SIEMENS

SCE Lehrunterlagen

Siemens Automation Cooperates with Education | 09/2017

TIA Portal Modul 052-201

Hochsprachenprogrammierung
mit SCL und SIMATIC S7-1500

Cooperates with Education
Automation
SIEMENS

Passende SCE Trainer Pakete zu dieser Lern-/ Lehrunterlagen

- **SIMATIC S7 CPU 1516F-3 PN/DP**
  Bestellnr.: 6ES7516-3FN00-4AB2
- **SIMATIC STEP 7 Professional V14 SP1 - Einzel-Lizenz**
  Bestellnr.: 6ES7822-1AA04-4YA5
- **SIMATIC STEP 7 Professional V14 SP1 - 6er Klassenraumlizenz**
  Bestellnr.: 6ES7822-1BA04-4YA5
- **SIMATIC STEP 7 Professional V14 SP1 - 6er Upgrade-Lizenz**
  Bestellnr.: 6ES7822-1AA04-4YE5
- **SIMATIC STEP 7 Professional V14 SP1 - 20er Studenten-Lizenz**
  Bestellnr.: 6ES7822-1AC04-4YA5

Bitte beachten Sie, dass diese Trainer Pakete ggf. durch Nachfolge-Pakete ersetzt werden.
Eine Übersicht über die aktuell verfügbaren SCE Pakete finden Sie unter: siemens.de/sce/tp

Fortbildungen

Für regionale Siemens SCE Fortbildungen kontaktieren Sie Ihren regionalen SCE Kontaktpartner:
siemens.de/sce/contact

Weitere Informationen rund um SCE

siemens.de/sce

Verwendungshinweis

Die SCE Lehrunterlage für die durchgängige Automatisierungslösung Totally Integrated Automation (TIA) wurde für das Programm „Siemens Automation Cooperates with Education (SCE)“ speziell zu Ausbildungszwecken für öffentliche Bildungs- und F&E-Einrichtungen erstellt. Die Siemens AG übernimmt bezüglich des Inhalts keine Gewähr.

Diese Unterlage darf nur für die Erstausbildung an Siemens Produkten/Systemen verwendet werden. D.h. sie kann ganz oder teilweise kopiert und an die Auszubildenden zur Nutzung im Rahmen deren Ausbildung ausgehändigt werden. Die Weitergabe sowie Vervielfältigung dieser Unterlage und Mitteilung ihres Inhalts ist innerhalb öffentlicher Aus- und Weiterbildungsstätten für Zwecke der Ausbildung gestattet. Ausnahmen bedürfen der schriftlichen Genehmigung durch die Siemens AG. Ansprechpartner: Herr Roland Scheuerer roland.scheuerer@siemens.com.

Zuwiderhandlungen verpflichten zu Schadensersatz. Alle Rechte auch der Übersetzung sind vorbehalten, insbesondere für den Fall der Patentierung oder GM-Eintragung.

Der Einsatz für Industriekunden-Kurse ist explizit nicht erlaubt. Einer kommerziellen Nutzung der Unterlagen stimmen wir nicht zu.

Wir danken der TU Dresden, besonders Prof. Dr.-Ing. Leon Urbas und der Fa. Michael Dziallas Engineering und allen weiteren Beteiligten für die Unterstützung bei der Erstellung dieser SCE Lehrunterlage.

Inhaltsverzeichnis

Hochsprachenprogrammierung mit S7-SCL

1. Zielstellung

In diesem Kapitel lernen Sie die grundlegenden Funktionen der Hochsprache S7-SCL kennen. Weiterhin werden Testfunktionen zur Beseitigung logischer Programmierfehler aufgezeigt.

Es können die unter Kapitel 3 aufgeführten SIMATIC S7-Steuerungen eingesetzt werden.

2. Voraussetzung

Dieses Kapitel baut auf der Hardwarekonfiguration einer SIMATIC S7 auf. Es kann mit beliebigen Hardwarekonfigurationen, die digitale Eingangs- und Ausgangskarten besitzen, realisiert werden. Zur Durchführung dieses Kapitels können Sie z.B. auf das folgende Projekt zurückgreifen:

„SCE_DE_012_101_Hardwarekonfiguration_CPU1516F…..zap13“

Weiterhin sollten Grundlagenkenntnisse über Hochsprachenprogrammierung wie z.B. Pascal bekannt sein.

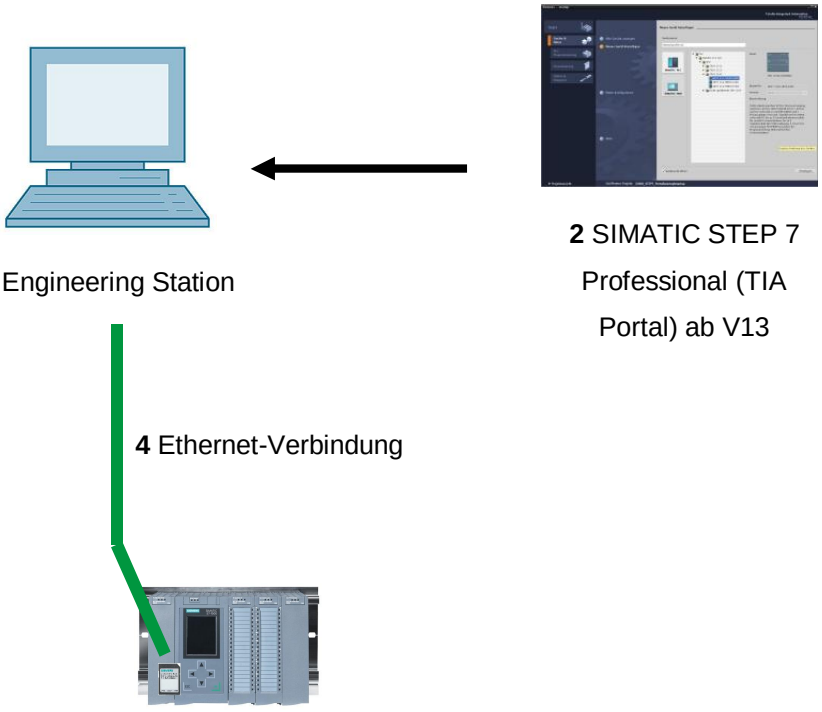
3. Benötigte Hardware und Software

1 Engineering Station: Voraussetzungen sind Hardware und Betriebssystem (weitere Informationen siehe Readme/Liesmich auf den TIA Portal Installations-DVDs)

2 Software SIMATIC STEP 7 Professional im TIA Portal – ab V13

3 Steuerung SIMATIC S7-1500/S7-1200/S7-300, z.B. CPU 1516F-3 PN/DP – ab Firmware V1.6 mit Memory Card und 16DI/16DO sowie 2AI/1AO

4 Ethernet-Verbindung zwischen Engineering Station und Steuerung



4. Theorie

4.1 Zur Programmiersprache S7-SCL

S7-SCL (Structured Control Language) ist eine höhere Programmiersprache, die sich an Pascal orientiert und eine strukturierte Programmierung ermöglicht. Die Sprache entspricht der in der Norm DIN EN-61131-3 (IEC 61131-3) festgelegten Ablaufsprache SFC „Sequential Function Chart“. S7-SCL enthält neben Hochsprachenelementen auch typische Elemente der SPS als Sprachelemente wie Eingänge, Ausgänge, Zeiten, Merker, Bausteinaufrufe usw.. Sie unterstützt das Bausteinkonzept von STEP 7 und ermöglicht daher neben Anweisungsliste (AWL), Kontaktplan (KOP) und Funktionsplan (FUP) die normkonforme Programmierung von Bausteinen. D.h. S7-SCL ergänzt und erweitert die Programmiersoftware STEP 7 mit ihren Programmiersprachen KOP, FUP und AWL.

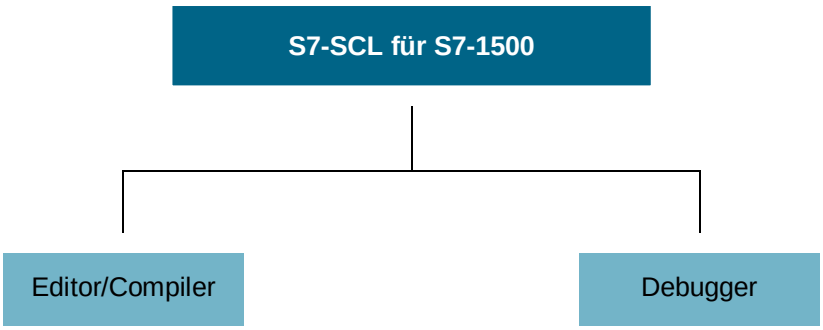
Sie müssen nicht jede Funktion selbst erstellen, sondern können auf vorgefertigte Bausteine wie Systemfunktionen oder Systemfunktionsbausteine zurückgreifen, die im Betriebssystem der Zentralbaugruppe vorhanden sind.

Bausteine, die mit S7-SCL programmiert sind, können Sie mit AWL-, KOP- und FUP-Bausteinen mischen. Das bedeutet, dass ein mit S7-SCL programmierter Baustein einen anderen Baustein, der in AWL, KOP oder FUP programmiert ist, aufrufen kann. Entsprechend können S7-SCL Bausteine auch in AWL-, KOP- und FUP-Programmen aufgerufen werden.

Die Testfunktionen von S7-SCL ermöglichen die Suche nach logischen Programmierfehlern in einer fehlerfreien Übersetzung.

4.2 Zur Entwicklungsumgebung S7-SCL

Zur Verwendung und zum Einsatz von S7-SCL gibt es eine Entwicklungsumgebung, die sowohl auf spezifische Eigenschaften von S7-SCL, als auch auf STEP 7 abgestimmt ist. Diese Entwicklungsumgebung besteht aus einem Editor/Compiler und einem Debugger.



Editor/Compiler

Der S7-SCL-Editor ist ein Texteditor, mit dem beliebige Texte bearbeitet werden können. Die zentrale Aufgabe, die Sie mit ihm durchführen können, ist das Erzeugen und Bearbeiten von Bausteinen für STEP 7-Programme. Während der Eingabe erfolgt eine grundlegende Syntaxprüfung, welche das fehlerfreie Programmieren vereinfacht. Syntaxfehler werden in unterschiedlichen Farben dargestellt.

Der Editor bietet folgende Möglichkeiten:

- Programmierung eines S7-Bausteines in der Sprache S7-SCL.
- Komfortables Einfügen von Sprachelementen und Bausteinaufrufen mittels Drag & Drop.
- Direkte Syntaxprüfung während der Programmierung.
- Einstellung des Editors nach Ihren Anforderungen, z.B. durch syntaxgerechtes Einfärben der verschiedenen Sprachelemente.
- Überprüfung des fertiggestellten Bausteines mittels übersetzen.
- Anzeigen aller Fehler und Warnungen, die beim Übersetzen auftreten.
- Lokalisieren der fehlerhaften Stellen im Baustein, optional mit Fehlerbeschreibung und Angaben zur Fehlerbeseitigung.

Debugger

Der S7-SCL-Debugger ermöglicht ein Programm in seinem Ablauf im Automatisierungssystem (AS) zu kontrollieren und somit mögliche logische Fehler zu finden.

S7-SCL bietet dazu zwei verschiedene Testmodi an:

– Kontinuierliches Beobachten

– Schrittweises Beobachten

Mit dem „Kontinuierlichen Beobachten“ können Sie eine Gruppe von Anweisungen innerhalb eines Bausteins testen. Während des Testlaufs werden die Werte der Variablen und Parameter in chronologischer Abfolge angezeigt und – sofern möglich – zyklisch aktualisiert.

Beim „Schrittweisen Beobachten“ wird der logische Programmablauf nachvollzogen. Sie können den Programm-Algorithmus Anweisung für Anweisung ausführen und in einem Ergebnisfenster beobachten, wie sich die dabei bearbeiteten Variableninhalte ändern

Ob das „Schrittweise Beobachten“ möglich ist hängt von der eingesetzten CPU ab. Diese muss den Einsatz von Haltepunkten unterstützen. Die in diesem Dokument eingesetzte CPU unterstützt keine Haltepunkte.

5. Aufgabenstellung

5.1 Beispielaufgabe Tankinhalt

Im ersten Teil soll die Berechnung eines Tankinhaltes programmiert werden.

5.2 Erweiterung der Beispielaufgabe

Im zweiten Teil soll die Aufgabe erweitert und eine Fehlerauswertung programmiert werden.

6. Planung

Der Tank hat die Form eines stehenden Zylinders. Der Füllstand des Inhaltes wird mit einem Analogsensor gemessen. Für den ersten Test soll der Wert des Füllstandes schon normiert – in der Einheit Meter – vorliegen.

Globale Parameter wie z.B. der Durchmesser und die Höhe des Tanks sollen in einem globalen Datenbaustein „Daten_Tank“ strukturiert abgelegt werden.

Das Programm zur Berechnung des Inhaltes soll in einer Funktion „Berechnung_Tankinhalt“ geschrieben werden und die Parameter die Einheit Meter bzw. Liter verwenden.

6.1 Globaler Datenbaustein „Daten_Tank“

Die globalen Parameter werden in einem globalen Datenbaustein in mehreren Strukturen abgelegt.

Name	Datentyp	Startwert	Kommentar
Abmessungen	STRUCT		
Hoehe	REAL	12.0	in Meter
Durchmesser	REAL	3.5	in Meter
Messwerte	STRUCT		
Fuellstand_per	INT	0	Wert zwischen 0...27648
Fuellstand_skal	REAL	0.0	Wert zwischen 0…12.0
Inhalt	REAL	0.0	Inhalt des Tanks in Liter
Fehlerflags	STRUCT		
berechne_inhalt	BOOL		im Fehlerfall = TRUE

Tabelle 1: Parameter im Datenbaustein "Daten_Tank"

6.2 Funktion „Berechnung_Tankinhalt“

Dieser Baustein berechnet den Inhalt des Tanks in Litern.

Im ersten Schritt soll keine Überprüfung auf Sinnhaftigkeit der übergebenen Parameter erfolgen.

Für diesen Schritt sind folgende Parameter erforderlich:

Input	Datentyp	Kommentar
Durchmesser	REAL	Durchmesser des Zylindertanks in Meter
Fuellstand	REAL	Füllstand des Tankinhaltes in Meter
Output		
Inhalt	REAL	Inhalt des Zyklindertanks in Liter

Tabelle 2: Parameter für FC "Berechnung_Tankinhalt" im ersten Schritt

Zur Lösung der Aufgabe wird die Formel zur Volumenberechnung eines stehenden Zylinders angewendet. Der Umrechnungsfaktor 1000 wird verwendet, um das Ergebnis in Litern zu berechnen.

$$V = \frac{d^2}{4} \cdot p \cdot h \quad \Rightarrow \quad \#Inhalt = \frac{\#Durchmesser^2}{4} \cdot 3.14159 \cdot \#Fuellstand \cdot 1000$$

6.3 Erweiterung der Funktion „Berechnung_Tankinhalt“

Der zweite Schritt prüft, ob der Durchmesser größer als Null ist. Des Weiteren soll getestet werden, ob der Füllstand größer oder gleich Null und kleiner oder gleich der Höhe des Tanks ist. Im Fehlerfall wird der neue Parameter „er“ auf TRUE gesetzt und der Parameter „Inhalt“ erhält den Wert -1.

Erweitern Sie die Schnittstelle dazu um den Parameter „er“ und „Hoehe“.

