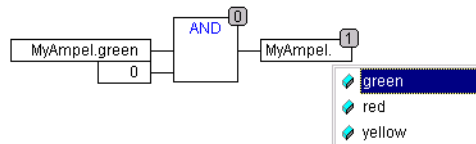
- Selektieren Sie z. B. im CFC-Editor einen Ein- oder Ausgang.

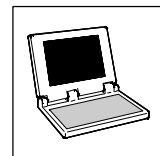


- Geben Sie direkt über die Tastatur den Variablennamen ein. Bei bereits vereinbarten Variablen muß der Name korrekt geschrieben sein. Ist die Variable nicht vorhanden, muß sie mit der Variablendeklaration vereinbart werden.



Wird der Name bzw. der Instanzname eines Bausteins, bzw. einer Struktur mit einem Punkt eingegeben, erscheint eine Auswahlliste aller lokalen Variablen.





7.4.5 Netzwerkeditoren im Online-Modus

Breakpoints

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 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.

Monitoring

Alle Werte werden an den Ein- und Ausgängen der Netzwerkbausteine gemonitort.

Werte schreiben im FUP

Ein Doppelklick auf eine Variable öffnet ein Dialogfeld, in dem Sie den aktuellen Wert der Variable verändern können (bei boolschen Variablen erscheint kein Dialogfeld, sie werden getoggelt).

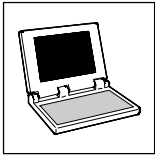
- Der neue Wert wird rot dargestellt.
- Mit **Online→Werte schreiben** werden alle veränderten Variablen auf den neuen Wert gesetzt und wieder in schwarz dargestellt.



Tip!

Wenn Sie den Mauszeiger im Online-Modus eine kurze Zeit über einer Variablen halten, wird der Typ und gegebenenfalls der Kommentar der Variablen in einem Tooltip angezeigt.

Im Offline-Modus wird nur der Typ der Variablen angezeigt.

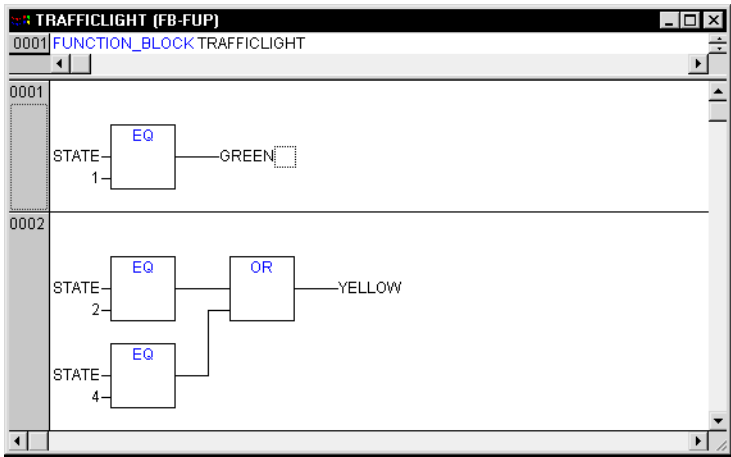


Drive PLC Developer Studio

Editoren

7.5 Funktionsplaneditor

Der Funktionsplaneditor ist ein graphischer Editor, der mit einer Liste von Netzwerken arbeitet. Jedes Netzwerk enthält eine Struktur, die jeweils einen logischen bzw. arithmetischen Ausdruck, den Aufruf eines Funktionsblocks, einer Funktion, eines Programms, einen Sprung oder eine Return-Anweisung darstellt.



Tip!
Die wichtigsten Befehle zum Funktionsplaneditor finden Sie auch im Kontextmenü (rechte Maustaste oder <Strg>+<F10>)

7.5.1 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:

	Jedes Textfeld (mögliche Cursorpositionen umrahmt)
	Jeder Eingang
	Jeder Baustein
	Ausgänge, wenn danach eine Zuweisung oder ein Sprung kommt
	Das Linienkreuz über einer Zuweisung, einem Sprung oder einer Return-Anweisung
	Das Linienkreuz unmittelbar vor einer Zuweisung
	Hinter dem letzten (rechten) Objekt eines Netzwerkes ("Letzte Cursorposition")



7.5.2 Cursor setzen

Den Cursor können Sie durch Klicken der Maus oder mit Hilfe der Tastatur auf eine bestimmte Position setzen.

- 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, 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.

7.5.3 Befehle im Menü "Einfügen"

7.5.3.1 Zuweisung

Symbol: 	Menübefehl: Einfügen→Zuweisung	Tastaturbefehl: <Strg>+<A>
---	---------------------------------------	---

Dieser Befehl fügt eine Zuweisung ein.

Eingefügt wird, je nach selektierter Position,

- unmittelbar vor dem selektierten Eingang,
- unmittelbar nach dem selektierten Ausgang,
- unmittelbar vor dem selektierten Linienkreuz
oder
- am Ende des Netzwerkes.

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**.



Tip!

Dieser Befehl ist auch im Kontaktplaneditor verfügbar.

7.5.3.2 Sprung

Symbol: 	Menübefehl: Einfügen→Sprung	Tastaturbefehl: <Strg>+<L>
---	------------------------------------	---

Dieser Befehl fügt einen Sprung ein.

Eingefügt wird, je nach selektierter Position,

- unmittelbar vor dem selektierten Eingang,
- unmittelbar nach dem selektierten Ausgang,
- unmittelbar vor dem selektierten Linienkreuz
oder
- am Ende des Netzwerkes.

Zu einem eingefügten Sprung kann anschließend der eingetragene Text "???" selektiert und durch die Sprungmarke, an die gesprungen werden soll, ersetzt werden.



Drive PLC Developer Studio

Editoren

7.5.3.3 Return

Symbol:	Menübefehl: Einfügen→Return	Tastaturbefehl: <Strg>+<R>
---------	------------------------------------	---

Dieser Befehl fügt eine **RETURN**-Anweisung ein.
Eingefügt wird, je nach selektierter Position,

- unmittelbar vor dem selektierten Eingang,
- unmittelbar nach dem selektierten Ausgang,
- unmittelbar vor dem selektierten Lilienkreuz oder
- am Ende des Netzwerkes.

7.5.3.4 Baustein

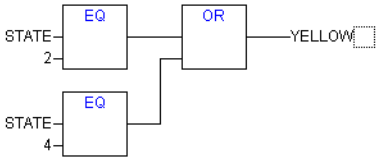
Symbol:	Menübefehl: Einfügen→Baustein	Tastaturbefehl: <Strg>+
---------	--------------------------------------	---

Dieser Befehl fügt Operatoren, Funktionen, Funktionsblöcke und Programme ein.

- Der Befehl fügt zuerst ein AND-Operator ein.
- Durch Überschreiben des Baustein-Typs, z. B. das AND durch OR, kann er umgewandelt werden.
- Mit der Eingabehilfe **<F2>** kann der gewünschte Baustein ausgewählt werden.
- Bei Funktionen und Funktionsblöcken werden die formalen Namen der Ein- und Ausgänge angezeigt.
- Bei Funktionsblöcken ist ein veränderbares Instanz-Feld über der Box vorhanden. Wird der Baustein-Typ geändert und dadurch ein unbekannter Funktionsblock aufgerufen, zeigt das System eine Operator-Box mit zwei Eingängen und dem angegebenen Typ.
- Ist das Instanz-Feld markiert, kann mit **<F2>** wird die Eingabehilfe mit den Kategorien zur Variablenauswahl aufgerufen.
- Ein neuer Baustein wird abhängig von der selektierten Position eingefügt.

Beispiel:

Die aktuelle Cursor-Position wird durch ein kleines rechteckiges Kästchen angezeigt. In dem Beispiel hinter der Zuweisung YELLOW.

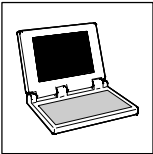


- Hat der neue Baustein eine andere Mindestanzahl von Eingängen, so werden diese angehängt.
 - Hat der neue Baustein eine kleinere Höchstzahl von Eingängen, so werden die letzten Eingänge samt der davorliegenden Äste gelöscht.
- Alle Eingänge des Bausteins, die nicht verbunden werden konnten, erhalten den Text "???".
- Klicken Sie auf diesen Text, um ihn in die gewünschte Konstante bzw. Variable zu ändern.

Eingefügt wird je nach selektierter Position:

Eingang selektiert

- Ist ein Eingang selektiert, wird der Operator vor diesem Eingang eingefügt.
- Der erste Eingang dieses Operators wird mit dem Zweig links vom selektierten Eingang verbunden.
 - Der Ausgang des neuen Operators wird mit dem selektierten Eingang verbunden.



Ausgang selektiert

Ist ein Ausgang selektiert, dann wird der Operator nach diesem Ausgang eingefügt.

- Der erste Eingang des Operators wird mit dem selektierten Ausgang verbunden.
- Der Ausgang des neuen Operators wird mit dem Zweig, mit dem der selektierte Ausgang verbunden war, verbunden.

Baustein, Funktion oder Funktionsblock selektiert

Ist ein Baustein, eine Funktion oder ein Funktionsblock selektiert, so wird das alte Element durch den neuen Operator 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.

Sprung oder Return selektiert

Ist ein Sprung oder ein Return selektiert, so wird der Baustein vor diesem Sprung, bzw. Return eingefügt.

- Der erste Eingang des Operators wird mit dem Zweig links des selektierten Elements verbunden.
- Der Ausgang des Operators wird mit dem Zweig rechts des selektierten Elements verbunden.

Letzte Cursorposition im Netzwerk selektiert

- Ist die letzte Cursorposition eines Netzwerks selektiert, so wird der Baustein nach dem letzten Element eingefügt.
- Der erste Eingang des Bausteins wird mit dem Zweig links der selektierten Position verbunden.

7.5.3.5

Eingang

Symbol: 	Menübefehl: Einfügen→Eingang	Tastaturbefehl: <Strg>+<U>
---	-------------------------------------	---

Dieser Befehl fügt einen Operatoreingang ein.

Die Zahl der Eingänge ist bei vielen Operatoren variabel (z. B. kann **ADD** zwei oder mehr Eingänge haben).

Um einen solchen Operator um einen Eingang zu erweitern, muß der Eingang, vor dem ein weiterer eingefügt werden soll, oder der Operator selbst, wenn ein unterster Eingang angefügt werden soll, selektiert werden.

Der eingefügte Eingang ist mit dem Text "???" vorbelegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Konstante bzw. Variable zu ändern. Sie können dazu auch die Eingabehilfe (<F2>) verwenden.



Tip!

Dieser Befehl ist auch im Kontaktplaneditor verfügbar.



Drive PLC Developer Studio

Editoren

7.5.3.6

Ausgang

Symbol: 	Menübefehl: Einfügen→Ausgang	Tastaturbefehl: -
---	-------------------------------------	-------------------

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 bzw. der unmittelbar davorliegende Ausgang selektiert, wird nach den bereits vorhandenen Zuweisungen eine weitere angefügt.
- Ist das Linienkreuz direkt vor einer Zuweisung selektiert, wird vor dieser Zuweisung eine weitere eingefügt.

Der eingefügte Ausgang ist mit dem Text "???" belegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Variable zu ändern. Sie können dazu auch die Eingabehilfe (<F2>) verwenden.



Tip!

Dieser Befehl ist auch im Kontaktplaneditor verfügbar.

7.5.3.7

Kommentar

Symbol: 	Menübefehl: Einfügen→Kommentar	Tastaturbefehl: -
---	---------------------------------------	-------------------

Mit diesem Befehl kann ein Kommentar in das aktuelle Netzwerk eingefügt werden.

Nach Ausführung des Befehls erscheint das Wort "Comment" im oberen Bereich des aktuellen Netzwerkes. Mit einem Mausklick auf das Wort wird das Textfeld aktiv und "Comment" kann überschrieben werden.

7.5.4

Befehle im Menü "Extras"

7.5.4.1

Negation

Symbol: 	Menübefehl: Extras→Negation	Tastaturbefehl: <Strg>+<N>
---	------------------------------------	----------------------------

Mit diesem Befehl können Eingänge, Ausgänge, Sprünge oder RETURN-Anweisungen negiert werden.

Das Symbol für die Negation ist ein kleiner Kreis auf einer Verbindung.

- Wenn ein Eingang selektiert ist, dann wird dieser Eingang negiert.
- Wenn ein Ausgang selektiert ist, dann wird dieser Ausgang negiert.
- Wenn ein Sprung oder ein RETURN markiert ist, dann wird der Eingang dieses Sprungs, bzw. RETURNS negiert.

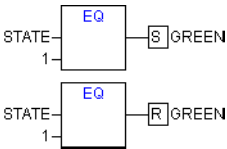
Eine Negation kann durch erneutes Negieren gelöscht werden.



7.5.4.2 Set/Reset

Symbol:	Menübefehl: Extras→Set/Reset	Tastaturbefehl: -
---------	-------------------------------------	-------------------

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:



- 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.

7.5.4.3 Zoom

Symbol: -	Menübefehl: Extras→Zoom	Tastaturbefehl: <Alt>+<Eingabetaste>
-----------	--------------------------------	---

- Mit diesem Befehl wird ein selektierter Baustein in seinen Editor geladen.
- Handelt es sich um einen Baustein aus einer Bibliothek, wird bei dem Ausführen des Befehls der Bibliotheksverwalter aufgerufen.

7.5.4.4 Instanz öffnen

Symbol: -	Menübefehl: Extras→Instanz öffnen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl entspricht dem Befehl **Projekt→Instanz öffnen**.

7.5.5 Befehle im Menü “Bearbeiten”

7.5.5.1 Ausschneiden, Einfügen, Kopieren und Löschen

Symbol: -	Menübefehl: Bearbeiten→Ausschneiden Bearbeiten→Einfügen Bearbeiten→Kopieren Bearbeiten→Löschen	Tastaturbefehl: <Strg>+<X> <Strg>+<V> <Strg>+<C> <Entf>
-----------	---	--

Ausschneiden, Kopieren, Löschen

- Ist ein Linienkreuz selektiert, werden die darunterliegenden Zuweisungen, Sprünge oder **RETURN**-Anweisungen ausgeschnitten, kopiert oder gelöscht.
- Ein selektierter Baustein wird mit allen an den Eingängen anliegenden Äste mit Ausnahme des ersten Astes ausgeschnitten, kopiert oder gelöscht.
- Ansonsten wird der gesamte vor der Cursorposition liegende Ast ausgeschnitten, kopiert oder gelöscht.



Drive PLC Developer Studio

Editoren

Einfügen

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 Baustein geladen wurde (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.
- Ansonsten 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.



Tip!

Durch **Ausschneiden** und **Einfügen** läßt sich 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, soll aber mit dem zweiten Eingang verbunden sein.

Selektieren Sie den ersten Eingang und wählen Sie **Bearbeiten→Ausschneiden**. Anschließend selektieren Sie den zweiten Eingang und wählen **Bearbeiten→Einfügen**.

Der Ast hängt nun am zweiten Eingang.

7.5.6

Der Funktionsplan-Editor 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 vor dem Netzwerk auf dem der Breakpoint steht und das Netzwerknummernfeld wird rot. Beim Steppen wird jedes Netzwerk angesprungen.

Zu jeder Variablen wird der aktuelle Wert dargestellt.



Tip!

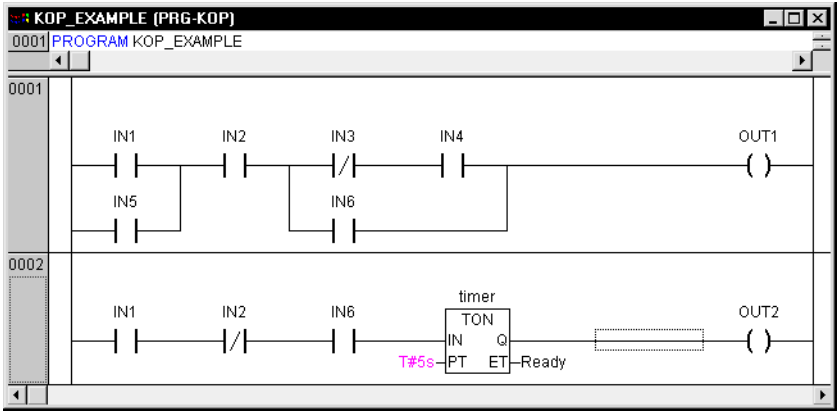
Wenn der Eingang eines Funktionsblocks ein Ausdruck ist, wird nur die erste Variable des Ausdrucks gemonitort (sichtbar).

- Doppelklick auf die Variable öffnet den Dialog zum Schreiben einer Variablen.
- Es ist dann möglich den aktuellen Wert der Variablen zu verändern.
- Sind es boolsche Variablen erscheint kein Dialog. Sie werden getoggelt.
- Der neue Wert wird cyan und bleibt unverändert.
- Wird der Menübefehl **Online→Werte schreiben** ausgeführt, werden alle Variablen auf die gewählten Werte gesetzt und schwarz dargestellt.



7.6 Kontaktplaneditor

Der Kontaktplaneditor ist ein graphischer Editor, der mit einer Folge von Netzwerken arbeitet.



Tip!
Die wichtigsten Befehle zum Kontaktplaneditor finden Sie auch im Kontextmenü (rechte Maustaste).

7.6.1 Cursorpositionen im KOP

Bausteine mit EN-Eingängen und daran geknüpfte andere Bausteine werden behandelt wie im Funktionsplaneditor.

Folgende Stellen können Cursorpositionen sein, wobei Funktionsblock- und Programmaufrufe wie Kontakte behandelt werden können:

	Jedes Textfeld (mögliche Cursorpositionen markiert)
	Jeder Kontakt oder Funktionsblock
	Jede Spule
	Die Verbindungslinie zwischen Kontakt und Spule

7.6.2 Befehle im Menü “Einfügen”

7.6.2.1 Kontakt

Symbol:	Menübefehl: Einfügen→Kontakt	Tastaturbefehl: <Strg>+<K>
---------	-------------------------------------	---

Dieser Befehl fügt einen Kontakt vor der markierten Stelle im Netzwerk ein.

- Ist die markierte Stelle eine Spule oder die Verbindungslinie zwischen Kontakten und Spule, wird der neue Kontakt seriell zur bisherigen Kontaktschaltung geschaltet.

Der eingefügte Kontakt ist mit dem Text “???” vorbelegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Konstante bzw. Variable zu ändern. Sie können dazu auch die Eingabehilfe (<F2>) verwenden.



Drive PLC Developer Studio

Editoren

7.6.2.2 Paralleler Kontakt

Symbol: 	Menübefehl: Einfügen→Paralleler Kontakt	Tastaturbefehl: <Strg>+<R>
---	--	---

Dieser Befehl fügt einen Kontakt parallel zur markierten Stelle im Netzwerk ein.

- Ist die markierte Stelle eine Spule oder die Verbindung zwischen Kontakt und Spule, wird der neue Kontakt parallel zur gesamten bisherigen Kontaktschaltung geschaltet.

Der eingefügte Kontakt ist mit dem Text "???" vorbelegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Konstante bzw. Variable zu ändern. Sie können dazu auch die Eingabehilfe (<F2>) verwenden.

7.6.2.3 Spule

Symbol: 	Menübefehl: Einfügen→Spule	Tastaturbefehl: <Strg>+<L>
---	-----------------------------------	---

Dieser Befehl fügt eine Spule parallel zu den bisherigen Spulen ein.

- Wenn die markierte Stelle die Verbindung zwischen den Kontakten und den Spulen ist, dann wird die neue Spule als letzte eingefügt.
- Wenn die markierte Stelle eine Spule ist, dann wird die neue Spule direkt darüber eingefügt.

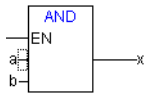
Die Spule erhält als Vorbelegung den Text "???".

- Klicken Sie auf den Text, um ihn in die gewünschte Variable zu ändern. Sie können dazu auch die Eingabehilfe (<F2>) verwenden.

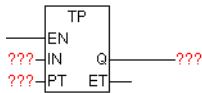
7.6.2.4 Einfügen an Baustein

Symbol: -	Menübefehl: Einfügen→Einfügen an Baustein →Eingang →Ausgang →Box →Zuweisung	Tastaturbefehl: <Strg>+<U> <Strg>+<A>
-----------	--	---

Dieses Untermenü im Menü **Einfügen** enthält Befehle, um an einen bereits eingefügten Baustein mit EN-Eingang weitere Elemente anzufügen.



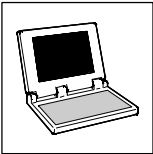
Ein Baustein mit EN kann den Namen eines Funktionsblocks bekommen, dann kann **Einfügen→Einfügen an Bausteinen** auch ausgeführt werden.



In diesen Beispiel wurde **AND** mit dem Namen des Timers **TP** überschrieben.

- Die Befehle in diesem Untermenü sind an den selben Cursorpositionen ausführbar wie die entsprechenden Befehle im Funktionsplan.

Eingang	Mit Eingang fügen Sie einen neuen Eingang an den Baustein an.
Ausgang	Mit Ausgang fügen Sie an einem Ausgang des Bausteins eine weitere Zuweisung an.
Box	Mit Box fügen Sie einen neuen Baustein an den markierten an. Der Ausgang des neuen Bausteins wird an den selektierten Eingang angelegt. Der Baustein ist mit AND vorbelegt.
Zuweisung	Mit Zuweisung fügen Sie eine Zuweisung zu einer Variablen ein.



7.6.2.5 Sprung

Symbol: -	Menübefehl: Einfügen→Sprung	Tastaturbefehl: -
-----------	-----------------------------	-------------------

Dieser Befehl fügt 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 oder eine Spule sein.

Der Sprung erhält als Vorbelegung den Text "???".

- Klicken Sie auf den Text, um ihn in die gewünschte Sprungmarke zu ändern.

7.6.2.6 Return

Symbol: -	Menübefehl: Einfügen→Return	Tastaturbefehl: -
-----------	-----------------------------	-------------------

Dieser Befehl fügt eine RETURN-Anweisung parallel am Ende zu den bisherigen Spulen ein.

- Wenn die eingehende Leitung den Wert "An" liefert, wird die Abarbeitung des Bausteins in diesem Netzwerk abgebrochen.
- Die markierte Stelle muß die Verbindung zwischen den Kontakten und den Spulen oder eine Spule sein.

7.6.3 Bausteine mit EN-Eingängen

Wenn Sie Ihr KOP-Netzwerk zur Steuerung von Aufrufen anderer Bausteine verwenden wollen, dann müssen Sie einen Baustein mit einem EN-Eingang oder einen Baustein mit boolschen Eingang einfügen.

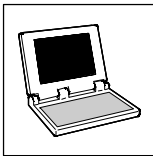
- Ein Baustein mit EN-Eingängen wird parallel zu den Spulen geschaltet.
- Von einem EN-Baustein ausgehend können Sie das Netzwerk wie im Funktionsplan weiterentwickeln.
- Die Befehle zum Einfügen von Elementen an einen bereits eingefügten EN-Baustein finden Sie im Menü **Einfügen** im Untermenü **Einfügen an Baustein**.

Ein Operator, ein Funktionsblock, ein Programm oder eine Funktion mit EN-Eingang verhält sich wie der entsprechende Baustein im Funktionsplan, nur 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, wird der Baustein ausgewertet.

KOP als FUP

Wenn einmal ein Baustein mit EN-Eingang angelegt wurde, kann mit diesem Baustein ein Netzwerk wie im Funktionsplan angelegt werden. In einem EN-Baustein können Daten von Operatoren, Funktionen, Funktionsblöcken fließen, und ein EN-Baustein kann Daten an solche Bausteine transportieren.

Wenn Sie also im KOP-Editor ein Netzwerk wie in FUP programmieren möchten, müssen Sie nur in ein neues Netzwerk zuerst einen EN-Operator einfügen, Sie können dann anschließend von diesem Baustein aus ihr Netzwerk weiterbilden wie im FUP-Editor. Ein so gebildetes Netzwerk verhält sich wie das entsprechende Netzwerk in FUP.



Drive PLC Developer Studio

Editoren

7.6.3.1 Baustein mit EN

Symbol: -	Menübefehl: Einfügen→Baustein mit EN	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl fügt einen Operator mit EN-Eingang ein.

- Die markierte Stelle muß die Verbindung zwischen Kontakten und Spulen oder eine Spule sein.

Der neue Operator wird parallel zu den Spulen unterhalb der markierten eingefügt und ist stets ein **AND**.

- Dieser kann durch Selektieren und Überschreiben des Textes in jeden anderen Operator umgewandelt werden.
- Mit der Eingabehilfe (<F2>) können Sie den gewünschten Operator aus der Liste der unterstützten Operatoren auswählen.

7.6.4 Befehle im Menü “Extras”

7.6.4.1 Dahinter Einfügen

Symbol: -	Menübefehl: Extras→Dahinter Einfügen	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl fügt 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.

7.6.4.2 Darunter Einfügen

Symbol: -	Menübefehl: Extras→Darunter Einfügen	Tastaturbefehl: <Strg>+<U>
-----------	---	----------------------------

Dieser Befehl fügt 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.

7.6.4.3 Darüber Einfügen

Symbol: -	Menübefehl: Extras→Darüber Einfügen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl fügt 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.

7.6.4.4 Negation

Symbol: 	Menübefehl: Extras→Negation	Tastaturbefehl: <Strg>+<N>
---	------------------------------------	----------------------------

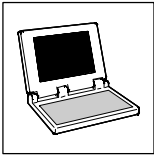
Dieser Befehl negiert einen Kontakt, eine Spule, eine Sprung-/RETURN-Anweisung oder einen Ein- bzw. Ausgang eines EN-Bausteins an der aktuellen Cursorposition.

- Zwischen den runden Klammern der Spule bzw. zwischen den geraden Strichen des Kontakts erscheint ein Schrägstrich: (/) bzw. I/I.
- Bei Sprüngen, Return, 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.



7.6.4.5 Set/Reset

Symbol: -	Menübefehl: Extras→Set/Reset	Tastaturbefehl: -
-----------	------------------------------	-------------------

Mit diesem Befehl können Spulen als Set- bzw. Reset-Spule definiert werden.

Eine Set-Spule wird mit einem "S" im Spulensymbol gekennzeichnet.

Wenn das Netzwerkergebnis TRUE ist, dann wird die boolsche Variable auf Grund der Set-Spule auf TRUE gesetzt. Ansonsten bleibt die Variable unbeeinflußt.

Eine Reset-Spule überschreibt niemals den Wert FALSE in der zugehörigen boolschen Variablen.

Eine Reset-Spule wird mit einem "R" im Spulensymbol gekennzeichnet.

Wenn das Netzwerkergebnis TRUE ist, dann wird die boolsche Variable auf Grund der Set-Spule auf FALSEs gesetzt. Ansonsten bleibt die Variable unbeeinflußt.

Bei mehrfachen Ausführen des Befehls wechselt die Spule zwischen Set-, Reset- und normaler Spule.

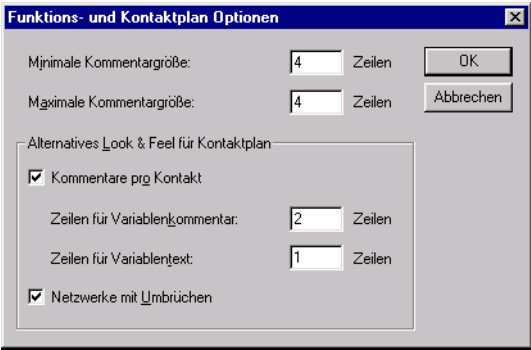
7.6.4.6 Instanz öffnen

Dieser Befehl entspricht dem Befehl **Projekt→Instanz öffnen**.

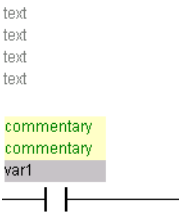
7.6.4.7 Optionen

Symbol: -	Menübefehl: Extras→Optionen	Tastaturbefehl: -
-----------	-----------------------------	-------------------

Mit diesem Menübefehl öffnen Sie das Dialogfeld *Funktions- und Kontaktplan Optionen*.



Wenn Sie die Eingabefelder wie im Bild gezeigt ausfüllen wird der entsprechende Kontakt mit folgendem Kommentarfeld ergänzt.



In den Eingabefeldern **Minimale Kommentargröße** und **Maximale Kommentargröße** kann ein Kommentar zum aktuellen Kontakt oder zur aktuellen Spule eingegeben werden.

Ist das Kontrollkästchen **Kommentare pro Kontakt** aktiviert, können Sie mit den Eingabefeldern **Zeilen für Variablenkommentar** einen Kommentar zur Variable eingeben, und in **Zeilen für Variablentext** einen Namen für die Variable vergeben.

Ist das Kontrollkästchen **Netzwerke mit Umbrüchen** aktiviert, bricht das Netzwerk abhängig von der Fenstergöße um.



Drive PLC Developer Studio

Editoren

7.6.5 Der Kontaktplan im Online Modus

- Im Online Modus werden im Kontaktplan alle Kontakte und Spulen, die im Zustand **AN** (TRUE) sind, blau markiert.
- Im Online Modus werden im Kontaktplan alle Leitungen mit dem Zustand TRUE blau markiert.
- An den Ein- und Ausgängen von Funktionsblöcken werden die Werte der entsprechenden Variablen angezeigt.
- Breakpoints können nur auf Netzwerke gesetzt werden. Beim Steppen wird nacheinander zu jedem Netzwerk gesprungen.
- Ist die Ablaufkontrolle aktiv, werden die Nummernfelder der durchlaufenden Netzwerke grün markiert.



Tip!

Steht der Mauszeiger kurze Zeit auf einer Variablen, wird durch einen Tooltip die Adresse, Typ und evtl. Kommentar zu der Variablen angezeigt.



7.7 CFC-Editor

Fehlermeldung beim Speichern



Hinweis!

Sollte es beim Speichern des Projektes zu der Fehlermeldung
Internal error...: A CFC in this project is corrupted, and got restored. Please check its logics.
kommen, gehen Sie bitte wie folgt vor.

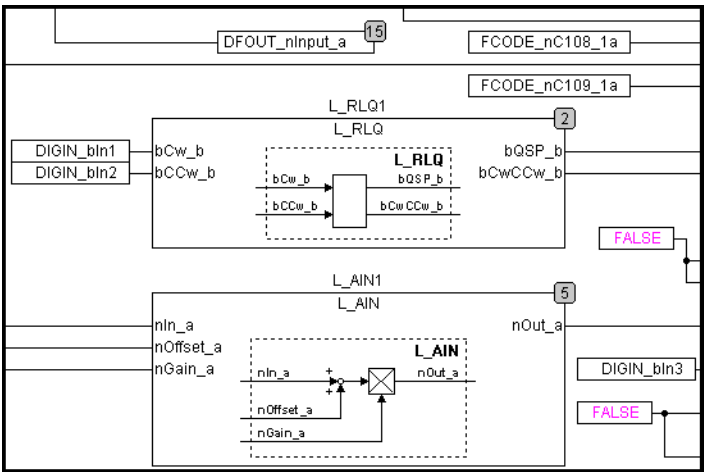
- Die Fehlermeldung nicht wegdrücken.
- Sichern Sie eine vorhandene Sicherungskopie (automatische Sicherung, letzter Arbeitsstand) unter einem anderen Namen.
Die Sicherungskopie durch automatische Sicherung hat die Endung *.asd und befindet sich im selben Ordner in dem sich die *.pro Datei befindet.
- Bestätigen Sie anschließend die Fehlermeldung.
Das DDS versucht den Fehler automatisch zu korrigieren
- Prüfen Sie die Logik des CFC-Editors.
- Werden dabei Fehler festgestellt, benutzen Sie die Sicherungskopie.
Löschen Sie die *.pro Datei und ändern Sie die *.asd Datei in *.pro.

Durch diese Maßnahme haben Sie immer einen sicheren und aktuellen Stand des Projektes.
Im CFC-Editor (*Continuous Function Chart*-Editor) werden keine Netzwerke verwendet, sondern die Elemente können frei platziert werden.



Hinweis!

Wenn Sie mit dem CFC-Editor arbeiten, sollte die automatische Sicherung aktiv sein.
Projekt→Optionen Kategorie *Laden & Speichern*
Nach Erweiterungen oder Änderungen sollten die Arbeitsstände gespeichert werden.





Drive PLC Developer Studio

Editoren

Folgende Elemente können im CFC-Editor verwendet werden:

- Baustein
- Eingang
- Ausgang
- Sprung
- Label
- Return
- Kommentar

Die wichtigsten Befehle zum CFC-Editor finden Sie auch im Kontextmenü (rechte Maustaste)

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 CFC-Editors gegenüber dem gewöhnlichen Funktionsplaneditor ist, daß Rückkopplungen direkt eingefügt werden können.

7.7.1 Cursorpositionen im CFC

	Jedes Textfeld (Selektierter Text ist farbig hinterlegt und kann geändert werden.)
	Der Rumpf der Elemente Baustein, Eingang, Ausgang, Sprung, Label, Return und Kommentar
	Eingänge der Elemente Baustein, Ausgang, Sprung und Return
	Ausgänge der Elemente Baustein und Eingang

7.7.2 Elemente selektieren

Um ein Element zu selektieren, klicken Sie auf den Rumpf des Elements.

- Um mehrere Elemente zu markieren, halten Sie die **<Umschalt>**-Taste gedrückt und klicken 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 Sie alle Elemente selektieren.

7.7.3 Elemente kopieren, löschen

Zum Kopieren und Löschen von ein oder mehreren selektierten Elementen stehen Ihnen die Standard-Befehle der Windows-Zwischenablage im Menü **Bearbeiten** zur Verfügung. (7-2)



7.7.4 Elemente verschieben

Ein oder mehrere selektierte Elemente können Sie mit den Pfeiltasten bei gedrückter **<Umschalt>**-Taste verschieben.

Desweiteren können Sie die Elemente mit gedrückter linker Maustaste verschieben.

- Diese Elemente werden durch Loslassen der linken Maustaste an der aktuellen Position abgelegt.
- Wenn die verschobenen Elemente an dieser Position andere Elemente überdecken oder die vorgesehene Größe des Editors überschreiten, wird das Verschieben der Elemente nicht ausgeführt und es ertönt ein Warnton.

7.7.5 Befehle im Menü “Einfügen”

7.7.5.1 Baustein

Symbol: 	Menübefehl: Einfügen→Baustein	Tastaturbefehl: <Strg>+
---	--------------------------------------	---

Dieser Befehl fügt im CFC-Editor einen Operator ein.

Der eingefügte Operator ist stets ein **AND**.

- 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 (**<F2>**) 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 samt der davorliegenden Äste gelöscht.

7.7.5.2 Eingang

Symbol: 	Menübefehl: Einfügen→Eingang	Tastaturbefehl: <Strg>+<E>
---	-------------------------------------	---

Dieser Befehl fügt im CFC-Editor einen Eingang ein.

Der eingefügte Eingang ist mit dem Text "???" vorbelegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Konstante bzw. Variable zu ändern. Sie können dazu auch die Eingabehilfe (**<F2>**) verwenden.

7.7.5.3 Ausgang

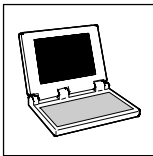
Symbol: 	Menübefehl: Einfügen→Ausgang	Tastaturbefehl: <Strg>+<A>
---	-------------------------------------	---

Dieser Befehl fügt im CFC-Editor einen Ausgang ein.

Der eingefügte Ausgang ist mit dem Text "???" vorbelegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Variable zu ändern. Sie können dazu auch die Eingabehilfe (**<F2>**) verwenden.

Mit dem Element Ausgang kann der übergebene Wert einer entsprechenden Variable gespeichert werden.



Drive PLC Developer Studio

Editoren

7.7.5.4 Sprung

Symbol: 	Menübefehl: Einfügen→Sprung	Tastaturbefehl: <Strg>+<J>
---	------------------------------------	---

Dieser Befehl fügt im CFC-Editor einen Sprung ein.

Der eingefügte Sprung ist mit dem Text "???" vorbelegt.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Sprungmarke, zu der gesprungen werden soll zu ändern.
- Eine Sprungmarke fügen Sie mit **Einfügen→Marke** ein.

7.7.5.5 Marke

Symbol: 	Menübefehl: Einfügen→Marke	Tastaturbefehl: <Strg>+<L>
---	-----------------------------------	---

Dieser Befehl fügt im CFC-Editor eine Sprungmarke ein.

- Klicken Sie auf diesen Text, um ihn in die gewünschte Sprungmarke zu ändern.
- Einen Sprung zu dieser Sprungmarke fügen Sie mit **Einfügen→Sprung** ein.

7.7.5.6 Return

Symbol: 	Menübefehl: Einfügen→Return	Tastaturbefehl: <Strg>+<R>
---	------------------------------------	---

Dieser Befehl fügt im CFC-Editor eine RETURN-Anweisung ein.

7.7.5.7 Kommentar

Symbol: 	Menübefehl: Einfügen→Kommentar	Tastaturbefehl: <Strg>+<K>
---	---------------------------------------	---

Dieser Befehl fügt im CFC-Editor einen Kommentar ein.

- Mit **<STRG>+<Eingabetaste>** fügen Sie einen Zeilenwechsel im Kommentar ein.

7.7.5.8 Bausteineingang

Symbol: -	Menübefehl: Einfügen→Bausteineingang	Tastaturbefehl: <Strg>+<U>
-----------	---	---

Dieser Befehl fügt im CFC-Editor einen Bausteineingang ein.

Die Zahl der Eingänge ist bei vielen Operatoren variabel (z. B. kann der Baustein ADD zwei oder mehr Eingänge haben).

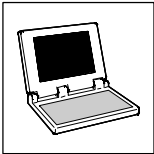
- Um einen solchen Operator um einen Eingang zu erweitern, muß der Operator selbst selektiert werden.

7.7.5.9 In-Pin / Out-Pin

Symbol: 	Menübefehl: Einfügen→In-Pin	Tastaturbefehl: -
Symbol: 	Menübefehl: Einfügen→Out-Pin	Tastaturbefehl: -

Diese Befehle stehen nur bei einem geöffneten Makro zu Verfügung.

- Sie unterstützen das Einfügen von In- bzw. Out-Pins als Ein- und Ausgänge eines Makros.
- Sie erhalten keinen Positionsindex.



7.7.6 Befehle im Menü “Extras”

7.7.6.1 Negieren

Symbol: 	Menübefehl: Extras→Negieren	Tastaturbefehl: <Strg>+<N>
---	------------------------------------	---

Mit diesem Befehl können im CFC-Editor Eingänge, Ausgänge, Sprünge oder RETURN-Anweisungen negiert werden.

Das Symbol für die Negation ist ein kleiner Kreis auf einer Verbindung.

- Wenn ein Eingang selektiert ist, dann wird dieser Eingang negiert.
- Wenn ein Ausgang selektiert ist, 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.

7.7.6.2 Set/Reset

Symbol: 	Menübefehl: Extras→Set/Reset	Tastaturbefehl: <Strg>+<T>
---	-------------------------------------	---

Mit diesem Befehl können im CFC-Editor Ausgänge als Set- bzw. Reset-Ausgänge definiert werden.

Beim Set-Ausgang wird dessen Eingang mit [S] und beim Reset-Ausgang dessen Eingang mit [R] dargestellt.

- Ein Set-Ausgang wird auf **TRUE** gesetzt, wenn an dessen Eingang **TRUE** anliegt. Der Ausgang behält nun diesen Wert, auch wenn dessen Eingangszustand auf **FALSE** wechselt.
- Ein Reset Ausgang wird auf **FALSE** gesetzt, wenn an dessen Eingang **TRUE** anliegt. Der Ausgang behält nun diesen Wert, auch wenn dessen Eingangszustand auf **FALSE** wechselt.

Bei mehrfachen Ausführen des Befehls wechselt der Ausgang zwischen Set-, Reset- und normalen Ausgang.

7.7.6.3 Alles Markieren

Symbol: -	Menübefehl: Extras→Alles Markieren	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl markiert alle Objekte (Bausteine, Anweisungen) im aktiven CFC-Editor. Die markierten Objekte können dann gleichzeitig editiert werden.

- Ausschneiden
- Rückgängig
- Verschieben
Dazu muß in ein Objekt mit der linken Maustaste angeklickt werden. Alle Objekte können gleichzeitig verschoben werden.
- Negationen
Bei Objekten mit Eingängen werden Negationen erzeugt.

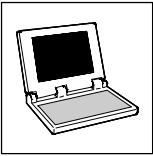
7.7.6.4 EN/ENO

Symbol: 	Menübefehl: Extras→EN/ENO	Tastaturbefehl: <Strg>+<I>
---	----------------------------------	---

Dieser Befehl erweitert im CFC-Editor einen selektierten Baustein um einen zusätzlichen Freigabe-Eingang EN und einen Freigabe-Ausgang ENO für Variable vom Typ **BOOL**.

- Ein Baustein mit EN-Eingang/ENO-Ausgang wird nur ausgeführt, wenn am EN-Eingang der Zustand **TRUE** anliegt.
- Nach Ausführung eines Bausteins mit EN-Eingang/ENO-Ausgang liefert der ENO-Ausgang den Wert **TRUE**, ansonsten **FALSE**.

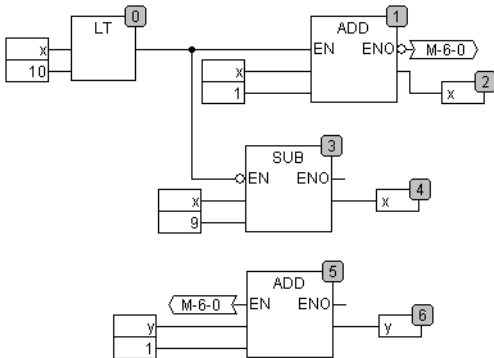
Auch EN-Eingänge/ENO-Ausgänge können mit **Extras→Negieren** negiert werden.



Drive PLC Developer Studio

Editoren

Beispiel für eine Verkettung von Freigaben:



Die Nummern in der rechten oberen Ecke der Bausteine geben die Abarbeitungsreihenfolge an.

1. x soll in diesem Beispiel mit 1 und y mit 0 initialisiert sein.
2. SUB(3) und ADD(5) sind nicht freigeschaltet.
3. Solange $x < 10$ ist, ist ADD(1) freigeschaltet und erhöht x jeweils um 1 ($x=x+1$).
4. Bei $x = 10$ liefert der Ausgang des Bausteins LT(0) FALSE und schaltet SUB(3) frei.
5. ADD(1) wird gesperrt und schaltet mit seinen ENO-Ausgang ADD(5) frei.
6. SUB(3) und ADD(5) werden ausgeführt, x wird also auf den Wert 1 zurückgesetzt ($x=x-9$) und y wird um 1 erhöht ($y=y+1$).
7. SUB(3) und ADD(5) werden durch LT(0) wieder gesperrt und der Kreislauf beginnt erneut bei Punkt 3.

In diesem Beispiel zählt y also, wie oft x die Werte 1 bis 10 durchläuft.

7.7.6.5 Eigenschaften

Symbol: -	Menübefehl: Extras→Eigenschaften	Tastaturbefehl: -
-----------	----------------------------------	-------------------

Dieser Befehl öffnet im CFC-Editor das Dialogfeld *Parameter bearbeiten*, in dem konstante Eingangsparameter (**VAR_INPUT CONSTANT**) von Funktionen und Funktionsblöcken des selektierten Bausteins angezeigt und in ihrem Wert verändert werden können.

- Diese Eingangsparameter werden im CFC-Editor nicht direkt angezeigt.
- Statt den Rumpf des Bausteins zu selektieren und **Extras→Eigenschaften** zu wählen, können Sie das Dialogfeld *Parameter bearbeiten* auch durch einen Doppelklick auf den Rumpf des entsprechenden Bausteins öffnen.

Um den Wert eines konstanten Eingangsparameters (**VAR_INPUT CONSTANT**) im Dialogfeld *Parameter bearbeiten* zu verändern, markieren Sie den entsprechenden Eintrag in der Spalte "Wert".

- Durch erneuten Mausklick oder durch Drücken der **<Leertaste>** kann dieser Wert bearbeitet werden.
- Mit der **<Eingabetaste>** bestätigen Sie die Änderung eines Wertes, mit der **<Esc>**-Taste können Sie die Änderungen verwerfen.

Mit der Schaltfläche **OK** verlassen Sie das Dialogfeld und speichern dabei alle Änderungen, mit **Abbrechen** verlassen Sie das Dialogfeld und verwerfen dabei alle durchgeführten Veränderungen.

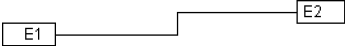
7.7.6.6 **Verbindungsmarke**

Symbol: -	Menübefehl: Extras→Verbindungsmarke	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl verbindet im CFC-Editor Elemente statt über Verbindungslinien mit Hilfe von Verbindungsmarken (Konnektoren).

- Dabei werden der Ausgang und der zugehörige Eingang mit einer Verbindungsmarke mit eindeutigen Namen versehen.
- Der Name der Verbindungsmarke wird automatisch vom Programm vergeben, kann aber verändert werden.

Liegt bereits eine Verbindung zwischen zwei Elementen vor, die nun in eine Verbindung mit Hilfe von Verbindungsmarken umgewandelt werden soll, selektieren Sie zunächst den Ausgang, an dem die Verbindung startet und wählen Sie dann **Extras→Verbindungsmarke**.

Vor Ausführung des Befehls:	Nach Ausführung des Befehls:
	



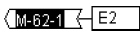
Tip!

Verbindungen über Verbindungsmarken können Sie wieder in gewöhnliche Verbindungen umwandeln, indem Sie den Ausgang der Verbindung selektieren und den Befehl **Extras→Verbindungsmarke** erneut ausführen.

Ändern des Namen der Verbindungsmarke

Beachten Sie, daß der Name der Verbindungsmarke eine Eigenschaft des Ausgangs einer Verbindung ist und mit dieser abgespeichert wird.

Editieren der Verbindungsmarke an einem Eingang:



- Wenn Sie den Text in der Verbindungsmarke verändern, wird der neue Name bei allen zugehörigen Verbindungsmarken an Eingängen übernommen.
- Sie können den Text jedoch nicht in einen Namen ändern, der bereits zu einer anderen Verbindungsmarke gehört, da sonst die Eindeutigkeit des Namens nicht mehr vorliegen würde.

Editieren der Verbindungsmarke an einem Ausgang:

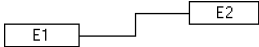
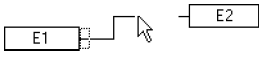


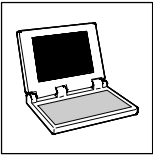
- Wird der Text im Konnektor, ändert er sich auch in der zugehörigen Verbindungsmarke am anderen Baustein.

7.7.7 **Verbindungen erstellen**

Ein- und Ausgänge von Elementen können durch Ziehen einer Verbindung mit gedrückt gehaltener linker Maustaste verbunden werden.

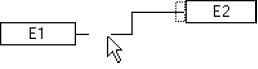
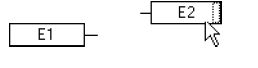
- Ein Eingang eines Elementes kann mit genau einem Ausgang eines Elementes verbunden werden.
- Ein Ausgang eines Elementes kann mit mehreren Eingängen von Elementen verbunden werden.

	Es gibt mehrere Möglichkeiten, den Eingang des Elementes E2 mit dem Ausgang des Elementes E1 zu verbinden:
	1. Mit der linken Maustaste auf den Ausgangs des Elements E1 klicken, die linke Maustaste gedrückt halten, den Mauszeiger auf den Eingang des Elements E2 ziehen und dort die linke Maustaste loslassen. – Während des Ziehvorganges mit der Maus wird eine Verbindung vom Ausgang des Elements E1 zum Mauszeiger gezeichnet.



Drive PLC Developer Studio

Editoren

	2. 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.
	3. Eines der Elemente E1 oder E2 so verschieben, daß sich der Ausgang von Element E1 und der Eingang von Element E2 berühren.
	4. Falls das Element E2 ein Baustein mit einem freien Eingang ist, kann mit dem Mauszeiger auch eine Verbindung von einem Ausgang von E1 in den Rumpf von E2 gezogen werden. – Beim Loslassen der linken 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 der Methoden 1, 2 und 4 können auch der Ausgang und Eingang eines Bausteins miteinander verbunden werden (Rückkopplung).

	Verbindung zwischen zwei Pins
	– Mit der linken Maustaste auf einen Pin klicken und die Taste gedrückt halten. – Die Maus auf den gewünschten Pin ziehen und die Maustaste loslassen.

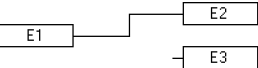
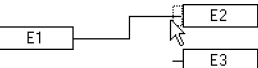
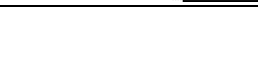


Tip!

Es ist darauf zu achten, daß Sie das Editorfenster beim ziehen versehentlich nicht verlassen. Es wird dann gescrollt.

Für einfache Datentypen erfolgt während dem verbinden eine Typprüfung. Sind die Typen nicht kompatibel ist keine Verbindung möglich.

7.7.8 Verbindungen ändern

	Eine Verbindung zwischen dem Ausgang des Elementes E1 und dem Eingang des Elementes E2 kann leicht in eine Verbindung zwischen dem Ausgang von E1 und dem Eingang von Element E3 geändert werden:
	Mit dem Mauszeiger auf den Eingang von E2 klicken und mit gedrückt gehaltener linker Maustaste die Verbindung vom Eingang E2 weg auf den Eingang E3 ziehen.
	
	
	

7.7.9 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 und die **<Entf>**-Taste betätigen 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 und die **<Entf>**-Taste betätigen oder den Befehl **Bearbeiten→Löschen** ausführen.
- Den Eingang von E2 selektieren und mit gedrückter linker Maustaste die Verbindung vom Eingang E2 wegziehen. Wird die linke Maustaste dann in einem freien Bereich losgelassen, so wird die Verbindung gelöscht.



7.7.10 Rückkopplungen

Im CFC-Editor können im Gegensatz zum gewöhnlichen Funktionsplaneditor Rückkopplungen direkt dargestellt werden.

Beachten Sie, 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.
- Bei Konstanten ergibt sich der Datentyp 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`, der zweite Eingang ist aufgrund der Rückkopplung undefiniert. Somit ist auch die Zwischenvariable vom Typ `SINT`. Der Wert der Zwischenvariable wird erst danach der Ausgangsvariablen zugewiesen.

Untenstehende Abbildung zeigt eine Addition mit Rückkopplung und eine Addition direkt mit einer Variablen. Die Variablen x und y sollen dabei vom Typ `INT` sein:

Addition mit Rückkopplung:	Addition direkt mit einer Variablen:

Zwischen den beiden Additionen gibt es folgende Unterschiede:

- Die Variable y kann mit einem Wert ungleich 0 initialisiert werden, die Zwischenvariable der Addition mit Rückkopplung jedoch nicht.
- Die Zwischenvariable der Addition mit Rückkopplung hat den Datentyp `SINT`, die der Addition mit der Variable y den Datentyp `INT`.
- Die Variablen x und y haben ab dem 128ten Aufruf unterschiedliche Werte. Die Variable x, obwohl vom Typ `INT`, erhält den Wert -128, weil die Zwischenvariable einen Überlauf hat. Die Variable y erhält dagegen den Wert 128.

7.7.11 Abarbeitungsreihenfolge

Beim CFC-Editor 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.
- Ist die Anzeige der Reihenfolge aktiviert, so erscheint bei den Elementen in der rechten oberen Ecke in einem grauen Feld die jeweilige Abarbeitungsnummer.

