EINZELSTEUERFUNKTIONEN

LERNZIEL

Die Studierenden können nach der Bearbeitung dieses Moduls den Begriff der Einzelsteuerfunktion im Rahmen der objektorientierten Softwarestrukturierung definieren und einordnen. Sie verstehen das Konzept, den Aufbau sowie die Funktionsweise von Einzelsteuerfunktionen und kennen typische Einzelsteuerfunktionen sowie deren Umsetzung in ***PCS 7***.

THEORIE IN KÜRZE

Das Ziel der objektorientierten Softwarestrukturierung besteht darin, die Struktur der realen Anlage durch eine entsprechende Modularisierung der Anwendersoftware möglichst eindeutig nachzubilden. Dazu wird für jeden Feldgerätetyp mindestens ein Funktionsbaustein bereitgestellt, der die gesamte Ansteuerungslogik, notwendige Schutz- und Überwachungsfunktionen sowie geeignete Bedien- und Visualisierungsmöglichkeiten bereitstellt. Das Anwenderprogramm nutzt diesen Baustein um das gewünschte Betriebsverhalten einer Maschine oder eines Prozesses zu realisieren.

Motoren und Ventile sind steuerungstechnische Einrichtungen, die im Sinne einer objektorientierten Automatisierung ebenfalls nicht direkt angesteuert, sondern zunächst als Funktionsbaustein-Typen modelliert werden. Derartige Funktionsbaustein-Typen werden als ***Einzelsteuerfunktionen (ESF)*** oder ***Individual Drive Functions (IDF)*** bezeichnet. Sie ermöglichen die Steuerung, Überwachung und Bedienung der steuerungstechnischen Einrichtung durch die Bereitstellung von entsprechenden Anschlüssen für Stell- und Steuersignale sowie für Parametrier- und Überwachungsfunktionen. Die technische Umsetzung der Steuerung wird durch eine Instanz des Funktionsbaustein-Typs realisiert und bleibt dem Nutzer verborgen. Abbildung 1 zeigt den Übergang vom realen Motor, in diesem Fall eine Pumpe, hin zu einem Baustein der entsprechenden Einzelsteuerfunktion.

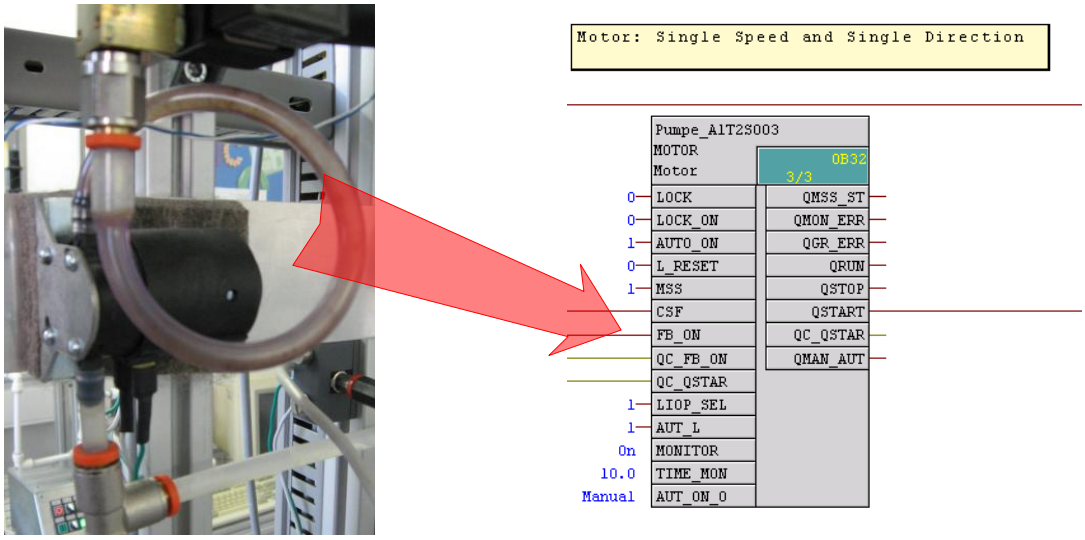


Abbildung 1: Der Übergang vom realen Motor zum Bausteine der Einzelsteuerfunktion

Steuerungstechnische Einrichtungen können grundsätzlich in vier verschiedenen Betriebsarten betrieben werden. Das Gerät kann sich

- ***Außer Betrieb***
- im ***Handbetrieb***
- im ***Automatikbetrieb*** oder
- im ***Vor-Ort-Betrieb***

befinden. Eine Einzelsteuerfunktion darf sich dabei immer nur in genau einer Betriebsart befinden. Die genannten Betriebsarten können gleichwertig oder über Prioritäten hierarchisch gegliedert sein. Einzelsteuerfunktionen bieten zudem Funktionen zum Schutz vor Geräte- und Prozessfehlern an. Dazu werden verschiedene Verriegelungen sowie eine Laufzeitüberwachung für das Gerät und den gesteuerten Prozess implementiert.

Funktionsbaustein-Typen, in ***PCS 7*** als Bausteintypen bezeichnet, stellen vorgefertigte Programmteile für die Bearbeitung wiederkehrender Funktionen dar. Sie können in CFC-Pläne eingefügt werden und dort als Instanzen parametriert, verschalten und an die Projekterfordernisse angepasst werden. Dabei legt der Baustein-Typ die Charakteristik für sämtliche Instanzen dieses Typs fest. ***PCS 7*** bietet bereits eine Vielzahl von leistungsfähigen und getesteten Einzelsteuerfunktionen als Bausteintypen in den leittechnischen Bibliotheken an. Diese modellieren jeweils eine steuerungstechnische Einrichtung und stellen die gesamte Ansteuerlogik bereit. Zusätzlich werden Funktionen angeboten

- zum ***Bedienen*** und ***Beobachten*** der Einzelsteuerfunktion
- zum ***Steuern*** von Signalen
- zur ***Überwachung*** und ***Alarmierung***
- zur ***Betriebszustandsauswahl***
- für ***Verriegelungen***.

Bildbausteine mit verschiedenen ***Sichten*** ermöglichen die nahtlose Integration in ein entsprechendes Prozessleitsystem.

Einzelsteuerfunktionen ermöglichen eine effiziente Entwicklung leistungsfähiger, qualitativ hochwertiger Lösungen. Sie modularisieren und typisieren wiederkehrende Funktionalitäten. Damit werden diese wiederverwendbar und zentral änderbar, was den Entwicklungsprozess erheblich beschleunigt.

THEORIE

OBJEKTORIENTIERTE SOFTWARESTRUKTURIERUNG

Das Ziel der objektorientierten Softwarestrukturierung besteht darin, die Struktur der realen Anlage durch eine entsprechende Modularisierung der Anwendersoftware möglichst eindeutig nachzubilden. Dazu wird für jedes Feldgerät in der Anlage ein eigenes Programm erstellt. Für jeden Feldgerätetyp wird mindestens ein Funktionsbaustein bereitgestellt.

Dieser Baustein realisiert die gesamte Ansteuerungslogik für diesen Feldgerätetyp. Darüber hinaus stellt er notwendige Schutz- und Überwachungsfunktionen sowie geeignete Bedien- und Visualisierungsmöglichkeiten bereit. Er kapselt damit die gesamte Funktionalität, die im Zusammenhang mit dem entsprechenden Feldgerätetyp notwendig ist. Das Anwenderprogramm nutzt diesen Baustein, um eine gewünschte Steuerung für eine Maschine oder einen Prozess zu realisieren, ohne dabei auf Kenntnisse der internen Daten und Operationen des Funktionsbausteins zurückgreifen zu müssen.

KANALFUNKTIONEN (DRIVER)

In Ergänzung zur Behandlung der Feldgeräte durch eigene wiederverwendbare Bausteine ist es häufig sinnvoll, die Peripherieanbindung ebenfalls durch ***Kanalbausteine*** (engl. Driver) zu abstrahieren. Es ist zwar stets möglich, über Symbolnamen oder Adresse direkt auf das Prozessabbild zuzugreifen. Allerdings müssen dann die möglicherweise vielfältigen Parameter zur Konfiguration des Kanals an anderer Stelle gesetzt werden. Dies führt schnell zu unübersichtlichen Programmen. ***PCS 7*** bietet eine Reihe von Kanalbausteinen, die zum einen die Statussignale der Baugruppen auswerten und zum anderen das Testen und die Inbetriebnahme von Automatisierungsprogrammen durch Simulationsmodi unterstützen. In den analogen Kanalbausteinen entsprechend Tabelle 1 erfolgt zudem durch die Parameter VLRANGE und VHRANGE eine Abbildung von den internen digitalen Größen auf die physikalischen Rechen- und Anzeigegrößen. ***PCS 7*** kann bei Verwendung von Kanalbausteinen die notwendigen Treiber automatisch erzeugen. Die Kanalbausteine werden deshalb in den Vorlagen der ***PCS 7*** Bibliotheken vielfach eingesetzt.

Tabelle 1: Auflistung verschiedener Kanalbausteine zur Abstraktion der Peripherieanbindung

Kanalbaustein	Baustein	Verschaltung, Parameter	Signalqualität
Digitaler Ausgang	CH_DO	VALUE	QBAD, QUALITY
Digitaler Eingang	CH_DI	VALUE	QBAD, QUALITY
Analoger Ausgang	CH_AO	VALUE, VLRANGE, VHRANGE	QBAD, QUALITY
Analoger Eingang	CH_AI	VALUE, VLRANGE, VHRANGE	QBAD, QUALITY

EINZELSTEUERFUNKTIONEN

Motoren und Ventile sind als steuerungstechnische Einrichtungen in der Fabrik- und Prozessautomatisierung von zentraler Bedeutung. Es existiert eine Vielzahl gängiger Typen mit spezifischem Bedien- und Meldeverhalten. Im Sinne einer objektorientierten Automatisierung werden derartige Einrichtungen nicht direkt angesteuert, sondern zunächst als Funktionsbaustein-Typen modelliert. Die Ansteuerung erfolgt dann stets indirekt über eine Instanz des entsprechenden Funktionsbaustein-Typs. Funktionsbausteine für Motoren und Ventile werden als ***Einzelsteuerfunktionen (ESF)*** oder ***Individual Drive Functions (IDF)*** bezeichnet. Einzelsteuerfunktionen ermöglichen die Steuerung, Überwachung und Bedienung von steuerungstechnischen Einrichtungen durch die Bereitstellung von entsprechenden Anschlüssen für Stell- und Steuersignale sowie für Parametrier- und Überwachungsfunktionen. Die technische Umsetzung der Steuerung, wie zum Beispiel das Anlaufverhalten, die Ansteuerung des Antriebs oder die Geräteüber-

wachung wird durch die Funktionsbaustein-Instanz realisiert und bleibt dem Nutzer verborgen. ***PCS 7*** bietet bereits eine Vielzahl von leistungsfähigen und getesteten Einzelsteuerfunktionen als Bausteintypen in den leittechnischen Bibliotheken an. Tabelle 2 fasst die Einzelsteuerfunktionen der ***PCS 7 Standard Library*** [2] zusammen.

Komplexere Einzelsteuerfunktionen werden in der ***PCS 7 Advanced Process Library*** [3] bereitgestellt.

Tabelle 2: Die Einzelsteuerfunktionen der ***PCS 7 Standard Library***

Einzelsteuer-funktion	Anwendungsbereich	Objektname
MOTOR	Ansteuerung von Motoren mit einem Steuersignal (ein/aus) und einer Laufrückmeldung	FB 66
MOT_REV	Ansteuerung von Motoren mit zwei Drehrichtungen (Linkslauf/Rechtslauf) und maximal zwei Rückmeldungen	FB 67
MOT_SPED	Ansteuerung von Motoren mit zwei Geschwindigkeiten (langsam/schnell) und maximal zwei Rückmeldungen	FB 68
VALVE	Ansteuerung von Steuerventilen mit einem Steuersignal (öffnen/schließen) und Stellungs-rückmeldesignalen (offen/geschlossen)	FB 73
VAL_MOT	Ansteuerung von Motorventilen mit zwei Steuersignalen und Stellungsrückmeldesignalen (offen/geschlossen).	FB 74

SCHUTZMAßNAHMEN

Bei der Ansteuerung von steuerungstechnischen Einrichtungen sind verschiedene Schutzmaßnahmen zu treffen. Zum einen sind die Einrichtungen selbst vor Fehlern zu schützen. Zum anderen muss der gesteuerte Prozess im Fehlerfall in einen sicheren Zustand überführt und dort solange gehalten werden, bis der Fehlerfall beseitigt ist.

Gerätefehler (zum Beispiel Kabelbruch, Achsbruch) können steuerungstechnisch nicht verhindert werden. Die Auswirkungen können jedoch durch Redundanzkonzepte minimiert werden. ***Prozessfehler*** (zum Beispiel Behälterüberlauf, Trockenlauf einer Pumpe) sollen hingegen direkt durch die Steuerung verhindert werden. Dazu werden entsprechende ***Verriegelungen*** implementiert. Erkennt die Einzelsteuerfunktion aufgrund der aktuellen Eingangswerte einen gefährlichen Prozesszustand, so wird das gesteuerte Gerät in einen sicheren Zustand überführt (siehe Kapitel ‚Anlagensicherung'). Das Gerät wird solange in diesem Zustand gehalten, wie der gefährliche Prozesszustand andauert. Üblicherweise werden Verriegelungen mithilfe einer ***Verriegelungsmatrix*** spezifiziert.

Um einen eingetretenen Gerätefehler zu erkennen, führt eine Einzelsteuerfunktion häufig eine ***Laufzeitüberwachung*** durch. Mithilfe bestimmter Sensorinformationen, zum Beispiel von Endlagensensoren in Ventilen, überprüft die Einzelsteuerfunktion, ob die ausgegebenen Stellsignale auch die geforderte Wirkung erzielen. Stehen die gemessenen Werte über einen bestimmten Zeitraum hinweg in Widerspruch zu den ausgegebenen Stellsignalen, so liegt eine Störung vor. Wird ein solcher Laufzeitfehler erkannt, so wird das übergeordnete Leitsystem alarmiert und das gesteuerte Gerät wird deaktiviert. Das Gerät bleibt solange inaktiv, bis der Laufzeitfehler beseitigt und der Alarm quittiert worden ist. Häufig werden zur Erkennung von Gerätefehlern einfache binäre Schutzschalter verwendet.

BETRIEBSARTEN

Steuerungstechnische Einrichtungen werden im Allgemeinen nicht ausschließlich automatisch betrieben. Zeitweise ist es notwendig die Steuerung in der Leitwarte manuell durchzuführen oder das Gerät direkt vor Ort zu ansteuern, zum Beispiel für Reparaturarbeiten. Es wird daher zwischen vier grundlegenden Betriebsarten unterschieden:

– ***Außer Betrieb***: Das Gerät ist nicht aktiv.
– ***Automatikbetrieb***: Die Einzelsteuerfunktion wird von einem übergeordneten Programm automatisch angesteuert.
– ***Handbetrieb***: Die Einzelsteuerfunktion wird über eine Bediengraphik des Leitsystems direkt durch den Bediener angesteuert.
– ***Vor-Ort-Betrieb***: Der Bediener bedient das Gerät direkt vor Ort, zum Beispiel über ein Bedientableau.

Eine Einzelsteuerfunktion darf sich immer nur in genau einer Betriebsart befinden. Es existieren verschiedene Konzepte, wie die damit verbundene Betriebsartenumschaltung sicher und eindeutig realisiert werden kann. Grundsätzlich lassen sich diese Konzepte unterscheiden zwischen einer Gleichberechtigung der Betriebsarten und einer Betriebsartenhierarchie. Im letztgenannten Fall werden die möglichen Betriebsarten zusätzlich eindeutig priorisiert. Eine angewählte Betriebsart wird in diesem Fall genau dann geändert, wenn das Gerät entweder nicht aktiv ist (Betriebsart ***Außer Betrieb***) oder wenn die gewünschte neue Betriebsart eine höhere Priorität hat als die bereits angewählte.

FUNKTIONSBAUSTEIN-TYPEN IN PCS 7

Funktionsbaustein-Typen werden in ***PCS 7*** als Bausteintypen bezeichnet und stellen vorgefertigte Programmteile für die Bearbeitung wiederkehrender Funktionen dar. Sie können in CFC-Pläne eingefügt werden und dort als Instanzen parametriert, verschalten und an die Projekterfordernisse angepasst werden.

Dabei legt der Baustein-Typ die Charakteristik für sämtliche Instanzen dieses Typs fest. Die verwendeten Bausteintypen eines Projektes werden dazu in der Stammdatenbibliothek abgelegt. Wird der dort abgelegte Bausteintyp geändert, so werden die Änderungen unmittelbar von sämtlichen Instanzen übernommen. Dieses Konzept der Typisierung unterstützt das rationelle Engineering durch die Wiederverwendbarkeit und die zentrale Änderbarkeit häufig wiederkehrender Funktionen.

Eine Einzelsteuerfunktion in ***PCS 7*** modelliert eine steuerungstechnische Einrichtung und stellt die gesamte Ansteuerlogik bereit. Abbildung 2 beschreibt am Beispiel der Einzelsteuerfunktion ***MOTOR*** (FB 66) den grundsätzlichen Aufbau des entsprechenden Motorbausteins.

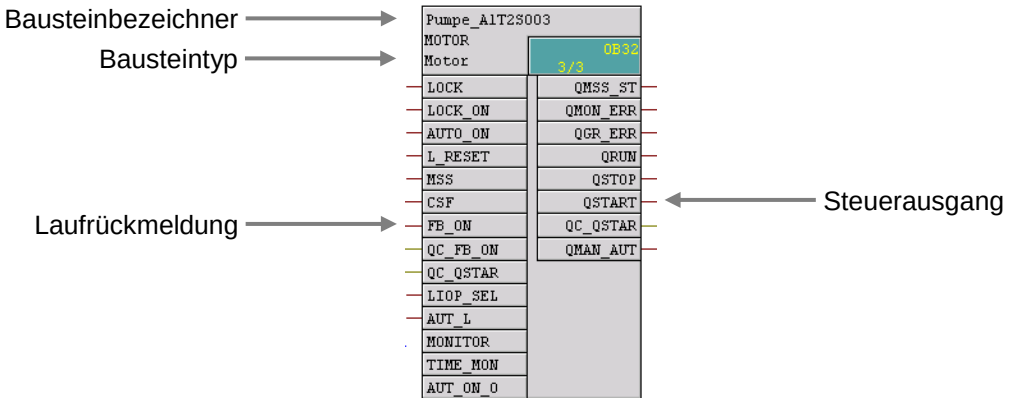


Abbildung 2: Baustein der Einzelsteuerfunktion MOTOR (FB 66)

Der Baustein bietet darüber hinaus die folgenden Funktionen:

Bedienen, Beobachten, Melden

Über einen Anzeige- und Bedienbereich können Prozess- und Sollwerte bedient und beobachtet werden. Bedienberechtigungen und Wartungsfreigaben können gesteuert werden. Allgemeine und instanzspezifische Meldungen geben Aufschluss über den Geräte- und Prozessstatus.

Steuerung von Signalen

Steuersignale können statisch oder gepulst ausgegeben werden. Der Signalstatus, also die Qualität der Stellsignale, wird überwacht. Es können interne und externe Sollwerte sowie simulierte Werte vorgegeben werden. Außerdem können Rampen oder Totzonen eingestellt werden.

Überwachung

Der Baustein kann Grenzwerte überwachen und bei Grenzwertverletzungen entsprechende Warnungen oder Alarme generieren. Außerdem können Rückmeldungen von Stellsignalen überwacht werden.

Verriegelungsfunktionen

Der Baustein ermöglicht eine einfache Einschaltfreigabe, eine Verriegelung ohne Rücksetzen sowie eine Verriegelung mit Rücksetzen. Er implementiert eine Motorschutzfunktion, die den Motor bei thermischer Überlast abschalten kann. Zusätzlich ist für Motoren ein Schnellstopp verfügbar, der in allen Betriebsarten und –zuständen höchste Priorität hat. Im Fall einer Verriegelung wird das Gerät automatisch in einen energielosen Zustand und damit in eine definierte Sicherheitsstellung überführt.

Betriebszustandsauswahl

Die eingangs vorgestellten Betriebsarten Vor-Ort-Betrieb, Automatikbetrieb, Handbetrieb und Außer Betrieb stehen für sämtliche Einzelsteuerfunktionen für ***PCS 7*** zur Verfügung. Sie sind in der angegebenen Reihenfolge absteigend priorisiert, Automatik- und Handbetrieb besitzen die gleiche Priorität. Darüber hinaus ist es möglich, den Baustein über verschaltbare Eingangsparameter unabhängig von der aktuell anstehenden Ansteuerung in einen anderen Betriebszustand zu versetzen (***Erzwingen*** bzw. ***Forcen*** von Betriebszuständen).

