Automation Letter Nr. 26

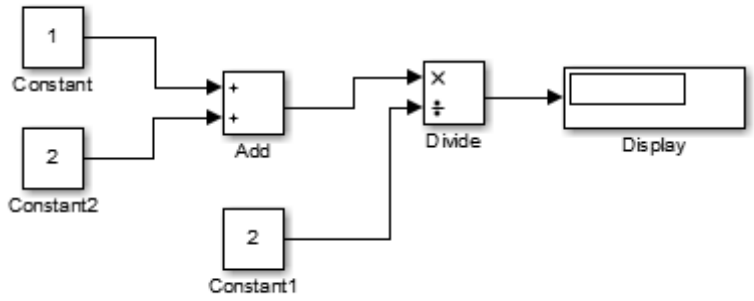
Robert Mille

PLC Coder von MATLAB für PLS Freelance von ABB

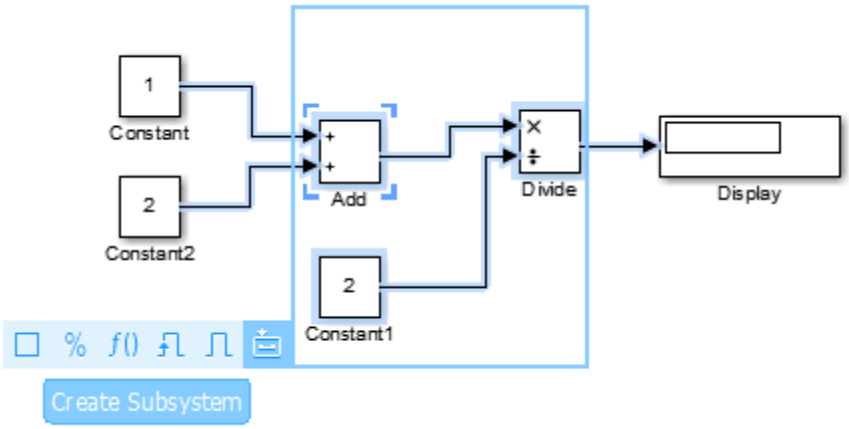
Entwurf eines Freelance-Funktionsbausteins mit strukturiertem Text.

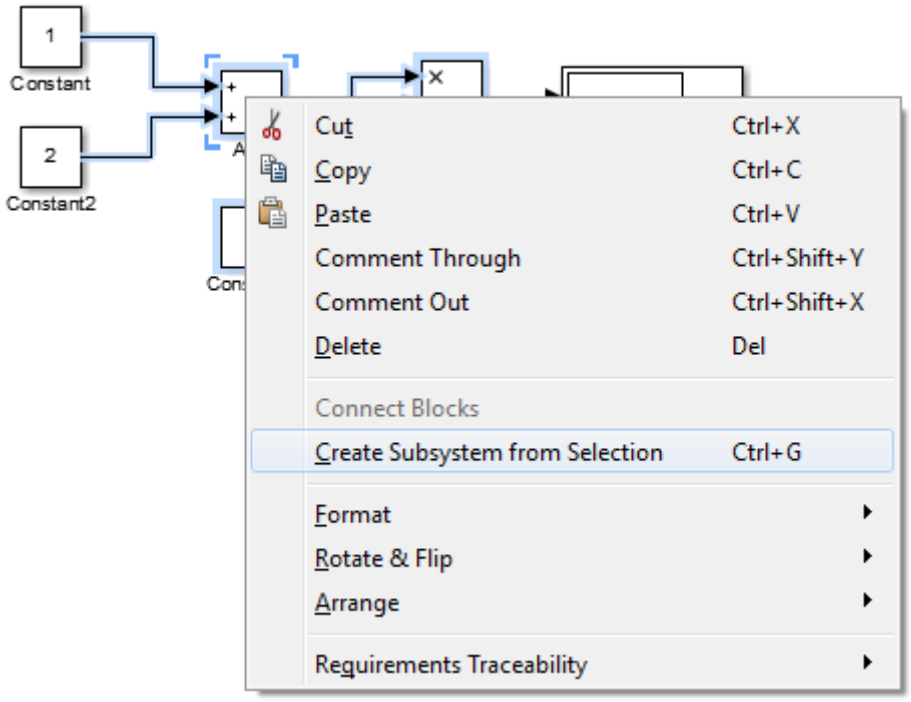
Eine Anleitung zur Verwendung des *PLC Coders* und zum Erstellen eines *Freelance ST Bausteins*. Als Beispiel wird nur ein einfaches Simulink-Modell in ST Code umgewandelt. Die vorliegende Arbeit wurde von Dr. S. Zacher betreut.