Input	Datentyp	Kommentar
Hoehe	REAL	Hoehe des Zylindertanks in Meter
Durchmesser	REAL	Durchmesser des Zylindertanks in Meter
Fuellstand	REAL	Füllstand des Tankinhaltes in Meter
Output		
er	BOOL	Fehlerflag; bei Fehler = TRUE
Inhalt	REAL	Inhalt des Zyklindertanks in Liter

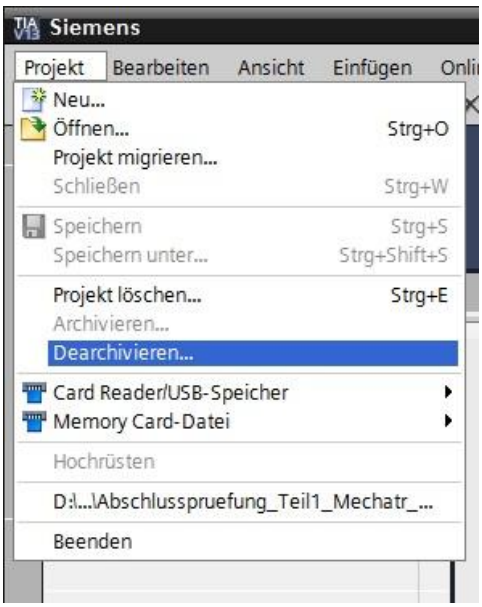
Tabelle 3: Parameter für FC "Berechnung_Tankinhalt" im zweiten Schritt

7. Strukturierte Schritt-für-Schritt-Anleitung

Im Folgenden finden Sie eine Anleitung zur Umsetzung der Planung. Sollten Sie gut zurechtkommen, so reichen Ihnen die nummerierten Schritte zur Bearbeitung aus. Ansonsten folgen Sie einfach den folgenden detaillierten Schritten der Anleitung.

7.1 Dearchivieren eines vorhandenen Projekts

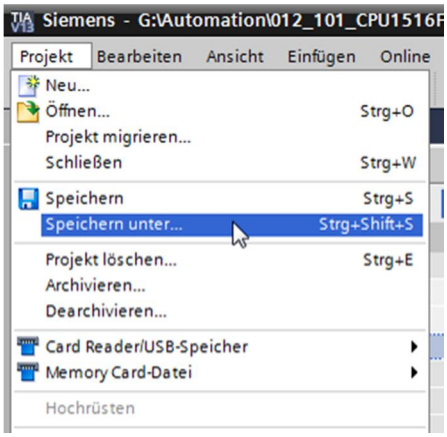
® Bevor wir mit der Programmierung beginnen können, benötigen wir ein Projekt mit einer Hardwarekonfiguration. (z.B. SCE_DE_012-101_Hardwarekonfiguration_S7-1516F_....zap). Zum Dearchivieren eines vorhandenen Projekts müssen Sie aus der Projektansicht heraus unter ® Projekt ® Dearchivieren das jeweilige Archiv aussuchen. Bestätigen Sie Ihre Auswahl anschließend mit öffnen. (® Projekt ® Dearchivieren ® Auswahl eines .zap-Archivs ® öffnen)



® Als Nächstes kann das Zielverzeichnis ausgewählt werden, in welchem das dearchivierte Projekt gespeichert werden soll. Bestätigen Sie Ihre Auswahl mit „OK“. (® Projekt ® Speichern unter ® OK)

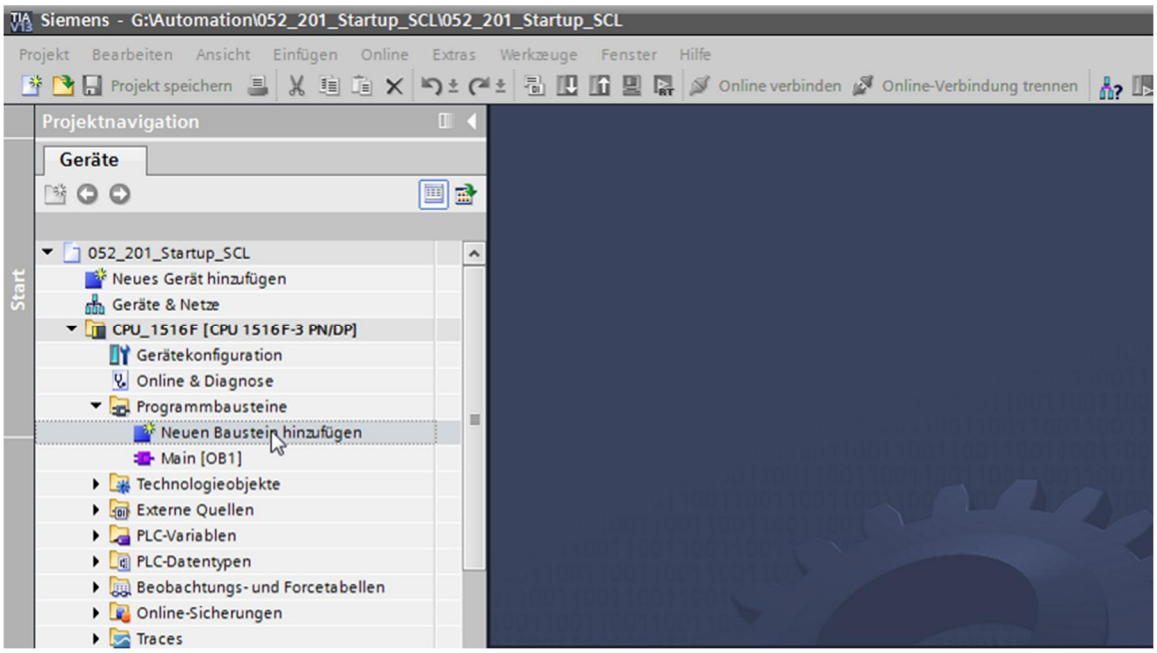
7.2 Speichern des Projektes unter einem neuen Namen

® Das geöffnete Projekt speichern Sie unter dem Namen 052-201_Startup_SCL. (® Projekt ® Speichern unter … ® 052-201_Startup_SCL ® Speichern)

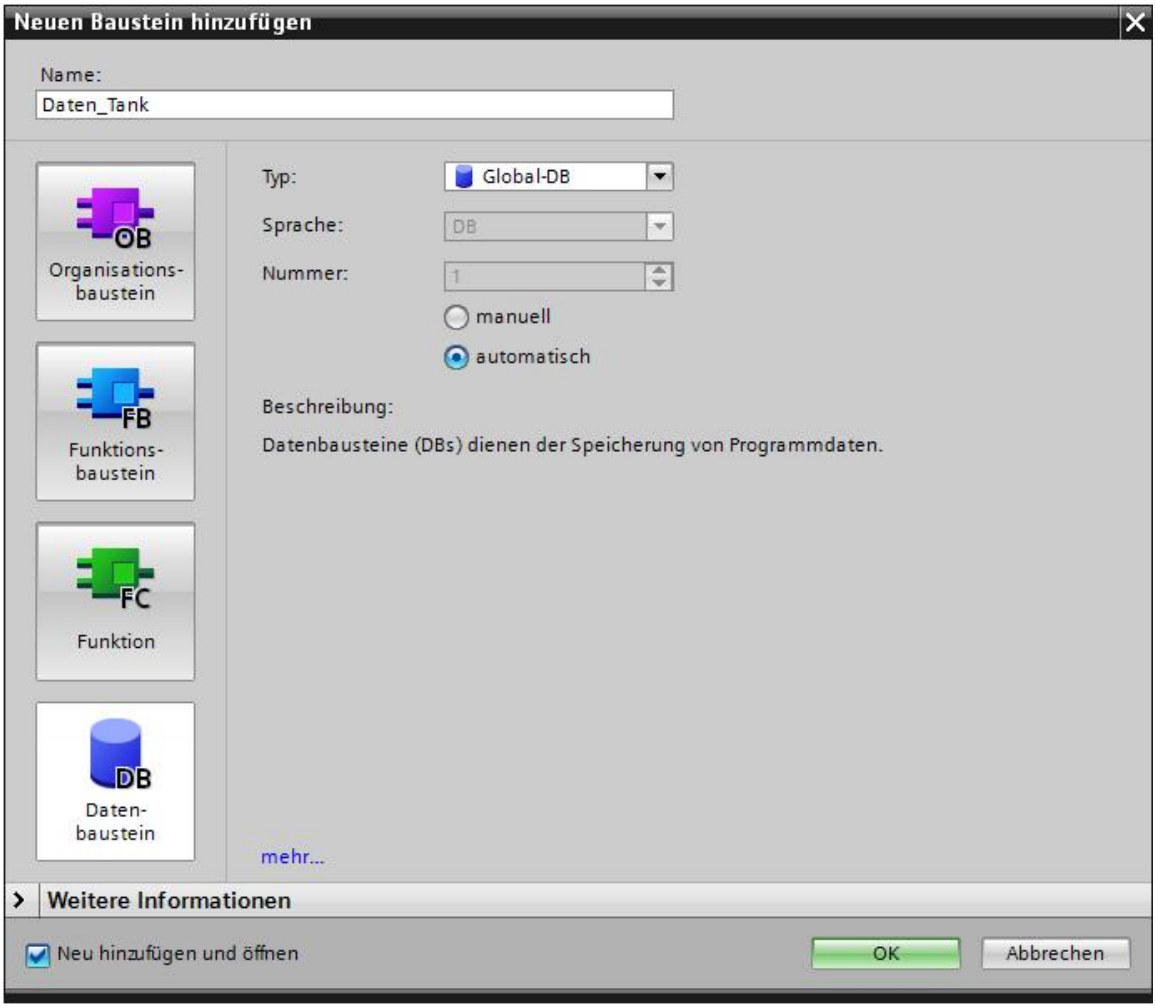


7.3 Anlegen des Datenbausteins „Daten_Tank“

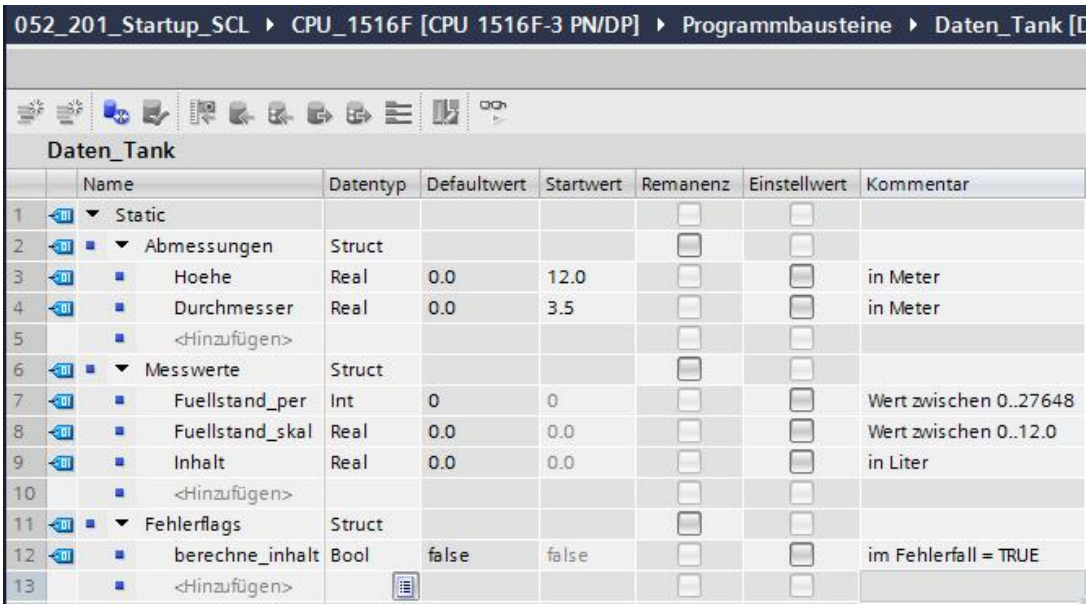
® Navigieren Sie in der Projektansicht zu den ® Programmbausteinen und erstellen Sie einen neuen Baustein, indem Sie auf ® Neuen Baustein hinzufügen doppelklicken.



® Wählen Sie nun einen Datenbaustein an und geben den Namen ein. (®  ® „Daten_Tank“ ® OK)



® Geben Sie nun die unten angegebenen Namen der Variablen mit Datentyp, Startwert und Kommentar ein.

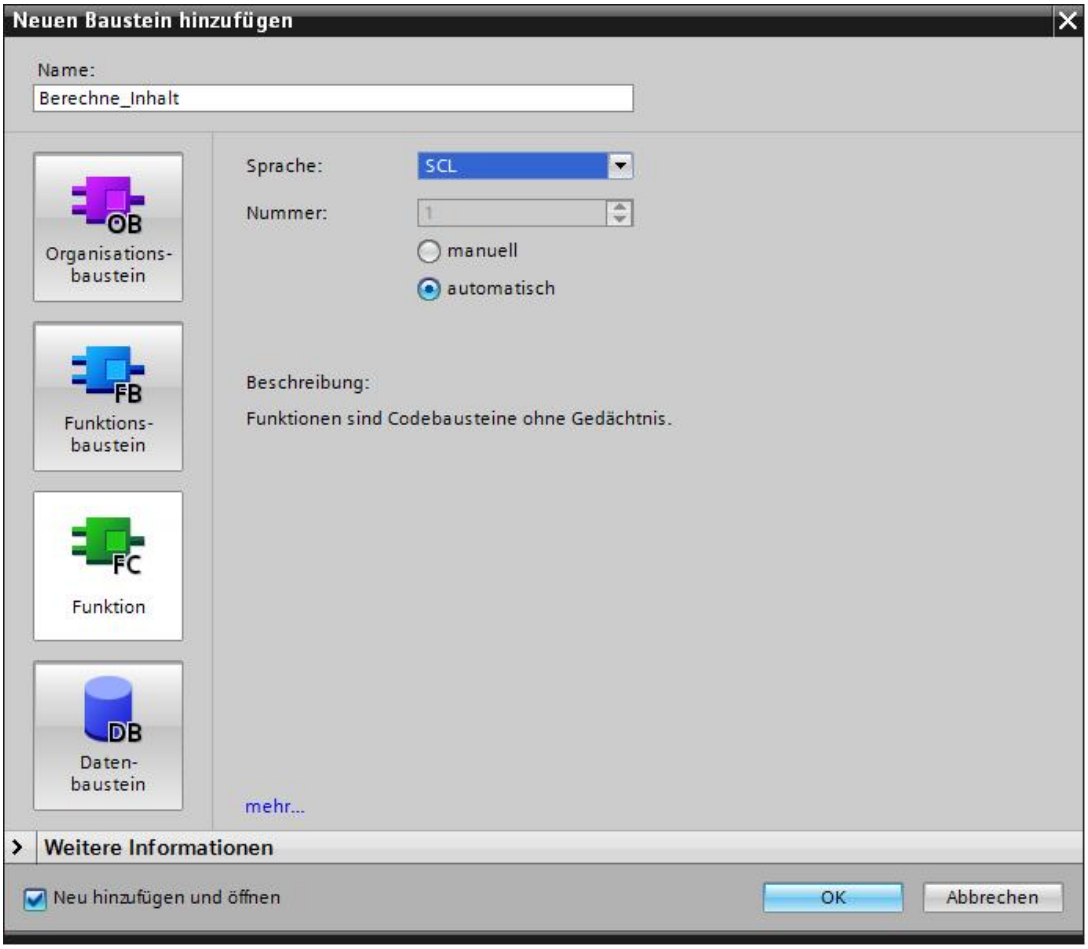


	Name	Datentyp	Defaultwert	Startwert	Remanenz	Einstellwert	Kommentar
1	Static						
2	Abmessungen	Struct					
3	Hoehe	Real	0.0	12.0			in Meter
4	Durchmesser	Real	0.0	3.5			in Meter
5	<Hinzufügen>						
6	Messwerte	Struct					
7	Fuellstand_per	Int	0	0			Wert zwischen 0..27648
8	Fuellstand_skal	Real	0.0	0.0			Wert zwischen 0..12.0
9	Inhalt	Real	0.0	0.0			in Liter
10	<Hinzufügen>						
11	Fehlerflags	Struct					
12	berechne_inhalt	Bool	false	false			im Fehlerfall = TRUE
13	<Hinzufügen>						

7.4 Erstellen der Funktion „Berechne_Inhalt“

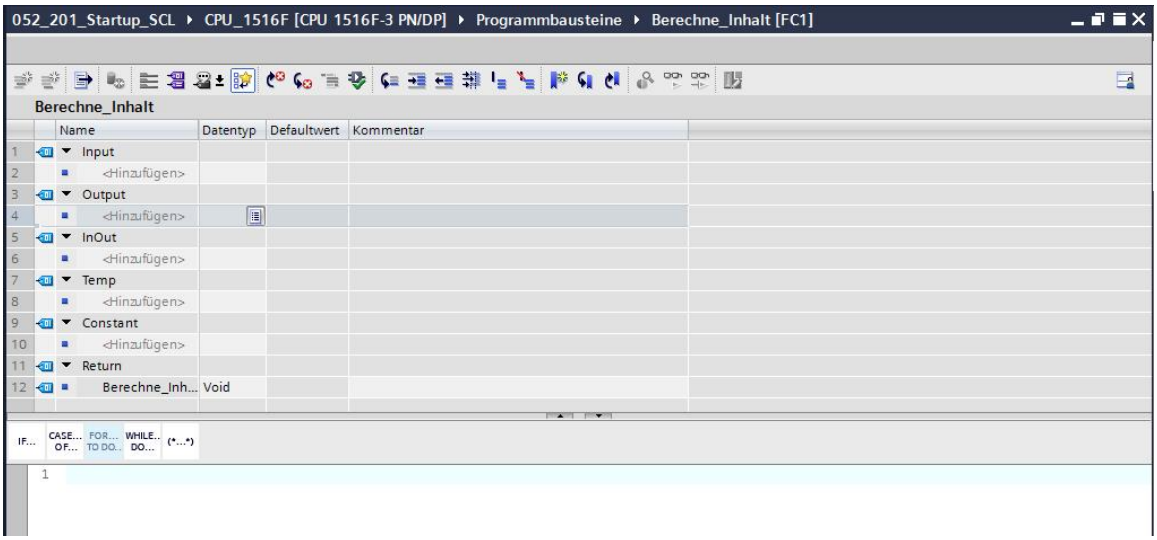
® Jetzt fügen Sie eine Funktion hinzu, geben den Namen ein und wählen die Sprache aus.

