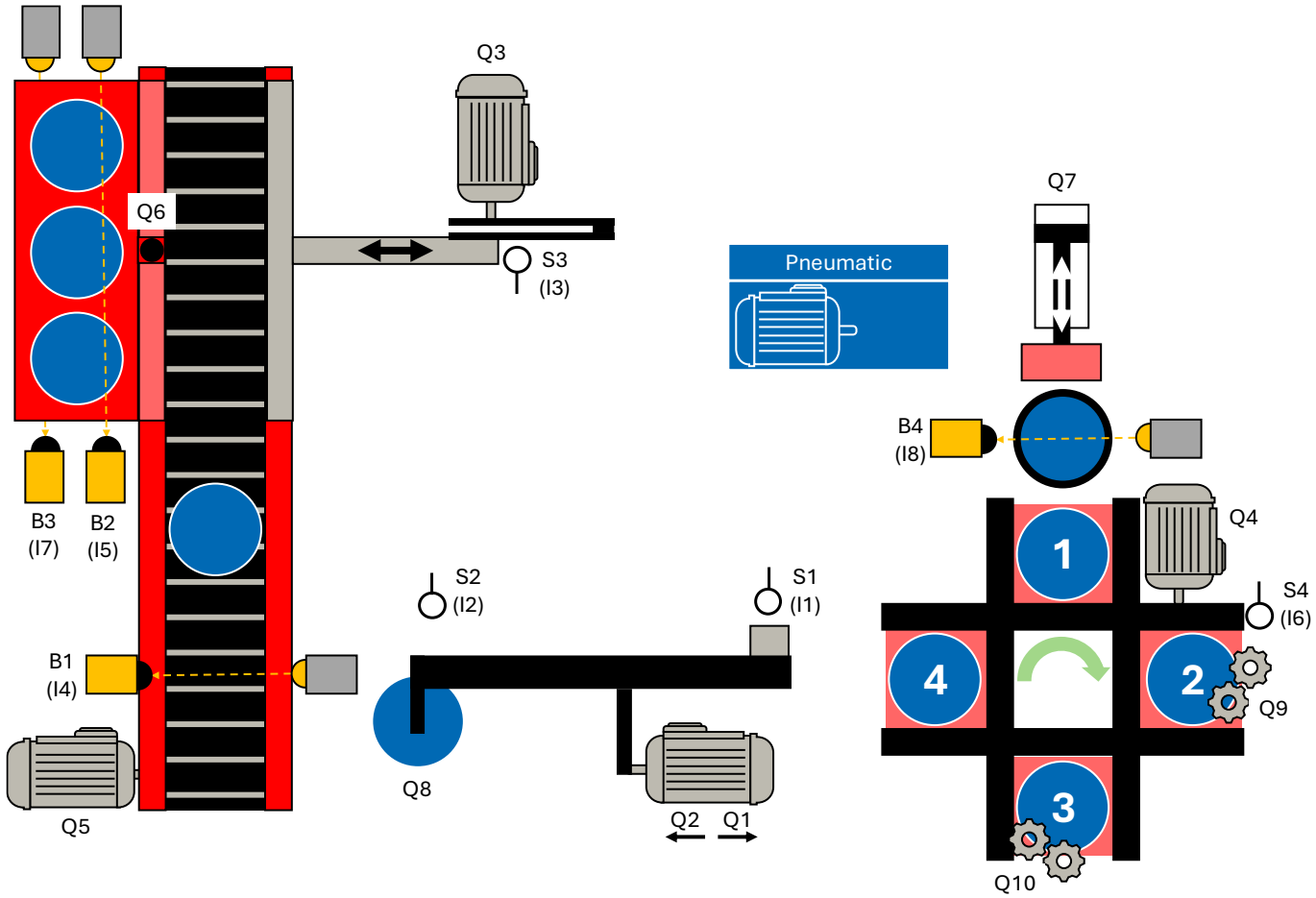
fischertechnik

GROLLMUS

WISSEN | SOFTWARE | AUTOMATISIERUNG

Fertigungslinie 24V

Strukturierte Programmierung



Inhaltsverzeichnis

5 Strukturierte Programmierung

5.1 Einführung

Strukturierte Programmierung in SPS-Systemen dient dazu, komplexe Programme durch die Aufteilung in kleinere, übersichtliche Bausteine zu organisieren. Dies führt zu einer verbesserten Lesbarkeit, Wartbarkeit und Wiederverwendbarkeit des Codes. Das Anwenderprogramm kann nach technologischen oder funktionellen Gesichtspunkten strukturiert werden.

In einem SPS-Programm werden Bausteine wie Funktionen (FC) und Funktionsbausteine (FB) verwendet, um Programmteile zu strukturieren.

Die Bausteine sollten über ihre Bausteinschnittstellen miteinander kommunizieren, anstatt direkt auf globale Variablen zuzugreifen. Die Parameterübergabe erfolgt über Ein- und Ausgänge sowie InOut-Parameter.

Um die Codebausteine im Steuerungsprogramm auszuführen, müssen sie aufgerufen werden.

5.2 Funktion

Funktionen (FCs) sind Codebausteine ohne Gedächtnis. Sie haben keinen Datenspeicher, in denen Werte von Bausteinparametern gespeichert werden könnten. Deshalb müssen beim Aufruf einer Funktion alle Schnittstellenparameter beschaltet werden. Um Daten dauerhaft zu speichern, müssen zuvor globale Datenbausteine angelegt werden.

Funktionen sind ideal für Aufgaben, die keinen Speicherbedarf über mehrere Zyklen hinweg haben, wie mathematische Berechnungen oder logische Verknüpfungen.

Eine Funktion enthält ein Programm, das immer dann ausgeführt wird, wenn die Funktion von einem anderen Codebaustein aufgerufen wird.

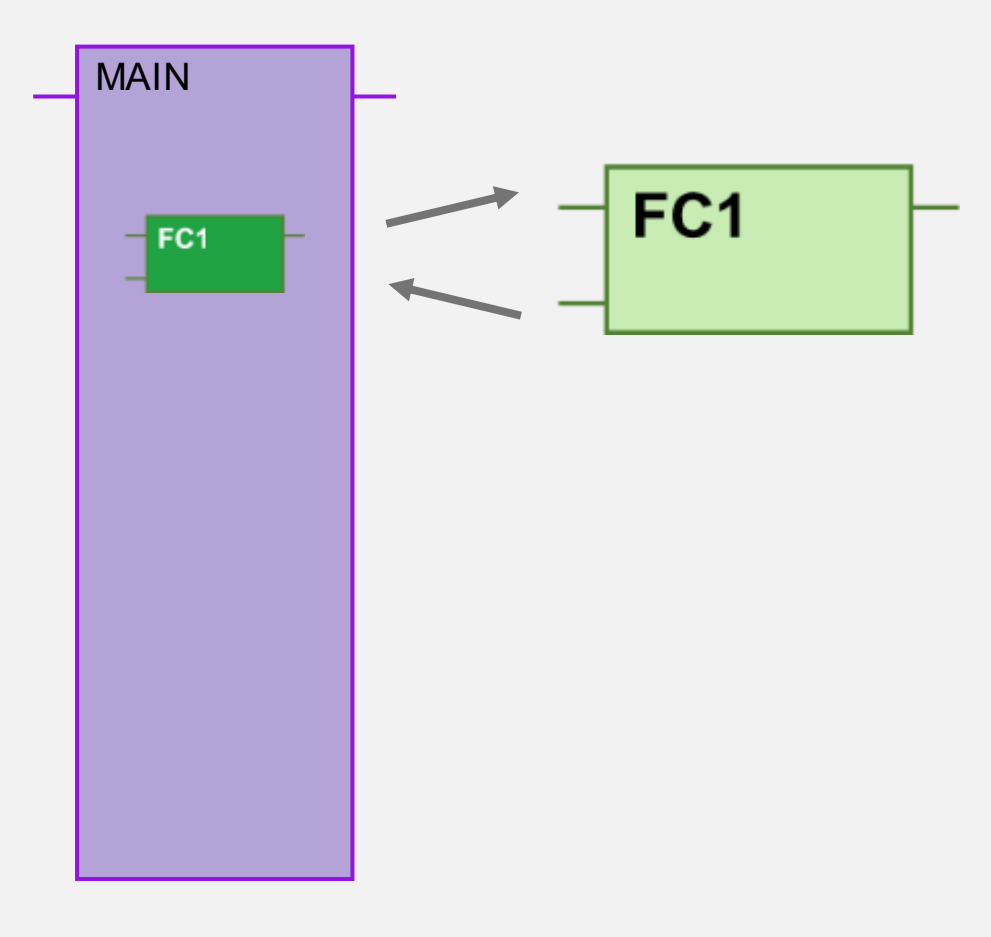


Bild 1 Beispiel: Aufruf einer Funktion aus MAIN

Eine Funktion kann auch mehrmals an verschiedenen Stellen innerhalb eines Programms aufgerufen werden.

5.3 Funktionsbaustein

Funktionsbausteine (FBs) sind Codebausteine, die ihre Eingangsvariablen, Ausgangsvariablen, Durchgangsvariablen und auch die statischen Variablen dauerhaft in Instanz-Datenbausteinen ablegen, sodass sie auch nach der Bausteinbearbeitung zur Verfügung stehen. Deshalb werden sie auch als Bausteine mit Gedächtnis bezeichnet.

Funktionsbausteine werden bei Aufgaben verwendet, die mit Funktionen nicht realisierbar sind:

- Immer wenn in den Bausteinen Zeiten und Zähler benötigt werden oder
- wenn eine Information in dem Programm gespeichert werden muss (z.B. Zustand der Schrittkette).

Funktionsbausteine werden stets ausgeführt, wenn ein Funktionsbaustein von einem anderen Codebaustein aufgerufen wird.