1. Verwendung des MATLAB PLC Coders

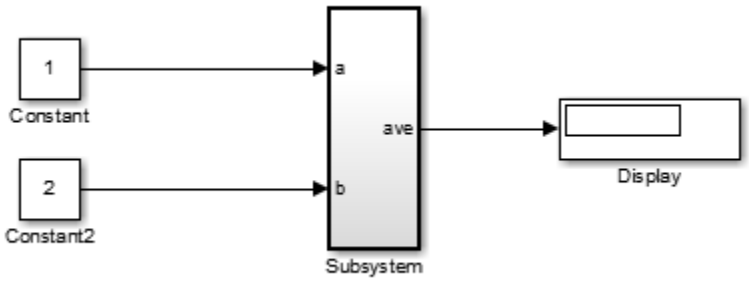


Zunächst erstellt man ein *Subsystem*, indem man die entsprechenden Bausteine in Simulink markiert und auf *Create Subsystem* klickt.

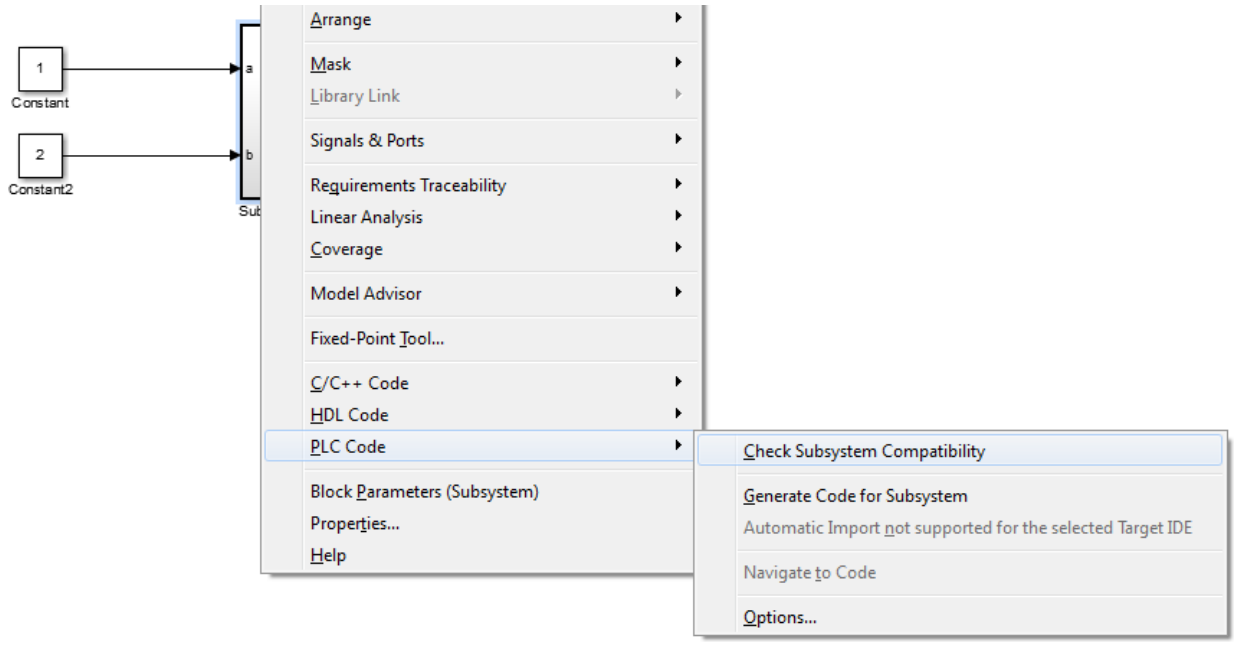




Die markierten Bausteine werden zu einem Subsystem mit Ein- und Ausgängen zusammengefasst.