(® Neuen Baustein hinzufügen ® ® „Berechne_Inhalt“ ® SCL ® OK)



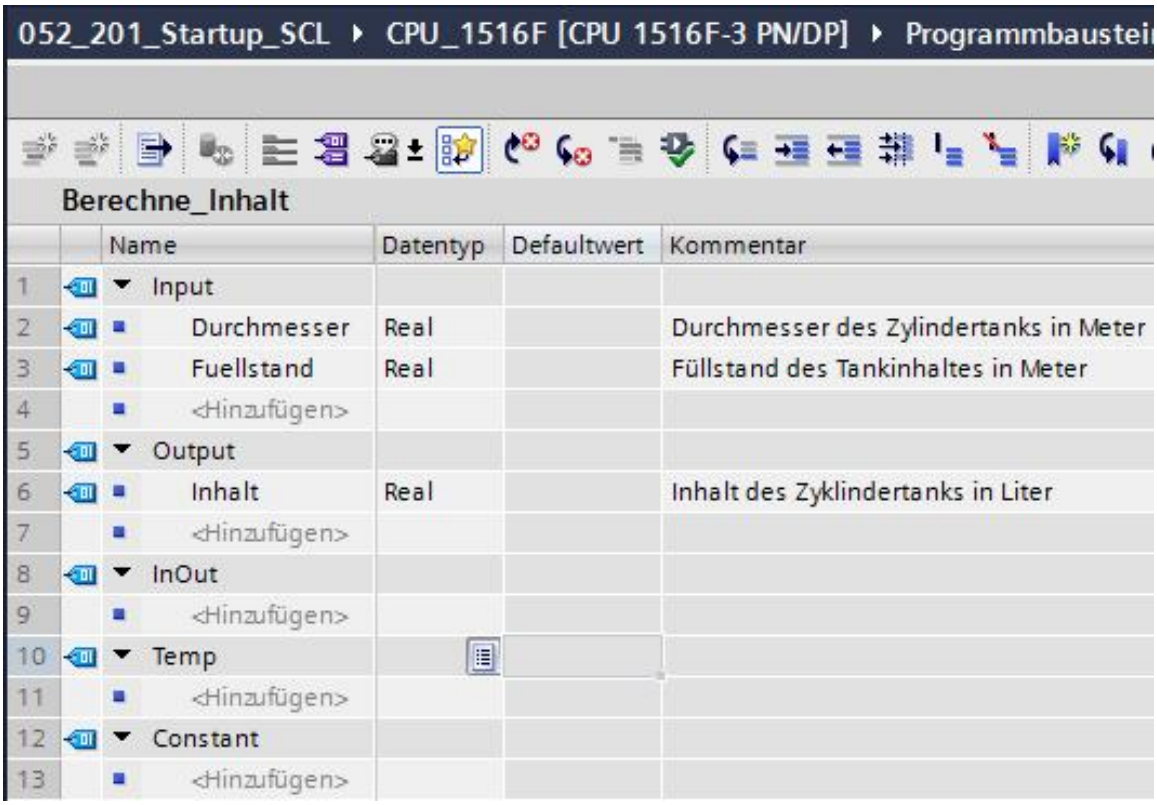
7.5 Schnittstelle der Funktion „Berechne_Inhalt“ festlegen

® Im oberen Abschnitt Ihrer Programmieransicht finden Sie die Schnittstellenbeschreibung Ihrer Funktion.



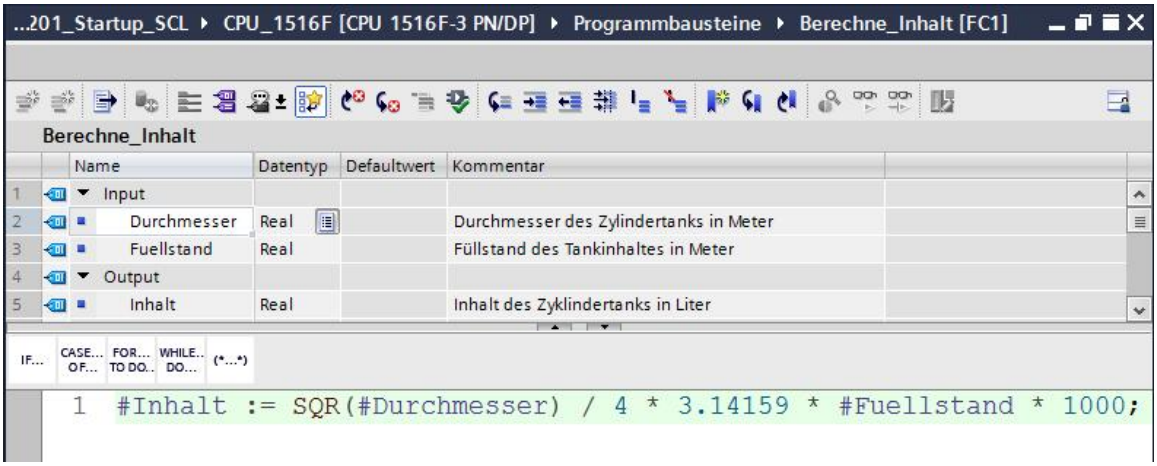
® Legen Sie die folgenden Input- und Outputparameter an.

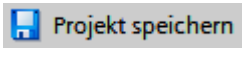
(® Name ® Datentyp ® Kommentar)



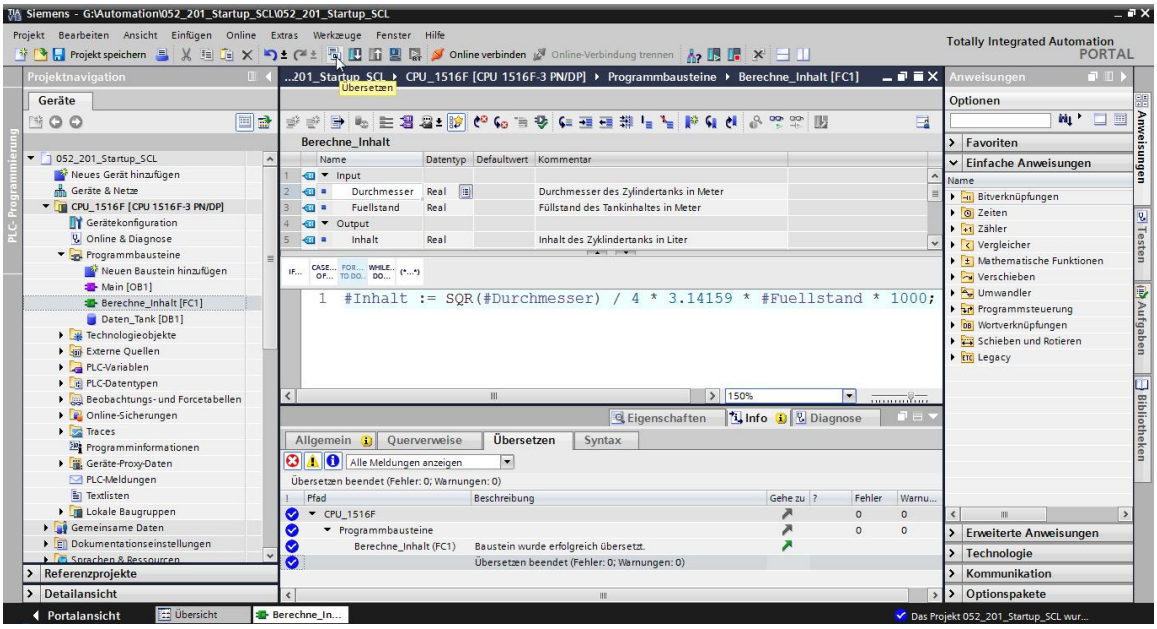
7.6 Programmierung der Funktion „Berechne_Inhalt“

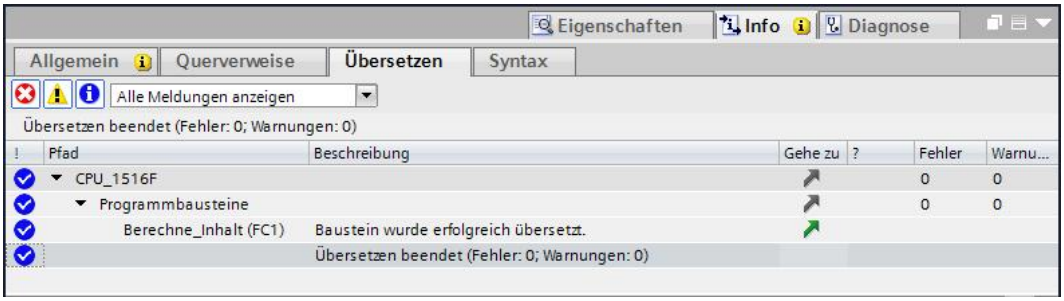
® Geben Sie unten stehendes Programm ein. (® Programm eingeben)



® Übersetzen Sie nun Ihr Programm und überprüfen es auf syntaktische Fehler. Diese werden im Inspektorfenster unterhalb der Programmierung angezeigt. Beheben Sie gegebenenfalls die Fehler und übersetzen anschließend erneut. Speichern Sie danach Ihr Programm. (®  ® Fehler beheben® 

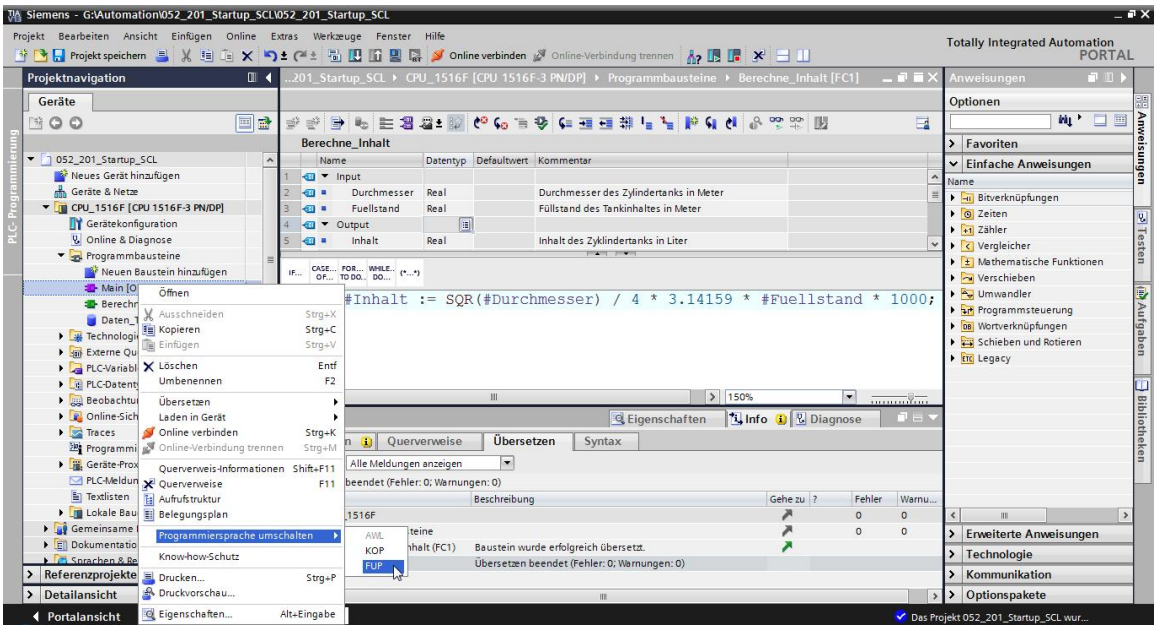
)



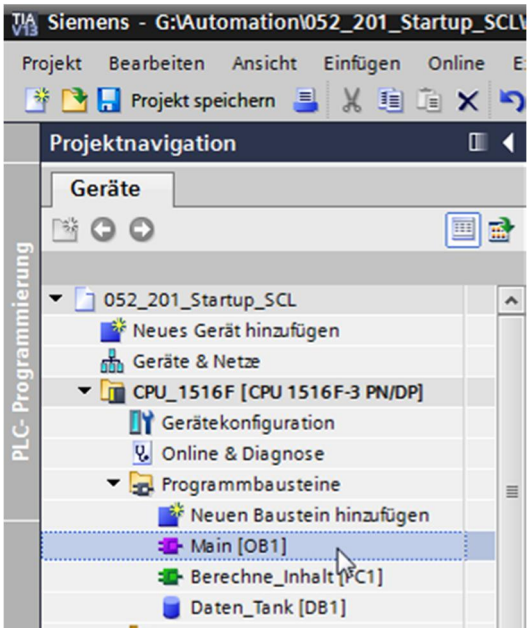


7.7 Programmierung des Organisationsbausteins „Main [OB1]“

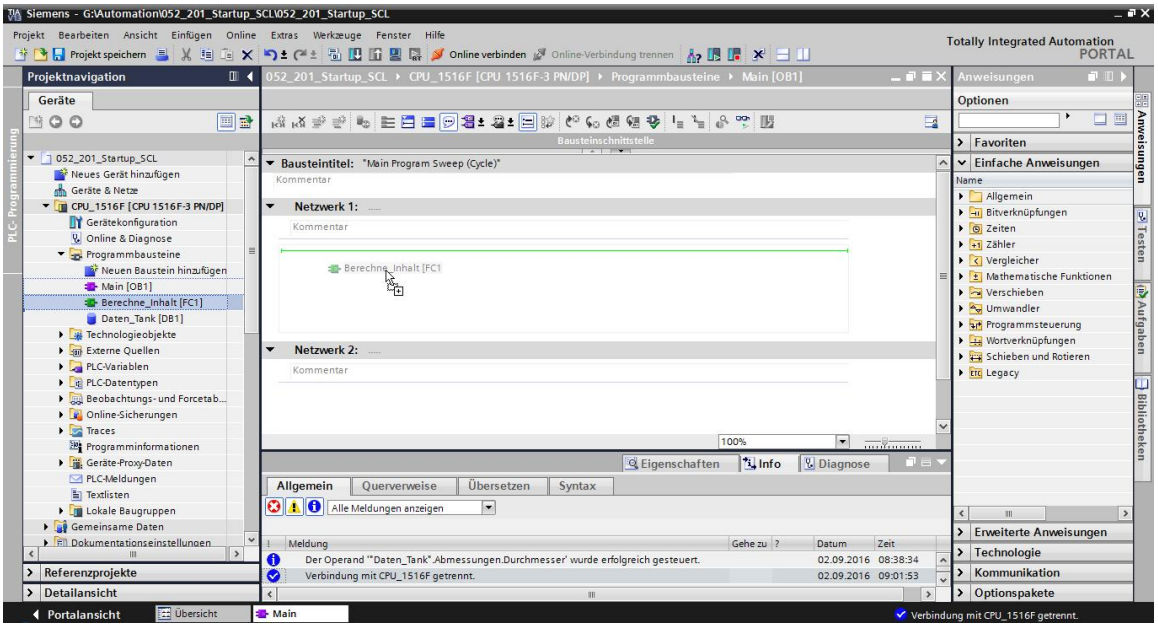
® Vor der Programmierung des Organisationsbausteins „„Main [OB1]““ stellen wir die Programmiersprache auf FUP um. Klicken Sie hierzu vorher mit der linken Maustaste im Ordner „Programmbausteine“ auf „„Main [OB1]““. (® CPU_1516F[CPU 1516F-3 PN/DP] ® Programmbausteine ® Main [OB1] ® Programmiersprache umschalten ® FUP)