Bildbausteine mit verschiedenen Sichten

Bildbausteine bieten für jeden Bausteintyp ein entsprechendes Bausteinsymbol und je nach Anwendungsfall entsprechende Sichten an. Typische Bildbausteine sind zum Beispiel das Bausteinsymbol selbst, die Parametersicht von Motoren und Ventilen oder die Grenzwertsicht von Motoren.

Diese Aufzählung zeigt deutlich die Komplexität und den Funktionsumfang einer üblichen Einzelsteuerfunktion. Die Anzahl der verfügbaren Ein- und Ausgänge bei diesen Bausteintypen ist dementsprechend groß. So besitzt die Einzelsteuerfunktion ***MOTOR*** (FB 66) insgesamt 53 Anschlüsse. Um die Komplexität des Programmentwurfs dennoch gering zu halten ist es möglich, nicht benötigte Ein- oder Ausgänge zu verbergen. Die Einzelsteuerfunktionen in ***PCS 7*** verwenden zudem ein einheitliches und durchgängiges Schema für die Bezeichnung der Ein- und Ausgänge.

Die Einzelsteuerfunktionen in ***PCS 7*** bieten einen großen Funktionsumfang und garantieren sie eine konstant hohe Qualität und Zuverlässigkeit der verwendeten Algorithmen. Sämtliche Bausteintypen sind umfangreich getestet und haben sich bereits industriell bewährt. Damit verkürzt sich der Aufwand für die Entwicklung leistungsfähiger, qualitativ hochwertiger Lösungen erheblich.

LITERATUR

[1] Seitz, M. (2008): Speicherprogrammierbare Steuerungen. Hanser Fachbuchverlag

[2] SIEMENS (2009): Prozessleitsystem PCS 7: PCS 7 Standard Library V71.

[3] SIEMENS (2009): Prozessleitsystem PCS 7: PCS 7 Advanced Process Library V71.

SCHRITT-FÜR-SCHRITT-ANLEITUNG

AUFGABENSTELLUNG

Ein Pumpenmotor zum Ablassen der Flüssigkeit aus dem Reaktor R001 soll als erstes Programm im ***Continuous Function Chart (CFC)*** erstellt werden. Der Pumpenmotor hat einen Ausgang zum Ansteuern der Pumpe und eine Rückmeldung zur Kontrolle, ob die Pumpe auch läuft.

Tabelle 3: Zuordnungsliste

Symbol	Adresse	Datentyp	Symbolkommentar
A1.T2.A1T2S003.SO+.O+	E 6.1	BOOL	Pumpe Ablass Reaktor R001 Rückmeldung ein
A1.T2.A1T2S003.SV.C	A 6.3	BOOL	Pumpe Ablass Reaktor R001 Stellsignal

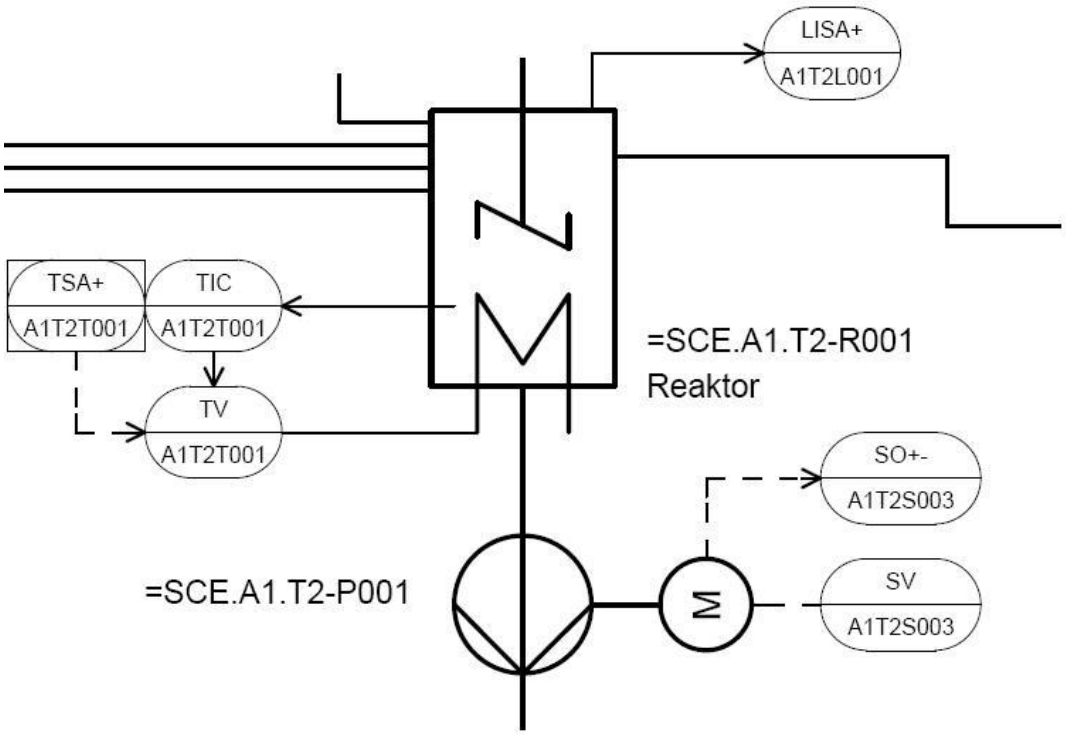


Abbildung 3: Ausschnitt aus dem R&I-Fließbild

Bei der Erstellung des Programms wird ein vorgefertigter Plan ‚MOTOR' aus einer ***PCS 7***-Bibliothek verwendet. Dieser wird in die projekteigene Stammdatenbibliothek kopiert und dort angepasst. Anschließend wird das Programm in die SPS-Simulation geladen und getestet.

LERNZIEL

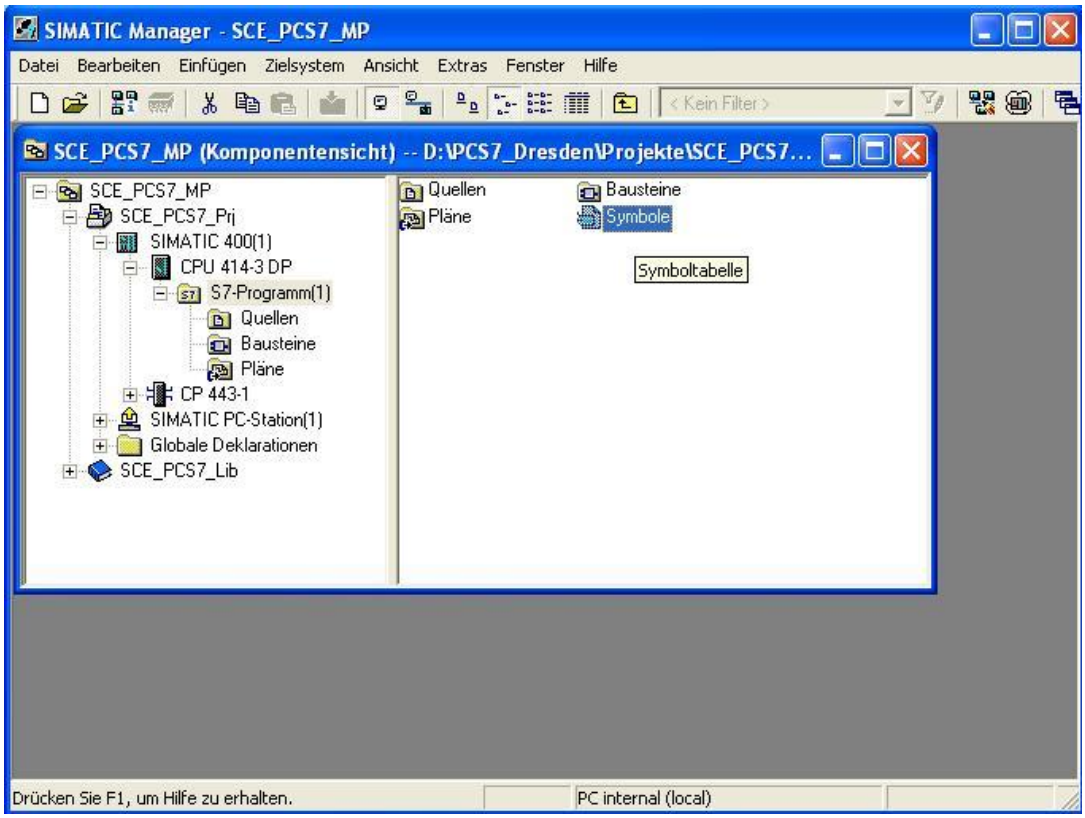
In diesem Kapitel lernt der Studierende:

- Symboltabelle, Anlegen und Importieren von Symbolen
- Verwendung von Stammdatenbibliotheken und Messstellentypen
- CFC- Pläne anlegen und bearbeiten
- Projekt zentral übersetzen und laden
- Testen des Programms mit Hilfe der Steuerfunktionen im CFC

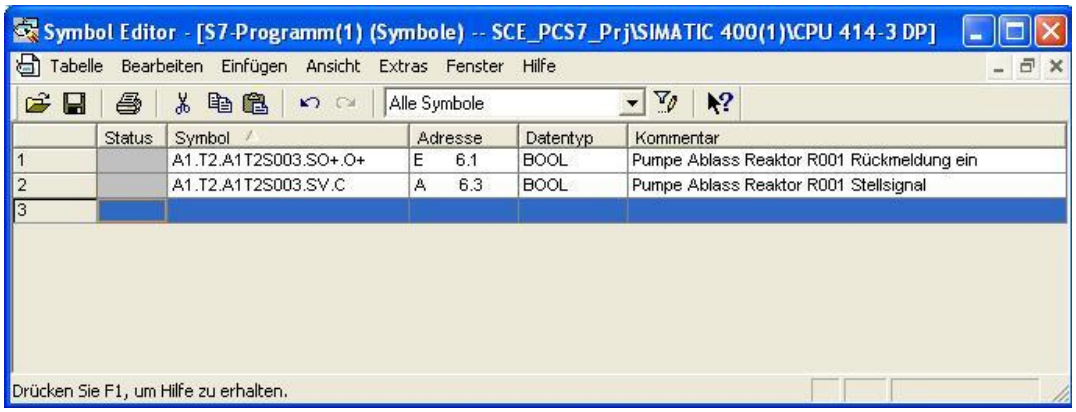
PROGRAMMIERUNG

1. Bevor wir mit der Programmierung der Einzelsteuerfunktion für den Pumpenmotor beginnen, werden die Symbole für die globalen Variablen angelegt. Wir wählen hierzu die Komponentensicht im SIMATIC Manager, markieren den Ordner ‚S7-Programm(1)' und öffnen mit einem Doppelklick auf Symbole die Symboltabelle.

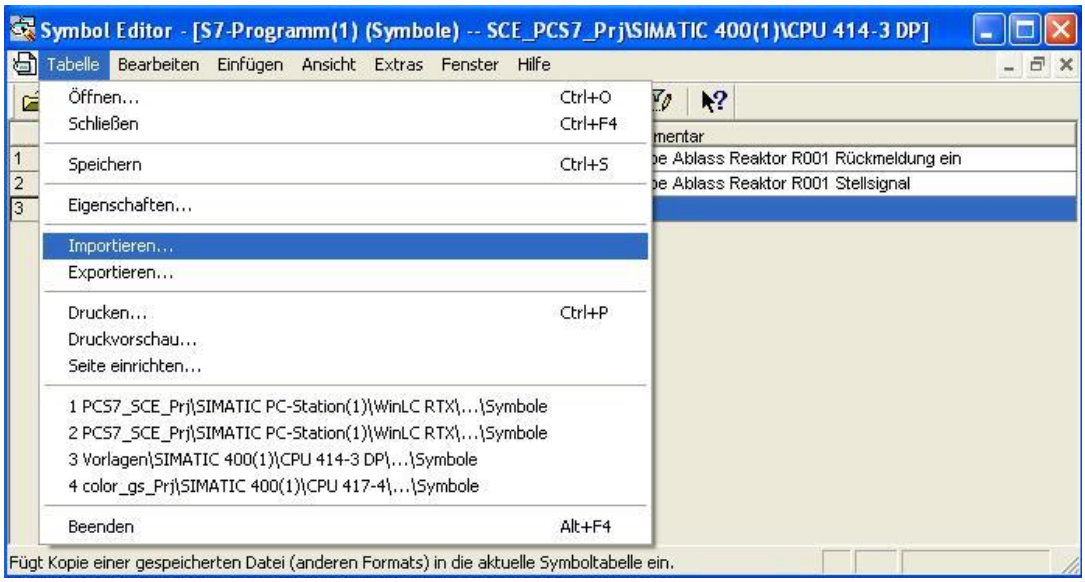
   (→ SIMATIC Manager → Ansicht → Komponentensicht → S7-Programm(1) → Symbole)



2. In der Symboltabelle können nun Symbol und Symbolkommentar zu jedem Operanden festgelegt werden.

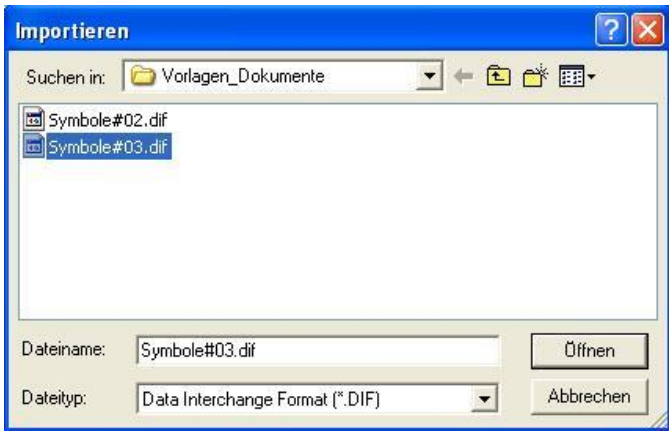


3. Falls vorhanden, kann auch der Inhalt für die komplette Symboltabelle im *.dif- Format importiert werden. (→ Tabelle → Importieren). In diesem Fall wird die importierte Tabelle in die bereits vorhandene Tabelle integriert.



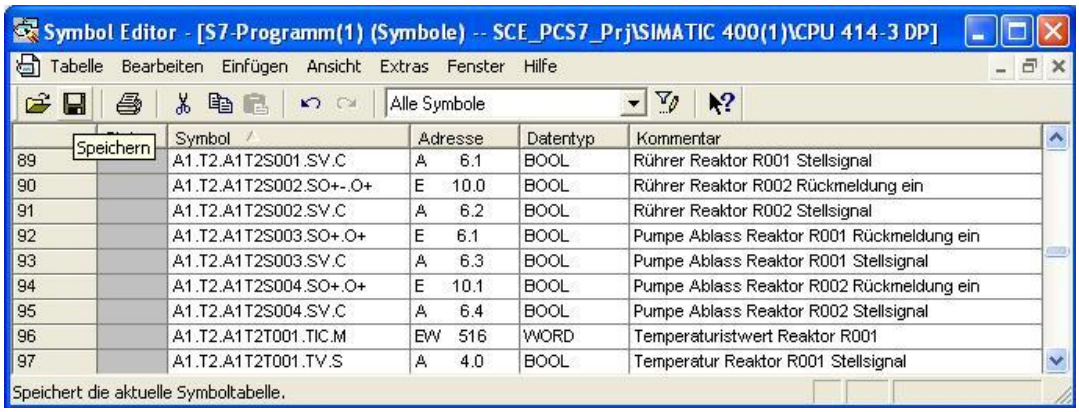
4. Nun erfolgt die Auswahl der Quelldatei im ‚Data Interchange Format(*.DIF)

   (→ Symbole#03.dif → Öffnen)



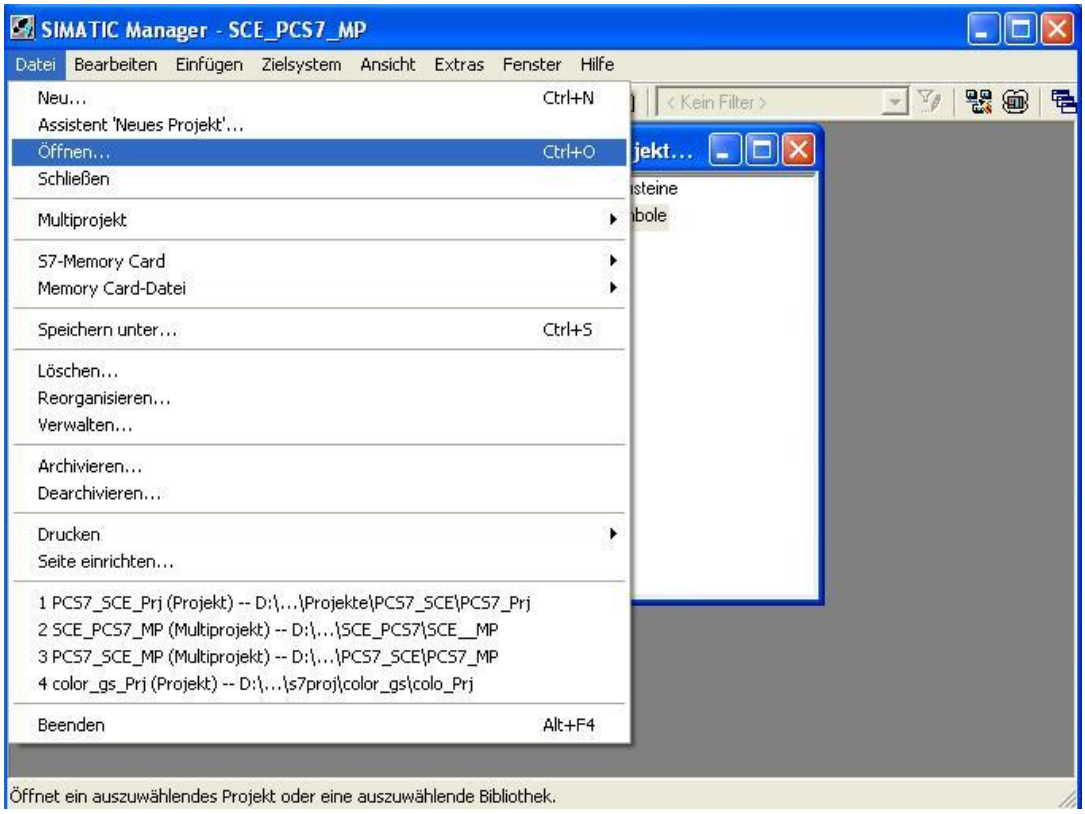
5. Die fertige Symboltabelle muss vor dem Schließen noch gespeichert werden.

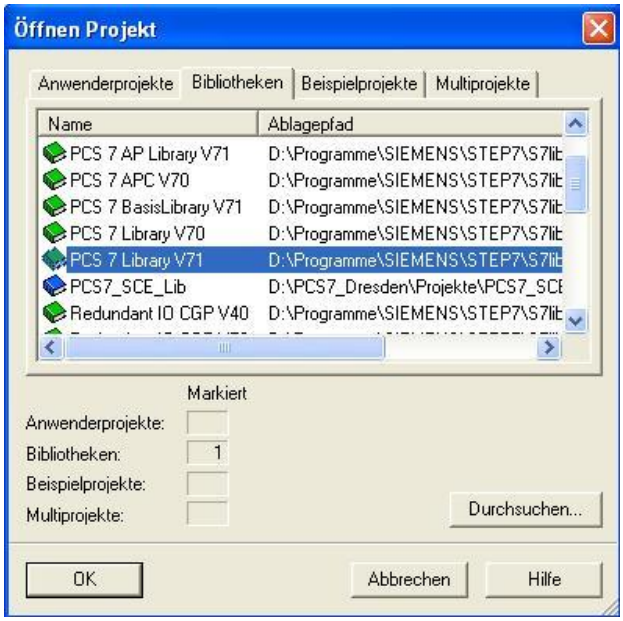
   (→ Speichern →)



6. ***PCS 7*** stellt in umfangreichen Bibliotheken eine Vielzahl an fertigen Bausteinen und auch vorgefertigte Pläne, so genannte Templates, zur Verfügung. Für unseren Pumpenmotor wollen wir genau so ein Template nutzen. Dafür öffnen wir die ***PCS 7 Library V71***.

