Karl Schmitt

SPS-Programmierung mit ST

nach IEC 61 131 mit CODESYS und mit Hinweisen zu STEP 7 im TIA-Portal

Ein Fachbuch von elektrotechnik

Vogel Business Media

Dipl.-Ing. (FH) Karl Schmitt

SPS-Programmierung mit ST

nach IEC 61 131 mit CODESYS und mit Hinweisen zu STEP 7 im TIA-Portal

2., aktualisierte und erweiterte Auflage

Vogel Business Media

Dipl.-Ing. (FH) KARL SCHMITT
Ingenieurstudium, Ausbildung zum Berufsschullehrer, Lehrer an der Technikerschule für Elektrotechnik in Aschaffenburg

*Ein „**SIMATIC STEP 7 Professional V13 SP1, Trial**" im Totally Integrated Automation Portal (TIA Portal) ist 21 Tage nutzbar und steht als Download zur Verfügung.*
http://support.automation.siemens.com/WW/view/de/106448872

*Ein „**SIMATIC STEP 7 Safety Advanced V13 SP1, Trial**" im Totally Integrated Automation Portal (TIA Portal) ist 21 Tage nutzbar und steht als Download zur Verfügung.*
http://support.automation.siemens.com/WW/view/de/108442820

*Ein „**SINAMICS Startdrive V13 SP1, Trial**" im Totally Integrated Automation Portal (TIA Portal) ist 21 Tage nutzbar und steht als Download zur Verfügung.*
http://support.automation.siemens.com/WW/view/de/68034568

*Ein „**SIMATIC WinCC Advanced V13 SP1, Trial**" im Totally Integrated Automation Portal (TIA Portal) ist 21 Tage nutzbar und steht als Download zur Verfügung.*
http://support.automation.siemens.com/WW/view/de/106567563

Weitere Informationen zum TIA Portal finden Sie unter:
http://support.automation.siemens.com/WW/view/de/65601780

Weitere Informationen erhalten Sie im Internet unter
«https://www.siemens.de/sce/promotoren»
«https://www.siemens.de/sce/module»
«https://www.siemens.de/sce/tp»
«https://www.siemens.de/sce»

Print-ISBN: 978-3-8343-3369-8
E-Book-ISBN: 978-3-8343-6200-1

2. Auflage. 2015

Vorwort

ST
SCL

Die Komplexität heutiger Automatisierungsaufgaben ist enorm gestiegen. Es stellt sich die Frage, ob die klassischen Programmiersprachen FBS (FUP), KOP und AWL ausreichend sind, um übersichtliche, leicht wartbare Programme zu schreiben. Mit der Hochsprache "Strukturierter Text" (engl. Structured Text, Abkürzung: **ST**, bei der Firma Siemens **SCL** für "Structured Control Language") ist dies leichter möglich.
Bausteine die in ST bzw. SCL programmiert wurden, können auch in den anderen IEC-Sprachen wie AWL, FBS (FUP) und KOP verwendet werden.

IEC-61131-3

Der Sprachumfang von ST ist in **IEC 61131-3** festgelegt. ST ist eine textuelle Sprache und enthält neben Hochsprachenelementen auch typische Elemente der SPS als Sprachelemente, wie Eingänge, Ausgänge, Bausteinaufrufe usw. Das ST-Programm besteht aus einer Folge von Anweisungen, die mit einem Strichpunkt abschließen. Schleifen, wie FOR... bis REPEAT..., sowie Auswahlanweisungen, wie IF... und CASE..., sind in dieser Sprache möglich.

Dieses Buch wendet sich an SPS-Programmierer sowie Schüler und Studenten an beruflichen Gymnasien, Technikerschulen und Hochschulen, welche die Hochsprache "Strukturierter Text" erlernen wollen. Grundkenntnisse vom Aufbau und Funktion einer SPS sind von Vorteil, aber nicht unbedingt notwendig.
In diesem Workshop werden die Grundlagen dieser Sprache an einfachen Beispielen aus der Praxis erarbeitet.

CODESYS

Die Aufgaben und Übungen können Sie mit **CODESYS** (Controller Development System) Version 2.3 oder Version 3 bearbeiten oder auch mit e!COCKPIT. Entscheiden Sie sich zunächst für eines der drei Programmiertools. Sie können Visualisierungen erstellen und im Simulationsmodus Ihr Programm testen. CODESYS ist ein weit verbreitetes SPS-Programmiersystem nach IEC 61131.

e!COCKPIT

In diesem Buch wird die Demoversion **WAGO-I/O-Pro** – diese basiert auf CoDeSys V2.3, **e!COCKPIT** – eine Entwicklungsumgebung der Firma WAGO, die auf CODESYS V3 basiert, oder die Demoversion CODESYS V3 Demo verwendet. Die Software WAGO-I/O-Pro und CODESYS V3.5 Demo finden Sie auf der Buch-DVD. Die Demoversion von WAGO-I/O-Pro (759-912) sowie e!COCKPIT können Sie auch von der Webseite der Firma WAGO www.wago.de herunterladen. Über der Webseite der Firma 3S-Smart Software Solution GmbH www.3s-software.com ist, nach einer Registrierung, ein kostenloser Download der aktuellen Version von CODESYS V3 möglich.

STEP 7 im TIA-Portal
STEP 7 V5

Da der Sprachumfang genormt ist, können Sie das Gelernte auch in anderen Programmier-Tools, z. B. im SCL-Editor von der Firma Siemens, anwenden. Sie finden Hinweise zum Umgang mit dem SCL-Editor in **STEP 7 im TIA-Portal** sowie in **STEP 7 V5**.

Im Buch finden Sie auch die **Lösungen** zu den Aufgaben und Übungen sowie Auszüge von Datenblättern zu den Baugruppen.

DVD

Die beiliegende **DVD** beinhaltet auch Exportdateien, die Sie in CoDeSys V2.3, e!COCKPIT und CODESYS V3 in Ihr Projekt importieren können, sowie Textdateien, aus denen Sie den Programmcode kopieren und in Ihr Projekt einfügen können. Weiterhin finden Sie notwendige Baustein-Bibliotheken und als Lernzielkontrolle Selbsttests zu den einzelnen Kapiteln sowie eine Archivdatei mit Aufgaben und Übungen für STEP 7 V5 und STEP 7 im TIA-Portal.

Alle Aufgaben und Übungen wurden ausschließlich aus didaktischer Sicht erstellt und können so nicht als Lösungen für Anlagen verwendet werden.

Ich wünsche Ihnen viel Freude beim Erlernen der Hochsprache ST bzw. SCL und bei der Anwendung Ihres neu erworbenen Wissens.

Sulzbach — Karl Schmitt

Inhalt der DVD

Die beiliegende DVD enthält folgende Software:

CODESYS

1. **WAGO-I/O Pro Demo.** (759-912)
   Dieses Enwicklungstool basiert auf **CoDeSys V2.3**. Mit diesem Tool können Sie die Aufgaben und Übungen von Kapitel 1 bis 17 bearbeiten und testen.
   Die Demo ist **zeitlich nicht begrenzt**. Noch viele Zielsysteme von verschiedenen Herstellern, die CoDeSys benutzen, arbeiten mit der CoDeSys V2.3.

2. **CODESYS V3**
   Auch mit diesem Tool sowie e!COCKPIT können Sie die Aufgaben und Übungen bearbeiten und testen. Die CODESYS-Version 2.3 ermöglicht keine objektorientierte Programmierung (OOP). Kapitel 18 ist eine Einführung in die OOP, deshalb kann dieses Kapitel nur mit CODESYS V3 oder e!COCKPIT bearbeitet und getestet werden.
   Eine Steuerungskonfiguration mit I/O-Baugruppen ist mit der CODESYS-V3-Demoversion jedoch nicht möglich. Die CODESYS V3 Demo ist **zeitlich nicht begrenzt**.
   Beachten Sie das Kapitel "Hinweise zum Umgang mit CODESYS V3".

Startseite

3. **Startseite**
   Über die Startseite "Start.htm", die Sie auf Ihrer Buch-DVD finden und in Ihrem Browser, z. B. den Internet Explorer, öffnen können, erreichen Sie zu jedem Kapitel:

   Export- und Textdateien als Vorlagen

   – Links zu **Export- und Textdateien,** die Sie von der DVD herunterladen können.
   Vorgehensweise:
   Erstellen Sie ein neues Projekt mit Ihrem Zielsystem (z. B. 750-881). Laden Sie die Export- oder Textdatei über den jeweiligen Link auf der Startseite herunter (rechte Maustaste–Ziel speichern unter). Importieren Sie die Exportdatei in Ihr Projekt oder fügen Sie den kopierten Programmcode aus der Textdatei in Ihren Projekt-Baustein ein.
   Auf der Startseite steht das A... für Aufgabe und Ü... für Übung.
   Eine Konfiguration der Steuerung, z. B. mit DI-, DO-, AI- und AO-Baugruppen (Kapitel 2), ist nur für wenige Aufgaben bzw. Übungen notwendig.
   Fügen Sie, wenn nötig, Bibliotheken über die Bibliotheksverwaltung und Taskkonfigurationen ein.

   Biblio-theken

   – Links zu **Bibliotheken**, die Sie von der DVD herunterladen und in CODESYS in Ihr Projekt einbinden können.

   Selbsttests

   – Links zu **Selbsttests**, die denen im Buch entsprechen. Achten Sie darauf, dass Ihr Browser aktive Inhalte zulässt.

   – Links zu **weiteren Infos** und **Hilfen**.

   SCL-Dateien

   – Einen Link zur **SCL-Archivdatei** für **STEP 7 v5x** mit allen SCL-Quellen zu den Kapiteln. Laden Sie diese Datei herunter und dearchivieren Sie diese im SIMATIC-Manager. Sie finden im Datei-Ordner "STEP7_TIA_V11" Aufgaben und Übungen, die Sie in **STEP 7 im TIA-Portal** ab V11 öffnen können.

Inhaltsverzeichnis

1 Anweisung, Berechnungen

Lernziele: – Einfaches Programm in der Sprache "ST" schreiben und testen
– Programmorganisationseinheiten (POEs), Variable, Anweisungen, Operatoren, Visualisierung

Aufgabe 1.1 Die Scheinleistung, das Produkt aus Strom und Spannung, soll mit Hilfe eines Software-Bausteins berechnet werden. Die Messwerte I und U werden über den PC mit Hilfe einer Visualisierung eingegeben.