Anzeige der Abarbeitungsreihenfolge eingeschaltet:	Anzeige der Abarbeitungsreihenfolge ausgeschaltet:



Drive PLC Developer Studio

Editoren

7.7.12 Befehle im Menü “Extras” im Untermenü “Reihenfolge”

7.7.12.1 Anzeigen

Symbol: -	Menübefehl: Extras→Reihenfolge→anzeigen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl schaltet im CFC-Editor die Anzeige der Abarbeitungsreihenfolge ein bzw. aus.

- Standardmäßig ist die Anzeige eingeschaltet.
- Bei eingeschalteter Anzeige erscheint vor dem Menübefehl ein Haken.

7.7.12.2 Topologisch anordnen

Symbol: -	Menübefehl: Extras→Reihenfolge→topologisch anordnen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl ordnet im CFC-Editor die selektierten Elemente in topologischer Reihenfolge an. 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 hierbei keine Rolle, es zählt nur die Lage der Elemente.

Arbeitsweise

1. Alle selektierten Elemente werden aus der Abarbeitungsliste herausgenommen.
2. Die verbliebene Abarbeitungsliste wird aktualisiert, so daß wieder eine durchgängige Numerierung besteht.
3. Die selektierten Elemente werden dann einzeln von rechts unten nach links oben wieder in die verbliebende Abarbeitungsliste eingefügt.
4. Jedes markierte Element wird dabei in der Abarbeitungsliste mit der Reihenfolge-Zahl des topologischen Nachfolgers eingefügt und die Reihenfolge-Zahl aller topologischen Nachfolger wird um 1 erhöht.

Beispiel zur Arbeitsweise des Programms

	Die selektierten Elemente E1, E2 und E3 sollen topologisch angeordnet werden.
	Alle selektierten Elemente werden aus der Abarbeitungsliste herausgenommen und die verbliebene Abarbeitungsliste wird so aktualisiert, daß wieder eine durchgängige Numerierung entsteht.
	Die selektierten Elemente werden dann einzeln von rechts unten nach links oben wieder in die verbliebende Abarbeitungsliste eingefügt. Zuerst wird also Element E3 mit der Reihenfolge-Zahl des topologischen Nachfolgers (E5) eingefügt und die Reihenfolge-Zahl aller topologischen Nachfolger (E5, E6) um 1 erhöht.
	Element E2 wird mit der Reihenfolge-Zahl des topologischen Nachfolgers (E6) eingefügt und die Reihenfolge-Zahl aller topologischen Nachfolger (E6) wird um 1 erhöht.



	Element E1 wird mit der Reihenfolge-Zahl des topologischen Nachfolgers (E2) eingefügt und die Reihenfolge-Zahl aller topologischen Nachfolger (E2, E6) wird um 1 erhöht.
	Nach Ausführung des Befehls hat dieses Beispiel nebenstehende Abarbeitungsreihenfolge.

7.7.12.3 Alles nach Datenfluß anordnen

Symbol: -	Menübefehl: Extras→Reihenfolge→Alles nach Datenfluß anordnen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl ordnet im CFC-Editor alle Elemente nach ihrem Datenfluß an.

- Dieser Befehl wird **auf alle** Elemente angewandt.
- Die Abarbeitungsreihenfolge wird bestimmt vom Datenfluß der Elemente und nicht von ihrer Lage.

Vor Ausführung des Befehls:	Nach Ausführung des Befehls:

7.7.12.4 Eins vor

Symbol: -	Menübefehl: Extras→Reihenfolge→Eins vor	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl schiebt im CFC-Editor alle selektierten Elemente innerhalb der Abarbeitungsliste um einen Platz nach vorne.

- Ausgenommen sind die Elemente, die sich schon am Anfang der Abarbeitungsliste befinden.

7.7.12.5 Eins zurück

Symbol: -	Menübefehl: Extras→Reihenfolge→Eins zurück	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl schiebt im CFC-Editor alle selektierten Elemente innerhalb der Abarbeitungsliste um einen Platz nach hinten.

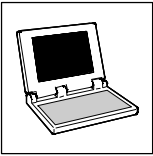
- Ausgenommen sind die Elemente, die sich schon am Ende der Abarbeitungsliste befinden.

7.7.12.6 An den Anfang

Symbol: -	Menübefehl: Extras→Reihenfolge→An den Anfang	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl schiebt im CFC-Editor alle selektierten Elemente an den Anfang der Abarbeitungsliste.

- Die Reihenfolge innerhalb der selektierten Elemente bleibt erhalten.
- Auch die Reihenfolge innerhalb der nicht-selektierten Elemente bleibt erhalten.



Drive PLC Developer Studio

Editoren

7.7.12.7
Ans Ende

Symbol: -	Menübefehl: Extras→Reihenfolge→Ans Ende	Tastaturbefehl: -
-----------	---	-------------------

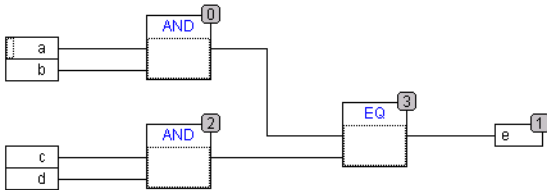
- Dieser Befehl schiebt im CFC-Editor alle selektierten Elemente an das Ende der Abarbeitungsliste.
- Die Reihenfolge innerhalb der selektierten Elemente bleibt erhalten.
 - Auch die Reihenfolge innerhalb der nicht-selektierten Elemente bleibt erhalten.

7.7.12.8
Makro erzeugen

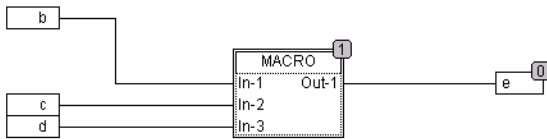
Symbol:	Menübefehl: Extras→Makro erzeugen	Tastaturbefehl: -
---------	-----------------------------------	-------------------

- Mit diesem Befehl können mehrere Bausteine zu einem Block, der als Makro mit einem Namen versehen werden kann, zusammengefaßt werden.
- Makros werden mit kopieren und einfügen vervielfältigt. Jede Kopie ist ein eigenes Makro mit frei wählbarem Namen. Makros sind keine Referenz.
 - Durch Makros getrennte Verbindungen erzeugen In- bzw. Out-Pins am Makro. Verbindungen zu Eingängen erzeugen z. B. einen In-Pin. Neben dem erzeugten In-Pin erscheint der Default-Name in Form In-<n> und bei Outputs erscheint Out-<n>.
 - Betroffene Verbindungen, die vor der Erzeugung des Makros Verbindungsmarken hatten, erhalten die Verbindungsmarke am PIN des Makros.
 - Ein Makro erhält zunächst den Default-Name *MACRO*. Dieser kann geändert werden.

In dem Beispiel sind die drei Bausteine und ein Eingang selektiert.



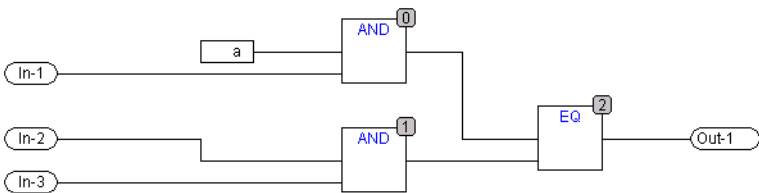
Mit dem Menübefehl **Extras→Makro erzeugen** wird folgender Makro erzeugt. Durch Verschieben der Bausteine kann die Einheit effizienter dargestellt werden.

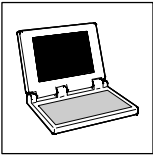


7.7.12.9
In Makro springen

Symbol: -	Menübefehl: Extras→In Makro springen	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

Durch diesen Menübefehl oder durch Doppelklick auf den Rumpf des Makros wird das Makro im Editorfenster zum Bearbeiten geöffnet.



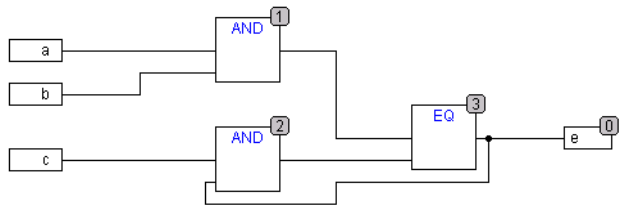


- Die erzeugten Pin-Boxen können wie normale Bausteine bearbeitet werden. Sie unterscheiden sich in der Darstellung und besitzen keinen Positionsindex.
- Pin-Boxen haben abgerundete Ecken. Der Text der Pin-Box entspricht dem Namen des Pins in der Makrodarstellung.
- Die Reihenfolge der Pins an der Makro-Box richtet sich nach der Abarbeitungsreihenfolge der Elemente des Makros.
 - niedriger Reihenfolgenindex vor hohem
 - oberer Pin vor unterem
- Die Abarbeitungsreihenfolge im Makro ist eine geschlossene Prozedur. Das Makro wird, in seinem übergeordneten Baustein, als ein Block behandelt. Befehle wirken nur im Makro.

7.7.12.10 Makro expandieren

Symbol: -	Menübefehl: Extras→Makro expandieren	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

Mit diesem Menübefehl wird das selektierte Makro wieder expandiert und die einzelnen Elemente substituieren das Makro wieder.



- Die Verbindungen zu den In- Out-Pins werden durch Ein- Ausgänge der Elemente dargestellt.
- Ist im Editor durch bei Expansion nicht ausreichend Platz wird die Einheit automatisch verschoben.
- Nach der Expansion können die Elemente wieder in effiziente Positionen gebracht werden.



Hinweis!

Wird das Projekt in eine anderer Sprache konvertiert, werden alle Makros expandiert.

7.7.12.11 Eine Makroebene zurück

Symbol: 	Menübefehl: Extras→Eine Makroebene zurück	Tastaturbefehl: -
---	---	-------------------

Dieser Befehl ist aktiv, wenn in ein Makro gesprungen wurde, bzw. das Makro zur Bearbeitung geöffnet ist.

Bei verschachtelten Makros ist es dann möglich eine Makroebene zurückzuspringen.

7.7.12.12 Alle Makroebenen zurück

Symbol: 	Menübefehl: Extras→Alle Makroebenen zurück	Tastaturbefehl: -
---	--	-------------------

Dieser Befehl ist aktiv, wenn in ein Makro gesprungen wurde, bzw. das Makro zur Bearbeitung geöffnet ist.

Bei verschachtelten Makros ist es dann möglich alle Makroebenen zurückzuspringen.

7.7.12.13 Instanz öffnen

Symbol: -	Menübefehl: Extras→Instanz öffnen	Tastaturbefehl: -
-----------	-----------------------------------	-------------------

Dieser Befehl gleicht dem Menübefehl **Projekt→Instanz öffnen**.



Drive PLC Developer Studio

Editoren

7.7.12.14 Zoom

Symbol: -	Menübefehl: Extras→Zoom	Tastaturbefehl: <Alt>+<Enter>
-----------	--------------------------------	--

Die Implementierung eines markierten Bausteins kann mit diesem Menübefehl geöffnet werden.

7.7.13 CFC im Online Modus

Monitoring

Die Werte für Ein- und Ausgänge werden innerhalb der Input- bzw. Output Boxen dargestellt. Konstanten werden dabei nicht gemonitort (sichtbar). Für nicht boolsche Variablen werden die Boxen entsprechen den angezeigten Werten vergrößert. Ist der Wert TRUE werden in boolschen Verbindungen Variablennamen und die Verbindung blau markiert.

Interne boolsche Verbindungen werden Online auch blau markiert.

Breakpoints

Haltepunkte können auf alle Elemente gesetzt werden, die einen Abarbeitungsreihenfolgen-Index besitzen. Die Abarbeitung des Programms wird vor dem Ausführen des jeweiligen Elements angehalten. Als Position für den Haltepunkt während des Breakpoint-Dialogs wird der Abarbeitungsreihenfolgen-Index des Elements verwendet.

Setzen der Haltepunkte auf ein selektiertes Element durch

- Taste F9
- **Online→Breakpoint Ein/Aus**
- das Kontext-Menü des CFC-Editors

Nach wiederholten Ausführen des Menübefehls **Online→Breakpoint Ein/Aus** wird der Breakpoint wieder gelöscht. Der Haltepunkt kann durch Doppelklick auf des Element getoggelt werden.

Die Darstellung der Breakpoints erfolgt unter **Projekt→Optionen** Kategorie *Farben*.

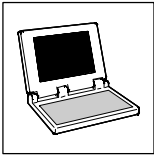
Return-Marke

Im Online-Modus wird automatisch eine Sprungmarke mit der Bezeichnung RETURN in der ersten Spalte und nach dem letzten Element im CFC-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

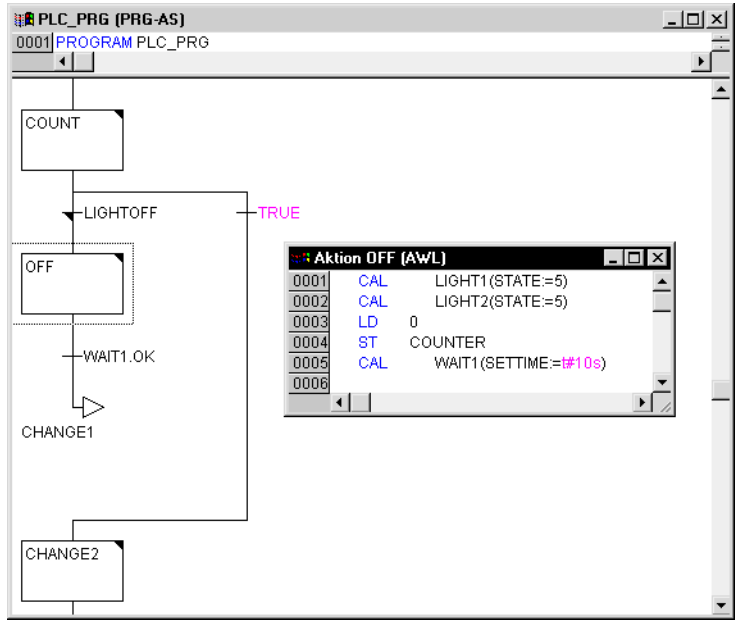
Ist der Menübefehl **Online→Einzelschritt über** aktiv wird immer zum Element mit dem höheren Reihenfolgen-Index gesprungen.

Ist der Menübefehl **Online→Einzelschritt in** aktiv wird bei einem Makro oder Baustein in die Implementierung verzweigt.



7.8 Ablaufspracheneditor

Der Ablaufspracheneditor ist ein grafischer Editor, mit dem die zeitliche Abfolge verschiedener Aktionen innerhalb eines Programms beschrieben werden kann.



Tip!

Die wichtigsten Befehle zum Ablaufspracheneditor finden Sie auch im Kontextmenü (rechte Maustaste)
Tooltips zeigen den gesamten Namen von Transitionen, Aktionen, usw. an.

7.8.1 Blöcke markieren

Ein markierter Block ist eine Menge von AS-Elementen, die von einem gepunkteten Rechteck umgeben sind.

- Sie können ein Element (einen Schritt, eine Transition oder einen Sprung) auswählen, indem Sie auf dieses Element klicken oder die Pfeiltasten verwenden.
- Um eine Menge von mehreren Elementen zu markieren, drücken Sie zusätzlich zu einem bereits markierten Element die **<Umschalt>**-Taste 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.



Tip!

Beachten Sie, daß alle Befehle nur dann ausgeführt werden können, wenn sie nicht den Sprachkonventionen widersprechen.



Drive PLC Developer Studio

Editoren

7.8.2 Befehle im Menü “Einfügen”

7.8.2.1 Schritt-Transition (davor)

Symbol: 	Menübefehl: Einfügen→Schritt-Transition (davor)	Tastaturbefehl: <Strg>+<T>
---	--	---

Dieser Befehl fügt im AS-Editor einen Schritt gefolgt von einer Transition vor dem markierten Block ein.



Tip!

Sie können einen Schritt gefolgt von einer Transition wieder entfernen, indem Sie den Schritt **und** die Transition markieren und die **<Entf>**-Taste betätigen.
(Halten Sie zum Markieren mehrerer Objekte die **<Strg>**-Taste gedrückt.)

7.8.2.2 Schritt-Transition (danach)

Symbol: 	Menübefehl: Einfügen→Schritt-Transition (danach)	Tastaturbefehl: <Strg>+<E>
---	---	---

Dieser Befehl fügt im AS-Editor einen Schritt gefolgt von einer Transition nach der ersten Transition im markierten Block ein.



Tip!

Sie können einen Schritt gefolgt von einer Transition wieder entfernen, indem Sie den Schritt **und** die Transition markieren und die **<Entf>**-Taste betätigen.
(Halten Sie zum Markieren mehrerer Objekte die **<Strg>**-Taste gedrückt.)

7.8.2.3 Alternativzweig (rechts)

Symbol: 	Menübefehl: Einfügen→Alternativzweig (rechts)	Tastaturbefehl: <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.

7.8.2.4 Alternativzweig (links)

Symbol: 	Menübefehl: Einfügen→Alternativzweig (links)	Tastaturbefehl: -
---	---	--------------------------

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.



7.8.2.5 Parallelzweig (rechts)

Symbol:	Menübefehl: Einfügen→Parallelzweig (rechts)	Tastaturbefehl: <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.

7.8.2.6 Parallelzweig (links)

Symbol:	Menübefehl: Einfügen→Parallelzweig (links)	Tastaturbefehl: -
---------	---	-------------------

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 versehen werden.

7.8.2.7 Sprung

Symbol:	Menübefehl: Einfügen→Sprung	Tastaturbefehl: <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.
- Die Sprungmarke **Step** muß durch den Namen des Schrittes ersetzt werden, zu dem gesprungen werden soll.

7.8.2.8 Transition-Sprung

Symbol:	Menübefehl: Einfügen→Transition-Sprung	Tastaturbefehl: -
---------	---	-------------------

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.
- Die Sprungmarke **Step** muß durch den Namen des Schrittes ersetzt werden, zu dem gesprungen werden soll.

7.8.2.9 Eingangsaktion hinzufügen

Symbol: -	Menübefehl: Einfügen→Eingangsaktion hinzufügen	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl fügt zu einem Schritt eine Eingangsaktion hinzu.

- 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.



Drive PLC Developer Studio

Editoren

7.8.2.10 Ausgangsaktion hinzufügen

Symbol: -	Menübefehl: Einfügen→Ausgangsaktion hinzufügen	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl fügt zu einem Schritt eine Ausgangsaktion hinzu.

- 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.

7.8.3 Befehle im Menü "Extras"

7.8.3.1 Parallelzweig einfügen (rechts)

Symbol: -	Menübefehl: Extras→Parallelzweig einfügen (rechts)	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl fügt den Inhalt der Zwischenablage als rechte Parallelverzweigung des markierten Blocks ein.

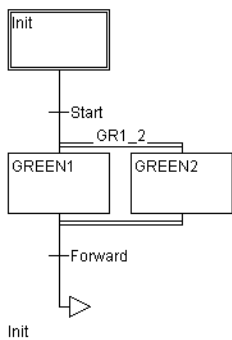
- Hierfür muß der markierte Block mit einem Schritt beginnen und enden.
- Der Inhalt der Zwischenablage muß ebenfalls ein AS-Block sein, der mit einem Schritt beginnt und endet.

7.8.3.2 Marke zu Parallelzweig hinzufügen

Symbol: -	Menübefehl: Extras→Marke zu Parallelzweig hinzufügen	Tastaturbefehl: -
-----------	---	-------------------

Wenn eine eingefügte Parallelverzweigung mit einer Sprungmarke versehen werden soll, muß die Transition direkt vor der eingefügten Parallelverzweigung markiert sein.

- Führen Sie den Menübefehl **Extra→Marke zu Parallelverzweigung hinzufügen** aus.
- Es wird der Name Parallel 1, 2 usw. vergeben. Dieser Name kann editiert werden. In dem Beispiel wurde er GR_1_2 genannt.



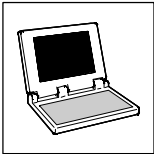
- Eine Sprungmarke wird durch entfernen des Textes der Sprungmarke gelöscht.

7.8.3.3 Einfügen danach

Symbol: -	Menübefehl: Extras→Einfügen danach	Tastaturbefehl: -
-----------	---	-------------------

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).

- Der Befehl wird nur dann ausgeführt, wenn die resultierende AS-Struktur nach den Sprachnormen korrekt ist.



7.8.3.4 Zoom Aktion/Transition

Symbol: -	Menübefehl: Extras→Zoom Aktion/Transition	Tastaturbefehl: <Alt>+<Eingabetaste>
-----------	---	--------------------------------------

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.

7.8.3.5 Lösche Aktion/Transition

Symbol: -	Menübefehl: Extras→Lösche Aktion/Transition	Tastaturbefehl: -
-----------	---	-------------------

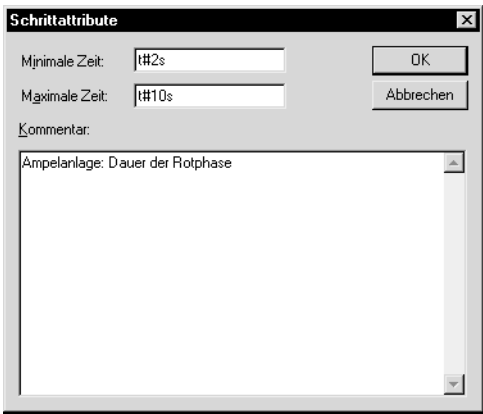
Dieser Befehl löscht die Aktionen des ersten Schritts des markierten Blocks bzw. den Transitionsrumpf der ersten Transition des markierten Blocks.

- Ist bei einem Schritt entweder die Aktion, die Eingangsaktion oder die Ausgangsaktion implementiert, wird diese mit dem Befehl gelöscht. Andernfalls erscheint ein Dialogfeld, 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 assoziierten Aktion selektiert, wird diese Assoziation gelöscht. Bei einem IEC-Schritt mit mehreren Aktionen erscheint ein Dialogfeld zur Auswahl.

7.8.3.6 Schritt Attribute

Symbol: -	Menübefehl: Extras→Schritt Attribute	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

Dieser Befehl öffnet einen Dialog, in dem Sie Attribute zu dem markierten Schritt editieren können.



Sie können drei verschiedene Einträge im Schrittattribute-Dialog vornehmen:

- Im Eingabefeld *Minimale Zeit* geben Sie die Zeitdauer ein, die die Abarbeitung dieses Schritts mindestens dauern soll. (Typ **TIME**)
Beachten Sie: Auch wenn die nachfolgende Transition **TRUE** ist, bleibt der Schritt diese Zeit lang aktiv.
- Im Eingabefeld *Maximale Zeit* geben Sie die Zeitdauer ein, die die Abarbeitung des Schrittes höchstens dauern soll. (Typ **TIME**)
- Im Eingabefeld *Kommentar* können Sie einen Kommentar zum Schritt eingeben.



Drive PLC Developer Studio

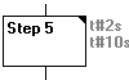
Editoren

Über **Extras→Optionen** können Sie einstellen, ob im AS-Editor Kommentare oder Zeiteinstellungen zu Ihren Schritten dargestellt werden soll. Rechts neben dem Schritt erscheinen dann entweder der Kommentar oder die Zeiteinstellungen.



Tip!
Bei Überschreiten der maximalen Zeit werden AS-Flags gesetzt, die der Benutzer abfragen kann. Im Online-Modus wird angezeigt, wie lange der Schritt bereits aktiv ist.

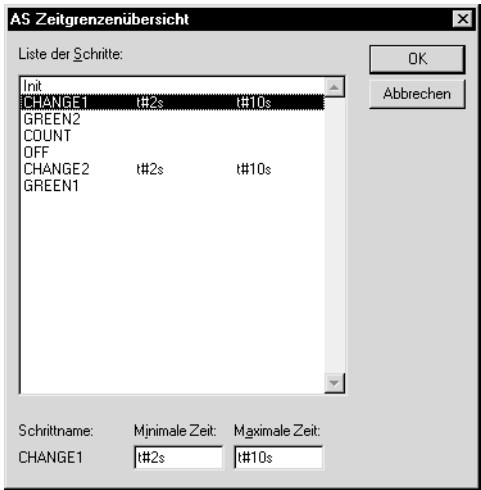
Im folgenden Beispiel ist ein Schritt dargestellt, dessen Ausführung mindestens zwei und höchstens zehn Sekunden dauern soll:



7.8.3.7 Zeitenüberblick

Symbol: -	Menübefehl: Extras→Zeitenüberblick	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl wird das Dialogfeld *Zeitgrenzenübersicht* geöffnet, in dem Sie die Zeiteinstellungen Ihrer AS-Schritte editieren können.



Im Dialogfeld *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).

Zeitbegrenzungen editieren

Sie können außerdem die Zeitbegrenzungen editieren:

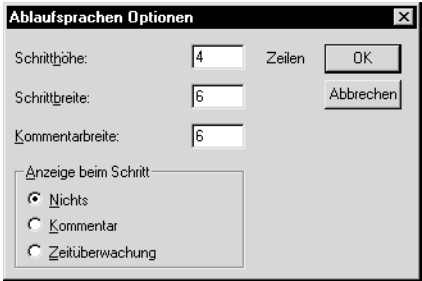
1. Klicken Sie in der Übersicht auf den gewünschten Schritt. Der Schrittname wird unten im Fenster angezeigt.
2. Geben Sie in die Eingabefelder **Minimale Zeit** / **Maximale Zeit** die gewünschte Zeitbegrenzung (Konstante oder Variable vom Typ **TIME**) ein, z. B. T#3s.
3. Klicken Sie auf **OK**, um das Dialogfeld zu schließen und die Veränderungen zu speichern.



7.8.3.8 Optionen

Symbol: -	Menübefehl: Extras→Optionen	Tastaturbefehl: -
-----------	-----------------------------	-------------------

Dieser Befehl öffnet ein Dialogfeld, in dem Sie verschiedene Optionen zu Ihrem AS-Baustein einstellen können.



Schritthöhe

Im Textfeld **Schritthöhe** können Sie eingeben, wie viele Zeilen ein AS-Schritt in Ihrem AS-Editor hoch sein soll (Standardeinstellung: 4).

Schrittbreite

Im Textfeld **Schrittbreite** können Sie eingeben, wie viele Spalten ein Schritt breit sein soll (Standardeinstellung: 6).

Anzeige beim Schritt

Durch Auswahl eines Optionsfeldes können Sie einstellen, welche Anzeige beim Schritt gemacht werden soll.

- **Nichts:** Keine Anzeige.
- **Kommentar:** Der Kommentar zum Schritt wird angezeigt.
- **Zeitüberwachung:** Die Zeitüberwachung wird angezeigt.

Kommentar bzw. Zeitüberwachung werden so angezeigt, wie Sie es unter **Extras→Schritt Attribute** festgelegt haben.

7.8.3.9 Aktion assoziieren

Symbol: -	Menübefehl: Extras→Aktion assoziieren	Tastaturbefehl: -
-----------	---------------------------------------	-------------------

Dieser Befehl assoziiert Aktionen und boolsche Variablen zu IEC-Schritten.

- Rechts neben dem 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 **<F2>** benutzen.
- Neue Aktionen für IEC-Schritte werden im *Object Organizer* zu einem AS-Baustein mit dem Befehl **Projekt→Aktion hinzufügen** angelegt.



Drive PLC Developer Studio

Editoren

7.8.3.10 IEC-Schritte benutzen

Symbol: 	Menübefehl: Extras→IEC-Schritte benutzen	Tastaturbefehl: -
---	---	-------------------

Mit diesem Befehl werden beim Einfügen von Schritt- Transitionen und Parallelzweigen statt den vereinfachten Schritten die IEC-Schritte eingefügt.

- Bei aktiviertem Befehl erscheint ein Haken vor dem Menüpunkt und das Symbol in der Funktionsleiste wird gedrückt dargestellt.
- Beim Anlegen eines AS-Bausteins wird der Init-Schritt als IEC-Schritt angelegt.
- Durch nochmaliges Ausführen wird der Befehl wieder deaktiviert.
- Diese Einstellung wird in der ini-Datei gespeichert und beim nächsten Start des DDS wieder hergestellt.

7.8.4 Befehle im Menü “Projekt”

7.8.4.1 Aktion hinzufügen

Symbol: -	Menübefehl: Projekt→Aktion hinzufügen	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl erzeugt zum selektierten AS-Baustein im *Object Organizer* eine Aktion, die bei den IEC-Schritten dieses Bausteins verwendet werden kann.

- Nach Auswahl des Befehls erscheint ein Dialogfeld, in dem Sie den Aktionsnamen und die Sprache auswählen können, in der die Aktion implementiert werden soll.

Die neue Aktion wird im *Object Organizer* unter ihren AS-Baustein gehängt. Vor dem AS-Baustein erscheint nun ein Pluszeichen.

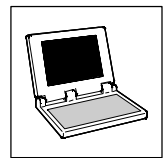
- Durch Klicken auf das Pluszeichen erscheinen die Aktionsobjekte und vor dem Baustein wird nun ein Minuszeichen angezeigt.
- Durch Klicken auf das Minuszeichen werden die Aktionen wieder ausgeblendet und es erscheint wieder das Pluszeichen.



Tip!

Das Ein-/Ausblenden können Sie auch mit den Kontextmenübefehlen **Knoten Expandieren** und **Knoten Kollabieren** erreichen.

Durch Doppelklicken auf die Aktion oder durch Betätigen der **<Eingabetaste>** können Sie die Aktion in den Editor laden.



7.8.5 AS-Flags

Wenn in der Ablaufsprache ein Schritt länger aktiv ist als in seinen Attributen angegeben, dann werden sogenannte AS-Flags gesetzt.

Darüber hinaus gibt es weitere Variablen, die Sie definieren können, um den Ablauf in der Ablaufsprache zu steuern.

Um AS-Flags zu benutzen, müssen diese deklariert werden. Dabei kann die Deklaration global oder lokal, als Aus- oder Eingangsvariable erfolgen.

SFCEnableLimit

- Variable ist vom Typ **BOOL**.
- Wenn diese Variable **TRUE** ist, werden Zeitüberschreitungen bei den Schritten in **SFCError** registriert. Ansonsten werden Zeitüberschreitungen ignoriert.

SFCInit

- Variable vom Typ **BOOL**.
- Wenn diese Variable **TRUE** ist, wird die Ablaufsprache auf den Init-Schritt zurückgesetzt und die anderen AS-Flags zurückgesetzt. Der Init-Schritt bleibt so lange aktiv, wird aber nicht ausgeführt, so lange die Variable **TRUE** ist. Erst wenn **SFCInit** wieder auf **FALSE** gesetzt wird, wird der Baustein wieder normal abgearbeitet.

SFCReset

Diese Variable vom Typ **BOOL** verhält sich ähnlich wie die **SFCInit**. Hier wird allerdings nach der Initialisierung der Init-Schritt weiter abgearbeitet. Das kann dazu benutzt werden, um im Init-Schritt das **SFCReset**-Flag sofort wieder auf **FALSE** zu setzen.



Tip!

Die gleichzeitige Verwendung von **SFCInit** und **SFCReset** in einem Baustein ist nicht möglich. **SFCInit** hat dann immer Vorrang.

Beispiel-Deklaration

```
PROGRAM Flags
VAR
    SFCEnableLimit:BOOL;
    SFCError:BOOL;
    SFCErrorStep:STRING;
    SFCReset AT %IX1.0.2: BOOL;
    SFCInit AT %IX1.0.3:BOOL;
END_VAR
```

SFCQuitError

Variable vom Typ **BOOL**.

Solange diese 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, dann werden alle bisherigen Zeiten in den aktiven Schritten zurückgesetzt.

SFCPause

- Variable vom Typ **BOOL**.
- Solange diese Variable **TRUE** ist, wird die Abarbeitung des AS-Diagramms angehalten.



Drive PLC Developer Studio

Editoren

SFCTrans

- Variable vom Typ **BOOL**.
- Diese Variable wird auf **TRUE** gesetzt, wenn eine Transition **TRUE** wird.

SFCErrror

- Variable vom Typ **BOOL**.
- Diese Variable wird gesetzt, wenn in einem SFC-Diagramm eine Zeitüberschreitung aufgetreten ist.

SFCErrrorStep

- Variable vom Typ **String**.
- Im Falle einer Zeitüberschreitung wird in dieser Variable der Name des Schrittes gespeichert, der die Zeitüberschreitung verursacht hat.

SFCErrrorPOU

- Variable vom Typ **String**.
- Im Falle einer Zeitüberschreitung wird in dieser Variable der Name des Bausteins gespeichert, in dem die Zeitüberschreitung aufgetreten ist.

SFCCurrentStep

- Variable vom Typ **String**.
- In dieser Variablen wird der Name des Schrittes gespeichert, der aktiv ist, unabhängig von der Zeitüberwachung.
- Bei einer Parallelverzweigung wird der Schritt im äußersten rechten Zweig gespeichert.

SFCTip, SFCTipMode

Diese Variablen vom Typ **BOOL** erlauben den Tip-Bereich 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 **TRUE** gesetzt ist, kann zusätzlich auch über die Transitionen weitergeschaltet werden.

SFCErrrorAnalyzation

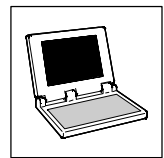
Diese Variable ist vom Typ **STRING**. Wird durch das AS-Flag **SFCErrror** eine Zeitüberschreitung registriert, gibt **SFCErrrorAnalyzation** die verantwortlichen Variablen bzw. Transitionausdrücke aus.

Für diese Funktion muß die Bibliothek Analyzation.lib in das DDS-Projekt eingebunden sein.



Tip!

Wenn einmal eine Zeitüberschreitung aufgetreten ist und die Variable **SFCErrror** nicht wieder zurückgesetzt wurde, wird keine weitere Zeitüberschreitung registriert.



7.8.6 Ablaufsprache im Online-Modus

Monitoring

- Beim Ablaufspracheneditor werden im Onlinebetrieb die aktuell aktiven Schritte als blaue Schritte angezeigt.
- 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.
- Wenn Sie es unter **Extras→Optionen** das Optionsfeld **Zeitüberwachung** gewählt haben, wird neben den Schritten die Zeitüberwachung dargestellt.
- Unter den von Ihnen eingegebenen Unter- und Obergrenzen erscheint eine dritte Zeitangabe, die anzeigt, wie lange der Schritt bereits aktiv ist:



Im obigen 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.



Tip!

Beim sichtbar machen (Monitoring) von Ausdrücken (z. B. A AND B) in Transitionen wird nur der Gesamtwert der Transition dargestellt.

Breakpoints

Mit **Online→Breakpoint an/aus** kann ein Breakpoint auf einen Schritt gesetzt werden. Der Breakpoint kann in einer Aktion an den für die verwendete Sprache zugelassenen Stellen gesetzt werden.

- Die Bearbeitung hält dann vor Ausführung dieses Schrittes bzw. Programmstelle der Aktion an.
- Schritte bzw. Programmstellen auf die ein Breakpoint gesetzt ist, sind Hellblau dargestellt.

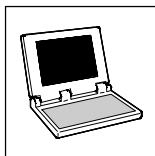
Einzelschrittabarbeitung

Auch in AS wird ein schrittweises Steppen unterstützt (**Online→Einzelschritt über**). Dabei wird stets zum nächsten Schritt gesteppt, dessen Aktion ausgeführt wird.

Ist die aktuelle Position:

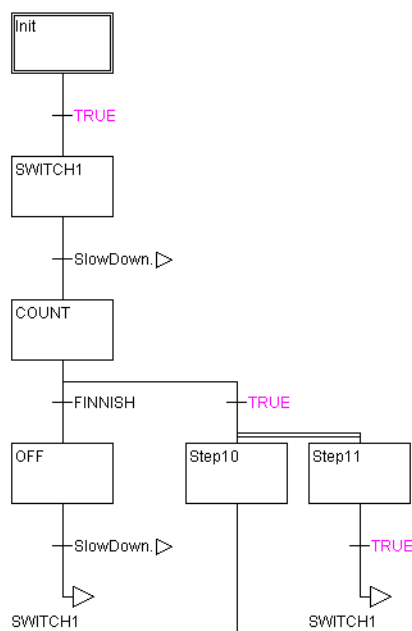
- Ein Schritt in einem linearen Ablauf eines Bausteins oder ein Schritt im ganz rechten Parallelzweig eines Bausteins, dann wird der AS-Baustein verlassen und zum Aufrufer zurück gekehrt. Ist der Baustein das Hauptprogramm, beginnt der nächste Zyklus.
- Ein Schritt in einer Parallelverzweigung nicht ganz rechts, wird zum aktiven Schritt im nächsten Parallelzweig gesprungen.
- Die letzte Breakpoint-Position innerhalb einer Aktion, wird zum Aufrufer des SFC gesprungen.
- Die letzte Breakpoint-Position innerhalb einer IEC-Aktion, wird zum Aufrufer des SFC gesprungen.
- Die letzte Breakpoint-Position innerhalb einer Eingangsaktion oder Ausgangsaktion, wird zum ersten Aktiven Schritt gesprungen.

Mit **Online→Einzelschritt in** können Sie zusätzlich in Aktionen hineinsteppen. Soll in eine Eingangs-, Ausgangs-, oder IEC-Aktion gesprungen werden, muß dort ein Breakpoint vorhanden sein. Innerhalb der Transitionen oder Aktionen stehen Ihnen alle Debugging-Funktionalitäten des entsprechenden Editors zur Verfügung.

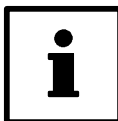


Drive PLC Developer Studio

Editoren



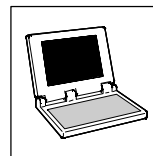
Abarbeitungsreihenfolge der Elemente einer Schrittkette



Tip!

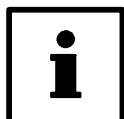
Die Schritte einer Schrittkette werden von oben nach unten und von links nach rechts durchlaufen.

1. Zunächst werden alle Action Control Block Flags der IEC-Aktionen zurückgesetzt, die in dieser Schrittkette verwendet werden. Die Flags von IEC-Aktionen, die innerhalb von Aktionen aufgerufen werden, bleiben gesetzt.
2. Alle Schritte in der Schrittkette werden in der Reihenfolge, in der sie die Schrittkette einnehmen, geprüft, ob die Bedingung zum Ausführen der Ausgangsaktion gegeben ist und gegebenenfalls ausgeführt.
3. Für alle Schritte wird folgendes durchgeführt:
 - Die abgelaufene Zeit wird in die entsprechende Schrittvariable kopiert.
 - Eine Zeitüberschreitung wird überprüft und AS-Error-Flags werden entsprechend gesetzt.
 - Ist der aktuelle Schritt kein IEC-Schritt, wird nur die dazugehörige Aktion ausgeführt.
4. Die IEC-Aktionen der Schrittkette, werden in alphabetischer Reihenfolge ausgeführt. Während des Vorgangs 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.
5. Die Transitionen werden ausgewertet und der nachfolgende Schritt wird aktiviert, wenn folgende Punkte erfüllt sind.
 - Der Schritt im aktuellen Zyklus war aktiv.
 - Die nachfolgende Transition TRUE liefert.
 - Die minimal aktive Zeit abgelaufen ist.



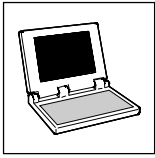
Implementierung von Aktionen

- Wenn eine Aktion in mehreren Schrittketten assoziiert ist, kann sie in einem Zyklus mehrfach ausgeführt werden
- Wird dieselbe IEC-Aktion gleichzeitig in verschiedenen Ebenen eines AS verwendet, sind unerwünschte Effekte durch die oben beschriebene Abarbeitungsreihenfolge möglich.
- In diesem Fall wird eine Fehlermeldung ausgegeben.
- Möglicherweise kann dieses bei der Bearbeitung von Projekten, die mit älteren Versionen erstellt wurden, auftreten.



Tooltip

Wenn Sie den Mauszeiger im Online-Modus eine kurze Zeit über einer Variablen halten, wird der Typ und gegebenenfalls der Kommentar der Variablen in einem Tooltip angezeigt.



Drive PLC Developer Studio

Editoren



8 Ressourcen

In der Registerkarte **Ressourcen** des *Object Organizers* befinden sich Objekte zum Konfigurieren und Organisieren Ihres Projektes und zur Verfolgung von Variablenwerten:



-  Mitgelieferte Bibliotheken, die über die Bibliotheksverwaltung in das Projekt eingebunden wurden. (📖 8-44)
-  Globale Variablen des Projektes und der eingebundenen Bibliotheken. (📖 8-2)
-  Codestellen-Initialisierungswerte zum Initialisieren von Codestellen mit einem Anfangswert. (📖 8-5)
-  Parameter-Monitor zur Parametrierung und Überwachung von Codestellenwerten. (📖 8-7)
-  Instanz-Parametermanager (📖 8-13) und Typ-Parametermanager (📖 8-20) für den Zugriff auf Variablen über Codestellen.
-  Prozeßabbild zur Anzeige von zu Systembausteinen zugehörigen Prozeßabbildern. (📖 8-23)
-  Steuerungskonfiguration zum Konfigurieren Ihrer Hardware. (📖 8-26)
-  Taskmonitor zur Überwachung der Tasklaufzeiten. (📖 8-31)
-  Taskkonfiguration zur Steuerung Ihres Programms über Tasks. (📖 8-32)
-  Watch- und Rezepturverwalter zum Anzeigen und Vorbelegen von Variablenwerten. (📖 8-38)
-  Zielsystemeinstellungen zur Auswahl und Parametrierung der Zielhardware. (📖 8-41)



Hinweis!

Es ist abhängig vom Zielsystem, welche von den abgebildeten Objekten dargestellt werden.



Drive PLC Developer Studio

Ressourcen

8.1 Globale Variablen



In der Registerkarte **Ressourcen** des *Object Organizers* befindet sich im Ordner **Globale Variablen** die globale Variablenliste (Globale_Variablen).

- Alle in diesem Objekt definierten Variablen sind im ganzen Projekt bekannt.
- Per Doppelklick auf das Objekt **Globale_Variablen** öffnen Sie die Variablenliste zum Bearbeiten.

Globale Variablen

Normale globale Variablen werden in einem Objekt zwischen den Schlüsselwörtern **VAR_GLOBAL** und **END_VAR** definiert:

```
VAR_GLOBAL
    (* Variablendeklarationen *)
END_VAR
```

Remanente globale Variablen

Remanente globale Variablen erhalten zusätzlich das Schlüsselwort **RETAIN**:

```
VAR_GLOBAL RETAIN
    (* Variablendeklarationen *)
END_VAR
```

Globale Konstanten

Globale Konstanten erhalten zusätzlich das Schlüsselwort **CONSTANT**:

```
VAR_GLOBAL CONSTANT
    (* Variablendeklarationen *)
END_VAR
```

Remanente globale Konstanten

Remanente globale Konstanten erhalten zusätzlich das Schlüsselwort **CONSTANT RETAIN**:

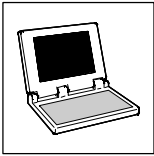
```
VAR_GLOBAL CONSTANT RETAIN
    (* Variablendeklarationen *)
END_VAR
```

8.1.1 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.

1. Selektieren Sie im *Object Organizer* den Ordner *Globale Variablen* oder eines der bestehenden Objekte mit globalen Variablen.
2. Führen Sie den Menübefehl **Projekt→Objekt einfügen** aus.
3. Geben Sie dem Objekt in der erscheinenden Dialogbox einen entsprechenden Namen.

Damit wird ein weiteres Objekt mit dem Schlüsselwort **VAR_GLOBAL** angelegt.



8.1.1.1 Alle Instanzpfade

Symbol: -	Menübefehl: Einfügen→Alle Instanzpfade	Tastaturbefehl: -
-----------	--	-------------------

Ist der Dialog Variablen-Konfiguration geöffnet, wird beim Ausführen dieses Befehls ein

```
VAR_CONFIG
```

```
END_VAR
```

Block erzeugt. Dieser enthält alle im Projekt vorhandenen Instanzpfade.
Um bestehende Adressen zu erhalten, werden vorhandene Deklarationen nicht neu eingefügt.
Ist das Projekt kompiliert, steht der Menüpunkt im Fenster der Variablenkonfiguration zu Verfügung.

8.1.2 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 Menübefehl **Extras→Doku-Vorlage erstellen**.

Dokumentvorlage editieren

Die Doku-Vorlage ist eine ASCII-Datei, die Sie in einen beliebigen Texteditor laden und editieren können.

Die Doku-Vorlage beginnt mit der Zeile **DOKUFILE**, dann folgt eine Auflistung der Projektvariablen. Für jede Projektvariable werden drei Zeilen vergeben:

- Die 1. Zeile enthält das Schlüsselwort **VAR**, um anzuzeigen, daß eine Variable beginnt.
- Die 2. Zeile enthält den Namen der Variablen.
- Die 3. Zeile ist eine leere Zeile, in die Sie einen Kommentar eingeben können.

Variablen, die Sie nicht dokumentieren wollen, löschen Sie einfach aus dem Text. Sie können beliebig viele Dokumentvorlagen zu Ihrem Projekt erstellen.

Dokumentvorlage verwenden

Um eine Dokumentvorlage zu benutzen, verwenden Sie den Menübefehl **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!

8.1.2.1 Doku-Vorlage erstellen

Symbol: -	Menübefehl: Extras→Doku-Vorlage erstellen	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl erstellt eine Dokumentvorlage.

- Der Befehl steht zur Verfügung, wenn ein Objekt **Globalen Variablen** im *Object Organizer* im Register **Ressourcen** selektiert ist.

Nachdem der Befehl gewählt wurde, erscheint das Dialogfeld *Datei speichern unter*.

- Geben Sie im Textfeld **Dateiname** einen neuen Namen für die Datei ein.
- Im Textfeld **Dateityp** ist der Zusatz **"*.txt"** bereits vorgegeben.

Es wird nun eine Textdatei erstellt, in der sämtliche Variablen Ihres Projekts aufgelistet sind.



Drive PLC Developer Studio

Ressourcen

8.1.2.2 Doku-Vorlage auswählen

Symbol: -	Menübefehl: Extras→Doku-Vorlage auswählen	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl öffnet das Dialogfeld zum Auswählen der Dokumentvorlage.

- Wählen Sie die gewünschte Dokumentvorlage aus und klicken Sie auf **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!

8.2 Codestellen-Initialisierungswerte



Die Codestellen-Initialisierungswerte befinden sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.

Priorität der Initialisierungswerte

Reset-Arten	Priorität in absteigender Reihenfolge
Reset	Codestellen-Initialisierungswert Parametermanager
Reset (Kalt)	Codestellen-Initialisierungswert Parametermanager
Reset (Ursprung)	Löscht das gesamte Programm im Zielsystem. Dadurch ist kein Wert mehr gültig.
Programm-Download	Codestellen-Initialisierungswert Parametermanager
Netzschalten	Es gilt der zuletzt mit C0003 gespeicherte Wert.
Werksabgleich laden	Parametermanager

Sie können dieses Objekt zum Initialisieren von Codestellen mit einem Anfangswert verwenden. Neben den im Parametermanager angelegten Codestellen können auch alle Systemcodestellen initialisiert werden. Diese Codestellen werden netzausfallsicher am Ende des Downloads im Parametersatz des Zielsystems gespeichert.

- Alle gewählten Codestellen werden im Parametersatz überschrieben.
- Es kann jede beliebige Codestelle beschrieben werden.
- Überschriebene im Parametermanager eingestellte Initialwerte sind nur noch im Werksabgleich gespeichert. (siehe Codestelle C0002 und C0003)
- Die eingetragenen Codestellen werden der Reihe nach im Parametersatz gespeichert. Ausnahme: Die Codestellen C0086 und C0025 werden zu Beginn der Initialisierung verändert, weil sie auf andere Codestellen wirken.
- Unbekannte Codestellen und Initialisierungswerte außerhalb der Codestellenwertegrenzen werden vom Zielsystem ignoriert.

Durch Doppelklick auf das Objekt **Codestellen-Initialisierungswerte** im *Object Organizer* öffnen Sie das Dialogfeld *Codestellen-Initialisierungswerte*.

Codestellen Initialisierungswerte

Codestelle	Subc...	Initialwert	Bemerkung
C0114	001	0.0000	Klemmer E1=High aktive
C0114	002	0.0000	Klemmer E2=High aktive
C0114	003	0.0000	Klemmer E3=High aktive
C0114	004	0.0000	Klemmer E4=High aktive
C0114	005	0.0000	Klemmer E5=High aktive
C0118	001	0.0000	Ausgang A1= High aktive
C0118	002	0.0000	Ausgang A2= High aktive
C0118	003	0.0000	Ausgang A3= High aktive
C0118	004	0.0000	Ausgang A4= High aktive
C0911	000	1.0000	Touch-Probe über E4
C2104	000	1.0000	Autostart nach Netzein

OK

Neu



Drive PLC Developer Studio

Ressourcen

Codestelle mit Initialwert hinzufügen:

1. Wählen Sie im Dialogfeld *Codestellen-Initialisierungswerte* die Schaltfläche **Neu**, um eine Codestelle mit Initialwert hinzuzufügen.
2. Geben Sie im Dialogfeld *Initialisierungswert* im Eingabefeld **Codestelle** die Nummer der Codestelle (ohne vorangestelltes "C") ein, für die Sie einen Initialwert festlegen möchten.
3. Geben Sie im Eingabefeld **Initialisierungswert** den Wert ein, mit dem die Codestelle initialisiert werden soll.
4. Falls die Codestelle Subcodestellen beinhaltet, geben Sie im Eingabefeld **Subcode** den entsprechenden Subcode der Codestelle ein.
5. Optional können Sie im Eingabefeld **Bemerkung** eine Bemerkung eingeben, die dann im Dialogfeld *Codestellen-Initialisierungswerte* mit angezeigt wird.
6. Bestätigen Sie mit **OK**, um die Eingaben zu übernehmen, bzw. wählen Sie **Abbrechen**, um den Vorgang abzubrechen und das Dialogfeld zu schließen.

Initialwert einer Codestelle ändern:

1. Klicken Sie mit dem Mauszeigersymbol  im Listenfeld auf die Zeile mit der Codestelle, deren Einstellungen Sie ändern möchten.
2. Nehmen Sie im Dialogfeld *Initialisierungswert* die gewünschten Änderungen vor und bestätigen Sie mit **OK**.

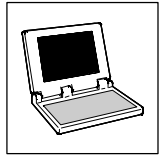
Codestelle mit Initialwert löschen:

1. Klicken Sie mit dem Mauszeigersymbol  im Listenfeld auf die Zeile mit der Codestelle, die Sie löschen möchten.
2. Wählen Sie im Dialogfeld *Initialisierungswert* die Schaltfläche **Löschen**, um die ausgewählte Codestelle zu löschen.

Beispiel:

Für ein Projekt mit der **9300 Servo PLC** soll immer derselbe Motortyp verwendet werden. Um den Motortyp nicht jedesmal konfigurieren zu müssen, kann dieser mit Hilfe der Codesstellen-Initialisierungswerte fest im Projekt gespeichert werden:

- 0086 = Codestelle zur Auswahl des Motortyps
- Initialisierungswert 108 = Motortyp MDSK5XX036-13, f_N : 200 Hz



8.3 Parameter-Monitor



Der Parameter-Monitor befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.