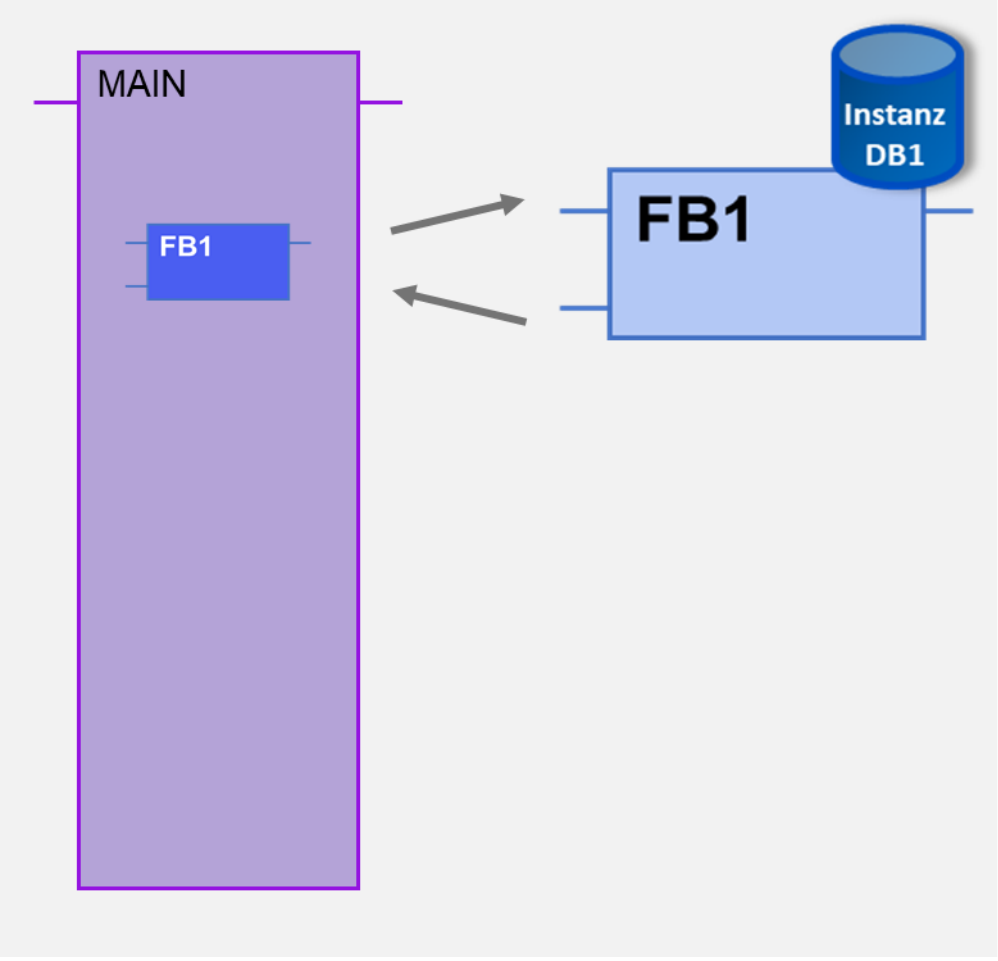


Bild 2 Beispiel: Aufruf eines Funktionsbausteins aus MAIN

Ein Funktionsbaustein kann auch mehrmals an verschiedenen Stellen innerhalb eines Programms aufgerufen werden.

Ein Aufruf eines Funktionsbausteins wird als Instanz bezeichnet. Jeder Instanz eines Funktionsbausteins wird ein Speicherbereich zugeordnet, der die Daten enthält, mit denen der Funktionsbaustein arbeitet.

5.4 Neuen Baustein hinzufügen

Im TIA-Portal werden die Bausteine in der Projektnavigation unterhalb der PLC im Ordner "Programmbausteine" verwaltet.
Durch einen Doppelklick auf den Befehl "Neuen Baustein hinzufügen" innerhalb des Ordners "Programmbausteine" wird der Dialog "Neuen Baustein hinzufügen" geöffnet, über den ein neuer Baustein angelegt werden kann.

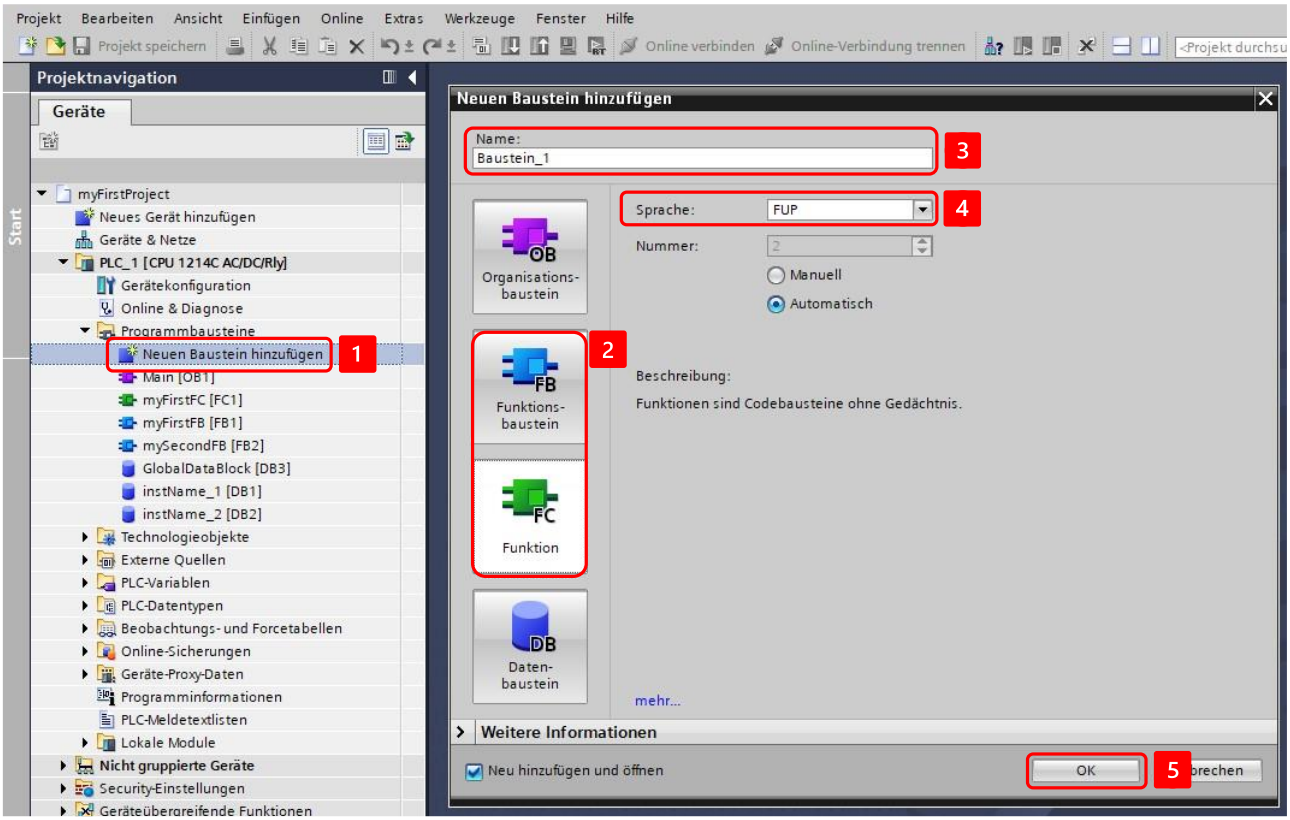


Bild 3 Neuen Baustein hinzufügen

Hier ist der Bausteintyp (2), Name (3), sowie die gewünschte Programmiersprache (4) zu wählen.

5.5 Bausteinaufruf

Um die Codebausteine im Steuerungsprogramm auszuführen, müssen sie aufgerufen werden. Der für die zyklische Programmbearbeitung zuständige Codebaustein wird üblicherweise als "MAIN" bezeichnet. Dieser wird vom Betriebssystem gestartet und bildet die Schnittstelle zum Betriebssystem. Die CPU verarbeitet den Programmcode, der sich im "MAIN" befindet. Innerhalb des "MAIN" können die in Funktionen und Funktionsbausteinen strukturierten Programmteile aufgerufen werden.

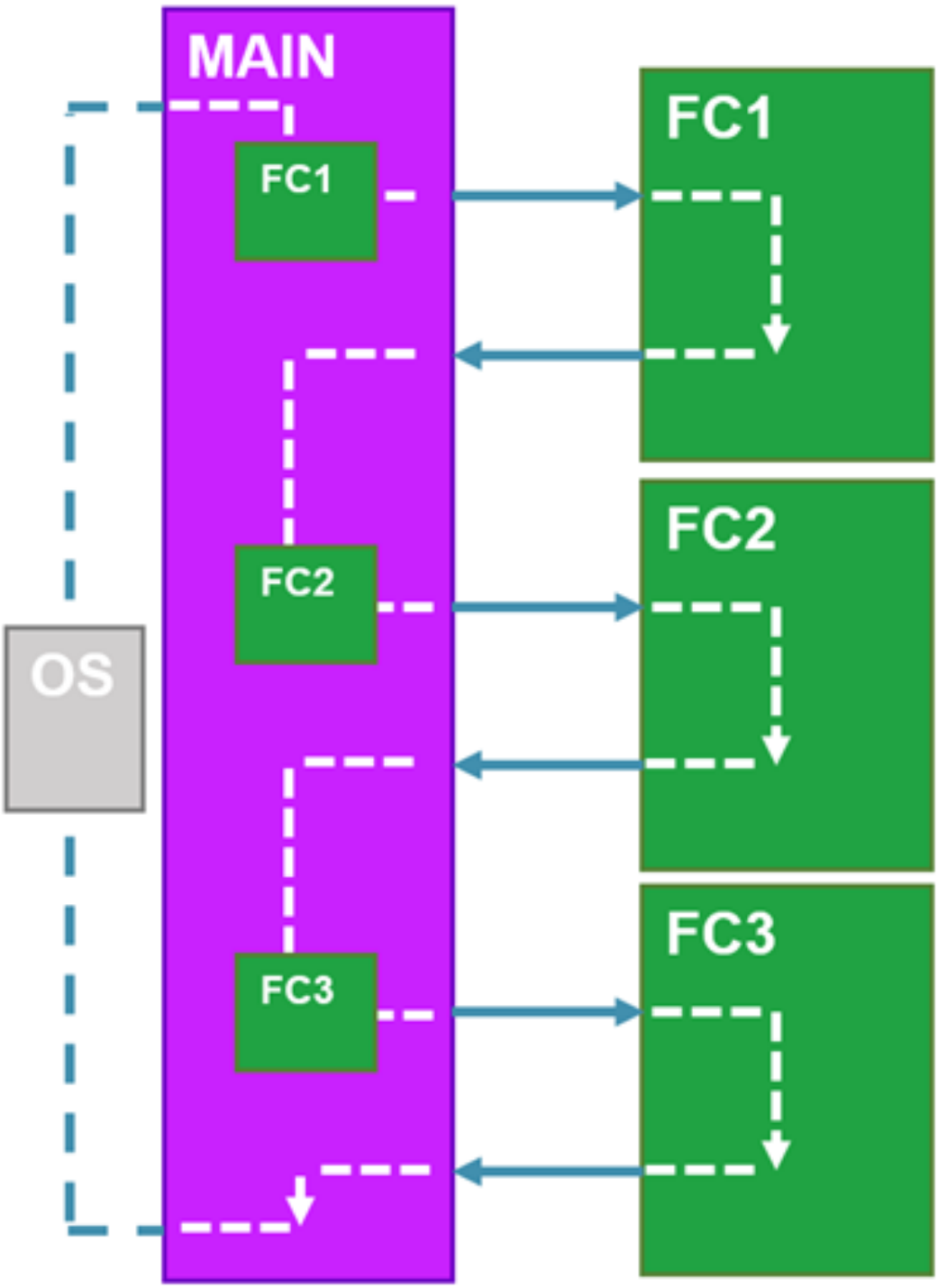


Bild 4 Bausteinaufruf im MAIN

Funktionen und Funktionsbausteine strukturieren das Programm, wodurch es lesbarer und wartbarer wird.

Alle aufgerufenen Bausteine werden nacheinander abgearbeitet.
Das Betriebssystem der CPU ruft den " MAIN " nach dem Programmzyklus erneut auf, wobei alle darin programmierten Befehle erneut ausgeführt werden.

Ein Baustein kann abgearbeitet werden, indem er beispielsweise aus dem "MAIN" aufgerufen wird. Alternativ kann er auch aus einem FB oder FC aufgerufen werden, die ihrerseits im "MAIN" aufgerufen werden.