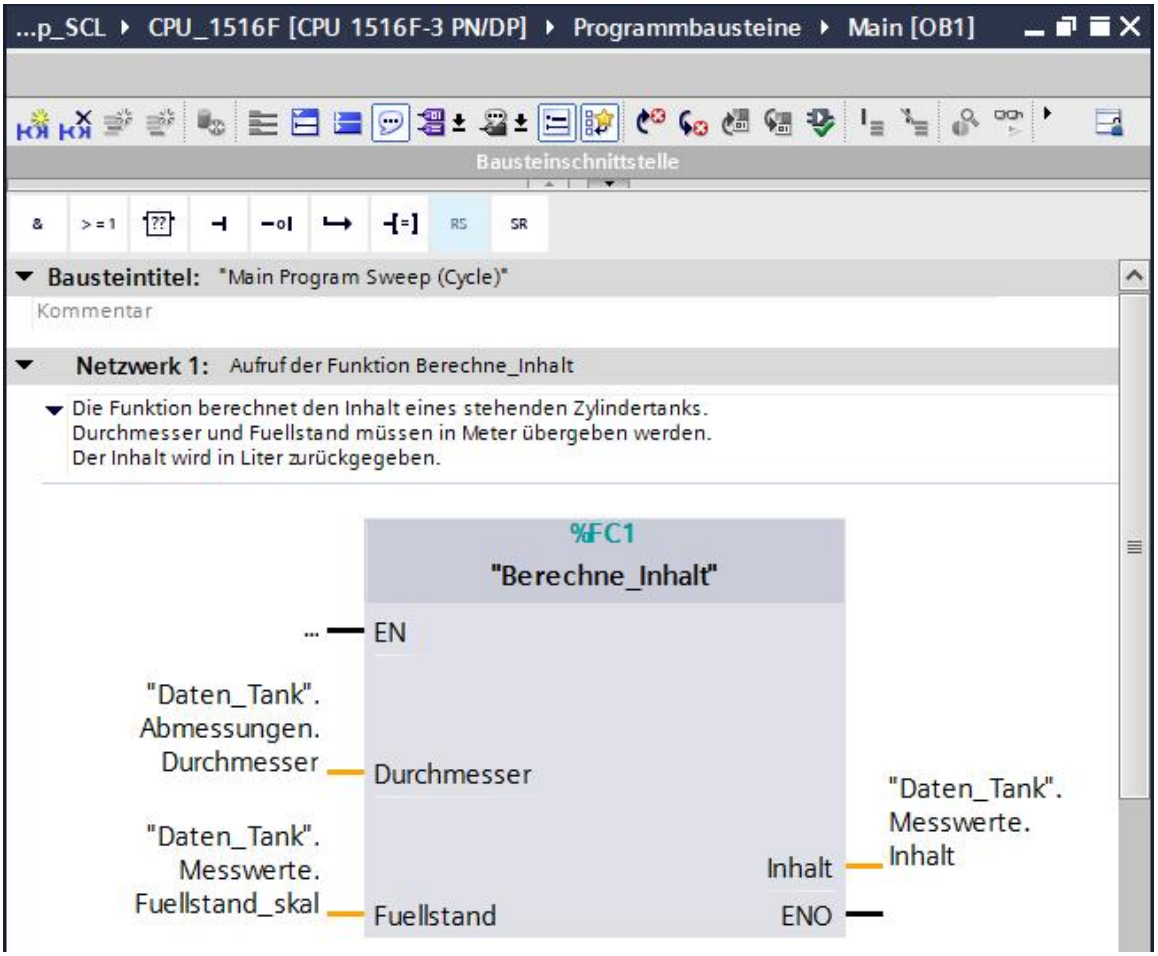


® Öffnen Sie nun den Organisationsbaustein „Main [OB1]“ mit einem Doppelklick.



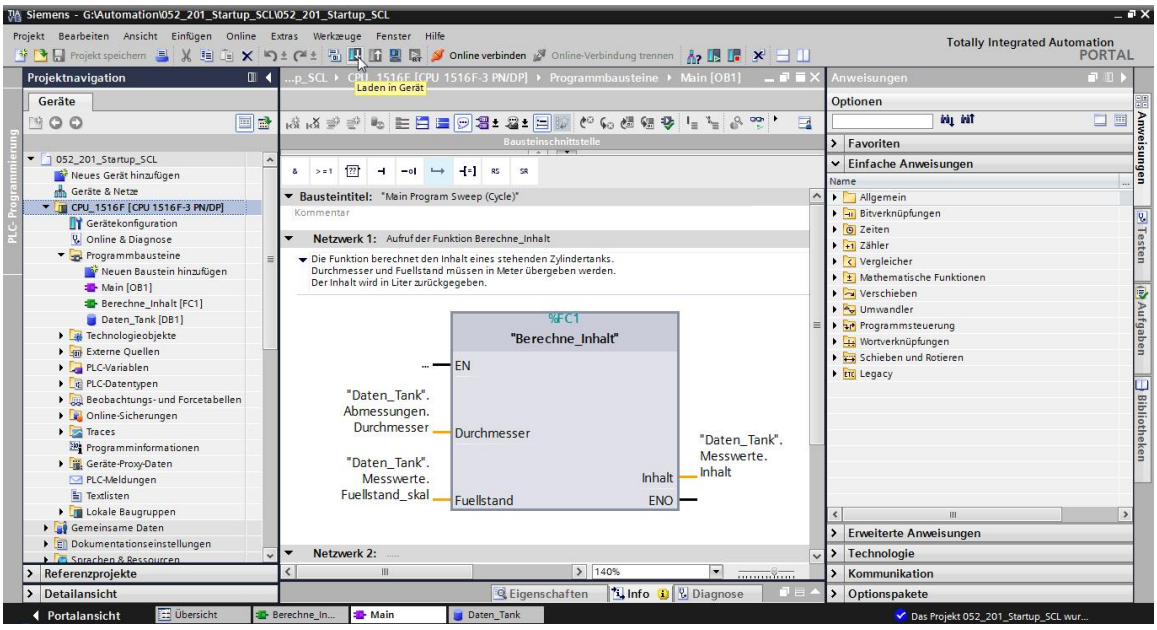
® Rufen Sie die Funktion „Berechne_Inhalt“ im ersten Netzwerk auf. Vergeben Sie Netzwerktitel, Kommentar und beschalten Sie die Parameter. Speichern Sie danach Ihr Projekt ab. (® Aufruf „Berechne_Inhalt“ ® Netzwerktitel vergeben ® Netzwerkkommentar schreiben ® Parameter beschalten ® Projekt speichern)



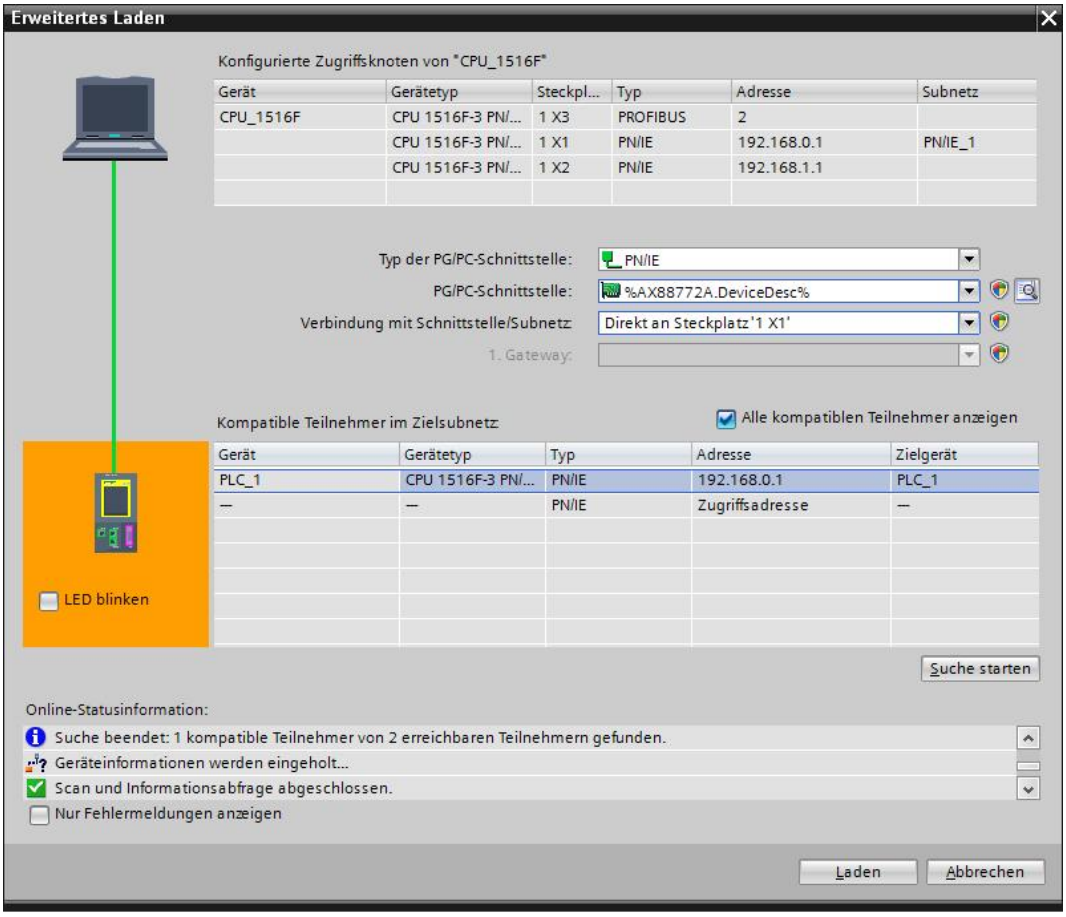


7.8 Programm übersetzen und laden

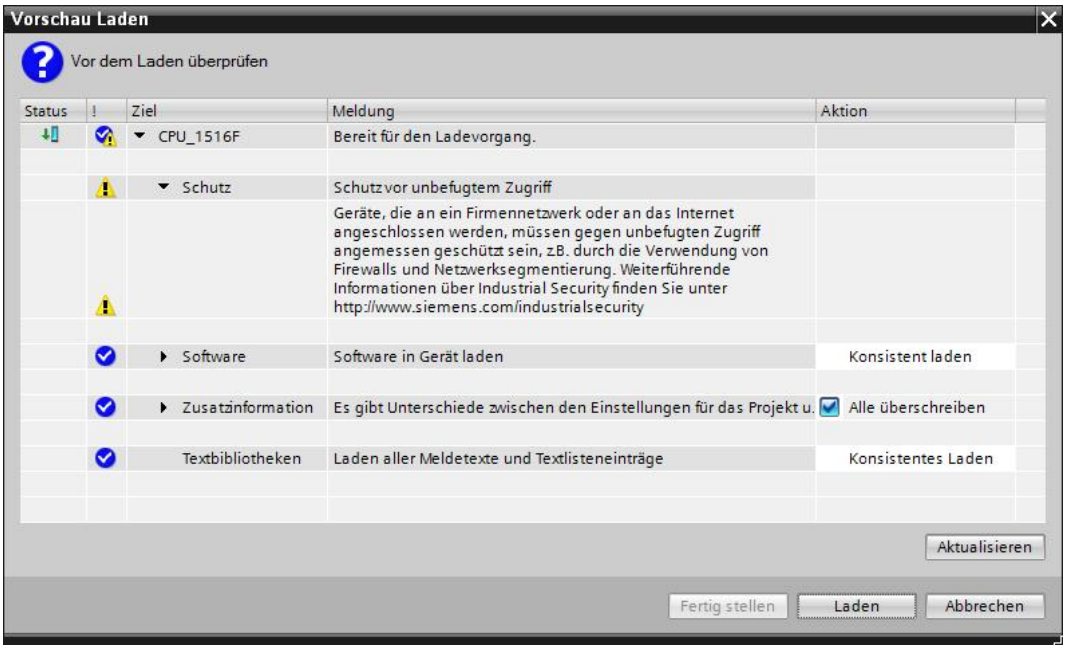
® Klicken Sie auf den Ordner „Programmbausteine“ und übersetzen Sie das gesamte Programm. Nach erfolgreichem Übersetzen laden Sie das Projekt in die Steuerung. (®  ® )



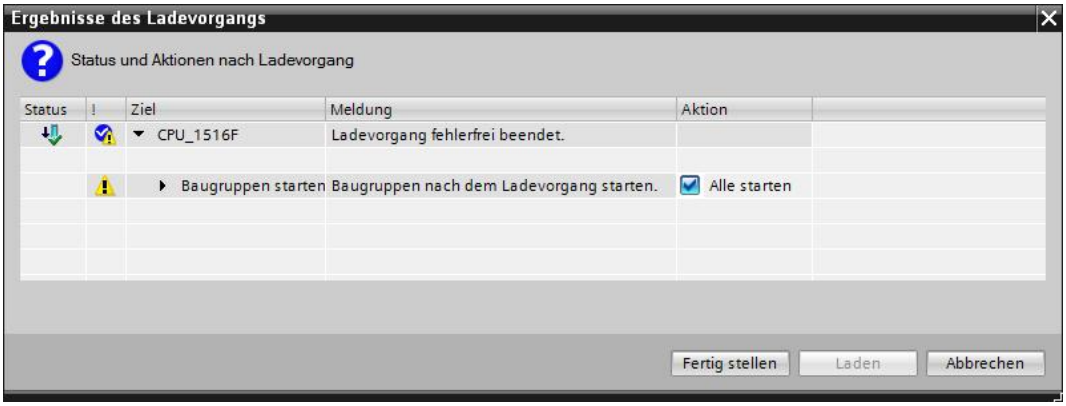
® PG/PC-Schnittstelle auswählen ® Subnetz auswählen ® Suche starten ® Laden