Mit Hilfe des Parameter-Monitors können Sie Codestellen *online* auf der PLC parametrieren. Zu den parametrierbaren Codestellen gehören die über den Parametermanager neu angelegten Codestellen ebenso wie die Codestellen der ausgewählten PLC (z. B. 9300 Servo PLC, Drive PLC).



Tip!

Geänderte Codestellenwerte werden nach dem Ausloggen nicht im Projekt gespeichert!

Wenn Sie die Änderungen von Codestellenwerten speichern möchten, verwenden Sie eine der folgenden Möglichkeiten.

Doppelklicken Sie im *Object Organizer* auf das Objekt **Parameter-Monitor**, um den Parameter-Monitor zu öffnen:

- Speichern Sie die Änderungen im Parametersatz der PLC (C0003=1).
- Speichern Sie die Änderungen im Projekt durch das Eintragen der neuen Werte in das Objekt **Codestellen-Initialisierungswerte** im *Object Organizer*. (□ 8-5)
- Speichern Sie die Änderungen mit Hilfe des Softwaretools Global Drive Control (GDC).

Code	Text	Wert	Einheit
C0025	Rückführung	RSx (Resolver)	
C0026/1	FCODE(OffsetAIN)	0.00	%
C0026/2	FCODE(OffsetAIN)	0.00	%
C0027/1	FCODE(Verst.AIN)	100.00	%
C0027/2	FCODE(Verst.AIN)	0.00	%
C0030	DFOUT Konstante	2048 Inkr./Umdr.	
C0032	FCODE(Getriebefaktor Zähler)	1	
C0034	Leitspannung/Leitstrom	-10V...+10V	
C0037	Sollwertvorgabe (rpm)	0	rpm
C0040	Reglerfreigabe RFR	Regler frei	
C0042	DIS: Quickstop	QSP: nicht aktiv	
C0043	Fehler rücksetzen	TRIP-Reset	
C0050	DIS: Drehzahl Sollwert	0.00	%
C0051	DIS: Drehzahl Istwert	0	rpm
C0052	DIS: Motorspannung	0	V
C0053	DIS: Zwischenkreisspannung	0	V
C0054	DIS: Imot (Motorstrom)	0.0	A
C0055	DIS: Motorstrom	0	%

Der Parameter-Monitor enthält auf der linken Seite einen Menübaum mit Ordnern, denen Codestellen oder weitere Unterordner zugeordnet sind.

- Ordner mit vorangestelltem Pluszeichen beinhalten weitere Unterordner.
 - Mit einem Klick auf das Pluszeichen wird der Ordner aufgeklappt und die untergeordneten Ordner erscheinen. Anstelle des Pluszeichens wird nun ein Minuszeichen angezeigt.
 - Mit einem Klick auf das Minuszeichen können die untergeordneten Ordner wieder ausgeblendet werden.
- Ordner ohne vorangestelltes Plus- oder Minuszeichen enthalten keine weiteren Unterordner sondern Codestellen.
 - Wenn Sie einen solchen Ordner auswählen, werden die dem Ordner zugeordneten Codestellen in der Liste auf der rechten Seite angezeigt.
 - Codestellen, die nur lesbar sind, werden in der Liste in grauer Schrift dargestellt.
 - Codestellen, deren Wert noch nicht aus der PLC gelesen wurde, werden in grüner Schrift dargestellt.



Drive PLC Developer Studio

Ressourcen

8.3.1 System-Codestellen/User-Codestellen

Bei den Codestellen wird zwischen System-Codestellen und User-Codestellen unterschieden:

System-Codestellen

System-Codestellen sind fest in der PLC integrierte, werkseitig vorgelegte Codestellen, über die die PLC mit Hilfe von Parametriertools wie GDC oder Keypad parametrieren werden kann.

- Mit Hilfe dieser Codestellen können Sie z. B. bei der 9300 Servo PLC Einstellungen für Motortyp, Konstante des Drehzahlreglers, Rückführsystem, Feldbus, Klemmenpolarität usw. vornehmen.
- Die Parametrierung dieser Codestellen können Sie mit dem Parameter-Monitor *online* auf der PLC durchführen und mit C0003=1 im Parametersatz der PLC abspeichern.

User-Codestellen

User-Codestellen sind mit Hilfe des Instanz-Parametermanagers angelegte Codestellen, die in fester Verbindung zu einer Programmvariablen stehen.

- Mit Hilfe dieser Codestellen können Sie über Feldbusprofile, Parametrierprogramme sowie Bedien- und Anzeigetools auf Variablen im SPS-Programm zugreifen.

Eingruppierung der Codestellen

- Im Ordner **Codeliste** sind alle Codestellen enthalten, die System-Codestellen der ausgewählten PLC sowie die im Projekt angelegten User-Codestellen.
- Im Ordner **Eigene IEC 61131-Codestellen** sind nur die im Projekt angelegten User-Codestellen enthalten.
- Alle anderen Ordner enthalten wiederum System-Codestellen der ausgewählten PLC zur Parametrierung einer bestimmten Funktionalität. So sind im Ordner **Multitasking** alle Codestellen enthalten, über die das Multitasking-Verhalten der PLC parametrieren werden kann.

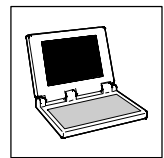
8.3.2 Codestellen parametrieren



Tip!

Damit Sie über den Parameter-Monitor Codestellen parametrieren können, muß das DDS *online* mit der PLC verbunden sein.

- Klicken Sie mit dem Mauszeiger auf die Codestelle, deren Einstellungen Sie ändern möchten. Je nach Art der Codestelle wird ein Dialogfeld geöffnet, in dem Sie die gewünschten Änderungen vornehmen können.



8.3.3 Unterscheidung zwischen Online- und Offline-Modus

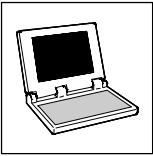
Im **Offline-Modus**

- werden für Schreibcodestellen die Codestellenwerte angezeigt, die zuletzt vom DDS geschrieben wurden.
- werden für Lesecodestellen die voreingestellten Werte (in der Regel 0) angezeigt.

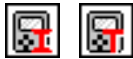
Im **Online-Modus**

- werden die im Parameter-Monitor sichtbaren Codestellenwerte permanent aus der PLC gelesen und angezeigt. Codestellen, deren Wert noch nicht aus der PLC gelesen wurde, werden in grüner Schrift dargestellt.
- werden durchgeführte Wertänderungen an Codestellen mit Schreibzugriff während Netz-Aus nicht gespeichert. Erst über die Funktion "Parametersatz speichern" (C0003=1) werden die veränderten Werte im Parametersatz der PLC gespeichert und stehen auch nach einem Netz-Aus wieder zur Verfügung.

Mit geöffneten Parameter-Monitor ist das Monitoring von Variablen nur eingeschränkt möglich. Variablen werden möglicherweise nur langsam oder gar nicht aktualisiert.



8.4 Parametermanager



Die IEC 61131-3 verwendet Variablen, um Anwenderdaten zu initialisieren, weiterzuverarbeiten und zwischenspeichern. Diese Variablen werden im Deklarationsteil jeder POE deklariert, d.h. es wird ihre Zuordnung zu einem bestimmten Datentyp (wie Byte oder Integer) bekanntgegeben. Bei dieser Deklaration können Variableneigenschaften wie Netzausfallsicherheit, festgelegte Anfangswerte oder Zuordnungen zu festen physikalischen Adressen festgelegt werden.

Bei Antriebsreglern ist die Verwendung von Variablen nicht gängig. Das Bedienen von Antriebsregler basiert auf Codestellen. Feldbusprofile, Parametrierprogramme, Bedien- und Anzeigetools nutzen diese Codestellen zur Bearbeitung der Daten, die im Antriebsregler abgelegt sind.

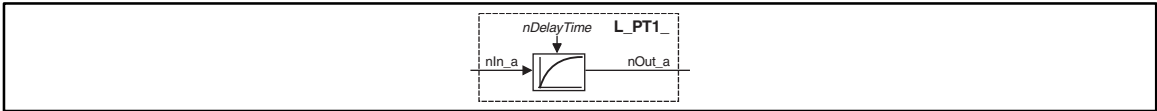
Mit dem Parametermanager stellt Ihnen das Drive PLC Developer Studio ein Werkzeug zur Verfügung, mit dem Sie die Verbindung von Codestelle und Variable herstellen und so über Codestellen auf Variablen zugreifen können. Anwendercodestellen sowie die bekannten Codestellen der Lenze-Funktionsblöcke können über den Parametermanager verwaltet werden.

Notwendigkeit des Parametermanagers

Der Anwender schreibt mit Hilfe des Drive PLC Developer Studio sein IEC 61131-3 Programm. Um im Online-Betrieb das Programm zu parametrieren, zu diagnostizieren oder über Feldbus/Systembus Daten auszutauschen, vergibt der Anwender für bestimmte Variable eine Codestelle.

- Variablen vom Typ `VAR_IN_OUT` sowie Variablen innerhalb eines Funktionsblocks vom Typ `VAR` können keine Codestellen zugewiesen werden.

Beispiel Lenze-Funktionsblock `L_PT1_`:



VariableName	DataType	SignalType	VariableType
nIn_a	Integer	analog	VAR_INPUT
nOut_a	Integer	analog	VAR_OUTPUT
nDelayTime	-	-	VAR_CONSTANT_RETAIN

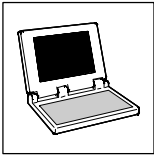


Tip!

Nur die im Parametermanager vergebenen Codestellen können über GDC, Keypad oder dem Parameter-Monitor im Drive PLC Developer Studio angezeigt und parametrieren werden.

Auf die restliche Programmstruktur (Variablen ohne Verbindung zu Codestellen) kann nicht über die bekannten Feldbusprofile, Parametrierprogramme und Bedien- und Anzeigetools zugegriffen werden.

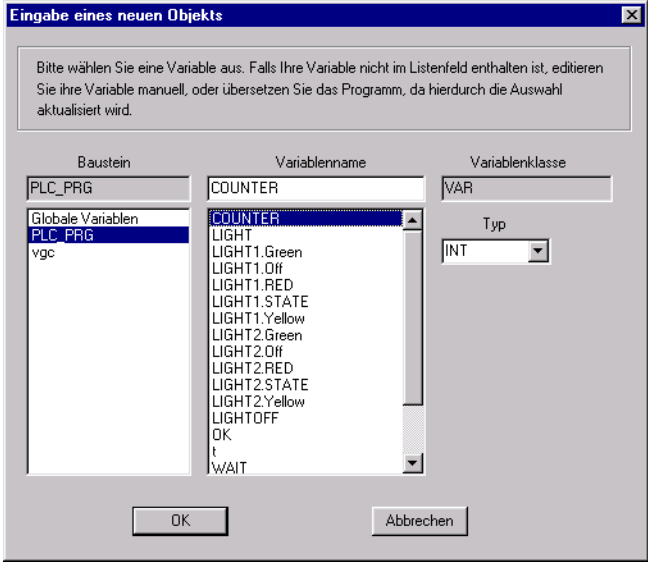
Vergibt der Anwender z. B. der Variablen `nDelayTime` die Codestelle `C6000`, so kann der Anwender über diese Codestelle die Variable `nDelayTime` parametrieren oder diagnostizieren. Nur die vom Anwender zugewiesene Codestelle kann über die Parametrierumgebung diagnostiziert/parametriert werden.

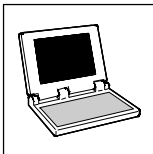


8.4.1 Neues Objekt

Symbol: -	Menübefehl: Einfügen→Neues Objekt hinzufügen	Tastaturbefehl: -
-----------	--	-------------------

Beim Anlegen eines neuen Objektes wird neben dem Namen des Bausteins, der Variablen und des Variablentyps auch die Variablenklasse angezeigt.
Das Dialogfeld ist beim Instanz-Parametermanager und Typ-Parametermanager gleich.





Drive PLC Developer Studio

Ressourcen

8.4.2 Begriffe des Parametermanagers

Begriff	Beschreibung
Usercodestellen	Der Anwender kann für Variablen, die in seinem IEC 61131-3 Programm verwendet werden, anwenderspezifische Codestellen (User-Codestellen) vergeben. Die User-Codestellen sind zielsystemabhängig und liegen in dem Bereich C3000 bis C7999.
Codestelleneigenschaften	Codestelleneigenschaften (z. B. nur Lesen, Lesen/Schreiben, direkte Übernahme oder direkte Übernahme bei Bestätigung) können für jede vergebene Codestelle festgelegt werden.
Codstellenwert	Der Codstellenwert ist der Wert, der über die Codestelle (mit Hilfe von GDC, Keypad oder dem Parameter-Monitor) angezeigt oder eingegeben wird. Dieser Wert entspricht dem physikalischen Wert, der über eine Skalierung als Variablenwert im Anwenderprogramm bekannt ist.
Displaytext	Für die Parametrierung/Diagnose mit dem Keypad werden die anwenderspezifischen Texte angezeigt. Der freie Text kann bis zu 13 Zeichen enthalten und wird zur Codestelle abgezeigt.
Eingabefenster mit gelber Hintergrundfarbe	Verändert der Anwender eine Standardeinstellung, dann ändert sich die Hintergrundfarbe des Feldes auf gelb. Änderungen im Typ-Parametermanager werden für dieses Feld dann nicht weiter berücksichtigt.
Einheiten	Zahlenwerten können auch physikalische oder definierte Einheiten zugeordnet werden, z. B. rpm, Nm, Flaschen/min usw.
Erstinstanzen	Bei Verwendung vom Lenze-Funktionsblöcken können den Instanzen dieser FB's (z. B. des 9300 Servo V2.0) komplett die Parameter-Codestellen, die bei Lenze für die FB's üblich sind, mit der Funktion Erstinstanz zugeordnet werden. Eine Nachbearbeitung der Parameter-Codestellen ist dann nicht mehr möglich. Die Zuordnung der Parameter-Codestellen kann auch wieder komplett aufgehoben werden. Lenze Funktionsblöcke können mit beliebigen Codestellen verbunden werden. Für Standardaufgaben stellt Lenze FB's, die über Codestellen parametrierbar sind, zur Verfügung. Bei den PLC-Geräten werden die FB's durch Einbinden einer Bibliothek (z.B. LenzeDrive.lib) genutzt. Sollen dem FB (z.B. L_NSET) die standardmäßigen Codestellen zum Parametrieren zugeordnet werden, dann wird der FB als L_NSET1 vom Typ L_NSET instanziiert. Erstinstanzierung Für diesen FB können im Parametermanager die Codestellen inkl. Skalierung angelegt werden.
Parametrierprogramm - Sprachen	Für die Parametrierung/Diagnose wird in GDC bzw. im Parameter-Monitor zu jeder Codestelle ein Text mit max. 35 Zeichen angezeigt, dieser Text kann im Parametermanager für mehrere Sprachen (Standardsprache/Sprache1/Sprache2) definiert werden. Die Sprache kann bei der Erstellung der GDC-Beschreibungsdatei ausgewählt werden, somit kann der Anwender entscheiden, in welcher Sprache die Anzeigetexte dargestellt werden.
Initialisierung	Der Codestelle kann ein Initialwert zugeordnet werden, die bei der Werkseinstellung für die Codestelle als Initialwert übernommen wird (C0002).
Instanz-Parametermanager	Vergabe von Codestellen für verwendete Variablen der erstellten Programm-POE oder für globale Variablen eines Projektes.
Skalierfaktor/Skalieroutine	Interne Variablen können über Skalierfaktoren transparenter dargestellt werden. Im IEC 61131-3 Programm wird z. B. mit einem 16 Bit-Wert (z. B. Integer) gerechnet. Über die Skalieroutine kann aber im GDC ein Wert zwischen $\pm 200\%$ eingestellt werden. Der 16 Bit Wert wird einfach von ± 32767 auf ± 200 skaliert. Die Grenzen des Zahlenbereiches können in der definierten Codestelle für den Anwender festgelegt werden, z. B. daß der Integerwert der Codestelle nur zwischen $\pm 16384 \rightarrow \pm 100\%$ eingestellt werden kann. Über Skalieroutinen können komplexe Skalierungen vorgenommen werden. Skalieroutinen sind selbst ein kleines IEC 61131-3 Programm, das der Anwender programmieren kann. Das Programm kann interne Umrechnungsskalierungen, Grenzwertberechnungen o. ä. enthalten.
Subcodestellen	Subcodestellen werden automatisch vergeben, wenn die Variable vom Datentyp Array ist. Beispiel: <code>var1:ARRAY[0..4] OF INT</code> (hier werden bis zu 5 Subcodestellen vergeben) Alle Subcodestellen erhalten im Parametermanager den gleichen Initialisierungswert.
Typ-Parametermanager	Definieren einer Standardeinstellung. Vordefinierung der Variablen der erstellten Funktionsbaustein-POE's.
Variablentypen	Variablen vom Typ VAR_IN_OUT sowie Variablen innerhalb eines Funktionsblocks vom Typ VAR können keine Codestellen zugewiesen werden.
Variablenwert	Der Variablenwert ist der Wert, der im IEC 61131-3 Programm abgearbeitet wird (z. B. ein Integer Wert von 16384).



8.4.3 Instanz-Parametermanager



Der Instanz-Parametermanager befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*. Er bietet Ihnen die Möglichkeit, den im Projekt deklarierten Variablen Codestellen zuzuweisen.

- Die Variablen können global oder lokal in POE's vom Typ Programm deklariert sein.
- Nach dem Starten des Instanz-Parametermanagers kann für das gesamte Projekt die Codestellenzuweisung erfolgen.



Tip!

Die verfügbaren Erstinstanzen eines Funktionsblocks sind im entsprechenden Handbuch zur Funktionsbibliothek und in der Online-Hilfe im Kapitel "Lenze Funktionsbibliotheken" beschrieben.

8.4.3.1 Instanz-Parametermanager öffnen

Doppelklicken Sie im *Object Organizer* auf das Objekt **Instanz-Parametermanager**, um den Instanz-Parametermanager zu öffnen:

Instanz-Parametermanager											
Codenr.	Programm-Baustein	Variable	Parameterstext	Var.Type	Initialisierung	Einheit	Codest.Max.	Codest.Min.	int.Max.	int.Min.	ext.Datent...
C0672	MotorControl	L_RFG1.dnTir	L_RFG1 Tir (Ablaufzeit)	DINT	5.000	s	999.999	0.000	999999	0	Festkomma
C0671	MotorControl	L_RFG1.dnTir	L_RFG1 Tir (Hochlaufzeit)	DINT	5.000	s	999.999	0.000	999999	0	Festkomma

- Im Instanz-Parametermanager werden Ihnen Informationen zu den bereits vergebenen Codestellen angezeigt.
- Häufig benötigte Befehle stehen Ihnen im Instanz-Parametermanager auch über das Kontextmenü (rechte Maustaste) zur Verfügung:

Neues Objekt hinzufügen
Objekt bearbeiten...
Objekt löschen...
Objekt kopieren
Objekt einfügen...
Lenze-Erstinstanz-Codestellen vergeben



Tip!

Über die Schaltflächen **Codenr.** und **Programm-Baustein** in der Kopfzeile der Liste können Sie die Sortierung für die Anzeige der Einträge festlegen:

- Klicken Sie auf die Schaltfläche **Codenr.**, um die Einträge nach Codenummern numerisch aufsteigend sortiert anzuzeigen.
- Klicken Sie auf die Schaltfläche **Programm-Baustein**, um die Einträge nach Programm-Bausteinname und Variablenname alphabetisch aufsteigend sortiert anzuzeigen.

8.4.3.2 Codestelle editieren

Die Einstellungen für eine bereits vergebenen Codestelle können Sie editieren, indem Sie

- mit dem Mauszeiger auf die entsprechenden Zeile doppelklicken - oder -
- die entsprechende Zeile markieren und **Einfügen→Objekt bearbeiten...** wählen.

Daraufhin öffnet sich das Dialogfeld *Instanz-Parametermanager*, in dem Sie die Einstellungen für die Codestelle editieren können. (☞ 8-14)



Drive PLC Developer Studio

Ressourcen

8.4.3.3 Codestelle hinzufügen

Wählen Sie **Einfügen→Neues Objekt hinzufügen**, um eine neue Codestelle hinzuzufügen.



Hinweis!

Es ist nicht zulässig auf Systemvariablen Codestellen anzulegen.

Für REAL Variablen können keine Codestellen angelegt werden.

- Im Dialogfeld *Eingabe eines neuen Objekts* wählen Sie die Variable aus, mit der die Codestelle verknüpft werden soll:
- Nach Auswahl der Variable und Übernahme durch **OK** wird das Dialogfeld *Instanz-Parametermanager* geöffnet, in dem Sie die Einstellungen für die Codestelle vornehmen können.

Folgende Datentypen werden vom Instanzparameter-Manager unterstützt.

- BOOL
- BYTE
- SINT
- USINT
- INT
- UINT
- DINT
- UDINT
- WORD
- DWORD



Hinweis!

Beim Kopieren und Einfügen von Objekten wird der Inhalt komplett ersetzt.

8.4.3.4 Dialogfeld Instanz-Parametermanager

Im Dialogfeld *Instanz-Parametermanager* können Sie die Einstellungen zu einer bereits vorhandenen Codestelle editieren bzw. Einstellungen für eine neue Codestelle vornehmen:

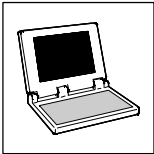


Hinweis!

Systemvariablen dürfen nicht mit Schreib/Lese Codestellen verknüpft werden.

Codestelle/Index

- Es steht der Codestellenbereich C3000 bis C7999 zur freien Verfügung. Die Eingabe der Codestellennummer erfolgt ohne ein vorangestelltes 'C'.
- Wenn Sie die Schaltfläche **Erstinstanz** betätigen, wird in das Eingabefeld Codestelle die bekannte Parametercodestelle eingefügt. Die Eingabefelder **Codest. Max./Codest. Min.** und **int. Max./int. Min.** werden für Eingaben deaktiviert (und in grau dargestellt).
- Wurde eine bereits vergebene Codestelle ausgewählt, wird eine Fehlermeldung nach der Betätigung der Schaltfläche **OK** ausgegeben.



Codest. Max./int. Max. und Codest. Min./int. Min.

Diese vier Eingabefelder dienen zur Definition von Eingabegrenzen und zur Skalierung des Codestellenwertes.

In den Eingabefeldern **Codest. Max./Codest. Min.** werden die Grenzen des Codestellenwertes eingestellt, auf die die Codestelle beim Beschreiben überprüft wird. Werden die Grenzen beim Beschreiben nicht eingehalten, so wird der Codestellenwert vom Laufzeitsystem zurückgewiesen.

Zusammen mit den in den Eingabefeldern **int. Max./int. Min.** eingegebenen Werten wird eine Zuordnung zwischen Codestellenwert und dem normierten, im Programm verwendeten Variablenwert getroffen. Für die Skalierung wird eine Geradengleichung verwendet, deren Steigung und Nullpunkt von den vier eingegebenen Faktoren bestimmt wird.

Die Geradengleichung lautet:

$$m = \frac{\text{int. Max.} - \text{int. Min.}}{\text{Codest. Max.} - \text{Codest. Min.}}$$

$$\text{Variablenwert} = m(\text{Codestellenwert} - \text{Codest. Max.}) + \text{Int. Max.}$$

Beispiel:

- Die Codestellengrenzen sollen zwischen $\pm 100\%$ liegen
- Die Variablengrenzen sollen zwischen ± 16384 (Integer) liegen
- Wie groß ist der Variablenwert bei einem eingestellten Codestellenwert von 50%?

$$m = \frac{16384 - (-16384)}{100\% - (-100\%)}$$

$$\text{Variablenwert} = m(50\% - 100\%) + 16384 = 8192$$

Bei einem Codestellenwert von 50% beträgt der Variablenwert 8192. Diese Rechnung verdeutlicht, daß ein externer Codestellenwert von 50% eingegeben werden kann, intern aber mit dem Variablenwert 8192 gerechnet wird. Es fand somit eine Skalierung von 50% auf 8192 statt.

Skalierroutine

Optional kann zu der POE eine Skalieroutine als Unterprogramm (IEC 61131-3 Programm) eingefügt werden. Mit Hilfe dieses Programmes kann eine komplexe Skalierung einer Variablen, der eine Codestelle zugewiesen wurde, programmiert werden. Siehe Typ-Parametermanager: "Skalierroutinen" (8-20)

Zugriff

Im Gruppenfeld **Zugriff** legen Sie fest, ob auf die Codestelle schreibend und lesend oder nur lesend zugegriffen werden darf.

Initialisierung

Im Eingabefeld **Initialisierung** können Sie den Codestellenwert eingeben, mit dem die Variable durch Laden von "C0002 = Werkseinstellung" initialisiert werden soll. Beim Initialisierungswert (Codestellenwert) handelt es sich um den skalierten Wert, d. h. der Initialwert wird, bevor er als Variablenwert im IEC 61131-3 Programm zur Verfügung steht, über die Skalierfaktoren skaliert.

- Wurde eine Variable im Deklarationseditor wie auch im Parametermanager mit einem Wert initialisiert, so ist der Initialwert des Parametermanagers dominierend.



Tip!

Die Initialisierung kann auch ohne Vergabe einer Codestelle erfolgen.

Sie können eine Variable mit einem skalierten Wert, der beim Start des SPS-Programms übergeben wird, initialisieren. Dies hat den Vorteil, daß z. B. physikalische Werte als Initialwerte verwendet werden.



Drive PLC Developer Studio

Ressourcen

Einheit

Im Listenfeld **Einheit** können Sie der Codestelle eine Einheit aus einer Auswahl physikalischer und fiktiver Einheiten zuweisen. Über die Schaltfläche **Neu** können Sie auch neue Anwendungsspezifische Einheiten (z. B. Flaschen/Minute) definieren.

Ext. Datentyp

Im Auswahlfeld **ext. Datentyp** können Sie der Codestelle einen speziellen Datentyp aus folgender Auswahl vergeben:

Datentyp	Nachkommastellen	Länge	Vorzeichen
Festkomma*	vier feste Nachkommastellen	32 Bit	mit
8-Bit mit Vorz.	keine	8 Bit	mit
16-Bit mit Vorz.	keine	16 Bit	mit
32-Bit mit Vorz.	keine	32 Bit	mit
8-Bit ohne Vorz.	keine	8 Bit	ohne
16-Bit ohne Vorz.	keine	16 Bit	ohne
32-Bit ohne Vorz.	keine	32 Bit	ohne

* 1,0000 z. B. entspricht einer externen Vorgabe von 10000. Die letzten vier Dezimalstellen bilden den Nachkommateil.

Display

Im Gruppenfeld **Display** können Sie Einstellungen für die Parametrierung über Keypad vornehmen.

- In das Eingabefeld **Text** können Sie für die Codestelle einen Displaytext eingeben, der im Keypad angezeigt wird. Der Text wird auf eine Länge von 13 Zeichen begrenzt.
- Ist das Auswahlkästchen **Bestätigung** markiert, so muß auf dem Keypad zur Übernahme des Codestellenwertes der jeweiligen Codestelle die SHIFT- und PRG-Taste gleichzeitig gedrückt werden. Bei einer Codestelle mit Nur-Lese-Zugriff wird diese Einstellung nicht berücksichtigt.
 - RSP
Codestelle kann nur mit Reglersperre verändert werden.
 - SPS Stopp
Codestelle kann nur bei gestoppter SPS verändert werden.

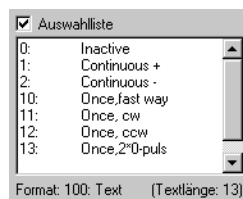
Parametrierprogramm

Im Gruppenfeld **Parametrierprogramm** können Sie Codestellentexte eingeben, die im Parametrierprogramm GDC sowie im Parameter-Monitor angezeigt werden.

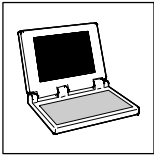
- Bei der Erstellung der Gerätebeschreibung für GDC kann einer der drei möglichen Texte ausgewählt werden.
- Diese Eintragungen sind sinnvoll, um GDC-Beschreibungsdateien in verschiedenen Landessprachen zu erstellen.

Auswahlliste

Wenn Sie das Auswahlkästchen **Auswahlliste** markieren, können Sie im zugehörigen Texteingabefeld Codestellenwerten frei definierbare Auswahltexte zuweisen:



- Bei einer Codestelle mit Nur-Lese-Zugriff werden diese Einstellungen nicht berücksichtigt.
- Werden Lenze-Funktionsblöcke aus Bibliotheken verwendet, stehen für die Erstinstanzen fertige Auswahllisten zur Verfügung. Sie können diese Auswahllisten übernehmen oder erweitern bzw. verändern.



- Sie können über den Typ-Parametermanager Auswahllisten vordefinieren, die im Instanz-Parametermanager übernommen werden.
- Wenn Sie vordefinierte Listen verwenden, werden die Listen automatisch aktualisiert, wenn Änderungen in den Ursprungsdateien (Bibliotheken, Funktionsbaustein-POE's) erfolgt sind.
- Wenn Sie eine vordefinierte Liste verändern, wechselt die Hintergrundfarbe der Liste von grau auf gelb. Es erfolgt dann keine automatische Aktualisierung mehr.
- Es können nur die Codestellenwerte, die in der Auswahlliste definiert worden sind, über die Parametrierumgebung angesprochen werden.

Schaltfläche "Erstinstanz"

Wenn Sie die Schaltfläche **Erstinstanz** betätigen, wird in das Eingabefeld Codestelle die bekannte Parametercodestelle eingefügt. Die Eingabefelder **Codest. Max./Codest. Min.** und **int. Max./int. Min.** werden für Eingaben deaktiviert (und in grau dargestellt).

Schaltfläche "Standard"

Beim Betätigen der Schaltfläche **Standard** werden die Standardeinstellungen der Variablen übernommen.

Dies bedeutet:

- Bei einer Variablen, die im Typ-Parametermanager eingestellt wurde, werden die Einstellungen übernommen. Es braucht dann nur noch die Codestelle vergeben werden.
- Bei einer Variablen, die noch nicht bekannt ist, werden die Grenzen des im Deklarationseditor eingestellten Datentyps übernommen.

Werden nachträglich Änderungen durch den Anwender vorgenommen, wechselt die Hintergrundfarbe des jeweiligen Eingabefeldes auf gelb, um eine Änderung gegenüber der Typdefinition kenntlich zu machen.

Nachträgliche Änderungen im Typ-Parametermanager oder Datentypänderungen im Deklarationseditor werden automatisch für die jeweilige Codestelle aktualisiert.

8.4.3.5 Umgang mit Lenze Funktionsblöcken

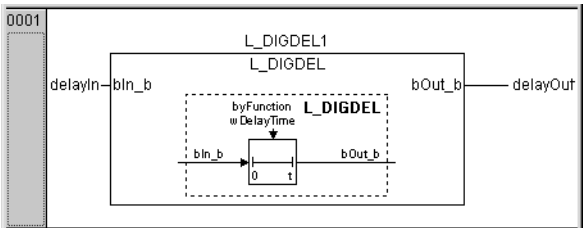
Wenn Sie in POE's vom Typ Programm Lenze-Funktionsblöcke verwenden, so können Sie den **VAR CONSTANT RETAIN** Variablen dieser FB's komplett die bekannten Parameter-Codestellen (z. B. vom 9300 Servo-Umrichter V2.0) über den Parametermanager zuweisen. Diese Zuweisungen können natürlich auch wieder aufgehoben werden.

Damit diese Zuweisung erfolgen kann, muß im Deklarationseditor der POE vom Typ Programm der richtige Instanzname verwendet werden:

- Der Instanzname ist der Name des Funktionsblocks gefolgt von einer Zahl (1, 2, 3, ...)
- Diese Syntax gilt für alle Lenze-Funktionsblöcke.
- Wird diese Syntax nicht eingehalten, so können den Instanzen nur Anwendercodestellen (C3000 - C7999) zugewiesen werden.

Beispiel der genauen Deklaration anhand des Funktionsblockes L_DIGDEL für die 9300 Servo PLC:

Wenn Sie z. B. den Funktionsblock **L_DIGDEL** aus der Lenze-Bibliothek **LenzeDrive.lib** in einer POE vom Typ Programm verwenden, dann muß der FB-Instanz-Name genau **L_DIGDEL1** heißen (**L_DIGDEL1** von Funktionsblocktyp **L_DIGDEL**):





Drive PLC Developer Studio

Ressourcen

- Nur bei der richtigen Auswahl des FB-Instanz-Namens können die bekannten Parameter-Codestellen (siehe nachfolgende Tabelle) der **9300 Servo PLC** durch Betätigen der Schaltfläche **Erstinstanz** dem Funktionsblock zugewiesen werden.
- Weitere Instanzen heißen dann **L_DIGDEL2**, **L_DIGDEL3**, usw.

Beachten Sie, daß außer **L_DIGDEL1** nur noch der Instanz **L_DIGDEL2** die Parameter-Codestellen der **9300 Servo PLC** vergeben werden können, weil diese PLC nur zwei Instanzen dieses Funktionsblockes beinhaltet. Weiteren Instanzen (z. B. **L_DIGDEL3**) können nur noch Anwendercodestellen zugewiesen werden.



Tip!

Die verfügbaren Instanzen eines Funktionsblocks sind in der Online-Hilfe im Teil "Lenze Funktionsbibliotheken" beschrieben.

Werden Variablen aus Lenze FB's mit Codestellen verknüpft, sind die Einstellbereiche und Skalierungen entsprechend der Dokumentation einzustellen.

Beispiel: Instanzen des Funktionsblockes L_DIGDEL der 9300 Servo PLC

Variablenname	L_DIGDEL1	L_DIGDEL2		Einstellbereich	Lenze
byFunction	C0720			0 ... 2	2
		C0725			0
wDelayTime	C0721	C0726		0.001 ... 60.000 s	1.000



Tip!

Die Parametervariablen können im SPS-Programm nur gelesen, nicht aber beschrieben werden.

Beispiel:

```
LD L_DIGDEL1.byFunction
ST variable_x
```

Es kann nur über die vergebene Codestelle auf die **VAR CONSTANT RETAIN**-Variable zugegriffen werden.

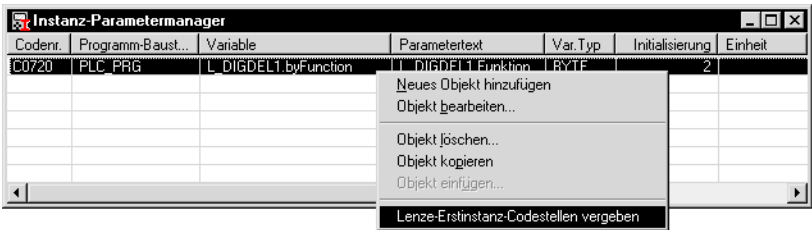
- Mit Hilfe des FBs **L_ParWrite** aus der Funktionsbibliothek "LenzeDrive.lib" kann auf Codestellen im SPS-Programm geschrieben werden.

Übernahme aller Lenze-Codestellen eines Funktionsblocks

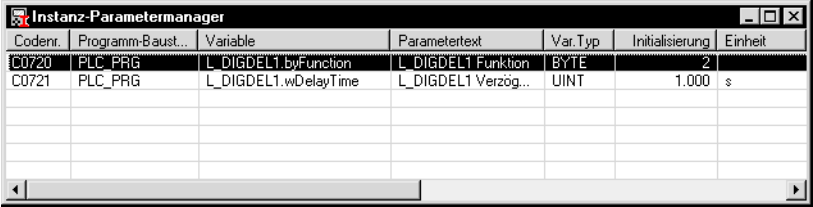
Im Dialogfeld *Instanz-Parametermanager* wird durch Betätigen der Schaltfläche **Erstinstanz** nur der ausgewählten Variable die entsprechende Codestelle zugeordnet.

Wenn Sie alle Lenze-Codestellen der Funktionsblock-Instanz übernehmen möchten, markieren Sie im Instanz-Parametermanager eine Variable des entsprechenden Funktionsblocks und wählen Sie den Befehl **Einfügen→Lenze-Erstinstanz-Codestellen vergeben**.

- Dieser Befehl ist auch über das Kontextmenü verfügbar:



Nach Ausführung des Befehls sind alle Codestellen der Funktionsblock-Instanz im Instanz-Parametermanager angelegt (in diesem Beispiel L_DIGDEL1.wDelayTime):



Codenr.	Programm-Baust...	Variable	Parametertext	Var.Typ	Initialisierung	Einheit
C0720	PLC_PRG	L_DIGDEL1.byFunction	L_DIGDEL1 Funktion	BYTE	2	
C0721	PLC_PRG	L_DIGDEL1.wDelayTime	L_DIGDEL1 Verzög...	UINT	1.000	s

8.4.3.6 Zugriff auf Arrays über Subcodestellen

Wenn Sie mit Hilfe von Codestellen (z. B. durch Verwendung des Bedienterminals **Keypad**) auf einzelne Felder eines Arrays zugreifen möchten, so geschieht dies über sogenannte Subcodestellen. Subcodestellen sind eine Untermenge von Codestellen einer Mastercodestelle.

Beispiel:

C6000	Mastercodestelle
C6000/1	Subcodestelle 1
C6000/2	Subcodestelle 2
...	...
C6000/n	Subcodestelle n

Anlegen von Subcodestellen

Subcodestellen können mit dem Parametermanager nur Variablen vom Typ **ARRAY OF** zugewiesen werden, hierbei ist folgendes zu beachten:

- Sie brauchen nur die Mastercodestelle definieren, die Subcodestellen werden dann automatisch entsprechend der Länge des Arrays angelegt.
- Subcodestelle 1 ist immer das erste Element des Arrays.
- Die Subcodestellen werden global über die Mastercodestelle eingestellt, d.h. jede Subcodestelle hat die gleiche Konfiguration wie die Mastercodestelle. Da dies auch für die Anzeigetexte gilt, sollten Sie einen allgemeinen Text wählen, damit es später nicht zu Mißverständnissen kommt.
- Eine Einstellung der einzelnen Subcodestellen ist über den Parameter-Monitor möglich. (8-7)

Beispiel: Zusammenhang zwischen Feld-Variablen und Subcodestellen:

Variable im SPS-Programm:	zugewiesene Codestelle:
X: ARRAY [0..4] OF INT	C6000
Felder des Arrays:	automatisch erzeugte Subcodestellen:
X[0]	C6000/1
X[1]	C6000/2
X[2]	C6000/3
X[3]	C6000/4
X[4]	C6000/5



Drive PLC Developer Studio

Ressourcen

8.4.3.7 Daten exportieren

Symbol: -	Menübefehl: Extras→CSV Export	Tastaturbefehl: -
-----------	-------------------------------	-------------------

Ist der Instanz-Parameter aktiv können Sie mit diesem Menübefehl die Inhalte des Instanz-Parametermanagers exportieren.

Die Export-Datei wird im gleichen Ordner der DDS Projektdatei *.pro mit dem Namen *.csv gespeichert. CSV (comma sequented values) gibt die Daten des Instanz-Parametermanagers in der gleichen Spaltenform wieder. Um die Datei nachträglich zu bearbeiten, kann sie z. B. mit Excel geöffnet werden.



Tip!

Öffnen Sie die Datei nicht mit Doppelklick direkt im Explorer, sondern über den Menübefehl **Datei→Öffnen** in Excel. Dadurch kann die exportierte Datei besser konfiguriert werden.

8.4.4 Typ-Parametermanager



Der Typ-Parametermanager befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.

Werden in einem Projekt Bibliotheken oder POE's vom Typ Funktionsbaustein verwendet, können mit Hilfe des Typ-Parametermanager u. a. die Grenzen/Skalierungen der Variablen vordefiniert werden.

Wird ein Funktionsblock, der im Typ-Parametermanager erstellt wurde, in einer POE vom Typ Programm aufgerufen, dann werden die im Typ-Parametermanager definierten Einstellungen bei Betätigung der Schaltfläche **Standard** im Instanz-Parametermanager übernommen.



Tip!

Der Umgang mit dem Typ-Parametermanager ist analog dem des Instanz-Parametermanagers mit dem Unterschied, daß Sie statt einer Funktionsblock-Instanz einen Funktionsblock-Typ auswählen. Lesen Sie daher zum Umgang mit dem Typ-Parametermanager bitte die Kapitel zum Instanz-Parametermanager. (8-13)

8.4.5 Skalerroutinen

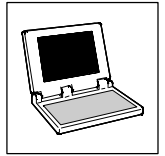
Codestellen können Sie mit einer Skalerroutine verknüpfen.

- Eine Skalerroutine ist ein vom Anwender geschriebenes IEC 61131-Programm zur Skalierung dieser Parameter/Variablen.

Skalerroutine für Codestellen mit Schreibzugriff

Verändert sich der Wert einer Codestelle, verändert sich auch der Wert der damit verbundenen Variable. Bei (jedem) Schreiben der Codestelle kann eine vom Anwender erstellte Skalerroutine gestartet werden.

- Soll beim Schreibzugriff auf eine Codestelle die Abarbeitung der zugehörigen Skalerroutine aufgrund einer Bedingung abgewiesen werden, so muß in der Skalerroutine der Wert der Variablen `g_bScaleFunctionError` auf **TRUE** gesetzt werden.
Die globale Variable `g_bScaleFunctionError` vom Typ **BOOL** ist in der **Standard.lib** enthalten.



Skalierroutine für Codestellen mit Lesezugriff

Wird auf eine Codestelle mit einem Parametriertool (z. B. GDC, Keypad oder HMI) lesend zugegriffen, kann eine vom Anwender erstellte Skalieroutine gestartet werden.

- In der Skalieroutine muß dazu auf die entsprechende Skaliervariable geschrieben werden.
- Der Wert der Skaliervariable wird nach der linearen Skalierung über die Codestelle ausgegeben.
- Die Skaliervariable ist eine globale Variable der **Standard.lib** und muß entsprechend dem Datentyp ausgewählt werden:

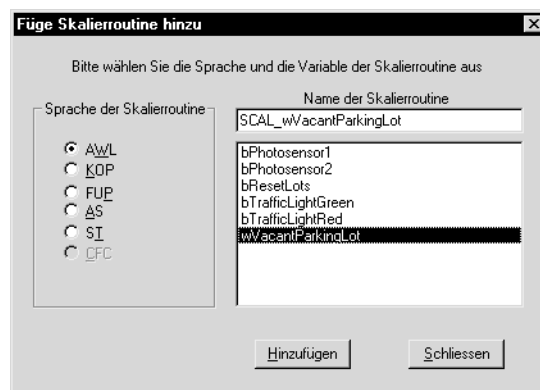
```
g_fScaleFunctionReal : REAL;
g_fScaleFunctionByte : BYTE;
g_fScaleFunctionSint : SINT;
g_fScaleFunctionInt : INT;
g_fScaleFunctionWord : WORD;
g_fScaleFunctionDint : DINT;
g_fScaleFunctionDword : DWORD;
g_fScaleFunctionBool : BOOL;
```

Beispiel:

- Ist die Anzeige-Codestelle einer Variablen vom Typ **BYTE** zugewiesen, so muß in der entsprechenden Skalieroutinen auf **g_fScaleFunctionByte** geschrieben werden.

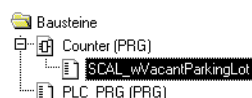
Skalierroutine erstellen

1. Markieren Sie im *Object Organizer* in der Registerkarte **Bausteine** den Funktionsblock, dem Sie eine Skalieroutine hinzufügen möchten.
2. Wählen Sie **Projekt→Skalierroutine hinzufügen**, um das Dialogfeld *Füge Skalieroutine hinzu* zu öffnen.
3. Wählen Sie im Dialogfeld *Füge Skalieroutine hinzu* die Programmiersprache und die entsprechende Variable aus. Der Name der Skalieroutine wird automatisch vergeben ("SCAL_" + Name der Variable).



4. Wählen Sie die Schaltfläche **Hinzufügen**, um die Skalieroutine hinzuzufügen bzw. **Schließen**, um den Vorgang abzubrechen und das Dialogfeld zu schließen.

Die Skalieroutine wird im *Object Organizer* unterhalb des entsprechenden FBs angehängt. Per Doppelklick öffnen Sie die Skalieroutine im Editor, wo Sie dann die entsprechende Programmierung vornehmen können.





Drive PLC Developer Studio

Ressourcen

Verwendung von Systembausteinen in Skalieroutinen

Wird ein Systembaustein in einer Skalieroutine verwendet, muß dieser mindestens in einer weiteren POE verwendet werden. Es wird andernfalls kein Prozessabbild für den Systembaustein erstellt. Als POE kann z.B. PLC_PRG oder eine andere POE, die durch eine Task aufgerufen wird verwendet werden.

Werden Systembausteine nur in Skalieroutinen oder System-POEs verwendet, wird für diese kein Prozessabbild erstellt, da diese Bausteine keiner Task zugeordnet sind.

Auch wenn ein Systembaustein in weiteren Tasks verwendet wird und damit ein Prozeßabbild erstellt wird, kann es zu Problemen kommen (z.B. bei PLC_Coldstart).

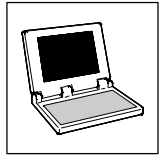
Ab DDS 2.0 wird bei der Verwendung von Systembausteinen in Skalieroutinen und System-POEs eine Warnung ausgegeben

Die Verwendung von Skalieroutinen in globalen Funktionsblockinstanzen führt zu einem PI Trip auf dem Zielsystem. Ab DDS 2.0 wird bei der Verwendung einer Skalieroutine in einer globalen Funktionsblockinstanz ein Compilerfehler ausgegeben.



Hinweis!

Überprüfen Sie im Prozeßabbildmonitor, daß für den Systembaustein, der in der Skalieroutine verwendet wird ein Prozeßabbild erstellt wird.



8.5 Prozeßabbild



Das Prozeßabbild befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.

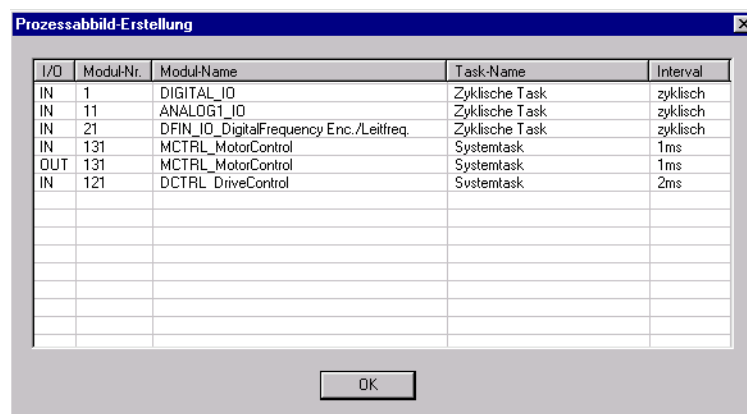
- Als Prozeßabbild bezeichnet man eine Momentaufnahme vom Zustand der I/O's aller im Zielsystem verwendeten Systembausteine.



Tip!

- Das Prozeßabbild der Lenze Zielsysteme besteht aus den I/O-Zuständen der Systembausteine, die in der Steuerungskonfiguration vom Anwender eingetragen wurden. (Siehe auch Kapitel 8.6, "Steuerungskonfiguration",  8-26)
- Einige Systembausteine werden immer vom Laufzeitsystem abgearbeitet. Diese Systembausteine sind einer bestimmten (System-)Task mit festem Zeitintervall zugeordnet. (Siehe jeweilige SB-Beschreibung im Handbuch zum entsprechenden Lenze Zielsystem.)

- Durch Doppelklick auf das Objekt **Prozeßabbild** im *Object Organizer* wird das Dialogfeld *Prozeßabbild-Erstellung* geöffnet:



Das Dialogfeld *Prozeßabbild-Erstellung* zeigt Ihnen an, in welcher Task das Prozeßabbild eines Systembausteins erstellt bzw. geschrieben wird.



Hinweis!

Ist ein vermeidlich verwendeter Systembaustein nicht in diesem Dialogfeld aufgeführt, wird kein Prozeßabbild erstellt und der Systembaustein wird auch nicht aufgerufen bzw. abgearbeitet!

Mit Hilfe des Dialogfeldes *Prozeßabbild-Erstellung* können Sie so schnell feststellen, warum Daten in einem Systembaustein nicht verarbeitet werden.

Ein Prozeßabbild für die Ausgänge eines Systembausteins wird nur erstellt, wenn schreibend auf die Ausgänge zugegriffen wird.

Wenn nur lesend zugegriffen wird, ist das Ereignis nicht im Prozeßabbild aufgeführt.



Drive PLC Developer Studio

Ressourcen

I/O	Anzeige, ob es sich um die Eingänge (IN) oder Ausgänge (OUT) des Systembausteins handelt.
Modul-Nr.	Modulnummer des entsprechenden Systembausteins (Modulnummern siehe entsprechendes Handbuch zum Zielsystem bzw. Online-Hilfe, Kapitel "Lenze Zielsysteme")
Modul-Name	Name des Systembausteins
Task-Name	Name der Task, in der das entsprechende Prozeßabbild eines Systembausteins erstellt bzw. geschrieben wird. • Entweder der in der Taskkonfiguration gewählte Taskname oder die Zyklische Task bzw. Systemtask (bestimmte Systembausteine sind der Systemtask zugeordnet). • Das System ermittelt für den Systembaustein automatisch den optimierten Zeitzyklus.
Interval	Diese Spalte gibt an, in welchem Zeitzyklus das Prozeßabbild aktualisiert wird.



Tip!
Prüfen Sie vor dem Download, ob auch alle verwendeten Systembausteine im Dialogfeld *Prozeßabbild-Erstellung* aufgeführt sind. Fehler können hierdurch schnell aufgedeckt werden.

8.5.1 Erstellung des Prozeßabbildes

- Das Prozeßabbild setzt sich zusammen aus:
- Prozeßeingangsabbild = Zustand aller Systembaustein-Eingänge
 - Prozeßausgangsabbild = Zustand aller Systembaustein-Ausgänge

Zyklische Task (PLC_PRG)

- Verwenden Sie nur eine Zyklische Task, so wird für die in dieser Task verwendeten Systembausteine das Prozeßeingangsabbild am Anfang der Zyklischen Task eingelesen und das Prozeßausgangsabbild am Ende der Zyklischen Task zurückgeschrieben.
- Während der Dauer der Abarbeitung des Zielsystems sind die Zustandsinformationen des Prozeßeingangsabbildes konstant, Datenkonsistenz ist gewährleistet.
 - Das Aktualisieren des Prozeßabbildes ist allerdings von der Abarbeitungszeit der Zyklischen Task abhängig, also von der Länge des Programms und dessen Komplexität.

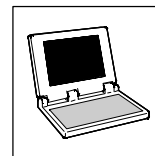
Zeitgesteuerte Task (INTERVAL)

- Verwenden Sie INTERVAL-Tasks, wird für die in dieser Task verwendeten Systembausteine das Prozeßeingangsabbild am Anfang der INTERVAL-Task eingelesen und das Prozeßausgangsabbild am Ende der INTERVAL-Task zurückgeschrieben.
- Wird ein bestimmter Systembaustein in mehreren INTERVAL-Tasks verwendet, wird das zugehörige Prozeßabbild in der schnelleren INTERVAL-Task erstellt (unabhängig davon, ob dieser Systembaustein auch in der Zyklischen Task verwendet wird).



Tip!
Beachten Sie, daß die INTERVAL-Task langsamer sein kann als die Zyklische Task.
Für das Prozeßabbild sollte ggf. eine schnelle Task eingerichtet werden, um nur das Prozeßabbild zeitgerecht zu aktualisieren.