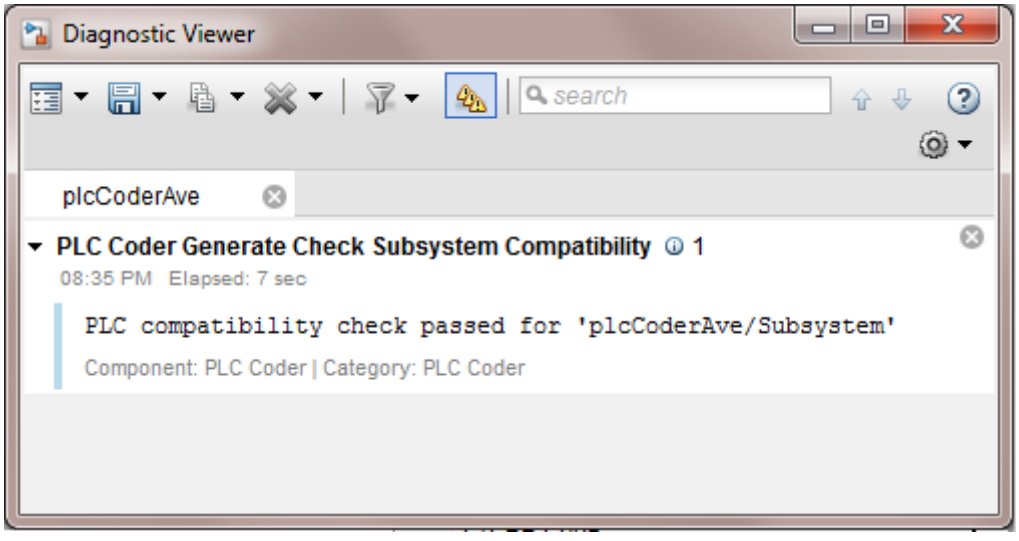


Im Anschluss wählt man das *Subsystem* mit Rechtsklick aus und klickt danach auf *Check Subsystem Compatibility*, um die Möglichkeit der Umwandlung in Programmcode zu prüfen.

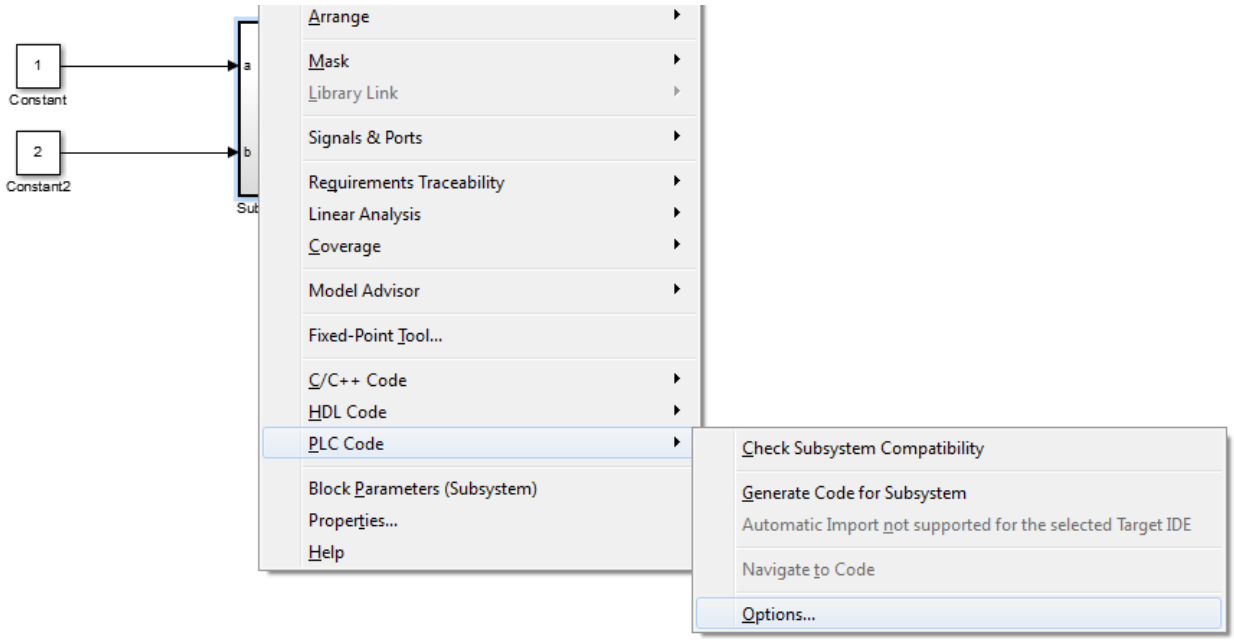


Der *Diagnostik Viewer* zeigt das Ergebnis des Prüfvorgangs. Verwendet man einen Simulink Baustein im Bildbereich, so muss dieser Baustein zunächst in den diskreten z-Bereich transformiert werden. Hierzu kann man den *Model Discretizer* unterstützend verwenden:

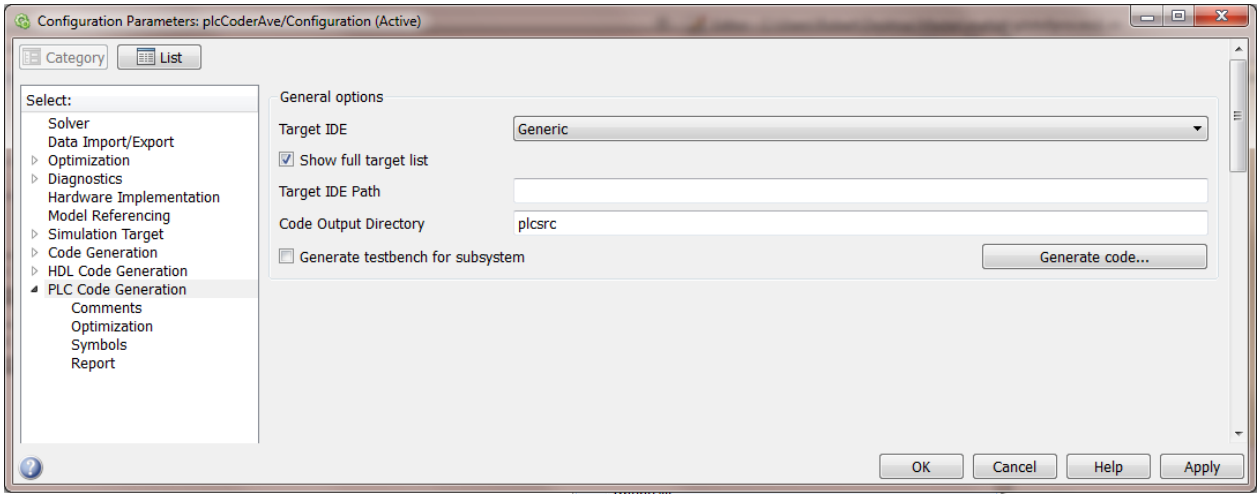
http://www.zacher-international.com/Automation_Letters/05_Model_Discretizer.pdf

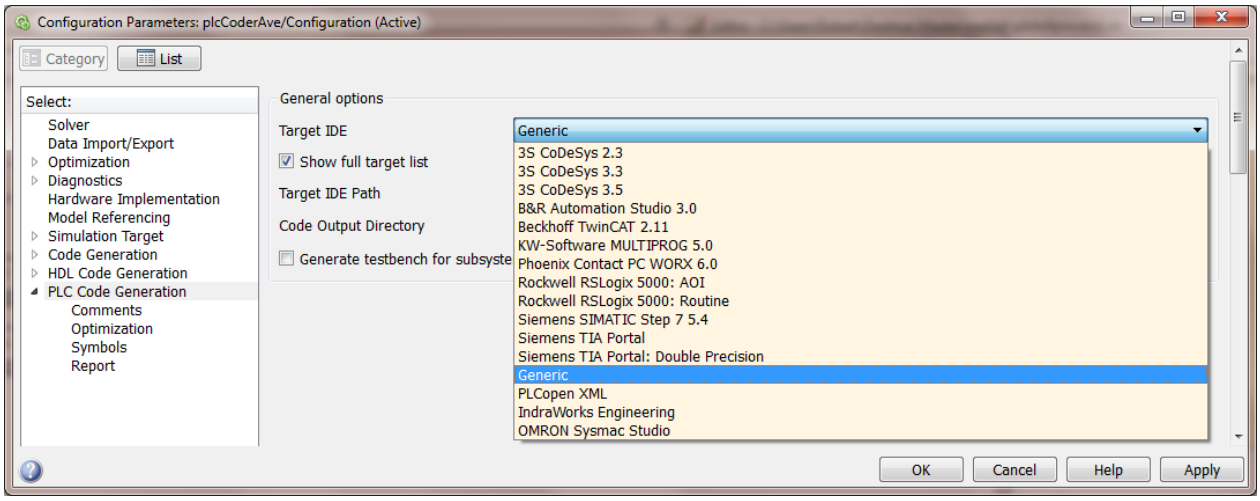


Nach dem Bestehen des Tests „*PLC Compatibility check passed for [subsystem]*" wählt man das Subsystem mit Rechtsklick aus und wählt unter PLC Code die Optionen aus.