® Evtl. Auswahl treffen ® Laden

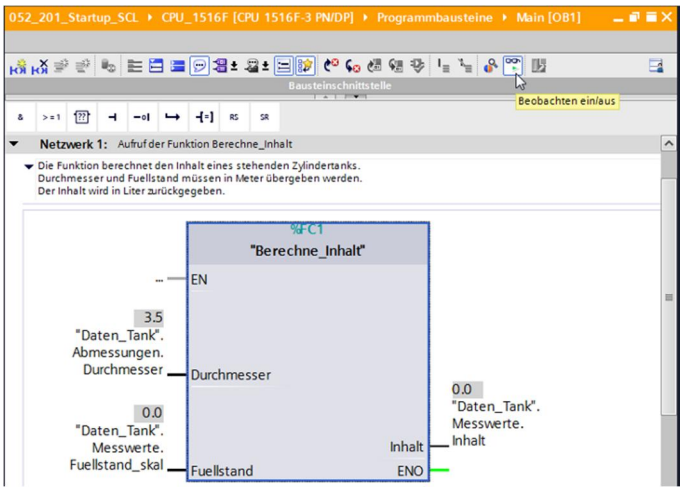


® Fertig stellen

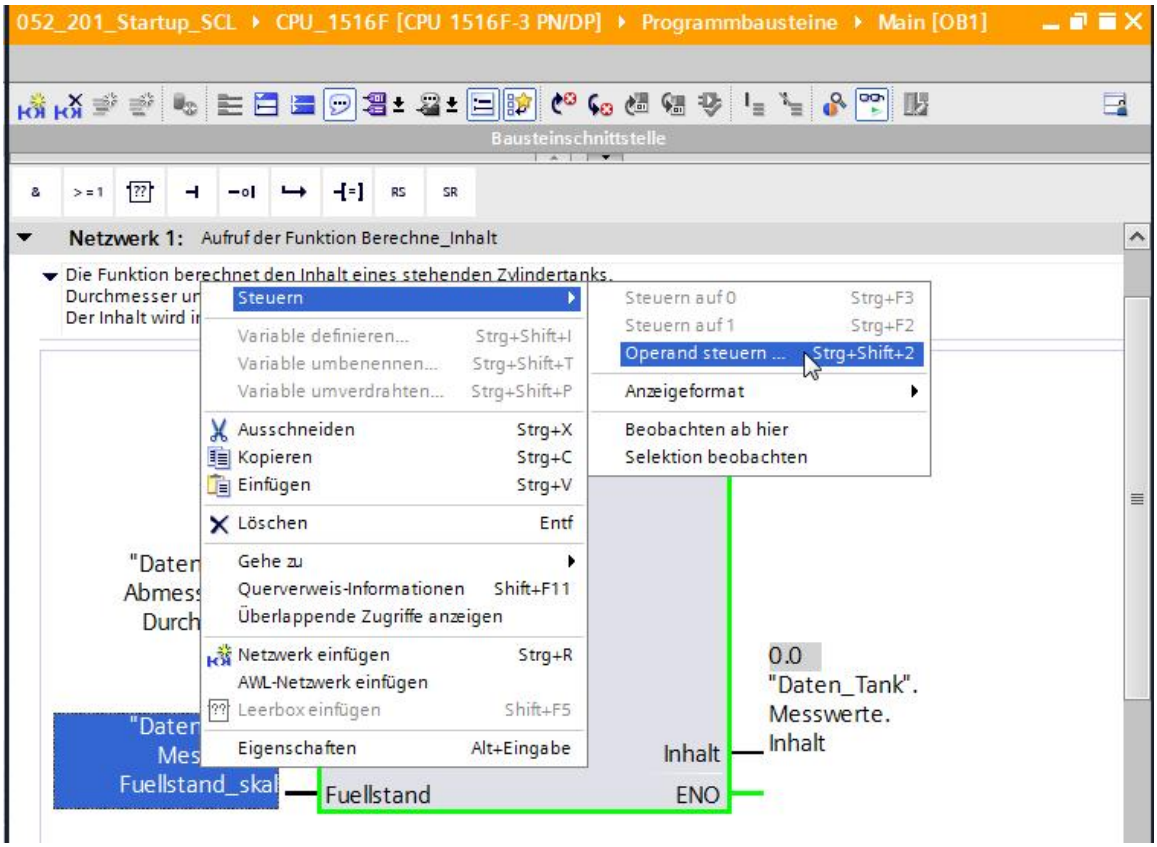


7.9 Organisationsbaustein beobachten und testen

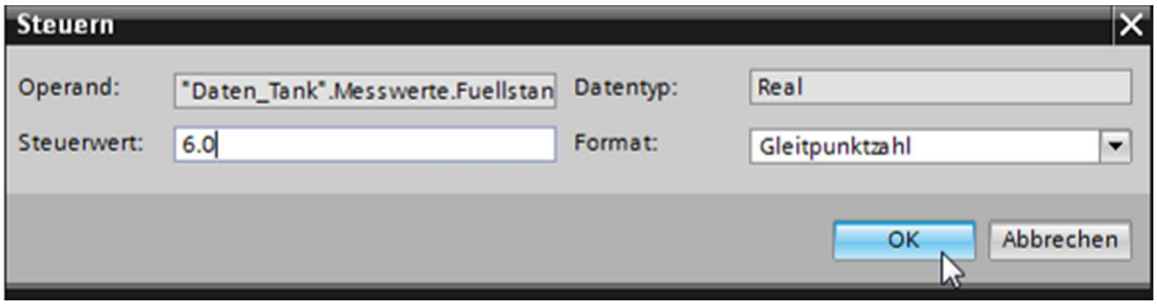
® Klicken Sie im geöffneten OB1 auf das Symbol  , um den Baustein zu beobachten.



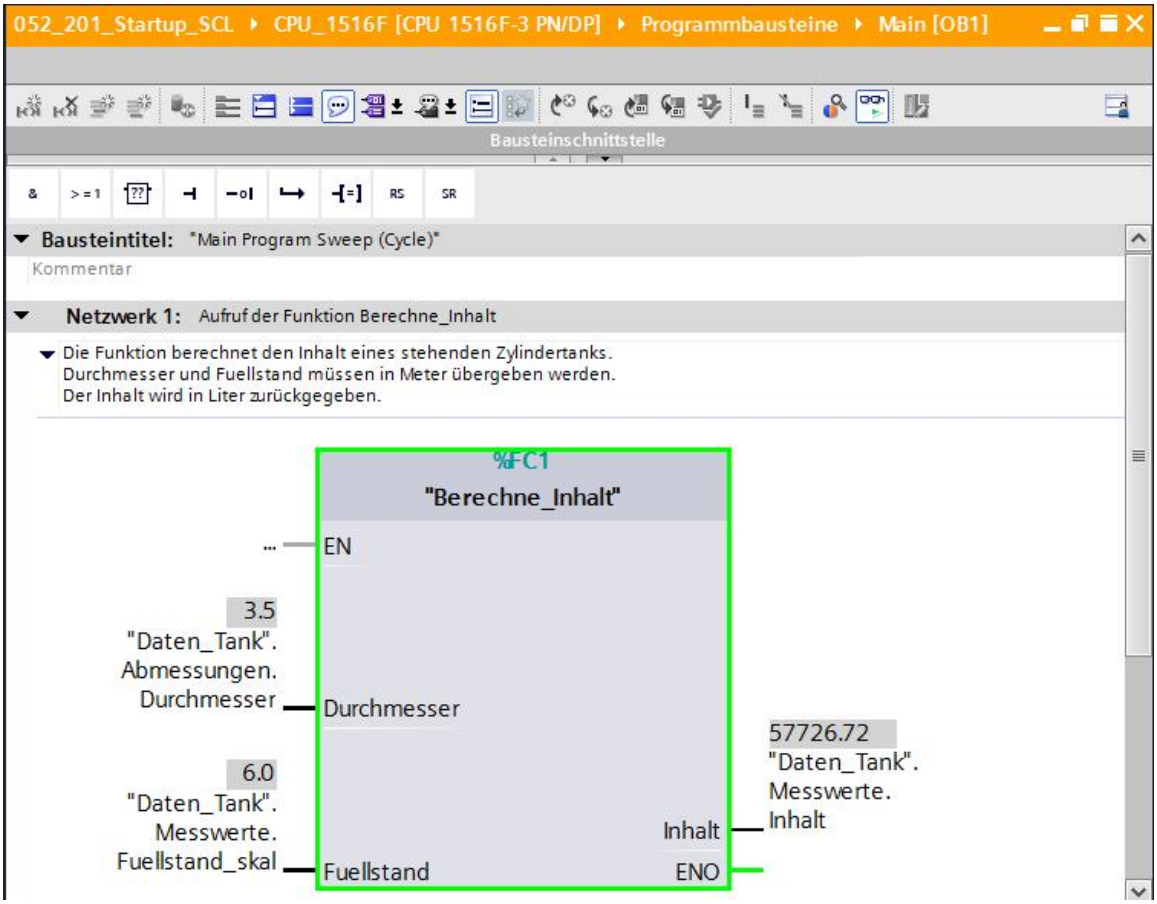
® Testen Sie Ihr Programm, indem Sie einen Wert in die Variable „Fuellstand_skal“ im Datenbaustein schreiben.(® Rechtsklick auf „Fuellstand_skal“ ® Menü „Steuern“ ® Operand steuern)



® Wert 6.0 eintragen ® OK

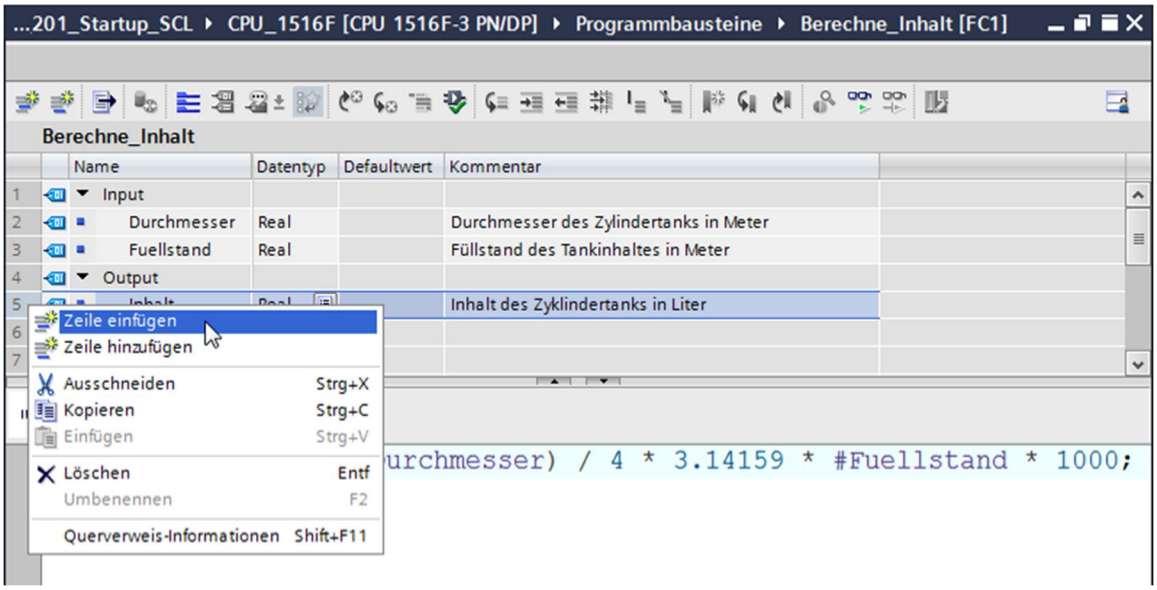


® Überprüfen Sie das Ergebnis auf Richtigkeit.



7.10 Erweiterung der Funktion „Berechne_Inhalt“

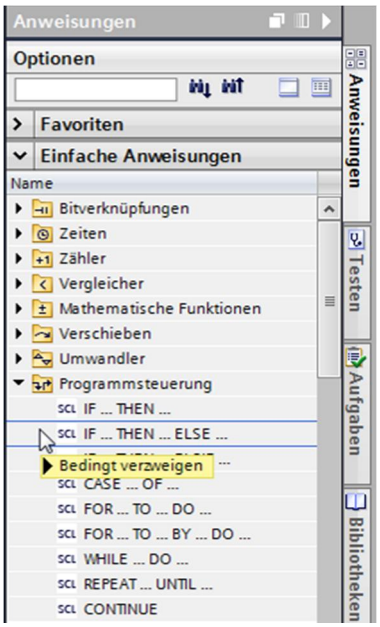
® Öffnen Sie die Funktion „Berechne_Inhalt“ und fügen Sie durch Rechtsklick, auf die Zeile in der Schnittstelle, eine Zeile bei den Outputparametern ein. (® „Berechne_Inhalt“ öffnen ® Rechtsklick auf Zeile 5 ® Zeile einfügen)



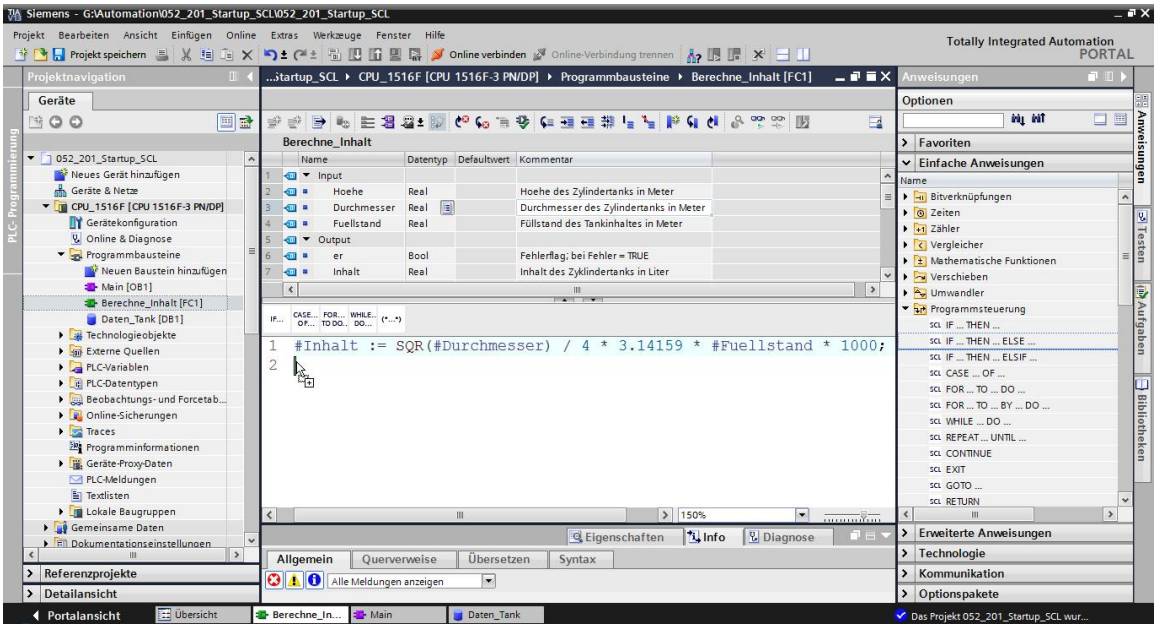
® Tragen Sie den Parameter „er“ mit Datentyp BOOL und Kommentar ein.

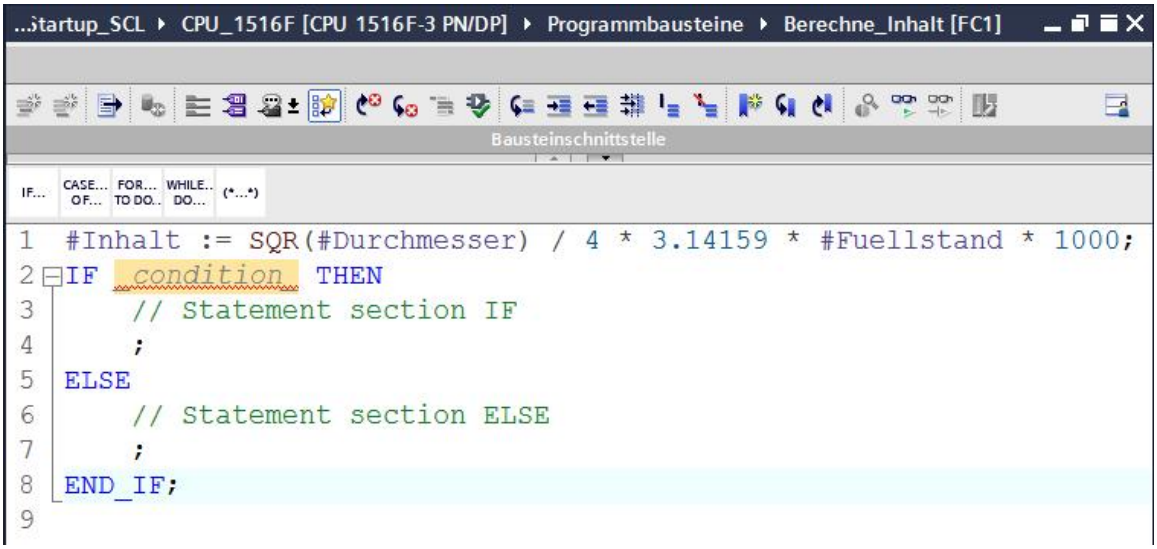
® Nachfolgend fügen Sie auf die gleiche Weise die Variable „Hoehe“ mit Datentyp Real und Kommentar ein.