- Beispiel:**
Analog-Eingang 1 wird in der Zyklischen Task (Laufzeit: 40 ms) und in einer INTERVAL-Task (Laufzeit: 3 ms) verwendet.
- Das Prozeßabbild für Analog-Eingang 1 wird nun alle 3 ms aktualisiert. Die Zyklische Task erhält alle 3 ms einen aktuellen Wert innerhalb ihrer 40 ms dauernden Abarbeitung.



Tip!

Beachten Sie, daß es zu Dateninkonsistenz kommen kann, wenn an mehreren Stellen innerhalb der Zyklischen Task der Analog-Eingang 1 verwendet wird:

- Durch die Aktualisierung des Prozeßabbildes alle 3 ms können innerhalb der 40 ms dauernden Abarbeitung der Zyklischen Task unterschiedliche Eingangswerte für den Analog-Eingang 1 vorliegen.

Ereignisgesteuerte Task (EVENT)

- Verwenden Sie Systembausteine in einer ereignisgesteuerten Task , dadurch wird dessen Prozeßabbild in der ereignisgesteuerten Task erstellt.
- Verwenden Sie in zwei ereignisgesteuerten Tasks die gleichen Systembausteine, die nicht auch in einer zeitgesteuerten Task verwendet werden, dadurch wird dessen Prozeßabbild in der Zyklischen Task erstellt.



Tip!

Weitere Informationen zu Tasks finden Sie im Kapitel 8.8, "Taskkonfiguration". (□ 8-32)

Interruptgesteuerter Task

- Verwenden Sie nur in einer ereignisgesteuerten Task Systembausteine, dadurch wird dessen Prozeßabbild in der ereignisgesteuerten Task erstellt.
- Verwenden Sie in einem ereignisgesteuerten Tasks und in einem interruptgesteuerten die gleichen Systembausteine, die nicht auch in einer zeitgesteuerten Task verwendet werden, dadurch wird dessen Prozeßabbild in der Zyklischen Task erstellt.



Drive PLC Developer Studio

Ressourcen

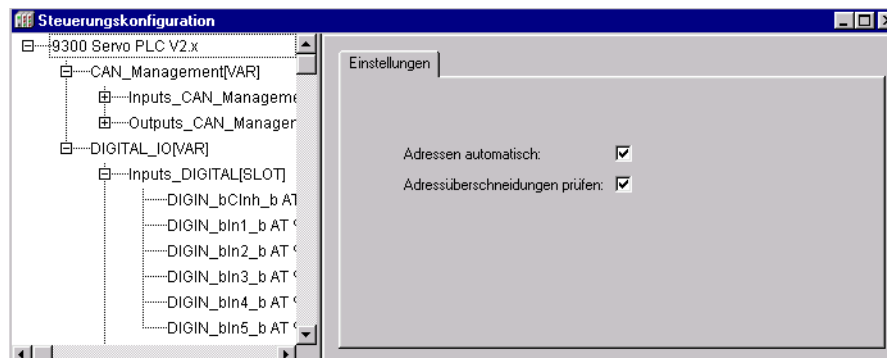
8.6 Steuerungskonfiguration



Die Steuerungskonfiguration befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*. Sie stellt die Ressourcen des Zielsystems zusammen.

Über das Dialogfenster *Steuerungskonfiguration* wird das gewählte Zielsystem für das Projekt beschrieben.

Die Ressourcen des Zielsystems werden durch die verwendeten Systembausteine beschrieben. Die zur Verfügung stehenden Systembausteine sind vom ausgewählten Zielsystem abhängig.



- Die Konfiguration wird im Editor als Baumstruktur dargestellt. Sie kann über Menübefehle und Dialoge bearbeitet werden.
- Die Systembausteine werden in der Steuerungskonfiguration als Unterelemente an das Zielsystem angehängt.
- Ein- und Ausgänge erscheinen im Editor mit einer IEC-Adresse. Damit kann auf die Ein- und Ausgänge zugegriffen werden. Zur Kennzeichnung kann im Standardfall für jeden Ein- und Ausgang ein symbolischer Namen vergeben werden, der anschließend vor der IEC-Adresse steht.
- Werden Projekte geöffnet, die mit einer älteren DDS-Version erstellt wurden und enthalten diese eine Steuerungskonfiguration können diese in das neue Format der **Vereinheitlichten Steuerungskonfiguration** überführt werden. (8-28)
- Für die Programmerstellung ist es wichtig, die benötigten Systembausteine des gewählten Lenze Zielsystems einzufügen.

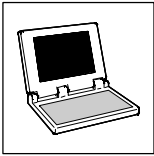
Systembausteine

Systembausteine sind im Gerät fest integrierte Schnittstellen/Bausteine (quasi Hardware-Schnittstellen), die nicht über das DDS programmiert werden können (z. B. MCTRL_MotorControl, DIGITAL_IO, DCTRL_DriveControl, AIF_IO, ANALOG1_IO, ...)

- Anhand der jeweiligen Systembausteinbeschreibung überprüft das DDS, ob die im Programm verwendeten IEC-Adressen auch wirklich in dem Zielsystem existieren.

Die Systembausteine in der Steuerungskonfiguration bestehen aus einer Reihe von Elementen, die entweder Ein- oder Ausgänge sind.

- Jedem Ein-/Ausgang ist ein symbolischer Name (Bezeichner), der - wenn nötig - verändert werden kann, und eine IEC-Adresse zugeordnet. Dadurch kann jeder Ein-/Ausgang über direkte Adressierung (z. B. %IX1.0.1) oder System-Variablen (DIGIN_bIn1_b) im Programm angesprochen werden.



8.6.1 Arbeiten in der Steuerungskonfiguration

Der Konfigurationseditor besteht aus zwei Fenstern.

Im linken Fenster sind die Ein-/und Ausgänge der Module strukturiert dargestellt. Wird ein Baustein ausgewählt, werden die Daten im rechten Fenster unter Basisparameter dargestellt.

Auf oberster Ebene im Dialogfenster *Steuerungskonfiguration* steht der Name des ausgewählten Zielsystems.

Elemente, die von einem Pluszeichen angeführt werden, sind Organisationselemente und enthalten Unterelemente.

- Zum Einblenden der Unterelemente klicken Sie auf das Pluszeichen. Ein erneuter Klick auf das nun angezeigte Minuszeichen blendet die Unterelemente wieder aus.
- Per Doppelklick auf ein Element können Sie die Bezeichnung der Ein-/Ausgangsvariablen ändern.

Die Variablen/direkten Adressen der Systembaustein-Ein-/Ausgänge können nun als System-Variablen in die POE's zur Programmierung aufgerufen werden.



Tip!

Die wichtigsten Befehle zum Dialogfenster *Steuerungskonfiguration* finden Sie auch im Kontextmenü (rechte Maustaste) oder <Umschalt>→<F10>.

Steuerungskonfiguration im Online-Modus



Hinweis!

Nur wenn das Zielsystem läuft, zeigt die Steuerungskonfiguration im Online-Modus die Zustände der Ein- und Ausgänge der Steuerung an.

- Hat ein boolscher Ein- bzw. Ausgang den Wert `TRUE`, wird das Kästchen vor dem Ein- bzw. Ausgang blau dargestellt.
- Variablenwerte werden mit dem aktuellen Wert angezeigt.

Die boolschen Eingänge können per Mausklick getoggelt werden, bei anderen Eingängen erscheint ein Dialogfeld zum Eingeben des neuen Wertes. Der neue Wert wird sofort nach Bestätigung durch **OK** in der Steuerung gesetzt.

8.6.1.1 Eigenschaften

Symbol:	-	Menübefehl:	Extras→Eigenschaften	Tastaturbefehl:	-
---------	---	-------------	----------------------	-----------------	---

Dieser Befehl öffnet ein Dialogfeld, in dem Sie zum selektierten Ein-/Ausgangsmodul in der Steuerungskonfiguration weitere Informationen (z. B. einen Kommentar) hinterlegen können.

8.6.1.2 Element einfügen

Symbol:	-	Menübefehl:	Einfügen→Element einfügen	Tastaturbefehl:	-
---------	---	-------------	---------------------------	-----------------	---

Dieser Befehl fügt das ausgewählte Element in die Steuerungskonfiguration vor dem selektierten Element ein.

8.6.1.3 Unterelement anhängen

Symbol:	-	Menübefehl:	Einfügen→Unterelement anhängen	Tastaturbefehl:	-
---------	---	-------------	--------------------------------	-----------------	---

Dieser Befehl fügt das ausgewählte Element in die Steuerungskonfiguration als letztes Unterelement an das selektierte Element an.



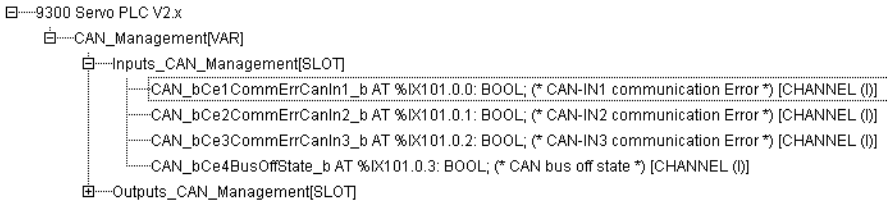
Drive PLC Developer Studio

Ressourcen

8.6.1.4 Element ersetzen

Symbol: -	Menübefehl: Extras→Element ersetzen	Tastaturbefehl: -
-----------	-------------------------------------	-------------------

Ist die Konfigurationsdatei richtig definiert, können Sie ein selektiertes Element im Konfigurationsbaum durch ein anderes ersetzen. Das umfaßt auch das Umschalten von Kanälen, daß sie mit der Konfiguration als Ein- oder Ausgang eingesetzt werden können.



8.6.1.5 Adressen berechnen

Symbol: -	Menübefehl: Extras→Adressen berechnen	Tastaturbefehl: -
-----------	---------------------------------------	-------------------



Hinweis!

Diese Funktion wird nicht von allen Zielsystemen unterstützt.

Um diesen Menüpunkt ausführen zu können, muß in den allgemeinen Einstellungen der Steuerungskonfiguration das Kontrollkästchen *Adressen automatisch* markiert sein.

Das Berechnen der Adressen gilt ab dem selektierten Modul und erfaßt alle folgenden Elemente.

8.6.1.6 Zu der Standardkonfiguration zurückkehren

Symbol: -	Menübefehl: Extras→Standardkonfiguration	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl wird die eingestellte Konfiguration gelöscht und zu der ursprünglich, eingestellten Konfiguration zurückgekehrt.

8.6.1.7 Überführen alter Steuerungskonfigurationen

Symbol: -	Menübefehl: Extras→Überführen	Tastaturbefehl: -
-----------	-------------------------------	-------------------

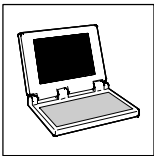
Wird ein Projekt geöffnet, für das mit einer älteren DDS-Version eine Steuerungskonfiguration erstellt wurde, kann das Projekt mit dem Befehl in das Format der aktuellen Steuerungskonfiguration überführt werden.

Überführen Sie das ältere Projekt nur, wenn Sie die Funktionen vom DDS 2.x benötigen und nutzen wollen.



Hinweis!

Die alte Konfiguration kann nicht mehr hergestellt werden.



8.6.2 Touch-Probe Interface

Bei einigen Zielsystemen ist es möglich digitale Eingänge wahlweise als Touch-Probe Eingang oder als einfachen digitalen Eingang zu verwenden.

Die Konfiguration der Eingänge erfolgt über die Steuerungskonfiguration in DDS.



Hinweis!

Über das Feld Registerparameter kann die Quelle für den Last-Scan-Wert konfiguriert werden.

Bei der Verwendung von digitalen Eingängen als Touch-Probe sind folgende Punkte zu beachten.

- Wird ein digitaler Eingang als Touch-Probe verwendet kann er nicht zum Starten einer Task verwendet werden.
- Jeder Touch-Probe-Eingang belegt eine Task, dadurch wird die Anzahl der für die Anwendung zur Verfügung stehenden Interrupt-Tasks wird reduziert.

8.6.3 Konfiguration eines I/O-Moduls

Ist ein I/O-Modul im Konfigurationsbaum selektiert, erscheint das Dialogfeld *Basisparameter* mit folgenden Einträgen.

Modul-Id

Sie ist eine eindeutige Kennung des Moduls in der gesamten Konfigurationsumgebung. Sie wird aus der Konfigurationsdatei übernommen und ist nicht editierbar.

Knotennummer

Die Knotennummer ergibt sich aus einem Eintrag in der Konfigurationdatei. Liegt dort kein Eintrag vor, ergibt sie sich aus der Position im Konfigurationsbaum.



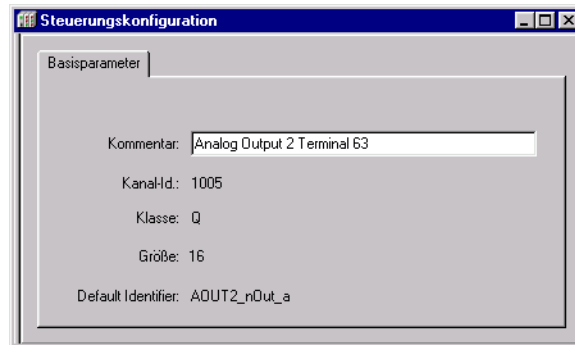
Drive PLC Developer Studio

Ressourcen

8.6.4 Konfiguration eines Kanals

Basisparameter eines Kanals

Ist ein Kanal (Channel) im Konfigurationsbaum selektiert, erscheint das Dialogfeld *Basisparameter*.



Kommentar

Zusätzliche Information zum Kanal. In dem Textfeld kann ein Kommentar editiert werden oder ein neuer eingegeben werden.

Kanal-Id

Global eindeutige Kennzeichnung des Kanals.

Klasse

Angabe über die Verwendung des Kanals

I	Eingang
Q	Ausgang
I&Q	Eingang und Ausgang
I Q	Eingang oder Ausgang (umschaltbar)

Größe

Bereichsgröße des Kanals

Default Identifier

Symbolischer Name des Kanals, der in der Konfigurationsdatei vergeben wird. Der Kanalname wird in die Konfigurationsdatei vergeben. Der Name kann im Konfigurationsbaum editiert werden.

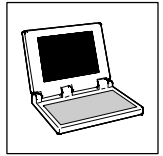
Kanalparameter

Dieses Dialogfeld dient entsprechend dem Modulparameter-Dialogfeld der Darstellung und Modifizierung der Parameterwerte des Kanals.

Wie bei den Modulen kann der durch ein anwendungsspezifisches Dialogfeld *Custom Parameters* ersetzt sein.

Bitkanäle

Die Basisparameter bei Bitkanälen enthalten nur das Eingabefeld **Kommentar**.



8.7 Task-Monitor



Der Task-Monitor befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*. Ist der PC online mit einem Zielsystem verbunden, so kann über den Task-Monitor die Laufzeit der einzelnen Tasks diagnostiziert werden. Im Offline-Modus wird im Task-Monitor der letzte Online-Zustand der Tasks angezeigt.

- Doppelklicken Sie im *Object Organizer* auf das Objekt **Task-Monitor**, um den Task-Monitor zu öffnen:

Taskmonitor											
Id	Name	letzte Lauf...	maximale ...	Auslastung	Zustand	Fehler	P...	Type	Inter...	Ereignis	Überwa...
✓ 0	Systemtask	0.50 ms	72.75 ms		Läuft	Kein Fehler					
✓ 1	Zyklische T...	30.25 ms	80.25 ms		Läuft	Kein Fehler		Zyklisch			10000 ms
✓ 3	Intervall_Task	8089.60 µs	8166.40 µs		Läuft	Kein Fehler	2	Intervall	10 ms		9 ms
ST 4	Event_Task	< 26 µs	< 26 µs		Geblockt	Überlauf	3	Ereignis		bVarEventtask_b	9 ms
✓ 2	Interrupt-Ta...	< 26 µs	< 26 µs		Läuft	Kein Fehler	1	Interrupt		%IX1.0.1	9 ms

Die Tasks werden zeilenweise mit folgenden Informationen angezeigt:

Spalte	Anzeige
Id	Id der Task (0 ... 9) sowie Status der Task: ✓ = Task ist bereit oder läuft ST = Task ist blockiert oder gesperrt
Name	Name der Task.
letzte Laufzeit	Laufzeit der letzten Taskabarbeitung.
maximale Laufzeit	Maximale Laufzeit, die während des Programmbetriebes aufgetreten ist.
Auslastung	Balkendiagramm mit der aktuell erfassten Taskzeit einer Intervall-Task. • Balkenfarbe grün: die aktuelle Taskzeit liegt unter 80 % der Intervallzeit. • Balkenfarbe rot: die aktuelle Taskzeit liegt über 80 % der Intervallzeit. • Die Spaltenbreite repräsentiert 100 % der Intervallzeit. • Der schwarze Strich zeigt die maximale Tasklaufzeit (peak hold) an, die während des Programmbetriebes aufgetreten ist. • Im Offline-Modus wird diese Zeit in % der Intervallzeit als Zahl angezeigt.
Zustand	Zustand der Task (läuft, geblockt oder unterbrochen).
Fehler	“Überlauf”: Die Task benötigt mehr Zeit als die eingestellte Überwachungszeit. “kein Fehler”: Die Task läuft innerhalb der eingestellten Überwachungszeit.
Priorität	Priorität der Task.
Type	Typ der Task (Intervall-, Zyklische-, Ereignis- oder Interrupt-Task).
Intervall	Intervallzeit einer Intervall-Task.
Ereignis	Ereignis einer Ereignis- bzw. Interrupt-Task.
Überwachungszeit	Überwachungszeit der Task



Tip!

Bei den angezeigten Tasklaufzeiten werden nicht die Nettozeiten, sondern die realen Laufzeiten mit allen Taskunterbrechungen in einer Auflösung von 25.6 µs dargestellt.

Zyklische Task/System-Task

- Die System-Task bearbeitet im Hintergrund notwendige, interne Systemaufgaben.
- Die Zyklische Task beinhaltet nicht nur die Task `PLC_PRG`, sondern auch die Aufgabe der Kommunikation mit dem PC.
- Die System-Task teilt sich mit der Zyklischen Task im Zeitscheibverfahren die niederpriore zyklisch abarbeitende Task.



Drive PLC Developer Studio

Ressourcen

8.8 Taskkonfiguration



Die Taskkonfiguration befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.

8.8.1 Definition einer Task

In der IEC 61131-3 Norm bezeichnet man eine Gruppierung von POE's des Typs PROGRAM als eine "Task" (engl.: Job, Aufgabe). POE's sind Softwareeinheiten, die zur Organisation des Projekts dienen (POE = ProgrammOrganisationsEinheit).

Die Task wird vom Laufzeitsystem ausgeführt und sorgt mit den in ihr enthaltenen Programm/en für die gewünschte Funktionalität des Laufzeitsystems.

Task-Arten

Lenze Zielsysteme (z. B. **9300 Servo PLC, Drive PLC**) unterstützen vier verschiedene Task-Arten:

- **Zyklische Task (PLC_PRG)**
PLC_PRG ist die Hauptschleife des Anwenderprogramms (Anzahl: 1).
 Sie ist immer aktiv, hat die niedrigste Priorität und kann nicht abgeschaltet werden.
 Sie können **PLC_PRG** in der gewünschten Programmiersprache erstellen und von **PLC_PRG** aus andere Programm-POE's aufrufen.
 Die Verwendung von **PLC_PRG** für ein Steuerungsprogramm ist nicht zwingend erforderlich.
 Die Laufzeit von **PLC_PRG** ist nicht äquidistant, d. h. sie hängt vom Programmumfang und von der aktuellen Rechnerbelastung ab.
- **Zeitgesteuerte Task (INTERVAL)**
 Sie können zeitgesteuerte Tasks einrichten, die periodisch gestartet werden.
 Die Periodendauer ist im Bereich 1 ms ... 16 s in 1 ms-Schritten einstellbar.
- **Ereignisgesteuerte Task (EVENT)**
 Sie können ereignisgesteuerte Tasks einrichten, die durch einen "boolschen Wechsel" einer definierbaren Variable oder durch einen Hardware-Eingang der PLC gestartet werden. Die Variable kann eine globale Variable oder eine Systemvariable sein.
 Reaktionszeit = 1 ms.
- **Interrupt Task**
 Sie können diese Task durch einen echten Hardware-Interrupt (digitaler Eingang) starten.
 Dieser Interrupt wird nicht über das IEC 61131-3 Programm sondern durch den Prozessor ausgelöst. Dies hat den Vorteil, daß die Task sehr schnell gestartet werden kann.
 Typische Reaktionszeit = 100 µs; "WorstCase"-Reaktionszeit = 250 µs.

Die Anzahl der Task (Zeitgesteuert, Ereignisgesteuert oder Interrupt) ist abhängig vom ausgewählten Zielsystem.



Hinweis!

Das Laufzeitsystem überwacht die Ausführungszeit aller Tasks.

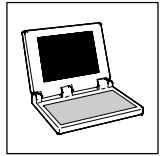
Kommt es z. B. durch große Rechenzeit oder häufige Unterbrechungen durch höher priore Tasks zu einer Ausführungszeit, die größer ist als die Zykluszeit einer Task, dann kommt es zu einer Systemstörung, die das Laufzeitsystem in Störung setzt.

Die Taskeigenschaften werden über das DDS eingestellt, wobei folgende Randbedingungen eingehalten werden müssen:

Ereignisgesteuerte Task (Interrupt/EVENT)

Bei einer ereignisgesteuerten Task muß zum Starten dieser Task ein wirklicher Wechsel (von **FALSE** auf **TRUE**) des boolschen Zustands der zugehörigen Variable oder des Hardware-Interrupts erfolgen.

- Der Anfangszustand **TRUE** der Variable startet die Task nicht!



Speicherung eines Start-Ereignisses

Wird eine Task vom Typ EVENT während der Abarbeitung erneut angefordert, dann wird die erneute Anforderung gespeichert und nach dem Beenden des Taskzyklus der ereignisgesteuerten Task sofort wieder gestartet. Es kann nur eine Anforderung gepuffert werden.

PRIORITY

Jede Anwendertask (nicht die **Zyklische Task**) hat eine definierte Priorität zwischen 0 und n (0 = Task startet nie; 1 = höchste Priorität ...n-1 = zweit niedrigste Priorität ...n = niedrigste Priorität).

- Eine Priorität kann nicht doppelt vergeben werden, Tasks mit gleicher Priorität können somit niemals vorkommen.
- Die Priorität einer Task kann **nicht** beim Betrieb des Zielsystems im Online-Modus geändert werden.
- Die Priorität einer Task entscheidet darüber, ob eine andere gerade laufende Task unterbrochen wird, oder nicht. Wird eine Task gestartet, deren Priorität kleiner ist als die der gerade laufenden Task, so muß sie warten und kann die laufende Task nicht unterbrechen.

8.8.2 Datenkonsistenz

Bei einem Multitasking-Betriebssystem oder allgemein bei allen Softwarestrukturen, die "nebenläufige" Prozesse mit unterschiedlichen Zykluszeiten aufweisen, besteht das Problem der Dateninkonsistenz, welches hier kurz erläutert werden soll.

Wann treten Dateninkonsistenzen auf?

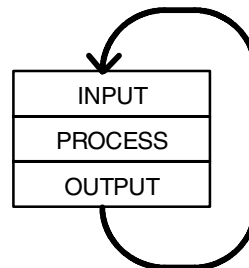
- Eine Variable A wird innerhalb einer Routine 1 an verschiedenen Stellen benutzt. Zur Laufzeit der Routine 1 wird Variable A durch eine Routine 2 verändert.
- Ein Datensatz besteht aus 4 Variablen, die zwecks Lesezugriff aus dem Speicher geholt werden müssen. Eine Interrupt-Service-Routine 2 verändert diesen Datensatz während des Lesezugriffs.

8.8.3 Einläufige Datenverarbeitung/EVA-Prinzip

Bei einer einläufigen Datenverarbeitung, wie z. B. bei einfachen SPS-Systemen, wird grundsätzlich nach dem EVA-Prinzip (Eingabe/Verarbeitung/Ausgabe)verfahren. Hier tritt das Problem der Dateninkonsistenz **nicht** auf!

Beim EVA-Prinzip besteht lediglich ein Datenverarbeitungszweig:

- Am Anfang der Schleife erfolgt das Einlesen aller notwendigen Daten.
- In der Mitte der Schleife erfolgt die Verarbeitung der Daten und die Berechnung der Ausgangswerte.
- Am Ende der Schleife (kurz vor dem "loopback") erfolgt die Ausgabe der Daten.



Bei diesem Verfahren sind die Eingangsdaten während der gesamten Verarbeitungszeit bis zum erneuten Einlesen am Anfang der Schleife unverändert und konstant. Da es keine weiteren Prozesse gibt, die während der Laufzeit der Schleife den EVA-Ablauf unterbrechen und u. U. auf die Daten des laufenden Prozesses zugreifen, kann es hier keine Dateninkonsistenzen geben!



Drive PLC Developer Studio

Ressourcen

8.8.4 Wenn ein Tasküberlauf zu einem Systemfehler führt

Wenn z. B. eine Endlosschleife oder zuviel Programminhalt in eine Task programmiert wurde, so daß die Task das Programm nicht in der zugewiesenen Intervall-Zeit abarbeiten kann, dann tritt ein Systemfehler in dem Zielsystem auf.

Es kann auch dadurch ein Systemfehler hervorgerufen werden, wenn eine laufende Task durch andere Tasks so häufig unterbrochen wird, daß sie ihr Programm in der vorgegebene Tasklaufzeit nicht mehr abarbeiten kann.

Ein Systemfehler bedeutet, daß ein Zielsystem in Störung geht.

- Das Verhalten der Motorregelung und der digitalen Ausgänge ist über **Projekt→Ausnahmeverhalten** einstellbar.
(Siehe auch Beschreibung System-POE's im Handbuch zum entsprechenden Lenze Zielsystem.)
- Erst nach einem TRIP-Reset oder Netzschalten kann das Programm wieder gestartet werden.

Mit einem TRIP-Reset (Keypad: STOP→RUN-Taste drücken; GDC: Codestelle C0043) wird das Lenze Zielsystem nach einem Tasküberlauf zurückgesetzt und die Fehlermeldung "Tasküberlauf" wird quittiert. Um keinen erneuten Tasküberlauf zu erhalten, muß das Zielsystem korrigiert werden.

Die Überwachung wird folgendermaßen eingestellt:

- Im Eigenschaftendialog der Task wird in das Eingabefeld **Überwachungszeit** die maximale Laufzeit der Task eingegeben.
- Intervall: Die Überwachungszeit liegt zwischen 0-100 % der Intervall-Zeit (Default=90 %) und kann von 1 ms bis 300 ms eingestellt werden.
(Bei einer 1 ms-Task beträgt die Überwachungszeit auch 1 ms = 100 % fix.)
- Event/Interrupt: Die Überwachungszeit kann nur kleiner 300 ms (Default=100 ms) eingestellt werden.
- Im Dialogfeld *Ausnahmeverhalten* (**Projekt→Ausnahmeverhalten**) wird in das Eingabefeld **Überwachungszeit** die maximale Laufzeit, die die Zyklische Task (PLC_PRG) max. laufen darf, eingegeben.

Auslösung der Überwachungszeit

- Eine Auslösung der Überwachungszeit löst eine Fehlermeldung in der PLC aus.
- Auf dem Keypad/im GDC die Codestelle C0168 wird diese Fehlermeldung angezeigt.
- Nach TRIP-Reset werden Fehlermeldung und SPS-Programm zurückgesetzt. Mit dem Startbefehl des DDS bzw. C2108=1 kann das Programm wieder gestartet werden.

8.8.5 Deklaration einer Task

Die Deklaration einer Task besteht aus dem Namen der Task, einem Eintrag für die Priorität, die die Task haben soll, und einem Eintrag für die Bedingung, unter der die Task ausgeführt werden soll.

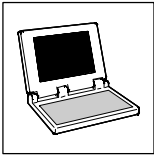
Die Bedingung kann entweder ein Zeitintervall (1 ms bis 16 s) sein, nach dem die Task ausgeführt werden soll, ein Ereignis (steigende Flanke am digitalen Eingang bzw. FALSE/TRUE-Wechsel einer globalen Variable) oder ein Hardware-Interrupt.

Zu jeder Task können Sie nun eine Folge von Programmen (POE's vom Typ Program) angeben, die von der Task aufgerufen werden sollen. Diese Programme werden in der angefügten Reihenfolge abgearbeitet.



Tip!

Die Einstellungen (Eigenschaften) einer Task können nicht im Online-Modus geändert werden!



Welche Task wird bearbeitet?

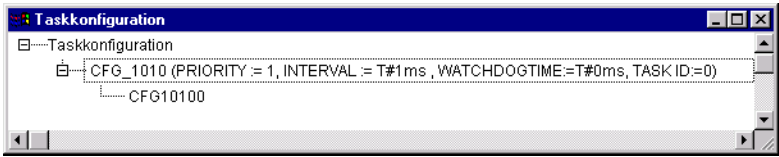
Für die Ausführung gelten folgende Regeln:

- Es wird die Task ausgeführt, deren Start-Bedingung zutrifft, das heißt, wenn z. B. die angegebene Intervallzeit abgelaufen ist oder bei Ereignissteuerung die angewählte Variable von **FALSE** auf **TRUE** wechselt.
- Haben mehrere Tasks eine gültige Start-Bedingung, dann wird die Task mit der höchsten Priorität ausgeführt.
- Mehrere Tasks mit gleicher Priorität sind nicht möglich (eingeschränktes Multitasking). Ausnahme: Priorität 0 = Task gesperrt.
- Wenn während der Abarbeitung einer Task die Start-Bedingung für eine Task höherer Priorität zutrifft, so wird die Task mit der niedrigeren Priorität für dessen Abarbeitung unterbrochen und erst nach Beendigung der Task mit höherer Priorität wieder fortgesetzt.

8.8.6 **Arbeiten in der Taskkonfiguration**

Die Taskkonfiguration befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.

Durch Doppelklick auf das Objekt **Taskkonfiguration** im *Object Organizer* wird in der rechten Fensterhälfte die aktuelle Taskkonfiguration im Taskeditor angezeigt.



Aufbau der Taskkonfiguration

Die Taskkonfiguration ist wie ein Verzeichnisbaum aufgebaut:

- In der obersten Hierachiestufe steht der Text "Taskkonfiguration".
- Eine Stufe darunter befindet sich eine Folge von Taskeinträgen (mit Name, Priorität, Intervall/Ereignis, Watchdog-Time, Task ID)
- Jeder Taskeintrag wiederum beinhaltet eine Folge von Programmaufrufen.
- Befindet sich vor einem Eintrag ein Pluszeichen, so beinhaltet dieser Eintrag Untereinträge, die durch Klicken auf das Pluszeichen eingeblendet werden können. Ein weiterer Klick auf das daraufhin angezeigte Minuszeichen blendet die Untereinträge wieder aus.
- Mit **Einfügen→Task einfügen** können Sie eine Task oberhalb der selektierten Task einfügen.
- Mit **Einfügen→Task anhängen** können Sie bei Selektion der obersten Ebene ("Taskkonfiguration") eine Task ans Ende der bestehenden Liste anhängen.
- Mit **Einfügen→Programmaufruf einfügen** fügen Sie der selektierten Task einen Programmaufruf hinzu. Werden der Task mehrere Programme angehängt, so werden diese POE's der Reihe nach abgearbeitet.
- Mit **Extras→Eintrag bearbeiten** können Sie je nach selektiertem Element die Taskeigenschaften oder den Programmaufruf editieren.
- Durch Klicken auf einen Task- oder Programmnamen oder durch Drücken der **<Leertaste>** setzen Sie einen Editierraum um den Namen. Sie können dann direkt im Taskeditor die Bezeichnung ändern.



Drive PLC Developer Studio

Ressourcen

8.8.6.1 Task einfügen / Task anhängen

Symbol: -	Menübefehl: Einfügen→Task einfügen	Tastaturbefehl: -
-	Einfügen→Task anhängen	-

Dieser Befehl fügt der Taskkonfiguration eine neue Task hinzu.

- Ist eine Task selektiert, steht der Befehl **Task einfügen** zur Verfügung, der die neue Task vor dem Cursor einfügt.
- Ist das Wort "Taskkonfiguration" selektiert, steht der Befehl **Task anhängen** zur Verfügung, der die neue Task ans Ende der bestehenden Liste anhängt.

Nach Auswahl eines der beiden Befehle öffnet sich das Dialogfeld *Taskeigenschaften*, in dem Sie die gewünschten Attribute eingeben können:

- **Name** der Task (max. 40 Zeichen)
- **Priorität** der Task
(Zahl zwischen 0 und n; 0 = Task gesperrt, 1 = höchste Priorität, n = niedrigste Priorität)
- **Intervall**, nach dem die Task wieder gestartet werden soll.
- **Ereignis** (Variable, die nach steigender Flanke eine Ausführung der Task bewirken soll).
Zur Auswahl des Ereignisses aus den deklarierten Variablen können Sie durch Klicken auf **Auswählen...** die Eingabehilfe aufrufen.
- **Interrupt**
Zur Auswahl eines echten Interrupt klicken Sie auf die Schaltfläche **Interrupts....**
- **Überwachungszeit** der Task

Wenn

- in den Textfeldern **Interrupt**, **Intervall** und **Ereignis** kein Eintrag vorgenommen wurde,
 - eine falsche Intervallzeit bzw. Priorität eingegeben wurde,
 - eine bereits vergebene Priorität eingegeben wurde,
- dann wird die **OK**-Schaltfläche nicht freigegeben.



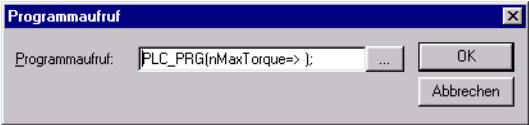
8.8.6.2 **Programmaufruf einfügen / Programmaufruf anhängen**

Symbol: -	Menübefehl: Einfügen→Programmaufruf einfügen Einfügen→Programmaufruf anhängen	Tastaturbefehl: -
-----------	--	-------------------

Mit diesen Befehlen öffnen Sie das Dialogfeld zum Eintrag eines Programmaufrufs zu einer Task in der Taskkonfiguration.

Mit **Programmaufruf einfügen** wird der neue Programmaufruf vor dem Cursor eingefügt und mit **Programmaufruf anhängen** ans Ende der bestehenden Liste angehängt.

Geben Sie in das Eingabefeld **Programmaufruf** einen gültigen Programmnamen Ihres Projekts an, oder öffnen Sie mit der Schaltfläche ... die Eingabehilfe zur Auswahl gültiger Programmnamen.



Wenn das ausgewählte Programm Eingabevariablen erfordert, dann geben Sie diese in der üblichen Form vom deklarierten Typ an (siehe Beispiel).

Beispiel für Eingabevariable:

prg(invar:=17)

8.8.6.3 **Eigenschaften**

Symbol: -	Menübefehl: Extras→Eigenschaften	Tastaturbefehl: <Eingabetaste>
-----------	---	---

Mit diesem Befehl öffnen Sie in der Taskkonfiguration je nach selektiertem Element entweder das Dialogfeld *Taskeigenschaften* oder das Dialogfeld *Programmaufruf*.

- Steht der Cursor auf einem Taskeintrag, und am Taskeintrag hängt noch keine Liste von Programmaufrufen an, dann können Sie auch durch Doppelklicken auf den Eintrag oder Betätigen der **<Eingabetaste>** das Dialogfeld *Taskeigenschaften* öffnen.
- Steht der Cursor auf einem Eintrag für einen Programmaufruf, dann können Sie auch durch Doppelklicken auf den Eintrag das Dialogfeld *Programmeintrag* öffnen.
- Durch Klicken auf einen Task- oder Programmnamen oder durch Drücken der **<Leertaste>** setzen Sie einen Editerrahmen um den Namen. Sie können dann direkt im Taskeditor die Bezeichnung ändern.



Drive PLC Developer Studio

Ressourcen

8.9 Watch- und Rezepturverwalter



Der Watch- und Rezepturverwalter befindet sich als Objekt in der Registerkarte **Ressourcen** im *Object Organizer*.



Mit Hilfe des Watch- und Rezepturverwalters können

- Werte von ausgesuchten Variablen angezeigt werden.
- Variablen mit bestimmten Werten vorbelegt und auf einmal in die Steuerung übertragen werden.
- Aktuelle Werte der Steuerung als Vorbelegung in den Watch- und Rezepturverwalter eingelesen und abgespeichert werden.

Hilfreich sind diese Funktionen z. B. für die Einstellung und Erfassung von Regelungsparametern.

- Im linken Bereich des Watch- und Rezepturverwalters werden alle erzeugten Watchlisten angezeigt und können durch Klicken oder mit den Pfeiltasten ausgewählt werden.
- Im rechten Bereich des Watch- und Rezepturverwalters werden die jeweils zugehörigen Variablen angezeigt.

8.9.1 Watch- und Rezepturverwalter im Offline-Modus

Im Offline-Modus können Sie im Watch- und Rezepturverwalter mehrere Watchlisten mit **Einfügen → Neue Watchliste** erzeugen.

Zum Eingeben der zu beobachtenden Variablen können Sie eine Liste aller Variablen mit der Eingabehilfe aufrufen oder Sie geben 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`

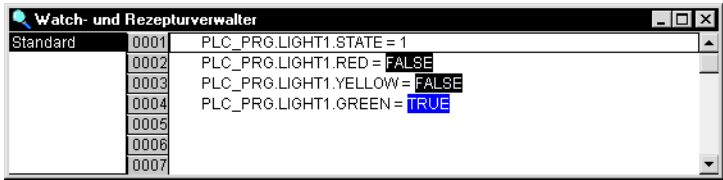
Die Variablen der Watchliste können mit konstanten Werten vorbelegt werden, d.h. im Online-Modus können diese Werte mit dem Menübefehl **Extras→Rezeptur schreiben** in die Variablen geschrieben werden. Dazu muß der konstante Wert mit `:=` der Variablen zugewiesen werden.

Beispiel:

`PLC_PRG.TIMER:=50`

8.9.2 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 einem Klick auf das Pluszeichen wird die Variable aufgeklappt. Anstelle des Pluszeichens wird nun ein Minuszeichen angezeigt. Mit einem Klick auf das Minuszeichen kann die Variable wieder zugeklappt werden.
- Ist eine Funktionsblock-Variable in der Watchliste markiert, wird das zugehörige Kontextmenü um die beiden Menüpunkte **Funktionsblock öffnen** und **Instanz öffnen** erweitert.
- Um neue Variablen einzugeben, können Sie die Anzeige durch den Befehl **Extras→Monitoring aktiv** ausschalten. Nachdem Sie die Variablen eingegeben haben, können Sie mit demselben Befehl das Anzeigen der Werte wieder aktivieren.
- Wenn Sie im Offline-Modus Variable mit einem konstanten Wert vorbelegt haben, können Sie mit **Extras→Rezeptur schreiben** diesen Wert in die Variable schreiben.
- Mit **Extras→Rezeptur lesen** wird die Vorbelegung der Variablen durch den aktuellen Wert der Variablen ersetzt.

8.9.3 Befehlsübersicht

8.9.3.1 Neue Watchliste

Symbol: -	Menübefehl: Einfügen→Neue Watchliste	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl fügt in den Watch- und Rezepturverwalter eine neue Watchliste ein.

- Geben Sie im erscheinenden Dialogfeld den gewünschten Namen der Watchliste ein.

8.9.3.2 Watchliste umbenennen

Symbol: -	Menübefehl: Extras→Watchliste umbenennen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl kann der Name einer Watchliste im Watch- und Rezepturverwalter geändert werden.

- Geben Sie im erscheinenden Dialogfeld den neuen Namen der Watchliste ein.

8.9.3.3 Watchliste speichern

Symbol: -	Menübefehl: Extras→Watchliste speichern	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl speichert eine Watchliste ab.

Nachdem der Befehl gewählt wurde, erscheint das Dialogfeld *Datei speichern unter*.

- Geben Sie im Eingabefeld **Dateiname** einen neuen Namen für die Datei ein.
- Im Eingabefeld **Dateityp** ist der Zusatz **"*.wtc"** bereits vorgegeben.

Die gespeicherte Watchliste können Sie mit **Extras→Watchliste laden** wieder laden.



Drive PLC Developer Studio

Ressourcen

8.9.3.4 Watchliste laden

Symbol: -	Menübefehl: Extras→Watchliste laden	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl lädt eine abgespeicherte Watchliste.

Nach Auswahl des Befehls öffnet sich das Dialogfeld *Öffnen*.

- Wählen Sie die gewünschte Datei mit dem Zusatz "*.wtc" aus und klicken Sie auf **OK**.

Im nun erscheinenden Dialogfeld können Sie der Watchliste einen neuen Namen geben.

- Das Eingabefeld ist vorbelegt mit den Dateinamen ohne Zusatz.

Mit **Extras→Watchliste speichern** können Sie eine Watchliste abspeichern.

8.9.3.5 Monitoring aktiv

Symbol: -	Menübefehl: Extras→Monitoring aktiv	Tastaturbefehl: -
-----------	--	-------------------

Dieser Befehl schaltet die Anzeige des Watch- und Rezepturverwalters im Online-Modus ein bzw. aus.

- Ist die Anzeige eingeschaltet, erscheint ein Haken vor dem Menüpunkt.

Um neue Variablen einzugeben oder einen Wert vorzubelegen, muß die Anzeige durch diesen Befehl ausgeschaltet werden. Nach Eingabe der Variablen kann die Anzeige durch erneutes Ausführen des Befehls wieder eingeschaltet werden.

8.9.3.6 Rezeptur schreiben

Symbol: -	Menübefehl: Extras→Rezeptur schreiben	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl können im Online-Modus des Watch- und Rezepturverwalters die vorbelegten Werte in die Variablen geschrieben werden.



Tip!

Es werden nur die Werte der Watchliste geladen, die im Watch- und Rezepturverwalter ausgewählt wurde!

8.9.3.7 Rezeptur lesen

Symbol: -	Menübefehl: Extras→Rezeptur lesen	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl wird im Online-Modus des Watch- und Rezepturverwalters die Vorbelegung der Variablen durch die aktuellen Werte ersetzt.

Beispiel:

```
PLC_PRG.Zaehler [:= <aktueller Wert>] = <aktueller Wert>
```



Tip!

Es werden nur die Werte der Watchliste geladen, die im Watch- und Rezepturverwalter ausgewählt wurden!

8.10 Zielsystemeinstellungen

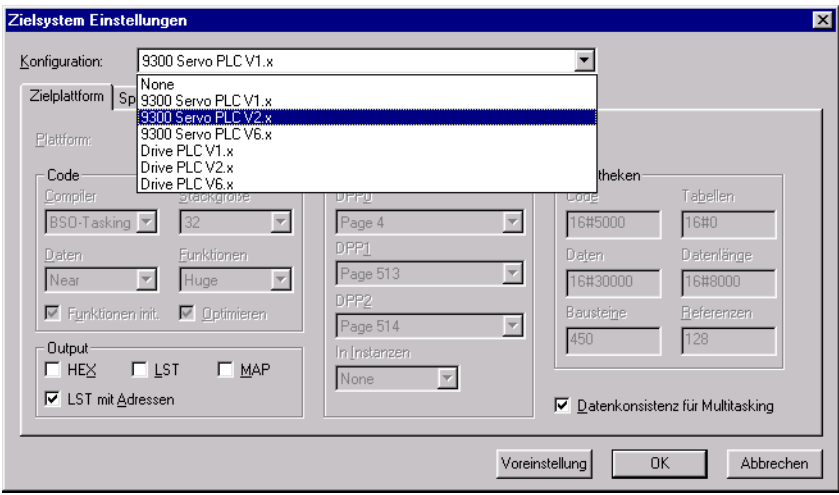


Die Zielsystemeinstellungen befinden sich als Objekt im Register Ressourcen.

- Festlegung mit welchem Zielsystem das Projekt laufen soll.
- Mit welchen Einstellungen das Projekt laufen soll.
- Nach dem Ausführen des Menübefehls **Projekt→Neu** muß ein Zielsystem gewählt werden.

Das Dialogfeld zur Zielsystemeinstellung öffnet sich automatisch beim Anlegen eines neuen Projekts. Das Dialogfeld kann außerdem in dem Register **Ressourcen→Zielsystemeinstellungen** aufgerufen werden.

Wählen Sie aus dem Kombinationsfeld **Konfiguration** ein Zielsystem aus.



Tip!

Mit der Schaltfläche **Voreinstellung** werden alle Markierungen auf diesem Dialogfeld in den Anfangszustand zurückgesetzt.



Drive PLC Developer Studio

Ressourcen

Zielplattform

In diesem Register ist nur das Gruppenfeld **Output** aktiv.

Mit dem Gruppenfeld **Output** können die vom DDS zu erstellenden Dateien konfiguriert werden.



Hinweis!

Die Dateitypen *.HEX, *.LST, *.MAP nur nach Aufforderung des Lenze Support aktivieren. Sie dienen ausschließlich der Diagnose.

Ist das Kontrollkästchen **HEX** aktiviert, wird eine *.HEX Datei mit dem HEX Dump erstellt.

Ist das Kontrollkästchen **LST** aktiviert, wird eine LIST Datei mit dem disassemblierten Programm erstellt.

Die Erstellung kann auch mit Adressen, **LST mit Adressen** erfolgen.

Ist das Kontrollkästchen **MAP** aktiviert, wird eine *.MAP Datei mit der Speicherbelegung erstellt.

Das Kontrollkästchen **Datenkonsistenz für Multitasking** hat eine besondere Eigenschaft.

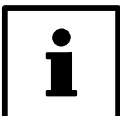
Fehler in der Datenkonsistenz

Einige Variablentypen können in einer Multitasking-Umgebung nicht mit einem Zugriff des Prozessors gelesen oder geschrieben werden. Es werden mehrere Zugriffe für die Abarbeitung benötigt.

Eine Dateninkonsistenz kann auftreten, wenn mehrere Task gleichzeitig auf eine Variable zugreifen.

Folgende Datentypen sind betroffen.

- DWORD, DINT, UDINT, REAL, TIME
- BOOL, wenn die BOOL-Variable eine absolute Adresse besitzt.
(bMyBool AT %MX0.0: BOOL;)

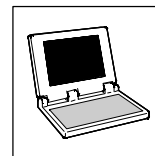


Hinweis!

Ist das Kontrollkästchen **Datenkonsistenz für Multitasking** aktiviert, kann während des Schreib- und Lesezugriffs eine höherpriori Task eine niederpriori Task nicht mehr unterbrechen. Der Schreibzugriff bzw Lesezugriff wird von der niederpriori Task abgeschlossen.

In den Folgenden Fällen muß der Anwender die Datenkonsistenz sicherstellen.

- Bitzugriffe der Form `byMyByte.1:=TRUE`



Arbeiten mit älteren Versionen



Hinweis!

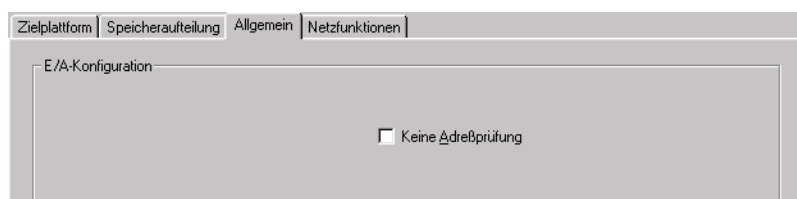
Folgende Einschränkungen oder Verhaltensweisen können beim Arbeiten mit älteren Programmversionen oder Projekten auftreten.

- Wird ein Projekt, das mit der Programmversion 2.1 oder 2.0 erstellt wurde, mit dem DDS 2.2 übersetzt, kann sich bei aktiviertem Kontrollkästchen **Datenkonsistenz für Multitasking** das Laufzeitverhalten so ändern, daß es zum Tasküberlauf kommt.
- Werden mit dem DDS 2.1 oder 2.0 Projekte geöffnet, die mit dem DDS 2.2 erstellt wurden, wird die **Datenkonsistenz für Multitasking** nicht unterstützt. Bei dem Übersetzen mit DDS 2.1 oder 2.0 kann es zu Fehlern in der Datenkonsistenz und einem geänderten Laufzeitverhalten kommen.

Speicheraufteilung

Das Register ist nicht editierbar.

Allgemein

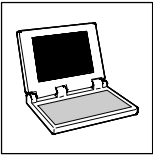


Standardmäßig ist die Adreßprüfung aktiviert, deshalb können im Projekt nur absolute Adressen, die in der Steuerungskonfiguration eingegeben sind, verwendet werden.

Mit dem Kontrollkästchen **Keine Adreßprüfung** wird die Adreßprüfung deaktiviert. Beim Übersetzen des Projektes werden die IEC Adressen nicht geprüft.

Netzfunktionen

Das Register ist nicht aktiv.



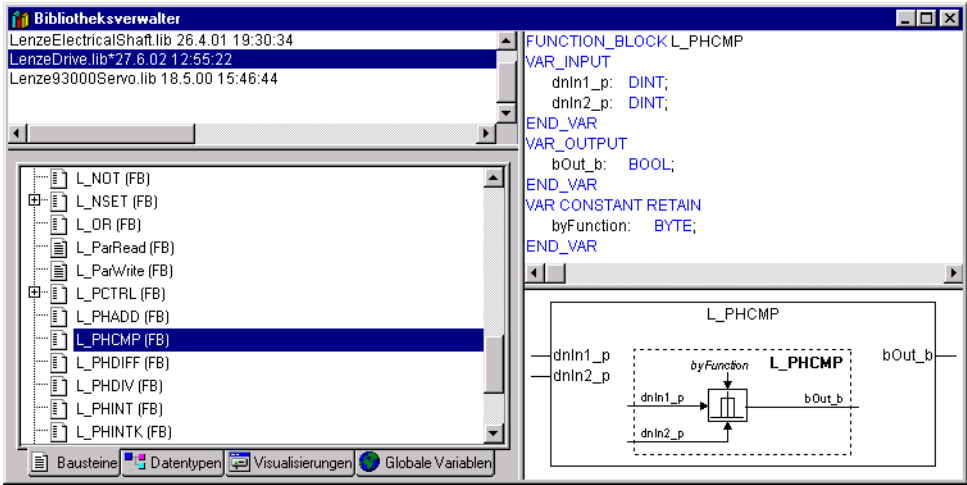
8.11 Bibliotheksverwaltung

Für die Verwaltung von Bibliotheken verwenden Sie im DDS den Bibliotheksverwalter, den Sie mit dem Befehl **Fenster→Bibliotheksverwaltung** aufrufen.

- Die Information über die eingebundenen Bibliotheken wird mit dem Projekt gespeichert.
- Der Bibliotheksverwalter zeigt alle Bibliotheken an, die an das aktuelle Projekt angeschlossen sind.
- Bausteine, Datentypen, Visualisierungen und globale Variablen der Bibliotheken können wie selbstdefinierte Bausteine, Datentypen Visualisierungen und globale Variablen verwendet werden.

8.11.1 Fenster des Bibliotheksverwalters

Das Fenster des Bibliotheksverwalters ist durch Bildschirmteiler in drei bzw. vier Bereiche aufgeteilt.



Bibliotheken

Im linken oberen Bereich sind die an das Projekt gebundenen Bibliotheken aufgelistet.