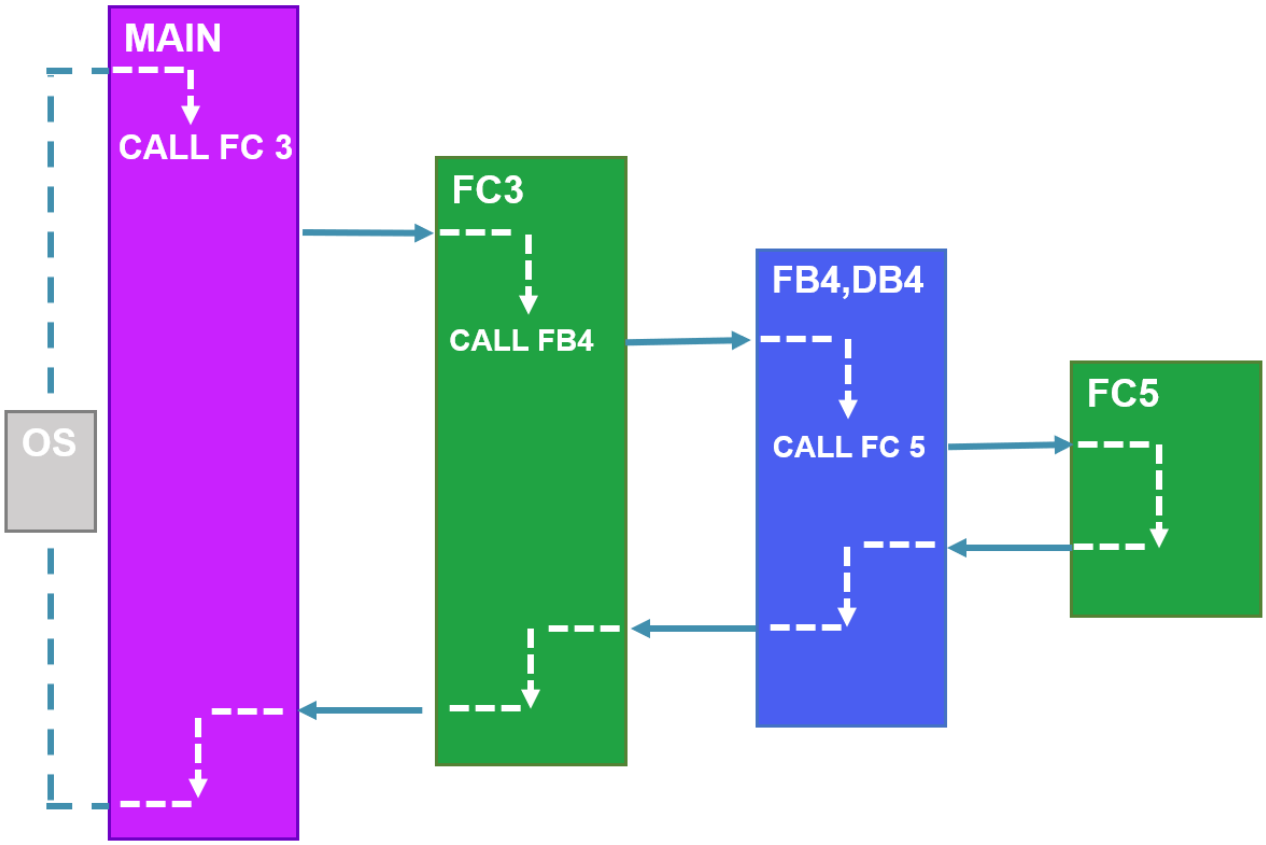


Bild 5 Bausteinaufruf

5.6 Parameterübergabe

Beim Aufruf von Funktionen und Funktionsbausteinen können Parameter (Variablen und Variablenwerte) übergeben werden. Der Programmcode innerhalb des Codebausteins arbeitet dann mit den übergebenen Werten der Formalparameter und kann Ergebnisse über Ausgangsparameter oder den Funktionswert zurückgeben.

Der Datenaustausch erfolgt über die Bausteinschnittstelle. In der Bausteinschnittstelle der Funktion (FC) oder des Funktionsbausteins (FB) werden diese deklariert und können lokal im Baustein verwendet werden.

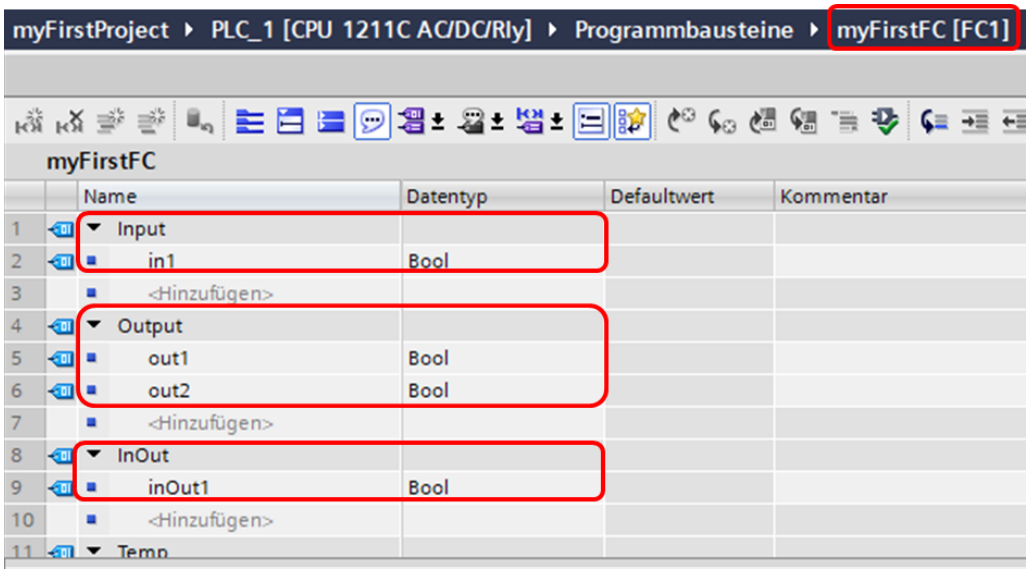


Bild 6 Formalparameter in der Bausteinschnittstelle

Beim Bausteinaufruf werden diese Formalparameter mit Aktualparametern (Globalen Variablen der SPS) verschaltet.

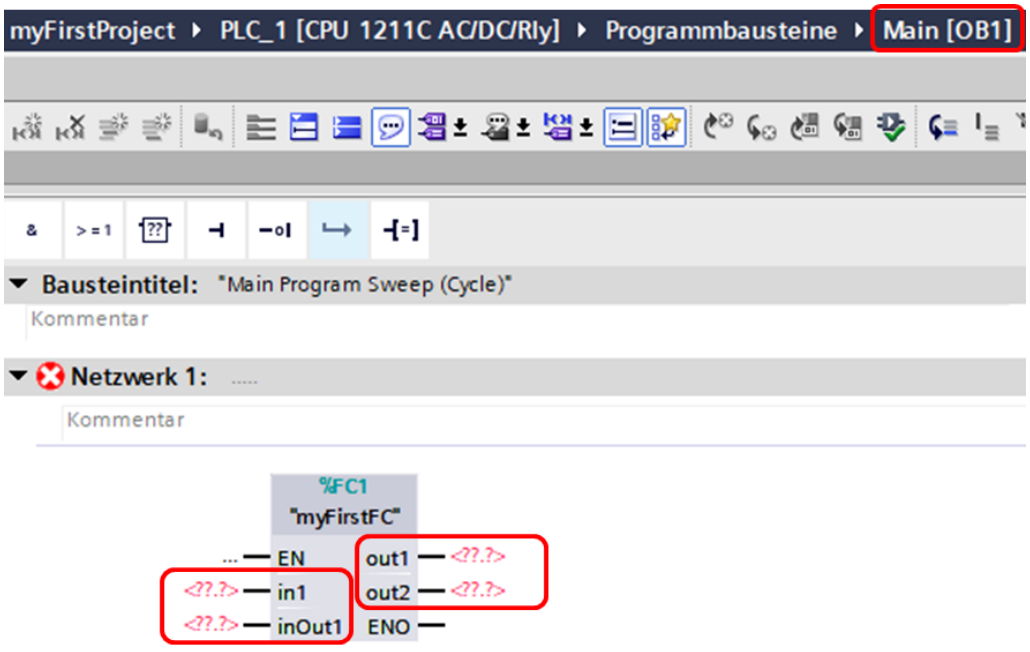


Bild 7 Übergabe der Akualparamter beim Aufruf

5.7 Aufruf einer Funktion (FC) in FUP

Damit die Funktion abgearbeitet wird, muss diese vom Programm aufgerufen werden. Der Aufruf kann über das Einfügen einer Leerbox (TIA-Shortcut "F8") erfolgen. Nachdem wir die Leerbox (1) eingefügt haben, ersetzen wir (2) in der Leerbox die Platzhalter ("???") durch den symbolischen Bausteinnamen, so wird die Leerbox durch den entsprechenden Bausteinaufruf ersetzt.

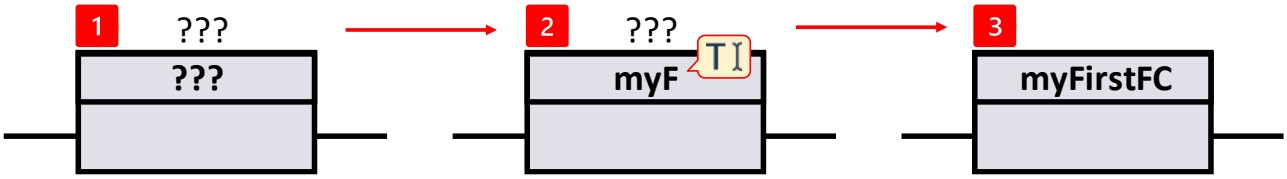


Bild 8 Bausteinaufruf aus Leerbox

Im TIA-Portal lässt der gewünschte Baustein auch mittels Drag & Drop, durch ziehen aus der Projektnavigation an die gewünschte Stelle, aufrufen:

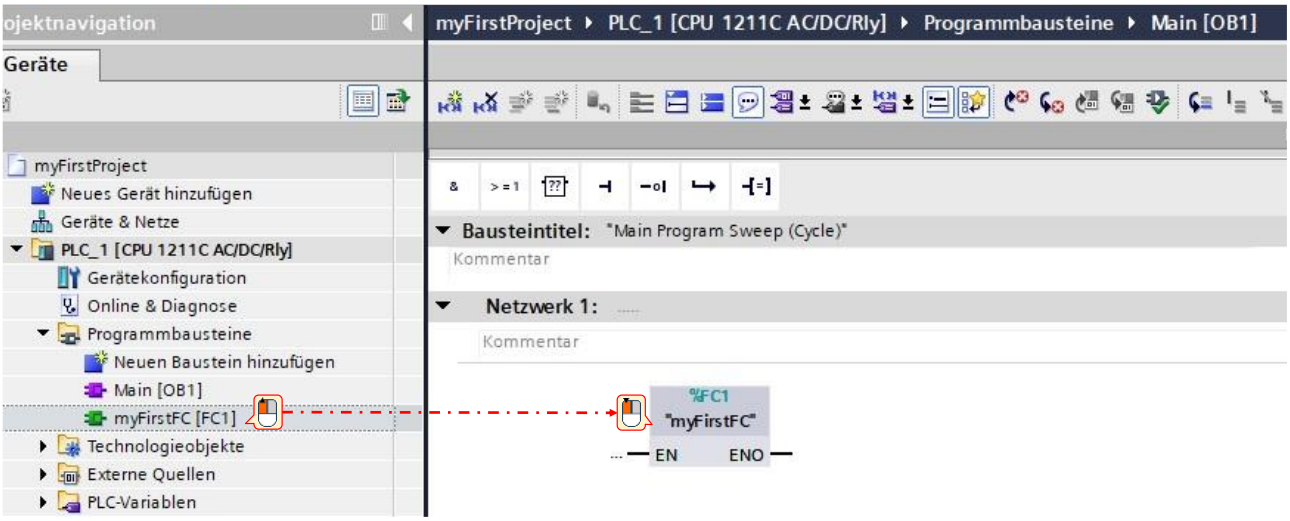


Bild 9 Bausteinaufruf im TIA Portal

Parameterübergabe

Besitzt der aufgerufene Baustein Schnittstellenparameter, so werden diese angezeigt. Bei Funktionen müssen die Formalparameter zwingend mit Aktualparametern versorgt werden.