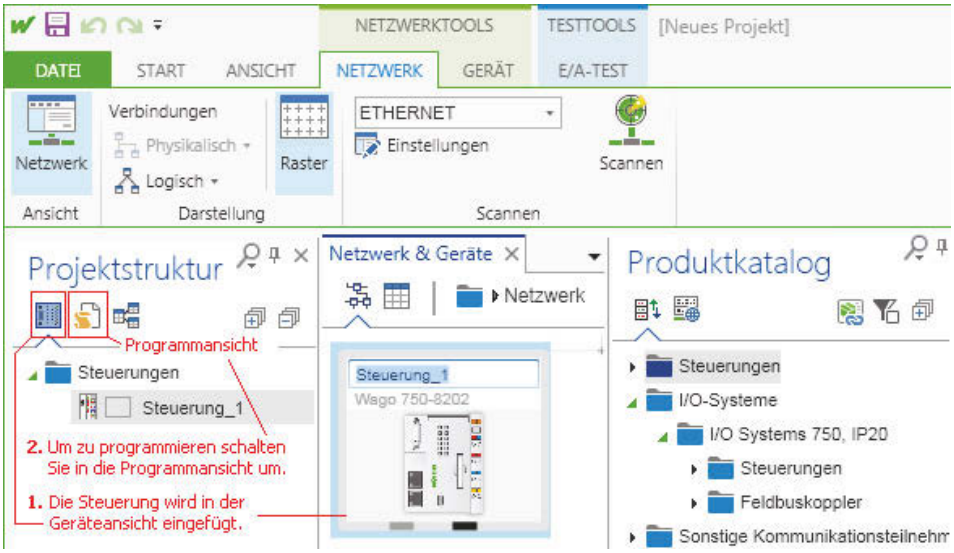
Vollziehen Sie das gezeigte Beispiel nach.
Entscheiden Sie sich zunächst, in welcher der drei Anwendungen Sie die Aufgabe lösen wollen.

Vorgehensweise:

1. **Neues Projekt:**
   In **CoDeSys V2.3**
   Starten Sie die Anwendung und **wählen Sie** im Menü Datei–Neu... **Konfigurieren Sie** als Zielsystem z. B. den Controller (SPS) WAGO_750-881_Demo.

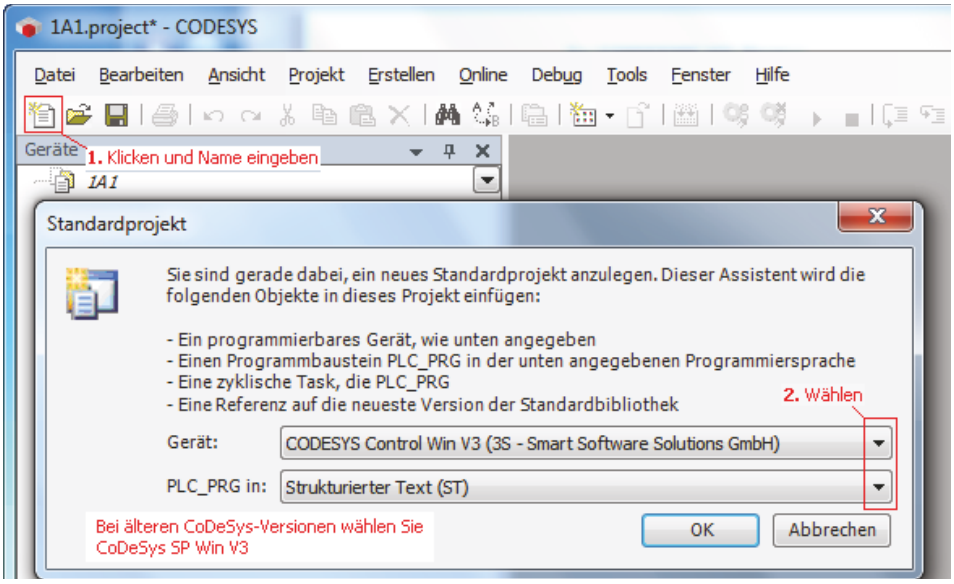
Neues Projekt

In e!COCKPIT (CODESYS V3), neues Projekt
Starten Sie die Anwendung **e!COCKPIT** und **wählen Sie** in der Produktserie 750… als Gerät z.B. einen PFC200 und für den Programmbaustein PLC_PRG die **Sprache ST** aus.



In CODESYS V3 Demo neues Projekt

Starten Sie die Anwendung, fügen Sie ein neues Standardprojekt ein, wählen Sie als programmierbares Gerät (Zielsystem) CODESYS Control Win V3, die Soft PLC(SPS).



PLC_PRG

MAIN

zyklisch

2. Software-Baustein einfügen:

Der Baustein PLC_PRG wird vom Automatisierungssystem, z.B. dem Ethernet-Controller 750-881, **freilaufend zyklisch** aufgerufen. Die Anweisungen im PLC_PRG werden sequenziell (nacheinander) abgearbeitet. Andere Zielsystemhersteller bezeichnen diesen Baustein auch als **MAIN** (Hauptprogramm). Nach IEC gibt es drei Programmorganisationseinheiten (POEs bzw. POUs): das Programm, den Funktionsblock (FB) und die Funktion (FC). FBs und FCs lernen Sie ab Kapitel 8 kennen.

In CoDeSys V2.3 Baustein einfügen

Wählen Sie den Baustein **PLC_PRG**, den Typ Programm und stellen Sie die Sprache auf ST, wie das nachfolgende Bild zeigt.

In e!COCKPIT und in CODESYS V3 Demo Baustein einfügen

In diesen Anwendungen wird der Programmbaustein PLC_PRG automatisch eingefügt.

3. Variable deklarieren und Anweisungen schreiben:

Variable deklarieren

REAL

Präfix r

Anweisung

Ausdruck

Zuweisung

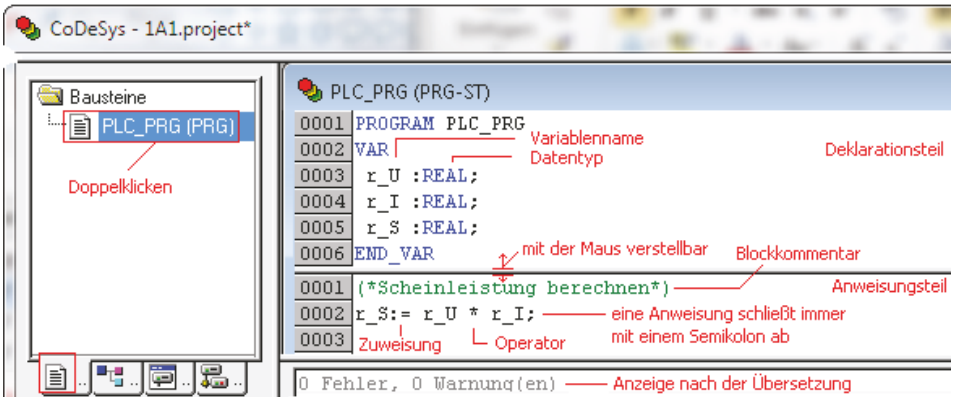
Operator
+ – * /

Kommentar

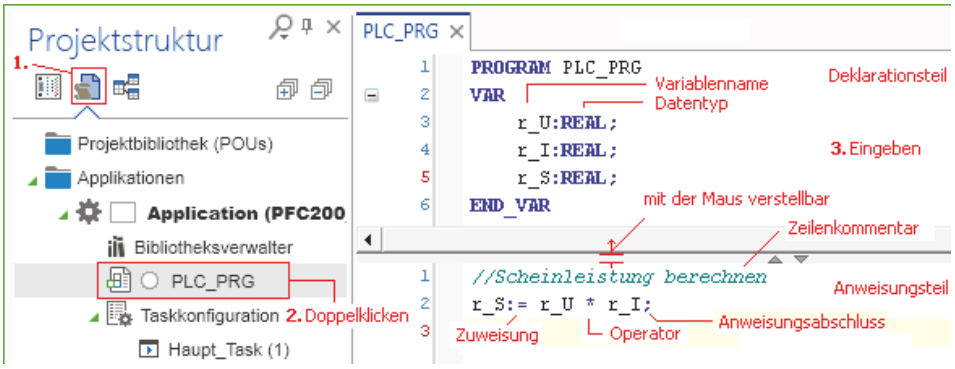
Wie Sie den folgenden Bildern entnehmen können, werden im oberen Teil des PLC_PRG-Fensters, dem Deklarationsteil, die Variablen deklariert, also "Platzhalter" für die Messwerte U, I und das Ergebnis S erstellt. Die Bezeichner, die Variablennamen, wie r_U ... und der Datentyp, hier REAL, werden zwischen den Schlüsselwörtern VAR und END_VAR eingetragen. Das Präfix r zeigt, dass die Variable vom Datentyp REAL ist. REAL heißt, die Variable kann den Wert von $+/-1.18*10^{-38}$... $3.4*10^{38}$ annehmen.

Im unteren Teil des Fensters, dem Anweisungsteil, werden die Anweisungen geschrieben, also die Anweisungen zur Berechnung der Scheinleistung. Das Ergebnis des Ausdrucks r_U * r_I wird der Variablen mit dem Namen r_S zugewiesen. Das Zuweisungszeichen ist :=.
Beachten Sie: Die Wertzuweisung erfolgt immer von rechts nach links. Eine Anweisung wird mit einem Semikolon abgeschlossen. Leerzeichen können zwischen Bezeichner und Operatoren eingefügt werden.
Über die Tastatur kann ein Operator (+, -, *, /) eingefügt werden. Beachten Sie die Prioritäten von Operatoren, Sie finden diese im Anhang Operatoren.
Ein Kommentar unterstützt die Lesbarkeit eines Programms, der Blockkommentar steht zwischen den Zeichen (* und *), der Zeilenkommentar nach den //, dieser ist nur in CODESYS V3 einfügbar.

In CoDeSys V2.3 Variable deklarieren und Anweisungen schreiben



In e!COCKPIT oder CODESYS V3 Variable deklarieren und Anweisungen schreiben



Übersetzen

4. Programmcode übersetzen:

Um ein lauffähiges Programm zu erzeugen, muss der Programmcode übersetzt werden. Achten Sie darauf, dass keine Fehlermeldungen ausgegeben werden. Korrigieren Sie eventuell angezeigte Fehler, z. B. ein fehlender Strichpunkt.

In CoDeSys V2.3: Wählen Sie im Menü Projekt–Alles übersetzen.

In e!COCKPIT: Wählen Sie im Menü Programm–Übersetzen.

In CODESYS V3 Demo: Wählen Sie im Menü Erstellen–Übersetzen.

5. Ein-/Ausgabe, Visualisierung:

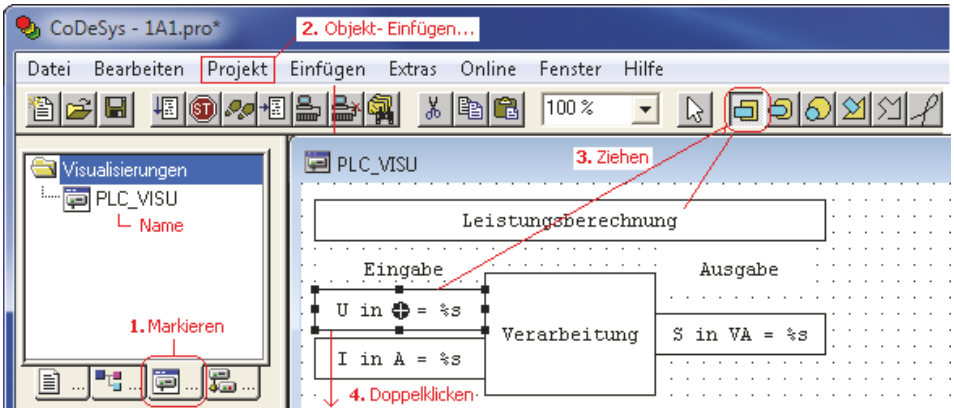
Mit Hilfe einer Visualisierung wird eine Ein- und Ausgabemöglichkeit der Variablen geschaffen. Sie können so auch das Programm leichter testen.