® Navigieren Sie danach zur Kontrollstruktur „IF…THEN…ELSE“ aus dem Ordner „Programmsteuerung“ der einfachen Anweisungen. (® Anweisungen ® Einfache Anweisungen ® Programmsteuerung ® „IF...THEN…ELSE“)

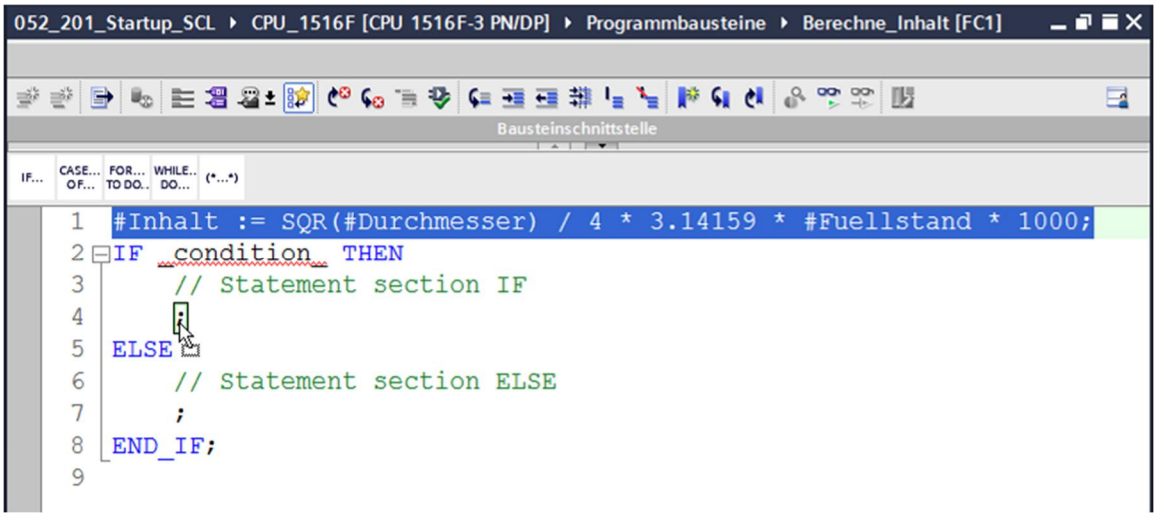


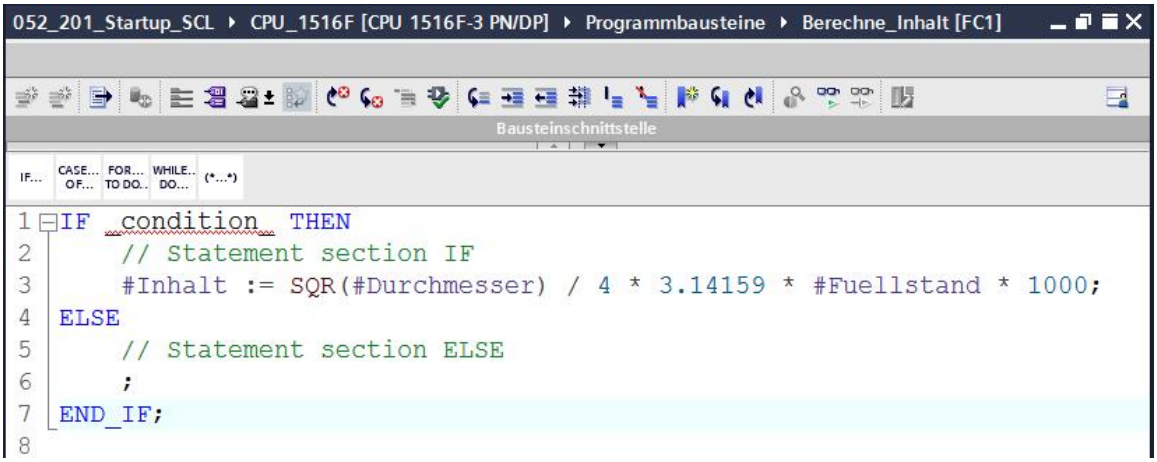
® Ziehen Sie anschließend die Kontrollstruktur „IF...THEN...ELSE“ per Drag & Drop in die zweite Zeile des Programms. (® „IF…THEN…ELSE“ ® Drag & Drop)



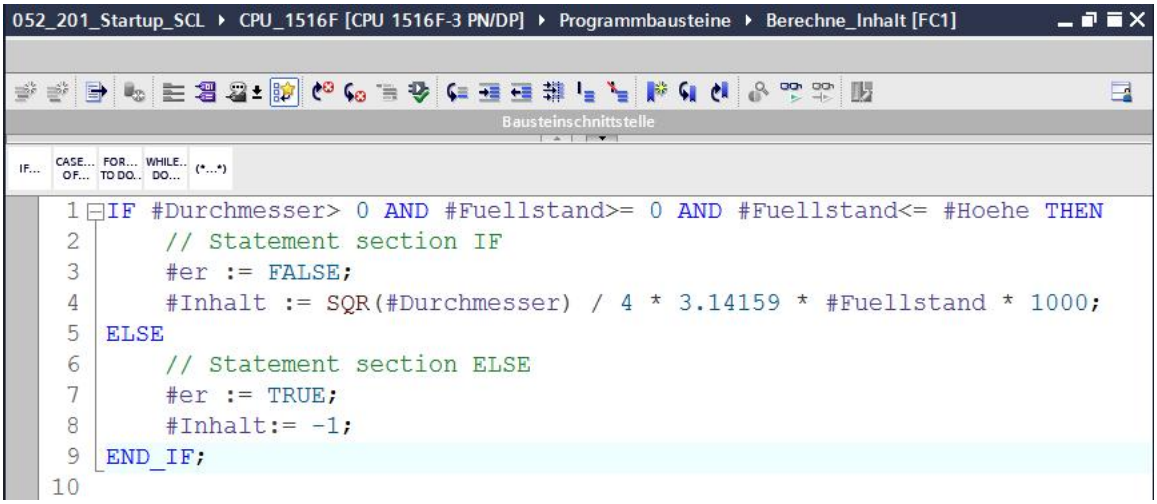


® Markieren Sie die mathematische Formel und ziehen Sie diese per Drag & Drop auf das Semikolon vor dem ELSE. (® markieren ® Drag & Drop)

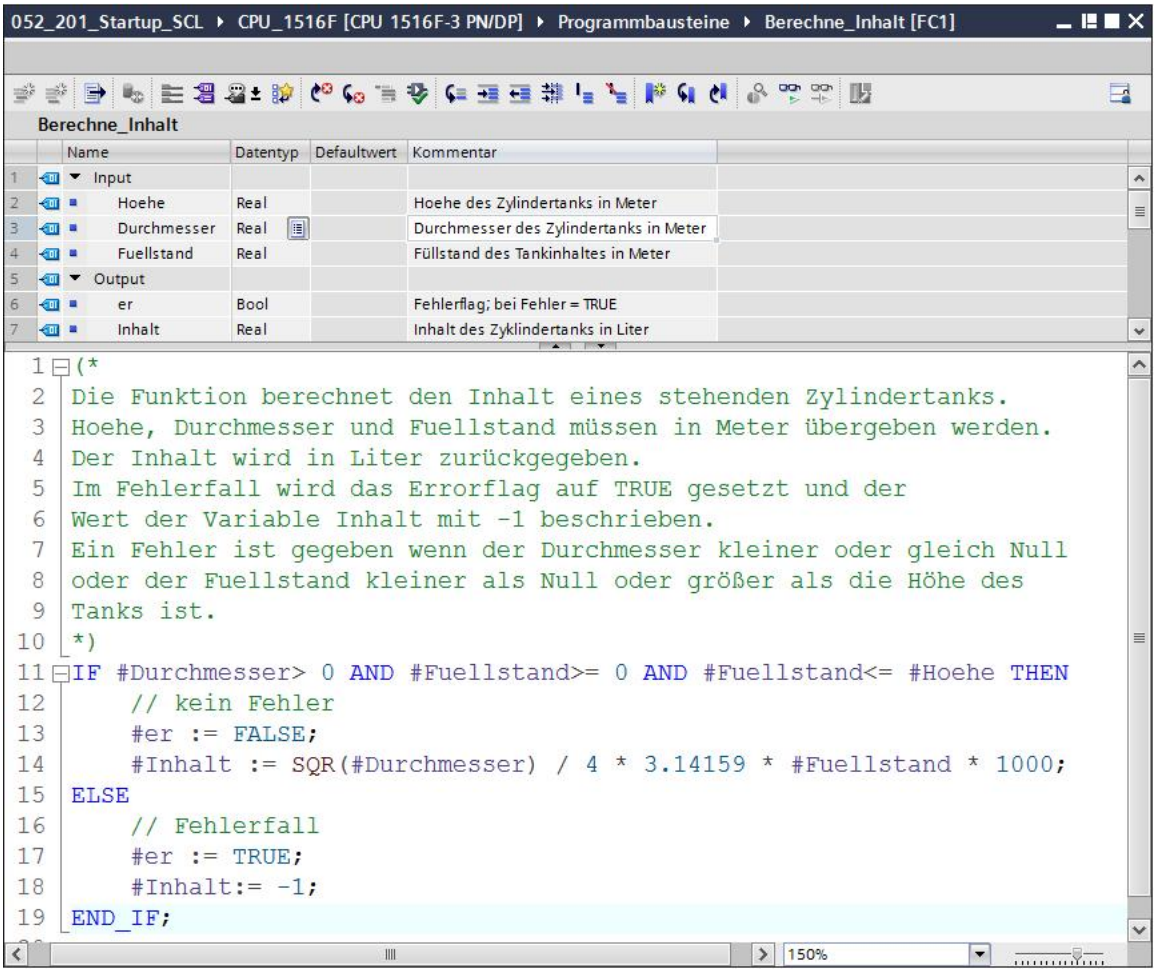




® Vervollständigen Sie die Funktion und überprüfen Ihr Programm durch Übersetzen. (® Programm ergänzen ® )

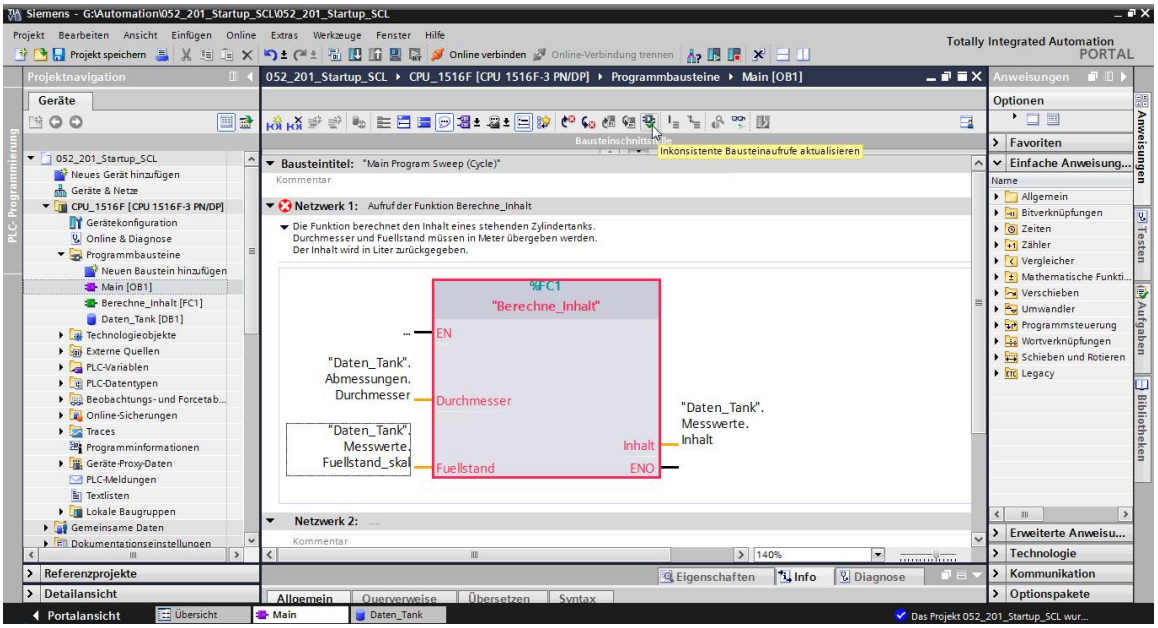


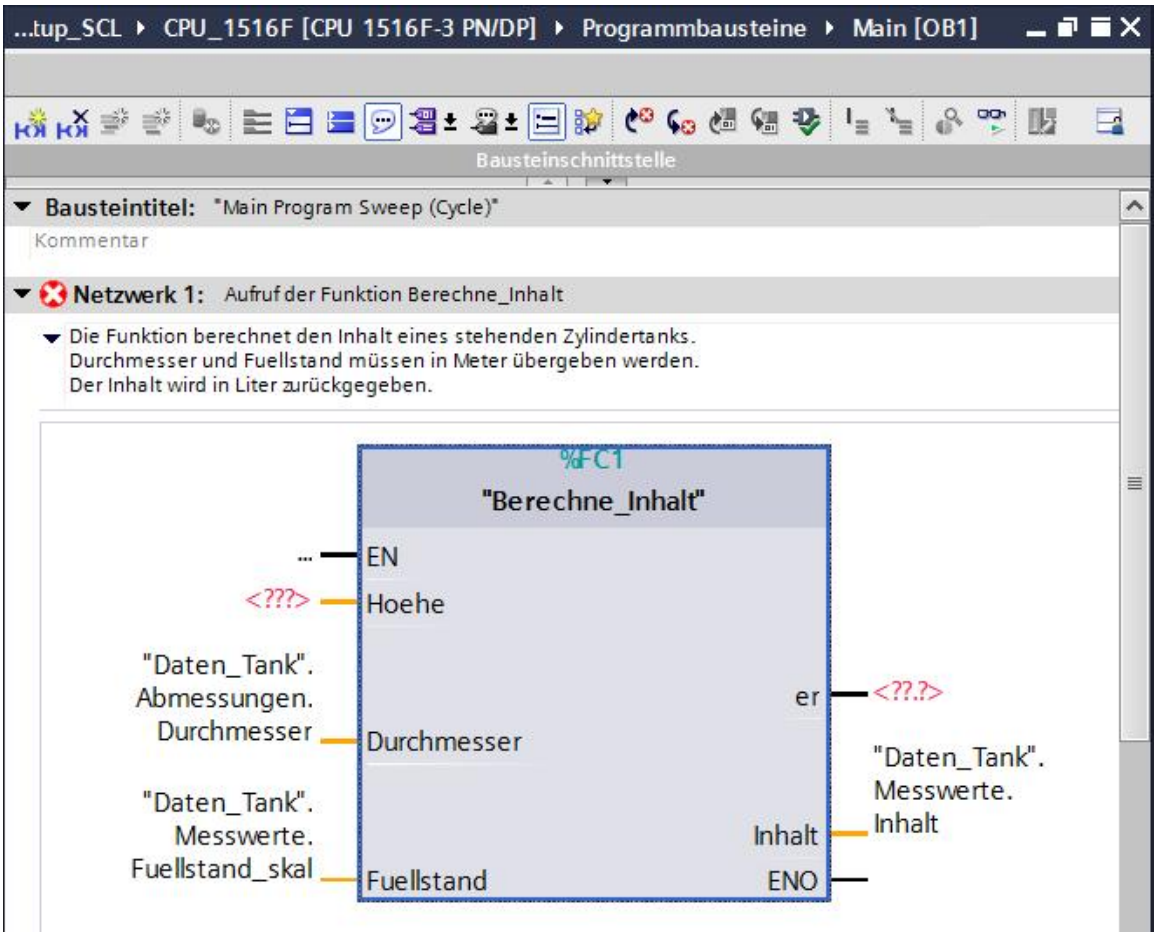
® Kommentare können mit „(**)“ als Blockkommentar und mit „//“ als Zeilenkommentar eingefügt werden. Jetzt können Sie Ihr Programm durch Kommentare ergänzen. (® Blockkommentar ab Zeile 1 einfügen ® Zeilenkommentar in Zeile 12 und 16 einfügen)