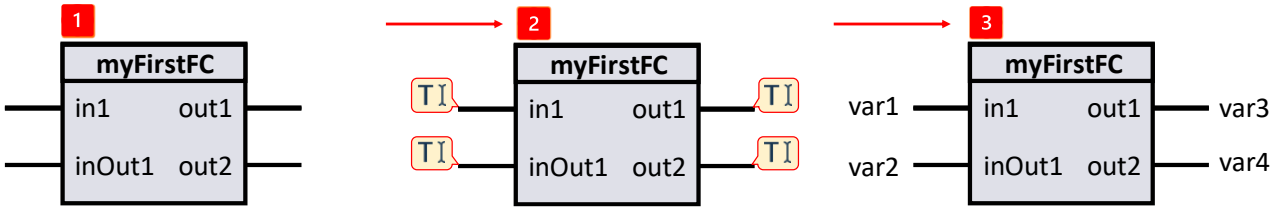


Bild 10 Funktion mit Bausteinschnittstelle

Parameter vom Typ "Input" sowie "InOut" werden links des Bausteines mittels Anschlussbeinchens dargestellt. Parameter vom Typ "Output" müssen rechts vom Baustein verschaltet werden.

Beispiel

Nach der Erstellung der Funktion "myFirstFC" wurde diese im "Main" (OB1) aufgerufen. Die Parameterübergabe ist noch nicht erfolgt; die zu verbindenden Formalparameter wurden vorerst mit Platzhaltern "<??.?>" versehen.

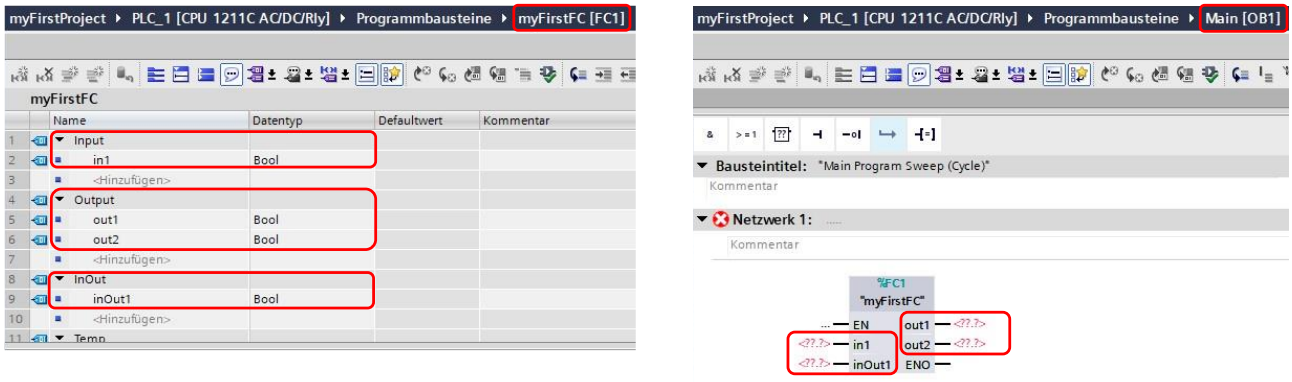


Bild 11 Bausteinschnittstelle im TIA-Portal

5.8 Aufruf eines Funktionsbausteins (FB) in FUP

Damit der Funktionsbaustein abgearbeitet wird, muss dieser vom Programm aufgerufen werden. Der Aufruf kann über das Einfügen einer Leerbox (TIA-Shortcut "F8") erfolgen. Nachdem wir die Leerbox (1) eingefügt haben und in der Leerbox die Platzhalter durch den Namen des Funktionsbausteins ersetzt haben, ersetzen wir den Platzhalter über der Leerbox durch den Instanznamen (2).

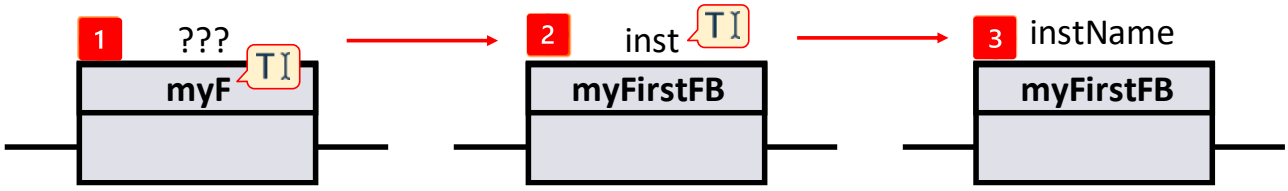


Bild 12 Funktionsbausteinaufruf aus Leerbox

Alternativ können Sie wie unter Aufruf einer Funktion gezeigt den Baustein per Drag & Drop aufrufen.

Parameterübergabe

Verfügt der aufgerufene Funktionsbaustein über Formalparameter in der Bausteinschnittstelle, werden diese angezeigt. Bei Funktionsbausteinen ist die Parameterübergabe nicht immer erforderlich, da der Instanz bereits ein privater Speicherbereich zugewiesen ist.

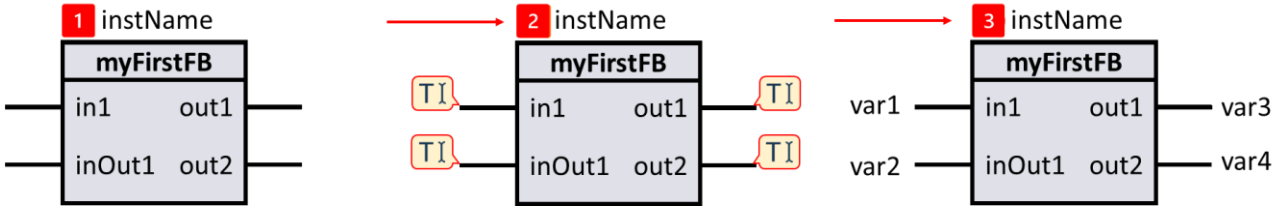


Bild 13 Funktionsbaustein mit Parameterübergabe

Funktionsbaustein mehrfach instanziieren

Wird ein Funktionsbaustein (FB) im Anwenderprogramm zweimal aufgerufen (instanziert), entstehen zwei separate Instanzen. In der sie ihre Daten über den Zyklus hinweg speichern können.

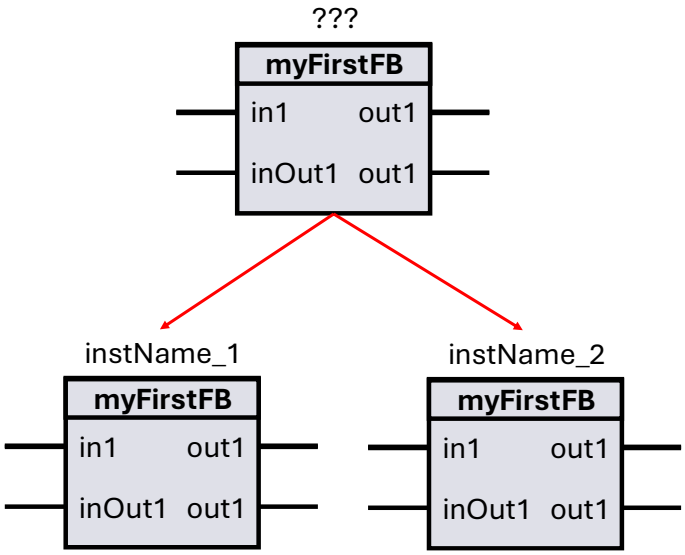


Bild 14 Instanzierung von zwei FBs

Aufrufoptionen

Die Instanzen können, abhängig von der Art des aufrufenden Bausteines, direkt in der Bausteinschnittstelle liegen (= Multiinstanz im TIA Portal) oder als Globale Instanzen (= Einzelinstanz im TIA Portal) abgelegt werden.

Beispiel

Enthält der FB beispielsweise das Anwenderprogramm für eine Berechnung von Schaltvorgängen, so stellt jeder Aufruf eine Instanz dar, die nur Zustände zur Laufzeit dieses Aufrufs verwendet. Für jeden Aufruf wird eine eigenständige Instanz benötigt.
Somit sind alle Daten (Informationen), die zu dieser Berechnung gehören, in dieser zugeordneten Instanz vorhanden.
Bevor eine Instanz erzeugt werden kann, muss der zugeordnete FB bereits existieren.
Die Variablen der jeweiligen Instanz können in dieser beobachtet werden.