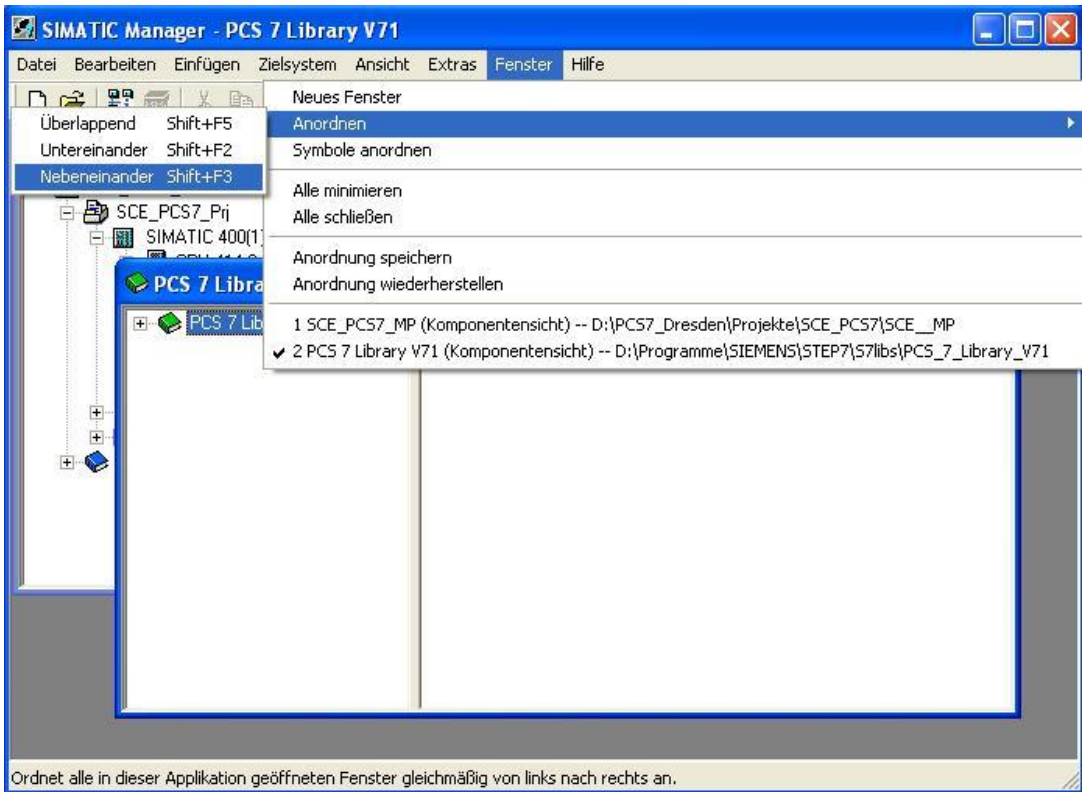
   (→ Datei → Öffnen → Bibliotheken → PCS 7 Library V71 → OK)





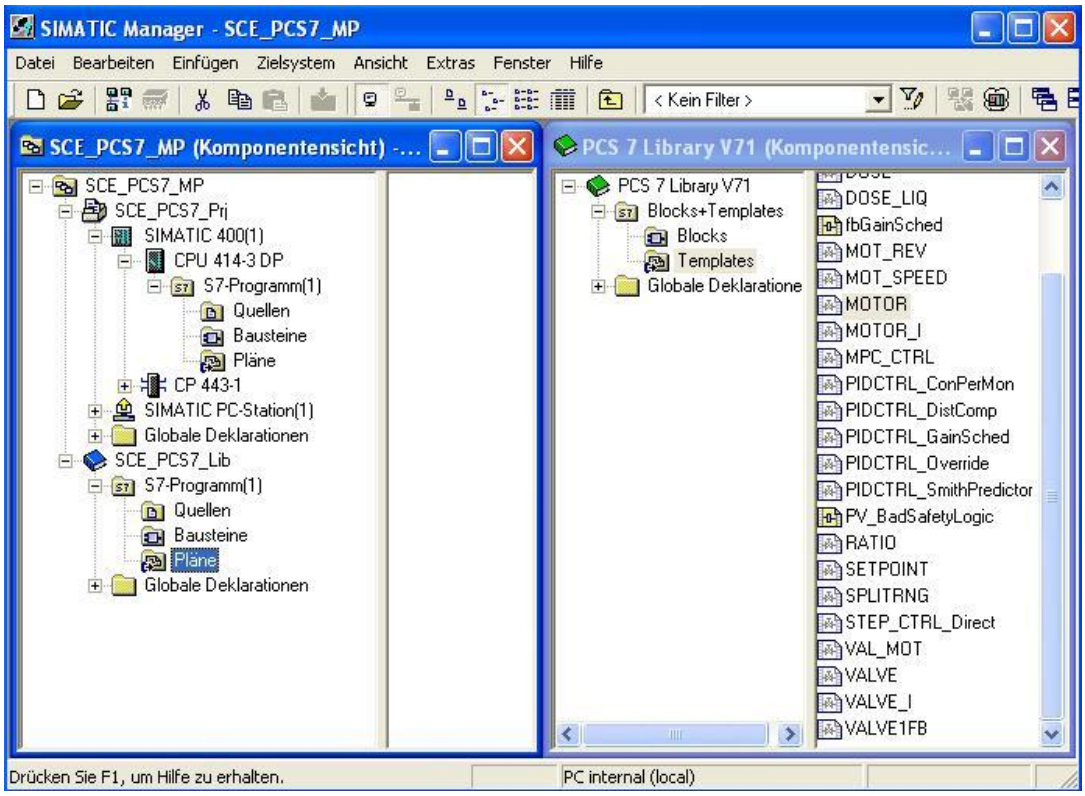
7. Um dieses Template aus der Bibliothek in unser Projekt zu ziehen, ordnen wir Projekt-Fenster und Bibliotheks-Fenster nebeneinander an.

   (→ Fenster → Anordnen → Nebeneinander)



8. In der Komponentensicht unseres Multiprojektes markieren wir in der Stammdatenbibliothek ‚SCE_PCS7_Lib' den Ordner ‚Pläne'. Dorthin ziehen wir mit der Maus per Drag&Drop das Template ‚MOTOR' aus dem Ordner ‚Templates' der ***PCS 7 Library V71***. Dann schließen wir die ***PCS 7 Library V71***.

   (→ SCE_PCS7_Lib → Pläne → PCS 7 Library V71 → Templates → MOTOR →)

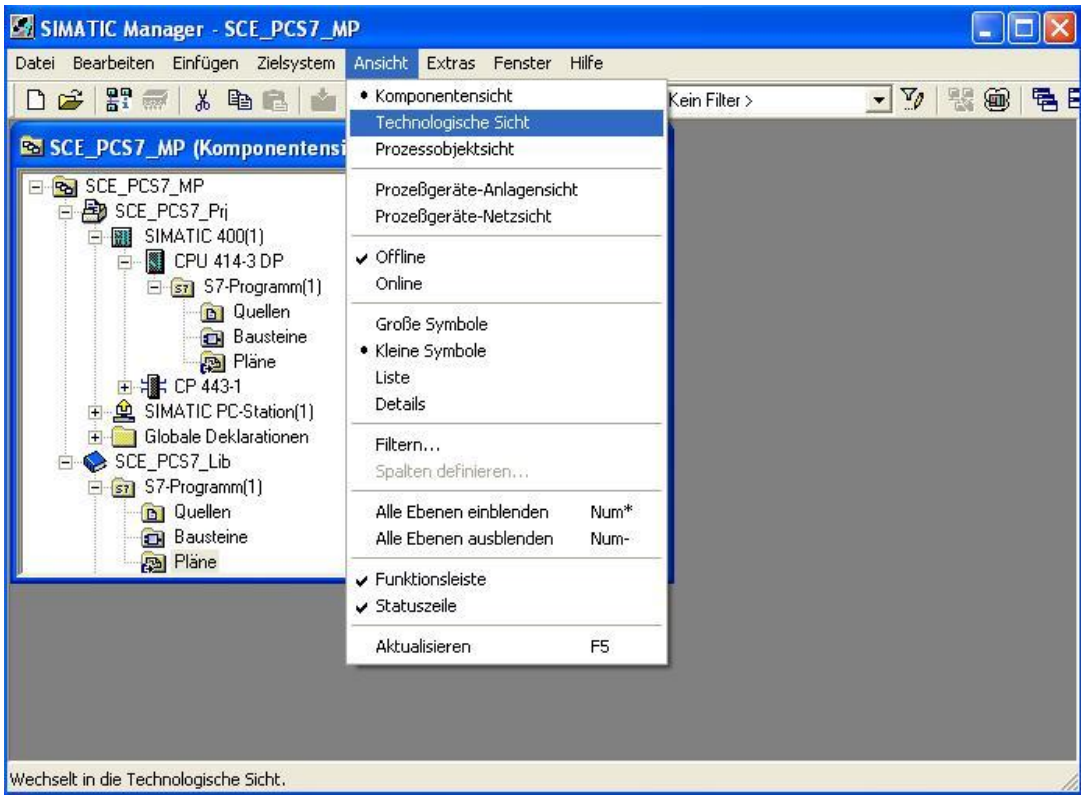


Hinweis: Da dieses Template in unserem Projekt mehrmals als Vorlage für Motoransteuerungen dienen soll, wird dieses zuerst in die Stammdatenbibliothek ‚SCE_PCS7_Lib' in unserem Multiprojekt kopiert.

Diese Stammdatenbibliothek stellt sicher, dass innerhalb eines Projektes immer derselbe Stand an Bausteinen und Planvorlagen (Messstellentypen) verwendet wird.

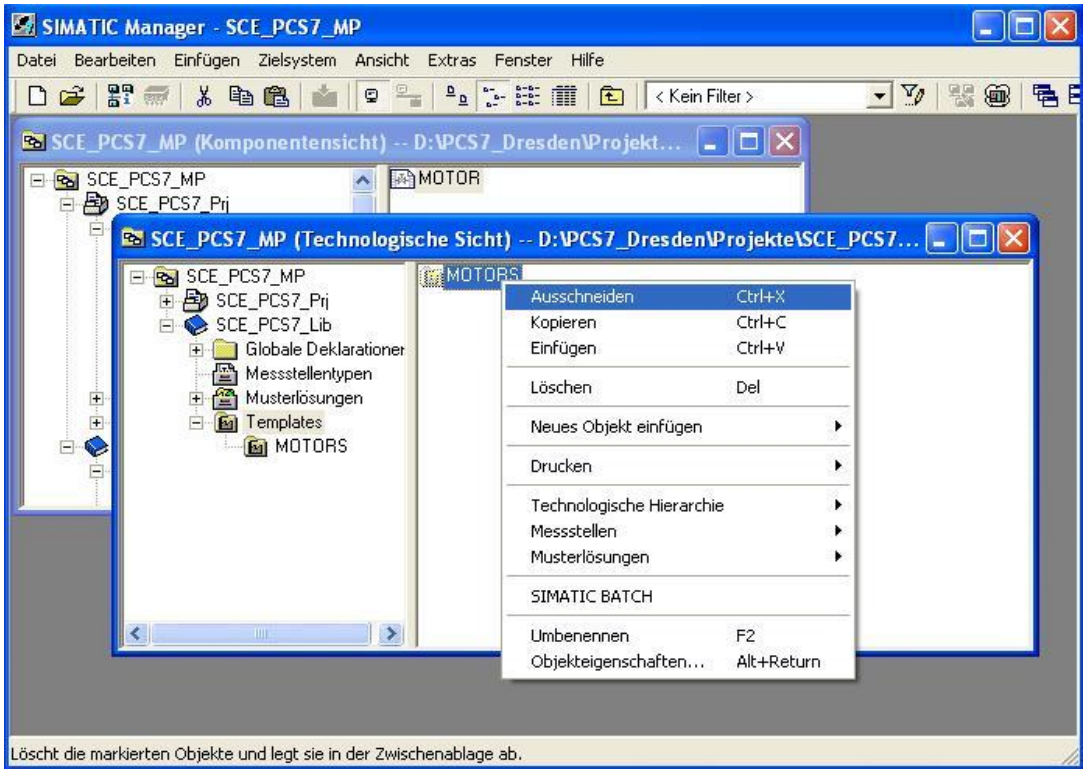
9. Die weiteren Schritte erfolgen wieder in der Technologischen Sicht.

   (→ Ansicht → Technologische Sicht)



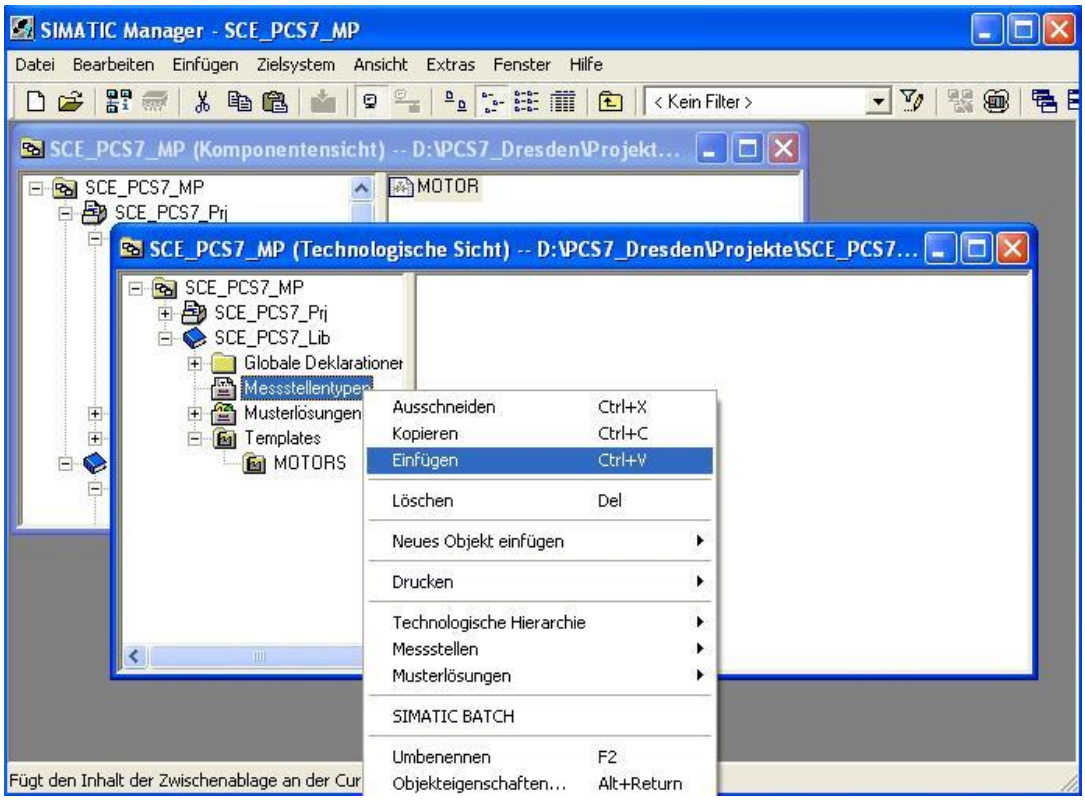
10. In der Stammdatenbibliothek wird nun der ganze Ordner ‚MOTORS' aus den Templates ausgeschnitten.

    (→ SCE_PCS7_Lib → Templates → MOTORS → Ausschneiden)



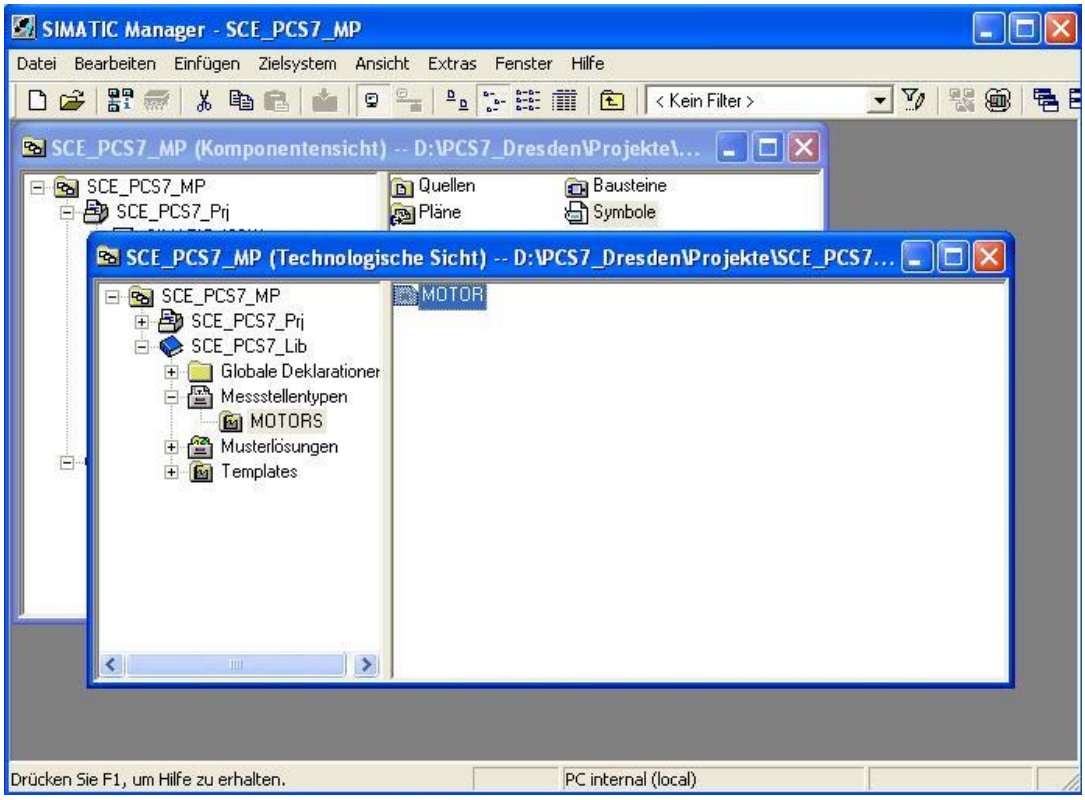
11. Der kopierte Ordner wird dann unter Messstellentypen wieder eingefügt.

    (→ Messstellentypen → Einfügen)



Hinweis: Die Verwendung von Messstellentypen als Vorlagen für Messstellen ermöglicht uns später, Änderungen zentral durchzuführen. Beim Ändern des Messstellentyps werden die im Projekt vorhandenen Messstellen automatisch abgeglichen.

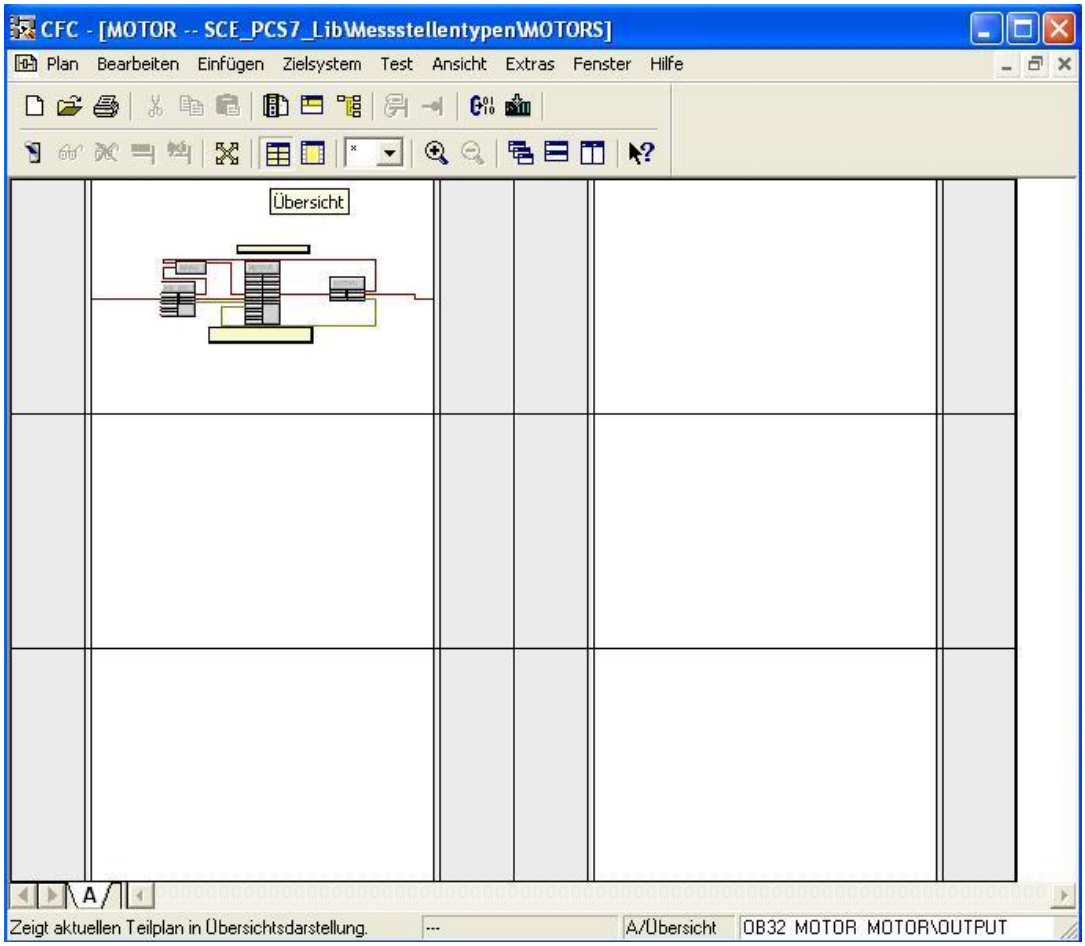
12. Soll nun eine Änderung zentral für alle Messstellen des Typs ‚MOTOR' erfolgen, so geschieht dies, indem der CFC- Plan ‚MOTOR' aus der Stammdatenbibliothek per Doppelklick geöffnet wird. (→ MOTOR)



Hinweis: CFC steht für Continuous Function Chart und ist eine grafische Programmiersprache für die Beschreibung kontinuierlicher Vorgänge. Im CFC werden vorgefertigte Bausteine platziert, parametriert und verschaltet. So erstellt der Programmierer eine Gesamt-Software-Struktur für die Steuerung und Regelung einer Maschine.