In den Optionen setzt man den Haken bei *Show full target list*, um alle möglichen Ziel-IDEs in der Dropdown Liste anzeigen zu können. In diesem Beispiel wird die *Target IDE Generic* für eine Umwandlung in Standard ST Code ausgewählt.





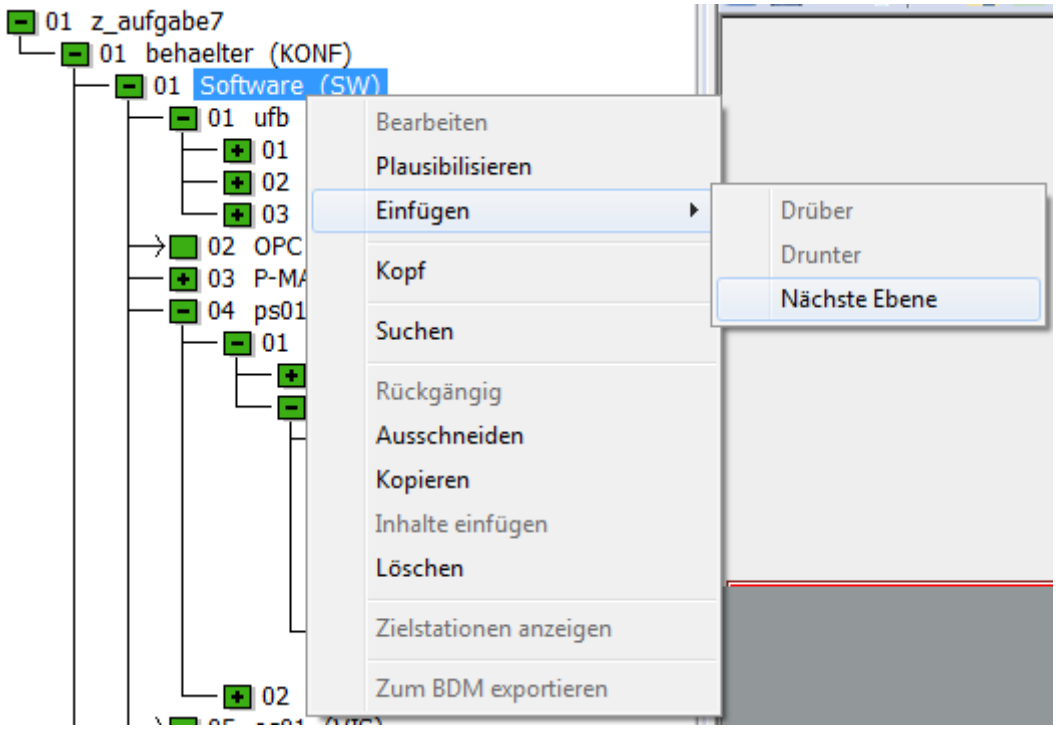
Danach klickt man auf *Generate Code*. Der Code für das obenstehende Beispiel ist unten gezeigt:.

```
FUNCTION_BLOCK Subsystem
VAR_INPUT
    a: LREAL;
    b: LREAL;
END_VAR
VAR_OUTPUT
    ave: LREAL;
END_VAR
VAR
END_VAR
VAR_TEMP
END_VAR
(* Outport: '<Root>/ave' incorporates:
 *  Constant: '<S1>/Constant1'
 *  Inport: '<Root>/a'
 *  Inport: '<Root>/b'
 *  Product: '<S1>/Divide'
 *  Sum: '<S1>/Add' *)
ave := (a + b) / 2.0;
END_FUNCTION_BLOCK
VAR_GLOBAL CONSTANT
END_VAR
VAR_GLOBAL
END_VAR
```