Bausteine/Datentypen/Visualisierungen/globale Variablen

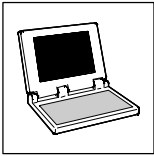
Im linken unteren Bereich werden je nach gewählter Registerkarte die Bausteine, Datentypen, Visualisierungen oder globale Variablen der im oberen Bereich gewählten Bibliothek aufgelistet.

- Ordner werden per Doppelklick oder Betätigen der **<Eingabetaste>** auf- und zugeklappt.
- 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. Anstelle des Pluszeichens wird nun ein Minuszeichen angezeigt.
	Mit einem Klick auf das Minuszeichen kann der Ordner wieder zugeklappt werden.

Deklaration Baustein/Grafische Darstellung des Bausteins

Wird ein Baustein durch Mausklick oder Auswahl per Pfeiltasten selektiert, erscheint im rechten oberen Bereich des Bibliotheksverwalters die Deklaration des Bausteins und im rechten unteren Bereich die grafische Darstellung als Black Box mit Ein- und Ausgängen.



Deklaration Datentypen/globale Variablen

Bei Datentypen und globalen Variablen wird im rechten Bereich des Bibliotheksverwalters die Deklaration angezeigt.

Deklaration Visualisierung

Bei Visualisierungen werden im rechten Bereich die verwendeten Platzhalter und die Anzeige der Visualisierung angezeigt.

8.11.1.1 Bibliothek einfügen

Symbol: -	Menübefehl: Einfügen→weitere Bibliothek	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl können Sie eine weitere Bibliothek an Ihr Projekt binden.

Nach Auswahl des Befehls öffnet sich das Dialogfeld *Öffnen*.

- Wählen Sie die gewünschte Datei mit dem Zusatz **.lib* aus und klicken Sie auf **OK**.

Die Bibliothek wird nun im Bibliotheksverwalter mit aufgelistet und Sie können die Objekte der Bibliothek wie selbstdefinierte Objekte verwenden.



Tip!

Dieser Befehl ist auch über das Kontextmenü (rechte Maustaste) im linken oberen Bereich des Bibliotheksverwalters verfügbar.

8.11.1.2 Eigenschaften

Symbol: -	Menübefehl: Extras→Eigenschaften	Tastaturbefehl: <Alt>+<Eingabe>
-----------	---	--

Wenn die Bibliotheksverwalter geöffnet ist, können Sie sich mit diesem Befehl die Eigenschaften der markierten Bibliothek anzeigen lassen.



Drive PLC Developer Studio

Ressourcen

8.11.2 Mitgelieferte Bibliotheken

Das DDS wird u. a. mit folgenden Bibliotheken ausgeliefert:



Tip!

Eine detaillierte Beschreibung der Funktionsbausteine finden Sie im Handbuch zur entsprechenden Funktionsbibliothek sowie in der Online-Hilfe.

IEC 61131-3 Standard Bibliothek "Standard.lib"

Die Bibliothek **Standard.lib** 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).

Bibliothek "LenzeDrive.lib"

Die Bibliothek **LenzeDrive.lib** beinhaltet Funktionsblöcke zur Umsetzung von Standard-Antriebsaufgaben für 9300 Servo PLC und Drive PLC.

- Eine mehrfache Instanzierung dieser Funktionsblöcke nach IEC61131-3 ist möglich.

Bibliothek "LenzeElectricalShaft.lib"

Die Bibliothek **LenzeElectricalShaft.lib** beinhaltet spezielle Funktionsblöcke für die "Elektrische Welle".

Bibliothek "Lenze9300Servo.lib"

Die Bibliothek **Lenze9300Servo.lib** beinhaltet spezielle Funktionsblöcke für die 9300 Servo PLC.

Bibliothek "lcsfc.lib" für IEC-Schritte im AS

Die Bibliothek **lcsfc.lib** beinhaltet IEC-Schritte, die Sie im Ablaufplan verwenden können.

8.11.3 Benutzerdefinierte Bibliotheken

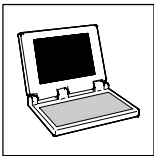
Sie können ein fertiges und fehlerfrei übersetztes Projekt mit dem Befehl **Datei→Speichern unter** als Interne Bibliothek (*.lib), Externe Bibliothek (*.lib) oder als Projekt (*.pro) ablegen.

- Das Projekt selbst bleibt unverändert.
- Nach dem Speichern steht es wie die Standardbibliothek unter dem eingegebenen Namen zur Verfügung.



Hinweis!

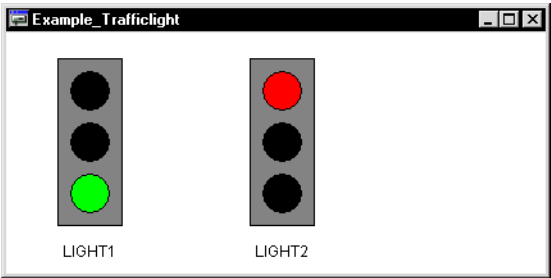
Beim Speichern der Bibliothek wird der Baustein PLC-PRG aus dem Projekt entfernt.



9 Visualisierung

Die Visualisierung befindet sich in der Registerkarte **Visualisierung** im *Object Organizer*.

Mit Hilfe der Visualisierung können Sie im Offline-Modus geometrische Elemente zeichnen, die dann im Online-Modus ihre Form oder Farbe in Abhängigkeit von bestimmten Variablenwerten ändern.



Tip!

Mit dem Menübefehl **Projekt→Objekt einfügen** erstellen Sie ein neues Visualisierungsobjekt.

Mit der Hilfe von Platzhaltern kann ein Visualisierungsobjekt mehrfach verwendet werden. Ein Visualisierungsobjekt kann in andere Visualisierungsobjekte eingefügt werden. Durch das Ersetzen der Platzhalter wird das Visualisierungsobjekt mit anderen Konfigurationen versehen. In andere Visualisierungen eingefügte Visualisierungsobjekte werden Referenz genannt. Es gibt zwei Möglichkeiten ein Visualisierungsobjekt zu konfigurieren.

Eine Referenz reagiert, je nach Konfiguration, im Online-Modus auf Aktionen unterschiedlich.

- Die einzelnen Elemente der Referenz reagieren wie die ursprüngliche Visualisierung.
- Die einzelnen Elemente der Referenz bleiben unberücksichtigt. Die Referenz reagiert als Gesamtobjekt auf Eingaben.

Anwendungsbeispiele:

- In einem Balkendiagramm das Verhalten einer Variablen während des Wachstums darstellen lassen.
- Eingaben für das Programm über Maus und Tastatur vornehmen.

Steht eine entsprechende Übersetzungsdatei zur Verfügung, können Texte innerhalb der Visualisierung Online in einer anderen Sprache angezeigt werden.

Projekt→In andere Sprache übersetzen (📖 6-12)



Drive PLC Developer Studio

Visualisierung

9.1 Visualisierungselemente einfügen

Einfügemodus

Wenn die Visualisierungsebene aktiv ist, sind unter dem Menübefehl **Einfügen** die verschiedenen Visualisierungsobjekte aufgelistet.

- Der Mauszeiger im Editierfenster ist mit dem entsprechenden Symbol für das einzufügende Element gekennzeichnet.
- In der Statusleiste wird der Name des einzufügenden Elements schwarz angezeigt.
- Nachdem ein Element eingefügt wurde, wird automatisch in den Selektionsmodus gewechselt.

Selektionsmodus

Im Selektionsmodus können Sie ein oder mehrere Elemente zum Bearbeiten selektieren.

- Möchten Sie ein weiteres Element einfügen, wechseln Sie dazu wieder in den Einfügemodus, indem Sie bei gedrückter **<Strg>**-Taste die rechte Maustaste betätigen oder erneut den entsprechenden Befehl zum Einfügen wählen.



Tip!

Sie können schnell zwischen Einfüge- und Selektionsmodus wechseln, indem Sie bei gedrückter **<Strg>**-Taste die rechte Maustaste betätigen.

9.1.1 Befehle im Menü “Einfügen”

9.1.1.1 Rechteck

Symbol: 	Menübefehl: Einfügen→Rechteck	Tastaturbefehl: -
---	--------------------------------------	-------------------

Dieser Befehl fügt ein Rechteck als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

- Ziehen Sie im Editierfenster das Element mit gedrückter linker Maustaste auf, bis es die gewünschte Größe hat.

9.1.1.2 Abgerundetes Rechteck

Symbol: 	Menübefehl: Einfügen→Abgerundetes Rechteck	Tastaturbefehl: -
---	---	-------------------

Dieser Befehl fügt ein Rechteck mit abgerundeten Ecken als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

- Ziehen Sie im Editierfenster das Element mit gedrückter linker Maustaste auf, bis es die gewünschte Größe hat.



9.1.1.3 Ellipse

Symbol:	Menübefehl: Einfügen→Ellipse	Tastaturbefehl: -
---------	-------------------------------------	-------------------

Dieser Befehl fügt eine Ellipse als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

- Ziehen Sie im Editierfenster das Element mit gedrückter linker Maustaste auf, bis es die gewünschte Größe hat.

9.1.1.4 Linienzug

Symbol:	Menübefehl: Einfügen→Linienzug	Tastaturbefehl: -
---------	---------------------------------------	-------------------

Dieser Befehl fügt einen Linienzug als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

1. Klicken Sie im Editierfenster an den gewünschten Anfangspunkt des Linienzuges.
2. Fügen Sie dem Linienzug durch erneutes Klicken weitere Eckpunkte hinzu.
3. Beenden Sie durch einen Doppelklick das Zeichnen des Linienzuges.

9.1.1.5 Kurve

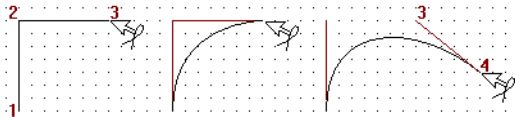
Symbol:	Menübefehl: Einfügen→Kurve	Tastaturbefehl: -
---------	-----------------------------------	-------------------

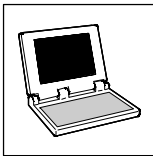
Dieser Befehl fügt eine Kurve (Beziers-Kurve) als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

1. Klicken Sie im Editierfenster an den gewünschten Anfangspunkt der Kurve.
 2. Klicken Sie zwei weitere Punkte an, die die Tangenten der Kurve definieren.
- Es wird ein Kurvenbogen gezeichnet, deren Endpunkt Sie durch Verschieben mit der Maus noch verändern können.
3. Beenden Sie die Kurve durch einen Doppelklick oder erweitern Sie mit einfachen Mausklicks die Kurve um weitere Bögen.

Beispiel





Drive PLC Developer Studio

Visualisierung

9.1.1.6 Polygon

Symbol: 	Menübefehl: Einfügen→Polygon	Tastaturbefehl: -
---	-------------------------------------	-------------------

Dieser Befehl fügt ein Polygon als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

1. Klicken Sie im Editierfenster an den gewünschten Anfangspunkt des Polygons.
2. Fügen Sie dem Polygon durch erneutes Klicken weitere Eckpunkte hinzu.
3. Beenden Sie durch einen Doppelklick das Zeichnen des Polygons.

9.1.1.7 Bitmap

Symbol: 	Menübefehl: Einfügen→Bitmap	Tastaturbefehl: -
---	------------------------------------	-------------------

Dieser Befehl fügt eine Grafik, die als Bitmap (.bmp) oder Tagged Image File Format (.tif) vorhanden sein muß, als Element in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet.

1. Ziehen Sie im Editierfenster mit gedrückter linker Maustaste den Bereich auf, in dem die vorhandene Grafik eingefügt werden soll.
2. Wählen Sie im Dialogfeld *Öffnen* die einzufügende Grafik-Datei aus.

9.1.1.8 Visualisierung

Symbol: 	Menübefehl: Einfügen→Visualisierung	Tastaturbefehl: -
---	--	-------------------

Dieser Befehl fügt eine bestehende Visualisierung in Ihre aktuelle Visualisierung ein.

Nach Auswahl des Befehls ist der Mauszeiger im Editierfenster mit dem entsprechenden Symbol gekennzeichnet. Eine eingefügte Visualisierung wird Referenz bezeichnet.

1. Ziehen Sie im Editierfenster mit gedrückter linker Maustaste den Bereich auf, in dem die Visualisierung eingefügt werden soll.
2. Wählen Sie im Dialogfeld *Visualisierung auswählen* die einzufügende Visualisierung aus.

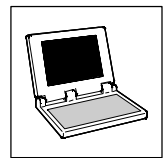
9.1.1.9 Schaltfläche

Symbol: 	Menübefehl: Einfügen→Schaltfläche	Tastaturbefehl: -
---	--	-------------------

Dieser Befehl fügt eine leere Schaltfläche in Ihre aktuelle Visualisierung ein.

Wird für eine Schaltfläche eine Toggle-Variable konfiguriert, wird diese durch die Schaltfläche visualisiert.

1. Ziehen Sie das Element mit gedrückter linker Maustaste auf die gewünschte Größe.
2. Die erzeugte Schaltfläche können Sie nachträglich formatieren. Wählen Sie dazu den Menübefehl **Extras→Konfigurieren**.



9.2 Visualisierungselemente bearbeiten

9.2.1 Informationen in der Statusleiste

Die Statusleiste enthält folgende Informationen zur Visualisierung:

X-/Y-Position

Die aktuelle X- und Y-Position des Mauszeigers wird in Pixel relativ zur oberen linken Ecke des Bildes angezeigt.

Elementnummer

Befindet sich der Mauszeiger auf einem Element oder wird ein Element bearbeitet, wird dessen Elementnummer angezeigt.

Einfügemodus

Ist der Einfügemodus aktiv, wird der Name des einzufügenden Elementes angezeigt.

9.2.2 Ausdrücke

Werden in der Konfiguration eines Visualisierungsobjektes SPS-Variablen wirksam, sind folgende Eingaben möglich.

- Variablenname mit Eingabehilfe
- Zusammengesetzte Ausdrücke aus
 - Komponentenzugriffen (z. B. STRUCT)
 - Feldzugriffen (ARRAY-Zugriff) mit konstantem Index
 - Variablen und direkten Adressen
- Operatoren und Adressen, die zusammen mit den genannten Ausdrücken kombiniert werden können.

Beispiele für zulässige Ausdrücke

`x+y`

`100*PLC_PRG.a`

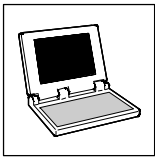
`TRUE`

`NOT PLC_PRG.b`

`q*sin(x+100)+cos(y+100)`

Folgende Eingaben sind nicht möglich.

- Funktionsaufrufe
Unzulässige Ausdrücke führen beim Einloggen zu einer Fehlermeldung (ungültiger Watchausdruck)



Drive PLC Developer Studio

Visualisierung

Beispiele für unzulässige Ausdrücke

`fun(88)`

`a:=9`

`RETURN`

Für globale Variablen sind zwei Schreibweisen möglich.

`.globvar` (Innerhalb eines zusammengesetzten Ausdrucks nicht möglich)

`globvar`

9.2.3

Visualisierungselemente selektieren



Tip!

Die Befehle zum Bearbeiten von Elementen beziehen sich immer auf das bzw. die aktuell selektierten Elemente.

Ein einzelnes Element selektieren

- Klicken Sie auf das Element, das Sie selektieren möchten.
- 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>**-Taste bei gleichzeitig gedrückter **<Umschalt>**-Taste anwenden, springen Sie in der Elementliste entsprechend rückwärts.

Mehrere Elemente selektieren

- Halten Sie die **<Umschalt>**-Taste gedrückt und klicken Sie nacheinander auf die entsprechenden Elemente.

oder

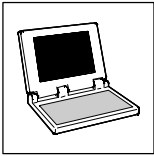
- Ziehen Sie bei gedrückter linker Maustaste ein Fenster über die zu markierenden Elemente auf.

Alle Elemente selektieren

- Verwenden Sie den Befehl **Extras→Alles Markieren**, um alle Elemente der Visualisierung zu selektieren.

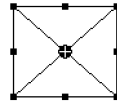
Aus der Elementliste bearbeiten

- Verwenden Sie den Befehl **Extras→Elementliste** um die Elementliste zu öffnen.
- Hier sind die Art und der Ort aller erzeugten Elemente aufgeführt. Mit der Schaltfläche **Bearbeiten** können Sie ein ausgewähltes Element bearbeiten.



9.2.4 Visualisierungselemente in Größe und Form verändern

Ist ein Element selektiert, werden dessen Ankerpunkte in Form schwarzer Quadrate sowie der Drehpunkt angezeigt:



- Sie können die Größe des selektierten Elements verändern, indem Sie den entsprechenden Ankerpunkt mit gedrückter linker Maustaste verschieben.
- Sie können den, im Visualisierungselement zentral liegenden, Drehpunkt mit gedrückter linker Maustaste verschieben. Um diesen Punkt rotiert dann das Element bei eingestellter Bewegung/Winkel.

Polygon

Eckpunkt verschieben

- Mit gedrückter linker Maustaste können Sie jeden Eckpunkt (Ankerpunkt) eines Polygons verschieben.

Eckpunkt einfügen

- Halten Sie die **<Strg>**-Taste gedrückt und ziehen Sie mit gedrückter linker Maustaste den neuen Eckpunkt aus einen bereits bestehenden Eckpunkt an die gewünschte Position.

Eckpunkt entfernen

- Halten Sie die **<Umschalt>+<Strg>**-Taste gedrückt und klicken Sie auf den Eckpunkt, der entfernt werden soll.

9.2.5 Visualisierungselemente verschieben

Ein oder mehrere ausgewählte Elemente können Sie mit gedrückter linker Maustaste oder den Pfeiltasten verschieben.

9.2.6 Visualisierungselemente kopieren, ausschneiden, einfügen

Ein oder mehrere selektierte Elemente können Sie mit Hilfe der Zwischenablage kopieren, ausschneiden bzw. einfügen.

Kopieren mit der **<Strg>**-Taste

Desweiteren können auf folgende Weise Elemente kopieren:

- Selektieren Sie die zu kopierenden Elemente.
- Drücken Sie gleichzeitig die **<Strg>** und **<C>** Taste.
- Drücken Sie gleichzeitig die **<Strg>** und **<V>** Taste. Die Kopie wird etwas versetzt zum Original abgelegt.
- Mit gedrückter linker Maustaste können Sie nun die Kopie an die gewünschte Position verschieben.

9.2.7 Visualisierungselemente löschen

Ein oder mehrere selektierte Elemente können Sie mit den Befehlen **Bearbeiten→Löschen** (**<Entf>**-Taste) bzw. **Bearbeiten→Ausscheiden** löschen.

- Wenn Sie **Bearbeiten→Ausscheiden** verwenden, stehen Ihnen die selektierten Elemente noch in der Zwischenablage zur Verfügung.



Drive PLC Developer Studio

Visualisierung

9.2.8 Befehle im Menü “Extras”

9.2.8.1 Elementliste

Symbol: -	Menübefehl: Extras→Elementliste	Tastaturbefehl: -
-----------	--	-------------------

Mit diesem Befehl öffnen Sie das Dialogfenster *Elementliste*, in dem alle Visualisierungselemente mit ihrer Nummer, Art und Position aufgelistet werden.

- Die Position bezieht sich auf die X- und Y-Position der linken, oberen und rechten, unteren Ecke des Elements.
- Wenn Sie ein 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.

Schaltflächen

- Mit **Nach vorne** legen Sie selektierte Visualisierungselemente in den Vordergrund.
- Mit **Nach hinten** legen Sie selektierte Visualisierungselemente in den Hintergrund.
- Mit **Löschen** werden die selektierten Visualisierungselemente entfernt.
- Mit **Rückgängig** und **Wiederherstellen** können Sie vorgenommene Aktionen rückgängig machen und wiederherstellen, analog zu den Befehlen **Bearbeiten→Rückgängig** und **Bearbeiten→Wiederherstellen**. Die durchgeführten Änderungen können Sie im Editorfenster beobachten.
- Mit **OK** bestätigen Sie Ihre Änderungen und schließen das Dialogfenster.

9.2.8.2 Nach vorn bringen

Symbol: -	Menübefehl: Extras→Nach vorn bringen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl bringen Sie selektierte Visualisierungselemente in den Vordergrund.

9.2.8.3 Nach hinten legen

Symbol: -	Menübefehl: Extras→Nach hinten legen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl bringen Sie selektierte Visualisierungselemente in den Hintergrund.

9.2.8.4 Ausrichten

Symbol: -	Menübefehl: Extras→Ausrichten	Tastaturbefehl: -
-----------	--------------------------------------	-------------------

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.
Rechts / Oben / Unten	Entsprechende Ausrichtung analog zu Links .
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.

9.2.8.5 Alles Markieren

Symbol: -	Menübefehl: Extras→Alles Markieren	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl selektieren Sie alle Visualisierungselemente des aktuellen Visualisierungsbjeks.



9.2.8.6 Platzhalterliste

Symbol:	Menübefehl: Extras→Platzhalterliste	Tastaturbefehl: -
---------	--	-------------------

An jeder Stelle des Konfigurationsdialoges **Extras→Konfigurieren**, an dem Variablen oder Text eingegeben werden kann, besteht die Möglichkeit für die Variablen oder des Textes auch ein Platzhalter einzugeben.

Das ist sinnvoll wenn das Visualisierungsobjekt in anderer Visualisierungsobjekte als Referenz eingebunden werden soll. Mit dem dem Aufruf der Referenz kann der Platzhalter entsprechend durch eine Variable oder Text ersetzt werden.

Der Dialog zeigt die im aktuellen Visualisierungsobjekt verwendeten Platzhalter an und ermöglicht eine Vordefinition. Das Visualisierungsobjekt muß den Eingabefokus haben.

Jeder Platzhalter kann beim Konfigurieren einer Referenz mit entsprechenden Werten ersetzt werden.

Platzhalter

In dieser Spalte werden die Platzhalter aufgelistet, die für die Visualisierung vorgesehen werden.

Elementnummern

Zeigt die Elementnummer an für das der Platzhalter erstellt wurde. Vergleichen Sie dazu die Nummern aus der Elementliste.

Ersetzungen

Hier können Sie eine Auswahl von Strings eingeben für die der Platzhalter steht. Der Platzhalter ist die Referenz auf den String. Die Strings müssen durch Komma getrennt werden. Ist kein Eintrag vorhanden, kann der Platzhalter bei der Konfiguration der Referenz durch beliebigen Text ersetzt werden.

Mögliche Ersetzungen

Platzhalter	Elementnummer	Ersetzungen
text	0	box_a, box_b
farbwechsel	0	pic_prg.var, var_farbwechsel1, pic_prg.var_farbwechsel2
text2	1	a, b, c, d
togglevar	0	



Drive PLC Developer Studio

Visualisierung

9.3 Visualisierungselemente konfigurieren

9.3.1 Befehle im Menü “Extras”

9.3.1.1 Einstellungen

Symbol: -	Menübefehl: Extras→Einstellungen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl öffnen Sie das Dialogfenster *Einstellungen Visualisierung*, in dem Sie folgende Einstellungen vornehmen können:

Kategorie Darstellung

- Im Textfeld **Zoom** können Sie einen Darstellungsfaktor zwischen 10 und 500 % angeben, um die Anzeige der Visualisierung zu vergrößern bzw. zu verkleinern.
- Ist das Kontrollkästchen **Elementnummern** aktiviert, wird im Offline-Modus zu jedem Element die zugehörige Elementnummer angezeigt.

Kategorie Rahmen

- Ist das Kontrollkästchen **Auto-Scrolling** aktiviert, wird beim Zeichnen oder Verschieben eines Visualisierungselements der sichtbare Bereich des Visualisierungsfensters automatisch mit verschoben, wenn der Fensterrahmen erreicht ist.
- Ist das Kontrollkästchen **Online automatisch anpassen** aktiviert, wird im Online-Modus die Visualisierung an die Fenstergröße angepaßt, so daß alle Elemente sichtbar sind.
- Ist das Kontrollkästchen **Hintergrund Bitmap einbeziehen** aktiviert, wird bei aktiviertem Kontrollkästchen **Online automatisch anpassen** für das Anpassen der Visualisierung an das Fenster die Größe des Bitmap in die Berechnung mit einbezogen, ansonsten werden nur die Elemente berücksichtigt.

Kategorie Raster

- Ist das Kontrollkästchen **Sichtbar** aktiviert, werden die Rasterpunkte im Offline-Modus angezeigt.
- Ist das Kontrollkästchen **Aktiv** aktiviert, können Elemente beim Zeichnen und Verschieben nur auf die Rasterpunkte gelegt werden.
- Im Textfeld **Größe** legen Sie den Abstand der Rasterpunkte fest.



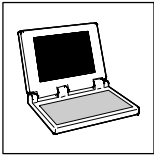
Tip!

Der Abstand der sichtbaren Punkte beträgt mindestens 10 Punkte, auch wenn die Angabe im Textfeld **Größe** kleiner ist.

Bei einer Rasterweite kleiner 10 Punkte erscheinen die Rasterpunkte dennoch mit 10 Punkt Abstand.

Kategorie Sprache

- Ist das Kontrollkästchen **Sprachdatei** aktiviert, kann eine spezielle Sprachdatei *.vis erstellt werden
- Mit der Schaltfläche ... öffnet sich ein Dialog zum Speichern der Sprachdatei.
- Mit dem Kombinationsfeld **Sprache** können Sie eine Sprache auswählen.
- Die Schaltfläche **Speichern** speichert die Sprachdatei an der ausgewählten Stelle.
- Mit der Schaltfläche **OK** werden die Einstellungen auf dem Dialogfeld gespeichert.



Sprachdatei anlegen

- Markieren Sie das Kontrollkästchen **Sprachdatei**. Das Gruppenfeld **Sprache** ist aktiv.
- Geben Sie im Kombinationsfeld **Sprache** z. B. "en" ein. (für Englisch)
- Geben Sie ein Pfad ein oder suchen Sie mit ... das Verzeichnis, in der die Sprachdatei abgelegt werden soll. Am Ende muß der Dateiname, z. B. English.vis stehen.
- Drücken Sie **Speichern**, wenn das Dialogfeld vollständig ausgefüllt ist. Im eingegebenen Zielpfad wird, in dem Beispiel, die Datei English.vis erzeugt.

Die angelegte Sprachdatei kann mit einem Texteditor geöffnet werden.

9.3.1.2 Hintergrund Bitmap auswählen

Symbol: -	Menübefehl: Extras→Hintergrund Bitmap auswählen	Tastaturbefehl: -
-----------	---	-------------------

Dieser Befehl lädt eine Bitmap als Hintergrund für die Visualisierung.

Nachdem der Befehl gewählt wurde, erscheint das Dialogfeld *Datei öffnen*.

- Wählen Sie die gewünschte Datei mit dem Zusatz "*.bmp" aus und klicken Sie auf **OK**.

Die ausgewählte Bitmap erscheint im Hintergrund Ihrer Visualisierung.

9.3.1.3 Hintergrund Bitmap löschen

Symbol: -	Menübefehl: Extras→Hintergrund Bitmap löschen	Tastaturbefehl: -
-----------	---	-------------------

Mit diesem Befehl löschen Sie die Bitmap im Hintergrund der aktuellen Visualisierung.

9.3.1.4 Konfigurieren

Symbol: -	Menübefehl: Extras→Konfigurieren	Tastaturbefehl: -
-----------	----------------------------------	-------------------

Mit diesem Befehl öffnen Sie das Dialogfeld *Element konfigurieren* zum Konfigurieren des selektierten Visualisierungselements.



Tip!

Sie können das Dialogfeld auch öffnen, indem Sie auf ein Element doppelklicken.

Konfigurieren→Form

verfügbar für: Rechteck, Abgerundetes Rechteck, Ellipse

Über **Extras→Konfigurieren**, Kategorie *Form* können Sie dem selektierten Element die Form *Rechteck*, *Abgerundetes Rechteck* oder *Ellipse* zuweisen.

- Die bereits festgelegte Größe des Elements bleibt erhalten.

Konfigurieren→Text

verfügbar für: alle

Über **Extras→Konfigurieren**, Kategorie *Text* können Sie dem selektierten Element einen Text zuweisen. Er wird direkt eingegeben oder durch einen Platzhalter vertreten.

Inhalt

Geben Sie in das Eingabefeld **Inhalt** den Text ein.

<Strg>+<Return> erzeugt Zeilenumbrüche

<Strg>+<Tabulator> erzeugt Tabstops

- Wenn Sie "%s" eingeben, wird dieser Text im Online-Modus durch den Wert der Variablen aus dem Textfeld **Textausgabe** der Kategorie *Variablen* ersetzt.



Drive PLC Developer Studio

Visualisierung



Achtung!

Wenn eine Übersetzungsdatei im Online-Modus ein Umschalten in eine andere Landessprache ermöglichen soll, muß der bestimmte Text in #Text# eingeschlossen sein.

#Pumpe 1# oder #Pumpe# 1

Der zweite Fall erspart beim mehrmaligen Auftreten des Textes Pumpe (Pumpe1, Pumpe2 etc.) ein mehrmaliges Auftreten in der Übersetzung.

Horizontal/Vertikal

Durch Auswahl der entsprechenden Kontrollkästchen legen Sie die Ausrichtung des konfigurierten Textes im Element fest.

Schrift

Durch Klicken auf **Schrift** öffnen Sie das Dialogfeld *Schriftart* zum Auswählen der Schriftart für den Text.

Standard-Schrift

Durch Klicken auf **Standard-Schrift** wird die Schriftart verwendet, die unter **Projekt→Optionen** gewählt ist.

- Wird die Schriftart unter **Projekt→Optionen** verändert, wirkt sich diese Änderung auf alle Elemente aus, denen nicht explizit eine andere Schriftart über die Schaltfläche **Schrift** zugewiesen wurde.

Konfigurieren→Linienstärke

verfügbar für: alle

Über **Extras→Konfigurieren**, Kategorie *Linienstärke* können die Linienstärken der markierten Objekte geändert werden.

Konfigurieren→Farben

verfügbar für: Rechteck, Abgerundetes Rechteck, Ellipse

Über **Extras→Konfigurieren**, Kategorie *Farben* können Sie dem selektierten Element Grundfarben und Alarmfarben für die Innenfläche sowie dem Rahmen zuweisen.

- Wenn Sie in der Kategorie *Variablen*, im Feld *Farbwechsel* eine boolsche Variable eingegeben haben, wird das Element in der eingestellten Farbe dargestellt, solange die Variable **FALSE** ist. Ist die Variable **TRUE**, wird das Element in der Alarmfarbe dargestellt.



Tip!

Die Farbwechsel-Funktion ist erst aktiv, wenn sich die Steuerung im Online-Modus bzw. in der Simulation befindet!

- Durch Klicken auf **Innen** bzw. **Rahmen** öffnen Sie das Dialogfenster *Farben* zum Auswählen der entsprechenden Farbe.

Konfigurieren→Bewegung absolut

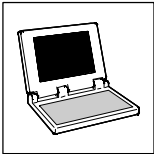
verfügbar für: alle

Über **Extras→Konfigurieren**, Kategorie **Bewegung absolut** können Sie für das selektierte Element eine Bewegung in Abhängigkeit von Variablenwerten festlegen.



Tip!

Für die Eingabe der Variablen können Sie die Eingabehilfe <F2> verwenden!



X-Offset/Y-Offset

In den Eingabefeldern **X-Offset/Y-Offset** können Sie Variablen eintragen, deren Werte eine Verschiebung des Elements in X- bzw. Y-Richtung bewirken.

Skalierung

In das Eingabefeld **Skalierung** können Sie eine Variable eingeben, deren Wert die Größe des Elements festlegt.

Eingabe und Auswirkungen

	Vergrößerung
1000	1:1 keine Änderung
100	x 0,1
10000	x 10

Winkel

In das Eingabefeld **Winkel** können Sie eine Variable eingeben, deren Wert eine Drehung des Elements bewirkt (positiver Wert = mathematisch positiv = Uhrzeigersinn).

- Der Wert wird in Grad ausgewertet.
- Der Drehpunkt eines selektierten Elements kann mit gedrückter linker Maustaste verschoben werden.

Eine Variable im Feld **Winkel** bewegt eine Drehung des aktuellen Elements. Die Drehung wird mit positiven Werten hervorgerufen, wobei die Drehung im Uhrzeigersinn erfolgt.

Konfigurieren→Bewegung relativ

verfügbar für: alle außer Polygon

Über **Extras → Konfigurieren**, Kategorie *Bewegung relativ* können Sie den Kanten des selektierten Elements Variablen zuordnen. In Abhängigkeit der Variablenwerte bewegen sich dann die entsprechenden Elementkanten.



Tip!

Für die Eingabe der Variablen können Sie die Eingabehilfe <F2> verwenden!

Für jede Seite des Elements steht ein Eingabefeld zur Verfügung.

- 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!

Konfigurieren→Variablen

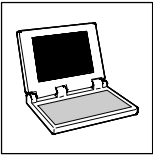
verfügbar für: alle

Über **Extras→Konfigurieren**, Kategorie *Variablen* können Sie Variablen festlegen, die den Zustand des selektierten Elements beschreiben sollen.



Tip!

Für die Eingabe der Variablen können Sie die Eingabehilfe <F2> verwenden!



Drive PLC Developer Studio

Visualisierung

Unsichtbar

Wenn die Variable im Eingabefeld **Unsichtbar** den Wert **FALSE** hat, dann ist das Visualisierungselement sichtbar. Besitzt die Variable den Wert **TRUE**, dann ist das Element unsichtbar.

Farbwechsel

Wenn die Variable im Eingabefeld **Farbwechsel** 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

In das Eingabefeld **Textausgabe** können Sie eine Variable eingeben, deren Wert ausgegeben wird, sofern Sie in das Feld **Inhalt** der Kategorie *Text* zusätzlich zum Text "%s" eingegeben haben.

9.3.2 Formatierte Textausgabe

Die folgenden Tabellen wird die formatierte Textausgabe allgemein dargestellt.

Format für Char

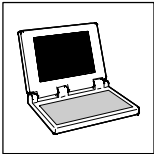
Element	Bedeutung
%	Die formatierte Textausgabe wird mit % eingeleitet. Soll das Zeichen aus bestimmten Gründen mit ausgegeben werden, muß die Einleitung mit %% beginnen.
-	Die Ausgabe ist linksbündig.
b	Minimalbreite des Ausgabefeldes
c	bei Char Variablen

Format für String

Element	Bedeutung
%	Die formatierte Textausgabe wird mit % eingeleitet. Soll das Zeichen aus bestimmten Gründen mit ausgegeben werden, muß die Einleitung mit %% beginnen.
-	Die Ausgabe ist linksbündig.
b	Minimalbreite des Ausgabefeldes
.n	Breite des vom String beschriebenen Teilfeldes
s	bei Char Arrays

Format für ganze Zahlen

Element	Bedeutung
%	Die formatierte Textausgabe wird mit % eingeleitet. Soll das Zeichen aus bestimmten Gründen mit ausgegeben werden, muß die Einleitung mit %% beginnen.
-	Die Ausgabe ist linksbündig.
+	positives Vorzeichen
b	Minimalbreite des Ausgabefeldes
.n	Mindestanzahl der ausgegebenen Ziffern
Von den folgenden Zeichen steht nur eins am Ende der Formatierung	
d	bei Int Variablen
ld	bei long Int Variablen



Format für vorzeichenlose ganze Zahlen

Element	Bedeutung
%	Die formatierte Textausgabe wird mit % eingeleitet. Soll das Zeichen aus bestimmten Gründen mit ausgegeben werden, muß die Einleitung mit %% beginnen.
-	Die Ausgabe ist linksbündig.
#	führende Null bei hexadezimaler und oktaler Darstellung
b	Minimalbreite des Ausgabefeldes
.n	Mindestanzahl der ausgegebenen Ziffern
Von den folgenden Zeichen steht nur eins am Ende der Formatierung	
d	Ziffer in Dezimal 0-9
x	Ziffer in Hexadezimal 0-9 und a-f
X	Ziffer in Hexadezimal 0-9 und A-F
o	oktal 0-7 bei Variablen ohne Vorzeichen

Format für Gleitkommazahlen

Element	Bedeutung
%	Die formatierte Textausgabe wird mit % eingeleitet. Soll das Zeichen aus bestimmten Gründen mit ausgegeben werden, muß die Einleitung mit %% beginnen.
-	Die Ausgabe ist linksbündig.
+	positives Vorzeichen
#	führende Null bei hexadezimaler und oktaler Darstellung
b	Minimalbreite des Ausgabefeldes
.n	Ziffern nach dem Punkt, die Voreinstellung ist 6
Von den folgenden Zeichen steht nur eins am Ende der Formatierung	
lf	bei double Variablen ddd.dddddd
f	bei float Variablen ddd.ddddd
e	d.dddddedefdd
E	d.dddddE±dd
g	wie in den Zeilen f oder e in kürzerer Darstellung
G	wie in den Zeilen f oder E in kürzerer Darstellung

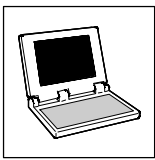
Programmbeispiel

Das folgende Programmbeispiel zeigt wie eine **Formatierte Zahlenausgabe** sich im DDS darstellt.

- Zuerst werden die Variablen deklariert und die Anweisungen geschrieben.

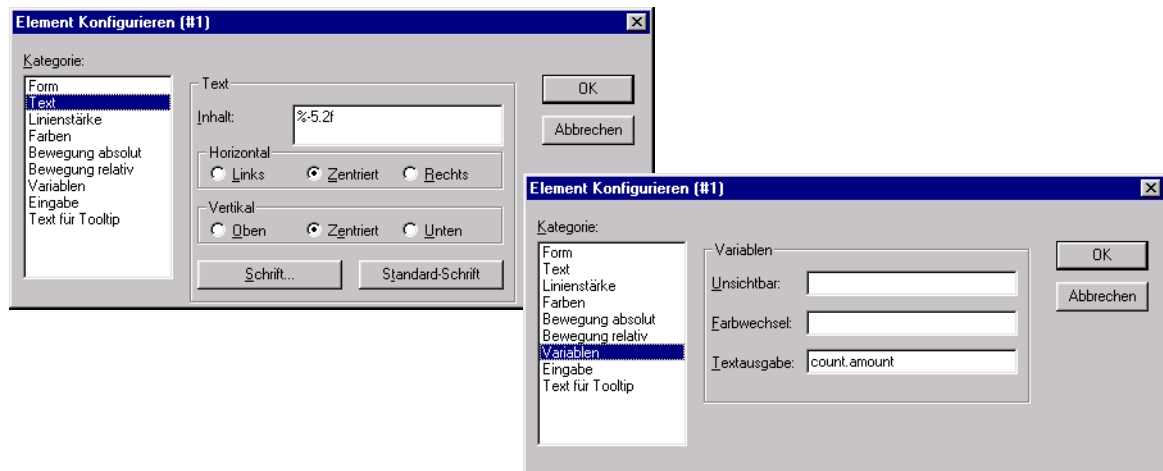
```
count (PRG-ST)
0001 PROGRAM count
0002 VAR
0003   counter: DINT;
0004   amount: REAL:=1;
0005   by1: BYTE;
0006   b1: BOOL;
0007 END_VAR
0008
0002
0003 amount:=2*amount;
0004
0005 by1:=by1-1;
0006
0007 b1:= NOT b1;
0008
```

- Dann wird ein Visualisierung mit zwei Rechtecke erzeugt. Element 0 soll folgende Konfiguration haben. Bei Element 1 muß dann nur unter der Kategorie *Variablen* in der **Textausgabe** `text.ftext2` eingegeben werden.
- In der Kategorie *Text*, im Eingabefeld **Inhalt** wird die **Formatierte Zahlenausgabe** eingegeben. Die vorgegebenen Einträge sollten zuerst übernommen werden.

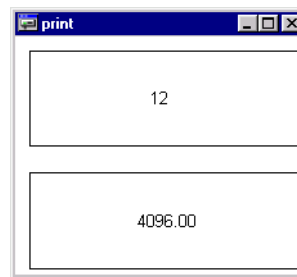


Drive PLC Developer Studio

Visualisierung



- Im Online-Modus sieht die Ausgabe dann so aus. Die vorher gemachten Einträge geben die untere Darstellung wieder.



Um die Auswirkung der **Formatierten Zahlenausgabe** zu testen, sollten auch andere Formate mit % eingegeben werden.

- "%s" wird im Online-Modus durch den Wert der Variablen im Eingabefeld **Textausgabe** ersetzt.

Konfigurieren→Eingabe

verfügbar für: alle

Über **Extras→Konfigurieren**, Kategorie *Eingabe* können Sie festlegen, ob im Online-Modus Eingaben mit der Maus und über Tastatur möglich sein sollen oder nicht.

Variable toggeln

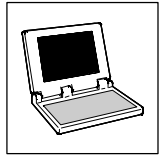
Ist das Kontrollkästchen **Variable toggeln** aktiviert, toggeln Sie mit jedem Mausklick den Wert der boolschen Variablen, die im Eingabefeld dahinter angegeben ist.

- Der Wert der boolschen Variable wechselt mit jedem Mausklick auf das Visualisierungselement von **TRUE** auf **FALSE** bzw. von **FALSE** auf **TRUE**.

Variable tasten

Ist das Kontrollkästchen **Variable tasten** aktiviert, wechseln Sie durch Drücken und Gedrückthalten der linken Maustaste den Wert der boolschen Variablen, die im Eingabefeld dahinter angegeben ist.

- Der Wert der boolschen Variable wechselt von **TRUE** auf **FALSE** bzw. von **FALSE** auf **TRUE**, wenn Sie den Mauszeiger auf dem Visualisierungselement platzieren und dann die linke Maustaste drücken und gedrückt halten.
- Der Wert der boolschen Variable wechselt in seinen Ausgangszustand zurück, wenn Sie die linke Maustaste wieder loslassen.
- Ist das Kontrollkästchen **FALSE tasten** markiert, wechselt der Wert bei **OK** von **TRUE** auf **FALSE**.



Achtung!

Die Zuverlässigkeit dieser Funktion ist nicht immer gewährleistet. Verwenden Sie das Tasten von Variablen nicht für sicherheitsrelevante Funktionen der Antriebssteuerung (z. B. Positionieren).

Zoomen nach Vis.

Ist das Kontrollkästchen **Zoomen nach Vis** aktiviert, wechseln Sie bei Mausklick auf das Visualisierungselement in das Fenster der im Eingabefeld angegebenen Visualisierung.

- Wenn Sie zusätzlich das Kontrollkästchen **Variable toggeln** aktiviert haben, wird die Variable getoggelt, die im Textfeld **Variable toggeln** angegeben ist.
- Wenn Sie anstelle der Visualisierung eine Programmvariable vom Typ **STRING** (z. B. `PLC_PRG.xxx`) im Textfeld angeben, kann über diese Variable der Name der Visualisierung vorgegeben werden (z. B. `xxx:='visu1'`).
- Geben Sie in das Textfeld **Zoomen nach Vis** den Befehl **ZOOMTOCALLER** ein, wird im Online Modus durch ein Mausklick auf das Element ein zurückspringen zur aufrufenden Visualisierung erreicht, falls eine solche Konstellation konfiguriert wurde.

Soll zu einer Visualisierungsreferenz mit Platzhaltern gesprungen werden, können diese direkt beim Aufruf durch Variablennamen bzw. Texte ersetzt werden.

Beachten Sie folgende Syntax.

```
<Visuname>(<Platzhalter1>:=<Text1>,<Platzhalter2>:=<Text2>,...,<Platzhaltern>:=<Textn>)
```

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)
```

Programm ausführen

Ist das Kontrollkästchen **Programm ausführen** aktiviert, starten Sie bei Mausklick auf das Visualisierungselement das im Eingabefeld angegebene ausführbare Programm.

Beispiel

notepad C:/help.txt (das Programm notepad wird gestartet und die Datei help.txt geöffnet.)

Text Eingabe der Variable 'Textausgabe'

Ist das Kontrollkästchen **Text Eingabe der Variable 'Textausgabe'** aktiviert, können Sie einer Variablen durch Mausklick auf das Visualisierungselement einen Wert zuweisen.

- Der Wert wird der Variablen zugewiesen, die in der Kategorie *Variablen*, im Textfeld **Textausgabe** angegeben ist.
- Wenn Sie auf das Visualisierungselement klicken, erscheint ein Eingabefeld, in das Sie den neuen Wert für die Variable eingeben können. Durch Drücken der **<Eingabetaste>** wird der Wert übernommen.

Konfigurieren→Text für Tooltip

verfügbar für: alle

Über **Extras→Konfigurieren**, Kategorie **Text für Tooltip** können Sie einen Text für das selektierte Element eingeben. Dieser Text wird angezeigt, wenn Sie den Mauszeiger für kurze Zeit über dem Element halten.

- Zeilenumbrüche im Text können Sie mit der Tastenkombination **<Strg>+<Eingabetaste>** einfügen.



Drive PLC Developer Studio

Visualisierung

Konfigurieren→Bitmap

verfügbar für: Bitmap

Über **Extras→Konfigurieren**, Kategorie *Bitmap* können Sie Einstellungen für das selektierte Bitmap vornehmen.

Bitmap

Im Eingabefeld **Bitmap** geben Sie die Bitmap-Datei mit ihrem Pfad an.

- Durch Klicken auf ... öffnen Sie den Standarddialog zum Browsen durch Ihre Dateistruktur, mit dem Sie die gewünschte Datei auswählen können.
- Mit dem Kontrollkästchen **Hintergrund transparent**, kann eine im Bitmap enthaltene Farbe als transparent definiert werden. Mit der Schaltfläche **Transparente Farbe** wird die Farbe festgelegt.

Rahmen

- Ist das Kontrollkästchen **Anisotropisch** aktiviert, ist das Bitmap so groß wie sein Rahmen und läßt sich durch Größenänderung beliebig verzerren.
- Ist das Kontrollkästchen **Isotropisch** aktiviert, bleiben die Proportionen der Bitmap auch bei Größenänderungen immer erhalten, d.h. das Verhältnis von Länge und Breite bleibt gewahrt.
- Ist das Kontrollkästchen **Fest** aktiviert, wird das Bitmap unabhängig von seinem Rahmen in der Originalgröße dargestellt.
- Ist das Kontrollkästchen **Abschneiden** aktiviert, wird bei der Einstellung *Fest* nur der Ausschnitt des Bitmaps angezeigt, der vom Rahmen umschlossen ist.
- Ist das Kontrollkästchen **Zeichnen** aktiviert, wird der Rahmen in der Farbe dargestellt, die über die Schaltflächen **Farbe** und **Alarmfarbe** festgelegt wurde. Die Alarmfarbe wird nur angezeigt, wenn die in der Kategorie **Variablen** im Feld *Farbwechsel* angegebene Variable **TRUE** ist.

Konfigurieren→Visualisierung

verfügbar für: Visualisierung

Über **Extras→Konfigurieren**, Kategorie *Visualisierung* können Sie Einstellungen für eine Visualisierung vornehmen, die als Element in die aktuelle Visualisierung eingefügt ist. Nach dem Einfügen wird diese als **Referenz** der ursprünglichen bezeichnet.

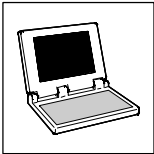
Visualisierung

Im Eingabefeld **Visualisierung** geben Sie den Objektnamen der Visualisierung an.

- Es können alle Visualisierungen angegeben werden. Die aktuelle Visualisierung und eine Referenz auf die aktuelle dürfen nicht angegeben werden.
- Durch Klicken auf ... öffnen Sie das Dialogfenster *Visualisierung auswählen*, in dem alle zur Verfügung stehenden Visualisierungen des Projektes aufgelistet sind.

Rahmen

- Ist das Kontrollkästchen **Zeichnen** aktiviert, wird der Rahmen in der Farbe dargestellt, die über die Schaltflächen **Farbe** und **Alarmfarbe** festgelegt wurde. Die Alarmfarbe wird nur angezeigt, wenn die in der Kategorie *Variablen* im Feld **Farbwechsel** angegebene Variable **TRUE** ist.
- Ist das Kontrollkästchen **Isotropisch** aktiviert, bleiben die Proportionen der Visualisierung auch bei Größenänderungen immer erhalten, d.h. das Verhältnis von Länge und Breite bleibt gewahrt.
- Ist die Kontrollkästchen **Ausschneiden** aktiviert, wird im Online-Modus nur der Originalausschnitt der Visualisierung angezeigt. Wandert z. B. ein Objekt außerhalb der ursprünglichen Anzeige, so wird dieses Objekt abgeschnitten und verschwindet möglicherweise ganz aus dem Blickfeld der Visualisierung.



- Die Schaltfläche **Platzhalter** führt zum Dialogfeld *Platzhalter ersetzen*. Es werden alle im eingefügten Visualisierungsbaustein verwendeten Platzhalter aufgelistet. Die Spalte **Ersetzungen** bietet die Möglichkeit, diese durch einen bestimmten Wert zu ersetzen. Welche Ersetzungen jeweils möglich sind, hängt davon ab, ob im Dialogfeld *Platzhalterliste* eine Wertemenge vordefiniert wurde. Die Auswahl erscheint in einem Kombinationsfeld. Wurde nichts vordefiniert, kann durch Doppelklick auf das entsprechende Feld in der Spalte Ersetzung den Platzhalter ein Editierfeld geöffnet und beliebig ausgefüllt werden.
- Eine weitere Möglichkeit, Platzhalter in Referenzen zu ersetzen, besteht direkt beim Aufruf einer Visualisierung über einen Eintrag im Textfeld **Zoomen nach Vis.:** in der Kategorie *Eingabe* des Konfigurationsdialogs für ein Visualisierungselement.



Achtung!

Auf die zeitliche Reihenfolge der Ersetzung kann kein Einfluß genommen werden! Es sollten 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 Watchausdruck...")

Platzhalterkonzept

Beispiel für den Einsatz des Platzhalterkonzeptes

Instanzen eines Funktionsblocks können auf einfache Weise mit Hilfe von Referenzen derselben Visualisierung dargestellt werden.

Platzhalter ermöglichen bei dem Konfigurieren der Visualisierungselemente, daß ein Visualisierungsobjekt mehrfach verwendet werden kann.

Wird ein mit Platzhaltern erstelltes Visualisierungsobjekt in eine andere Visualisierung eingefügt, werden die Platzhalter durch Variablen oder Strings ersetzt.

Visualisierungsobjekte können neben Bausteinen und globalen Variablen auch in Bibliotheken abgelegt werden. Das Verwenden von Platzhaltern ermöglicht auf die Instanzen der Bausteine innerhalb der Bibliothek zu referenzieren



Achtung!

Wenn Sie nach Einfügen einer Visualisierung diese Referenz markieren und konfigurieren, wird sie als Objekt betrachtet und reagiert im Online Betrieb als solches entsprechend seiner Konfiguration auf Eingaben. Werden für die Referenz an eingefügter Stelle keine Konfigurationseinträge vorgenommen, reagieren ihre Einzelelemente online wie diejenigen der Ursprungsvisualisierung.



Drive PLC Developer Studio

Visualisierung

Visualisierungsobjekt mit Platzhaltern

- Deklarieren Sie folgenden Funktionsblock in ST.