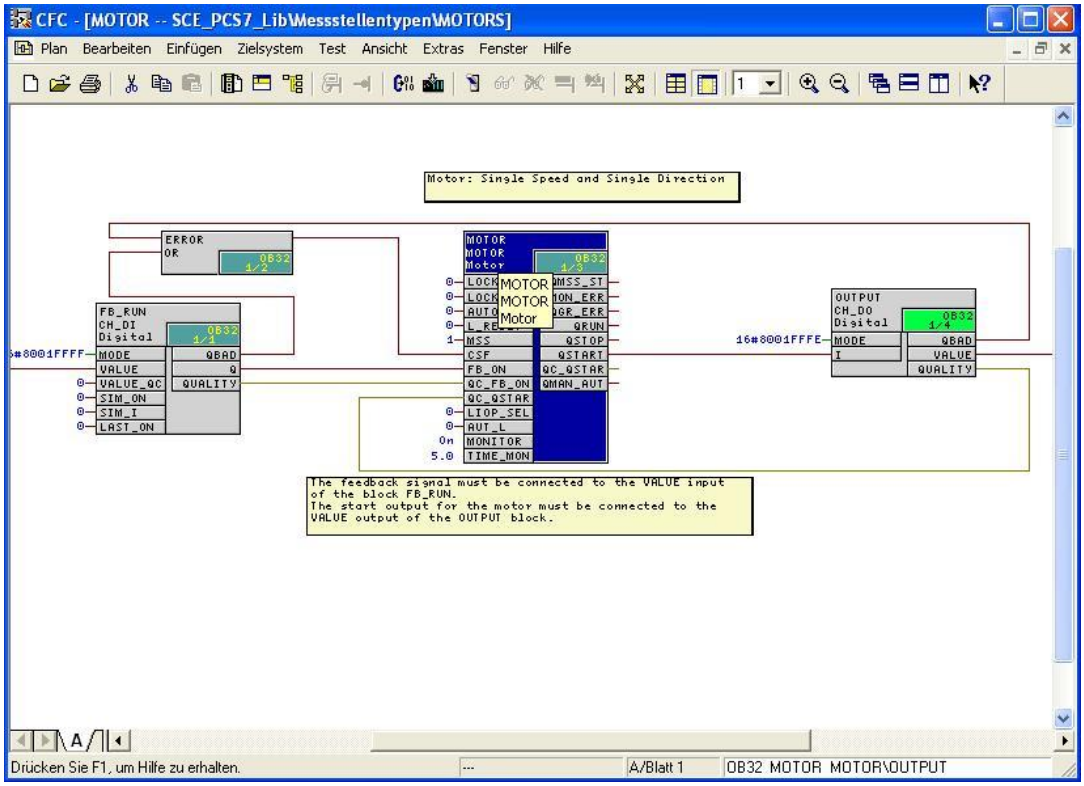
13. Ein CFC- Plan besteht aus Teilplänen mit jeweils sechs Blättern. In der Übersicht werden alle sechs Blätter mit ihren grauen Randleisten angezeigt. An den Randleisten sehen wir links die kommenden Signale und rechts die gehenden Signale eines Blattes. Mit einem Doppelklick auf ein Blatt können Sie in die Blattsicht wechseln.

    (→ Übersicht → Doppelklick auf erstes Blatt)

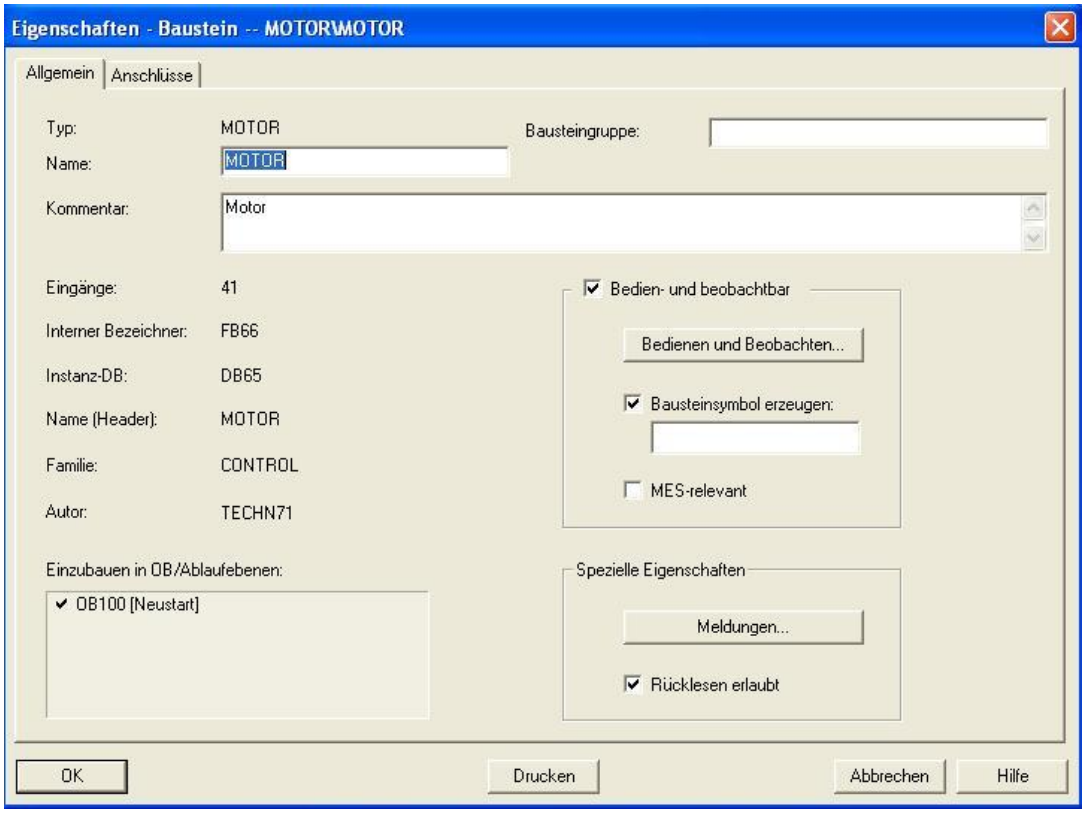


Hinweis: Mit den Registern in der unteren Leiste können Sie zwischen den Teilplänen (maximal A-Z) wechseln. Hier existiert jedoch zuerst nur ein Teilplan A.

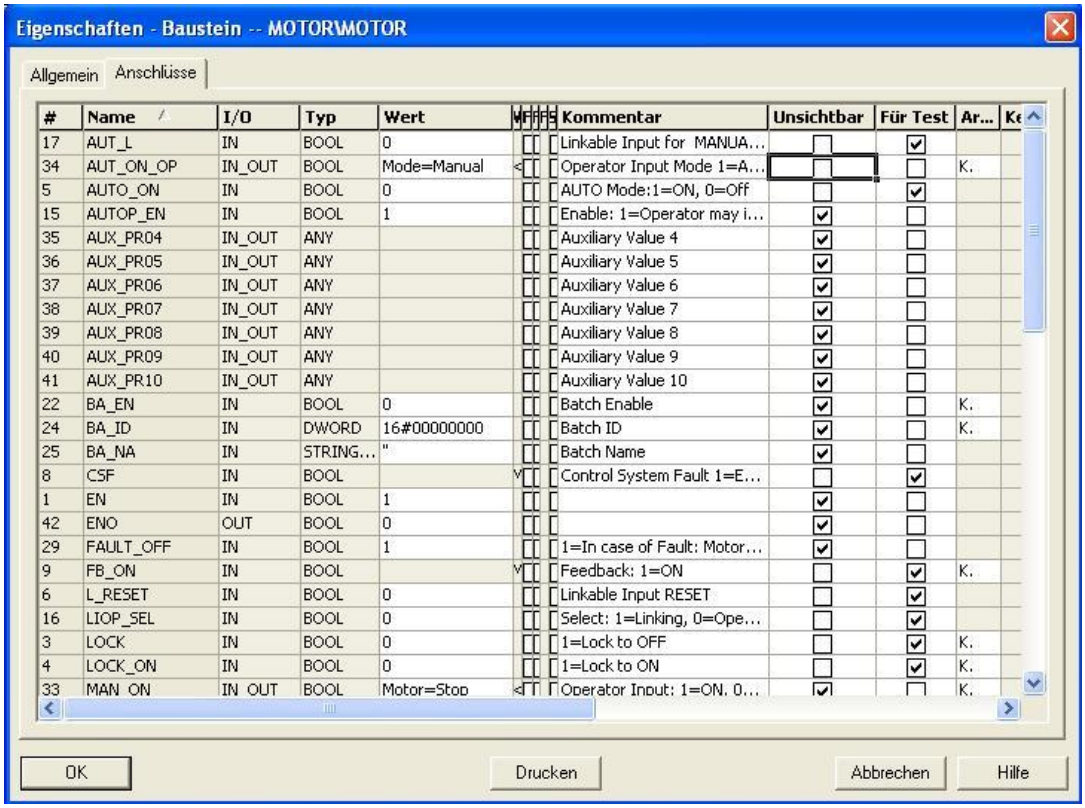
14. In unserem Messstellentyp ‚MOTOR' wollen wir bei dem Baustein ‚MOTOR' eine Veränderung vornehmen. Dazu werden dessen Eigenschaften mit Doppelklick geöffnet.(→ MOTOR)



15. Die allgemeinen Eigenschaften wie Bedien- und Beobachtbarkeit wollen wir nicht verändern, deshalb wechseln wir zu den Anschlüssen. (→ Anschlüsse)



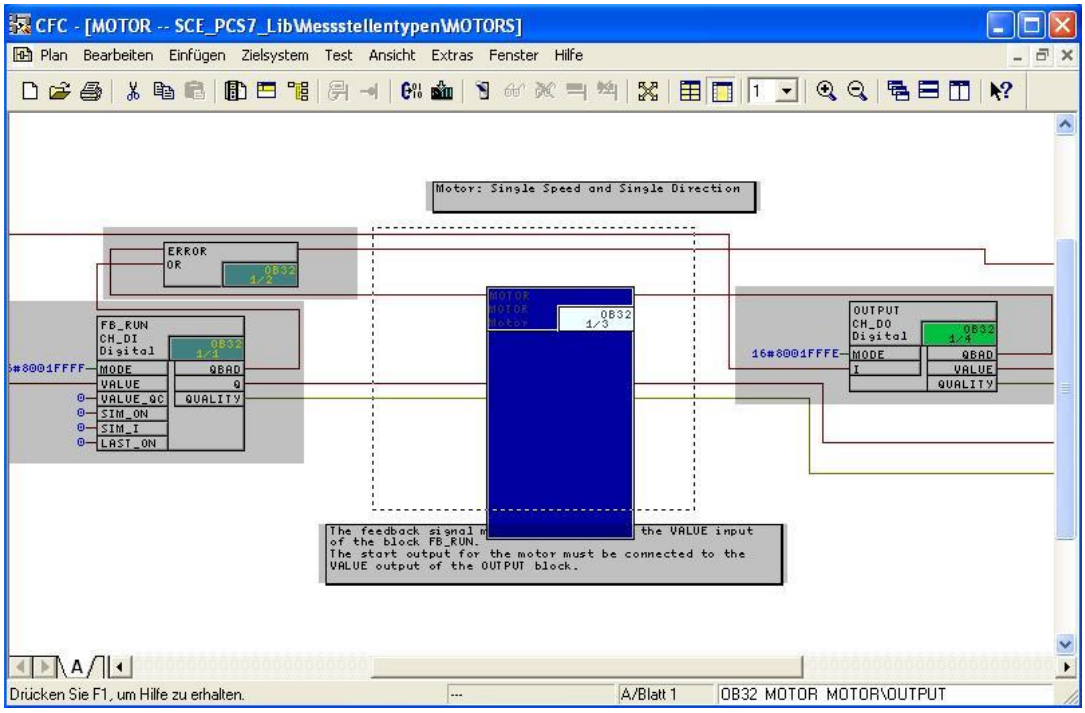
16. Die Anschlüsse werden in einer Tabelle zusammen mit einer Vielzahl an einstellbaren Eigenschaften dargestellt. Die wichtigsten Eigenschaften werden wir im Folgenden noch kennen lernen. In unserem Messstellentyp ‚MOTOR' wollen wir hier lediglich bei ‚AUT_ON_OP' die Unsichtbarkeit löschen. Dadurch wird dieser Anschluss in dem Blatt mit dargestellt. (→ AUT_ON_OP → ☐ → OK)



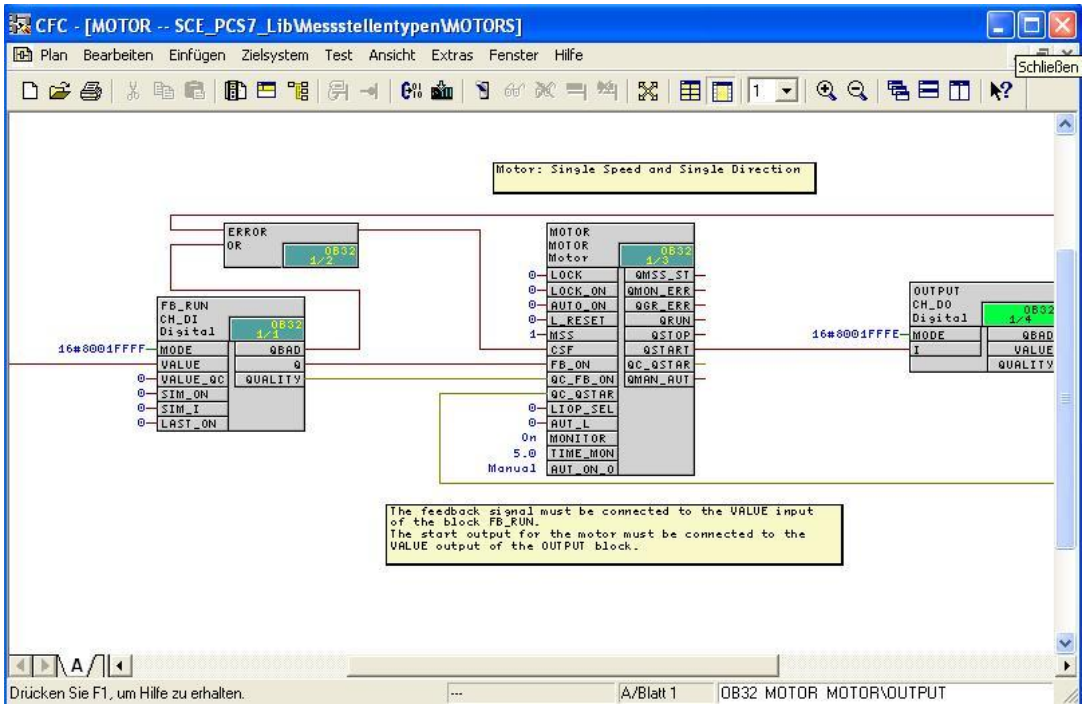
#	Name	I/O	Typ	Wert	Kommentar	Unsichtbar	Für Test	Ar...
17	AUT_L	IN	BOOL	0	Linkable Input for MANUA...	☐	☑	
34	AUT_ON_OP	IN_OUT	BOOL	Mode=Manual	Operator Input Mode 1=A...	☐	☐	K.
5	AUTO_ON	IN	BOOL	0	AUTO Mode:1=ON, 0=Off	☐	☑	
15	AUTOP_EN	IN	BOOL	1	Enable: 1=Operator may i...	☑	☐	
35	AUX_PR04	IN_OUT	ANY		Auxiliary Value 4	☑	☐	
36	AUX_PR05	IN_OUT	ANY		Auxiliary Value 5	☑	☐	
37	AUX_PR06	IN_OUT	ANY		Auxiliary Value 6	☑	☐	
38	AUX_PR07	IN_OUT	ANY		Auxiliary Value 7	☑	☐	
39	AUX_PR08	IN_OUT	ANY		Auxiliary Value 8	☑	☐	
40	AUX_PR09	IN_OUT	ANY		Auxiliary Value 9	☑	☐	
41	AUX_PR10	IN_OUT	ANY		Auxiliary Value 10	☑	☐	
22	BA_EN	IN	BOOL	0	Batch Enable	☑	☐	K.
24	BA_ID	IN	DWORD	16#00000000	Batch ID	☑	☐	K.
25	BA_NA	IN	STRING...	''	Batch Name	☑	☐	
8	CSF	IN	BOOL		Control System Fault 1=E...	☐	☑	
1	EN	IN	BOOL	1		☑	☐	
42	ENO	OUT	BOOL	0		☑	☐	
29	FAULT_OFF	IN	BOOL	1	1=In case of Fault: Motor...	☑	☐	
9	FB_ON	IN	BOOL		Feedback: 1=ON	☐	☑	K.
6	L_RESET	IN	BOOL	0	Linkable Input RESET	☐	☑	
16	LIOP_SEL	IN	BOOL	0	Select: 1=Linking, 0=Ope...	☐	☑	
3	LOCK	IN	BOOL	0	1=Lock to OFF	☐	☑	K.
4	LOCK_ON	IN	BOOL	0	1=Lock to ON	☐	☑	K.
33	MAN_ON	IN_OUT	BOOL	Motor=Stop	Operator Input: 1=ON, 0...	☑	☐	K.

Hinweis: Durch einen Klick mit der Maus auf die Überschrift einer Eigenschaft kann die Darstellung nach dieser Spalte alphabetisch sortiert werden.