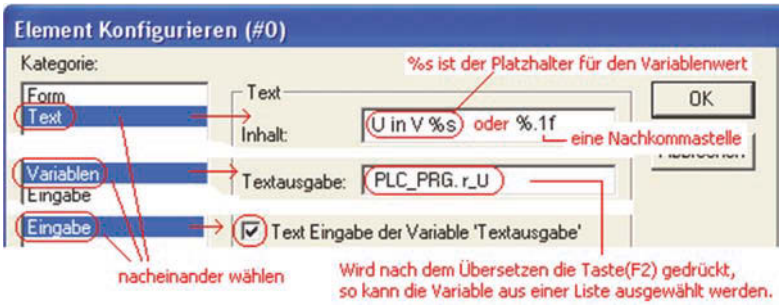
Visualisierung

In CoDeSys V2.3 Vorgehensweise, um die Visualisierung zu erstellen:

1. Im Projektbaum Visualisierung auswählen, siehe folgendes Bild.
2. Visualisierungsfenster einfügen–Menü Projekt–Objekt–Einfügen...Name, z. B. PLC_VISU vergeben. Beachten Sie auch dazu das Kapitel 17.
3. Grafik-Elemente einfügen, z. B. das Eingaberechteck für den Variablenwert r_U.
4. Doppelklicken Sie das eingefügte Eingaberechteck und konfigurieren Sie dieses. Kopien können Sie für die weiteren Ein-/Ausgaberechtecke einfügen und umkonfigurieren, z. B. für den Wert der Variablen `r_I` usw.

Visualisierungseditor V2.3



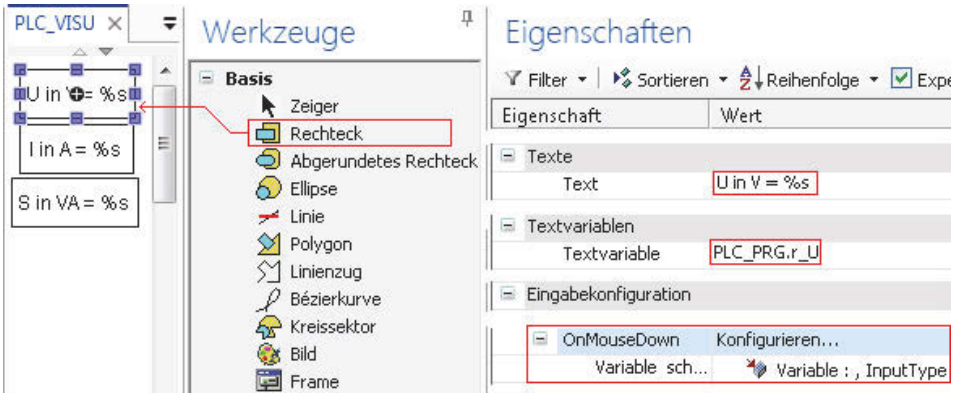


Siehe auch CODESYS-Menü Hilfe–Inhalt–Index–Visualisierung.

In e!COCKPIT oder CODESYS V3 Visualisierung erstellen
Vorgehensweise:

1. Fügen Sie bei markierter Application in der Projektstruktur über das Kontextmenü das Visualisierungselement ein.
2. Wählen Sie für die Eingabe des Variablenwertes von r_U im Werkzeugfenster unter Basis das Rechteck aus und ziehen es in das Arbeitsfenster.
3. Konfigurieren Sie das Eingaberechteck. Sie können das konfigurierte Rechteck kopieren und einfügen und umkonfigurieren für die Variablenwerte r_I und r_S.

Visualisierungs-editor V3



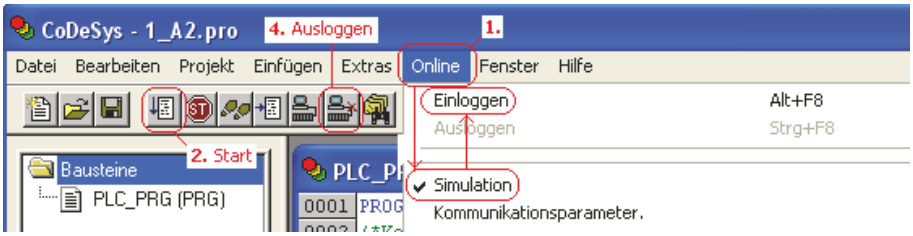
6. Testen mit der PC-Simulation:

Da die Programmorganisationseinheit (POE bzw. POU) PLC_PRG freilaufend zyklisch vom Betriebssystem aufgerufen wird, wird auch der Wert der Variablen r_S immer wieder neu berechnet.

In CoDeSys-Fenster V2.3 testen mit der Simulation und der Visualisierung
Vorgehensweise:

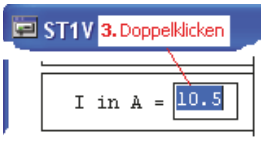
1. Menü **Online**–Simulation mit Häkchen und im gleichen Menü **Einloggen**.
2. Menü Online–**Start**. Die Anweisungen im Programm werden jetzt abgearbeitet.

Online Simulation Ein-/Aus-loggen



3. Im Visualisierungsfenster können die Werte der Variablen U, I durch Anklicken verändert werden.
   Beachten Sie:
   An Stelle eines Kommas muss ein **Punkt** eingegeben werden.

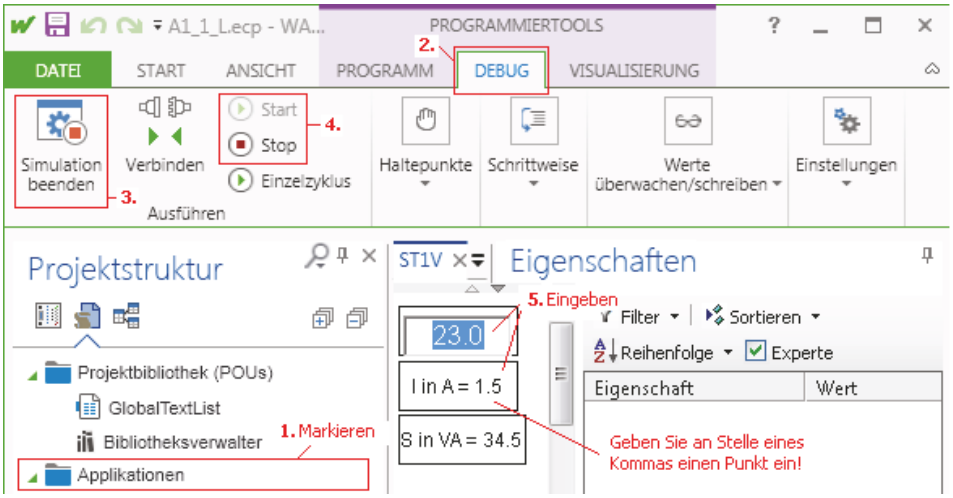
Punkt



4. Um den Code weiter bearbeiten zu können, müssen Sie über Menü Online– wieder ausloggen.

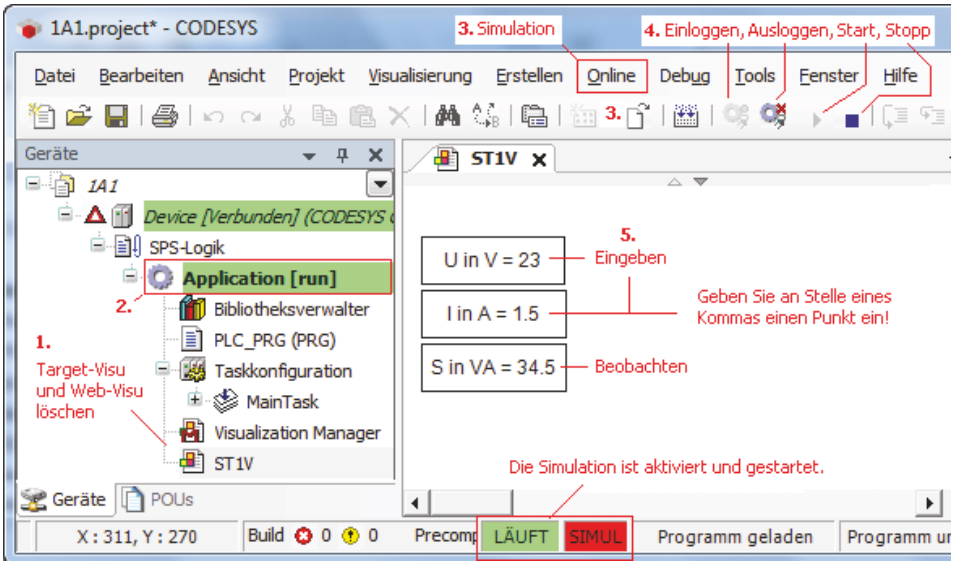
In e!COCKPIT (CODESYS V3) testen mit der Simulation und der Visualisierung:

Markieren Sie die Applikation, wählen Sie im Menü DEBUG- Simulation starten und Start. Geben Sie in der Visualisierung die Werte für U und I ein, wie dies das nachfolgende Bild zeigt. Beobachten Sie die Werte von S.



In CODESYS V3 Demo testen mit der Simulation und der Visualisierung:

Löschen Sie in der Projektstruktur die Objekte Target-Visualisierung und Web-Visualisierung. Markieren Sie die Applikation, aktivieren Sie im Menü Online die Simulation, loggen Sie ein und dann im Menü Debug- Start. Geben Sie in der Visualisierung die Werte für U und I ein, wie dies das nachfolgende Bild zeigt. Beobachten Sie die Werte von S.

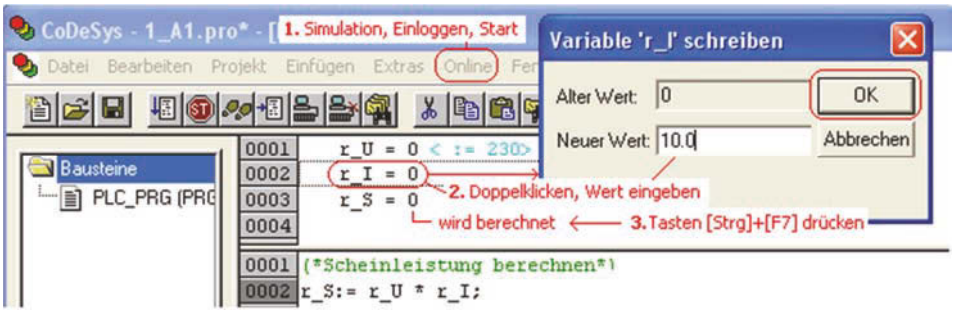


Sie können auch ohne eine Visualisierung Ihr Programm testen, wie das folgende Bild zeigt.