5.8.1 Vorgehensweise Aufruf mit Einzelinstanz

Das Vorgehen zum zweimaligen Aufrufen des Funktionsbausteines "myFirstFB" wird nun schrittweise im TIA-Portal gezeigt:

1. Erstellen des Funktionsbausteines "myFirstFB":

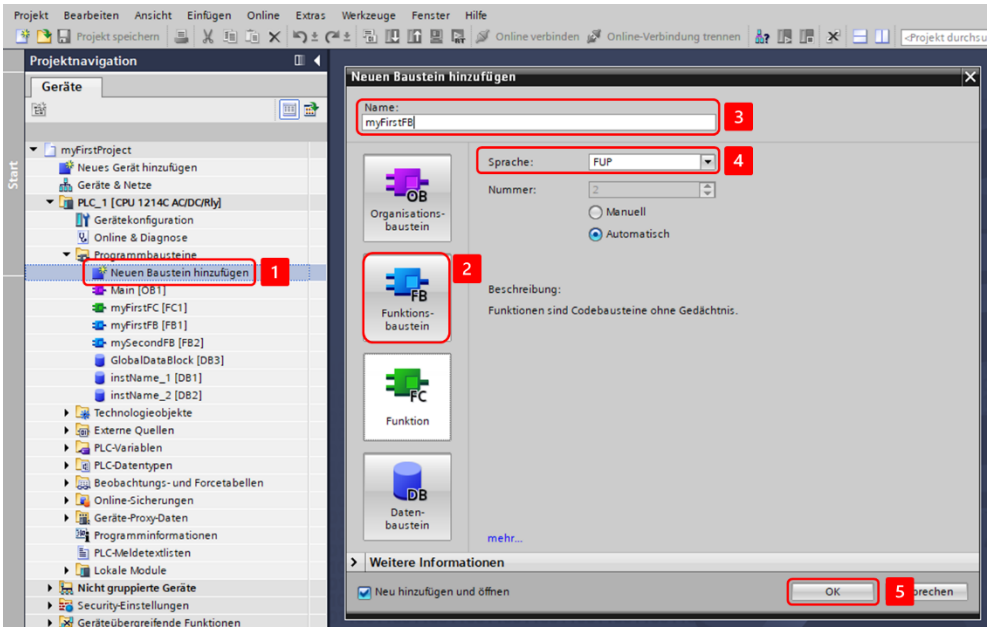


Bild 15 Neuen Baustein hinzufügen

2. Deklaration der Bausteinschnittstelle und Verschaltung der Variablen im Anweisungsteil des Bausteines:

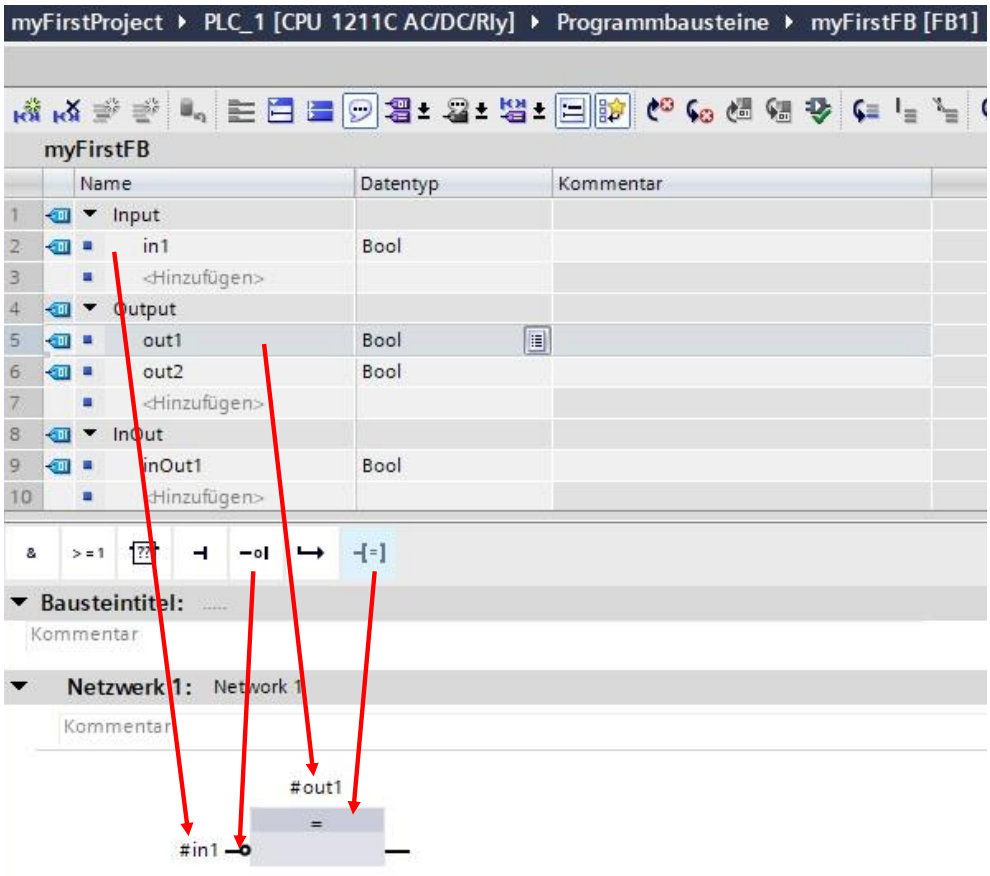


Bild 16 FB mit Bausteinschnittstelle im TIA Portal

3. Erster Aufruf von "myFirstFB" im ersten Netzwerk des OBs "MAIN" mittels Drag & Drop und Deklaration der ersten Instanz als Einzelinstanz:

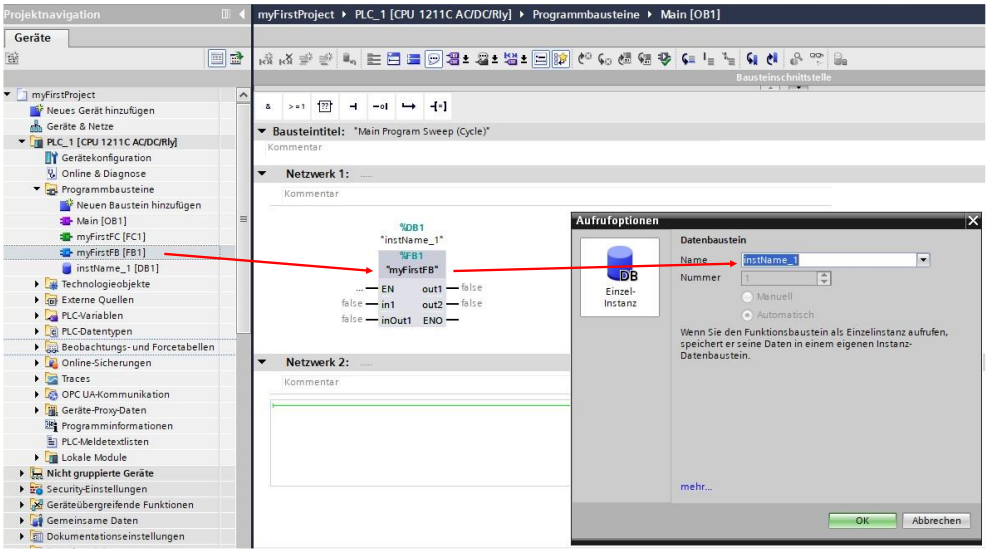


Bild 17 Instanzierung erster Bausteinaufruf

4. Zweiter Aufruf von "myFirstFB" im zweiten Netzwerk des OBs "MAIN" mittels Drag & Drop und Deklaration der zweiten Instanz als Einzelinstanz:

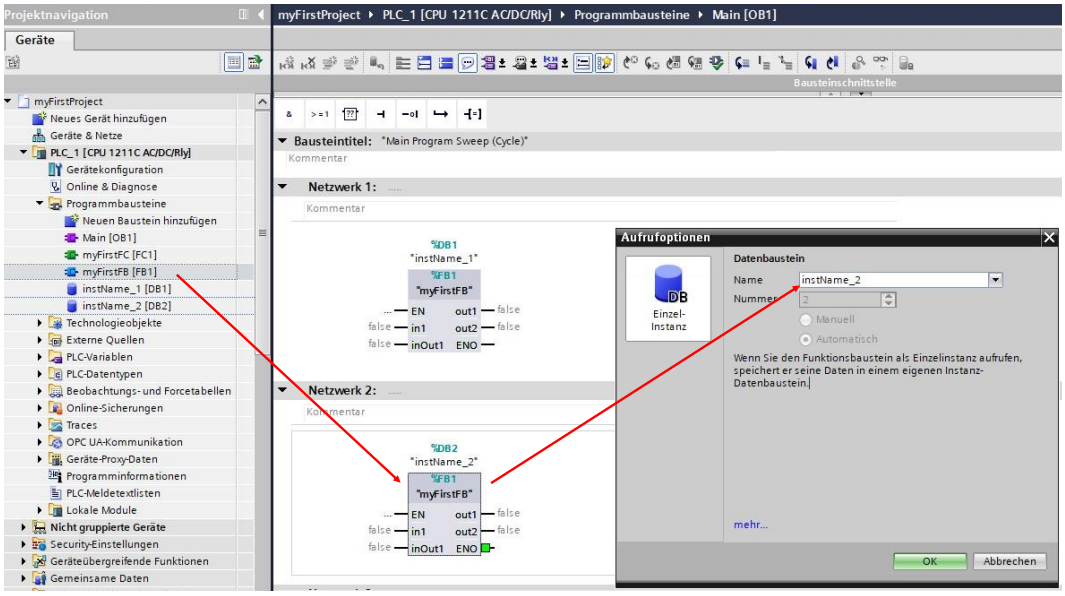


Bild 18 Instanzierung zweiter Bausteinaufruf

5. Beide Instanzdatenbausteine können beobachten werden:

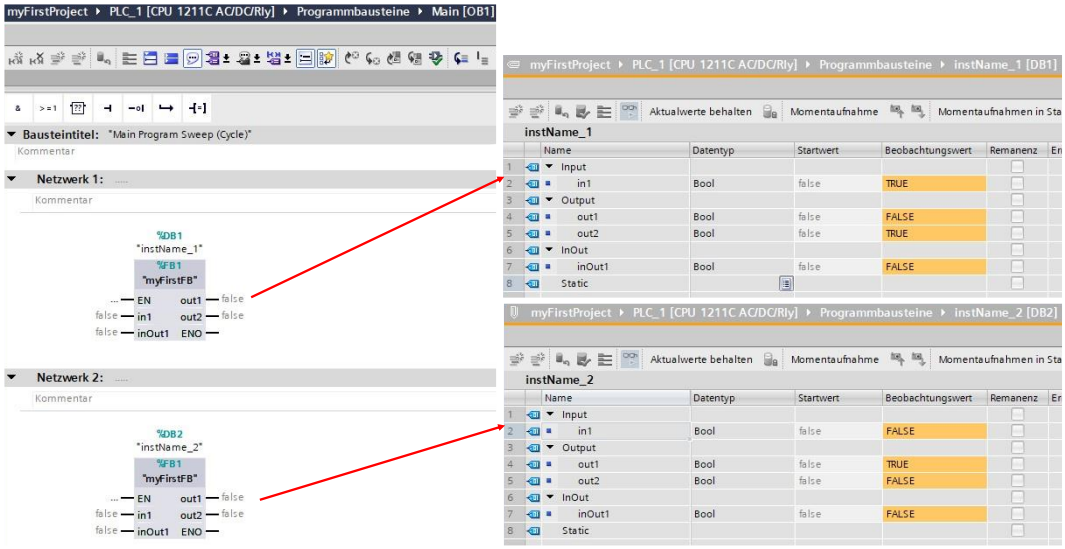


Bild 19 Aktualwerte in den Instanzen

Instanzdatenbausteine, sowie die Möglichkeit diese zu beobachten und zu steuern wird im nächsten Kapitel (Datenbausteine) detailliert beschrieben.

5.8.2 Aufrufoption als Multiinstanz (TIA-Portal)

Wird ein Funktionsbaustein in einem anderen Funktionsbaustein aufgerufen, so lässt sich in den Aufrufoptionen auch die Multiinstanz auswählen.
Durch die Verwendung von Multiinstanzen kann die Anzahl der Instanzdatenbausteine reduziert werden. Bei der Erstellung von Unterprogrammen ist der Einsatz von Multiinstanzen oft zwingend erforderlich. Möchte man z.B. einen Laufzeitzähler in einem Ansteuerbaustein für einen Motor realisieren benötigt jede Motorinstanz eine eigene IEC-Zählerinstanz. Multiinstanzen werden in die Bausteinschnittstelle des aufrufenden Bausteins gelegt.