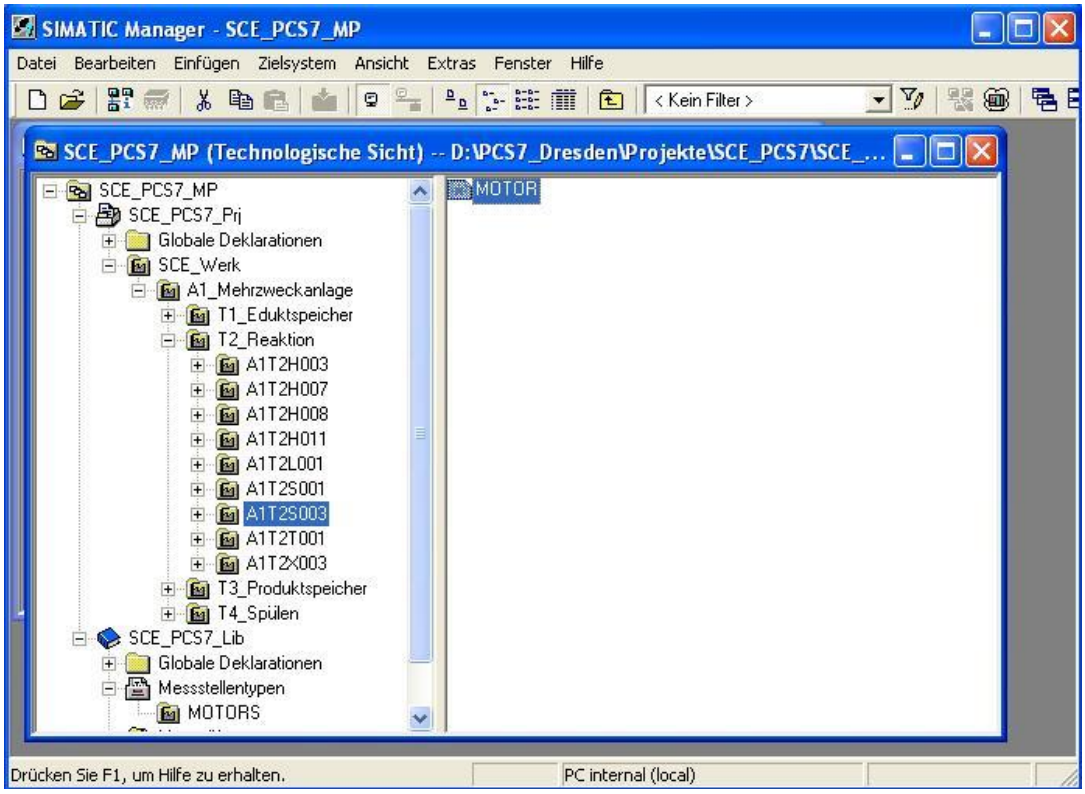
17. Der Baustein ‚MOTOR' ist nun in dem Blatt natürlich größer geworden und überlappt mit dem Kommentarfeld. Durch Verschieben des Bausteins oder des Kommentarfeldes wird dieses Problem gelöst und der Baustein wird wieder komplett dargestellt. (→ MOTOR)



18. Der Messstellentyp kann nun einfach geschlossen werden. Das Speichern erfolgt automatisch bei jeder Änderung. (→ )

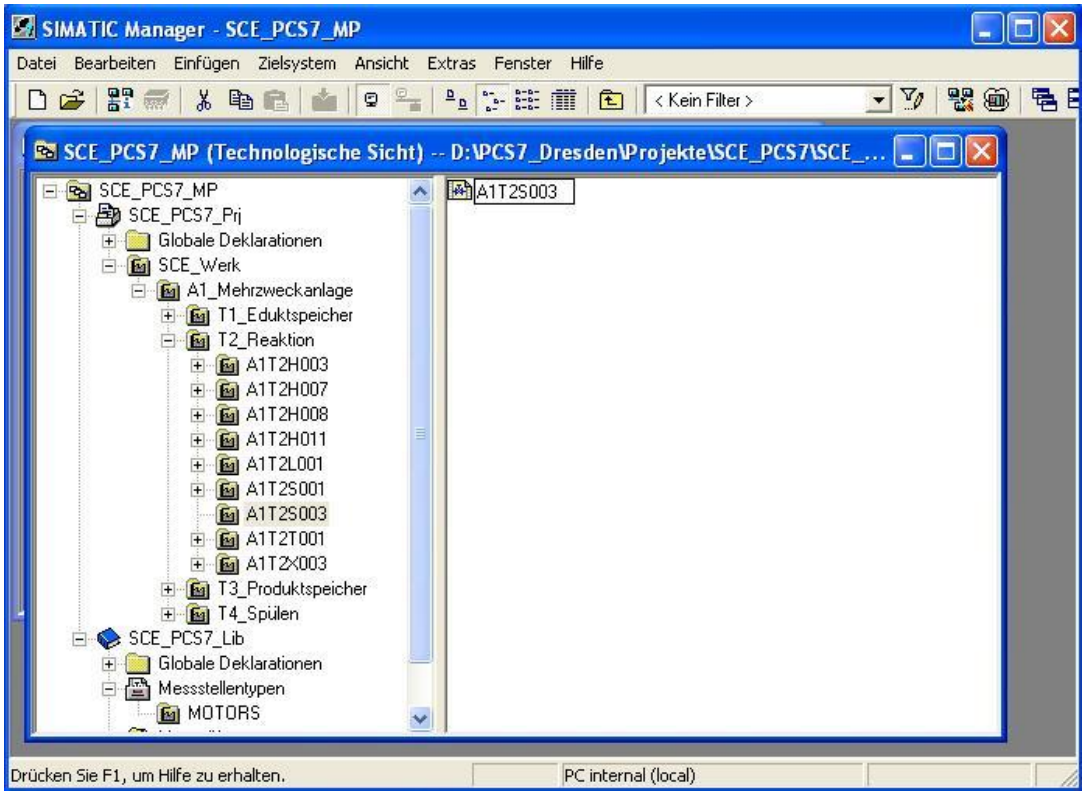


19. Zur Verwendung des Messstellentyps ‚MOTOR' wird dieser direkt per Drag&Drop in den Planordner ‚A1T2S003' geschoben. (→ MOTOR → A1T2S003)



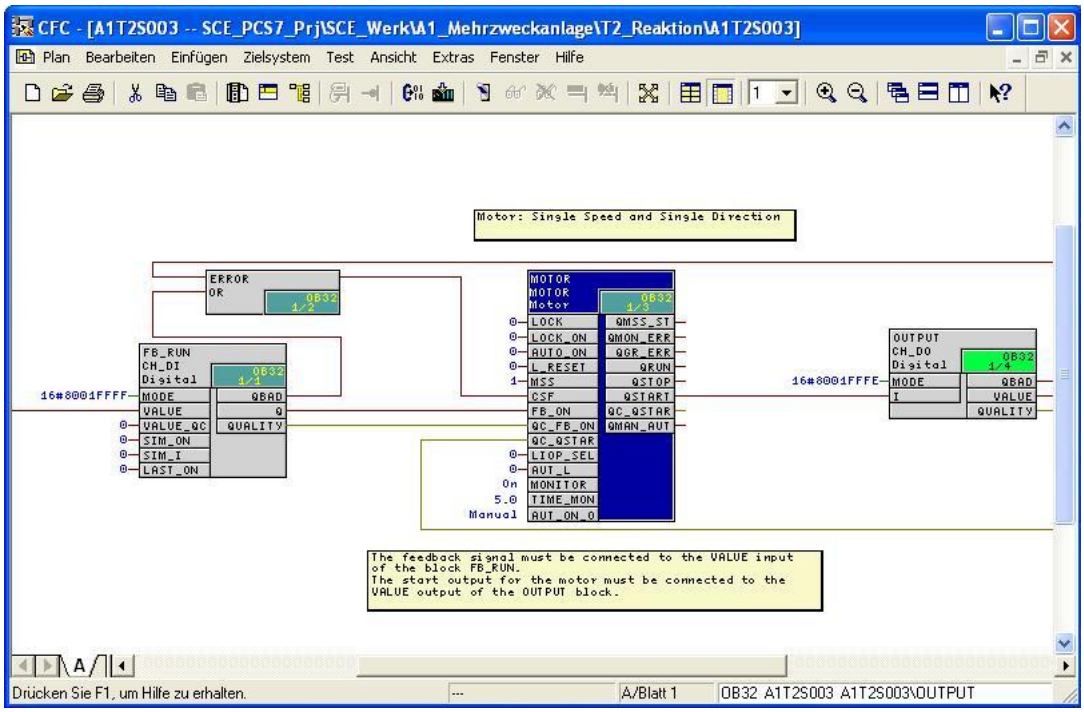
20. Der Plan wird nun noch so wie der Planordner benannt und per Doppelklick geöffnet.

(→ A1T2S003 → A1T2S003)



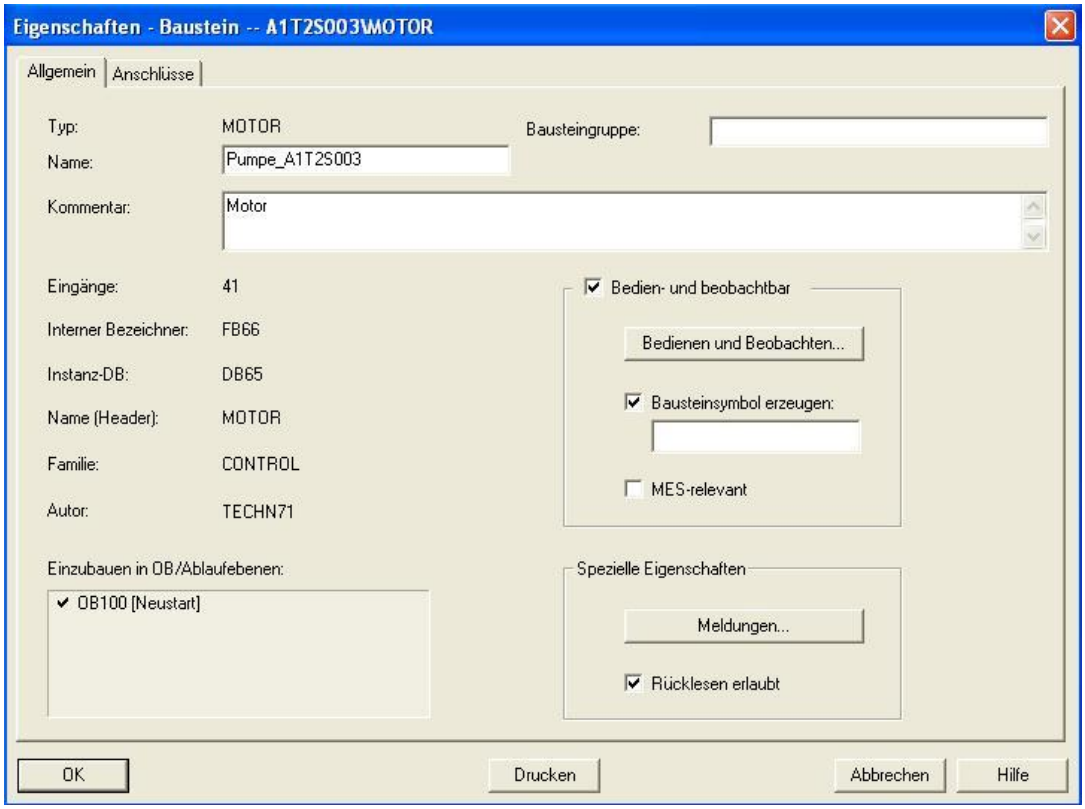
21. Per Doppelklick werden die Eigenschaften des Bausteins ‚MOTOR' geöffnet.

    (→ MOTOR)

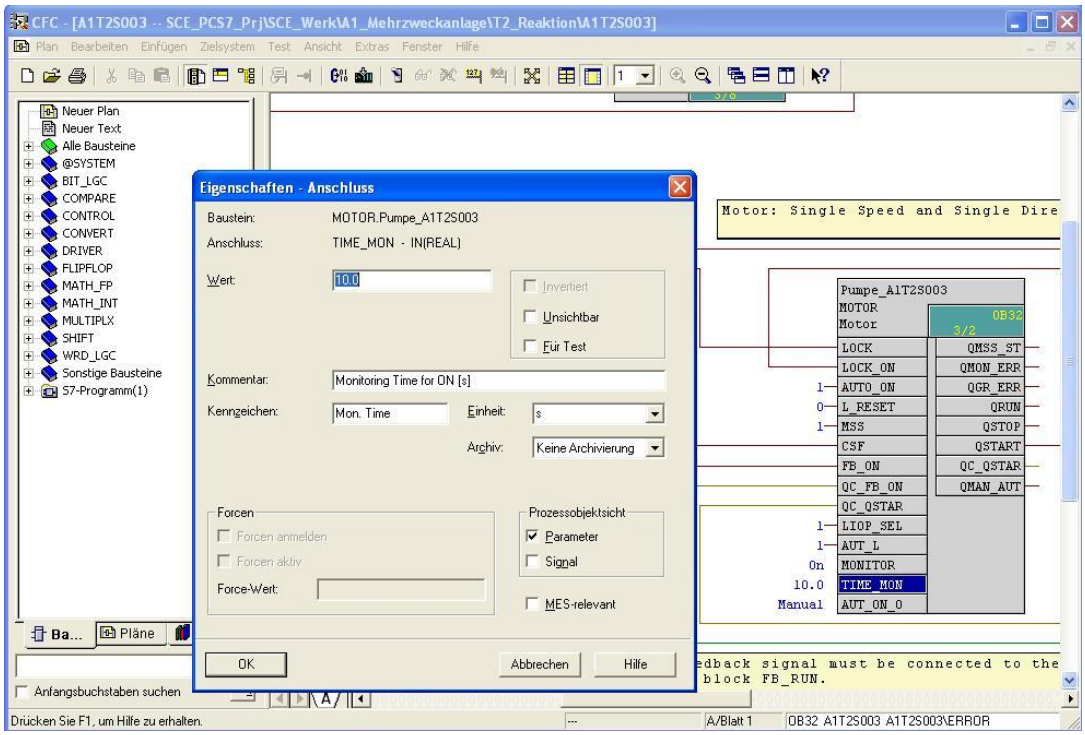


22. Bei den allgemeinen Eigenschaften wird der Name des Bausteins geändert.

    (→ Pumpe_A1T2S003 → OK)

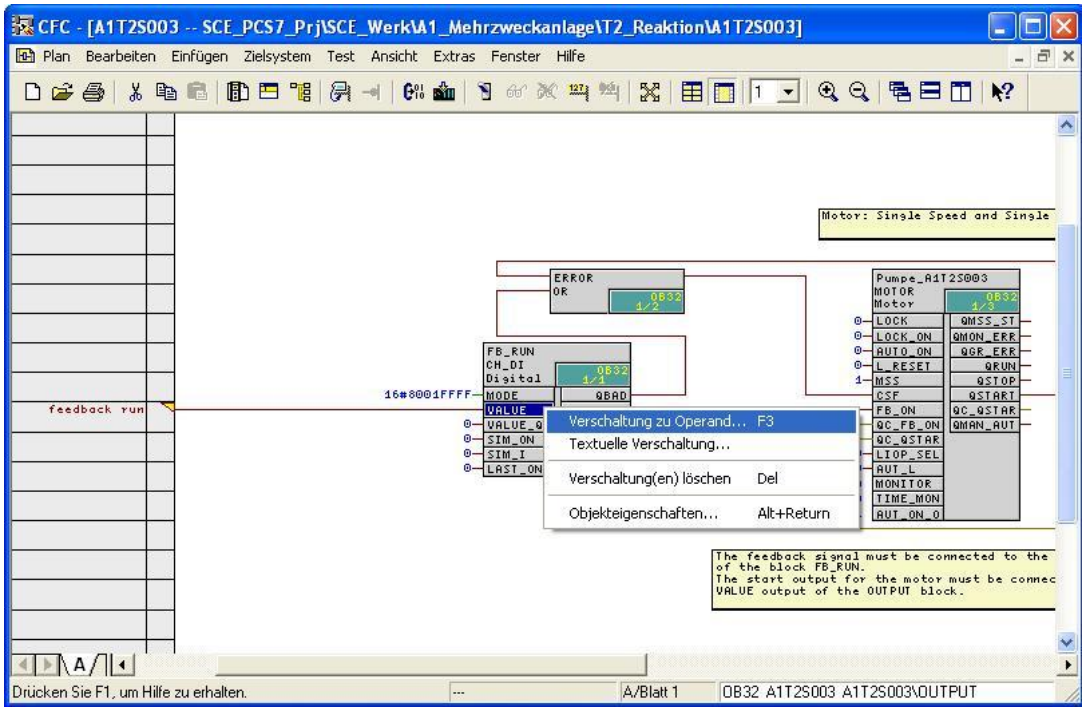


23. Nun wollen wir die Zeit für die Rückmeldungsüberwachung des Motorbausteins auf 10.0 Sekunden ändern. Dafür wird der Eigenschaftsdialog für den Eingang ‚TIME_MON' mit einem Doppelklick geöffnet. (→ TIME_MON→ 10.0 → OK)



24. Nun erfolgt noch die Verschaltung der Rückmeldung zum Eingangsoperanden. Dies geschieht, indem der Eingang ‚VALUE' am CH_DI Baustein mit der rechten Maustaste angeklickt wird. Dann wählt man Verschaltung zum Operand.

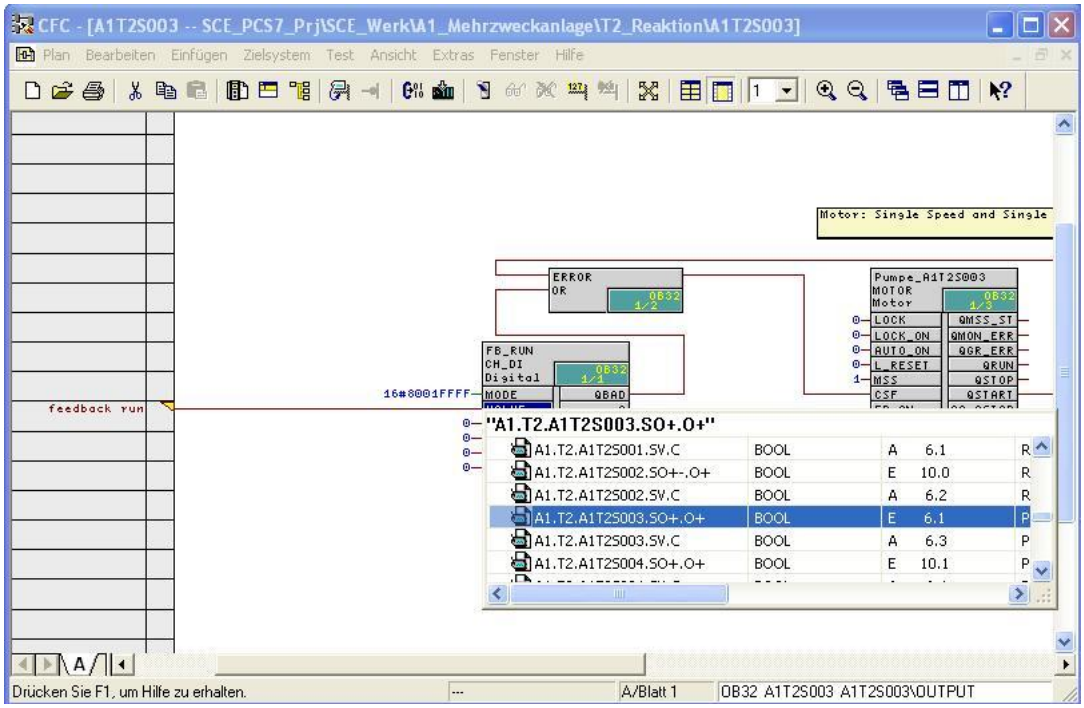
(→ VALUE → Verschaltung zum Operand)



Hinweis: Der CH-DI Baustein ist ebenso wie der CH-DO ein Treiberbaustein für die Kopplung zur Peripherie der SPS. Wird an einem dieser Bausteine der Wert VALUE mit einem Operanden verschaltet, der auch in der Hardwarekonfiguration projektiert ist, so wird später beim Übersetzungslauf automatisch der Eingang MODE mit Daten versorgt.

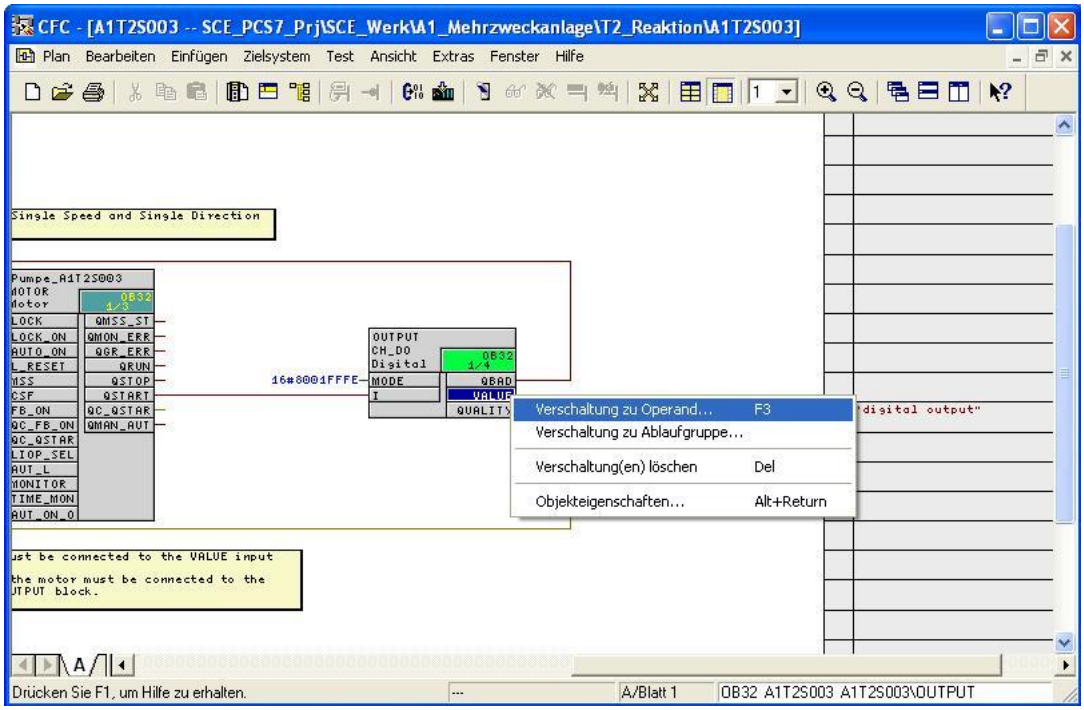
Damit das geschieht, muss später beim Übersetzungslauf ‚Baugruppentreiber erzeugen' angewählt werden.