In CoDeSys V2.3 testen – nur mit der Simulation:

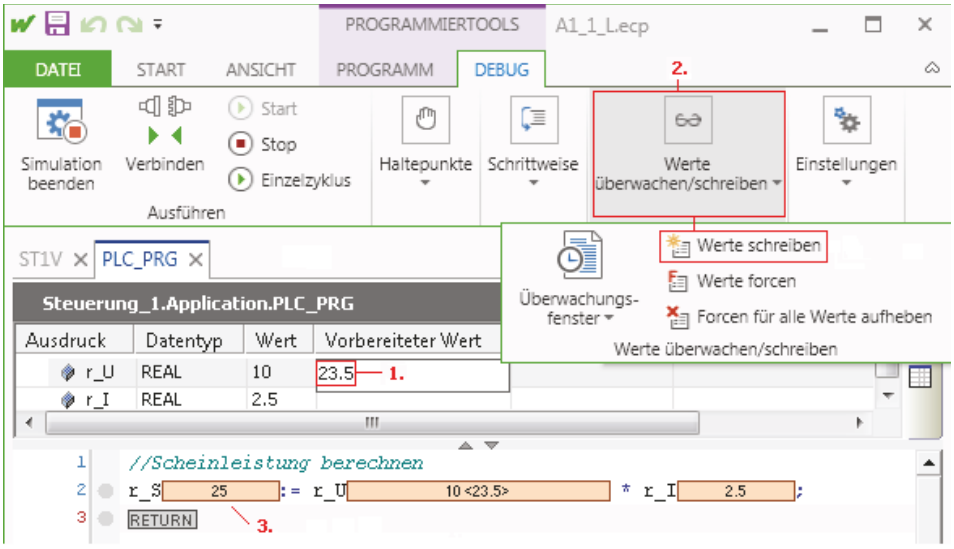
Doppelklicken Sie nach dem Aktivieren der Simulation, dem Einloggen und Start im Deklarationsteil die Variable, geben Sie den Wert ein und wählen Sie im Menü Online–Werte schreiben. Überprüfen Sie das Ergebnis von `r_S`.

Werte schreiben



In e!COCKPIT testen – nur mit der Simulation:

1. Geben Sie nach Simulation starten den Wert in die Spalte – Vorbereiteter Wert – ein.
2. Wählen Sie im Menü DEBUG–Werte schreiben.
3. Überprüfen Sie das Ergebnis.



In CODESYS V3 Demo testen – nur mit der Simulation:

Aktivieren Sie im Menü Online die Simulation und wählen Sie dann im Menü Online–Einloggen. Wählen Sie im Menü DEBUG–Start, geben Sie in der Spalte „Vorbereiteter Wert" Werte ein (siehe Bild oben) und im Menü DEBUG–Werte schreiben. Überprüfen Sie das Ergebnis in `r_S`.

Aufgabe 1.2 Kompensationsanlage

Die Scheinleistung S, die Blindleistung Q sowie der Leistungsfaktor λ sollen mit Hilfe eines Software-Bausteins berechnet werden.
Die Messwerte U, I und die Wirkleistung P werden über den PC mit Hilfe einer Visualisierung eingegeben, S, Q und λ werden berechnet und angezeigt.

$$S = U * I; \quad Q = \sqrt{S^2 - P^2}; \quad \lambda = \frac{P}{S}$$

Da nicht für alle Rechenoperationen Operatoren wie +, -, *, /... zur Verfügung stehen, werden **IEC-Operatoren** bzw. **Standard-Funktionen** verwendet. Sie finden eine Auflistung im Anhang. Benutzen Sie zur Berechnung der Wurzel die Funktion SQRT(...) und für das Quadrat die Funktion EXPT(...,2). Siehe auch CODESYS–Hilfe–Index ... SQRT bzw. EXPT.

Vorgehensweise:

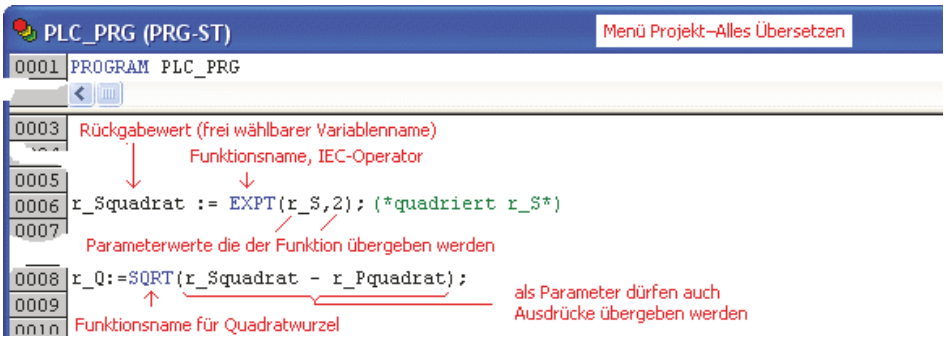
1. Benutzen Sie das vorhergehende Projekt als Vorlage.
2. **Deklarieren Sie** die neuen Variablen und **ergänzen Sie** den Code. Bei mehreren Anweisungen ist es sinnvoll, den Code öfter zwischendurch zu **übersetzen**, um **Syntax-Fehler** (Abweichungen von den Sprachvereinbarungen) leichter zu finden. Beim Übersetzen wird der Maschinencode für das Zielsystem, z. B. den Controller 750-881, erzeugt.

Übersetzen

Syntax-Fehler

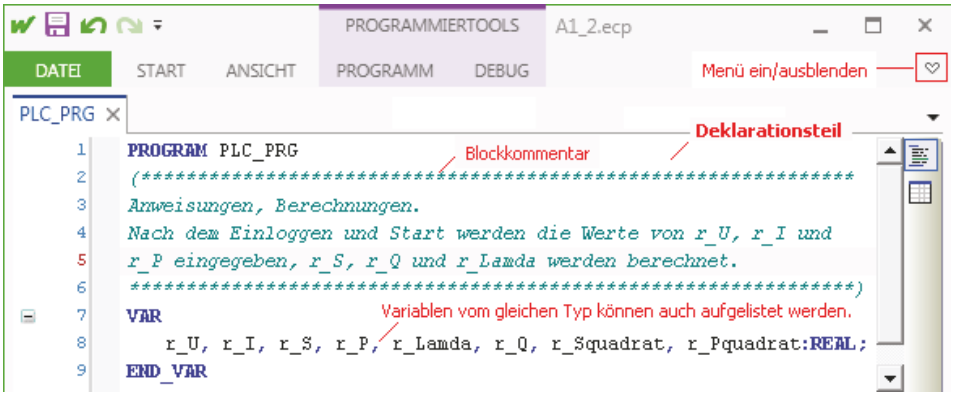
3. **Ergänzen Sie** die Visualisierung mit den weiteren Ein- und Ausgabemöglichkeiten.
4. **Testen Sie** die Funktion des Programms, siehe Aufgabe 1.1.
   Beispiel: r_U = 230; r_I = 10,5; r_P = 1500 ergibt r_S = 2415; r_lamda = 0,621...;
   r_Q = 1892,677

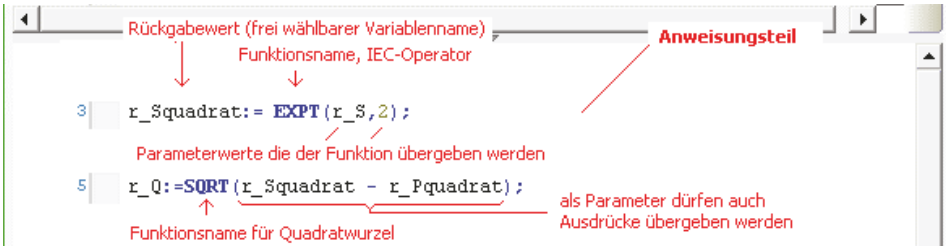
In CoDeSys V2.3



SQRT()
EXPT(,)

In e!COCKPIT

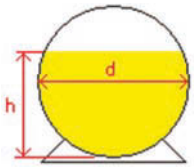




In CODESYS V3

Geben Sie den Code wie im vorhergehenden Bild ein und wählen Sie im Menü Erstellen–Übersetzen.

Übung 1.1 **Volumenberechnung**



Das Füllvolumen eines kugelförmigen Tanks, der mit einer Flüssigkeit gefüllt ist, soll berechnet werden. Das Füllvolumen V ist abhängig von der Füllhöhe h und dem Durchmesser d.

$$V = \pi * h^2 * (d/2 - h/3);$$

Bei d = 2,5 m und h = 1,5 m -> V = 5,29875 m^3

Stichpunkte Nach der Bearbeitung des Kapitels sollten die folgenden Stichpunkte für Sie verständlich sein:

Neues Projekt, PLC_PRG, MAIN, zyklisch, Variable deklarieren, REAL, Präfix, Anweisung, Ausdruck, Operator, Kommentar, Visualisierung, Online-Simulation, SQRT(), EXPT(,), übersetzen, Syntaxfehler

1. Selbsttest – Anweisung, Berechnungen

Wählen Sie die richtigen Ergänzungen bzw. Aussagen aus.

---

1. Ändert sich ein Eingabewert, so wird der Ausgabewert sofort neu berechnet, da der Programmbaustein PLC_PRG

   ☐ bei einer Änderung eines Eingabewertes neu aufgerufen wird.

   ☐ nur einmal aufgerufen wird.

   ☐ zyklisch aufgerufen wird.

   ☐ sequenziell aufgerufen wird.

---

2. Das Zuweisungszeichen im Programm ist das

   ☐ = ☐ ; ☐ (**) ☐ :=

---

3. Eine Variable ist ein Platzhalter für Werte; ihr Wertebereich und somit die Speichergröße wird durch

   ☐ den Variablennamen festgelegt.

   ☐ den Datentyp festgelegt.

   ☐ den Operator festgelegt.

   ☐ die Anweisung festgelegt.

---

4. `r_Q:= SQRT(r_Squadrat - r_Pquadreat);`

   ☐ Der Wert von r_Q wird der Variablen SQRT zugewiesen.

   ☐ Der Wert der Quadratwurzel des Ausdrucks wird der Variablen r_Q zugewiesen.

   ☐ Die Differenz von `r_Squadrat - r_Pquadrat` wird quadriert und in die Variable r_Q geschrieben.

   ☐ Die Anweisung ist nicht übersetzbar, da ein Syntaxfehler vorliegt.

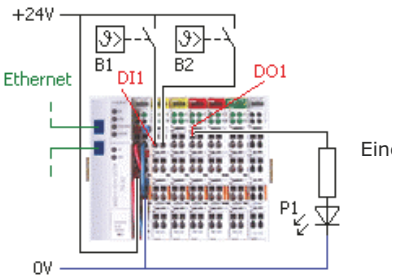
Einen Selbsttest zu diesem Kapitel finden Sie auf der im Buch beiliegenden DVD oder auf InfoClick. Doppelklicken Sie die Datei "Start.htm" und dann den entsprechenden Link.