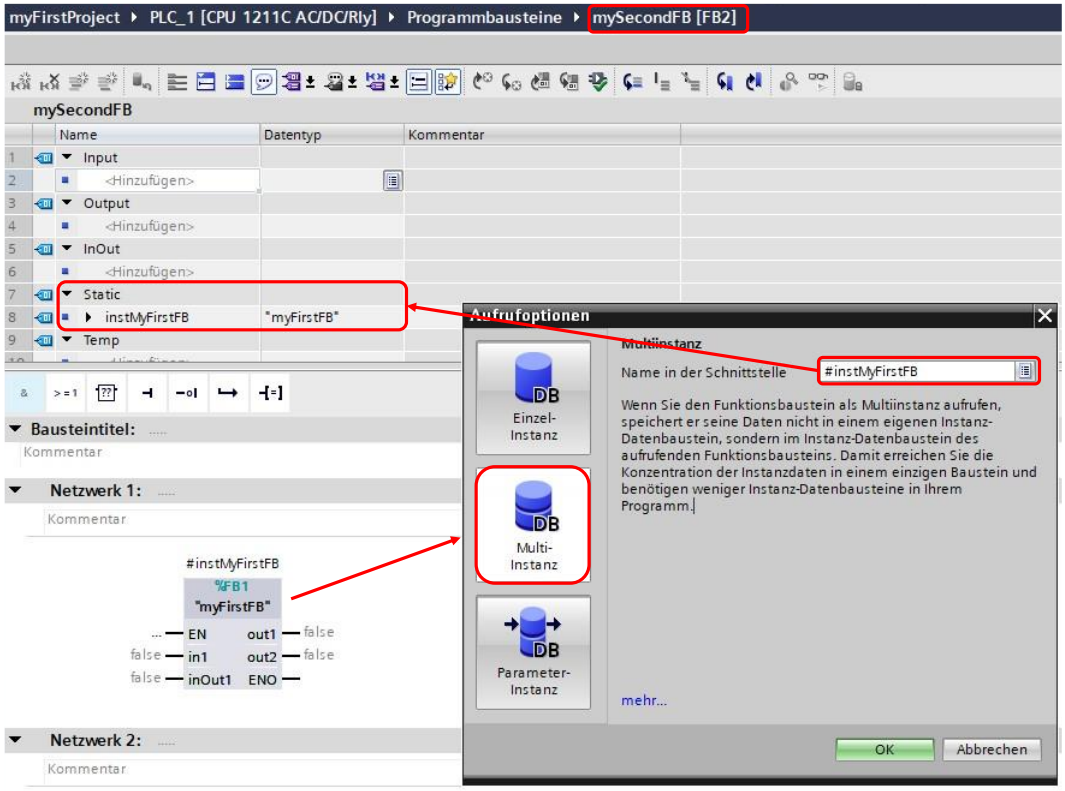


Bild 20 Aufrufoptionen im TIA Portal

5.8.3 Textuelle Deklaration als Multiinstanz (CODESYS / Beckhoff)

Die textuelle Deklaration der Instanzen erfolgt nach folgendem Schema.
Syntax:

Instanzname (Variablenname) : Bausteinname (Datentyp);

Beispiel:

```
//Deklaration der Instanzen
VAR
  instName_1 : myFirstFB;   //Instanz Aufruf 1
  instName_2 : myFirstFB;   //Instanz Aufruf 2
END_VAR
```

Bild 21 Deklaration der Instanzen

5.9 Aufruf einer Funktion (FC) in ST / SCL

Damit die Funktion abgearbeitet wird wollen wir diese im Programm aufrufen. Ein Funktionsaufruf ohne Rückgabewert in ST / SCL erfolgt durch den Funktionsnamen, gefolgt von "()" und einem Semikolon. In den runden Klammern erfolgt die Parameterübergabe, und das Semikolon schließt die Anweisung ab.

```
myFirstFC();
```

Bild 22 Funktionsaufruf ST / SCL

Im TIA-Portal lässt der gewünschte Baustein auch mittels Drag & Drop, durch ziehen aus der Projektnavigation an die gewünschte Stelle, aufrufen:

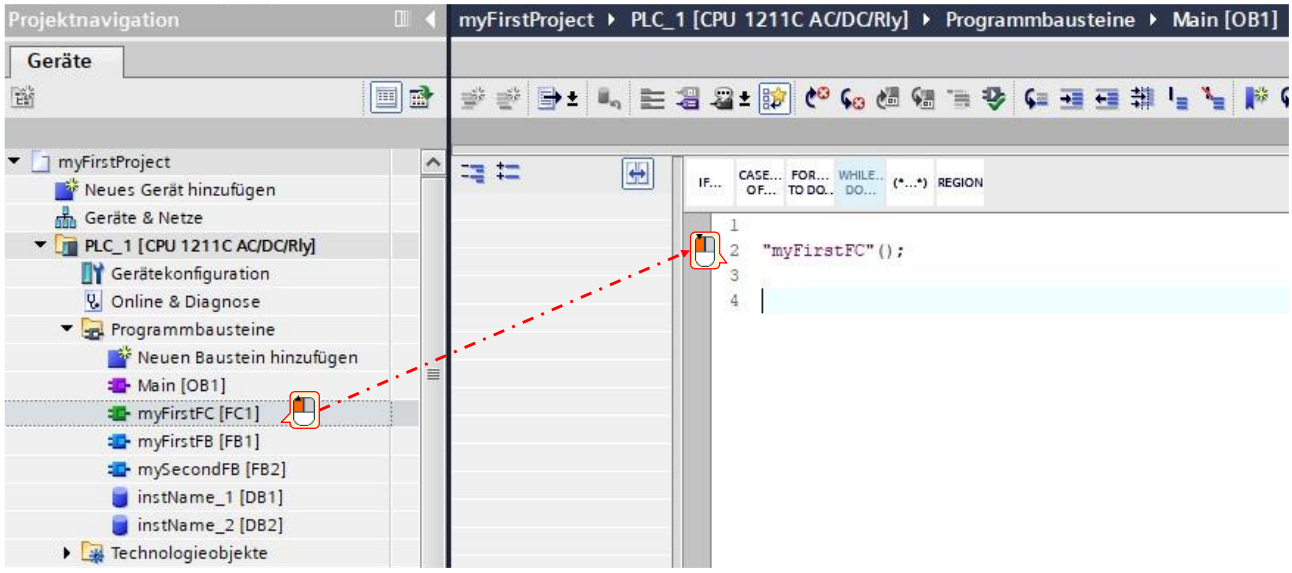


Bild 23 Bausteinaufruf im TIA Portal

Parameterübergabe

Besitzt der aufgerufene Baustein Schnittstellenparameter, so werden diese angezeigt. Bei Funktionen müssen die Formalparameter zwingend mit Aktualparametern versorgt werden.
Die Parameterübergabe erfolgt in runden Klammern, Parameter werden mit "," voneinander getrennt.

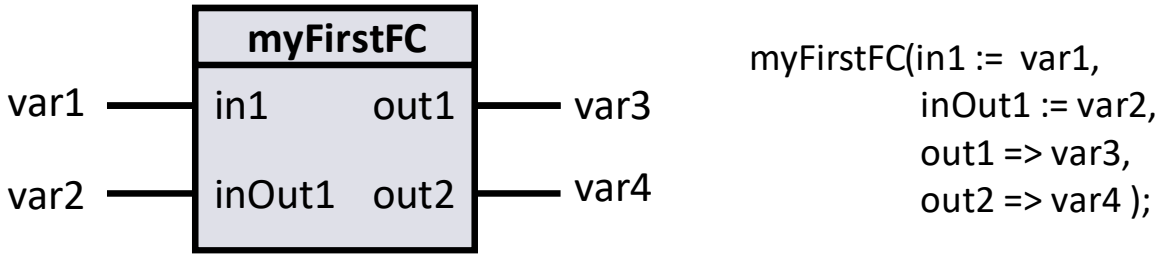


Bild 24 Syntax Funktionsaufruf, mit Parameterübergabe in SCL

Parameter vom Typ "Input" sowie "InOut" werden mittels ":=" zugewiesen.
Parameter vom Typ "Output" müssen mit "=>" verschaltet werden.

Beispiel

In diesem Bild wurden in der Funktion "myFirstFC" folgende Schnittstellen-parameter deklariert.

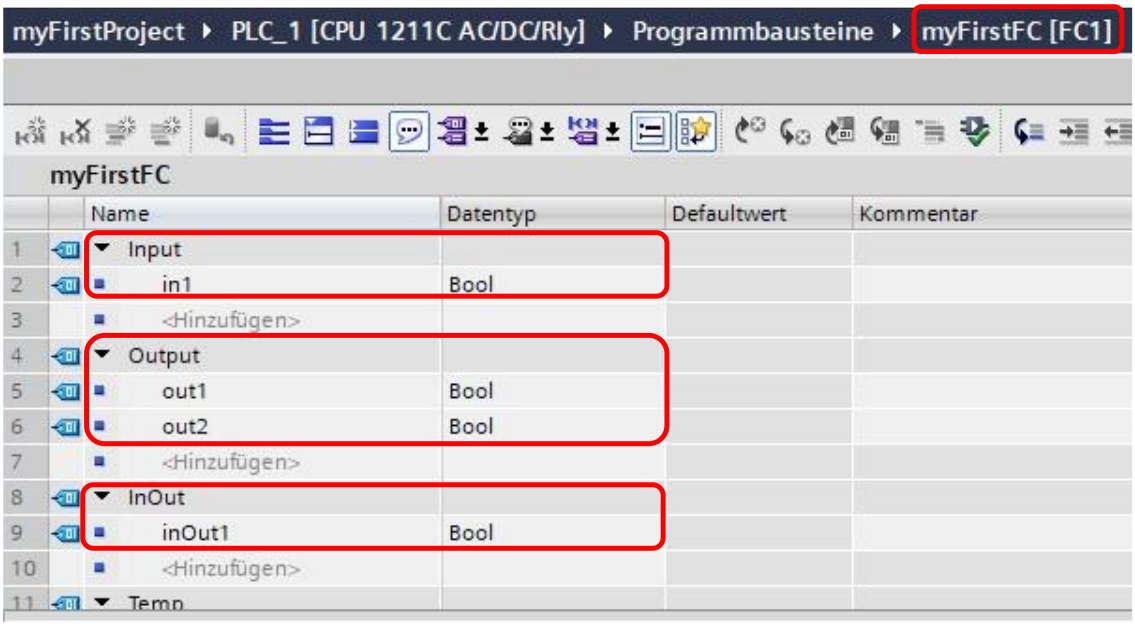


Bild 25 Bausteinschnittstelle einer Funktion im TIA-Portal

Nach der Erstellung der Funktion "myFirstFC" wurde diese im "MAIN" (OB1) aufgerufen. Die Parameterübergabe wurde noch nicht durchgeführt, und die zu verbindenden Formalparameter sind vorerst mit Platzhaltern belegt. Diese Platzhalter geben Auskunft über Datentyp und Parameterart (Input, Output, InOut).