25. Der Operand kann dann komfortabel direkt aus der Symboltabelle ausgewählt werden. (→ A1.T2.A1T2S003.S0+.0+)

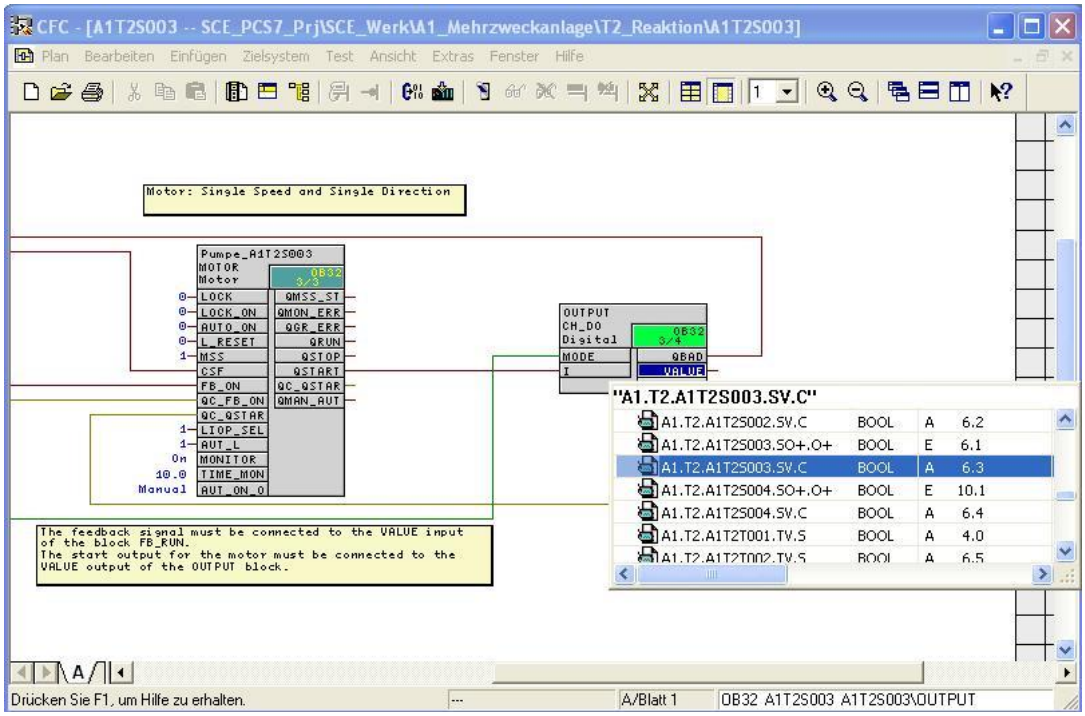


26. Nun erfolgt noch die Verschaltung der Ansteuerung des Ausgangsoperanden. Dies geschieht, indem der Eingang ‚VALUE' am CH_DO Baustein mit der rechten Maustaste angeklickt wird. Dann wählt man Verschaltung zum Operand.

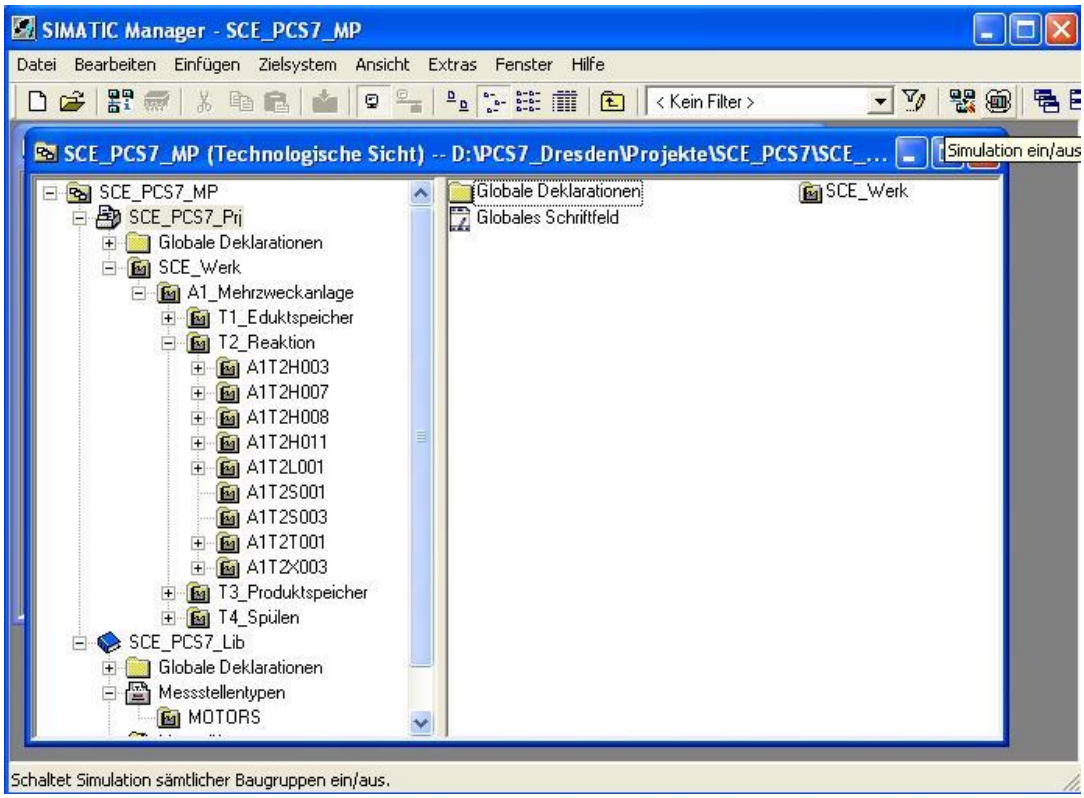
    (→ VALUE → Verschaltung zum Operand)



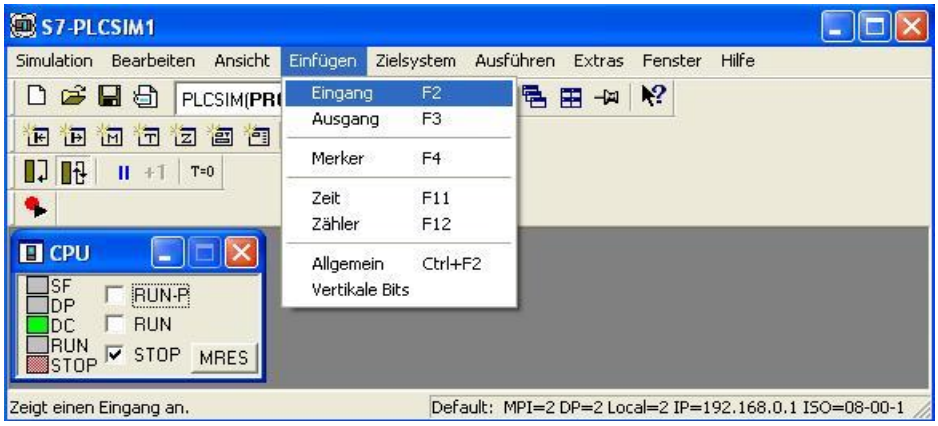
27. Der Operand kann dann wieder komfortabel direkt aus der Symboltabelle ausgewählt werden. (→ A1.T2.A1T2S003.SV.C)



28. Bevor das Programm für den Pumpenmotor übersetzt und geladen werden kann, muss nun noch die SPS- Simulation ***S7-PLCSIM*** gestartet werden. (→ )



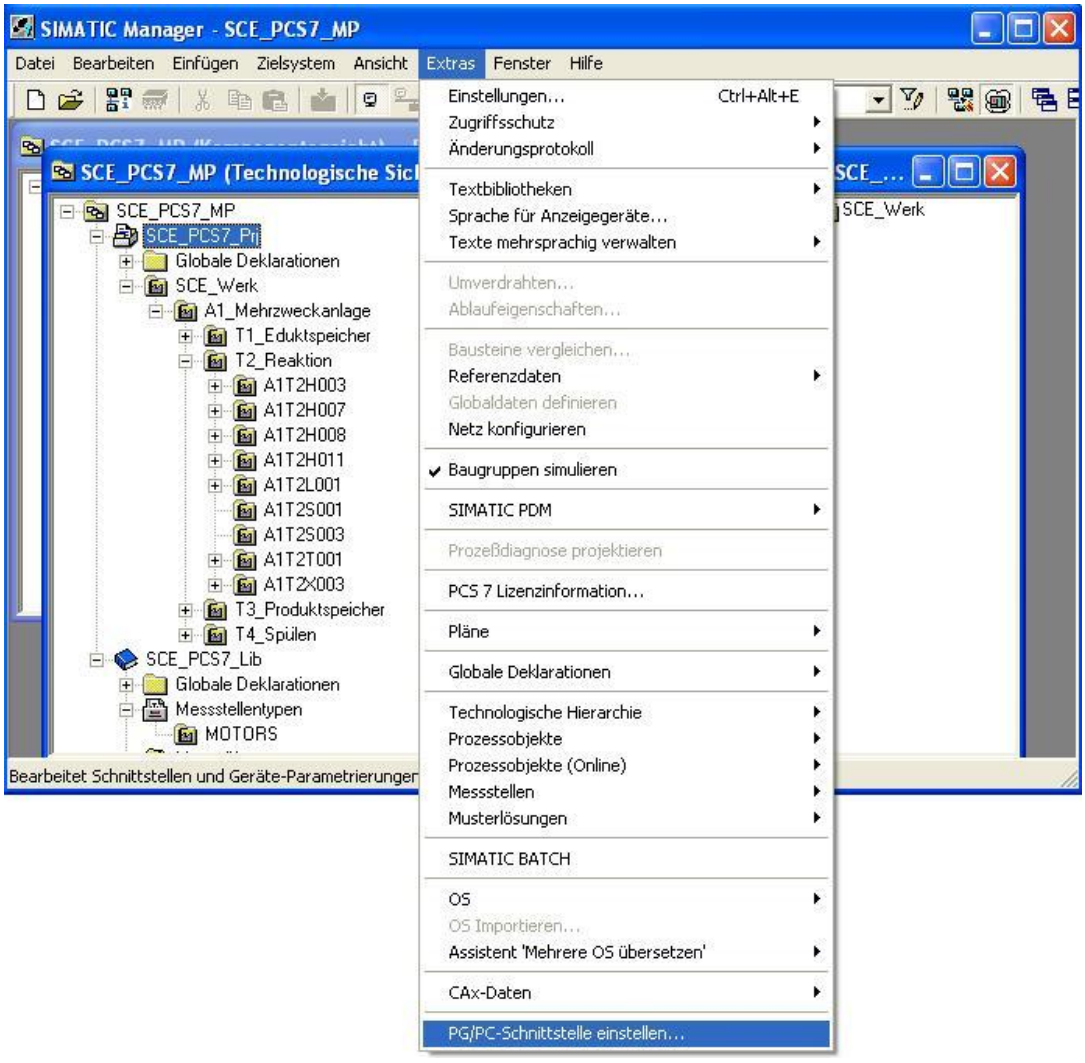
29. Die SPS-Simulation verhält sich wie eine richtige SIMATIC S7 CPU. Jedoch müssen Ein- und Ausgänge zuerst eingefügt werden, bevor diese beobachtet und geschaltet werden können. (→ Einfügen → Eingang → Einfügen → Ausgang)



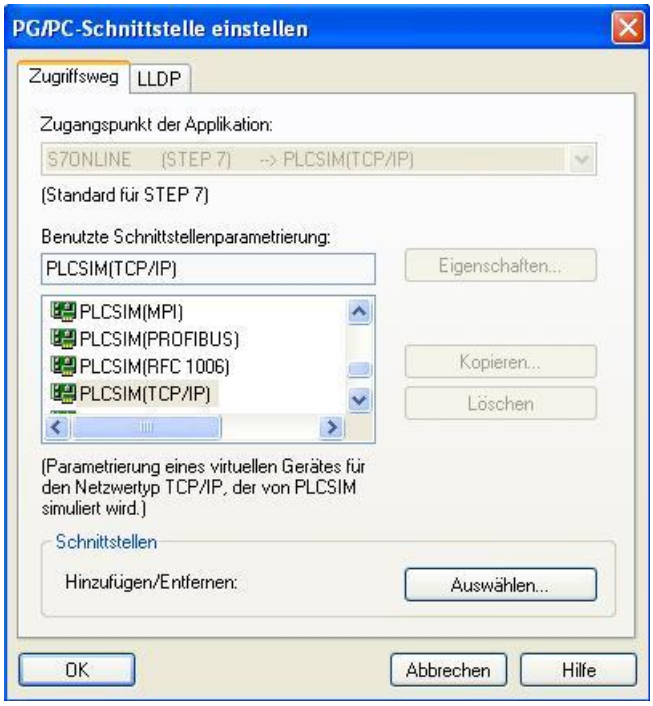
30. Nun müssen noch die richtigen Byte-Adressen eingetragen werden. (→ EB6 → AB6)

31. Damit aus dem SIMATIC Manager auch über die richtige Schnittstelle in ***S7-PLCSIM*** geladen werden kann, wird dann noch die PG/PC-Schnittstelle richtig eingestellt.

    (→ SIMATIC Manager → Extras → PG/PC-Schnittstelle einstellen)

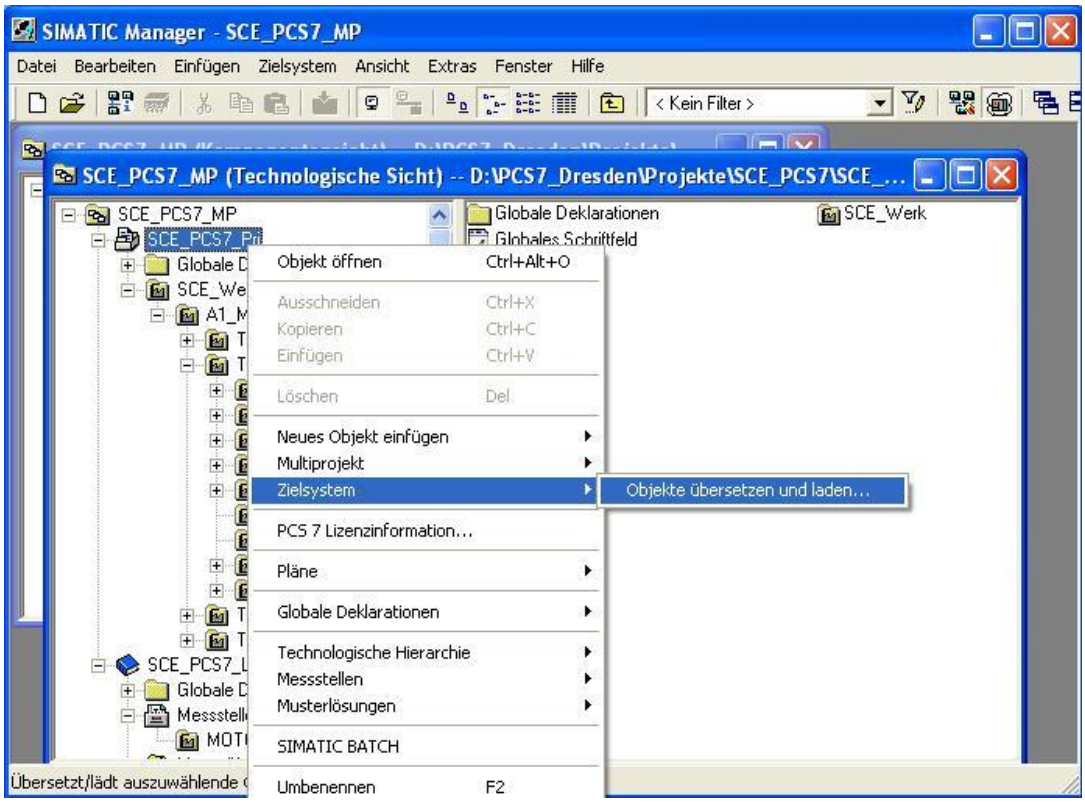


32. Als Schnittstelle wird hier PLCSIM(TCP/IP) eingestellt. (→ PLCSIM(TCP/IP) → OK)

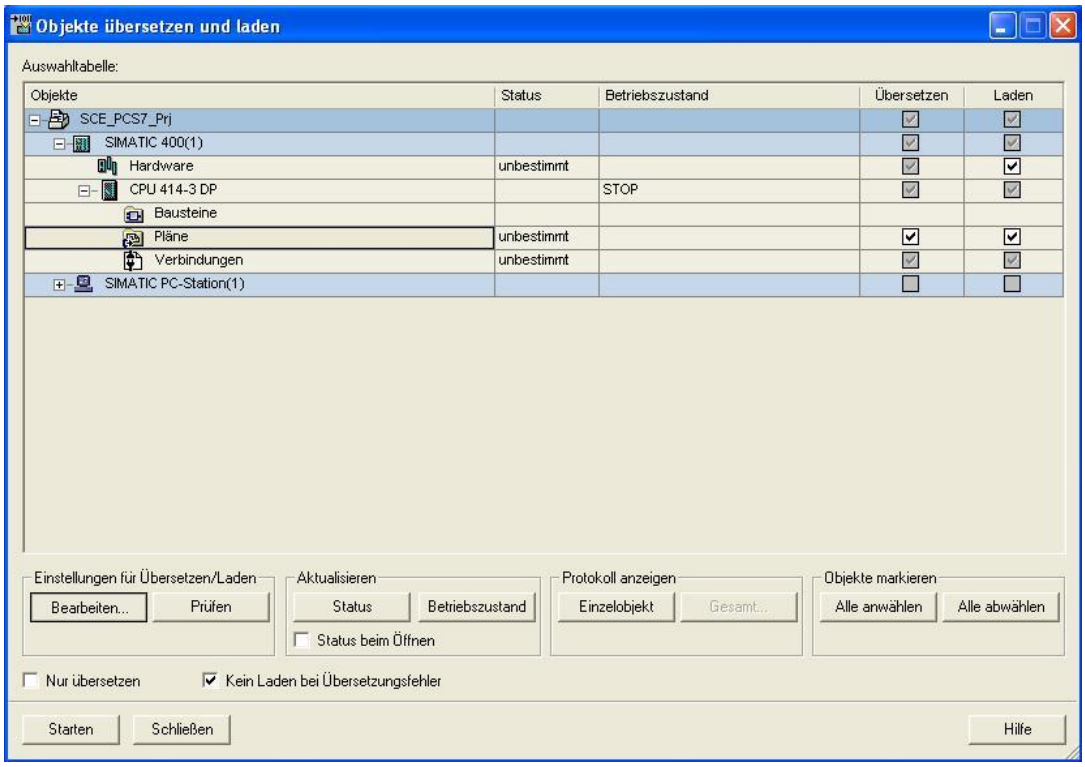


33. Nun kann in der Technologischen Sicht der Projektordner markiert und mit dem Übersetzen und Laden der Objekte begonnen werden.