2 Boolesche Operationen, Steuerungskonfiguration

Lernziele:
- Anweisungen mit logischen Operationen schreiben, Datentyp BOOL
- Steuerungskonfiguration, Zugriff auf Ein- und Ausgänge, Schlüsselwort AT
- Von der Funktionstabelle und vom KV-Diagramm zur Anweisung

Aufgabe 2.1 Antivalenz-Funktion

Es sind zwei Temperatursensoren B1 und B2 in einem Behälter eingebaut und an der SPS angeschlossen. Sind beide Sensoren funktionsfähig, so haben bei Übertemperatur beide Sensoren angesprochen, d.h., deren Kontakte sind geschlossen. Liegt die Behältertemperatur unter dem Grenzwert, so sind beide Kontakte offen. rote Leuchtdiode (LED), am Ausgang der SPS, soll anzeigen, wenn einer der beiden Sensoren defekt ist.
Vollziehen Sie das gezeigte Beispiel nach.
Erstellen Sie dazu ein neues Projekt (Kapitel 1).



Prozessabbild

PAE
PAA

Die Zustände der Eingänge (logisch 1 bzw. 0) werden vom Betriebssystem des Controllers zyklisch in das Prozessabbild der Eingänge (PAE) geschrieben. Die Zustände der Ausgänge werden aus dem Prozessabbild der Ausgänge (PAA) gelesen. Die Prozessabbilder sind Speicherbereiche im Controller. Die Werte im PAE sind über die Adressen %I..., z. B. %IX2.0 vom Programm lesbar, die Werte im PAA werden über die Adressen %Q..., z. B. %QX0.0 vom Programm beschrieben.

Vorgehensweise:

I/Os

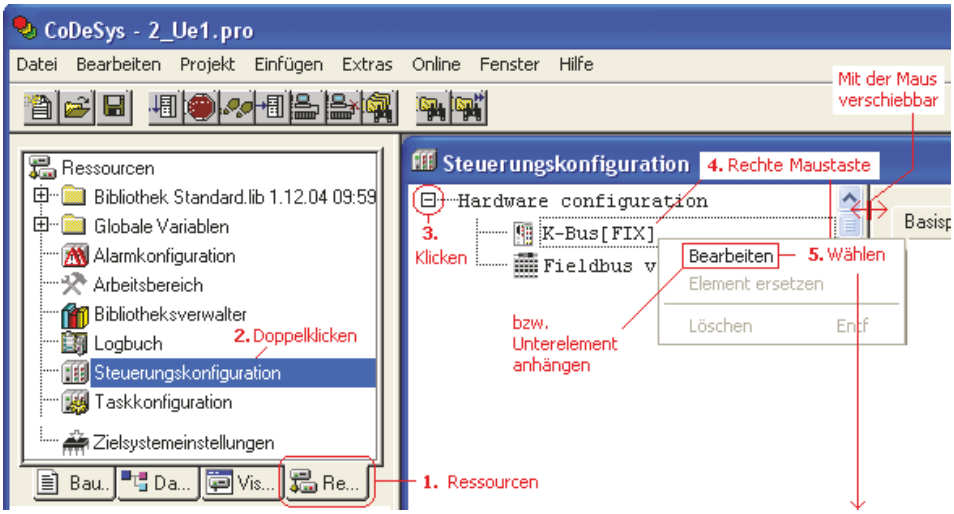
1. **Welche Ein- und Ausgänge (I/Os) stehen bei der SPS bzw. dem Controller zur Verfügung?**

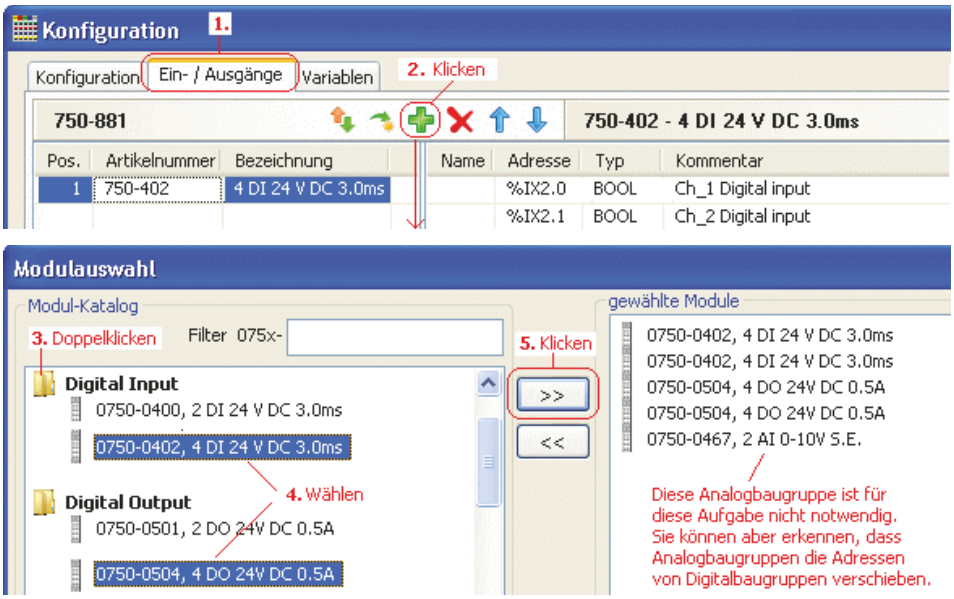
   Konfigurieren Sie die Steuerung mit Digital-Inputs (Eingänge) und -Outputs (Ausgänge) wie im Bild unten beschrieben. Handbuch-Auszüge der Baugruppen finden Sie im Anhang.

In CoDeSys V2.3 Steuerung konfigurieren

Ressourcen

Steuerungs-konfiguration

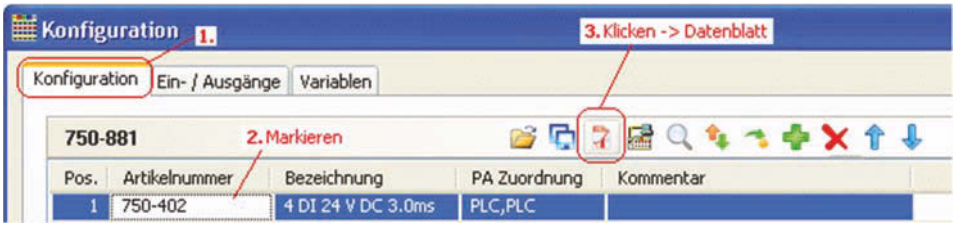




Digital Input

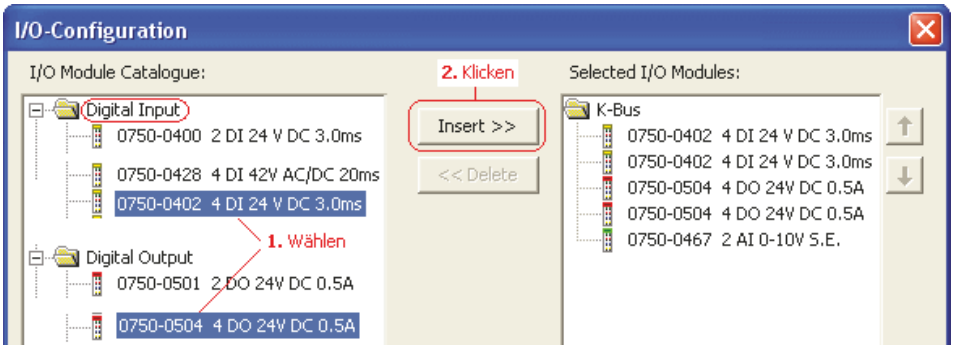
Digital Output

Sie können auch aus dem I/O-Konfigurator auf die **Datenblätter** der Baugruppen zugreifen.



Datenblatt

Falls Sie eine ältere WAGO-I/O-Pro-Version verwenden, wird die I/O-Konfiguration etwas anders dargestellt (Bild unten).



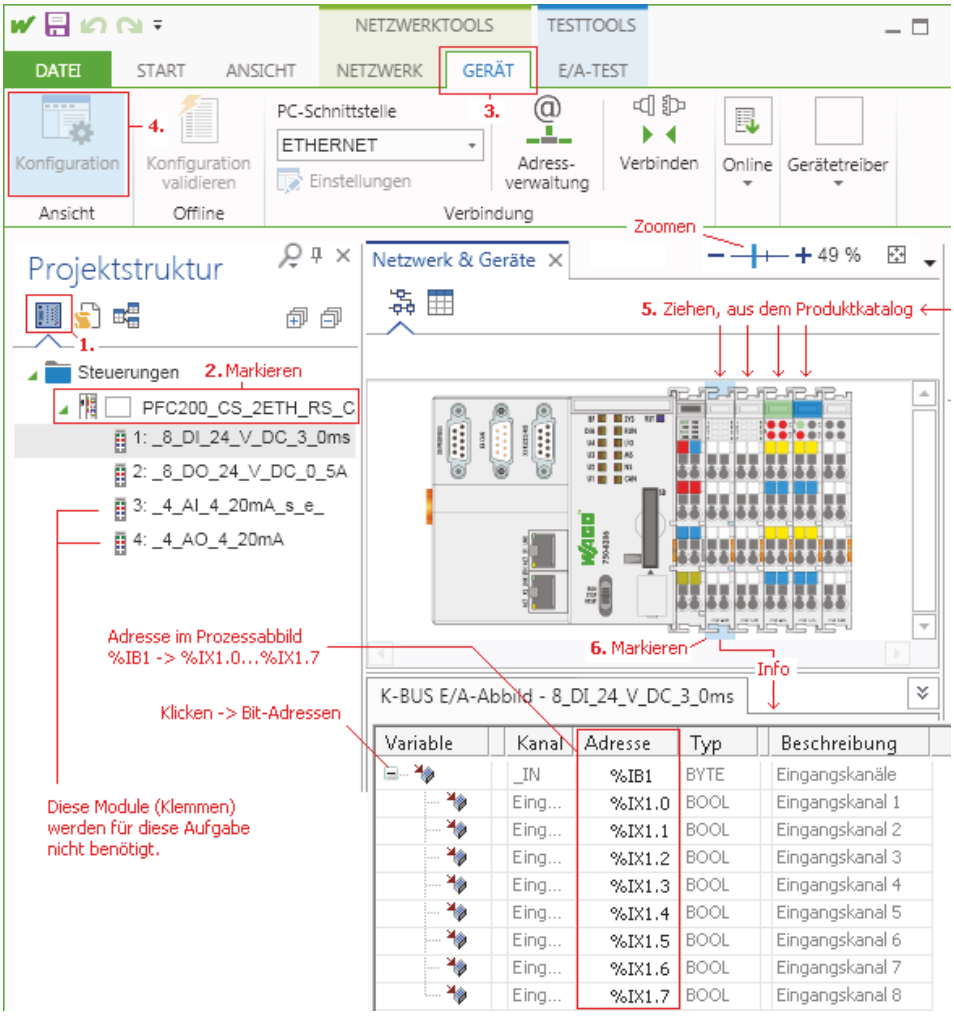
I/O-Konfiguration

Die IEC-Adresse eines Ein- oder Ausgangs ist in der Steuerungskonfiguration ablesbar. Abhängig von den Baugruppen belegen diese bestimmte Adressen.