```
PROGRAM PLC_PRG

VAR

    nVar10:INT:=10;

    nVar20:INT:=20;

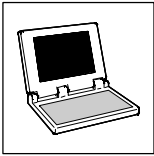
END_VAR
```

Im Anweisungsteil geben Sie ein Semikolon ";" ein.

- Erstellen Sie eine neue Visualisierung mit den Namen Screen1 und fügen Sie mit **Einfügen→Rechteck** ein Rechteck ein.
- Markieren Sie das Rechteck und öffnen Sie den Konfigurationsdialog. In der Kategorie *Text*, Eingabefeld **Inhalt** geben Sie "Wert der Referenz" ein. Die anderen Optionen lassen Sie unverändert.
- In der Kategorie *Variablen*, Eingabefeld **Textausgabe** geben sie "\$Ref\$" ein. Platzhalter werden in den Zeichen \$ eingeschlossen. Die Zeichen müssen aufgeführt werden.
- Bestätigen Sie den Dialog mit **OK**.
- Markieren Sie das Rechteck erneut. Öffnen Sie mit **Extras→Platzhalterliste** das Dialogfeld *Platzhalter: mögliche Ersetzungen festlegen*, vervollständigen Sie das Dialogfeld und bestätigen Sie mit **OK**.

Platzhalter	Elementnummer	Ersetzungen
Ref	0	PLC_PRG.nVar10,PLC_PRG.nVar20

- Erstellen sie eine weitere Visualisierung mit den Namen Screen2.
- Fügen Sie eine Visualisierungsreferenz ein. Die Visualisierung Screen2 muß aktiv sein. Führen Sie den Menübefehl **Einfügen→Visualisierung** aus und ziehen Sie in der Visualisierung Screen2 ein Rechteck auf. Es erscheint das Dialogfeld *Visualisierung auswählen*. Wählen Sie Screen1 und bestätigen Sie mit **OK**.
- Markieren Sie die gesamte Referenz und öffnen Sie den Konfigurationsdialog. In der Kategorie *Visualisierung* (Screen1) drücken Sie die Schaltfläche **Platzhalter**. Wählen Sie aus den Ersetzungen *PLC_PRG.nVar20*. aus und bestätigen mit **OK**.
- In der Kategorie *Visualisierung* (Screen1) markieren Sie nur die Kontrollkästchen **Isotropisch** und **Abschneiden** und bestätigen mit **OK**.
- Speichern Sie das Programm mit einem geeigneten Namen und loggen Sie ein.



9.3.2.1 Gruppieren

Symbol: -	Menübefehl: Extras→Gruppieren	Tastaturbefehl: -
-----------	-------------------------------	-------------------



Tip!

Eine Gruppierung hat den Vorteil, daß mehrere Elemente gleichzeitig verschoben werden können. Die Abstände zueinander bleiben dabei gleich.

Vorgehensweise beim Gruppieren

- Markieren der Elemente, die gruppiert werden sollen.
- **Extras→Gruppieren** auswählen.
- Gruppierung bearbeiten (verschieben, löschen)
- Sind die Elemente fertig bearbeitet kann die Gruppierung mit **Extras→Gruppierung aufheben** beendet werden.



Drive PLC Developer Studio

Visualisierung

9.4 Visualisierung in Bibliotheken

Visualisierungen können auch in Bibliotheken mit abgelegt werden und dadurch als Bibliotheksbausteine in Projekten zu Verfügung gestellt werden.

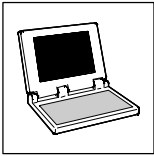
- Sie können, wie die im Projekt direkt vorliegenden Visualisierungen, als Referenz eingefügt werden.
- Mit dem Befehl **Zoomen nach Vis.** wird eine andere Visualisierung geöffnet.



Tip!

In einem Projekt müssen Visualisierungen eindeutige Namen haben. Referenzierte- und vorliegende Visualisierungen müssen eindeutig verschiedene Namen haben.

Während des Programmablaufs werden zuerst die Visualisierungen im Projekt, danach die der geladenen Bibliotheken berücksichtigt.



10 IEC 61131-3 Datentypen

10.1 Standard-Datentypen

Der Benutzer kann Standard-Datentypen und selbstdefinierte Datentypen beim Programmieren verwenden. Jedem Bezeichner wird ein Datentyp zugeordnet, der festlegt, wieviel Speicherplatz reserviert wird und welche Werte dem Speicherinhalt entsprechen.

Standard-Datentypen sind:

- BOOL
- Ganzzahlige Datentypen
(BYTE, WORD, DWORD, SINT, USINT, INT, UINT, DINT, UDINT)
- REAL
- STRING
- Zeitdatentypen
(TIME, TIME_OF_DAY, DATE, DATE_AND_TIME)

10.1.1 BOOL

Variablen vom Typ **BOOL** können die Wahrheitswerte **TRUE** und **FALSE** annehmen. Es werden 8 Bit Speicherplatz reserviert.

10.1.2 Ganzzahlige Datentypen

Zu den ganzzahligen Datentypen gehören **BYTE**, **WORD**, **DWORD**, **SINT**, **USINT**, **INT**, **UINT**, **DINT**, **UDINT**.

Die unterschiedlichen Datentypen decken einen unterschiedlichen Zahlenbereich ab. Für die ganzzahligen Datentypen gelten die folgenden Bereichsgrenzen:

Typ	Untergrenze	Obergrenze	Speicherplatz
BYTE	0	255	8 Bit
WORD	0	65535	16 Bit
DWORD	0	4294967295	32 Bit
SINT	-128	127	8 Bit
USINT	0	255	8 Bit
INT	-32768	32767	16 Bit
UINT	0	65535	16 Bit
DINT	-2147483648	2147483647	32 Bit
UDINT	0	4294967295	32 Bit

Dadurch kann es passieren, daß bei der Typkonvertierung von größere auf kleinere Typen Informationen verloren gehen.

10.1.3 REAL und LREAL

REAL und **LREAL** sind sogenannte Gleitpunkttypen. Sie sind nötig bei der Verwendung von rationalen Zahlen. Der reservierte Speicherplatz beträgt 32 Bit bzw. 64 Bit.



Hinweis!

LREAL wird von den Lenze Zielsystemen nicht unterstützt.

REAL Datentypen nur im PLC_PRG-Kontext verarbeiten. Es können Systemfehler auftreten.



Drive PLC Developer Studio

IEC 61131-3 Datentypen

10.1.4 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 **20 Zeichen** angenommen.

Beispiel einer Stringdeklaration mit 20 Zeichen:

```
str:STRING(20):='This is a string';
```

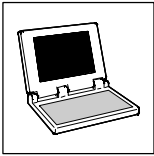
10.1.5 Zeitdatentypen

Die Datentypen **TIME**, **TIME_OF_DAY** (Kurz TOD), **DATE** und **DATE_AND_TIME** (kurz DT) werden intern wie **DWORD** behandelt.

- Bei **TIME** und **TIME** wird die Zeit in Millisekunden angegeben, wobei bei TOD ab 00:00 Uhr gerechnet wird.
- Bei **DATE** und **DT** wird die Zeit in Sekunden angegeben, wobei ab dem 1. Januar 1970 um 00:00 Uhr gerechnet wird.
- Bei der Zeitangabe steht **h** für Stunden, **m** für Minuten, **s** für Sekunden und **ms** für Millisekunden.

Beispiel

```
TIME#2h4m12s123ms
```



10.2 Definierte Datentypen

10.2.1 Array

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. Arrays können auch von Strukturen gebildet werden.

Syntax:

`<Feld_Name>: ARRAY [<ug1>..<og1>,<ug2>..<og2>] OF <elem. Typ>.`

- `<ug1>`, `<ug2>` geben die untere Grenze des Feldbereichs an.
- `<og1>`, `<og2>` geben die obere Grenze des Feldbereichs an.
- Die Grenzwerte müssen ganzzahlig sein.

Auf Komponenten von Arrays greift man bei einem zweidimensionalen Feld mit folgender Syntax zu:

`<Feld_Name>[Index1, Index2]`

Beispiel:

Deklaration

`Kartenspiel : ARRAY [1..13, 1..4] OF INT;`

Implementation

`Kartenspiel[9,2]`

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 in einer Struktur

`TYPE STRUCT1
 STRUCT
 p1:INT;
 p2:INT;
 p3:DWORD;
 END_STRUCT`

`ARRAY[1..3]OF STRUCT1:=(p1:=1,p2:=10,p3:=4723), (p1:=2,p2:=0,p3:=299),
(p1:=14,p2:=5,p3:=112);`



Drive PLC Developer Studio

IEC 61131-3 Datentypen

Elemente ohne vorgegeben Wert, werden mit dem Default-Initialwert des Basistypen initialisiert. Im obigen Beispiel werden also die Elemente anarray[6] bis anarray[10] mit 0 initialisiert.



Tip!

Mit einer Funktion CheckBounds aus der Bibliothek CheckBounds.lib können Bereichsüberschreitungen bei Arrays automatisch überprüft werden. Beachten Sie auch die CheckBounds.lib.

Bibliothek CheckBounds.lib

Die Bibliothek CheckBounds besitzt die globale Variable *g_bErrorCheckBounds*.

- Öffnen Sie die Bibliotheksverwaltung mit **Fenster→Bibliotheksverwaltung**.
- Fügen Sie die Bibliothek aus dem Bibliotheksordner ein. Mit dem Menübefehl **Einfügen→Weitere Bibliothek** können Sie die Bibliothek auswählen und in das aktuelle Projekt einfügen.
- Markieren Sie die Bibliothek in der Bibliotheksverwaltung. In der Registerkarte **Globale Variablen** ist dann die Variable mit Zusatzinformationen aufgeführt.



Hinweis!

Die globale Variable *g_bErrorCheckBounds* überwacht Zugriffe auf ein Array, das sich nicht in den definierten Grenzen befindet.

Die Funktion **CheckBounds** der Bibliothek erkennt, daß vom Zielsystem ein Array benutzt wird, das sich nicht in den definierten Grenzen befindet.

- Die globale Variable *g_bErrorCheckBounds* wird TRUE.
- Zustand TRUE wird während der Laufzeit nicht zurückgesetzt. Ist der Zustand TRUE, dann wurde mindestens einmal auf ein Array, das außerhalb der definierten Grenzen liegt zugegriffen.

Die Funktion CheckBounds wird, wenn sie Bestandteil des Projekts ist, automatisch aufgerufen. Ein gesonderter Aufruf der Funktion in der Implementierung ist nicht notwendig.

Ist die Funktion CheckBounds nicht Bestandteil des Projekts wird keine Bereichsüberschreitung unterbunden.

Erfolgt ein indirekter Zugriff auf das Array außerhalb der Array-Grenzen, dann erzwingt CheckBounds den Zugriff auf das erste bzw. letzte Element des Arrays.

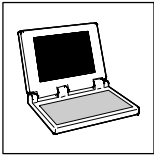
Implementierung

Deklarationsteil

```
FUNCTION Checkbounds : INT
VAR_INPUT
    index, lower, upper : INT;
END_VAR
```

Anweisungsteil

```
IF index < lower THEN
    CheckBounds := lower;
ELSEIF index > upper THEN
    CheckbBounds := 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 abgefangen werden.

Vereinbarungsteil

```
PROGRAM PLC_PRG
VAR
  A:ARRAY[0..7] OF BOOL;
  B:INT:=10;
```

Anweisungsteil

```
A[B] :=TRUE;
```

10.2.2 Pointer

In Pointern wird die Adresse von Variablen oder Funktionsblöcken zur Laufzeit eines Programms gespeichert.



Warnung!

Pointer können bei fehlerhafter Anwendung zum Absturz des Zielsystems führen. Verzichten Sie deshalb auf die Verwendung von Pointern.

Sollten Sie Pointer benutzen, dann immer mit großer sorgfalt anwenden. Das Schreiben von WORD-Datentypen auf ungerade Adressen oder der Zugriff auf Systemspeicherbereiche kann zu Fehlern führen.

Pointerdeklarationen haben folgende Syntax:

```
<Bezeichner>: POINTER TO <Datentyp/Funktionsblock>;
```

Ein Pointer kann auf jeden beliebigen Datentyp und Funktionsblock zeigen, auch selbstdefinierte Funktionsblöcke.

Mit dem Adreßoperator **ADR** wird dem Pointer die Adresse einer Variablen oder Funktionsblocks zugewiesen.

Das Dereferenzieren eines Pointers erfolgt über den Inhaltsoperator **^** nach dem Pointerbezeichner.

Beispiel:

Deklaration

```
pt:POINTER TO INT;
var_int1:INT := 5;
var_int2:INT;
```

Implementation

```
pt := ADR(var_int1);
var_int2:= pt^; (* var_int2 ist nun 5 *)
```



Drive PLC Developer Studio

IEC 61131-3 Datentypen

10.2.3 Aufzählungstyp

Ein Aufzählungstyp ist ein selbstdefinierter Datentyp, der aus einer Menge von Stringkonstanten besteht. Diese Konstanten bezeichnet man als Enumerationswerte.

Die Enumerationswerte sind im ganzen Projekt bekannt, auch wenn Sie in einem Baustein lokal deklariert wurden. 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
```

Die Werte sind zu ganzen Zahlen kompatibel, d.h. man kann damit Operationen wie mit **INT** durchführen. Dem <Bezeichner> kann eine Zahl x zugewiesen werden. Das gilt nur für Variable. Sind die Enumerationswerte nicht initialisiert, beginnt die Zählung bei 0. Mit jeder Aufzählung wird 1 Byte-Speicher belegt.

Achten Sie beim Initialisieren darauf, daß die Initialwerte aufsteigend sind. Die Gültigkeit der Zahl wird zur Laufzeit überprüft.

Beispiel:

```
TYPE
  TRAFFICLIGHT: (red, yellow, green:=10);
  (* red hat den Initialwert 0, yellow 1, green 10 *)
END_TYPE
```

Deklaration

```
TRAFFICLIGHT1:TRAFFICLIGHT:=0; (* TRAFFICLIGHT1 hat den Wert red *)
```

Implementation

```
IF TRAFFICLIGHT = red THEN
  TRAFFICLIGHT := GREEN;
END_IF
```

Der gleiche Enumerationswert darf nicht zweimal verwendet werden.

Beispiel:

```
AMPEL: (rot, gelb, gruen);
FARBE: (blau, weiss, rot);
```

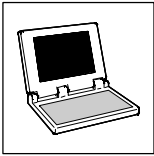
Fehler: rot darf nicht für AMPEL und FARBE verwendet werden.

10.2.4 Strukturen

Strukturen werden als Objekte im *Object Organizer* unter der Registerkarte **Datentypen** abgelegt. Sie beginnen mit den Schlüsselwörtern **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!**

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

Beispiel:

Strukturdefinition für einen 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
```

10.2.5 Referenzen

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[20] ;
END_TYPE;
```



Drive PLC Developer Studio

IEC 61131-3 Datentypen

10.2.6 Unterbereichstypen

Der Wertebereich eines Unterbereichstypen umfaßt nur eine Untermenge eines Basistypen. Die Deklaration kann im Register Datentypen erfolgen. Eine Variable kann auch direkt mit einem Unterbereichstypen deklariert werden.

Syntax für die Deklaration im Register Datentypen

```
TYPE<Name>:<Inttype>(<ug>..

```

<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 zum Basistypen kompatibel sein muß und die Untergrenze des Bereichstypen festlegt. Die Untergrenze gehört zu diesem Bereich.
<og>	Ist eine Konstante, die zum Basistypen kompatibel sein muß und die Obergrenze des Bereichstypen festlegt. Die Obergrenze gehört zu diesem Basistypen.

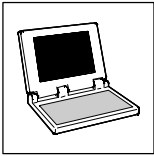
```
TYPE
    SubInt : INT (-4095..4095);
END_TYPE
```

- Direkte Deklaration einer Variablen mit einem Unterbereichstypen.
- Korrekte Angabe eines Initialwertes, wenn der Unterbereich nicht die 0 enthält.

```
VAR
    i1 : INT (-4095..4095);
    i2: INT (5..10):=5;
    ui : UINT (0..10000);
END_VAR
```

Fehlermeldung

Die zugewiesene Konstante eines Unterbereichstypen (in der Deklaration oder Implementation) muß in den Wertebereich fallen. Es wird sonst eine Fehlermeldung ausgegeben.



Bibliothek CheckRange.lib

Die Bibliothek CheckBounds besitzt die globale Variable *g_bErrorCheckRange*.

- Öffnen Sie die Bibliotheksverwaltung mit **Fenster→Bibliotheksverwaltung**.
- Fügen Sie die Bibliothek aus dem Bibliothekskunde ein. Mit dem Menübefehl **Einfügen→Weitere Bibliothek** können Sie die Bibliothek auswählen und in das aktuelle Projekt einfügen.
- Markieren Sie die Bibliothek in der Bibliotheksverwaltung. In der Registerkarte **Globale Variablen** ist dann die Variable mit Zusatzinformationen aufgeführt.



Hinweis!

Die globale Variable *g_bErrorCheckRange* überwacht kritische und nicht zu verarbeitende Wertebereiche.

Die Funktion **CheckRange** der Bibliothek erkennt, daß vom Zielsystem ein kritischer Wertebereich erreicht wurde.

- Die globale Variable *g_bErrorCheckBounds* wird TRUE.
- Zustand TRUE wird während der Laufzeit nicht zurückgesetzt. Ist der Zustand TRUE, dann wurde mindestens einmal ein kritischer Wertebereich erreicht.

CheckRangeSigned und CheckRangeUnsigned

Um diese Funktionen zu verwenden muß die Bibliothek CheckRange.lib eingebunden werden.

Um die Einhaltung der Bereichsgrenzen zur Laufzeit zu überprüfen, müssen die Funktionen CheckRangeSigned bzw. CheckRangeUnsigned eingefügt werden. Diese Funktionen vermeiden Bereichsverletzungen (durch abschneiden des Wertes oder Fehlerflag). Wird auf eine Variable geschrieben, werden CheckRangeSigned bzw. CheckRangeUnsigned implizit aufgerufen. Die Variable ist von einem Unterbereichstyp, der aus einem vorzeichenbehafteten oder vorzeichenlosen Typ gebildet wurde.

Im Falle einer Variable eines vorzeichenbehafteten Unterbereichstyps (var:i1 bzw. i2 Listing oben) wird die Funktion CheckRangeSigned aufgerufen. Die Programmierung könnte folgendermaßen aussehen, damit der Wert auf den erlaubten Bereich zurückgeschnitten wird.

```
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
```



Achtung!

Sind die beiden Funktionen CheckRangeSigned und CheckRangeUnsigned nicht vorhanden, findet zur Laufzeit keine Typüberprüfung der Unterbereichstypen statt! Die Variable i1 oder i2 könnte dann also durchaus beliebige Werte zwischen -32768 und 32767 annehmen!



Drive PLC Developer Studio

IEC 61131-3 Datentypen

Parametrierung der Funktion

value	Bekommt den Wert, der dem Bereichstypen zugewiesen werden soll.
lower	Ist die Untergrenze des Bereichs.
upper	Ist die Obergrenze des Bereichs.
Rückgabewert	Der tatsächliche zugewiesene Wert für den Bereichstypen.

Aus einer Zuweisung $i := 10 * y$; wird in diesem Beispiel implizit folgende Zuweisung erzeugt:
 $i := \text{CheckRangeSigned}(10 * y, -4095, 4095)$;

Wenn y beispielsweise den Wert 1000 hat, dann hat i nach dieser Zuweisung nur den Wert 4095.

Für die Funktion CheckRangeUnsigned müssen Funktionsname und Schnittstelle korrekt sein.

```
FUNCTION CheckRangeUnsigned : UDINT
VAR_INPUT
    value, lower, upper: UDINT;
END_VAR
```

Die Funktionen CheckRangeSigned und CheckRangeUnsigned sind in der Bibliothek CheckRange.lib enthalten. Fügen Sie die Bibliothek mit den Bibliotheksverwalter **Einfügen→Weitere Bibliothek** in das Projekt ein.



Tip!

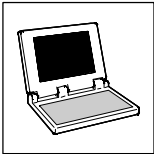
Wenn eine Funktion CheckRangeSigned bzw. CheckRangeUnsigned wie oben gezeigt implementiert ist, kann bei der Verwendung des Unterbereichstypen in einer FOR-Schleife eine Endlosschleife entstehen. Dies geschieht, wenn der für die FOR-Schleife angegebenen Bereich genauso groß oder größer ist als der des Unterbereichstypen.

```
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.

Der Inhalt der CheckRange-Funktionen bei der Verwendung von Inkrementationswerten in der FOR-Schleife ist zu beachten!



11 Übersicht der Operatoren

Die folgende Tabelle zeigt die **Operatoren** in ST und AWL mit den verfügbaren Modifikatoren in AWL.

In der Spalte **Operator AWL** wird nur die Zeile, mit der Verwendung des Operators, dargestellt.

Voraussetzung:
Ein erfolgreiches Laden in der vorangehenden Zeile des ersten benötigten Operators. (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 JMP, 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 dem Erreichen der abschließenden Klammer wird die vorangestellte Operation ausgeführt.

11.1 DDS integrierte IEC-Operatoren

Operator ST	Operator AWL	Mod.AWL	Bedeutung
'			Stringbegrenzung (z.B. 'string1')
.. []			Array: Darstellung des Array-Bereichs (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 den 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 Funktionsblockinstanz inst1 auf
<Fktname>(vx, vy,...)	<Fktname> vx, vy	CN	Rufe Funktion auf und übergebe Variablen 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,(	Bitweise AND
OR	OR	N,(	Bitweise OR
XOR	XOR	N,(	Bitweise exklusives OR
NOT	NOT		Bitweise NOT
+	ADD	(	Addition
-	SUB	(	Subtraktion
*	MUL	(	Multiplikation
/	DIV	(	Division
>	GT	(	größer
>=	GE	(	größer/gleich
=	EQ	(	gleich
<>	NE	(	ungleich
<=	LE	(	kleiner/gleich



Drive PLC Developer Studio

IEC 61131-3 Operatoren

Operator ST	Operator AWL	Mod.AWL	Bedeutung
<	LT	(	kleiner
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(K,in)	SHL K		bitweises Links-Shift eines Operanden um K
SHR(K,in)	SHR K		bitweises Rechts-Shift eines Operanden um K
ROL(K,in)	ROL K		bitweise Linksrotation eines Operanden um K
ROR(K,in)	ROR K		bitweise Rechtsrotation eines Operanden um K
SEL(G,in0,in1)	SEL in 0, in 1		binäre Selektion zw. 2 Operanden in0 (G ist FALSE) und in1 (G ist TRUE)
MAX(in0,in1)	MAX in 1		liefert von 2 Werten den größeren
MIN(in0,in1)	MIN in1		liefert von 2 Werten den kleinsten
LIMIT(MIN,in,Max)	LIMIT MIN, MAX		Limitierung des Wertebereichs (in wird bei Überschreitung auf Min bzw. Max zurückgesetzt)
MUX(K,in0,...,in_n)	MUX in0,...,in_1		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
BYTE_TO_<type>(in)	INT_TO_<type>		Typkonvertierung des BYTE-Operanden in anderen elementaren Typ
WORD_TO_<type>(in)	INT_TO_<type>		Typkonvertierung des WORD-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
DINT_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
DATE_TO_<type>(in)	DATE_TO_<type>		Typkonvertierung des Operanden in anderen elementaren Typ
DT_TO_<type>(in)	DT_TO_<type>		Typkonvertierung des Operanden in anderen elementaren Typ
STRING_TO_<type>(in)	STRING_TO_<type>		Typkonvertierung des Operanden in anderen elementaren Typ, in muß gültigen Wert des Zieltyps enthalten
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



11.2 Standard.lib integrierte IEC-Operatoren

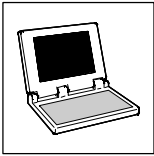
Operator ST	Operator AWL	Mod.AWL	Bedeutung
LEN(in)	LEN		Stringlänge des Operanden
LEFT(str,size)	LEFT size		linker Anfangsstring (Größe size) von String str
RIGHT(str,size)	RIGHT size		rechter Anfangsstring (Größe size) von String str
MID(str,size)	MID size		Teilstring der Größe size aus String str
CONCAT('str1','str2')	CONCAT 'str2'		Aneinanderhängen zweier Strings
INSERT('str1','str2',pos)	INSERT 'str2',pos		Einfügen String str1 in String str2 an Position pos
DELETE('str1',len,pos)	DELETE len,pos		Lösche Teilstring mit Länge len, beginnend an Position pos von str1
RE-PLACE('str1','str2',len,pos)	REPLACE 'str2',len,pos		Ersetze Teilstring von Länge len, beginnend an Position pos von str1 durch str2
FIND('str1','str2')	FIND 'str2'		Suchen eines Teilstrings str2 in str1
SR	SR		Bistabiler FB wird dominant gesetzt
RS	RS		Bistabiler FB wird zurückgesetzt
SEMA	SEMA		FB: Software Semaphor (unterbrechbar)
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
RTC	RTC		FB: Laufzeit-Uhr

Die detaillierte Beschreibung entnehmen Sie bitte den entsprechenden Bibliotheken.



Drive PLC Developer Studio

IEC 61131-3 Operatoren



12 IEC 61131-3 Operatoren

Das DDS unterstützt alle IEC-Operatoren.

Diese sind, im Gegensatz zu den Standardfunktionen, implizit im ganzen Projekt bekannt. In den Baustein-Implementationen werden Operatoren wie Funktionen benutzt.

12.1 Arithmetische Operatoren

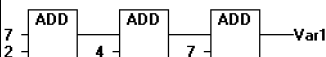
Typen:

Die Operanden der nachfolgenden arithmetischen Operatoren können vom Typ **BYTE**, **WORD**, **DWORD**, **SINT**, **USINT**, **INT**, **UINT**, **DINT**, **UDINT** und **REAL** sein.

12.1.1 ADD

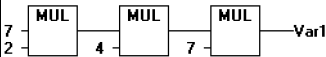
Addition

Es können auch zwei **TIME**-Variablen addiert werden, die Summe ist dann wieder eine Zeit.
(z. B. gilt $t\#45s + t\#50s = t\#1m35s$)

Beispiele		
AWL LD 7 ADD 2,4,7 ST Var1	ST Var1 := 7+2+4+7;	FUP 

12.1.2 MUL

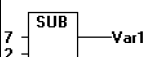
Multiplikation

Beispiele		
AWL LD 7 MUL 2,4,7 ST Var1	ST Var1 := 7*2*4*7;	FUP 

12.1.3 SUB

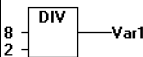
Subtraktion

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.

Beispiele		
AWL LD 7 SUB 2 ST Var1	ST Var1 := 7-2;	FUP 

12.1.4 DIV

Division

Beispiele		
AWL LD 8 DIV 2 ST Var1	ST Var1 := 8/2;	FUP 



Drive PLC Developer Studio

IEC 61131-3 Operatoren

12.1.5

Bibliothek CheckDiv.lib

Die Bibliothek CheckDiv besitzt die globale Variable *g_bErrorCheckDiv*

- Öffnen Sie den Bibliotheksverwaltung mit **Fenster→Bibliotheksverwaltung**.
- Fügen Sie die Bibliothek aus dem Bibliotheksordner ein. Mit dem Menübefehl **Einfügen→Weitere Bibliothek** können Sie die Bibliothek auswählen und in das aktuelle Projekt einfügen.
- Markieren Sie die Bibliothek in der Bibliotheksverwaltung. In der Registerkarte **Globale Variablen** ist dann die Variable mit Zusatzinformationen aufgeführt.



Hinweis!

Die globale Variable *g_bErrorCheckDiv* überwacht Divisionen durch Null.

Die Funktion **CheckDiv** der Bibliothek erkennt, daß vom Zielsystem eine Division durch Null durchgeführt wurde.

- Die globale Variable *g_bErrorCheckDiv* wird TRUE.
- Zustand TRUE wird während der Laufzeit nicht zurückgesetzt. Ist der Zustand TRUE, dann wurde mindestens einmal eine Division durch Null durchgeführt.



Hinweis!

Verwendung von CheckDiv Funktionen.

Die CheckDiv-Funktionen lassen keine Division durch Null zu. Der Divisor Null wird durch eins ersetzt.

Die CheckDiv.lib enthält folgende Funktionen.

CheckDivByte

CheckDivDint

CheckDivDWord

CheckDivInt

CheckDivReal

CheckDivSInt

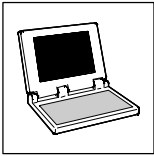
CheckDivUDint

CheckDivUInt

CheckDivUSInt

CheckDivWord

Wenn Sie einen DIV-Operator verwenden kann der Wert des Divisors geprüft werden. Die jeweilige Funktion muß genau die gezeigten Wortlaut haben. Bei einer Division mit dem Divisor Null ist (100/0), wird der Divisor mit eins ersetzt (100/1). Gleichzeitig wird die Variable *g_bErrorCheckdiv* auf **TRUE** gesetzt.



Implementierung von CheckDivReal

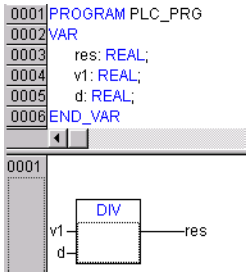
Vereinbarungsteil

```
FUNCTION CheckDivReal:REAL
VAR_INPUT
    divisor:REAL;
END_VAR
```

Anweisungsteil

```
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.



12.1.6 MOD

Modulo Division

Module Division einer Variablen vom Typ BYTE; WORD; DWORD; SINT; USINT; INT; UINT; DINT; UDINT mit einer anderen Variablen von einem dieser Typen. Als Ergebnis liefert diese Funktion den ganzzahligen Rest der Division.

Beispiele		
AWL LD 9 MOD 2 ST Var1 (* Var1 = 1 *)	ST Var1 := 9 MOD 2;	FUP

12.1.7 INDEXOF

Als Ergebnis liefert diese Funktion den internen Index eines Bausteins.

Beispiele		
AWL	ST Var1 := INDEXOF(baustein2);	FUP



Drive PLC Developer Studio

IEC 61131-3 Operatoren

12.1.8 SIZEOF

Als Ergebnis liefert diese Funktion die Anzahl der Bytes, die der angegebene Datentyp benötigt.

Beispiele		
AWL Deklaration arr1:ARRAY[0..4] OF INT; Var1:=INT; Implementation LD arr1 SIZEOF ST Var1 (* Var1 = 10 *)	ST Deklaration arr:ARRAY[0...4]OF INT; Var1:INT; Implementation Var1=SIZE OF (arr1);	FUP

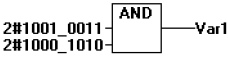
12.2 Bit-String-Operatoren

Typen:

Die Operanden der nachfolgenden Bit-String-Operatoren sollten vom Typ **BOOL**, **BYTE**, **WORD** oder **DWORD** sein.

12.2.1 AND

Bitweises AND von Bit-Operanden

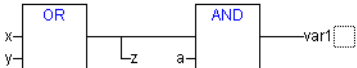
Beispiele	
AWL Var1 : BYTE; LD 2#1001_0011 AND 2#1000_1010 ST Var1 (* Var1 = 2#1000_0010 *)	FUP 
ST var1 := 2#1001_0011 AND 2#1000_1010	



Hinweis!

Bei der Verwendung von 68xxx- oder C-Code-Generatoren im FUP muß bei dem dargestellten Programmablauf folgendes beachtet werden.

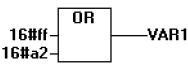
Sobald die Eingangsvariable a den Wert FALSE hat, wird die Zuweisung des Wertes der zweiten Eingangsvariablen am AND-Operator-Baustein zur Variablen z wird aufgrund der optimierten Abarbeitungsprozedur im FUP nicht mehr durchgeführt.





12.2.2 OR

Bitweises OR von Bit-Operanden

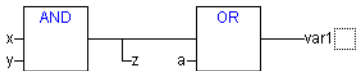
Beispiele	
AWL Var1 : BYTE; LD 2#1001_0011 OR 2#1000_1010 ST Var1 (* Var1 = 2#1001_1011 *)	FUP 
ST Var1 := 2#1001_0011 OR 2#1000_1010	



Hinweis!

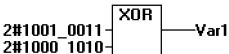
Bei der Verwendung von 68xxx- oder C-Code-Generatoren im FUP muß bei dem dargestellten Programmablauf folgendes beachtet werden.

Sobald die Eingangsvariable a den Wert TRUE hat, wird die Zuweisung des Wertes der zweiten Eingangsvariablen am OR-Operator-Baustein zur Variablen z wird aufgrund der optimierten Abarbeitungsprozedur im FUP nicht mehr durchgeführt.



12.2.3 XOR

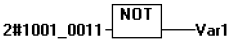
Bitweises XOR von Bit-Operanden

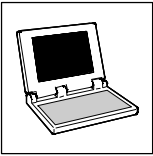
Beispiele	
AWL Var1 : BYTE; LD 2#1001_0011 XOR 2#1000_1010 ST Var1 (* Var1 = 2#0001_1001 *)	FUP 
ST Var1 := 2#1001_0011 XOR 2#1000_1010	

12.2.4 NOT

Bitweises Inversion eines Bit Operanden

Der Operand sollte vom Typ BOOL, BYTE, WORD oder DWORD sein.

Beispiele	
AWL Var1 : BYTE; LD 2#1001_0011 NOT ST Var1 (* Ergebnis = 2#0110_1100 *)	FUP 
ST Var1 := NOT 2#1001_0011	



Drive PLC Developer Studio

IEC 61131-3 Operatoren

12.3 Bit-Shift-Operatoren

Typen:

Die Operanden der nachfolgenden Bit-Shift-Operatoren sollten vom Typ **BYTE**, **WORD** oder **DWORD** sein.

12.3.1 SHL

Bitweises Links-Shift eines Operanden

 A := SHL (IN, N)	Beispiel AWL: LD 1 SHL 1 ST Var1 (* Var1 = 2 *)
---	---

IN wird um N Bits nach links geschoben, und von rechts mit Nullen aufgefüllt.

12.3.2 SHR

Bitweises Rechts-Shift eines Operanden

 A := SHR (IN, N)	Beispiel AWL: LD 32 SHL 2 ST Var1 (* Var1 = 8 *)
---	--

IN wird um N Bits nach rechts geschoben, und von links mit Nullen aufgefüllt.

12.3.3 ROL

Bitweise Linksrotation eines Operanden

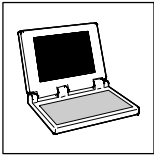
 A := ROL (IN, N)	Beispiel AWL: LD 2#1001_0011 ROL 3 ST Var1 (* Var1 = 2#1001_1100 *)
---	--

IN wird N mal um eine Bitstelle nach links geschoben, wobei das Bit ganz links von rechts wieder eingeschoben wird.



Tip!

Die Anzahl der Bits für die Rechenoperation werden durch den Datentyp der Eingangsvariable IN vorgegeben. Bei einer Konstante wird der kleinste Datentyp berücksichtigt. Der Datentyp der Ausgangsvariable bleibt ohne Auswirkung auf die Rechenoperation.



12.3.4 ROR

Bitweise Rechtsrotation eines Operanden

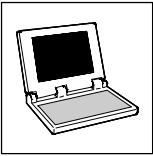
ROR in n A := ROR (IN, N)	Beispiel AWL: LD 2#1001_0011 ROR 3 ST Var1 (* Var1 = #0111_0010 *)
--	---

IN wird N mal um eine Bitstelle nach rechts geschoben, wobei das rechteste Bit von links wieder eingeschoben wird.



Tip!

Die Anzahl der Bits für die Rechenoperation werden durch den Datentyp der Eingangsvariable IN vorgegeben. Bei einer Konstante wird der kleinste Datentyp berücksichtigt. Der Datentyp der Ausgangsvariable bleibt ohne Auswirkung auf die Rechenoperation.



Drive PLC Developer Studio

IEC 61131-3 Operatoren

12.4 Auswahloperatoren

Sämtliche Auswahloperationen lassen sich auch auf Variablen durchführen.
Wegen der besseren Anschaulichkeit beschränken wir uns in den folgenden Beispielen auf Konstanten als Operanden.

12.4.1 SEL

Binäre Selektion

```
OUT := SEL(G, IN0, IN1)
```

bedeutet:

```
IF G THEN
  OUT:=IN1;
ELSE
  OUT:=IN0;
END_IF
```

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.

Beispiele		
AWL LD TRUE SEL 3,4 (*IN0=3, IN1=4*) ST Var1 (*Ergebnis=4 *) LD FALSE SEL 3,4 ST Var1 (*Ergebnis=3*)	ST Var1:=SEL(TRUE,3,4); (*Ergebnis Var1=4*)	FUP



Tip!

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.

In der Simulation werden aber sämtliche Zweige berechnet.

12.4.2 MAX

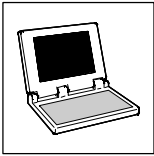
Maximumsfunktion

Liefert von zwei Werten den größten.

```
OUT := MAX(IN0, IN1)
```

IN0, IN1 und OUT können von beliebigem Typ sein.

Beispiele		
AWL LD 90 MAX 30 MAX 40 MAX 77 ST Var1 (* Var1 = 90 *)	ST	FUP



12.4.3 MIN

Minimumsfunktion

Liefert von zwei Werten den kleinsten.

```
OUT := MIN(IN0, IN1)
```

IN0, IN1 und OUT können von beliebigem Typ sein.

Beispiele		
AWL LD 90 MIN 30 MIN 40 MIN 77 ST Var1 (* Var1 = 30 *)	ST	FUP

12.4.4 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.

Beispiele		
AWL LD 90 LIMIT 30,80 ST Var1 (* Var1 = 80 *)	ST	FUP



Drive PLC Developer Studio

IEC 61131-3 Operatoren

12.4.5 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`).

Beispiele		
AWL	ST	FUP
LD 0 MUX 30,40,50,60,70,80 ST Var1 (* Var1 = 30 *)		



Tip!

Zum Zweck der Laufzeitoptimierung wird nun der Ausdruck berechnet, der `INk` vorgeschaltet ist berechnet. In der Simulation dagegen werden alle Zweige berechnet.

12.5 Vergleichsoperatoren

Typen:

Die Operanden der nachfolgenden Vergleichsoperatoren können vom Typ `BOOL`, `BYTE`, `WORD`, `DWORD`, `SINT`, `USINT`, `INT`, `UINT`, `DINT`, `UDINT`, `REAL`, `TIME`, `DATE`, `TIME_OF_DAY`, `DATE_AND_TIME` und `STRING` sein.

12.5.1 GT

Größer als

Ein boolescher Operator mit dem Ergebnis **TRUE**, wenn der erste Operand größer als der zweite ist.

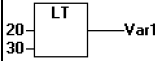
Beispiele		
AWL	ST	FUP
LD 20 GT 30 ST Var1 (* Var1 = FALSE *)	Var1 := 20 > 30;	



12.5.2 LT

Kleiner als

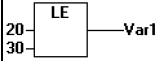
Ein boolescher Operator mit dem Ergebnis **TRUE**, wenn der erste Operand kleiner als der zweite ist.

Beispiele		
AWL LD 20 LT 30 ST Var1 (* Var1 = TRUE *)	ST Var1 := 20 < 30;	FUP 

12.5.3 LE

Kleiner oder gleich

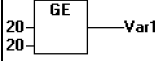
Ein boolescher Operator mit dem Ergebnis **TRUE**, wenn der erste Operand kleiner als der zweite Operand oder gleich groß wie der zweite Operand ist.

Beispiele		
AWL LD 20 LE 30 ST Var1 (* Var1 = TRUE *)	ST Var1 := 20 <= 30;	FUP 

12.5.4 GE

Größer oder gleich

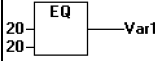
Ein boolescher Operator mit dem Ergebnis **TRUE**, wenn der erste Operand größer als der zweite Operand oder gleich groß wie der zweite Operand ist.

Beispiele		
AWL LD 60 GE 40 ST Var1 (* Var1 = TRUE *)	ST Var1 := 60 >= 40;	FUP 

12.5.5 EQ

Gleichheit

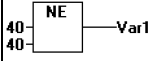
Ein boolescher Operator mit dem Ergebnis **TRUE**, wenn die Operanden gleich sind.

Beispiele		
AWL LD 40 EQ 40 ST Var1 (* Var1 = TRUE *)	ST Var1 := 40 = 40;	FUP 

12.5.6 NE

Ungleichheit

Ein boolescher Operator mit dem Ergebnis **TRUE**, wenn die Operanden ungleich sind.

Beispiele		
AWL LD 40 NE 40 ST Var1 (* Var1 = FALSE *)	ST Var1 := 40 <> 40;	FUP 



Drive PLC Developer Studio

IEC 61131-3 Operatoren

12.6 Adreßoperatoren

12.6.1 ADR

Adreßfunktion

ADR liefert die Datenspeicheradresse seines Arguments in einem **DWORD**. Die ermittelte Adresse kann innerhalb des Projektes an einen Pointer zugewiesen werden.

Beispiele		
AWL LD var1 ADR ST pt	ST	FUP

12.6.2 Inhaltsoperator

Die Dereferenzierung eines Pointers erfolgt über den Inhaltsoperator **^** nach dem Pointerbezeichner.

Beispiele		
AWL Deklaration pt:POINTER TO INT; var:int1:INT; var_int2:INT Implementation LD var_int1 ADR ST pt LD pt^ ST var_int2	ST Deklaration pt:POINTER TO INT; var_int1:INT; var_int2:INT; Implementation pt := ADR(var_int1); var_int2:=pt^;	FUP

12.7 Aufrufoperator

12.7.1 CAL

Aufruf eines Funktionsblocks

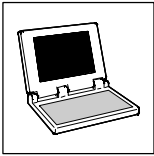
Mit **CAL** ruft man in AWL die Instanz eines Funktionsblock auf.

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**.

```
CAL INST(PAR1 := 0, PAR2 := TRUE)
```



12.8 Zuweisungsoperator

12.8.1 MOVE

Zuweisungsoperator

Der MOVE-Befehl ist nur im KOP-Editor sinnvoll.

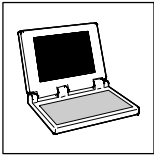
Ist *EN* auf TRUE wird der Inhalt der Variable *a* an die Variable *b* übergeben.

Beispiele		
AWL	ST	KOP



Drive PLC Developer Studio

IEC 61131-3 Operatoren



13 IEC 61131-3 Operanden

Im DDS können als Operanden Konstanten, Variablen, Adressen und evtl. Funktionsaufrufe auftreten.

13.1 Konstanten



Hinweis!

Nicht alle Zielsysteme unterstützen die verschiedenen Datentypen.

13.1.1 Zahlenkonstanten

Zahlenwerte können als Dualzahlen, Oktalzahlen, Dezimalzahlen und Hexadezimalzahlen auftreten.

- Wenn ein Integerwert keine Dezimalzahl ist, dann muß seine Basis gefolgt von einer Raute # 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 nicht erlaubt, mit Ausnahme von Dualzahlen.

Beispiele:

```
14          (* Dezimalzahl *)
2#1001_0011 (* Dualzahl *)
8#67        (* Oktalzahl *)
16#A        (* Hexadezimalzahl *)
```

- Der Typ dieser Zahlenwerte kann **BYTE**, **WORD**, **DWORD**, **SINT**, **USINT**, **INT**, **UINT**, **DINT**, **UDINT**, **REAL** oder **LREAL** sein.
- Implizite Konvertierungen von "größere" auf "kleinere" Typen sind nicht erlaubt. Eine **DINT**- Variable kann nicht ohne weiteres als **INT**-Variable benutzt werden. Hierfür benutzt man die Typkonvertierungsfunktionen der Standard-Bibliothek.

13.1.2 BOOL-Konstanten

BOOL-Konstanten sind die Wahrheitswerte **TRUE** und **FALSE**. Sie belegen 1 Byte im Speicher.

13.1.3 TIME-Konstanten

Im DDS 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 **ms**), wobei nicht alle Zeiten vorkommen müssen.



Drive PLC Developer Studio

IEC 61131-3 Operanden

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;
```

Beispiele für nicht korrekte TIME-Konstanten:

```
TIME1 := t#5m68s;      (*Überlauf bei einer niedrigeren Stelle*)
TIME1 := 15ms;         (*Es fehlt T#*)
TIME1 := t#4ms13d;     (*falsche Reihenfolge der Zeitangaben*)
```

Maximale Größe: DWORD 2^{32} -1ms $\approx$ 49Tage

13.1.4 DATE-Konstanten

Mit diesem Typ können Sie 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
```

13.1.5 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
```

13.1.6 DATE_AND_TIME-Konstanten

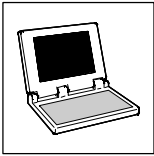
Datumskonstanten und Uhrzeiten können auch kombiniert werden zu sogenannten **DATE_AND_TIME**-Konstanten.

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
```



13.1.7 REAL-und-LREAL-Konstanten

REAL-Konstanten können als Dezimalbrüche und in Exponentialdarstellung angegeben werden. Man verwendet hierbei die amerikanische Schreibweise mit Punkt.

Beispiele:

```
7.4          (* statt 7,4 *)
1.64e+009    (* statt 1,64e+009 *)
```

13.1.8 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 8-Bit Zeichencodes interpretiert.

Außerdem werden, wenn sie in einer Zeichenfolge auftauchen, Kombinationen von zwei Zeichen, die mit dem Dollarzeichen beginnen, wie folgt interpretiert:

\$ \$	Dollarzeichen
\$ '	Hochkomma
\$L or \$l	Zeilenvorschub
\$N or \$n	Neue Zeile
\$P or \$p	Seitenvorschub
\$R or \$r	Zeilenumbruch
\$T or \$t	Tabulator

Beispiele:

```
'Hello'
'Susi und Claus'
':-)'
```

13.1.9 Type Konstanten (Type Literals)

Grundsätzlich wird bei der Verwendung von IEC-Konstanten der kleinstmögliche Datentyp verwendet. Soll ein anderer Datentyp verwendet werden, kann das mit Hilfe von Typed Literals erreicht werden, ohne daß die Konstante explizit deklariert werden muß.

Die Konstante wird mit einem Prefix versehen, das den Typ festlegt.

Schreibweise <Type>#<Literal>

- <Type> gibt den gewünschten Datentyp an. Der Typ muß in Großbuchstaben geschrieben werden.

BOOL, SINT, USINT, BYTE, INT, UINT, WORD, DINT, UDINT, DWORD, REAL, LREAL
- <Literal> gibt die Konstante an. Die Eingabe muß zum unter <Type> angegebenen Datentypen passen.

var1:=DINT#34;

Kann die Konstante nicht ohne Datenverlust in den Zieltyp überführt werden, wird eine Fehlermeldung ausgegeben. Typed Literals können dort verwendet werden, wo normale Konstanten verwendet werden.



Drive PLC Developer Studio

IEC 61131-3 Operanden

13.2 Variablen

Variablen werden entweder lokal im Deklarationsteil eines Bausteins deklariert oder in den globalen Variablenlisten.

- Variablen können überall dort verwendet werden, wo der deklarierte Typ es erlaubt.
- Die verfügbaren Variablen können Sie über die Eingabehilfe abrufen.

13.2.1 Systemvariable

Systemvariable sind implizit deklarierte Variable, die von dem ausgewählten Zielsystem abhängig sind und in der Steuerungskonfiguration des Zielsystems angehängt wurden.

Um herauszufinden, welche Systemvariable das Zielsystem besitzt, wählen Sie den Befehl **Einfügen→Operand**. Im Dialogfeld *Eingabehilfe* wählen Sie dann die Kategorie *Systemvariable*.

13.2.2 Zugriff auf Variablen von Arrays, Strukturen und Bausteinen

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>`

13.2.3 Adressierung von Bits in Variablen

In ganzzahligen Variablen können einzelne Bits angesprochen werden. Dazu wird die Variable mit einem Index des zu adressierenden Bits versehen. Der Bit-Index kann durch eine beliebige Konstante gegeben werden. Die Indizierung ist 0-basiert.

```
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, 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, wird folgende Fehlermeldung ausgegeben.

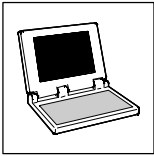
Ein Bit-Zugriff darf nicht einer VAR_IN_OUT-Variablen zugewiesen werden!

13.2.4 Bezeichner

Ein Bezeichner ist eine Folge von Buchstaben, Ziffern und Unterstrichen, die mit einem Buchstaben oder einem Unterstrich beginnt.

Ein Bezeichner einer Variablen darf

- keine Leerstellen und Umlaute enthalten,
- nicht doppelt deklariert werden
- und nicht identisch mit Schlüsselwörtern sein.



Desweiteren gilt:

- Groß-/Kleinschreibung bei Variablen wird **nicht** beachtet.
(Beispiel: VAR1, Var1 und var1 sind keine unterschiedlichen Variablen)
- Unterstriche im Bezeichner sind signifikant.
(Beispiel: A_BCD und AB_CD werden als unterschiedliche Bezeichner interpretiert.)
- Mehrfach aufeinanderfolgende Unterstriche am Anfang eines Bezeichners oder in einem Bezeichner sind **nicht** erlaubt.
- Die ersten 32 Zeichen sind signifikant.

13.3 Adressen

13.3.1 Adresse

Bei der Deklaration können Variablen einem physikalischen Speicherort (SPS-Adresse) mit dem Schlüsselwort **AT** zugeordnet werden.

Der Aufbau der Adreßangabe erfolgt mittels spezieller Zeichenreihen.

Die Zeichenreihe beginnt mit einem Prozentzeichen "%", gefolgt von einem Bereichspräfix und einem Präfix (Datentyp) für die Größe, und endet mit einer Zahlenfolge, die den direkten Speicherort angibt.

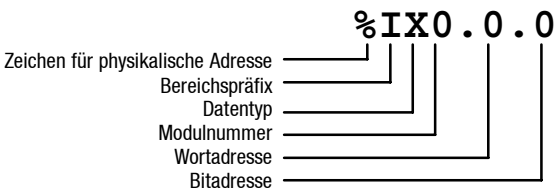
Folgende Bereichspräfixe werden unterstützt:

I	Eingang
Q	Ausgang
M	Merker (interner Speicherbereich)

Folgende Präfixe für die Größe werden unterstützt:

X	Einzelbit
None	optional
B	Byte (8 Bits)
W	Wort (16 Bits)
D	Doppelwort (32 Bits)

Ein-/Ausgänge:



Modulnummer:

Die Modulnummer gibt an, welches Zielsystemmodul (Systembaustein) angesprochen werden soll.

Beispiele:

%QX0.0.2	Ausgangsbit 2
%IW0.0.1	Eingangsbit 1
%MB7	Merkerbyte 7
%MW1	Merkerwort 1
%MD3	Merkerdoppelwort 3
%MX1.2	drittes Merkerbit im Merkerwort 1



Drive PLC Developer Studio

IEC 61131-3 Operanden

Speicher/Adressverwaltung

Die Aufteilung des Speicherbereichs ist wortweise. Ein Wort beinhaltet 16 Bits (Bit0 bis Bit15). Die folgende Tabelle veranschaulicht die Datenstruktur des Speichers:

BIT	BYTE	WORD	DWORD	
0				
1				
2	0			%MX0.2
3				
4				
5				
6				
7				
8		0		
9				
10	1			%MB1
11				
12				
13				
14				
15				
16			0	
17				
18				
19	2			
20				
21				
22				
23		1		
24				
25				
26				%MX1.10
27	3			
28				
29				
30				
31				



Hinweis!

Boolesche Werte werden byteweise zugewiesen, falls nicht explizit eine Einzelbitadresse angegeben wird. Eine Wertänderung von varbool1 AT%MW0 betrifft den Bereich von %MX0.0 bis %MX0.7

13.3.2

Merker



Tip!

Die Größe des Merkbereiches hängt vom ausgewählten Zielsystem ab.

Beispiel1:

%MD48

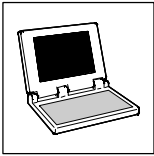


adressiert im Merkerbereich die Bytes Nr. 192, 193, 194 und 195 ($48 * 4 = 192$). Das erste Byte ist das Byte Nr. 0.