(→ Technologische Sicht → SCE_PCS7_Prj → Zielsystem → Objekte übersetzen und laden)

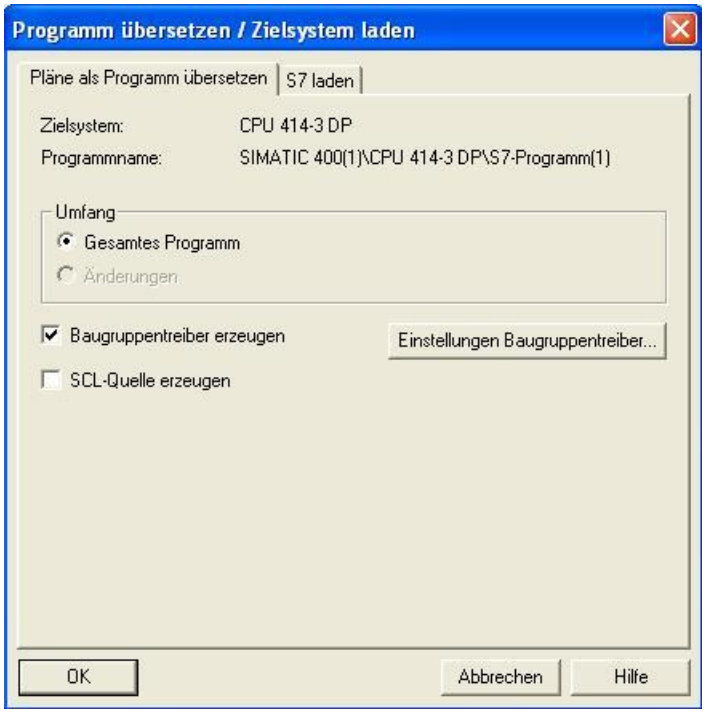


34. In der folgenden Auswahl wird dann für Hardware und die Pläne ‚Übersetzen und Laden' angewählt. Dann wird der Ordner ‚Pläne' markiert und dessen Einstellungen kontrolliert. (→ ☑ → ☑ → ☑ → Pläne → Bearbeiten)

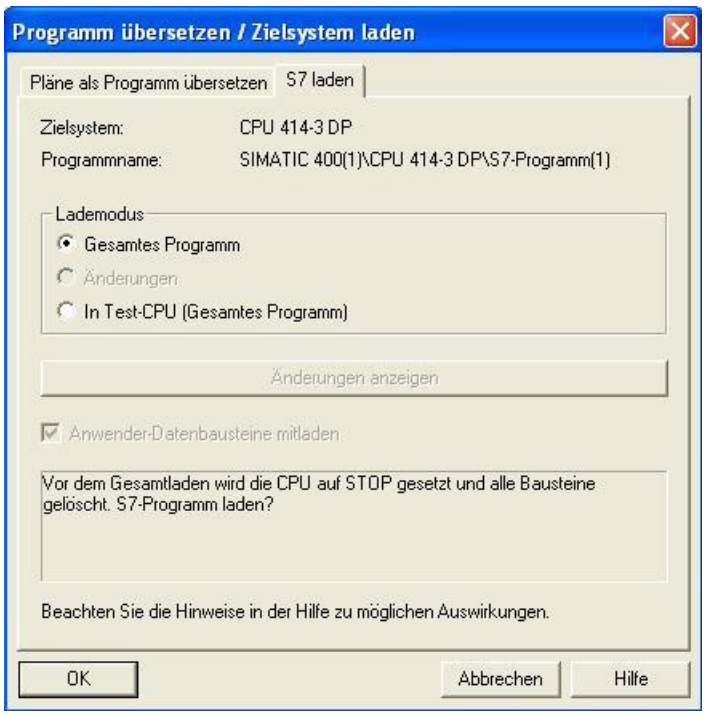


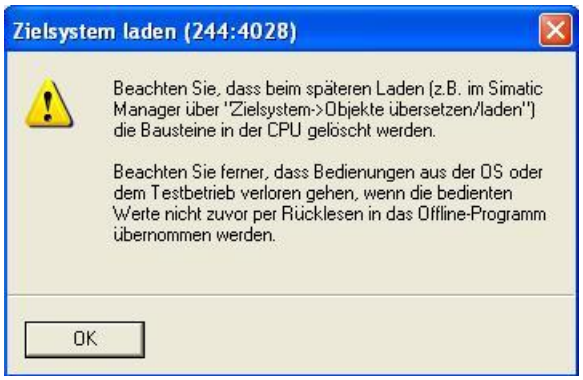
35. Beim Übersetzen der Pläne ist es wichtig das gesamte Programm zu übersetzen und die Baugruppentreiber erzeugen zu lassen.

    (→ Gesamtes Programm → Baugruppentreiber erzeugen → S7 laden)

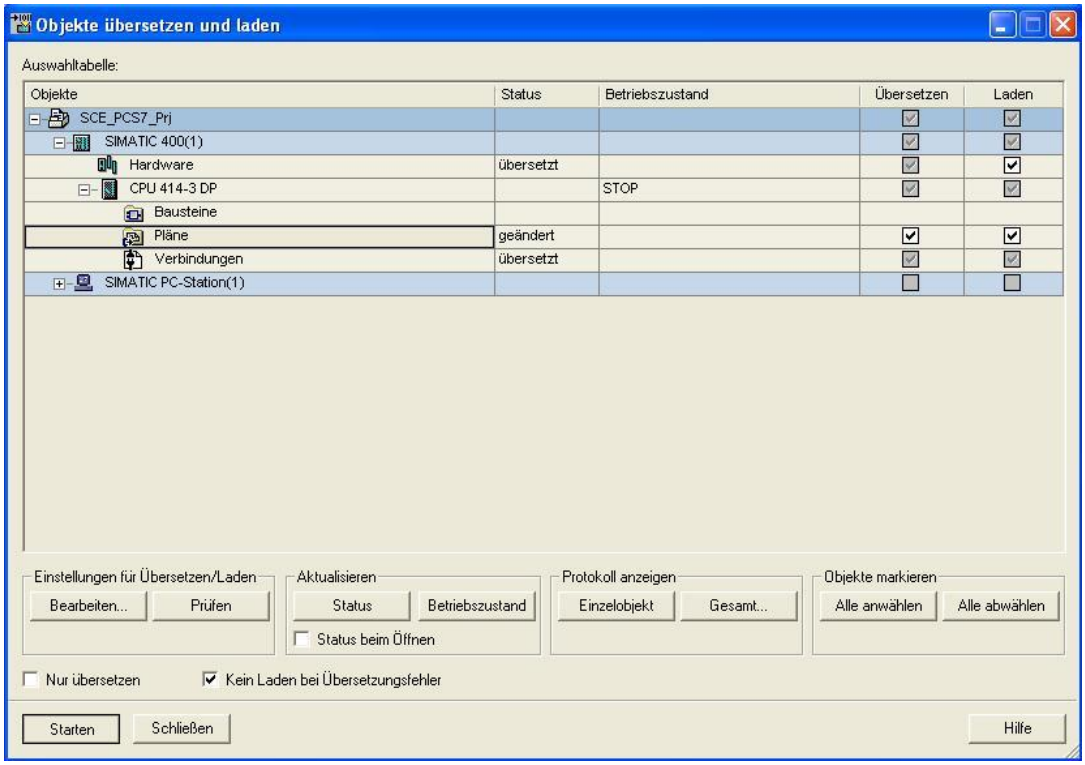


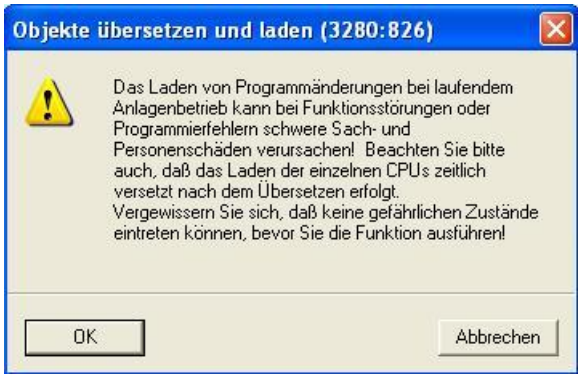
36. Beim Laden der Pläne ist es ebenfalls wichtig das gesamte Programm laden zu lassen. (→ Gesamtes Programm → OK → OK)

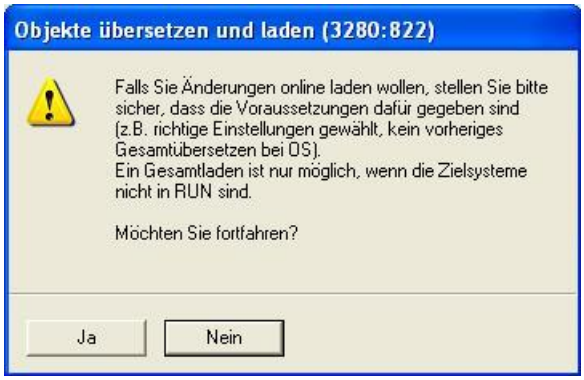




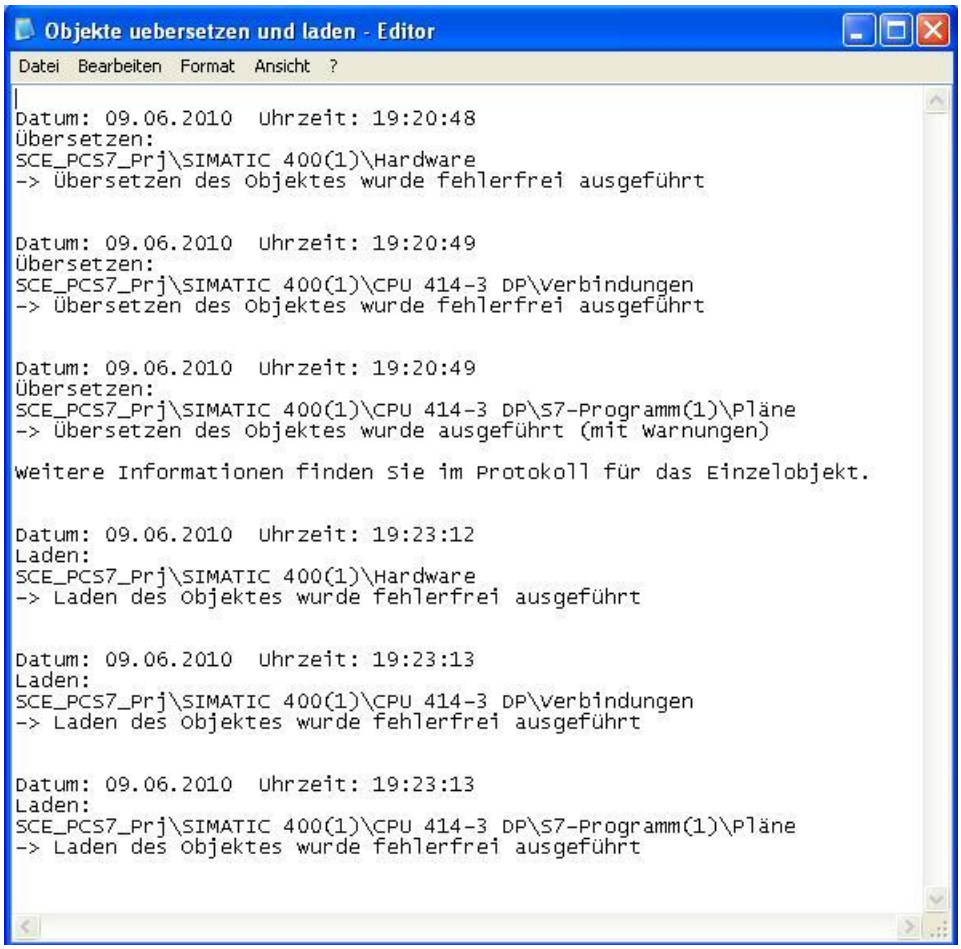
37. Schließlich kann ‚Übersetzen und Laden' gestartet werden. Die Warnungen und Hinweise zur Anlagensicherheit sollten sorgfältig gelesen werden. Die CPU muss vor ‚Übersetzen und Laden' auf ‚STOP' geschaltet worden sein. (→ Starten → OK → Ja)



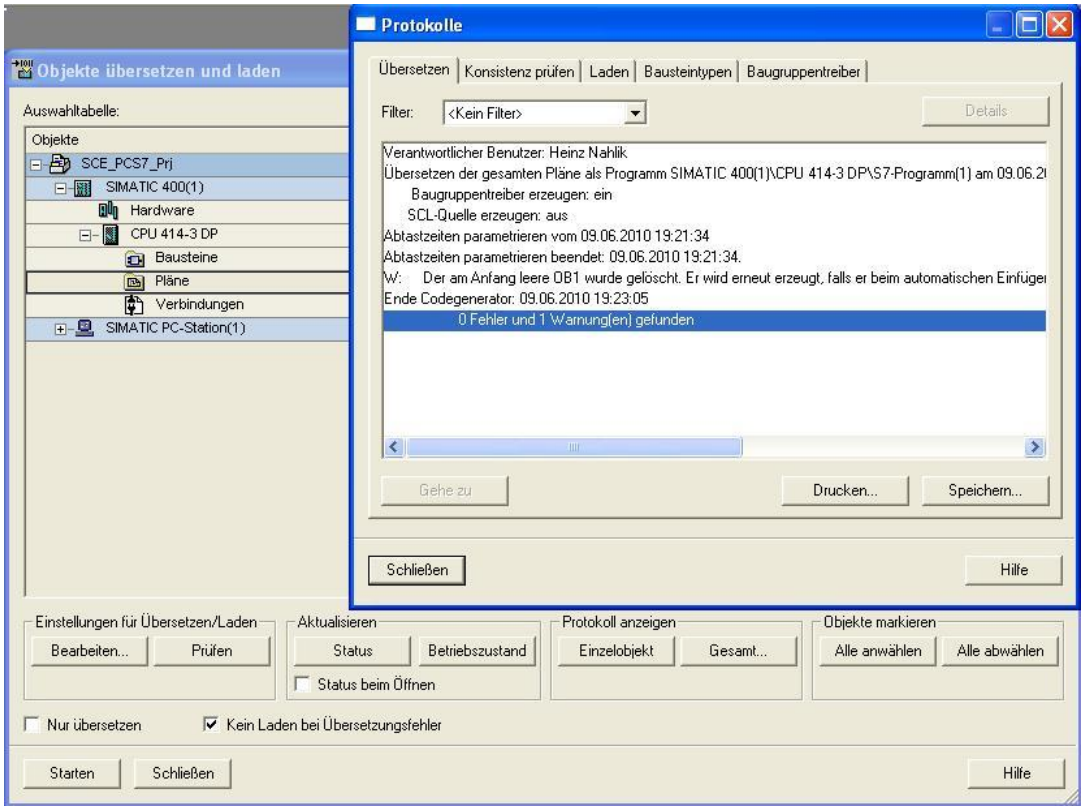




38. Am Ende werden in einem Protokoll Fehler und Warnungen angezeigt. Wir schließen das Fenster. (→)



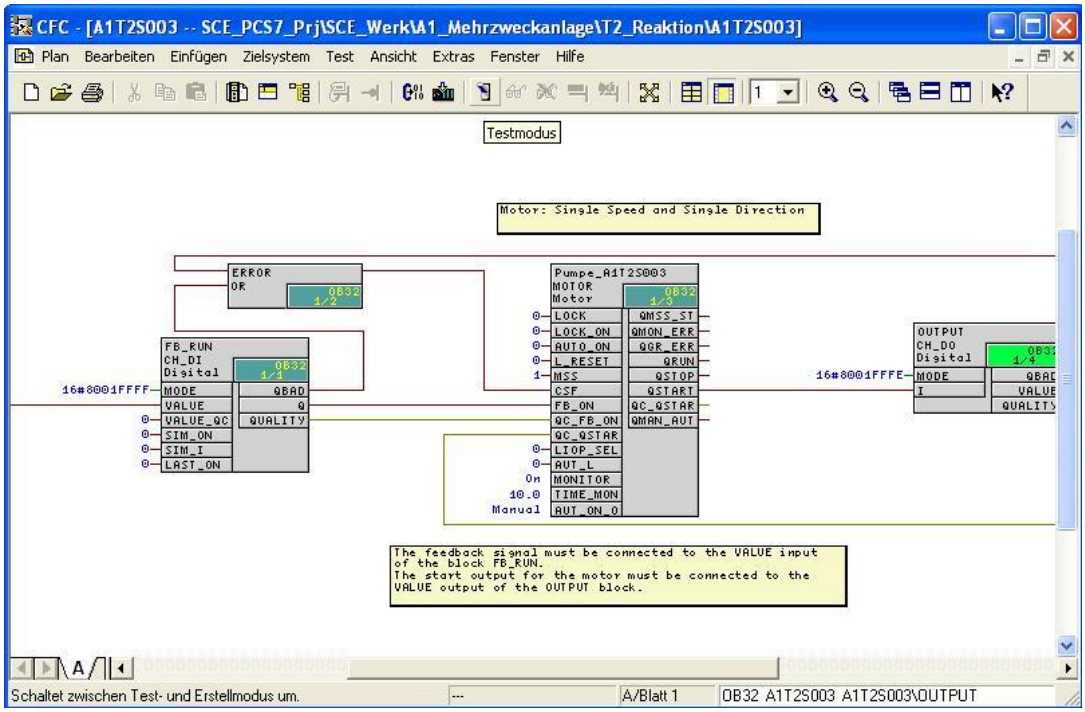
39. Wollen Sie Details zu dem Protokoll ansehen, so müssen Sie bei Protokoll anzeigen auf Einzelobjekt klicken. (→ Einzelobjekt → Schließen → Schließen)



Hinweis: Hier erscheint lediglich die Warnung, dass der leere OB1 vom System gelöscht worden ist.

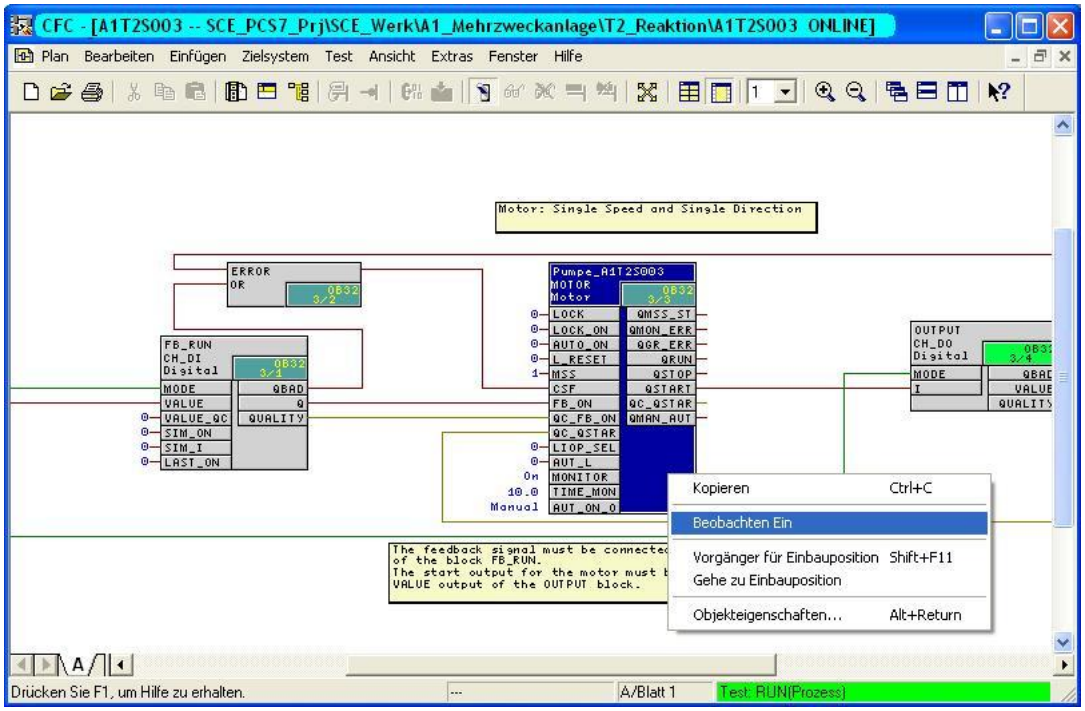
40. Zum Testen des Programms wird nun die CPU in ***S7-PLCSIM*** auf ‚RUN-P' geschaltet. (→ S7-PLCSIM → RUN-P)

41. Bevor im CFC die einzelnen Bausteine beobachtet werden können, muss der Plan zuerst einmal in den Testmodus geschaltet werden. (→ CFC → )



42. Dann kann immer noch nicht beobachtet werden, solange nicht die einzelnen Bausteine explizit zum Beobachten eingeschaltet wurden.