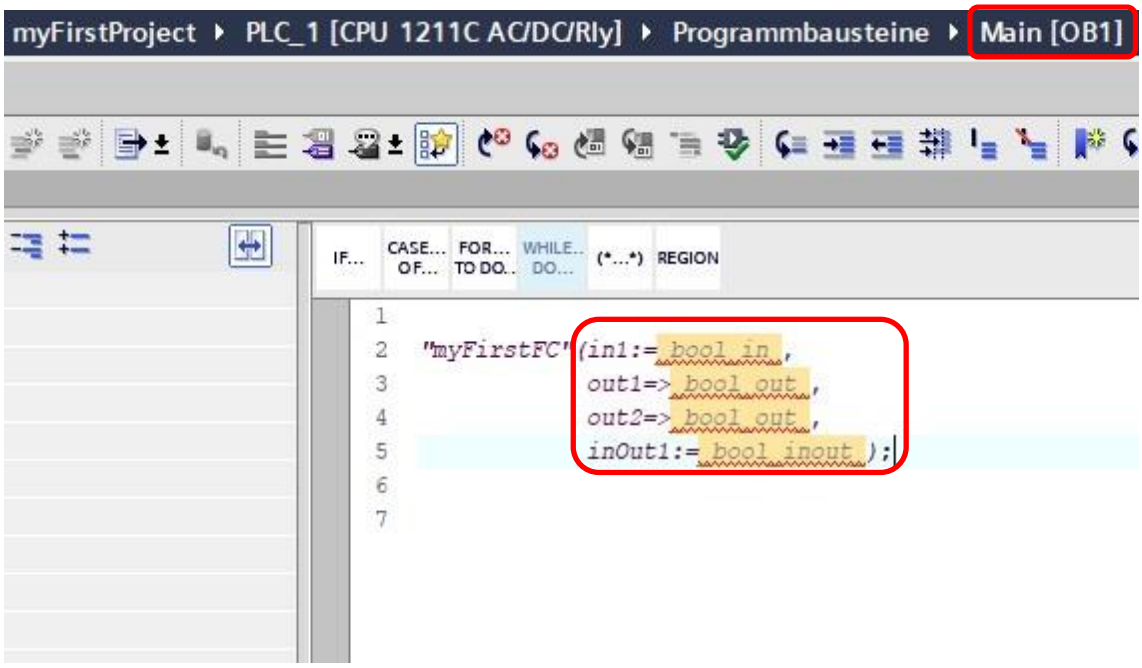


Bild 26 Bausteinschnittstelle im TIA-Portal

5.10 Aufruf eines Funktionsbausteins (FB) in ST / SCL

Um den Funktionsbaustein abzuarbeiten, rufen wir ihn im Programm auf. Der Aufruf eines Funktionsbausteins (FB) unterscheidet sich hauptsächlich dadurch, dass ihm eine Instanz zugeordnet wird. Der Aufruf erfolgt ähnlich wie bei einer Funktion (FC), jedoch wird anstelle des Bausteinnamens der Name der zuvor deklarierten Instanz verwendet, gefolgt von "()" und einem Semikolon. In den runden Klammern erfolgt die Parameterübergabe, und das Semikolon schließt die Anweisung ab.

```
instMyFirstFB();
```

Bild 27 Syntax Instanz, ohne Parameterübergabe in SCL

Parameterübergabe

Besitzt die aufgerufene Instanz Schnittstellenparameter, so werden diese angezeigt. Bei Funktiosbausteinen ist die Parameterübergabe nicht immer erforderlich, da der Instanz bereits ein privater Speicherbereich zugeordnet ist.

```
instMyFirstFB(in1 :=  var1,
              inOut1 := var2,
              out1 => var3,
              out2 => var4 );
```

Bild 28 Syntax Instanz, mit Parameterübergabe in SCL

Aufrufoptionen

Die Instanzen können, abhängig von der Art des aufrufenden Bausteines, direkt in der Bausteinschnittstelle liegen (= Multiinstanz im TIA Portal) oder als Globale Instanzen (= Einzelinstanz im TIA Portal) abgelegt werden.

Funktionsbaustein mehrfach instanziieren

Wird ein FB im Anwenderprogramm zweimal aufgerufen, so existieren zwei Instanzen. Jeder Instanz wird ein eigener Speicherbereich zugeordnet, in welchem die Instanz ihre Daten, über den Zyklus hinweg abspeichern kann.

Beispiel

Enthält der FB beispielsweise das Anwenderprogramm für eine Berechnung von Schaltvorgängen, so stellt jeder Aufruf eine Instanz dar, die nur Zustände zur Laufzeit dieses Aufrufs verwendet. Für jeden Aufruf wird eine eigenständige Instanz benötigt.
Somit sind alle Daten (Informationen), die zu dieser Berechnung gehören, in dieser zugeordneten Instanz vorhanden.
Bevor eine Instanz erzeugt werden kann, muss der zugeordnete FB bereits existieren.
Die Variablen der jeweiligen Instanz können in dieser beobachtet werden.

5.10.1 Vorgehensweise Aufruf mit Einzelinstanz

Das Vorgehen zum zweimaligen Aufrufen des Funktionsbausteins "myFirstFB" wird nun Schritt für Schritt im TIA-Portal erläutert.

1. Erstellen des Funktionsbausteines "myFirstFB":

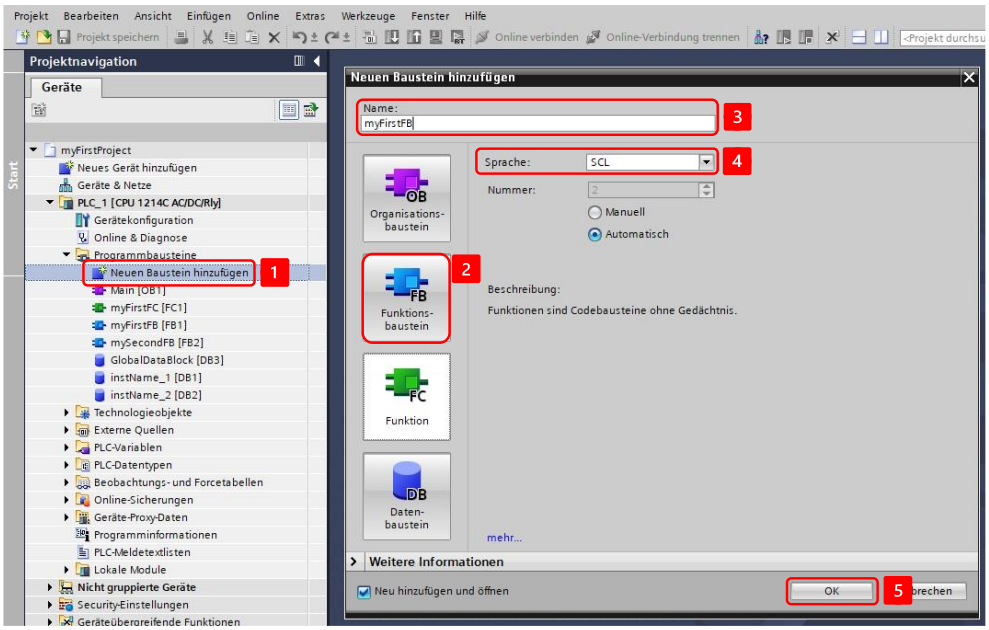


Bild 29 Neuen Baustein hinzufügen

2. Deklaration der Bausteinschnittstelle und Verschaltung der Variablen im Anweisungsteil des Bausteines:

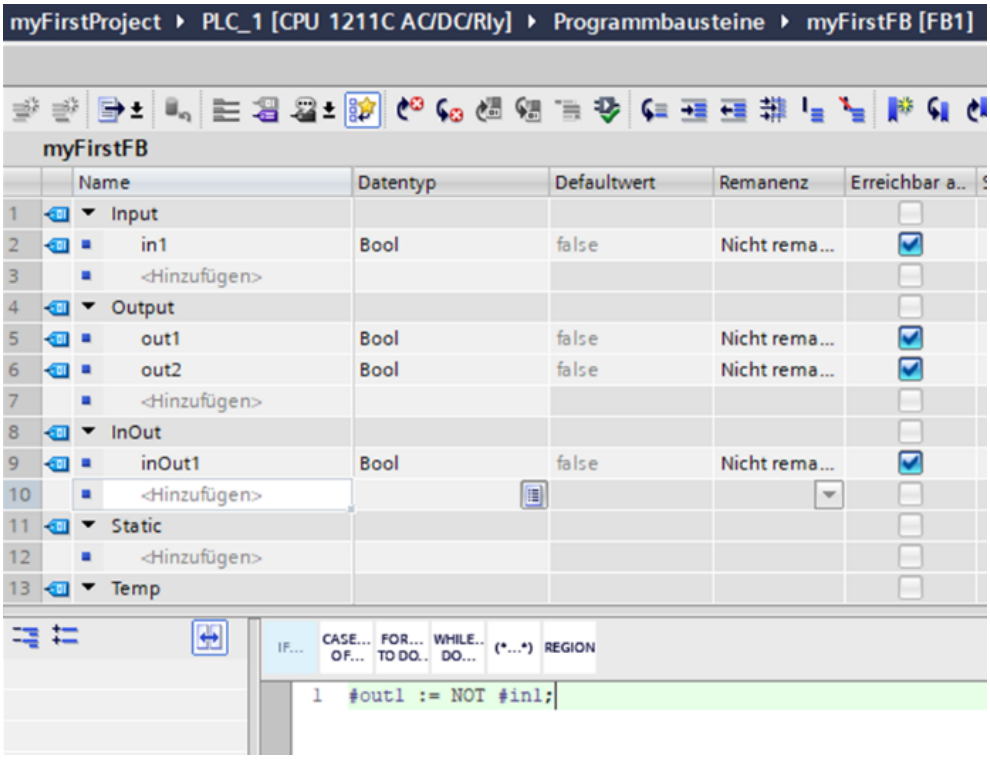


Bild 30 FB mit Bausteinschnittstelle im TIA Portal

3. Erster Aufruf von "myFirstFB" im OB "MAIN" mittels Drag & Drop und Deklaration der ersten Instanz als Einzelinstanz:

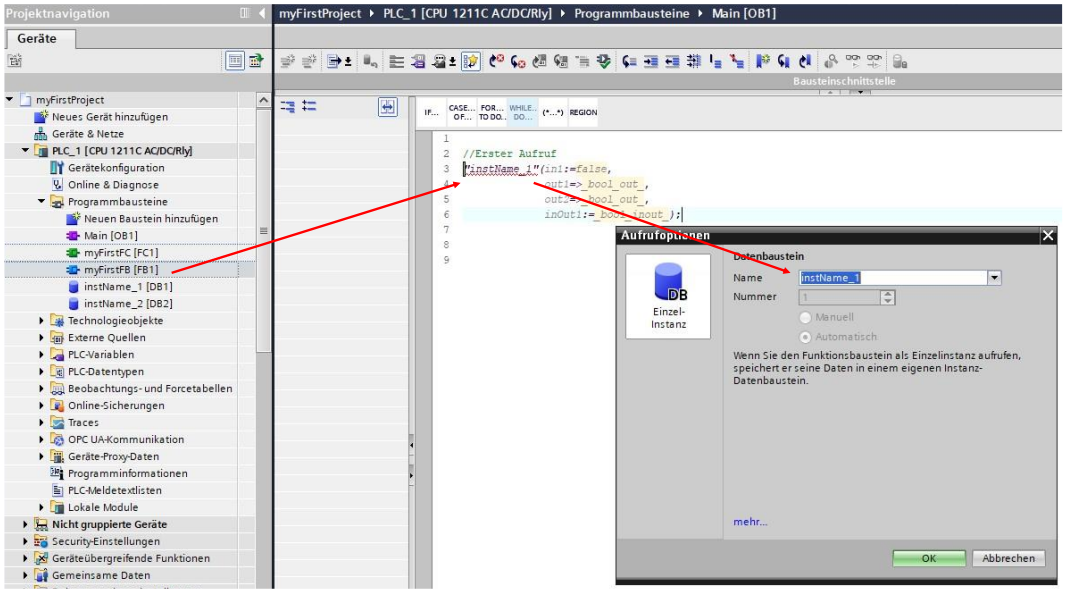


Bild 31 Instanzierung erster Bausteinaufruf

4. Zweiter Aufruf von "myFirstFB" im OB "MAIN" mittels Drag & Drop und Deklaration der zweiten Instanz als Einzelinstanz:

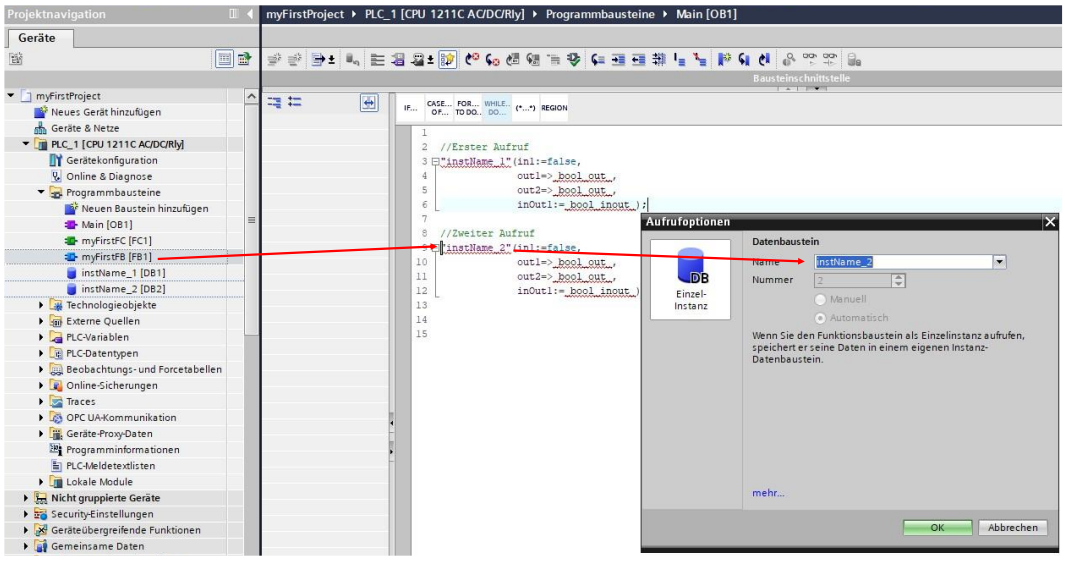


Bild 32 Instanzierung zweiter Bausteinaufruf

5. Beide Instanzdatenbausteine können beobachten werden:

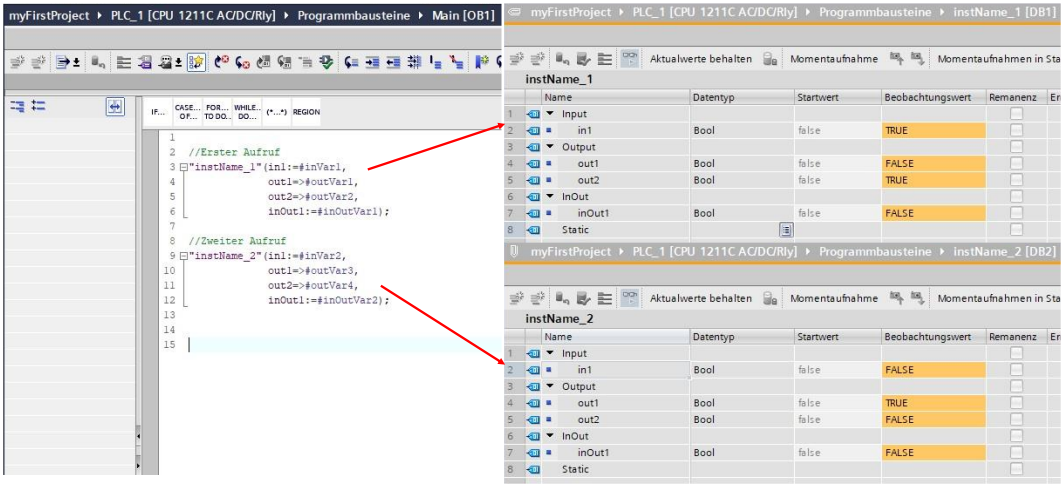


Bild 33 Aktualwerte in den Instanzen

Instanzdatenbausteine, sowie die Möglichkeit diese zu beobachten und zu steuern wird im nächsten Kapitel (Datenbausteine) detailliert beschrieben.

5.10.2 Aufrufoption als Multiinstanz (TIA-Portal)

Wird ein Funktionsbaustein in einem anderen Funktionsbaustein aufgerufen, so lässt sich in den Aufrufoptionen auch die Multiinstanz auswählen.
Durch die Verwendung von Multiinstanzen kann die Anzahl der Instanzdatenbausteine reduziert werden. Bei der Erstellung von Unterprogrammen ist der Einsatz von Multiinstanzen oft zwingend erforderlich. Möchte man z.B. einen Laufzeitzähler in einem Ansteuerbaustein für einen Motor realisieren benötigt jede Motorinstanz eine eigene IEC-Zählerinstanz. Multiinstanzen werden in die Bausteinschnittstelle des aufrufenden Bausteins gelegt.

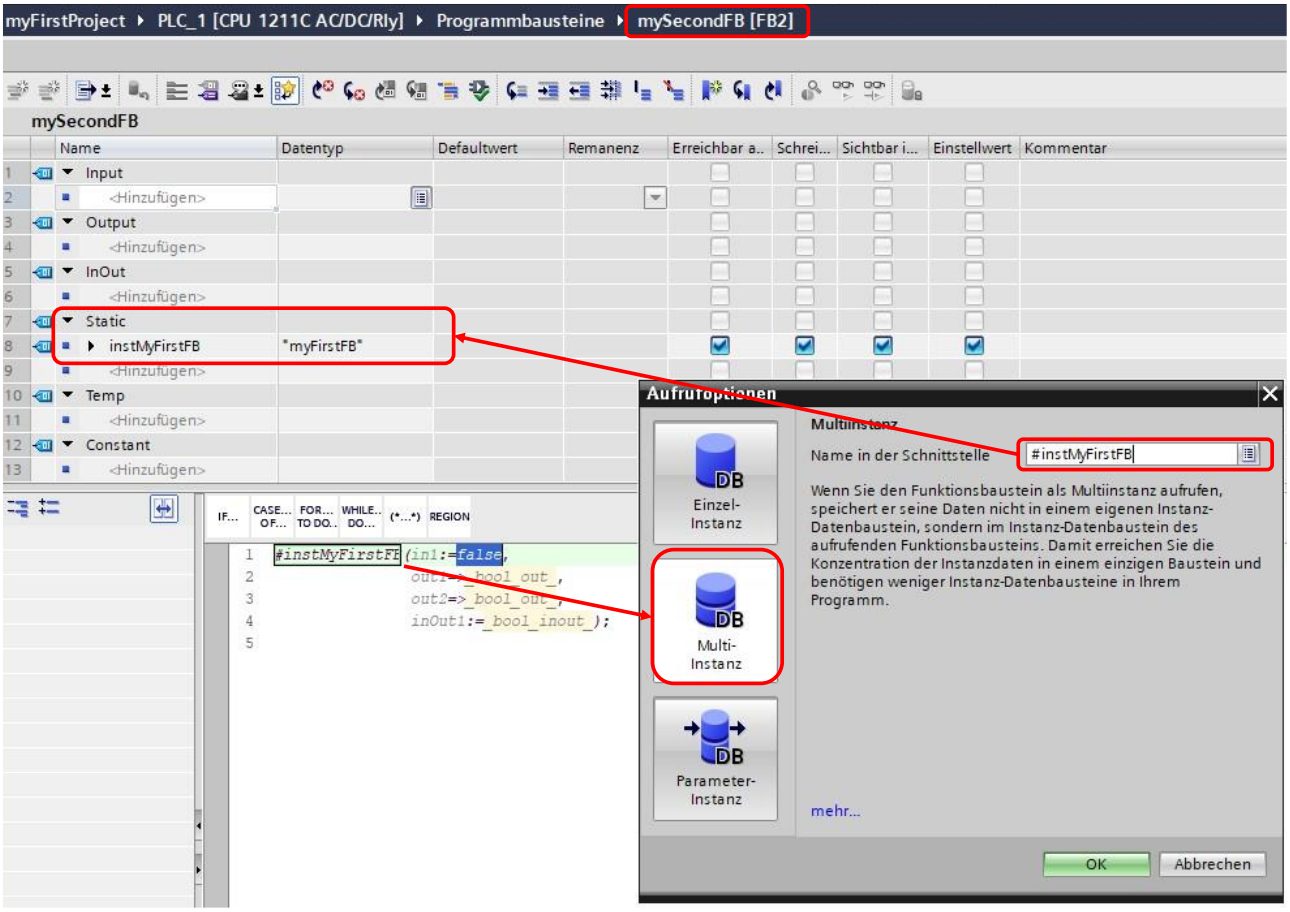


Bild 34 Aufrufoptionen im TIA Portal

5.10.3 Textuelle Deklaration als Multiinstanz (CODESYS / Beckhoff)

Die textuelle Deklaration der Instanzen erfolgt nach folgendem Schema.
Syntax:

Instanzname (Variablenname) : Bausteinname (Datentyp);

Beispiel:

```
//Deklaration in der Bausteinschnittstelle
VAR
  instMyFirstFB : myFirstFB;   //Deklaration der Instanz
END_VAR


//Implementierung im Programm
instMyFirstFB(in1:=       ,
              inOut1:=    ,
              out1=>      ,
              out2=>      );
```

Bild 35 Deklaration und Aufruf der Instanz

Funktionsbaustein mehrfach instanziieren

Wird ein FB im Anwenderprogramm zweimal aufgerufen, so existieren zwei Instanzen. Jeder Instanz wird ein eigener Speicherbereich zugeordnet, in welchem die Instanz ihre Daten, über den Zyklus hinweg abspeichern kann.

```
//Deklaration der Instanzen
VAR
   instName_1 : myFirstFB;   //Instanz Aufruf 1
   instName_2 : myFirstFB;   //Instanz Aufruf 2
END_VAR



//Implementierung im Programm
//Aufruf 1
instName_1(in1:=        ,
           inOut1:=     ,
           out1=>       ,
           out2=>       );

//Aufruf 2
instName_2(in1:=        ,
           inOut1:=     ,
           out1=>       ,
           out2=>       );
```

Bild 36 Instanzierung von zwei FBs