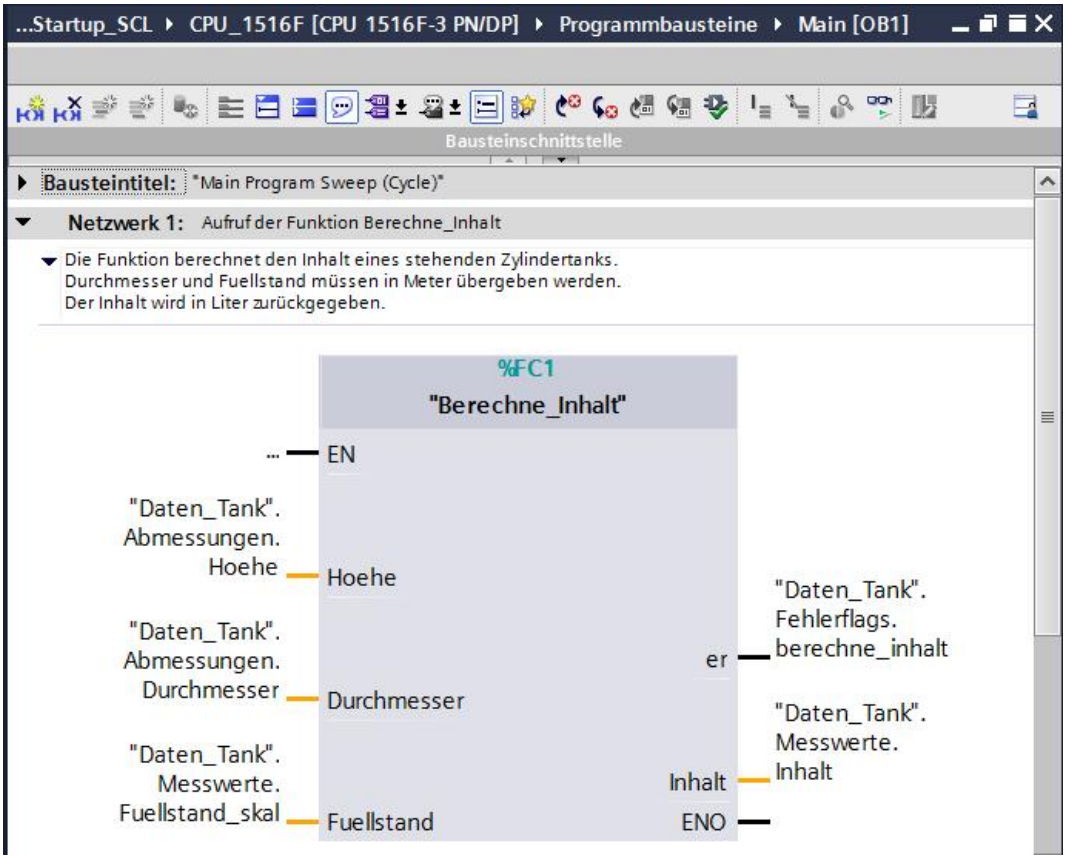
7.11 Organisationsbaustein anpassen

® Öffnen Sie den OB1 und aktualisieren Sie die inkonsistenten Bausteinaufrufe durch klicken auf  . (® OB1 öffnen ® )





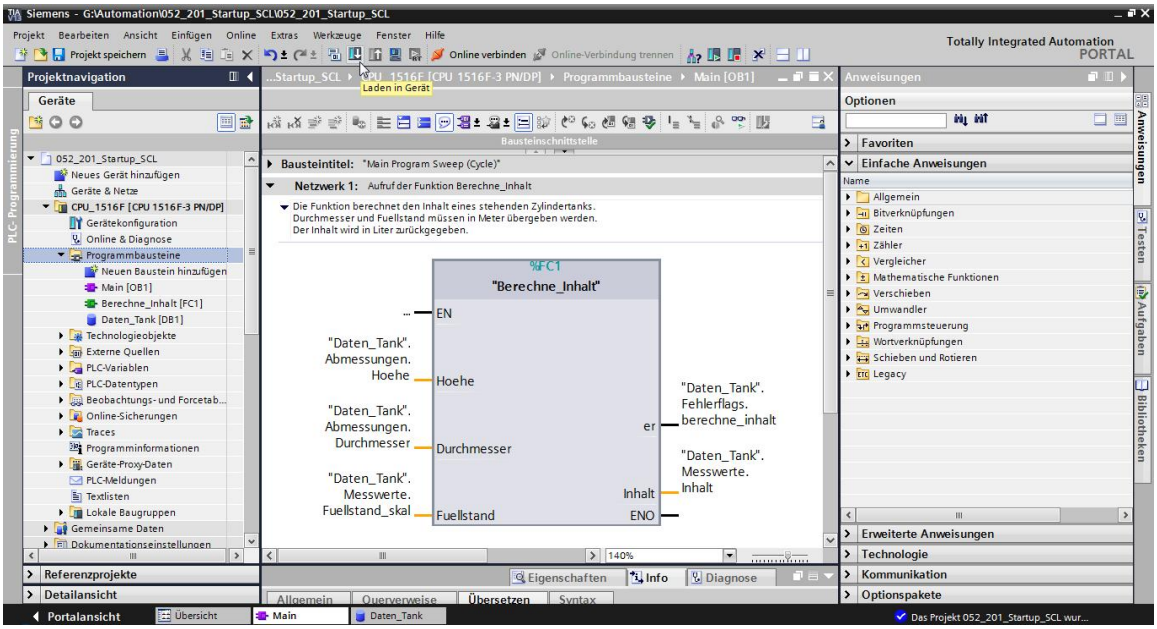
® Ergänzen Sie die Parameter „er“ und „Hoehe“.



7.12 Programm übersetzen und laden

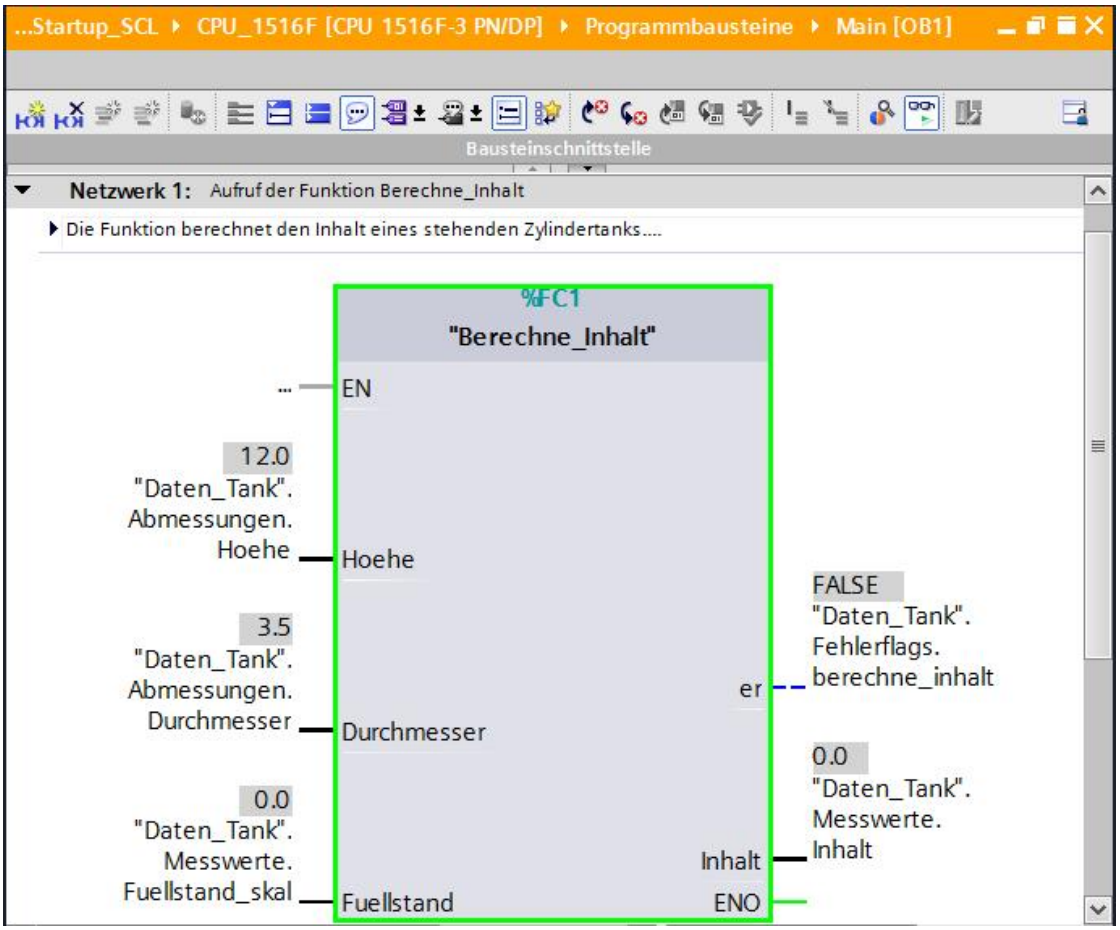
® Klicken Sie auf den Ordner „Programmbausteine“ und übersetzen Sie das gesamte Programm. Nach erfolgreichem Übersetzen laden Sie das Projekt in die Steuerung. Speichern Sie anschließend Ihr Projekt. (® Programmbausteine ®  ®  ® 

)

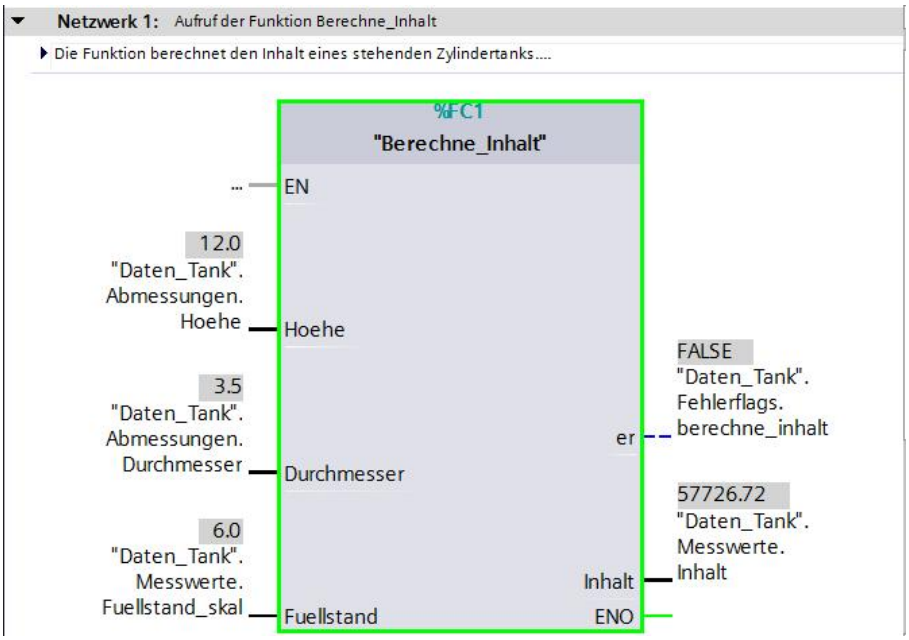


7.13 Organisationsbaustein beobachten und testen

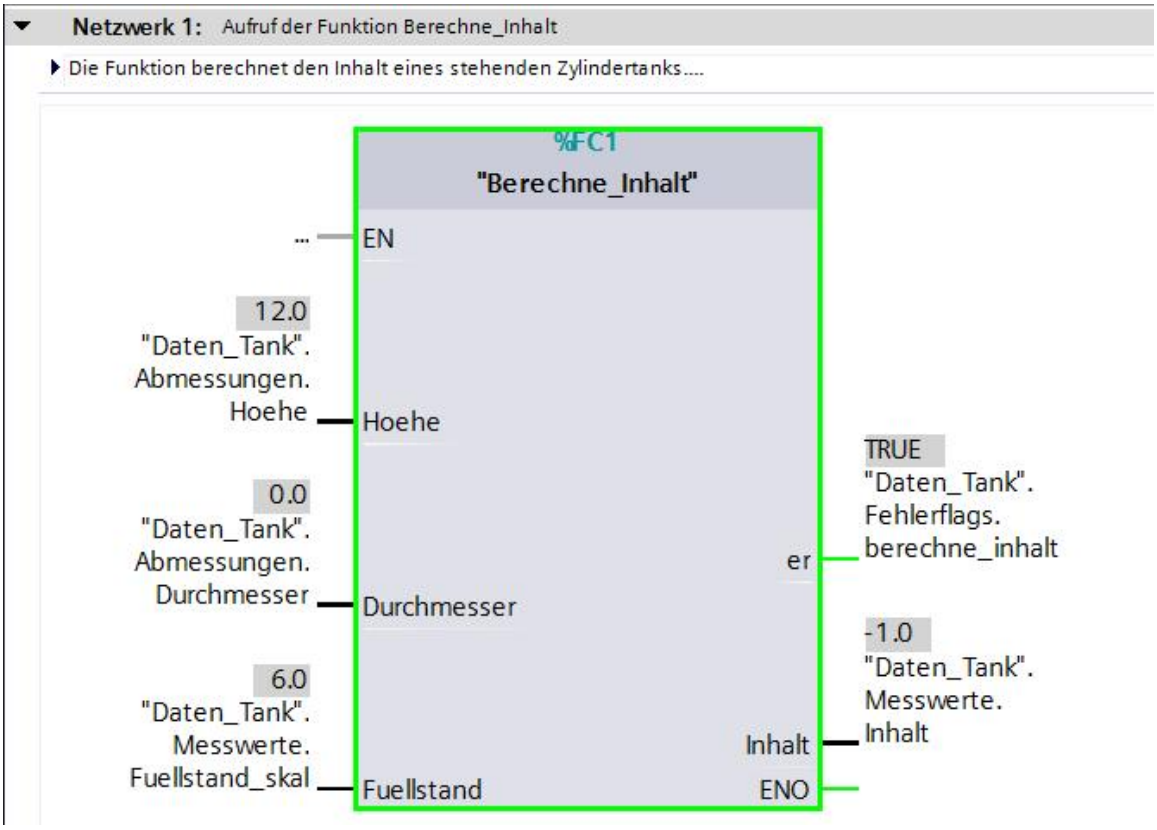
® Klicken Sie im geöffneten OB1 auf das Symbol  um den Baustein zu beobachten.



® Testen Sie Ihr Programm, indem Sie einen Wert in die Variable „Fuellstand_skal“ im Datenbaustein schreiben. (® Rechtsklick auf „Fuellstand_skal“ ® Menü „Steuern“ ® Operand steuern ® Wert 6.0 eingeben ® OK ® Überprüfen)

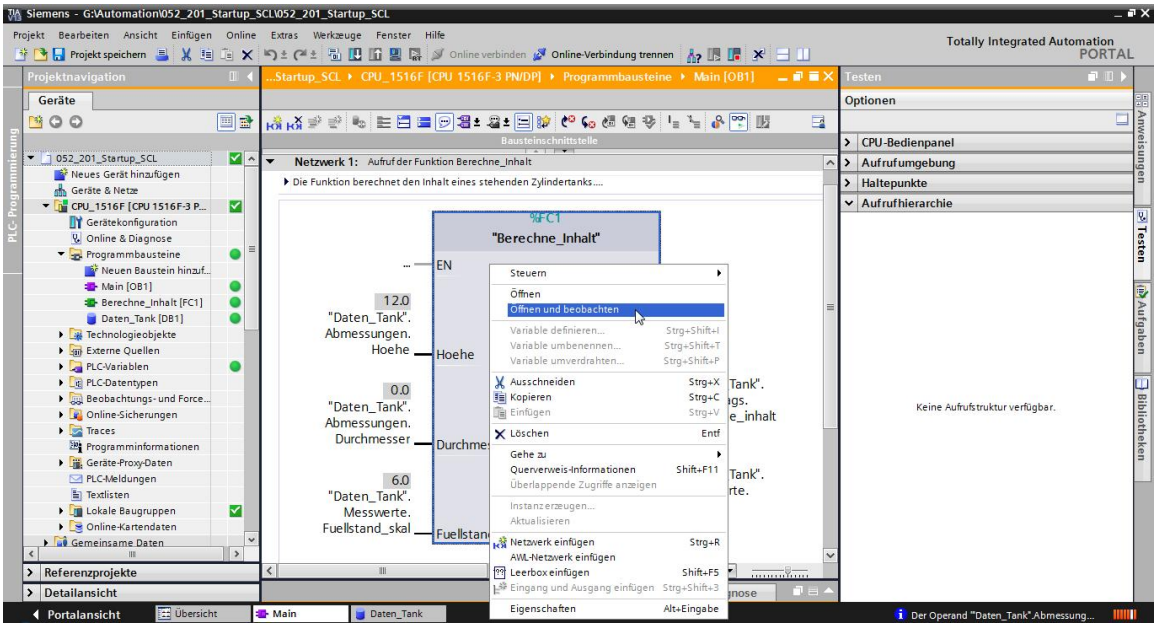


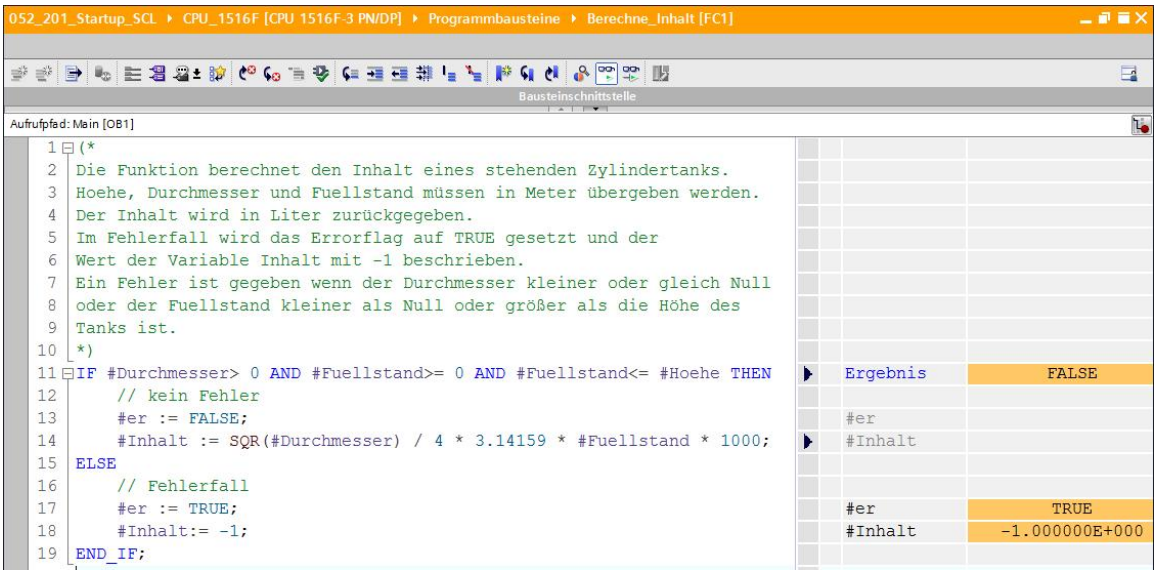
® Testen Sie nachfolgend, ob ein Fehler ausgegeben wird, indem Sie den Durchmesser auf null setzen. (® Rechtsklick auf „Durchmesser“ ® Menü „Steuern“ ® Operand steuern ® Wert 0.0 eingeben ® OK ® überprüfen)