(→ Pumpe_A1T2S003 → Beobachten Ein)



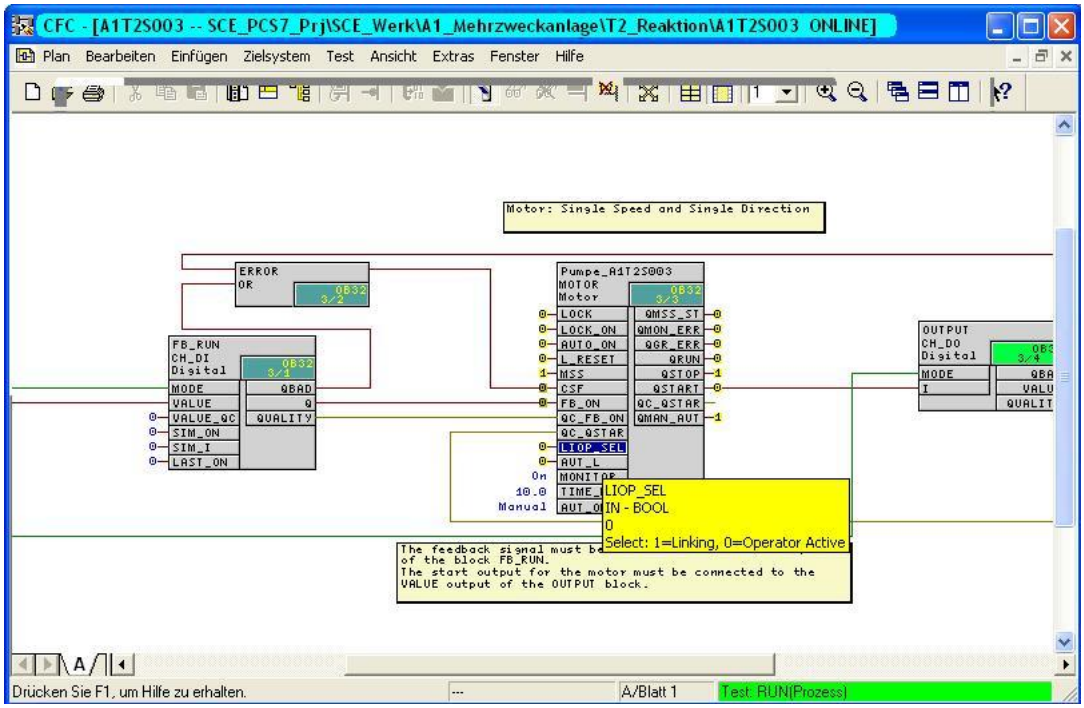
43. Um nun den Pumpenmotor aus dem CFC heraus einschalten zu können, müssen zuerst noch einige Eingänge geschaltet werden. Es sind dies in folgender Reihenfolge:

    Zum Aktivieren der Verschaltung und Passivieren der Bedienung in ***WinCC***
    LIOP_SEL == 1

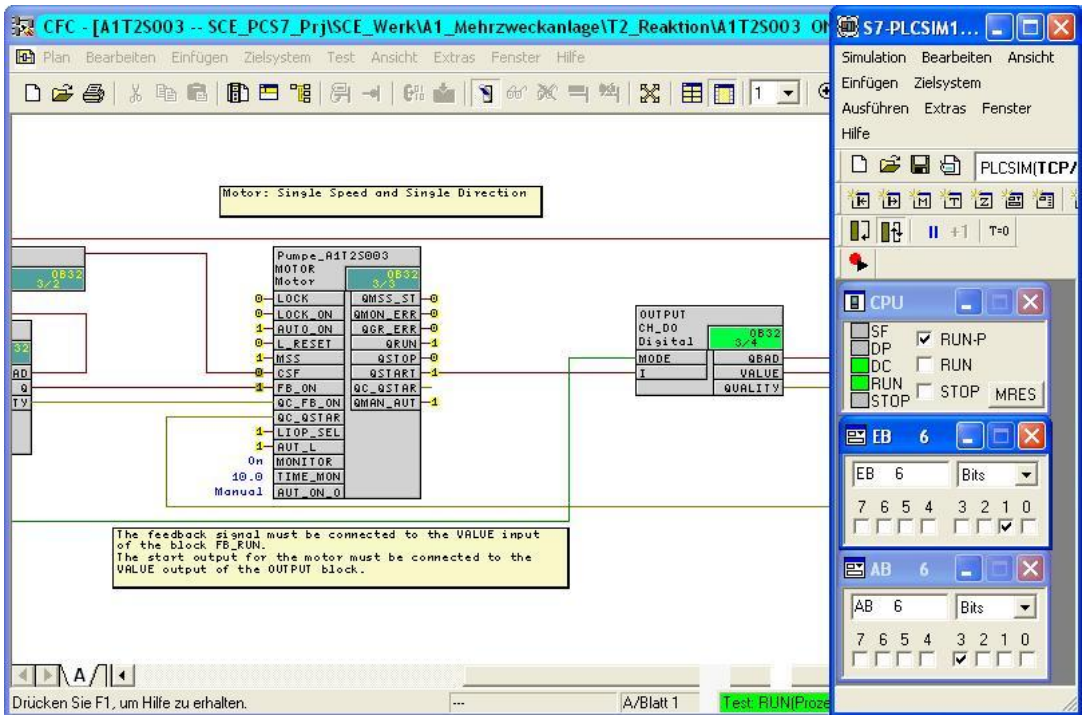
    Verschaltung für Hand-/ Automatikbetrieb auf Automatik
    AUT_L == 1

    Einschalten in Automatikbetrieb
    AUTO_ON == 1

    (→ LIOP_SEL → 1 → OK → AUT_L → 1 → OK → AUTO_ON → 1 → OK)



44. Der Prozess sowie die Reaktion der Anlage kann jetzt im CFC und im ***S7-PLCSIM*** beobachtet werden.

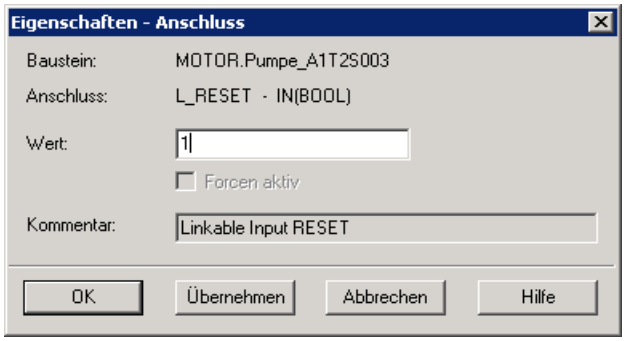


Hinweis: Beim Testen sollte nicht vergessen werden innerhalb von 10 Sekunden nach Ansteuerung des Ausgangs A 6.3 in ***S7-PLCSIM*** die Rückmeldung E 6.1 zu setzen. Wird dies vergessen, so schaltet der Baustein Pumpe_A1T2S003 ab und gibt einen Fehler aus.

Hinweis: Der Pumpenbaustein schaltet bei fehlender Rückmeldung nicht nur das Stellsignal QSTART auf 0, sondern zeigt durch Setzen des Ausgangs QMON_ERR auf 1 auch an, dass die Laufmeldung der Pumpe nicht rechtzeitig eingegangen war. Um Schäden durch wiederholte Einschaltversuche zu vermeiden, muss der Pumpenbaustein erst zurückgesetzt werden, bevor ein neuerlicher Versuch gestartet werden kann.

Hierzu ist der Eingang L_RESET kurz auf 1 und anschließend wieder auf 0 zu setzen!

Mit Doppelklick auf diesen Eingangsparameter des Pumpenbausteins Pumpe_A1T2S003 wird folgender Dialog geöffnet. Im Feld Wert wird zunächst eine 1 eingetragen, dieser Wert wird mit Click auf den Button ‚Übernehmen' an das Leitsystem übertragen und die Fehlerausgänge auf 0 gesetzt. Um zur normalen Funktionsweise zurückzukehren soll L_RESET durch Eingabe von 0 und nochmals ‚Übernehmen' abschließend auf den ursprünglichen Zustand zurückgesetzt werden.



Hinweis: Wenn AUTO_ON beim Zurücksetzen von L_RESET noch auf 1 gesetzt ist, wird der Baustein sofort wieder das QSTART-Signal setzen – und wenn die Laufrückmeldung ausbleibt auch wieder ausschalten!

TESTEN DER AUTOMATISIERUNGSLOGIK MIT DER SIMULATION

Die manuelle Eingabe von Prozesszuständen an das simulierte Leitsystem ist beim Testen kleiner Funktionen noch mit vertretbarem Aufwand möglich. Bei aufwändigeren Abläufen mit mehreren dynamischen Prozessgrößen sind jedoch schnell die Grenzen des machbaren erreicht. Hier empfiehlt sich der Einsatz einer Prozesssimulation.

Für diesen Kurs wurden deshalb die wesentlichen Zusammenhänge des hier zu automatisierenden Prozesses mit der Simulationssoftware ***SIMIT*** abgebildet. Das Modell bildet das dynamische Verhalten der Pumpen, Ventile, Behälter, Reaktoren sowie das Vor-Ort Bedienpanel mit Hauptschalter, Notaus, Umschaltung auf lokale Vor-Ort Bedienung und die entsprechenden Bedienelemente ab. Die dynamischen Vorgänge sind gegenüber der Realität um den Faktor 5 bis 50 beschleunigt, um die Wartezeiten kurz zu halten.

Die Bedienoberfläche des Simulators ist in Abbildung 4 abgebildet. Sie stellt auf der linken Seite das Prozessschema sowie die Signalpegel von Stell- und Messgrößen dar. Auf der rechten Seite ist oben das ockerfarben hinterlegte Vor-Ort-Bedienpanel dargestellt, unten sind eine Reihe Steuerelemente für die Simulation angebracht.

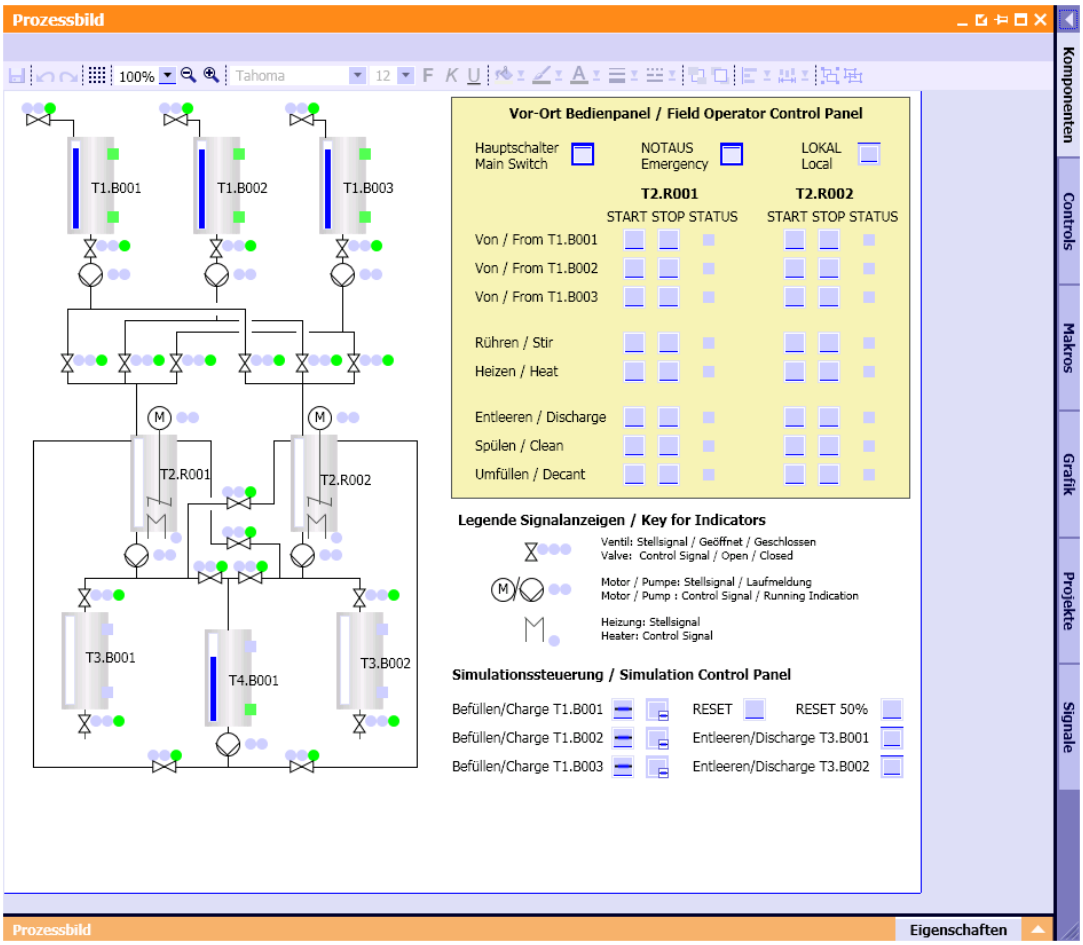


Abbildung 4: Bedienoberfläche der Prozesssimulation

Die Anwendung des Simulators ist denkbar einfach – es muss lediglich darauf geachtet werden, dass das Programm zum Einen nach ***S7-PLCSIM*** gestartet wird und zum Anderen die Belegung der Ein- und Ausgänge nicht verändert wurde.

Mit dem Simulator kann die Pumpenansteuerung nun sehr einfach überprüft werden:

1. Nach ***S7-PLCSIM*** wird das Simulationsprogramm gestartet.
2. Die Simulation beginnt mit 75 % gefüllten Edukttanks, alle anderen sind entleert. Dieser Zustand kann jederzeit mit der Option ‚RESET' in der Simulationssteuerung wieder hergestellt werden. Die Option ‚RESET 50 %' füllt alle Tanks wie in Abbildung 4 dargestellt zu 50 %.
3. Zum Testen wird der Motorbaustein wie im vorherigen Abschnitt unter Schritt 43 beschrieben angesteuert – in der Simulation leuchtet das Stellsignal der Pumpe grün auf.
4. Der simulierte Motoranlauf dauert etwa 2 Sekunden. Danach leuchtet zum einen die Laufanzeige des Motors in der Simulation grün auf, zum anderen wird der Signalpegel für den Binäreingang E 6.1 des Leitsystems gesetzt.
5. Um den Förderweg zu öffnen, muss zudem das Ventil zum Produkttank T3.B001 geöffnet werden. Dieses Ventil wird in der anschließenden Übung über eine geeignete Einzelsteuerfunktion angesprochen. In der Simulation kann bei eingeschalteter Pumpe und offenem Ventil beobachten werden, wie der Inhalt des Reaktors T2.R001 in den Produkttank T3.B001 gepumpt wird.
6. Über die Simulationssteuerungen können die Produkttanks entleert und die Edukttanks befüllt werden. Beim Befüllen über die Simulationssteuerung ist zunächst die Simulation von dem Leitsystem zu trennen – dazu wird der Button mit dem waagerechten Strich einmal angeklickt. Anschließend kann das Ventil mit dem Knopf rechts daneben geöffnet werden. Das Stellsignal leuchtet grün auf, nach etwa einer Sekunde folgt das Signal des Endlagenschalters für die Offen-Position, nach weiteren 5 bis 10 Sekunden sind erste Änderungen im Füllstand sichtbar.
7. Durch ‚RESET' und ‚RESET 50 %' kann die Simulation jederzeit wieder in einen definierten Zustand gebracht werden.

ÜBUNGEN

In den Übungsaufgaben soll Gelerntes aus der Theorie und der Schritt-für-Schritt-Anleitung umgesetzt werden. Hierbei soll das schon vorhandene Multiprojekt aus der Schritt-für-Schritt-Anleitung (PCS7_SCE_0104_R1105.zip) genutzt und erweitert werden.

Ziel dieser Übung ist es einen CFC zu erstellen, mit dem die Ventile der Anlage gesteuert werden können. Hierbei soll auf das Wissen aus der Schritt-für-Schritt-Anleitung aufgebaut werden, in der ein ähnlicher CFC zur Steuerung des Motors erstellt wurde.

Außerdem wird ein CFC zur Normierung des Füllstandes, also eines analogen Eingangswertes, erstellt.

ÜBUNGSAUFGABEN:

Die folgenden Übungen orientieren sich an der Schritt-für-Schritt-Anleitung. Für jede Übungsaufgabe können die entsprechenden Schritte der Anleitung als Hilfestellung genutzt werden.

1. Entscheiden Sie anhand von Tabelle 2, welche Einzelsteuerfunktion für das vorliegende Ventil genutzt werden soll und importieren Sie die Einzelsteuerfunktion in die Projektbibliothek. Die Vorlage für das Ventil soll dabei ebenfalls bei den Messstellentypen in der Stammdatenbibliothek eingefügt werden.
2. Im Planordner A1T3X001 soll eine Objektinstanz der gerade in der Stammdatenbibliothek erstellten Ventilsteuerung eingefügt werden. Passen Sie gegebenenfalls die Namen der Bausteine in der Objektinstanz an.
3. Benutzen Sie eine Objektinstanz ihres neu erstellten Objekttypen um das Ventil =SCE.A1.T3.V001 anzusteuern, mit dem eine Verbindung zwischen dem Reaktor =SCE.A1.T2.R001 und dem Produktbehälter =SCE.A1.T3.B001 geöffnet werden kann. Das Ventil ist über die in Tabelle 4 aufgeführten Symbole zu steuern.
4. Testen Sie nun Ihre Implementierung mit dem SIMIT-Modell.
5. Fügen Sie den FC 275 (CH_AI) aus der ***PCS 7 Library V71*** zu Ihrem Projekt hinzu.
6. Erstellen Sie einen neuen CFC im Planordner A1T2L001. Ziehen Sie den gerade hinzugefügten Baustein CH_AI (FC 275: Analog Input) auf das erste Blatt des CFC. Benennen Sie den CFC und den Baustein entsprechend. Verbinden Sie den ‘VALUE’-Eingang des Bausteins mit dem Symbol für den Füllstandistwert des Reaktors aus Tabelle 4. Setzen Sie den ‘VHRANGE’ des Bausteins auf -82,945 und den ‘VLRANGE’ auf 2398,945 (diese Werte dienen der Normierung des Messwertes).

Um sicherzustellen, dass die MODE-Eingänge der CH-DI und CH-DO Bausteine korrekt verschaltet werden, stellen Sie bitte sicher, dass die Option ‘Baugruppentreiber erzeugen’ beim Übersetzen ausgewählt ist.

Tabelle 4: Die Symbole für die Realisierung der Ventilsteuerung

Symbol	Adresse	Datentyp	Kommentar
A1.T3.A1T3X001.XV.C	A 2.0	BOOL	Auf/Zu-Ventil Zufluss Produkttank B001 Stellsignal
A1.T3.A1T3X001.GO+-.O+	E 12.3	BOOL	Auf/Zu-Ventil Zufluss Produkttank B001 Rückmeldung auf/ein
A1.T3.A1T3X001.GO+-.O-	E 12.5	BOOL	Auf/Zu-Ventil Zufluss Produkttank B001 Rückmeldung zu
A1.T2.A1T2L001.LISA+.M	EW 512	WORD	Füllstandistwert Reaktor R001

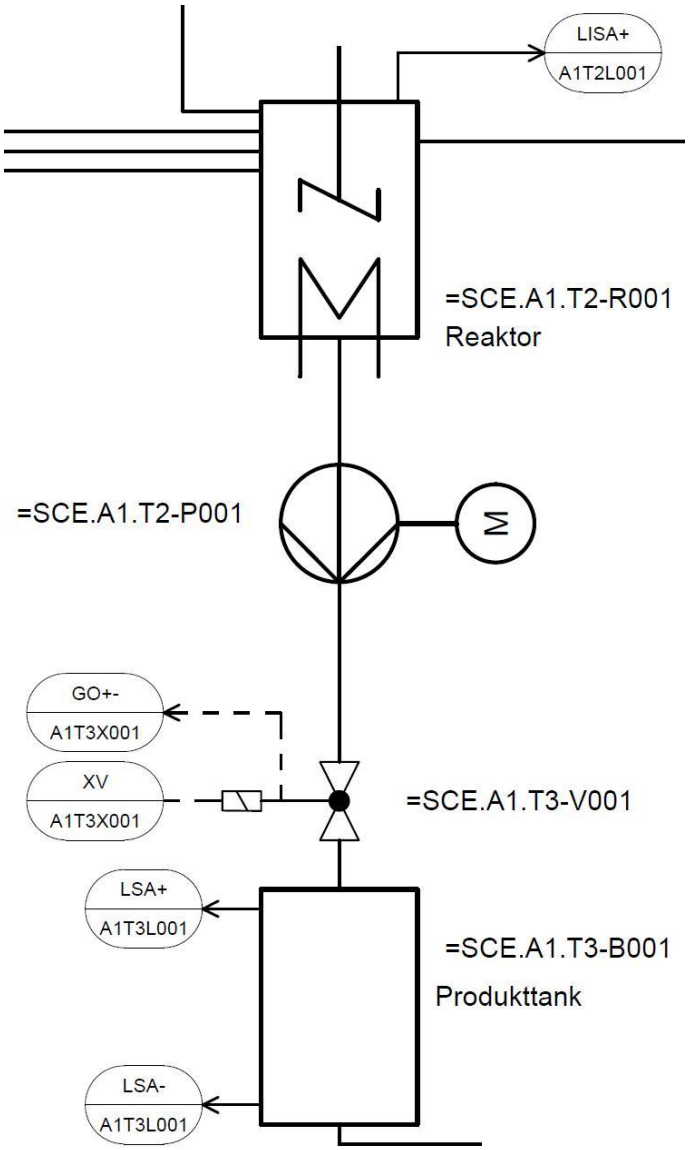


Abbildung 5: Ausschnitt aus dem R&I-Fließbild