Anmerkung:
Mit Hilfe der Software WAGO-I/O-CHECK und einer Verbindung zur Steuerung können Sie online die Module ermitteln und in Ihr Projekt übernehmen.

In e!COCKPIT Steuerung konfigurieren:

Markieren Sie die Steuerung und wählen Sie im Menü Geräte-Konfiguration. Ziehen Sie mit der Maus aus dem Produktkatalog I/O-Systeme- 750- Ein- und Ausgangsklemmen zum Controller. Öffnen Sie dazu den jeweiligen Ordner.



In der Geräteansicht und bei markierter Steuerung können Sie auch bei verbundenem Gerät die Konfiguration online ermitteln. Wählen Sie im Menü Gerät-Verbinden und Module ermitteln.

In CODESYS V3 Demo Steuerung konfigurieren:

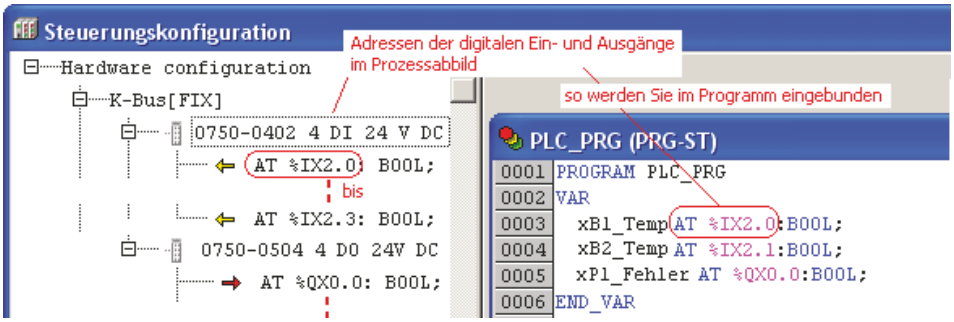
In dieser Demo von der Firma 3S können Sie keine I/O-Baugruppen konfigurieren. Geben Sie das Programm ein und ignorieren Sie beim Übersetzen die Warnung – fehlende I/Os.

2. **Wie werden die Ein-/Ausgänge im Programm eingebunden?**
   Wie werden die Ein-/Ausgänge den Variablen zugeordnet?

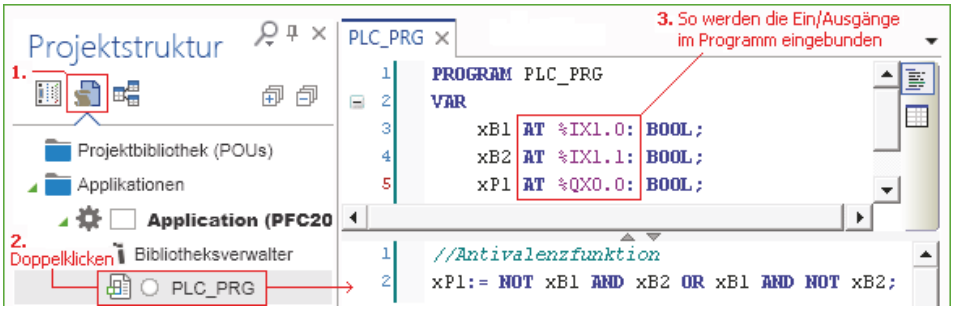
   Mit dem Schlüsselwort `AT` wird der Variablen bei der Deklaration ein absoluter Speicherplatz zugeordnet. Die Adresse, z.B. %IX2.0 oder %IX1.0, entnehmen Sie der Steuerungs- bzw. Gerätekonfiguration.

In CoDeSys V2.3 Variablendeklaration

AT

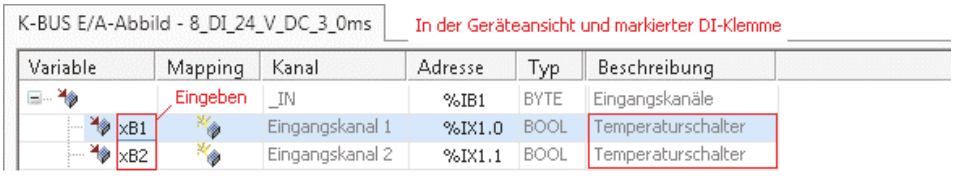


In e!COCKPIT oder CODESYS V3 Variablendeklaration und Programmcode



Mappen

Mappen: Sie können auch die Variable mappen, wenn diese auf I/O-Klemmen zugreifen. Die Deklaration dieser Variablen im Deklarationsteil von PLC_PRG kann dann entfallen.



AND, OR, NOT

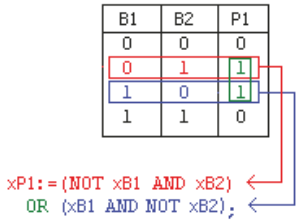
3. Wie sieht der Code in ST aus? Wie lautet die Anweisung?

Mit Hilfe einer Funktionstabelle wird die Anweisung ermittelt (siehe Bild rechts). Es können die Operatoren `NOT, AND, OR` verwendet werden.

BOOL

Beachten Sie:
Der Datentyp der Variablen ist **BOOL**, da ihr Wert nur TRUE (wahr, logisch 1) oder FALSE (falsch, logisch 0) sein kann. Die Bezeichner, die Namen der Variablen, sollten den Präfix x bekommen, z. B. `xB1`... `xP1`.

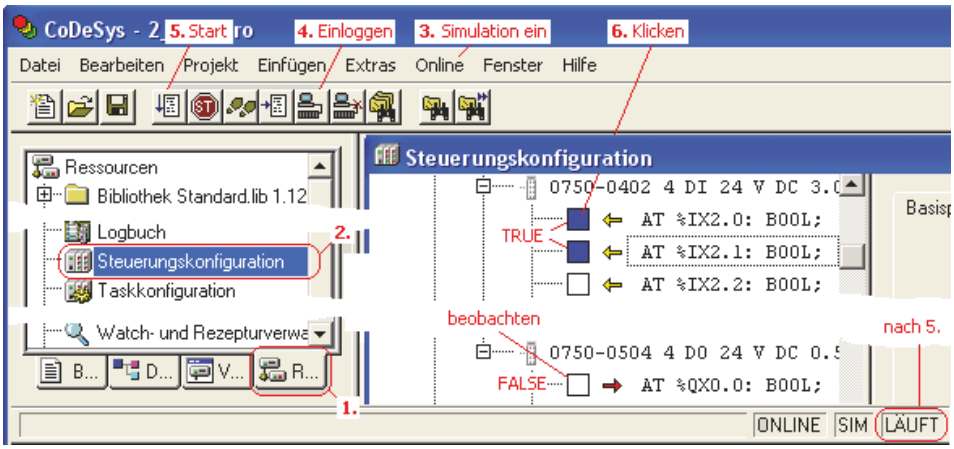
Von der Funktionstabelle zur Anweisung



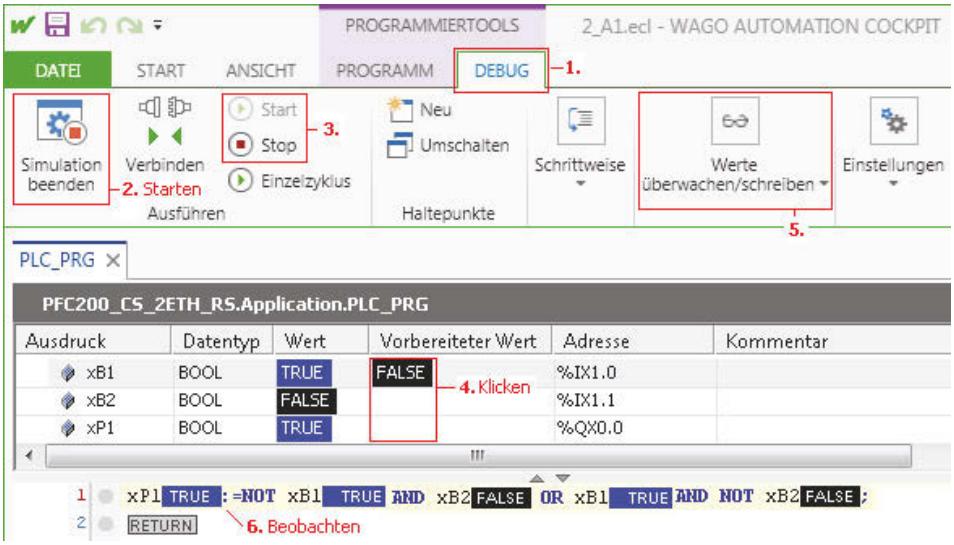
Nachdem Sie die Variablen deklariert haben, geben Sie im Anweisungsteil den Code ein. Returns, Leerzeichen sowie Tabulatoren können Sie zur Strukturierung Ihres Codes einsetzen. Diese Zeichen werden beim Übersetzen nicht gelesen. Den Programmcode zeigt das Bild oben auf dieser Seite. Da die Sprache genormt ist, ist er in allen Entwicklungsumgebungen gleich.

Test

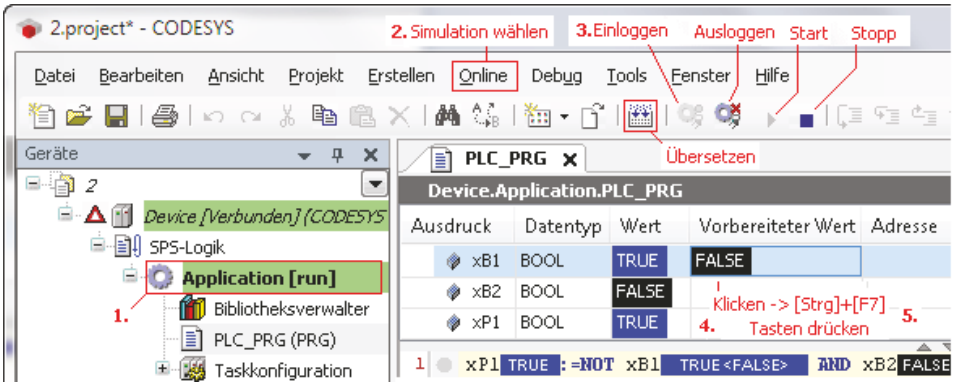
4. In CoDeSys V2.3 testen Sie das Programm, beachten Sie dabei die Reihenfolge.



In e!COCKPIT testen mit der Simulation



In CODESYS V3 DEMO testen mit der Simulation



Falls Sie den Variablen Ein- bzw. Ausgangsadressen zugeordnet haben, so ignorieren Sie nach dem Übersetzen die Meldungen.

5. **Übertragen Sie** Ihr getestetes Programm in den Controller, falls Ihnen einer zur Verfügung steht. Wie Sie die Kommunikation über das Ethernet konfigurieren können, können Sie im Anhang nachlesen.

Übung 2.1 Äquivalenz-Funktion (EQ, equal)

EQ

Eine grüne LED (P2) soll am zweiten Ausgang (%QX0.1) anzeigen, dass kein Sensor defekt ist.