Beispiel2:

%MX5.0

adressiert im Merkerbereich das erste Bit im fünften Wort. (Bits werden in der Regel wortweise abgelegt.)



13.4 Funktionsaufrufe

Im ST kann auch ein Funktionsaufruf als Operand auftreten.

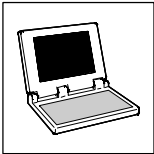
Beispiel:

```
Ergebnis := Fct(7) + 3;
```



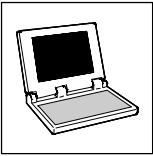
Drive PLC Developer Studio

IEC 61131-3 Operanden



14 IEC 61131-3 Standard-Funktionen

Typkonvertierungsfunktionen	
Konvertierungen zwischen ganzzahligen Zahlentypen	
BOOL_TO	BOOL → Typ X
TO_BOOL	Typ X → BOOL
TIME_TO / TIME_OF_DAY	TIME / TIME_OF_DAY → Typ X
DATE_TO / DT_TO	DATE / DATE_AND_TIME → Typ X
STRING_TO	STRING → Typ X
TRUNC	REAL → INT
Numerische Funktionen	
ABS	Absolutwert
SQRT	Quadratwurzel
LN	natürlicher Logarithmus
LOG	Logarithmus zur Basis 10
EXP	Exponentialfunktion
SIN	Sinusberechnung in rad
COS	Cosinusberechnung in rad
TAN	Tangensberechnung in rad
ASIN	Arcussinusberechnung in rad
ACOS	Arcuscosinusberechnung in rad
ATAN	Arcustangensberechnung in rad
EXPT	Potenzierung einer Variablen mit einer anderen
STRING-Funktionen	
LEN	gibt die Länge eines Strings aus
LEFT	liefert einen linken Anfangsstring eines Strings
RIGHT	liefert einen rechten Anfangsstring eines Strings
MID	liefert einen Teilstring eines Strings
CONCAT	Konkatenation (Aneinanderhängen) von zwei Strings
INSERT	fügt einen String ab einer bestimmten Stelle in einen anderen ein
DELETE	löscht ab einer bestimmten Stelle einen Teilstring aus einem String
REPLACE	ersetzt einen Teilstring eines Strings durch einen anderen String
FIND	sucht einen Teilstring in einem String
Bistabile Funktionsblöcke	
SR	Bistabiler Funktionsblock (dominant setzen)
RS	Bistabiler Funktionsblock (dominant zurücksetzen)
SEMA	Software-Semaphor (unterbrechbar)
Flankenerkennung	
R_TRIG	Detektor für eine ansteigende Flanke
F_TRIG	Detektor für eine fallende Flanke
Zähler	
CTU	Aufwärtszähler
CTD	Abwärtszähler
CTUD	Auf- und Abwärtszähler
Timer	
TP	Pulsgeber
TON	Timer on-delay
TOF	Timer off-delay
RTC*	Laufzeituhr*
* wird nicht unterstützt.	



Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

14.1 Typkonvertierungsfunktionen

Es ist nicht erlaubt, von einem "größeren" Typ auf einen "kleineren" Typ implizit zu konvertieren (beispielsweise von **INT** nach **BYTE** oder von **DINT** nach **WORD**). Wenn Sie dies tun möchten, müssen Sie spezielle Typkonvertierungsfunktionen anwenden.

Grundsätzlich kann von jedem elementaren Typ zu jeden anderen elementaren Typ konvertiert werden.

Syntax:

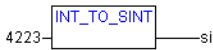
```
<elem.Typ1>_TO_<elem.Typ2>
```

14.1.1 Konvertierungen zwischen ganzzahligen Zahlentypen

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.

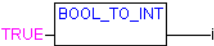
Beispiele		
AWL LD 4223 INT_TO_SINT ST si	ST si:=INT_TO_SINT(4223); (* si=127 *)	FUP 

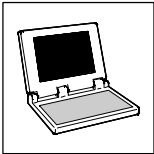
Wenn Sie die Integerzahl 4223 (16#107f in Hexadezimaldarstellung) in eine **SINT**-Variable speichern, dann enthält diese die Zahl 127 (16#7f in Hexadezimaldarstellung).

14.1.2 BOOL_TO

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		
AWL LD TRUE BOOL_TO_INT ST i LD TRUE BOOL_TO_TOD ST tof1	ST LD TRUE BOOL_TO_STRING ST str LD FALSE BOOL_TO_DATE ST dat	FUP LD TRUE BOOL_TO_TIME ST t LD TRUE BOOL_TO_DT ST dandt
ST i:=BOOL_TO_INT(TRUE); (* i=1 *) str:=BOOL_TO_STRING(TRUE); (* str='TRUE' *) t:=BOOL_TO_TIME(TRUE); (* t=T#1ms *) tof1:=BOOL_TO_TOD(TRUE); (* tof1=TOD#00:00:00.001 *) dat:=BOOL_TO_DATE(FALSE); (* dat=D#1970-01-01 *) dandt:=BOOL_TO_DT(TRUE); (* dandt=DT#1970-01-01-00:00:01 *)		
FUP 		



14.1.3 TO_BOOL

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		
AWL		
LD 213 BYTE_TO_BOOL ST b (* b=TRUE *)	LD 0 INT_TO_BOOL ST b (* b=FALSE *)	LD T#5ms TIME_TO_BOOL ST b (* b=TRUE *)
LD 'TRUE' STRING_TO_BOOL ST b (* b=TRUE *)		
ST		
b:=BYTE_TO_BOOL(2#11010101); (* b=TRUE *) b:=INT_TO_BOOL(0); (* b=FALSE *) b:=TIME_TO_BOOL(T#5ms); (* b=TRUE *) b:=STRING_TO_BOOL('TRUE'); (* b=TRUE *)		
FUP		

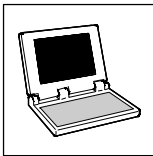
14.1.4 TIME_TO / TIME_OF_DAY

Konvertierung vom Typ *TIME* bzw. *TIME_OF_DAY* zu einem anderen Typ

Intern wird die Zeit als ein **DWORD** in Millisekunden abgespeichert (bei **TIME_OF_DAY** seit 00:00 Uhr). Dieser Wert wird konvertiert.

- Beim Typ **STRING** ist das Ergebnis die Zeitkonstante.
- Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen.

Beispiele		
AWL		
LD T#12ms TIME_TO_STRING ST str	LD T#300000ms TIME_TO_DWORD ST dw	LD TOD#00:00:00.012 TOD_TO_SINT ST si
ST		
str:=TIME_TO_STRING(T#12ms); (* str='T#12ms' *) dw:=TIME_TO_DWORD(T#5m); (* dw=300000 *) si:=TOD_TO_SINT(TOD#00:00:00.012); (* si=12 *)		
FUP		



Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

14.1.5 DATE_TO / DT_TO

Konvertierung vom Typ *DATE* bzw. *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.

- Beim Typ **STRING** ist das Ergebnis die Datumskonstante.
- Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen.

Beispiele		
AWL		
LD D#1970-01-01 DATE_TO_BOOL ST b	LD D#1970-01-15 DATE_TO_INT ST i	LD DT#1970-01-15-05:05:05 DT_TO_BYTE ST byt
LD DT#1998-02-13-14:20 DT_TO_STRING ST str		
ST		
b:=DATE_TO_BOOL(D#1970-01-01); (* b=FALSE *) i:=DATE_TO_INT(D#1970-01-15); (* i=29952 *) byt:=DT_TO_BYTE(DT#1970-01-15-05:05:05); (* byt=129 *) str:=DT_TO_STRING(DT#1998-02-13-14:20); (* str='DT#1998-02-13-14:20' *)		
FUP		

14.1.6 STRING_TO

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		
AWL		
LD 'TRUE' STRING_TO_BOOL ST b	LD '123' STRING_TO_WORD ST w	LD 't#127ms' STRING_TO_TIME ST t
ST		
b:=STRING_TO_BOOL('TRUE'); (* b=TRUE *) w:=STRING_TO_WORD('123'); (* w=123 *) t:=STRING_TO_TIME('T#127ms'); (* t=T#127ms *)		

14.1.7 TRUNC

Konvertierung vom Typ *REAL* zum Typ *INT*

- Es wird der ganzzahlige Anteil der Gleitkommazahl genommen.
- Bei der Typkonvertierung von größere auf kleinere Typen können Informationen verloren gehen.

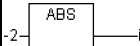
Beispiele		
AWL		ST
LD 1.9 TRUNC ST i (* i=1 *)	LD -1.4 TRUNC ST i (* i=-1 *)	i:=TRUNC(1.9); i:=TRUNC(-1.4);



14.2 Numerische Funktionen

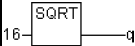
14.2.1 ABS

Liefert den Absolutwert einer Zahl

Beispiele		
AWL LD -2 ABS ST i (* i=2 *)	ST i:=ABS(-2);	FUP 

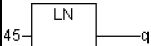
14.2.2 SQRT

Liefert die Quadratwurzel einer Zahl

Beispiele		
AWL LD 16 SQRT ST q (* q=4 *)	ST q:=SQRT(16);	FUP 

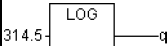
14.2.3 LN

Liefert den natürlichen Logarithmus einer Zahl

Beispiele		
AWL LD 45 LN ST q (* q=3.806663 *)	ST q:=LN(45);	FUP 

14.2.4 LOG

Liefert den Logarithmus zur Basis 10 einer Zahl

Beispiele		
AWL LD 314.5 LOG ST q (* q=2.497621 *)	ST q:=LOG(314.5);	FUP 

14.2.5 EXP

Liefert die Exponentialfunktion

Beispiele		
AWL LD 23 EXP ST q (* q=9.744791e+009 *)	ST q:=EXP(23);	FUP 

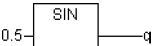


Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

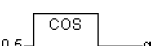
14.2.6 SIN

Liefert den Sinus einer Zahl in rad

Beispiele		
AWL LD 0.5 SIN ST q (* q=0.4794255 *)	ST q:=SIN(0.5);	FUP 

14.2.7 COS

Liefert den Cosinus einer Zahl in rad

Beispiele		
AWL LD 0.5 COS ST q (* q=0.8775826 *)	ST q:=COS(0.5);	FUP 

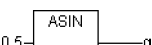
14.2.8 TAN

Liefert den Tangens einer Zahl in rad

Beispiele		
AWL LD 0.5 TAN ST q (* q=0.5463024 *)	ST q:=TAN(0.5);	FUP 

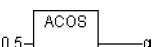
14.2.9 ASIN

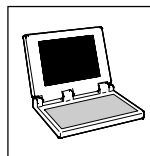
Liefert den Arcussinus (Umkehrfunktion von Sinus) einer Zahl in rad

Beispiele		
AWL LD 0.5 ASIN ST q (* q=0.5235988 *)	ST q:=ASIN(0.5);	FUP 

14.2.10 ACOS

Liefert den Arcuscosinus (Umkehrfunktion von Cosinus) einer Zahl in rad

Beispiele		
AWL LD 0.5 ACOS ST q (* q=1.047198 *)	ST q:=ACOS(0.5);	FUP 



14.2.11 ATAN

Liefert den Arcustangens (Umkehrfunktion von Tangens) einer Zahl in rad

Beispiele		
AWL LD 0.5 ATAN ST q (* q=0.4636476 *)	ST q:=ATAN(0.5);	FUP

14.2.12 EXPT

Potenzierung einer Variablen mit einer anderen:

OUT = IN1^{IN2}

- OUT ist vom Typ **REAL**.
- IN1 und IN2 können vom Typ **BYTE, WORD, DWORD, INT, DINT, REAL** sein.

Beispiele		
AWL LD 7 EXPT 2 ST var1 (* var1=49 *)	ST var1:=EXPT(7,2);	FUP

14.3 STRING-Funktionen



Hinweis!

Die zulässige Länge eines Strings hängt vom verwendeten Zielsystem ab. Wird die Länge überschritten, erscheint im Online-Modus eine Fehlermeldung.

14.3.1 LEN

Gibt die Länge eines Strings aus

- STR ist vom Typ **STRING**, der Rückgabewert der Funktion vom Typ **INT**.

Beispiele		
AWL LD 'SUSI' LEN ST Var1 (* Var1=4 *)	ST Var1:=LEN('SUSI');	FUP

14.3.2 LEFT

Liefert einen linken Anfangsstring eines Strings

- STR ist vom Typ **STRING**, SIZE vom Typ **INT**, der Rückgabewert der Funktion vom Typ **STRING**.
- LEFT (STR, SIZE) bedeutet: Nehme die ersten SIZE Zeichen von links im String STR.

Beispiele		
AWL LD 'SUSI' LEFT 3 ST Var1 (* Var1='SUS' *)	ST Var1:=LEFT('SUSI',3);	FUP



Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

14.3.3 RIGHT

Liefert einen rechten Anfangsstring eines Strings

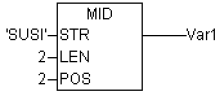
- STR ist vom Typ **STRING**, SIZE vom Typ **INT**, der Rückgabewert der Funktion vom Typ **STRING**.
- RIGHT (STR, SIZE) bedeutet: Nehme die ersten SIZE Zeichen von rechts im String STR.

Beispiele		
AWL LD 'SUSI' RIGHT 3 ST Var1 (* Var1='USI' *)	ST Var1:=RIGHT('SUSI',3);	FUP 

14.3.4 MID

Liefert einen Teilstring eines Strings

- STR ist vom Typ **STRING**, LEN und POS vom Typ **INT**, der Rückgabewert der Funktion vom Typ **STRING**.
- MID (STR, LEN, POS) bedeutet: Hole LEN Zeichen aus dem String STR, beginnend mit dem Zeichen an der Stelle POS.

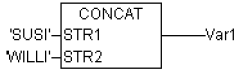
Beispiele		
AWL LD 'SUSI' MID 2,2 ST Var1 (* Var1='US' *)	ST Var1:=MID('SUSI',2,2);	FUP 

14.3.5 CONCAT

Concatenation (Verketten)

Aneinanderhängen von zwei Strings

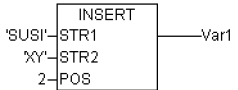
- STR1, STR2 und der Rückgabewert der Funktion sind vom Typ **STRING**.

Beispiele		
AWL LD 'SUSI' CONCAT 'WILLI' ST Var1 (* Var1='SUSIWILLI' *)	ST Var1:=CONCAT ('SUSI','WILLI');	FUP 

14.3.6 INSERT

Fügt einen String ab einer bestimmten Stelle in einen anderen ein

- STR1 und STR2 sind vom Typ **STRING**, POS vom Typ **INT**, der Rückgabewert der Funktion vom Typ **STRING**.
- INSERT (STR1, STR2, POS) bedeutet: Füge STR2 in STR1 nach der POS-ten Stelle ein.

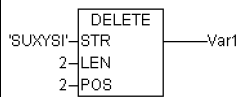
Beispiele		
AWL LD 'SUSI' INSERT 'XY',2 ST Var1 (* Var1='SUXYSI' *)	ST Var1 := INSERT ('SUSI','XY',2);	FUP 



14.3.7 DELETE

Löscht ab einer bestimmten Stelle einen Teilstring aus einem String

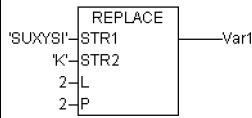
- STR1 ist vom Typ **STRING**, LEN und POS vom Typ **INT**, der Rückgabewert der Funktion vom Typ **STRING**.
- DELETE (STR, L, P) bedeutet: Lösche L Zeichen aus STR, beginnend mit dem P-ten.

Beispiele		
AWL LD 'SUXYSI' DELETE 2,2 ST Var1 (* Var1='SUSI' *)	ST Var1:=DELETE ('SUXYSI' , 2, 2);	FUP 

14.3.8 REPLACE

Ersetzt einen Teilstring eines Strings durch einen anderen String

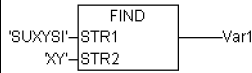
- STR1 und STR2 sind vom Typ **STRING**, LEN und POS vom Typ **INT**, der Rückgabewert der Funktion vom Typ **STRING**.
- REPLACE (STR1, STR2, L, P) bedeutet: Ersetze L Zeichen aus STR1 durch STR2 beginnend mit dem P-ten Zeichen.

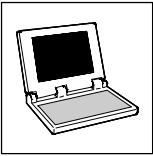
Beispiele		
AWL LD 'SUXYSI' REPLACE 'K',2,2 ST Var1 (* Var1='SKYSI' *)	ST Var1:=REPLACE ('SUXYSI' , 'K' , 2, 2);	FUP 

14.3.9 FIND

Sucht einen Teilstring in einem String

- STR1 und STR2 sind vom Typ **STRING**, der Rückgabewert der Funktion vom Typ **INT**.
- FIND (STR1, STR2) bedeutet: Finde die Position des ersten Zeichens des ersten Vorkommens von STR2 in STR1.
- Wenn STR2 in STR1 nicht vorkommt, dann gilt Rückgabewert:=0.

Beispiele		
AWL LD 'SUXYSI' FIND 'XY' ST Var1 (* Var1=3 *)	ST Var1:=FIND ('SUXYSI' , 'XY');	FUP 



Drive PLC Developer Studio

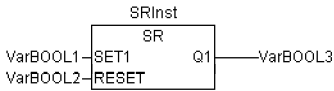
IEC 61131-3 Standard-Funktionen

14.4 Bistabile Funktionsblöcke

14.4.1 SR

Bistabiler Funktionsblock (dominant setzen)

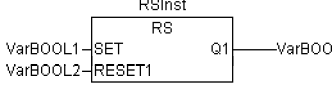
- Q1, SET1 und RESET sind vom Typ **BOOL**.
- $Q1 = SR (SET1, RESET)$ bedeutet: $Q1 = (NOT\ RESET\ AND\ Q1)\ OR\ SET1$

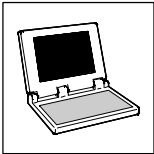
Beispiele	
Deklaration: SRInst : SR;	
AWL CAL SRInst (SET1:=VarBOOL1, RESET:=VarBOOL2) LD SRInst.Q1 ST VarBOOL3	FUP 
ST SRInst (SET1:=VarBOOL1, RESET:=VarBOOL2); VarBOOL3:=SRInst.Q1;	

14.4.2 RS

Bistabiler Funktionsblock (dominant zurücksetzen)

- Q1, SET und RESET1 sind vom Typ **BOOL**.
- $Q1 = RS (SET, RESET1)$ bedeutet: $Q1 = NOT\ RESET1\ AND\ (Q1\ OR\ SET)$

Beispiele	
Deklaration: RSInst : RS;	
AWL CAL RSInst (SET:=VarBOOL1, RESET1:=VarBOOL2) LD RSInst.Q1 ST VarBOOL3	FUP 
ST RSInst (SET:=VarBOOL1, RESET1:=VarBOOL2); VarBOOL3:=RSInst.Q1;	



14.4.3 SEMA

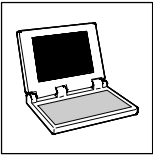
Software-Semaphor (unterbrechbar)
X ist eine interne BOOL-Variable, die mit FALSE initialisiert ist.
BUSY, CLAIM und RELEASE sind vom Typ BOOL.

- Wenn SEMA aufgerufen wird und BUSY ist TRUE, dann bedeutet das, daß SEMA bereits vorher belegt wurde (SEMA wurde mit CLAIM = TRUE aufgerufen).
- Wenn BUSY FALSE ist, dann wurde SEMA noch nicht aufgerufen, oder es wurde freigegeben (Aufruf mit RELEASE = TRUE).

BUSY = SEMA (CLAIM, RELEASE) bedeutet:

```
BUSY:=X;  
IF CLAIM THEN X:=TRUE;  
ELSIF RELEASE THEN BUSY:=FALSE; X:=FALSE;  
END_IF
```

Beispiele	
Deklaration: SEMAInst : SEMA;	
AWL CAL SEMAInst (CLAIM:=VarBOOL1, RELEASE:=VarBOOL2) LD SEMAInst.BUSY ST VarBOOL3	FUP <pre>graph LR subgraph SEMAInst [SEMAInst] SEMA end VarBOOL1 -- CLAIM --> SEMAInst VarBOOL2 -- RELEASE --> SEMAInst SEMAInst -- BUSY --> VarBOOL3</pre>
ST SEMAInst (CLAIM:=VarBOOL1, RELEASE:=VarBOOL2); VarBOOL3:=SEMAInst.BUSY;	



14.5 Flankenerkennung

14.5.1 R_TRIG

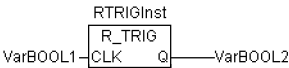
Detektor für eine ansteigende Flanke

```
FUNCTION_BLOCK R_TRIG
VAR_INPUT
    CLK : BOOL;
END_VAR
VAR_OUTPUT
    Q : BOOL;
END_VAR
VAR
    M : BOOL := FALSE;
END_VAR
    Q := CLK AND NOT M;
    M := CLK;
END_FUNCTION_BLOCK
```

Solange die Eingabevariable CLK **FALSE** liefert, solange werden die Ausgabe Q und die Hilfsvariable M **FALSE** sein.

Sobald CLK **TRUE** liefert, wird zuerst Q **TRUE** liefern, und dann M auf **TRUE** gesetzt.

D.h.: bei jedem weiteren Aufruf der Funktion wird Q wieder **FALSE** liefern, bis CLK eine fallende und wieder eine steigende Flanke hat.

Beispiele	
Deklaration: RTRIGInst : R_TRIG;	
AWL CAL RTRIGInst (CLK:=VarBOOL1) LD RTRIGInst.Q ST VarBOOL2	FUP 
ST RTRIGInst (CLK:=VarBOOL1); VarBOOL2:=RTRIGInst.Q;	



14.5.2 F_TRIG

Detektor für eine fallende Flanke

```
FUNCTION_BLOCK F_TRIG
VAR_INPUT
    CLK : BOOL;
END_VAR
VAR_OUTPUT
    Q : BOOL;
END_VAR
VAR
    M : BOOL := TRUE;
END_VAR
    Q := NOT CLK AND NOT M;
    M := NOT CLK;
END_FUNCTION_BLOCK
```

Solange die Eingabevariable CLK **TRUE** liefert, solange werden die Ausgabe Q und die Hilfsvariable M **FALSE** sein.

Sobald s1 **FALSE** liefert, wird zuerst Q **TRUE** liefern, und dann M auf **TRUE** gesetzt.

D.h.: bei jedem weiteren Aufruf der Funktion wird Q wieder **FALSE** liefern, bis CLK eine steigende und wieder eine fallende Flanke hat.

Beispiele	
Deklaration: FTRIGInst : F_TRIG;	
AWL CAL FTRIGInst (CLK:=VarBOOL1) LD FTRIGInst.Q ST VarBOOL2	FUP <pre>graph LR VarBOOL1 -- CLK --> FTRIGInst[FTRIGInst] FTRIGInst -- Q --> VarBOOL2</pre>
ST FTRIGInst (CLK:=VarBOOL1); VarBOOL2:=FTRIGInst.Q;	



Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

14.6 Zähler

14.6.1 CTU

Aufwärtszähler

CU, RESET und Q sind vom Typ **BOOL**, PV und CV sind vom Typ **INT**.

- Wenn RESET **TRUE** ist, wird die Zählvariable CV mit 0 besetzt.
- Mit jeder positiven Flanke am Eingang CU wird CV um 1 erhöht. Solange CV kleiner als PVmax ist (Wenn also kein Überlauf verursacht wird).
- Q liefert **TRUE**, wenn CV größer oder gleich der Obergrenze PV ist.

CTU(CU, RESET, PV, Q, CV) bedeutet:

- Wenn RESET **TRUE** ist, dann wird die Zählvariable CV mit 0 initialisiert.
- Wenn CU eine steigende Flanke hat, dann wird CV um 1 erhöht.

Beispiele	
Deklaration: CTUInst : CTU;	
AWL CAL CTUInst (CU:=VarBOOL1, RESET:=VarBOOL2, PV:=VarINT1) LD CTUInst.Q ST VarBOOL3 LD CTUInst.CV ST VarINT2	FUP
ST CTUInst (CU:=VarBOOL1, RESET:=VarBOOL2, PV:=VarINT1); VarBOOL3:=CTUInst.Q; VarINT2:=CTUInst.CV;	

14.6.2 CTD

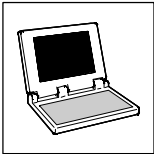
Abwärtszähler

CD, LOAD und Q sind vom Typ **BOOL**, PV und CV sind vom Typ **INT**.

- Wenn LOAD **TRUE** ist, wird die Zählvariable CV mit der Obergrenze PV gleichgesetzt.
- Wenn LOAD **FALSE** ist, dann wird mit jeder positiven Flanke an CD, CV um 1 erniedrigt. In jedem Aufruf des Funktionsblocks, CV um 1 erniedrigt.
- Q liefert **TRUE**, wenn CV kleiner oder gleich 0 ist.

CTD(CD, LOAD, PV, Q, CV)

Beispiele	
Deklaration: CTDInst : CTD;	
AWL CAL CTDInst (CD:=VarBOOL1, LOAD:=VarBOOL2, PV:=VarINT1) LD CTDInst.Q ST VarBOOL3 LD CTDInst.CV ST VarINT2	FUP
ST CTDInst (CD:=VarBOOL1, LOAD:=VarBOOL2, PV:=VarINT1); VarBOOL3:=CTDInst.Q; VarINT2:=CTDInst.CV;	



14.6.3 CTUD

Auf- und Abwärtszähler

CU, CD, RESET, LOAD, QU und QD sind vom Typ **BOOL**, PV und CV sind vom Typ **INT**.

- Wenn **RESET** gilt, dann wird die Zählvariable **CV** mit 0 initialisiert.
- Wenn **LOAD** gilt, dann wird **CV** mit **PV** initialisiert.
- Wenn positive Flanke **CU** gilt, dann wird **CV** um 1 erhöht, solange **CV** keinen Überlauf verursacht.
- Wenn positive Flanke **CD** gilt, dann wird **CV** jeweils um 1 erniedrigt, solange **CV** keinen Unterlauf verursacht.
- **QU** liefert **TRUE**, wenn **CV** größer oder gleich **PV** geworden ist.
- **QD** liefert **TRUE**, wenn **CV** kleiner oder gleich 0 geworden ist.

CTUD (CU, CD, RESET, LOAD, PV, QU, QD, CV)

Beispiele	
Deklaration: CTUDInst : CTUD;	
AWL CAL CTUDInst (CU:=VarBOOL2, RESET:=VarBOOL32, LOAD:=VarBOOL4; PV:=VarINT1) LD CTUDInst.QU ST VarBOOL5 LD CTUDInst.QD ST VarBOOL6 LD CTUDInst.CV ST VarINT2	FUP
ST CTUDInst (CU:=VarBOOL2, RESET:=VarBOOL32, LOAD:=VarBOOL4; PV:=VarINT1); VarBOOL5:=CTUDInst.QU; VarBOOL6:=CTUDInst.QD; VarINT2:=CTUDInst.CV;	



Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

14.7 Timer

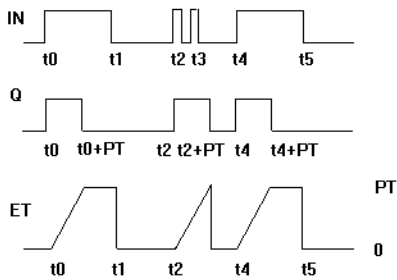
14.7.1 TP

Pulsgeber

TP (IN, PT, Q, ET) bedeutet:

- IN und PT sind Eingabevariablen vom Typ **BOOL** bzw. **TIME**.
- Q und ET sind Ausgabevariablen vom Typ **BOOL** bzw. **TIME**.
- Nach einer positiven Flanke am Eingang IN, wird der Ausgang Q für die Zeit an PT **TRUE**. Anschließend ist Q wieder **FALSE**.
- Sobald IN **TRUE** ist, wird in ET die Zeit in Millisekunden hochgezählt, bis der Wert gleich dem in PT ist, dann bleibt er gleich.
- Q liefert somit für den in PT angegebenen Zeitraum ein Signal.

Graphische Darstellung des zeitlichen Ablaufs von TP:



Beispiele	
Deklaration:	
TPInst : TP;	
AWL	FUP
CAL TPInst (IN:=VarBOOL1, PT:=T#5s)	
LD TPInst.Q	
ST VarBOOL2	
ST	
TPInst (IN:=VarBOOL1, PT:=T#5s);	
VarBOOL2:=TPInst.Q;	



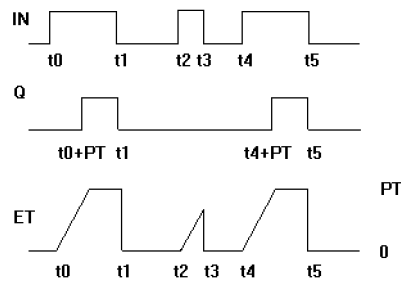
14.7.2 TON

Timer on-delay

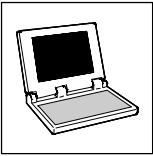
TON(IN, PT, Q, ET) bedeutet:

- IN und PT sind Eingabevariablen vom Typ **BOOL** bzw. **TIME**.
 - Q und ET sind Ausgabevariablen vom Typ **BOOL** bzw. **TIME**.
 - Wenn IN **FALSE** ist, sind die Ausgaben **FALSE** bzw. 0.
 - Sobald IN **TRUE** ist, wird in ET die Zeit in Millisekunden hochgezählt, bis der Wert gleich dem in PT ist, dann bleibt er gleich.
 - Q ist **TRUE** wenn IN **TRUE** und ET gleich PT ist. Andernfalls ist Q **FALSE**.
- Q hat somit eine steigende Flanke, wenn die in PT in Millisekunden angegebene Zeit abgelaufen ist.

Graphische Darstellung des zeitlichen Verhaltens von TON:



Beispiele	
Deklaration: TONInst : TON;	
AWL CAL TONInst (IN:=VarBOOL1, PT:=T#5s) LD TONInst.Q ST VarBOOL2	FUP
ST TONInst (IN:=VarBOOL1, PT:=T#5s); VarBOOL2:=TONInst.Q;	



Drive PLC Developer Studio

IEC 61131-3 Standard-Funktionen

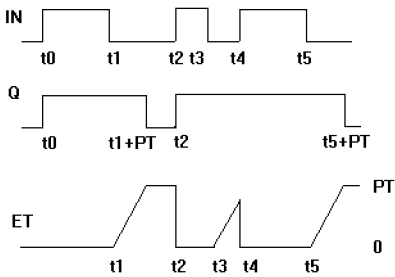
14.7.3 TOF

Timer off-delay

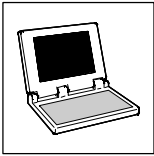
TOF(IN, PT, Q, ET) bedeutet:

- IN und PT sind Eingabevariablen vom Typ **BOOL** bzw. **TIME**.
 - Q und ET sind Ausgabevariablen vom Typ **BOOL** bzw. **TIME**.
 - Sobald IN **FALSE** ist, wird in ET die Zeit in Millisekunden hochgezählt, bis der Wert gleich dem in PT ist, dann bleibt er gleich.
 - Q ist **FALSE** wenn IN **FALSE** und ET gleich PT ist. Andernfalls ist Q **TRUE**.
- Q hat somit eine fallende Flanke, wenn die in PT in Millisekunden angegebene Zeit abgelaufen ist.

Graphische Darstellung des zeitlichen Verhaltens von TOF:



Beispiele	
Deklaration: TOFInst : TOF;	
AWL CAL TOFInst (IN:=VarBOOL1, PT:=T#5s) LD TOFInst.Q ST VarBOOL2	FUP
ST TOFInst (IN:=VarBOOL1, PT:=T#5s); VarBOOL2:=TOFInst.Q;	



15 Anhang

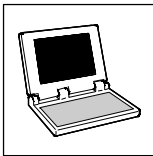
15.1 Kommandozeilen Befehle

Es besteht die Möglichkeit dem DDS 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.

/online	DDS versucht mit dem aktuellen Projekt nach dem Start online zu gehen.
/run	DDS startet nach dem Einloggen das Anwenderprogramm. Nur gültig in Verbindung mit /online.
/show /show hide /show icon /show max /show normal	Die Darstellung des DDS-Frame-Windows kann gesetzt werden. Das Fenster wird nicht angezeigt und erscheint nicht in der Task-Leiste. Das Fenster wird minimiert angezeigt. Das Fenster wird maximiert angezeigt. Das Fenster wird in dem Zustand angezeigt, den es beim letzten Schließen hatte.
/out <outfile>	Alle Meldungen werden in das Meldungsfenster und 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 DDS-Exe-Datei>" "<Pfad des Projekts>" / <Befehl1> / <Befehl2> ...

DDS-Datei *.pro wird geöffnet, das Fenster wird nicht angezeigt. Der Inhalt der Kommandodatei (cmdfile) command.cmd wird abgearbeitet.



Drive PLC Developer Studio

Anhang

15.2 Kommandodatei (Cmdfile) 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 in der Meldungsdatei (siehe unten) ausgegeben, außer dem Befehl wird ein " @" vorangestellt. Alle Zeichen nach einem Semikolon (;) werden ignoriert (Kommentar). Enthalten Parameter Leerzeichen, so müssen diese Parameter in Anführungszeichen angegeben werden.

Befehle des Online Menüs	
online login	Einloggen mit dem geladenen Projekt (Online→Einloggen)
online logout	Ausloggen (Online→Ausloggen)
online run	Starten des Anwenderprogramms (Online→Start)
online sim	Anschalten der Simulation ((✓) Online→Simulation)
online sim off	Ausschalten der Simulation (Online→Simulation)

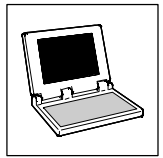
Befehle des Datei Menüs	
file new	Es wird ein neues Projekt angelegt (Datei→Neu)
file open <projectfile>	Es wird das angegebene Projekt geladen (Datei→Öffnen)
file close	Das geladene Projekt wird geschlossen Datei→Schließen)
file save	Das geladene Projekt wird gespeichert Datei→Speichern)
file saveas <projectfile>	Das geladene Projekt wird unter dem angegebenen Namen gespeichert (Datei→Speichern unter)
file quit	DDS wird beendet (Datei→Beenden)

Befehle des Projekt Menüs	
project compile oder project build	Das geladene Projekt wird inkrementell übersetzt (Projekt→Übersetzen)
project rebuild	Das geladene Projekt wird komplett übersetzt (Projekt→Alles Übersetzen)
project clean	Die Übersetzungsinformation und die Download-Informationen im aktuellen Projekt werden gelöscht (Projekt→Alles bereinigen)
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 <dir>	Jedes Objekt des geladenen Projekts wird in das Verzeichnis <dir> in eine separate Datei exportiert, die jeweils den Namen des Objekts trägt

Steuerung der Meldungsdatei	
out open <msgfile>	Öffnet die angegebene Datei als Meldungs Ausgabe. Neue Meldungen werden angehängt
out close	Schließt die derzeit geöffnete Meldungsdatei
out clear	Löscht alle Meldungen aus der derzeit geöffneten Meldungsdatei

Steuerung der Meldungs-Ausgaben	
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

Steuerung von Ersetzen von Objekten bzw. Dateien bei Import, Export, Ersetzen	
replace ok	
replace yes	Ersetzen
replace no	Nicht ersetzen
replace yes all	Alle ersetzen
replace no all	Nichts ersetzen



Steuerung des Default-Verhaltens von DDS-Dialogen	
query on	Dialoge werden angezeigt und erwarten Benutzereingaben
query of ok	Alle Dialoge verhalten sich entsprechend, dem anklicken von OK
query off no	Alle Dialoge verhalten sich entsprechend, dem anklicken von Nein
query off cancel	Alle Dialoge verhalten sich entsprechend, dem anklicken von Abbruch

Befehl zum Aufruf von Kommando-Dateien als Unterprogramme	
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.

Verwendete Verzeichnisse von DDS setzen	
dir lib <libdir>	Setzt <libdir> als Bibliotheksverzeichnis
dir compile <compiledir>	Setzt <compiledir> als Verzeichnis für die Übersetzungsdateien

Verzögerung der Abarbeitung des CMDFILES	
delay 5000	Wartet 5 Sekunden (Genauigkeit der Ausführung in 100ms-Schritten)

Steuerung des Watch- und Rezepturverwalters	
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 vorhandenen Werten (Extras→Rezeptur schreiben)

Einbinden von Bibliotheken	
library add <Bibliotheksdatei 1>, <Bibliotheksdatei 2>, ... <Bibliotheksdatei n>	Hängt die angegebenen Bibliotheksdateien an die Bibliotheksliste des aktuell geöffneten Projekts an. Handelt es sich beim Pfad der Datei um einen relativen Pfad, dann wird das im Projekt eingestellte Bibliotheksverzeichnis als Wurzel des Pfads eingesetzt.
library delete <Bibliothek 1> <Bibliothek 2> ... <Bibliothek n>]	Löscht alle Bibliotheken, bzw. die angegebene, aus der Bibliotheksliste des aktuell geöffneten Projekts.

Kopieren von Objekten	
object copy <Quellprojektdat> <Quellpfad> <Zielpfad>	Kopiert Objekte aus dem angegebenen Pfad der Quellprojektdat in den Zielpfad des gerade geöffneten Projekts. Ist der Quellpfad der Name eines Objektes, wird dieses kopiert. Handelt es sich um einen Ordner, werden alle Objekte unterhalb dieses Ordners kopiert. In diesem Fall wird die Ordnerstruktur unterhalb des Quellordners mit übernommen. Existiert der Zielpfad noch nicht, wird er erstellt.

Einstellen der Kommunikationsparameter (Gateway, Gerät)	
gateway local	Setzt den Gateway des lokalen Rechners als aktuellen Gateway
gateway tcpip <Adresse> <Port>	Setzt den im angegebenen Remote-Rechner eingestellten Gateway als aktuellen Gateway. <Adresse>: Tcp/Ip-Adresse oder Host-Name des Remote-Rechners <Port>: Tcp/Ip-Port des Remote-Gateway Achtung: Es können nur Gateways erreicht werden, die kein Passwort gesetzt haben!
device guid <guid>	Setzt das Gerät mit der angegebenen GUID als aktuelles Gerät. Die GUID muß folgendem Format entsprechen: {01234567-0123-0123-0123-0123456789ABC} Die geschweiften Klammern und die Bindestriche müssen an den angegebenen Positionen stehen.
device name <devicename>	Setzt das Gerät mit dem angegebenen Namen als aktuelles Gerät.



Drive PLC Developer Studio

Anhang

Einstellen der Kommunikationsparameter (Gateway, Gerät)	
device instance <Instanzname>	Setzt den Instanznamen für das aktuelle Gerät auf den angegebenen Namen.
device parameter <Id> <paramtername> <Wert>	Weist dem Parameter mit der angegeben Id oder optional dem angegebenen Parameternamen den angegebenen Wert zu, der dann vom Gerät interpretiert wird.

Systemaufruf	
system <Befehl>	Führt den angegeben Betriebssystembefehl aus.

Zielsystem auswählen	
target <Id> <name>	Setzt die Zielplattform für das aktuelle Projekt. Angabe der Target-Id oder optional des Target-Namens, wie in der Targetdatei angegeben

Kommandodatei command.cmd

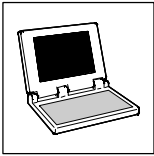
```
file open C:\projects\DDS_test\ampel.pro
query off ok
watchlist load c:\work\w.wtc
online login
online run
delay 1000
watchlist read
watchlist save c:\work\watch.wtc
online logout
file close
```

Diese Kommandodatei

- öffnet die Projektdatei ampel.pro
- lädt eine unter w.wtc geladene Watchliste
- startet das Anwenderprogramm
- schreib nach einer Sekunde die Variablenwerte in die Watchliste watch.wtc, die gespeichert wird
- schließt das Projekt danach wieder

Die Kommandodatei wird in der Kommandozeile folgendermaßen aufgerufen.

"<Pfad DDS-Datei>" "<Pfad cmd-Datei>"



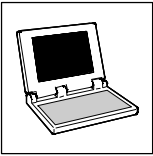
15.3 IEC-Schlüsselwörter

Schlüsselwörter sind eindeutige Kombinationen von Zeichen, die als individuelle Syntax-Elemente angewendet werden.

- Schlüsselwörter dürfen nicht als Bezeichner verwendet werden.
- Zu den Schlüsselwörtern im **Drive PLC Developer Studio** gehören auch die Namen der Lenze-Funktionsblöcke, die immer mit "L_" beginnen (L_ABS, L_ADD, ...).

Reservierte Schlüsselwörter für Programmiersprachen der IEC 61131-3:

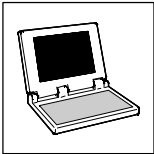
ABS	ACOS	ACTION	ADD	AND	ANDN
ANY	ANY_BIT	ANY_DATE	ANY_INT	ANY_NUM	ANY_REAL
ARRAY	ASIN	AT	ATAN		
BOOL	BY	BYTE			
CAL	CALC	CALCN	CASE	CD	CDT
CLK	CONCAT	CONFIGURATION	CONSTANT	COS	CTD
CTU	CTUD	CU	CV		
DATE	DATE_AND_TIME	DELETE	DINT	DIV	DO
DS	DT	DWORD			
ELSE	ESIF	END_ACTION	END_CASE	END_CONFIGURATION	END_FOR
END_FUNCTION	END_FUNCTION_BLOCK		END_IF	END_PROGRAM	END_REPEAT
END_RESOURCE	END_STEP	END_STRUCT	END_TRANSITION	END_TYPE	END_VAR
END_WHILE	EN	ENO	EQ	ET	EXIT
EXP	EXPT				
FALSE	F_EDGE	F_TRIG	FIND	FOR	FROM
FUNCTION	FUNCTION_BLOCK				
GE	GT				
IF	IN	INITIAL_STEP	INSERT	INT	INTERVAL
JMP	JMPC	JMPCN			
L	LD	LDN	LE	LEFT	LEN
LIMIT	LINT	LN	LOG	LREAL	LT
LWORD					
MAX	MID	MIN	MOD	MOVE	MUL
MUX					
N	NE	NEG	NOT		
OF	ON	OR	ORN		
P	PRIORITY	PROGRAM	PT	PV	
Q	Q1	QU	QD		



Drive PLC Developer Studio

Anhang

R	R1	R_TRIG	READ_ONLY	READ_WRITE	REAL
RELEASE	REPEAT	REPLACE	RESOURCE	RET	RETAIN
RETC	RETCN	RETURN	RIGHT	ROL	ROR
RS	RTC	R_EDGE			
S	S1	SD	SEL	SEMA	SHL
SHR	SIN	SINGLE	SINT	SL	SQRT
SR	ST	STEP	STN	STRING	STRUCT
SUB					
TAN	TASK	THEN	TIME	TIME_OF_DAY	TO
TOD	TOF	TON	TP	TRANS	TRUE
TYPE					
UDINT	UINT	ULINT	UNTIL	USINT	
VAR	VAR_ACCESS	VAR_EXTERNAL	VAR_GLOBAL	VAR_INPUT	VAR_IN_OUT
VAR_OUTPUT					
WHILE	WITH	WORD			
XOR	XORN				



15.4 Tastenkombinationen und Funktionstasten

Hier finden Sie eine Übersicht aller Tastenkombinationen und Funktionstasten:

Allgemeine Bedienung

Erläuterung der Funktion	Tastaturbefehl
Wechsel zwischen Deklarationsteil und Anweisungsteil eines Bausteins	<F6>
Wechsel zwischen <i>Object Organizer</i> , Objekt und Meldungsfenster	<Alt>+<F6>
Kontextmenü	<Umschalt>+<F10>
Kurzformmodus für Deklarationen	<Strg>+<Eingabetaste>
Auf- und Zuklappen mehrstufiger Variablen	<Eingabetaste>
Auf- und Zuklappen von Ordnern	<Eingabetaste>
Registerkartenwechsel im <i>Object Organizer</i> und Bibliotheksverwalter	<Pfeiltasten>
Weiterspringen in Dialogen	<Tabulator>
Kontextsensitive Hilfe	<F1>

Allgemeine Befehle

Menübefehl	Tastaturbefehl
Datei → Speichern	<Strg>+<S>
Datei → Drucken	<Strg>+<P>
Datei → Beenden	<Alt>+<F4>
Projekt → Objekt löschen	<Entf>
Projekt → Objekt einfügen	<Einfg>
Projekt → Objekt umbenennen	<Leertaste>
Projekt → Objekt bearbeiten	<Eingabetaste>
Projekt → Alles überprüfen	<Strg>+<F11>
Projekt → Allesübersetzen	<F11>
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	<Alt>+<F8>
Online → Ausloggen	<Strg>+<F8>
Online → Start	<F5>
Online → Stop	<Umschalt>+<F8>
Online → Breakpoint an/aus	<F9>
Online → Einzelschritt über	<F10>
Online → Einzelschritt in	<F8>
Online → Einzelzyklus	<Strg>+<F5>
Online → Werte schreiben	<Strg>+<F7>
Fenster → Meldungen	<Umschalt>+<Esc>



Drive PLC Developer Studio

Anhang

Befehle des FUP-Editors

Menübefehl	Tastaturbefehl
Einfügen → Netzwerk (danach)	<Strg>+<T>
Einfügen → Zuweisung	<Strg>+<A>
Einfügen → Sprung	<Strg>+<L>
Einfügen → Return	<Strg>+<R>
Einfügen → Operator	<Strg>+<O>
Einfügen → Funktion	<Strg>+<F>
Einfügen → Funktionsblock	<Strg>+
Einfügen → Eingang	<Strg>+<U>
Extras → Negation	<Strg>+<N>
Extras → Zoom	<Alt>+<Eingabetaste>

Befehle des KOP-Editors

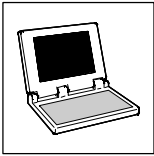
Menübefehl	Tastaturbefehl
Einfügen → Netzwerk (danach)	<Strg>+<T>
Einfügen → Kontakt	<Strg>+<O>
Einfügen → Paralleler Kontakt	<Strg>+<R>
Einfügen → Funktionsblock	<Strg>+
Einfügen → Spule	<Strg>+<L>
Extras → Darunter Einfügen	<Strg>+<U>
Extras → Negation	<Strg>+<N>

Befehle des AS-Editors

Menübefehl	Tastaturbefehl
Einfügen → Schritt-Transition (davor)	<Strg>+<T>
Einfügen → Schritt-Transition (danach)	<Strg>+<E>
Einfügen → Alternativzweig (rechts)	<Strg>+<A>
Einfügen → Parallellzweig (rechts)	<Strg>+<L>
Einfügen → Sprung	<Strg>+<U>
Extras → Zoom Aktion/Transition	<Alt>+<Eingabetaste>
Wechsel aus AS-Übersicht zurück in Editor	<Eingabetaste>

Befehle des CFC-Editors

Menübefehl	Tastaturbefehl
Einfügen → Baustein	<Strg>+
Einfügen → Eingang	<Strg>+<E>
Einfügen → Ausgang	<Strg>+<A>
Einfügen → Sprung	<Strg>+<J>
Einfügen → Label	<Strg>+<L>
Einfügen → Return	<Strg>+<R>
Einfügen → Kommentar	<Strg>+<K>
Einfügen → Bausteineingang	<Strg>+<U>
Extras → Negieren	<Strg>+<N>
Extras → Set/Reset	<Strg>+<T>
Extras → EN/ENO	<Strg>+<O>

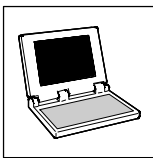


Bedienung der Steuerungskonfiguration

Erläuterung der Funktion	Tastaturbefehl
Auf- und Zuklappen von Organisationselementen	<Eingabetaste>
Editierrahmen um den Namen setzen	<Leertaste>

Bedienung der Taskkonfiguration

Erläuterung der Funktion	Tastaturbefehl
Editierrahmen um den Task- oder Programmnamen setzen	<Leertaste>



Drive PLC Developer Studio

Anhang

15.5 Fehlermeldungen

Die folgenden Fehlermeldungen, zeigt das DDS nach dem Einloggen oder Übersetzen im Meldungsfenster in alphabetischer Reihenfolge an.

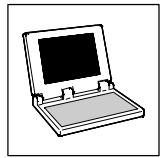


Hinweis!

Tritt eine Fehlermeldung auf, die nicht in diesem Kapitel beschrieben ist, wenden Sie sich bitte an Ihre Lenze-Vertretung.

15.5.1 Warnungen

Nr.	Ursache	Mögliche Abhilfe
1100	Unbekannte Funktion <Name> in Bibliothek.	Sie verwenden eine externe Bibliothek. Überprüfen Sie, daß 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 ein skalarem Eingang oder ein Programm mit dem Namen <Symbol> ohne 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, das 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, sind nicht in die Querverweisliste enthalten.
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 Analyse-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.
1500	Diese Expression enthält keine Zuweisung. Es wird kein Code generiert.	Das Ergebnis dieses Ausdrucks wird nicht verwendet. Deshalb wird für den gesamten Ausdruck kein Code generiert.
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. <pre>PROGRAM a VAR_GLOBAL a: INT; END_VAR ... a; (* Es wird nicht der Baustein a aufgerufen, sondern die Variable a geladen. *)</pre>
1503	Der Baustein hat keine Ausgänge. Die 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 abhängig vom logischen Ausdruck nicht ausgeführt.	Unter Umständen werden nicht alle Zweige des logischen Ausdrucks ausgeführt. <pre>IF a AND funct(TRUE) THEN...</pre> Wenn a FALSE ist, wird funct nicht mehr aufgerufen.
1505	Seiteneffekt in <Name>! Zweig wird möglicherweise nicht berechnet	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, 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.



Nr.	Ursache	Mögliche Abhilfe
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 ist nicht möglich.	Sie verwenden in der Konfiguration des Visualisierungsobjekts einen zusammengesetzten Ausdruck als Ziel einer Eingabeaktion. Ersetzen Sie diesen durch eine einzelne Variable.
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.
2900	Ein Systembaustein wurde in einer Skalieroutine verwendet. Es ist möglich, daß für diesen Systembaustein kein Prozeßabbild erstellt wird. Abschnitt Skalieroutinen (□ 8-20).	Stellen Sie mit dem Prozeßabbildmonitor sicher, daß ein Prozeßabbild für den verwendeten Systembaustein erstellt wird. □ 8-23
2901	Ein Systembaustein wurde in einer System POE verwendet. Es ist möglich, daß für diesen Systembaustein kein Prozeßabbild erstellt wird. Abschnitt System POEs in der Dokumentation des Zielsystems (□ 2-8)	Stellen Sie mit dem Prozeßabbildmonitor sicher, daß ein Prozeßabbild für den verwendeten Systembaustein erstellt wird. Achtung! Auch wenn ein Prozeßabbild erstellt wird, kann es bei einigen System POEs zu Problemen führen, wenn die Daten zum Verarbeitungszeitpunkt noch nicht vorhanden sind.
2902	Der Zielpfad für die GDC Gerätebeschreibung ist ungültig. Bei der Erstellung der GDC Gerätebeschreibung ist ein Fehler aufgetreten. Die Fehlerursache können Sie dem übertragenen Fehlercode entnehmen: 100: Kein gültiger Name für die Gerätebeschreibung 200: Die Basisgerätebeschreibung konnte nicht geöffnet werden 300: Fehler in der Basisgerätebeschreibung	100: Stellen Sie unter Projekt→Optionen , Kategorie <i>GDC Gerätebeschreibung</i> ein gültiges Verzeichnis ein. 200, 300: Deinstallieren Sie das DDS korrekt und installieren Sie es erneut. Sollte der Fehler weiterhin auftreten, wenden Sie sich an Ihre Lenze Vertretung.
2903	Eine Anwendercodestelle verwendet eine unbekannte Einheit.	Definieren Sie die verwendete Einheit im Instanzparametermanager. □ 8-13
2904	Die Codestellen-Referenz auf einen Baustein konnte nicht aufgelöst werden.	Der Baustein wurde gelöscht.
2905	Die Binädatei Projektname.bin (*.bin) konnte nicht erstellt werden.	Die *.bin Datei ist eventuell schreibgeschützt. Entfernen Sie den Schreibschutz.