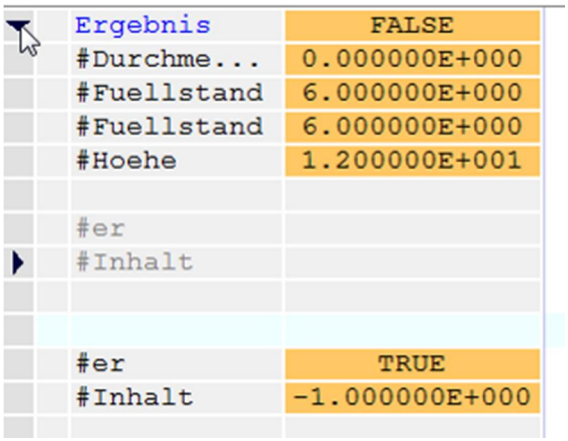
7.14 Funktion „Berechne_Inhalt“ beobachten und testen

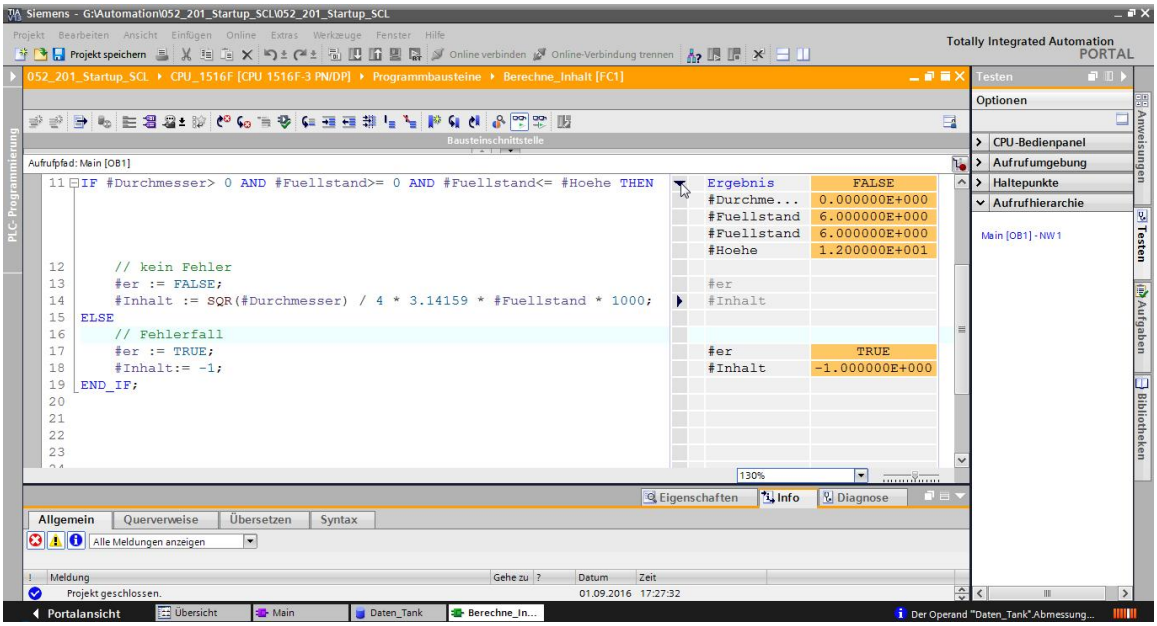
® Öffnen und beobachten Sie schließlich die Funktion „Berechne_Inhalt“, indem Sie mit Rechtsklick auf die Funktion, den Menüpunkt „Öffnen und beobachten“ auswählen. (® Rechtsklick auf Funktion ® Öffnen und beobachten)



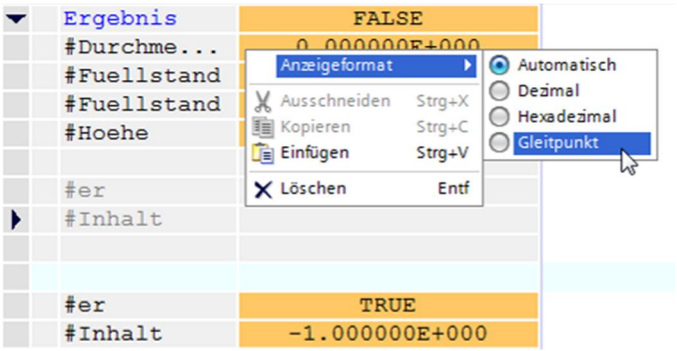


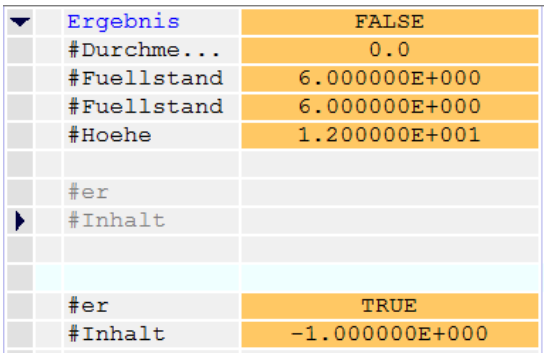
® Sie können die Werte der einzelnen Variablen der IF-Abfrage per Klick auf den schwarzen Pfeil ▼ einblenden. (® ▼)



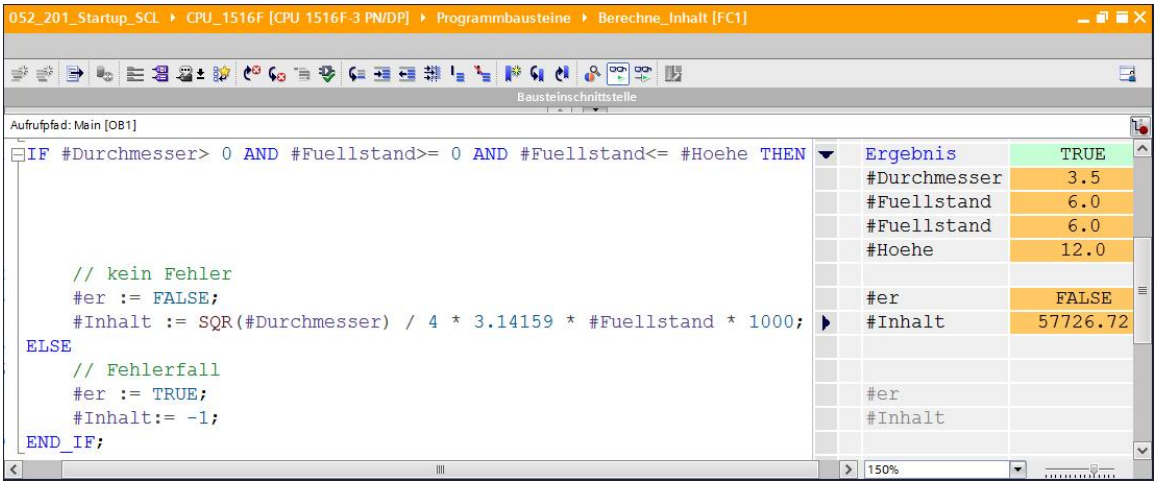


® Das Anzeigeformat kann durch Rechtsklick auf die Variable angepasst werden. (® Rechtsklick auf Variable ® Anzeigeformat ® Gleitpunkt)



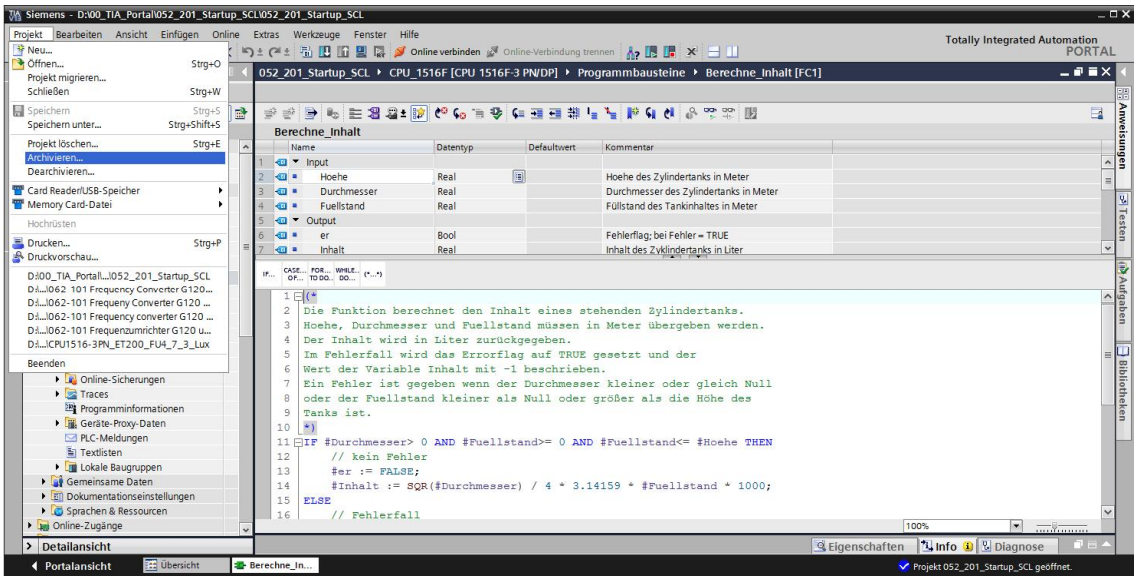


® Testen Sie nun den anderen Zweig der IF-Verzweigung, indem Sie den Durchmesser im OB1 wieder auf 3.5 Meter steuern. (® OB1 öffnen ® Durchmesser auf 3.5 steuern ® Funktion öffnen und beobachten)



7.15 Archivieren des Projektes

® Zum Abschluss soll das komplette Projekt noch archiviert werden. Wählen Sie bitte im Menüpunkt ® ‚Projekt' ® ‚Archivieren …' aus. Öffnen Sie den Ordner, in welchem Sie Ihr Projekt archivieren wollen und speichern Sie es als Dateityp ‚TIA Portal-Projektarchive' ab. (® Projekt ® Archivieren ® TIA Portal-Projektarchive ® SCE_DE_052-201 Startup SCL_S7-1500… ® Speichern)



8. Checkliste

Nr.	Beschreibung	Geprüft
1	Übersetzen erfolgreich und ohne Fehlermeldung	
2	Laden erfolgreich und ohne Fehlermeldung	
3	Operand steuern (Durchmesser = 0.0) Ergebnis Variable Inhalt = -1 Ergebnis Variable „er" = TRUE	
4	Operand steuern (Durchmesser= 3.5 und Fuellstand_skal= 0) Ergebnis Inhalt = 0 Ergebnis Variable „er" = FALSE	
5	Operand steuern (Fuellstand_skal= 6.0) Ergebnis Inhalt = 57726.72 Ergebnis Variable „er" = FALSE	
6	Operand steuern (Fuellstand_skal = 12.0) Ergebnis Inhalt = 115453.4 Ergebnis Variable „er" = FALSE	
7	Operand steuern (Fuellstand_skal = 14.0) Ergebnis Inhalt = -1 Ergebnis Variable „er" = TRUE	
8	Projekt erfolgreich archiviert	

9. Übung

9.1 Aufgabenstellung – Übung

In dieser Übung wird eine Funktion „Skalieren“ programmiert. Das Programm soll allgemeingültig für jegliche positiven Analogwerte anwendbar sein. In unserer Beispielaufgabe „Tank“ wird der Füllstand über einen Analogsensor eingelesen und mittels dieser Funktion skaliert im Datenbaustein abgelegt.

Im Fehlerfall soll der Baustein das Errorflag „er“ auf TRUE und als Ergebnis den Parameter „Analog_skal“ auf null setzen. Ein Fehlerfall besteht, wenn der Parameter „mx“ kleiner oder gleich „mn“ ist.

Die Funktion muss folgende Parameter beinhalten.

Input	**Datentyp**	**Kommentar**
Analog_per	INT	Analogwert von der Peripherie zwischen 0..27648
mx	REAL	Maximum des neuen Maßstabs
mn	REAL	Minimum des neuen Maßstabs
Output		
er	BOOL	Fehlerflag, kein Fehler = 0, Fehler = 1
Analog_skal	REAL	Analogwert skaliert zwischen mn..mx Im Fehlerfall = 0

Zur Lösung der Aufgabe wird folgende Formel verwendet:

$$\#\,\text{Analog\_skal} = \frac{\#\,\text{Analog\_per}}{27648} \cdot (\#\,\text{mx} - \#\,\text{mn}) + \#\,\text{mn}$$

Für diese Übungsaufgabe ist ein Analogsignal notwendig. Der hierzu verwendete Operand muss in die PLC-Variablentabelle eingetragen werden.

Name	**Datentyp**	**Adresse**	**Kommentar**
B1	INT	%EW64	Füllstand zwischen 0..27648

9.2 Planung

Planen Sie nun selbstständig die Umsetzung der Aufgabenstellung!

9.3 Checkliste – Übung

Nr.	Beschreibung	Geprüft
1	Operand in PLC-Variablentabelle eingefügt	
2	Funktion FC: „Skalieren“ erstellt	
3	Schnittstelle definiert	
4	Funktion programmiert	
5	Funktion „Skalieren“ ins Netzwerk 1 des OB1 eingefügt	
6	Eingangsvariablen verschaltet	
7	Ausgangsvariablen verschaltet	
8	Übersetzen erfolgreich und ohne Fehlermeldung	
9	Laden erfolgreich und ohne Fehlermeldung	
10	Analogwert für Füllstand auf null gesetzt Ergebnis Fuellstand_skal = 0 Ergebnis er = FALSE	
11	Analogwert für Füllstand auf 27648 gesetzt Ergebnis Fuellstand_skal = 12.0 Ergebnis er = FALSE	
12	Analogwert für Füllstand auf 13824 Ergebnis Fuellstand_skal = 6.0 Ergebnis er = FALSE	
13	Operand steuern (mx = 0.0) Ergebnis Fuellstand_skal = 0 Ergebnis Variable er = TRUE	
14	Projekt erfolgreich archiviert	

10. Weiterführende Information

Zur Einarbeitung bzw. Vertiefung finden Sie als Orientierungshilfe weiterführende Informationen, wie z.B.: Getting Started, Videos, Tutorials, Apps, Handbücher, Programmierleitfaden und Trial Software/Firmware, unter nachfolgendem Link:

www.siemens.de/sce/s7-1500