1. **Ergänzen Sie** das Programm, füllen Sie zunächst die Tabelle aus und entwickeln Sie daraus die Anweisung.

B1	B2	P2
0	0	...
0	...	...
1	...	...
1	1	...

```
xP2 := .....
```

2. **Überprüfen Sie** die Funktion. Siehe Beispiel oben.

Übung 2.2 XOR

XOR

Vereinfachen Sie die Aufgabe 2.1 mit dem XOR-Operator. Schreiben Sie die Anweisung auf.
Siehe CODESYS-Menü Hilfe–Suchen–XOR eingeben...

```
xP1_Uebertemp:= .....
```

Übung 2.3 Funktion 2 aus 3

B1	B2	B3	P1
0	0	0	0
0	0	1	...
0	1	0	...
0	1	1	1
1	0	0	...
1	0	1	...
1	1	0	...
1	1	1	1

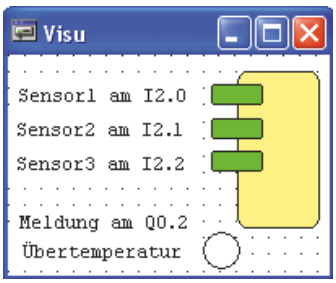
In einem Behälter soll die Temperatur überwacht werden. Es sind drei Temperaturschalter (B1 ... B3) eingebaut. Die Signale der Temperaturschalter sollen so verknüpft werden, dass eine Meldeleuchte (P1) "Übertemperatur" anzeigt, wenn zwei oder alle drei Temperaturschalter angesprochen haben.

Schreiben Sie die Gleichung mit Hilfe der Funktionstabelle auf.
Ergänzen Sie zunächst die Einträge in der Tabelle.

```
P1 := .....
   OR .....
   OR .....
   OR .....
```

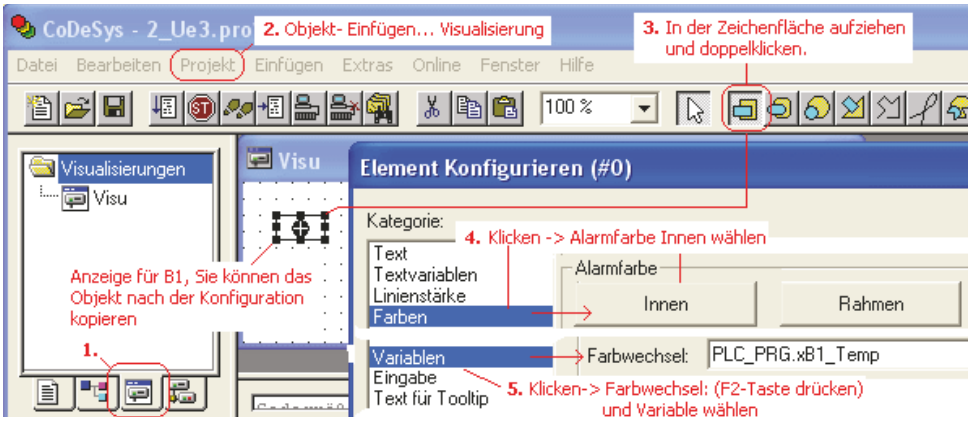
Geben Sie den Code ein.

Erstellen Sie eine Visualisierung – siehe folgendes Bild – und testen Sie die Funktion.



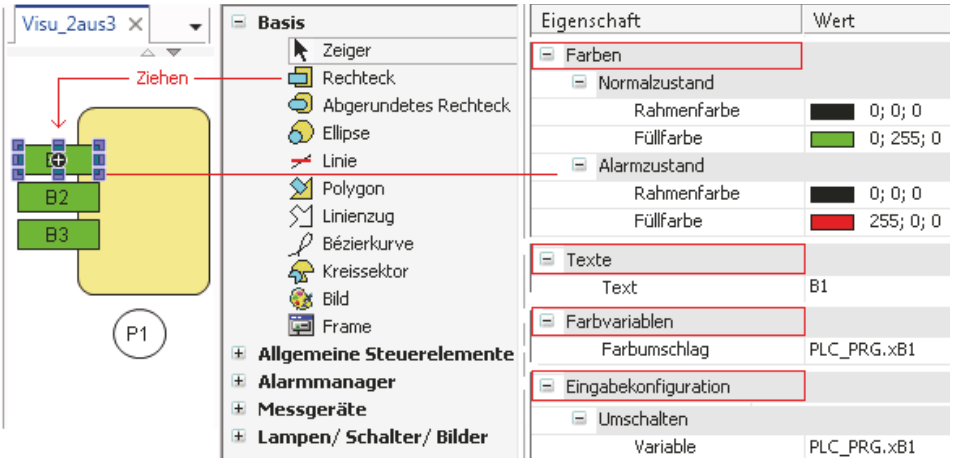
In CoDeSys V2.3 Visualisierung

Visualisierung



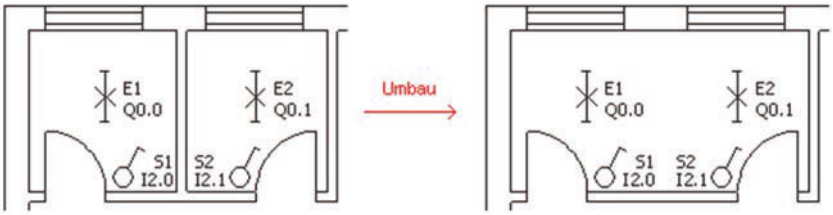
In e!COCKPIT oder CODESYS V3 Visualisierung

Fügen Sie bei markierter Application eine Visualisierung ein.



Übung 2.4 Wechselschaltung

Die Gebäudeinstallation beinhaltet einen I/O-Controller zur Gebäudeautomatisierung.
Sie erhalten den Auftrag, eine bestehende Beleuchtungsanlage für zwei unabhängige Räume umzuprogrammieren, da die Zwischenmauer entfernt wird. Beide Leuchten sollen dann von S1 und S2 schaltbar sein (Wechselschaltung).



S2	S1	E2 = E1
0	0	0
0	1	...
1	0	...
1	1	...

Entwerfen Sie mit Hilfe der Funktionstabelle die Anweisung für das Programm.

E1 := .....

E2 := E1; Testen Sie die Funktion.

Aufgabe 2.2 KV-Diagramm, Funktion 2 aus 3

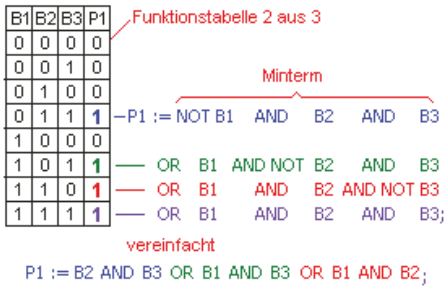
Funktions-tabelle -> Anweisung

Bei Verknüpfungssteuerungen ohne Speicher-verhalten kann mit Hilfe einer **Funktionstabelle** die notwendige Anweisung ermittelt werden.

Bei der disjunktiven Normalform (DNF) werden:

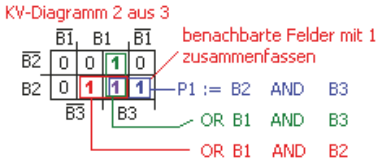
Minterme

- die Eingänge UND-verknüpft, bei denen der Ausgang "1" ist -> es entstehen die Minterme,
- die entstandenen **Minterme** werden ODER-verknüpft,
- die ermittelte Funktionsgleichung wird vereinfacht.



KV-Diagramm

Wird das **KV-Diagramm** benutzt, entfällt die Vereinfachung, da durch die Zusammenfassung von Feldern mit "1" die Funktionsgleichung in minimierter Form ermittelt wird.
Vollziehen Sie den Eintrag im KV-Diagramm nach.



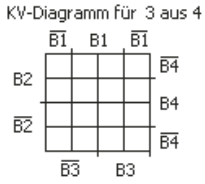
Auf der Info Click-Webseite und auf der DVD finden Sie eine kleine Animation.

Testen Sie die Richtigkeit in CODESYS.

1. Wie viele Ein- und Ausgänge sind nötig? -> Steuerungskonfiguration
2. Geben Sie die aus dem KV-Diagramm entwickelte Gleichung als Anweisung ein.
3. Testen Sie das Programm, simulieren Sie dabei B1 ... B3.
4. Falls Ihnen ein Controller zur Verfügung steht, übertragen Sie Ihr getestetes Programm in den Controller. Siehe Anhang „Programm in den Controller laden".

Übung 2.5 Funktion 3 aus 4

Schreiben Sie mit Hilfe des KV-Diagramms die Anweisung auf.
Testen Sie die Richtigkeit Ihrer Ergebnisse in CODESYS.



P1:= .....

OR .....

OR .....

OR ..... ;

Übung 2.6 **Funktion 2 aus 3 mit Öffnerkontakten**

Drahtbruch

Ändern Sie das Programm, da die **Öffnerkontakte** der Sensoren benutzt werden, um einen **Drahtbruch** zwischen Sensor und der Steuerung erkennen zu können. Ein Drahtbruch wirkt so, als ob der Öffner, z. B. ein Meldekontakt, angesprochen hätte.

In e!COCKPIT

Falls Sie eine Steuerung zur Verfügung haben, so können Sie die Konfiguration der Module wie I/O-Klemmen usw. aus der Steuerung lesen.
Projektieren Sie nur die Steuerung, verbinden Sie die Steuerung mit dem PC und wählen Sie in der Geräteansicht im Menü Gerät-Verbinden und -Module ermitteln.

Stichpunkte Steuerungskonfiguration, Digital-Input, -Output, Datenblatt, PAE, PAA, AT, AND, OR, NOT, BOOL, XOR, Visualisierung, Funktionstabelle -> Anweisung, Minterme, KV-Diagramm, Drahtbruch

2. Selbsttest – Boolesche Operationen

Wählen Sie die richtigen Ergänzungen bzw. Aussagen aus.

---

1. Am Anfang jedes Programmzyklus wird der Speicherbereich des PAA und des PAE aktualisiert. Wurde eine Eingabebaugruppe auf Byte 0 konfiguriert, so kann im Anwenderprogramm eine Variable mit Hilfe des Schlüsselworts "AT" auf den Speicherbereich

   ☐ %QX0.0...%QX0.7 zeigen. ☐ %IX0.0...%IX0.7 zeigen.

   ☐ %IX1.0...%IX1.7 zeigen. ☐ %QX1.0...%QX1.7 zeigen.

---

2. Der Datentyp

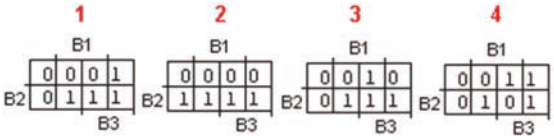
   ☐ INT ☐ REAL ☐ DINT ☐ BOOL

   kann nur die Werte "TRUE" oder "FALSE" annehmen.

---

3. xP1:= xB1 AND xB2 OR xB1 AND xB3 OR xB2 AND xB3;

   Das KV-Diagramm ☐ 1 ☐ 2 ☐ 3 ☐ 4 entspricht der Anweisung.