15.5.2 Übersetzungsfehler

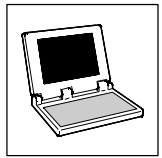
Nr.	Ursache	Mögliche Abhilfe
3100	Programm zu groß. Maximale Größe: <Anzahl> Byte (<Anzahl>K)	Die maximale Programmgröße ist überschritten. Verkleinern Sie das Programm.
3101	Datenbereich zu groß. Maximale Größe: <Anzahl> Byte (<Anzahl>K)	Der Datenspeicher ist aufgebraucht. Verringern Sie den Dateibedarf 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 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.



Drive PLC Developer Studio

Anhang

Nr.	Ursache	Mögliche Abhilfe
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. Legen Sie das Ergebnis einer IEC-Funktion auf die Variable.
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.
3200	<Name> (%d): Logischer Ausdruck zu komplex.	Die 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.	Innerhalb der Indexberechnung wurde eine Funktion aufgerufen.
3208	Umwandlung ist nicht implementiert.	Sie verwenden eine Konvertierungsfunktion, die für den aktuellen Codegenerator nicht implementiert ist.
3209	Operator ist 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 zehn mal verwendet werden.
3250	Real wird auf 8 Bit Controller nicht unterstützt	Das Zielsystem wird zur Zeit nicht unterstützt.
3251	Time of Day Typen werden auf 8 Bit Controller nicht unterstützt.	Das Zielsystem wird zur Zeit nicht unterstützt.
3252	Stackgröße übersteigt %ld Bytes.	Das Zielsystem wird zur Zeit nicht unterstützt.
3253	Hexfile nicht gefunden: <Name>	Das Zielsystem wird zur Zeit nicht unterstützt.
3254	Aufruf einer externen Bibliotheksfunktion konnte nicht aufgelöst werden.	Das Zielsystem wird zur Zeit nicht unterstützt.
3400	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.
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 Kategorie <i>Verzeichnisse</i> .
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.



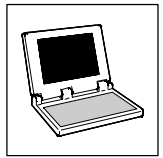
Nr.	Ursache	Mögliche Abhilfe
3454	Die Adresse <Name> verweist auf einen belegten Speicherbereich!	Sie haben die Option Adreßüberschreitungen prüfen im Dialog <i>Einstellungen</i> der Steuerungskonfiguration aktiviert und es wurde eine Überschneidung festgestellt. Beachten Sie, daß die Grundlage der Bereichsprüfung 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 Kategorie <i>Verzeichnisse</i> .
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.
3500	Kein VAR_CONFIG für <Name>	Fügen Sie für die genannte Variable in der globalen Variablenliste mit der Variablen Konfiguration eine Deklaration ein. Ressourcen→Globale Variablen 8-2
3501	Keine Adresse in VAR_CONFIG für <Name>	Fügen Sie für die genannte Variable in der globalen Variablenliste mit der Variablen Konfiguration eine Adresse ein. Ressourcen→Globale Variablen 8-2
3502	Falscher Datentyp von <Name> in VAR_CONFIG	Die genannte Variable ist in der globalen Variablenliste mit der Variablen Konfiguration, nicht mit dem gleichen Datentypen deklariert worden 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 parametrisiert.	Sie haben im Feld Programmaufruf Parameter angegeben, die nicht der Deklaration des Bausteins entsprechen.
3600	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.
3611	Das Übersetzungsverzeichnis <Name> ist ungültig	Sie haben in den Projekt→Optionen Kategorie <i>Verzeichnisse</i> ein ungültiges Verzeichnis für die Übersetzungsdateien eingetragen.
3612	Maximale Anzahl der Bausteine (<Anzahl>) überschritten! Übersetzung wird abgebrochen.	Sie verwenden zuviele Bausteine und Datentypen im Projekt. Verändern Sie die max. Anzahl der Bausteine in dem Dialogfeld <i>Zielsystemeinstellungen</i> Registerkarte <i>Speicheraufteilung</i> . Hinweis! Die Registerkarte <i>Speicheraufteilung</i> ist zur Zeit nicht editierbar.
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.
3700	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.



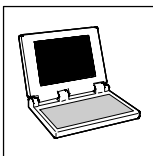
Drive PLC Developer Studio

Anhang

Nr.	Ursache	Mögliche Abhilfe
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.
3705	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.
3722	Auf der angegebenen Adresse dürfen nur einfache boolsche Variablen stehen	Ändern Sie die Adresse oder den in der Deklaration angegebenen Typ der Variablen.
3728	PLC_PRG(): Unzulässige Adresse %IX141.58.0.	Geben Sie eine korrekte Adresse ein.
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.
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, bzw. Initialisierung von Pointern pt:POINTER TO<typ>:= ADR(variable); nicht möglich	Verwenden Sie einen Initialwert, welcher der Typdefinition entspricht. Sie können den Variablendeklarationsdialog zu Hilfe nehmen. Shift+F2 oder, Bearbeiten→Variablenkonfiguration . Grundsätzlich sollten Pointer vermieden werden. Beachten Sie auch im Kapitel Datentypen den Abschnitt Pointer. 10-5
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 Bausteins 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.
3800	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.
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 dem Dialogfeld <i>Zielsystemeinstellungen</i> Registerkarte <i>Speicheraufteilung</i> 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 globale Variablen ist erschöpft. Der verfügbare Speicherplatz ist abhängig vom Zielsystem. Informationen über den zu Verfügung stehenden Speicherplatz für das Zielsystem finden Sie im Dialogfeld <i>Zielsystemeinstellungen</i> , Registerkarte Speicheraufteilung . 8-41.



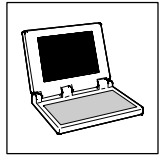
Nr.	Ursache	Mögliche Abhilfe
3803	Speicher für globale Variablen aufgebraucht. Variable <Name>, <Anzahl> Byte.	Der verfügbare Speicherplatz für globale Variablen ist erschöpft. - Der verfügbare Speicherplatz ist abhängig vom Zielsystem. - Informationen über den für das Zielsystem zur Verfügung stehenden Speicherplatz finden Sie im Dialogfeld Zielsystemeinstellungen, Registerkarte Speicheraufteilung.
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	Fügen Sie mindestens einen Eingabeparameter für die Funktion ein.
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.
3900	Mehrfache Unterstriche hintereinander im Bezeichner	Entfernen Sie mehrfache Unterstriche hintereinander im Bezeichnernamen. OK: VAR x: REAL; _u_: INT; END_VAR
3901	Es sind maximal vier 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 IEC 61131-3 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 IEC 61131-3 Format angegeben.
3906	Ungültige Tageszeitkonstante	Die Konstante ist nicht entsprechend dem IEC 61131-3 Format angegeben.
3907	Ungültige Datum/Zeit-Konstante	Die Konstante ist nicht entsprechend dem IEC 61131-3 Format angegeben.
3908	Ungültige Stringkonstante	Die Stringkonstante enthält ein ungültiges Zeichen.
4000	Bezeichner erwartet	Geben Sie an dieser Stelle einen gültigen Bezeichner an.
4001	Variable <Name> nicht deklariert	Deklariieren Sie die Variable lokal oder global.
4010	Unverträgliche Typen: Kann <Name> nicht in <Name> konvertieren.	Prüfen Sie die erforderlichen Typen des Operators 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 der 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.



Drive PLC Developer Studio

Anhang

Nr.	Ursache	Mögliche Abhilfe
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 BITADR. 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 keinen 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 einfügen 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.
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, weil 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, weil 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. Es handelt sich um eine Übergabe über Referenz.
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.
4100	^ 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's 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 Eingabeparameter des aufgerufenen Funktionsblocks und ändern Sie <Name> in eine dieser Variablen.



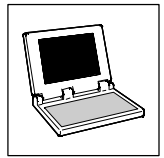
Nr.	Ursache	Mögliche Abhilfe
4200	LD erwartet	Fügen Sie im Editorfenster des AWL-Bausteins bzw. nach der Sprungmarke zumindest eine LD-Anweisung ein.
4201	AWL Operator erwartet	Jede AWL-Anweisung muß mit einem Operator oder einer Sprungmarke beginnen.
4202	Unerwartetes Ende des Klammerausdrucks	Fügen Sie die schließende Klammer ein.
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 und ELSIF folgt, ein boolscher Ausdruck ist.
4254	WHILE benötigt als Bedingung einen Boolschen Ausdruck	Stellen Sie sicher, daß die Bedingung, die einem WHILE folgt, ein boolscher Ausdruck ist.
4255	UNTIL benötigt als Bedingung einen Boolschen Ausdruck	Stellen Sie sicher, daß die Bedingung, die einem UNTIL folgt, ein boolscher Ausdruck ist.
4256	NOT verlangt einen boolschen Operanden	Stellen Sie sicher, daß die Bedingung, die einem NOT folgt, ein boolscher 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.
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 Selector 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 Funktionsbausteinanruf muß mit dem Namen einer Instanz beginnen	Der Kennzeichner im Funktionsbausteinanruf 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 zusätzlich die Position der Parameter (Reihenfolge) mit der in der Funktionsdefinition 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.
4300	Sprung bzw. Return benötigen eine boolsche Eingabe	Stellen Sie sicher, daß der Eingang für den Sprung bzw. die Return-Anweisung ein boolscher Ausdruck ist.
4301	Baustein <Name> verlangt genau <Anzahl> Eingänge	Die Anzahl der Ausgänge entspricht nicht der Anzahl der in der Bausteindefinition angegebenen VAR_INPUT und VAR_IN_OUT Variablen.



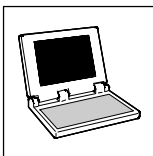
Drive PLC Developer Studio

Anhang

Nr.	Ursache	Mögliche Abhilfe
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 boolscher Ausdruck <Name> bei Kontakt benutzt	Das Schaltsignal für einen Kontakt muß ein boolscher Ausdruck sein.
4321	Nicht boolscher Ausdruck <Name> bei Spule benutzt	Die Ausgangsvariable einer Spule muß vom Typ BOOL sein.
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 boolsche Eingabe	Stellen Sie sicher, daß der Eingang für den Sprung ein boolscher Ausdruck ist.
4341	Return benötigt eine boolsche Eingabe	Stellen Sie sicher, daß der Eingang für die Return-Anweisung ein boolscher 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.
4353	Schrittname 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 Schrittname: <Name>	Verwenden Sie einen gültigen Bezeichner als Sprungziel.
4357	Die IEC-Bibliothek wurde nicht gefunden.	Prüfen Sie, ob im Bibliotheksverwalter die Bibliothek iecsc.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 boolscher 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 boolscher 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.



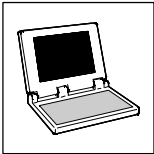
Nr.	Ursache	Mögliche Abhilfe
4366	Die Marke für den Parallelschritt ist kein zulässiger Bezeichner: <Name>	Tragen Sie neben dem Dreieck, das die Sprungmarke anzeigt, einen zulässigen Bezeichner ein.
4367	Die Marke <Name> ist bereits vorhanden.	Sie haben bereits eine Sprungmarke oder einen Schritt mit diesem Namen bezeichnet. Benennen Sie dementsprechend um.
4368	Aktion <Name> wird in mehreren übereinanderliegenden AS-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.
4400	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.
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 ist 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.
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	Zuviele 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	Zuviele 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.



Drive PLC Developer Studio

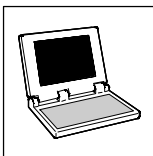
Anhang

Nr.	Ursache	Mögliche Abhilfe
4500	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 Dialogfeld <i>Zielsystemeinstellungen</i> Registerkarte <i>Netzfunktionen</i> festgelegten Bereich liegt.
4551	Subindex nicht in erlaubten Bereich: Variablen OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, daß der Subindex in dem Dialogfeld <i>Zielsystemeinstellungen</i> Registerkarte <i>Netzfunktionen</i> festgelegten Bereich liegt.
4552	Index nicht in erlaubtem Bereich: Parameter OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, daß der Index in dem Dialogfeld <i>Zielsystemeinstellungen</i> Registerkarte <i>Netzfunktionen</i> festgelegten Bereich liegt.
4553	Subindex nicht in erlaubtem Bereich: Parameter OD <Nummer>, Zeile <Zeilennummer>.	Stellen Sie sicher, daß der Subindex in dem Dialogfeld <i>Zielsystemeinstellungen</i> Registerkarte <i>Netzfunktionen</i> 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.
4800	Die maximale Anzahl der Tasks wurde überschritten. (Auch evtl. verwendete Touch-Probe-Eingänge zählen als Task)	Löschen Sie eine Task oder entfernen Sie einen Touch-Probe-Eingang.
4801	Ein Hardwareinterrupt zum Starten einer Task wurde mehrfach verwendet.	Überprüfen Sie alle verwendeten Hardwareinterrupts.
4802	Die Bezeichnung einer Task wurde mehrfach verwendet.	Prüfen Sie alle Taskbezeichnungen.
4803	Eine Taskpriorität wurde mehrfach vergeben. Die Priorität 0, Task nicht aktiv, darf mehrfach vergeben werden.	Überprüfen Sie alle vergebenen Prioritäten.
4804	Ein digitaler Eingang wurde zum Starten einer Task (Hardwareinterrupt) und als Touch-Probe-Eingang verwendet. Ein Eingang kann entweder zum Starten einer Task oder als TP verwendet werden. Beides ist nicht zulässig.	Entfernen Sie entweder den TP Eingang oder verwenden Sie einen anderen Eingang zum Starten einer Task.
4805	Ungültige Überwachungszeit (Überwachungszeit einer Task ist 0).	Überprüfen Sie die Überwachungszeit der Tasks.
4810	Die im Instanzparametermanager angelegten Codestellen benötigen zu viel Speicherplatz.	Entfernen Sie einige Codestellen.
4811	Für eine Codestelle wurden mehrere Initialisierungswerte vergeben. Die Initialisierung einer Codestelle muß eindeutig sein.	Überprüfen Sie alle Initialisierungswerte.
4812	Es wurde versucht, in einer Anwender-Bibliothek eine Codestelle anzulegen. Das Anlegen von Codestellen in Anwender-Bibliotheken ist nicht zulässig.	Öffnen Sie die betroffene Bibliothek und entfernen Sie alle Codestellen.
4813	Es wurde eine Skalieroutine in einer globalen Funktionsblock-Instanz verwendet.	Instanzieren Sie Funktionsblöcke mit Skalerroutinen nur lokal.
4820	Bei der Bearbeitung der Codestelleninitialisierungswerte ist ein Fehler aufgetreten.	Die Gerätebeschreibung des verwendeten Zielsystems wurde nicht gefunden.
4850	Die Erstellung einer externen Bibliothek ist mit der verwendeten Version des Drive PLC Developer Studio nicht möglich.	Bitte wenden Sie sich an Lenze.



15.5.3 Kommunikationsfehler

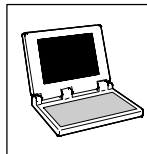
Nr.	Ursache	Mögliche Abhilfe
1 (0001)	Sendepuffer im Controller ist voll. CAN_ERR_XMTFULL	Systemnahe Kommunikationsfehler 15-23
2 (0002)	CAN-Controller wurde zu spät gelesen. CAN_ERR_OVERRUN	
4 (0004)	Busfehler: Ein Fehlerzähler (Errorcounter) erreichte das Limit. CAN_ERR_BUSLIGHT	
8 (0008)	Busfehler: Ein Fehlerzähler (Errorcounter) erreichte das Limit. CAN_ERR_BUSHEAVY	
16 (0010)	Busfehler: CAN-Controller ging CANopen Zustand Bus Off. CAN_ERR_BUSOFF	
32 (0020)	RcvQueue ist ausgelesen. CAN_ERR_QRCVEMPTY	
64 (0040)	RcvQueue wurde zu spät gelesen. CAN_ERR_QOVERRUN	
128 (0080)	Sende-Queue ist voll. CAN_ERR_QXMTFULL	
256 (0100)	Register-Test des 82C200 schlug fehl. CAN_ERR_REGTEST	
512 (0200)	Problem mit der Lokalisierung des VxD. CAN_ERR_NOVXD	
1024 (0400)	Netzhardware ist belegt. CAN_ERR_HWINUSE	
2048 (0800)	Netz ist von Clients belegt. CAN_ERR_NETINUSE	
5120 (1400)	Hardwarehandle war ungültig. CAN_ERR_ILLHW	
6144 (1800)	Netzhandle war ungültig. CAN_ERR_ILLNET	
7168 (1C00)	Clienthandle war ungültig. CAN_ERR_ILLCCLIENT	
8192 (2000)	Ressource (FIFO, Client, Timeout) ist nicht erzeugbar. CAN_ERR_RESOURCE	
40960 (A000)	Allgemeiner Fehler	Wenden Sie sich an Lenze.
40961 (A001)	Bei diesem Systembusadapter ist die gewählte Funktion nicht möglich.	Benutzen Sie einen passenden Systembusadapter.
40962 (A002)	Es ist kein Systembusadapter aktiviert.	Aktivieren Sie einen Systembusadapter.
41217 (A101)	Interner Programmfehler (NULL Pointer)	Starten Sie die Anwendung erneut. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41218 (A102)	Zeitüberschreitung beim Lesen von einem Busteilnehmer.	Vergrößern Sie im Lenze Systembus-Konfigurator, im Register Einstellungen für den ausgewählten Systembusadapter den Wert des Parameters Kommunikations-Timeout.
41219 (A103)	Interner Programmfehler (kein gültiges Netz)	Installieren Sie das CAN Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41220 (A104)	Interner Programmfehler (keine gültigen Daten)	Installieren Sie das CAN Treiber Update. Tritt der Fehler weiterhin auf, wenden Sie sich an Lenze.
41221 (A105)	Interner Programmfehler (keine gültige Operation)	Installieren Sie das CAN Treiber Update. Tritt der Fehler weiterhin auf, wenden Sie sich an Lenze.
41222 (A106)	Interner Programmfehler Fehler beim Lesen eines Datenblocks	Starten Sie die Anwendung erneut. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41223 (A107)	Interner Programmfehler (keine gültige Größe)	Installieren Sie das CAN Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41224 (A108)	Fehler während der Initialisierung des Daten-Downloads	Starten Sie die Anwendung erneut. Prüfen Sie mit dem Systembus-Konfigurator, ob das Zielsystem ansprechbar ist.
41225 (A109)	Fehler während der Initialisierung des Daten-Uploads	Starten Sie die Anwendung erneut. Prüfen Sie mit dem Systembus-Konfigurator, ob das Zielsystem ansprechbar ist.
41226 (A110)	Fehler während der Datenübertragung	Starten Sie die Anwendung erneut. Prüfen Sie mit dem Systembus-Konfigurator, ob das Zielsystem ansprechbar ist.
41227 (A10B)	Ungültige E/A-Adresse	Stellen Sie im Lenze Systembus-Konfigurator, im Register Einstellungen für den ausgewählten Systembus-Adapter eine gültige E/A-Adresse ein.



Drive PLC Developer Studio

Anhang

Nr.	Ursache	Mögliche Abhilfe
41228 (A10C)	Ungültiger Interrupt	Stellen Sie im Lenze Systembus-Konfigurator, im Register Einstellungen für den ausgewählten Systembus-Adapter einen gültigen Interrupt ein.
41229 (A10D)	Ungültige Baudrate	Stellen Sie im Lenze Systembus-Konfigurator, im Register Einstellungen für den ausgewählten Systembus-Adapter eine gültige Baudrate ein.
41230 (A10E)	Die für den ausgewählten Systembusadapter erforderlichen Treiber konnten nicht gefunden werden.	Installieren Sie die Treiber für den ausgewählten Systembusadapter erneut.
41231 (A10F)	Der Systembus-Adapter wird bereits von einer anderen Anwendung in einem nicht kompatiblen Modus verwendet.	Beenden Sie alle Anwendungen und versuchen Sie es erneut.
41232 (A110)	Fehler beim laden der Ressourcen.	Installieren Sie das CAN-Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41233 (A111)	Die Konfigurationsdatei konnte nicht geladen werden.	Installieren Sie den Systembus-Konfigurator erneut.
41234 (A112)	Fehler beim Lesen vom Zielsystem.	Setzen Sie das Zielsystem zurück. (Netzschalten) Versuchen Sie es erneut.
41235 (A113)	Fehler beim Schreiben vom Zielsystem.	Setzen Sie das Zielsystem zurück. (Netzschalten) Versuchen Sie es erneut.
41236 (A114)	Die maximale Anzahl der unterstützen Kommunikationsbaugruppen (z.B. CAN USB Adapter) wurde überschritten.	Entfernen Sie eine Kommunikationsbaugruppe.
41237 (A115)	Der ausgewählte Systembusadapter wird von den installierten Treibern nicht unterstützt.	Installieren Sie ein Treiber-Update für den ausgewählten Systembusadapter.
41238 (A116)	Es wurde ein dem System unbekannter Systembusadapter ausgewählt.	Installieren Sie ein Treiber-Update für den zu verwendenden Systembusadapter.
41239 (A117)	Beim Initialisieren des ausgewählten Systembusadapters trat ein Fehler auf.	Prüfen Sie, ob der ausgewählte Systembusadapter korrekt am PC angeschlossen ist.
41240 (A118)	Beim Schreiben der Konfigurationsdatei trat ein Fehler auf.	Installieren Sie den Systembus-Konfigurator erneut.
41241 (A119)	Das Format der Konfigurationsdatei ist ungültig.	Installieren Sie den Systembus-Konfigurator erneut.
41242 (A11A)	Es sind mehrere PC-Systembusadapter 2177 gleichzeitig am PC angeschlossen.	Schließen Sie nur den zu konfigurierenden PC-Systembusadapter 2177 an den PC an.
41243 (A11B)	Beim Schreiben der USB-Gerätenummer zum PC-Systembusadapter 2177 trat ein Fehler auf.	Prüfen Sie, ob der Systembusadapter korrekt und als einziger PC-Systembusadapter 2177 am PC angeschlossen ist.
41244 (A11C)	Beim Lesen der USB-Gerätenummer vom PC-Systembusadapter 2177 trat ein Fehler auf.	Prüfen Sie, ob der Systembusadapter korrekt und als einziger PC-Systembusadapter 2177 am PC angeschlossen ist.
41245 (A11D)	Die Datei "LSsystembusL1.DLL" konnte nicht geladen werden.	Installieren Sie den Systembus-Konfigurator erneut.
41246 (A11E)	Fehler beim Initialisieren des XML-Parsers.	installieren Sie das CAN Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41247 (A11F)	Der Systembus-Adapter konnte nicht initialisiert werden. Keine freien Handle.	installieren Sie das CAN Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41248 (A120)	Fehler bei der asynchronen Kommunikation.	Starten Sie die Anwendung erneut. Prüfen Sie mit dem Systembus-Konfigurator, ob das Zielsystem ansprechbar ist.
41249 (A121)	Die für den ausgewählten Systembusadapter eingestellte Baudrate entspricht nicht der Baudrate der am Bus angeschlossenen Teilnehmer.	Stellen Sie für den ausgewählten Systembusadapter im Register Einstellungen die Baudrate ein, die auch für die Teilnehmer am Bus eingestellt ist.
41250 (A122)	Interner Programmfehler Ungültige Filtereinstellung	Installieren Sie das CAN Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41251 (A123)	Der verwendete Parameterkanal wird bereits von einem anderen Busteilnehmer oder einem anderen PC-Programm verwendet.	Stellen Sie im Lenze Systembus-Konfigurator, im Register Einstellungen für den ausgewählten Systembus-Adapter einen anderen Parameterkanal ein. Der gewählte Parameterkanal soll nicht von anderen Busteilnehmern oder einem anderen PC-Programm belegt sein oder werden.
41252 (A124)	Die Datei LSsystembusL2.DLL konnte nicht geladen werden.	installieren Sie das CAN Treiber Update erneut.
41253 (A125)	Fehler während dem Einschalten des Download-Modes	Starten sie die Anwendung erneut.
41254 (A126)	Fehler während eines Programm-Downloads.	Starten Sie die Anwendung erneut.
41255 (A127)		
41256 (A128)	Auf dem Systembus wurden zu viele ungültige Telegramme festgestellt.	Überprüfen Sie die Installation des Systembusses (Abschlußwiderstand, Schirmung) auf die verwendeten Baudraten.



Nr.	Ursache	Mögliche Abhilfe
41728 (A300)	Interner Programmfehler OPC-Item konnte nicht angelegt werden	Starten sie die Anwendung erneut.
41729 (A301)	interner Programmfehler OPC-Item konnte nicht geschrieben werden	Starten sie die Anwendung erneut. Prüfen Sie mit dem Systembus-Konfigurator, ob das Zielsystem ansprechbar ist.
41730 (A303)	interner Programmfehler OPC-Item konnte nicht gelesen werden.	Starten sie die Anwendung erneut. Prüfen Sie mit dem Systembus-Konfigurator, ob das Zielsystem ansprechbar ist.
41731 (A303)	interner Programmfehler OPC-Verbindungen fehlgeschlagen.	Installieren Sie das CAN Treiber Update. Besteht das Problem weiterhin, wenden Sie sich an Lenze.
41732 (A304)	interner Programmfehler OPC-Verbindungspunkt für asynchrone Kommunikation konnte nicht angelegt werden.	Installieren Sie das CAN Treiber Update. Besteht der Fehler weiterhin, wenden Sie sich an Lenze.
41733 (A305)	interner Programmfehler OPC-Item konnte nicht gelöscht werden.	Starten Sie die Anwendung erneut.
41734 (A306)	interner Programmfehler OPC-Item Liste konnte nicht gelöscht werden.	Starten Sie die Anwendung erneut.
45569 (B201)	Das Zielsystem hat einen H05-Trip. Interne Störung.	Wenden Sie sich an Lenze.
45570 (B202)	Das Zielsystem hat einen PRO-Trip. Parametersatzfehler	Setzen Sie den Trip zurück.
45571 (B203)	Das Zielsystem hat einen PER-Trip. Programmstörung	Setzen Sie den Trip zurück. (Netzschalten)
45572 (B204)	Das Zielsystem hat einen CCR-Trip. Systemstörung	Setzen Sie den Trip zurück. (Netzschalten)
45573 (B205)	Das Zielsystem hat einen unbekannten Fehler.	Wenden Sie sich an Lenze.
45574 (B206)	Die Softwareversion des Zielsystems paßt nicht zu dem Programm, das geladen werden soll.	Verwenden Sie das entsprechende Zielsystem.
45575 (B207)	Das Zielsystem ist schreibgeschützt.	Entfernen Sie für einen Download den Schreibschutz.
45576 (B208)	Die Reglersperre ist im Zielsystem nicht gesetzt.	Setzen Sie die Reglersperre.
45577 (B209)	Das SPS-Programm ist nicht gestoppt.	Stoppen Sie das SPS-Programm.
45578 (B20A)	Das zu ladende Programm ist zu groß für das Zielsystem.	Verkeinnern Sie das Programm. Teilen Sie das Programm auf.
45579 (B20B)	Das Zielsystem verfügt nicht über genügend Technologie-Einheiten.	Verwenden Sie als Zielsystem eine xT-Variante. z. B. EVS93xx-xT
45580 (B20C)	Die Vorbereitungen für den Programmtransfer sind fehlgeschlagen.	Aktivieren Sie den Programmtransfer erneut. Besteht das Problem weiterhin, wenden Sie sich an Lenze.
45581 (B20D)	Fehler beim Lesen aus dem Zielsystem.	Setzen Sie das Zielsystem zurück. (netzschalten) Versuchen Sie es anschließend erneut.
45582 (B20E)	Fehler beim Schreiben aus dem Zielsystem.	Setzen Sie das Zielsystem zurück. (netzschalten) Versuchen Sie es anschließend erneut.

15.5.4 Systemnahe Kommunikationsfehler



Hinweis!

Es ist ein systemnaher Kommunikationsfehler am CAN-Bus aufgetreten.

Prüfen Sie

- die Netzwerk-Hardware.
(Verbindungen, Verkabelung, Abschlußwiderstände)
- die verwendeten Systembus-Adapter.
- die Konfigurations-Einstellungen.
(keine doppelt vergebenen Knoten-Adressen, gleiche Baudraten)
- Starten sie den PC neu.

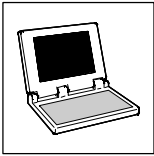


Drive PLC Developer Studio

Anhang

15.6 Glossar

Ablaufsprache	Ablaufsprache AS (<i>Sequential Function Chart - SFC</i>) ist eine Programmiersprache zur Beschreibung sequentieller und paralleler Steuerungsabläufe mit Zeit- und Ereignis-Steuerung.
Aktion	Boolsche Variable oder eine Reihe von Anweisungen, die über einen Aktionsblock angesteuert werden können, in AS.
Aktionsblock	Aktivierungsbeschreibung von Aktionen in AS.
Aktuelles Ergebnis	Zwischenergebnis in AWL von beliebigem Datentyp.
Anweisungsliste	Anweisungsliste (<i>Instruction List - IL</i>) ist eine weit verbreitete Assembler-ähnliche Programmiersprache für SPS-Systeme.
AS	Abk. für Ablaufsprache.
AWL	Abk. für Anweisungsliste.
Baustein	(Unter-) Programmeinheit, aus denen SPS-Programme zusammengesetzt werden. Bausteine können oft unabhängig voneinander in die SPS geladen werden. Vgl. POE.
CPU	Zentraleinheit (<i>Central Processing Unit</i>), z. B. einer SPS.
Deklaration	Bekanntgabe von Variablen und FB-Instanzen in einem Deklarationsblock unter Angabe eines Bezeichners, des Datentyps bzw. FB-Typs sowie ggf. Anfangswerte, Bereichsangabe und Feldeigenschaften.
Deklarationsblock	Zusammenfassung von Deklarationen einer Variablenart zu Beginn der POE.
Elementarer Datentyp	Ein durch die IEC 61131-3 vordefinierter Standard-Datentyp.
Erweiterung von Funktionen	Eine Funktion kann eine variable Anzahl von Eingängen besitzen.
FB	Abk. für Funktionsblock (<i>Function Block</i>), oftmals wird hierfür auch der Begriff "Funktionsbaustein" verwendet.
FB-Instanz	siehe Instanz
FB-Typ	Name eines Funktionsblocks mit Aufruf- und Rückgabeschnittstelle.
FBD	<i>Function Block Diagram</i> , siehe Funktionsplan
Funktion	Eine POE vom Typ Function
Funktionsbaustein	siehe Funktionsblock
Funktionsblock	Eine POE vom Typ Function_Block
Funktionsplan	Der Funktionsplan (<i>Function Block Diagram</i>) besteht aus einer Liste von Netzwerken, in denen sich beliebige Programmabläufe mit Hilfe von Verbindungselementen vollgrafisch realisieren lassen.
FUP	siehe Funktionsplan
IL	<i>Instruction List</i> , siehe Anweisungsliste
Indirekter FB-Aufruf	Aufruf einer FB-Instanz, dessen Name als VAR_IN_OUT -Parameter einer POE übergeben wurde.
Instanz	Strukturierter Datensatz eines FBs durch Deklaration eines Funktionsblocks unter Angabe des FB-Typs.
KOP	Abk. für Kontaktplan.
Konfiguration	Die Konfiguration (<i>Configuration</i>) definiert die Struktur der SPS, sie bildet die oberste Ebene im Softwaremodell der IEC 61131-3.
Kontaktplan	Kontaktplan (<i>Ladder Diagram - LD</i>) ist eine Programmiersprache zur Beschreibung von Netzwerken mit gleichzeitig arbeitenden boolschen, elektromechanischen Elementen wie Kontakten und Spulen.
LD	<i>Ladder Diagram</i> , siehe Kontaktplan
POE	Abk. für Programmorganisationseinheit (<i>Program Organization Unit - POU</i>)
POU	<i>Program Organization Unit</i> , siehe Programmorganisationseinheit
Programmorganisationseinheit	Ein Baustein der IEC 61131-3 des Typs Funktion, Funktionsblock oder Programm, aus dem Anwenderprogramme hierarchisch aufgebaut werden.
Ressource	Eine Ressource (<i>Resource</i>) stellt innerhalb einer Konfiguration eine Zentraleinheit (CPU) dar.
Schritt	Zustandsknoten in einem AS-Programm, in dem Anweisungen der zu einem Schritt zugehörigen Aktion angestoßen werden.
SFC	<i>Sequential Function Chart</i> , siehe Ablaufsprache
SPS	Speicherprogrammierbare Steuerung (<i>Programmable Controller</i>).



ST	Abk. für Strukturierter Text.
Standard-Funktionen	Menge der durch die IEC 61131-3 fest vordefinierten Funktionen zur Realisierung SPS-typischer Funktionalität.
Standard-Funktionsbausteine	siehe Standard-Funktionsblöcke
Standard-Funktionsblöcke	Menge der durch die IEC 61131-3 fest vordefinierten Funktionsblöcke (<i>Function Blocks</i>) zur Realisierung SPS-typischer Funktionalität.
Strukturierter text	Strukturierter Text (<i>Structured Text</i>) ist eine Programmiersprache zur Beschreibung von Algorithmen und Ausführungssteuerung mit den Mitteln einer modernen Hochsprache.
Task	Definition von Laufzeiteigenschaften eines Programms.
Transition	Übergang von einem AS-Schritt zum nächsten durch Auswertung der Transitionsbedingung.
Typdefinition	Definition eines benutzerspezifischen Datentyps auf der Basis bereits vorhandener Datentypen.
Variable	Bezeichnung eines Datenspeichers, der Werte annehmen kann, die durch den Datentyp sowie Angaben bei der Variablen-Deklaration festgelegt werden.
Zyklus	Ein Durchlauf des (periodisch aufgerufenen) Anwenderprogramms.
Zykluszeit	Die Zeit, die ein Anwenderprogramm für einen Zyklus benötigt.



Drive PLC Developer Studio

Anhang



16 Index

A

- Abgerundetes Rechteck, 9-2
- Ablaufsprache (AS), 4-11
- Absolutwert, 14-5
- Abwärtszähler, 14-14
- Addition, 12-1
- Administratorrechte, 6-21
- Adressen berechnen, 8-28
- Adressfunktion, 12-12
- Aktion assoziieren, 7-63
- Aktion hinzufügen, 7-64
- Alarmfarbe, 9-18
- Alle Schließen, 5-5
- Alles bereinigen, 6-8, 6-9
- Alles Markieren, 7-47, 9-8
- Alles nach Datenfluss anordnen, 7-53
- Alternativzweig (links), 7-58
- Alternativzweig (rechts), 7-58
- An den Anfang, 7-53
- Anfangsstring, 14-7
- Anhang, 15-1
- Anisotropisch, 9-18
- Ans Ende, 7-54
- Anweisungen, 4-5
- Anweisungsliste (AWL), 4-2
- Anzeige beim Schritt, 7-63
- Anzeigen, 7-52
- Arbeitsbereich, 5-9
- Arbeitsgruppen, 6-21
- Archiv, 6-3
- Arcuscosinus, 14-6
- Arcussinus, 14-6
- Arcustangens, 14-7
- Auf- und Abwärtszähler, 14-15
- Aufruf eines Funktionsblocks, 2-5, 12-12
- Aufrufbaum ausgeben, 6-29
- Aufrufhierarchie, 6-37
- Aufwärtszähler, 14-14
- Ausdrücke, 4-4

- Ausgabevariablen, 7-10
- Ausgang, 7-34, 7-45
- Ausgangsaktion hinzufügen, 7-60
- Ausloggen, 6-32
- Ausnahmeverhalten, 6-22
- Ausrichten, 9-8
- Ausschließen, 6-11
- Ausschneiden, 7-2, 7-35
- Auswertung von Ausdrücken, 4-4
- Automatisch deklarieren, 5-8
- Automatisch formatieren, 5-8
- Automatisch laden, 5-7
- Automatisch sichern, 5-6

B

- Basisparameter, 8-30
- Basissprache, 5-18
- Baustein, 2-2, 7-32, 7-40, 7-45
- Bausteine erzeugen, 3-2
- Bausteineingang, 7-46
- Beenden, 6-8
- Begriffsdefinitionen, 1-1
- Beispiel eines Programms, 2-7
- Beispielprogramm, 3-1
- Benutzerinformation, 5-7
- Bewegung absolut, 9-12
- Bewegung relativ, 9-13
- Bibliothek einfügen, 8-45
- Bibliotheken, 2-8
- Binäre Selektion, 12-8
- Binärfile, 5-13
- Bistabiler Funktionsblock, 14-10
- Bitkanäle, 8-30
- Bitmap, 9-4, 9-18
- Bitweise Linksrotation, 12-6
- Bitweise Rechtsrotation, 12-7
- Bitweises Links-Shift, 12-6
- Bitweises Rechts-Shift, 12-6
- Bitwerte, 5-9
- Breakpoint, 2-10



Drive PLC Developer Studio

Index

Breakpoint an/aus, 6-34
 Breakpoint löschen, 6-35
 Breakpoint setzen, 6-35
 Breakpoint-Dialog, 6-35
 Breakpoint, 7-29, 7-67

C

CFC, 4-20
 CheckBounds, 2-3
 CheckDiv, 12-2
 CheckRange, 10-9
 Codeliste, 8-8
 Codestellen, 8-10
 Codestellen parametrieren, 8-8
 Continuous Function Chart, 7-43
 Cosinus, 14-6

D

Dahinter Einfügen, 7-40
 Darstellung, 9-10
 Darüber Einfügen, 7-40
 Darunter Einfügen, 7-40
 Datei, 6-1
 Datenkonsistenz, 8-33
 Datentypen, 2-9
 DDS starten, 3-2
 Debugging, 2-2, 2-10, 5-13
 DEFAULT.DFR, 6-6
 Deklarationen als Tabelle, 5-8, 7-9, 7-18
 Deklarations-Schlüsselworte, 7-10
 Detektor für eine ansteigende Flanke, 14-12
 Detektor für eine fallende Flanke, 14-13
 Division, 12-1
 Doku-Vorlage auswählen, 8-4
 Doku-Vorlage erstellen, 8-3
 Dokumentieren, 6-13
 Download-Information, 6-9
 Drag & Drop, 6-25
 Druckbereiche, 5-10
 Drucken, 6-6

E

Editor, 5-7
 Eigenschaften, 6-28, 7-48, 8-27, 8-37
 Ein-/Ausgabevariablen, 7-10
 Ein-/Ausgänge, 13-5
 Einelementige Variablen, 7-19
 Einfügemodus, 9-2, 9-5
 Einfügen, 7-3, 7-35
 Einfügen an Baustein, 7-38
 Einfügen danach, 7-60
 Eingabe, 9-16
 Text für Tooltip, 9-17
 Eingabehilfe, 7-6
 Eingabevariablen, 7-10
 Eingang, 7-33, 7-45
 Eingangsaktion hinzufügen, 7-59
 Eingruppierung der Codestellen, 8-8
 Einläufige Datenverarbeitung, 8-33
 Einloggen, 6-32
 Eins vor, 7-53
 Eins zurück, 7-53
 Einstellungen, 9-10
 Einstellungen Dokumentation, 6-6
 Einzelschritt, 2-10
 Einzelschritt in, 6-36
 Einzelschritt über, 6-35
 Einzelschrittverarbeitung, 7-67
 Einzelzyklus, 2-10, 6-36
 Element einfügen, 8-27
 Element ersetzen, 8-28
 Elementliste, 9-8
 Elementnummer, 9-5
 Ellipse, 9-3
 EN/ENO, 7-47
 Ereignisgesteuerte Task, 8-32
 Ersetzen, 7-5
 EVA-Prinzip, 8-33
 EVENT, 8-32
 Exponentialfunktion, 14-5
 Exportieren, 6-14



F

- Farben, 5-10, 9-12
- Farbwechsel, 9-14
- Fehlermeldungen des Compilers, 15-11
- Form, 9-11
- Funktion, 2-2, 7-23
- Funktion CheckBounds, 2-3
- Funktionsblock, 2-4, 7-23
- Funktionsleiste, 5-9
- Funktionsplan (FUP), 4-19
- Funktionsplaneditor, 4-20

G

- GDC Gerätebeschreibung, 5-17, 5-18
- Gleichheit, 12-11
- Global Ersetzen, 6-19
- Global Suchen, 6-18
- Globale Konstanten, 8-2
- Globale Variablen, 8-2
- Größer als, 12-10
- Größer oder gleich, 12-11
- Gruppieren, 9-21

H

- Hintergrund Bitmap auswählen, 9-11
- Hintergrund Bitmap löschen, 9-11

I

- IEC 61131, 4-1
- IEC-Schritte benutzen, 7-64
- IEC1131-Codestellen, 8-8
- Importieren, 6-15
- In Pin Out Pin, 7-46
- Initialisierungswerte, 8-5
- Instanz, 6-29, 7-55
- Instanz Öffnen, 7-35
- Instanz öffnen, 7-41
- Instanz-Parametermanager, 8-13
- Instanzen von Funktionsblöcken, 2-4
- Interrupt Task, 8-32
- INTERVAL, 8-32

- IP-Adresse, 6-40
- Isotropisch, 9-18

K

- Kanalerstellung zum Gateway-PC, 6-41
- Kanalparameter, 8-30
- Kennworte, 5-14
- Kleiner als, 12-11
- Kleiner oder gleich, 12-11
- Knoten Expandieren, 6-25
- Knoten Kollabieren, 6-25
- Kommentar, 7-34, 7-46
- Kommentare, 5-13
- Kommunikations-Parameter, 6-39
- Kommunikationsparameter, 6-38
- Kommunikationstreiber, 6-38
- Konfigurieren, 9-11
- Konkatenation, 14-8
- Konstanten, 5-13, 7-10
- Kontakt, 7-37
- Kontaktplan (KOP), 4-21
- KOP als FUP, 7-39
- Kopieren, 6-17, 7-2, 7-35
- Kopieren mit der Strg-Taste, 9-7
- Kurve, 9-3

L

- Laden, 6-32
- Laden & Speichern, 5-6
- Länge eines Strings, 14-7
- lecsfc.lib, 8-46
- Lenze9300Servo.lib, 8-46
- LenzeDrive.lib, 8-46
- LenzeElectricalShaft.lib, 8-46
- Limitierung, 12-9
- Linienstärke, 9-12
- Linienzug, 9-3
- Local-Gateway, 6-38
- Logarithmus, 14-5
- Logbuch, 2-12, 5-12, 6-43
- Logbuch Laden, 6-44
- Logbuch Speichern, 6-44
- Logbuch-Fenster, 6-43



Drive PLC Developer Studio

Index

Logbuch-Menü, 6-44
lokale Variablen, 7-10
Lösche Aktion/Transition, 7-61
Löschen, 7-3, 7-35

M

Makro, 7-8
Makro, 7-54
Makro expandieren, 7-55
Makro springen, 7-54
Marke, 7-46
Marke zu Parallelzweig hinzufügen, 7-60
Markierung, 5-8
Maximale Kommentargröße, 7-27
Maximumsfunktion, 12-8
Mehrelementige Variablen, 7-19
Meldungen, 5-5
Minimale Kommentargröße, 7-27
Minimumsfunktion, 12-9
Modifikatoren, 4-2, 11-1
Modulnummer, 13-5
Modulo Division, 12-3
Monitoring, 2-11, 3-14, 7-29, 7-67
Monitoring aktiv, 8-40
Monitoring-Fenster konfigurieren, 7-24
Multiplexer, 12-10
Multiplikation, 12-1

N

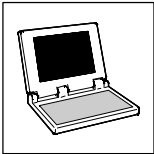
Nach hinten legen, 9-8
Nach vorn bringen, 9-8
Nächster Fehler, 7-8
Nebeneinander, 5-5
Negation, 4-21, 7-34, 7-40
Negieren, 7-47
Netzwerk (danach), 7-27
Netzwerk (davor), 7-27
Neu, 6-1
Neu, mit Vorlage, 6-1
Neue Variable in Deklarationstabelle einfügen, 7-18
Neue Watchliste, 8-39
Neuer Ordner, 6-25

O

Object Organizer, 8-1
Objekt bearbeiten, 6-27
Objekt einfügen, 6-26
Objekt konvertieren, 6-27
Objekt kopieren, 6-27
Objekt löschen, 6-26
Objekt umbenennen, 6-27
Objekte auswählen, 6-24
Öffnen, 6-2
Online-Modus, 5-4
Onlinebetrieb im Sicherheitsmodus, 5-9
Operand, 7-22
Operator, 7-22
Operatoren, 4-2, 11-1
Optionen, 7-63
Optionen für Netzwerkkommentare, 7-27

P

Parallellzweig (links), 7-59
Parallellzweig (rechts), 7-59
Parallel- bzw. Reihenschaltung, 4-21
Paralleler Kontakt, 7-38
Parallellzweig einfügen (rechts), 7-60
Parameter-Monitor, 8-7
Parametermanager, 8-10
 Begriffe, 8-12
Passwörter für Arbeitsgruppen, 6-21
Platzhalter, 6-7, 9-9
Platzhalterkonzept, 9-19
PLC auswählen, 3-2
PLC_PRG, 2-8, 8-32
Polygon, 9-4
Positionsinformationen, 6-11
Potenzierung, 14-7
Pragma, 7-20
PRIORITY, 8-33
Programm, 2-6
Programm ausführen, 9-17
Programmaufruf anhängen, 8-37
Programmaufruf einfügen, 8-37
Projekt, 2-2



Projekt als Bibliothek speichern, 6-3
Projekt erzeugen, 3-2
Projekt Logbuch anzeigen, 6-44
Projekt Logbuch speichern, 6-44
Projekt überprüfen, 6-20
Projekt übersetzen, 6-12
Projekt unter neuen Namen speichern, 6-2
Projekterstellung, 2-1
Projektinformation, 6-17
Projektinformation verlangen, 5-7
Projektstrukturierung, 2-1
Projektvergleich, 6-16
Prozeßabbild, 8-23, 8-24
Pulsgeber, 14-16

Q

Quadratwurzel, 14-5
Querverweisliste ausgeben, 6-30

R

Rahmen, 9-10
Raster, 9-10
Rechteck, 9-2
Registerkarte, 8-1
Registerkarten, 5-2
Reglerfreigabe, 6-41
Reglersperre, 6-42
Remanente globale Konstanten, 8-2
Remanente globale Variablen, 8-2
remanente Konstanten, 7-10
remanente Variablen, 7-10
Remote-Gateway, 6-38
Reset, 6-33, 6-34
Ressourcen, 2-8
Return, 7-32, 7-39, 7-46
Rezeptur lesen, 8-40
Rezeptur schreiben, 8-40
Rückgängig, 7-1

S

Schaltfläche, 9-4
Schließen, 6-2

Schrift, 5-8
Schritt Attribute, 7-61
Schritt Breite, 7-63
Schritt Höhe, 7-63
Schritt-Flag, 4-13
Schritt-Transition (danach), 7-58
Schritt-Transition (davor), 7-58
Selektionsmodus, 9-2
Set/Reset, 7-35, 7-41, 7-47
SFCCurrentStep, 7-66
SFCEnableLimit, 7-65
SFCErrror, 7-66
SFCErrrorPOU, 7-66
SFCErrrorStep, 7-66
SFCInit, 7-65
SFCPause, 7-65
SFCQuitError, 7-65
SFCTrans, 7-66
Sicherheitskopie erstellen, 5-6
Sicherungsintervall, 5-6
Simulation, 2-1, 2-12, 6-37
Sinus, 14-6
Skalierung, 9-13
Software-Semaphor, 14-11
Speicher/Adressverwaltung, 13-6
Speichern, 6-2
Speichern unter, 6-2
Speicherung eines Ereignisses, 8-33
Sprache, 5-9
Sprache, 9-10
Sprache übersetzen, 6-9
Sprung, 7-31, 7-39, 7-46, 7-59
Spule, 7-38
Standard.lib, 8-46
Standard-Schrift, 9-12
Standardkonfiguration, 8-28
Start, 6-33
Statistik, 6-18
Statusleiste, 5-9
Steuerungskonfiguration im Online-Modus, 8-27
Stop, 6-33
Strukturierter Text (ST), 4-4
Subcodestellen, 8-19



Drive PLC Developer Studio

Index

Subtraktion, 12-1
Suchen, 7-4
Symbole anordnen, 5-5
System-Codestellen, 8-8
System-POE, 2-8
Systembausteine, 8-26
Systemfehler, 8-34

T

Tab-Breite, 5-8
Tangens, 14-6
Task anhängen, 8-36
Task einfügen, 8-36
Task-Arten, 8-32
Task-Monitor, 8-31
Tasks
 Ereignisgesteuerte Task, 8-32
 Interrupt Task, 8-32
 PLC_PRG, 8-32
 System-Task, 8-31
 Zeitgesteuerte Task, 8-32
 Zyklische Task, 8-31, 8-32
Tasküberlauf, 8-34
TCP/IP-Netzwerk, 6-40
Teilstring, 14-8
Text, 9-11
Textausgabe, 9-14
Timer off-delay, 14-18
Timer on-delay, 14-17
Tooltip, 5-2, 7-19
Topologisch anordnen, 7-52
Transition-Sprung, 7-59
Typen für Variablendeklaration, 7-14

U

Überführen Steuerungskonfigurationen, 8-28
Überlappend, 5-5
Übersetzen, 6-8
Übersetzungsdatei, 6-11, 6-12, 6-13
Übersetzungsoptionen, 5-13, 5-14
Überwachungszeit, 8-34
Übersetzungsdatei erstellen, 6-12
Unbenutzte Variablen ausgeben, 6-31

Ungleichheit, 12-11
Unsichtbar, 9-14
Untereinander, 5-5
Unterelement anhängen, 8-27
User-Codestellen, 8-8

V

Variable tasten, 9-16
Variable toggeln, 9-16
Variablen, 9-13
Verbindungsmarke, 7-49
Vergleichen, 6-15
Verzeichnisse, 5-11
Visualisierung, 2-9, 9-1, 9-4
Vorheriger Fehler, 7-8

W

Warnungen, 5-10
Watchliste laden, 8-40
Watchliste speichern, 8-39
Watchliste umbenennen, 8-39
Weitersuchen, 7-4
Werte Online verändern, 2-11
Werte schreiben, 6-36
Werte schreiben im FUP, 7-29
Wiederherstellen, 7-1
Winkel, 9-13

X

X-/Y-Position, 9-5
X-Offset/Y-Offset, 9-13

Z

Zeitbegrenzungen editieren, 7-62
Zeitdauer-Flag, 4-13
Zeitenüberblick, 7-62
Zeitgesteuerte Task, 8-32
Zielsprachen, 6-11
Zoom, 7-35, 7-56
Zoom Aktion/Transition, 7-61
Zoomen nach Vis, 9-17
Zugriffsrechte, 5-15, 6-21, 6-28
Zuweisung, 7-31
Zuweisungsoperator, 4-5, 12-13
Zyklische Task, 8-32