4.

B2	B1	led_ok
0	0	1
0	1	0
1	0	0
1	1	1

Die Anweisung zur Tabelle lautet:

☐ xLed_ok:=(NOT xB2 AND xB1)OR(xB2 AND NOT xB1);

☐ xLed_ok:=(NOT xB2 OR xB1)AND(xB2 OR NOT xB1);

☐ xLed_ok:=(NOT xB2 AND NOT xB1)OR(xB2 AND xB1);

☐ xLed_ok:=(NOT xB1 AND xB2)OR(xB2 AND NOT xB1);

Einen Selbsttest zu diesem Kapitel finden Sie auf der im Buch beiliegenden DVD oder auf InfoClick.

3 Datentypen, Codierungen

Lernziele: Datentypen, Codes, Zahlendarstellung und Datentypenumwandlung kennen lernen

Einführung: Ein Computersystem kann Daten nur binär (0 und 1) verarbeiten. Um ein Bitmuster richtig zu interpretieren, muss bei der Deklaration einer Variablen ein Datentyp angegeben werden. Damit sind die Codierung, der Wertebereich, die Größe des notwendigen Speichers sowie die gültigen Operationen und Funktionen festgelegt.
Beispiel: Das Bitmuster 1111_1111_1111_1111 wird beim Datentyp "WORD" als 65535, jedoch beim Datentyp INT als -1 interpretiert.

3.1 Datentypen für logische Werte

BOOL
BYTE
WORD
DWORD

Für logische Operationen wie `AND, OR, XOR, NOT` ... werden folgende Datentypen gewählt:

1 Bit	-> **BOOL**	z. B.	`xQ1`
8 Bits = 1 Byte	-> **BYTE**	z. B.	`bySensorgruppe`
16 Bits = 1 Word	-> **WORD**	z. B.	`wMaske`
32 Bits = 1 Doppelword	-> **DWORD**	z. B.	`dwStatus`

Aufgabe 3.1 Sensorgruppen vergleichen

Das Bitmuster der Variablen `bySensorgruppe1` soll mit Hilfe des `XOR`-Operanten mit dem Bitmuster der Variablen `bySensorgruppe2` verglichen werden. Bei Abweichungen zeigt die Variable `byErgebnis` dies an.
Konfigurieren Sie die Steuerung, z. B. mit 2x8DI 750-430 und 1x8DO 750-530, geben Sie den Code ein und **analysieren Sie** das Programm mit der PC-Simulation. %IB1 ist die IEC-Adresse des Eingangsbytes 1, %QB0 die des Ausgangsbytes 0. Schalten Sie die Darstellung der Werte um, wie dies im Folgenden beschrieben ist, um die verschiedenen Darstellungen von Variablenwerten kennen zu lernen. Die Adressen können Sie der Steuerungskonfiguration entnehmen.

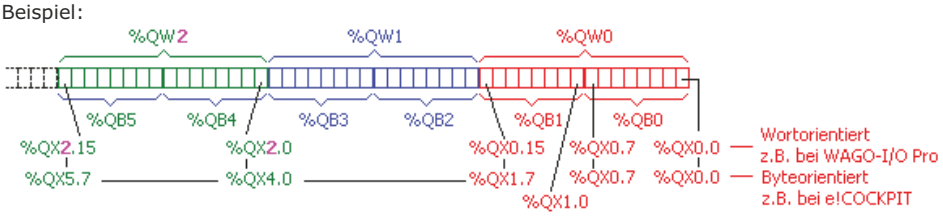
```
PROGRAM PLC_PRG
VAR
 bySensorgruppe1 AT %IB?:BYTE;
 bySensorgruppe2 AT %IB?:BYTE;
 byErgebnis AT %QB0:BYTE;
END_VAR
byErgebnis:= bySensorgruppe1 XOR bySensorgruppe2;
```

Alle Bits der Variablen byErgebnis sind FALSE, wenn beide Sensorgruppen den gleichen Wert haben.

Beachten Sie die **Speicherorganisation**, sie ist vom Zielsystem abhängig.

Speicher-
organisation

Beispiel:



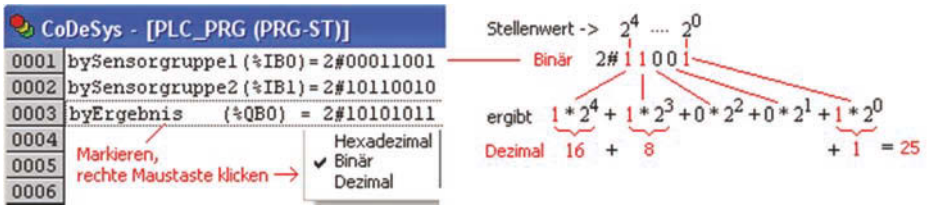
Darstellung der Werte und Codierung

Binärcode: Es verdoppelt sich die Wertigkeit der Bits von rechts nach links.

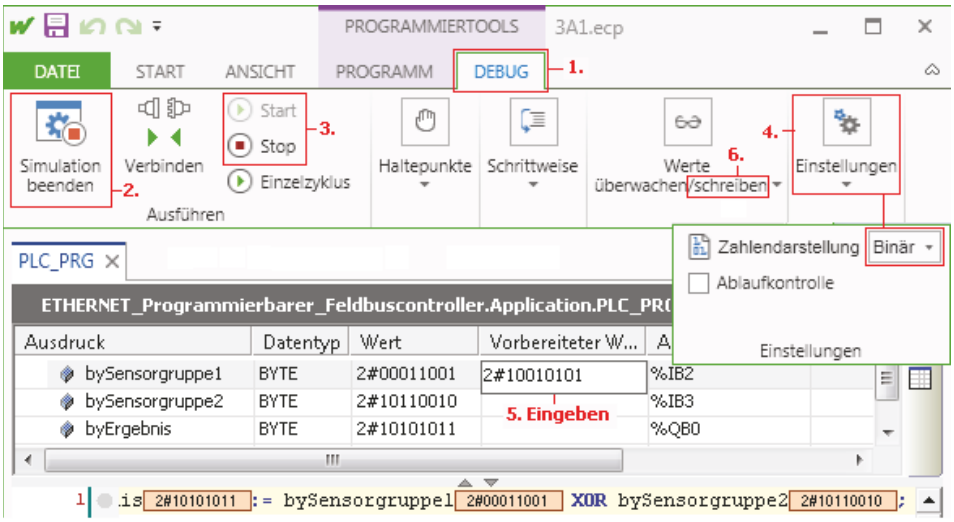
Aktivieren Sie die Simulation, loggen Sie ein und starten Sie die Abarbeitung des Programms. Sie können die Darstellung im Editor umschalten, wie im Bild unten gezeigt wird.

In CoDeSys V2.3 Test und Darstellung

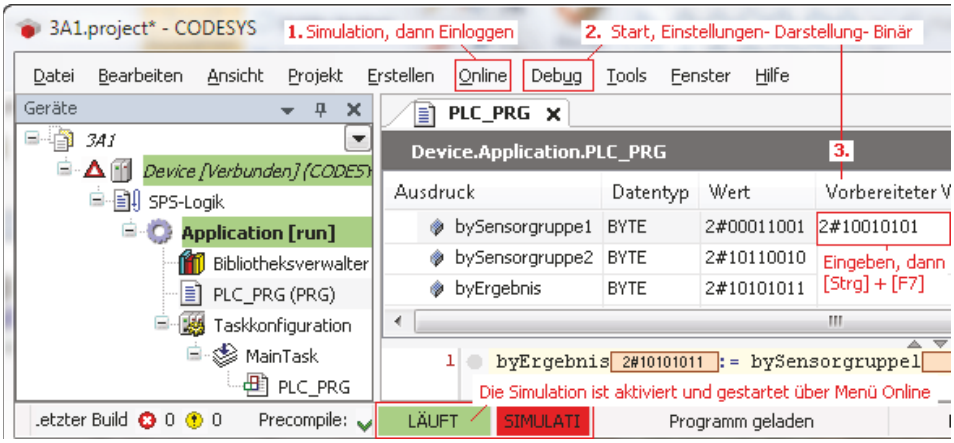
Binärcode
2#...



In e!COCKPIT Test und Darstellung



In CODESYS V3 Demo Test und Darstellung

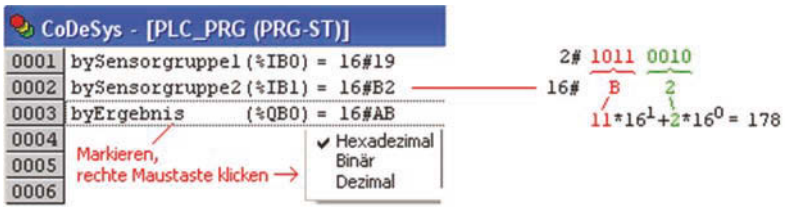


Sedezimalen(Hex)-Code

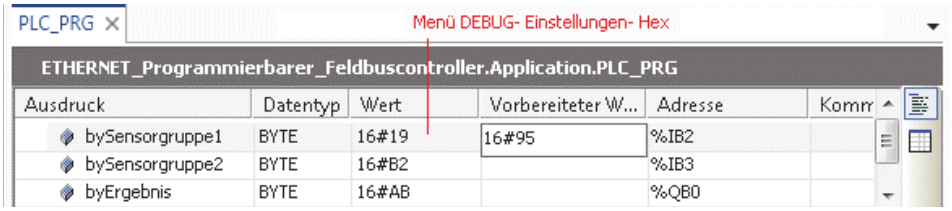
Um vielstellige binärcodierte Zahlen leichter lesen zu können, werden sie oft im Sedezimalen-(Hex)-Code dargestellt. Der Sedezimalcode kennt 16 Zeichen, 0, 1, 2 ... 9, A, B, ... F (A steht für 10 ... F für 15).
Es können dadurch vier Stellen zu einem Zeichen zusammengefasst werden.

In CoDeSys V2.3 Test und Darstellung

Hexcode 16#...



In e!COCKPIT oder CODESYS V3 Demo Test und Darstellung



Übung 3.1 Zahlendarstellung

BYTE

Die Werte der Variablen `bySensorgruppe1` (siehe oben) sollen unterschiedlich dargestellt werden. **Füllen Sie** die Tabelle aus.

Beispiele:

binär	hexadezimal	dezimal
2#1001_0101		
	16#A2	
		19

Steht das Bitmuster im Vordergrund, so sollte die binäre oder die hexadezimal Darstellung gewählt werden.

Aufgabe 3.2 Maske

Maske

WORD

Initialwert

Nur die Bits 0 und 8 ... 11 vom %IW0 (Eingangswort 0, dies ist ein Speicherbereich des PAE) sollen die gleichen Bits vom %QW0 steuern. Die restlichen Bits werden mit Hilfe einer "**Maske**" ausmaskiert. Der Variablen `wMaske` wird ein **Initialwert** zugewiesen. Nach einem Neustart nimmt die Variable dadurch diesen Wert an.
Vollziehen Sie das Beispiel nach.