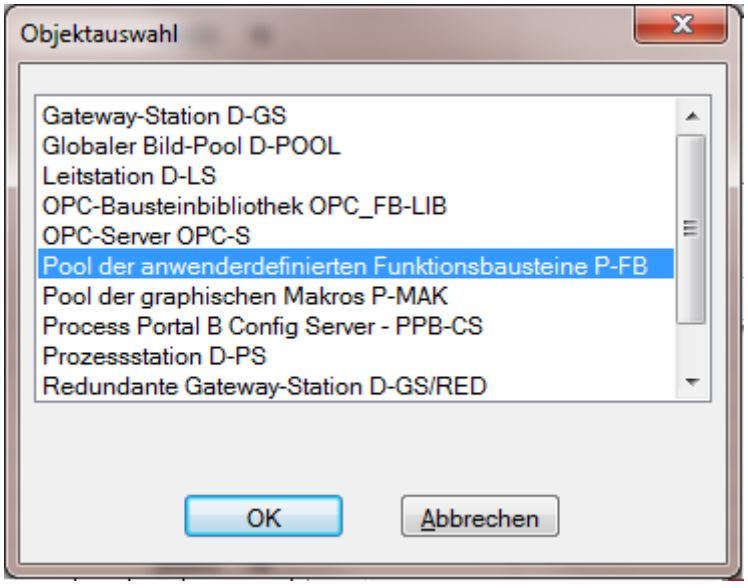
Bei der Umwandlung werden Kommentare und Variablenbereiche im Programmcode eingefügt. Diese werden vor dem Import in Freelance entfernt.

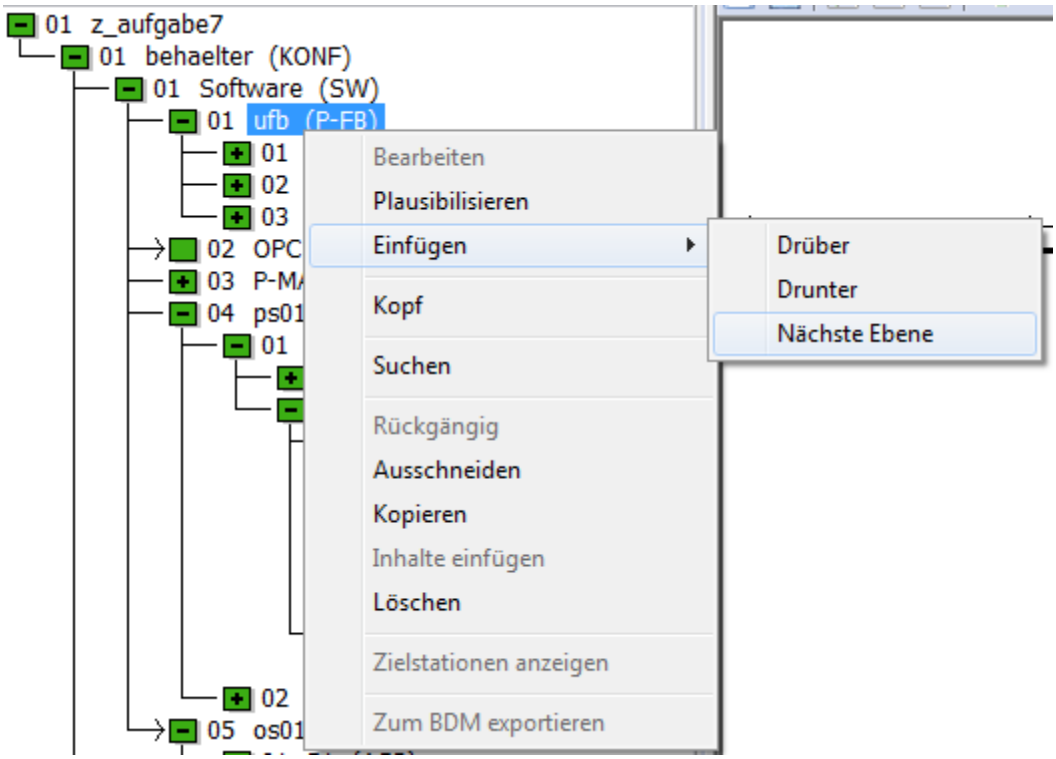
```
FUNCTION_BLOCK Subsystem

VAR_INPUT
    a: LREAL;
    b: LREAL;
END_VAR

VAR_OUTPUT
    ave: LREAL;
END_VAR

        ave := (a + b) / 2.0;

END_FUNCTION_BLOCK
```

2. Erstellen eines *Freelance*-Funktionsbausteins mit strukturiertem Text

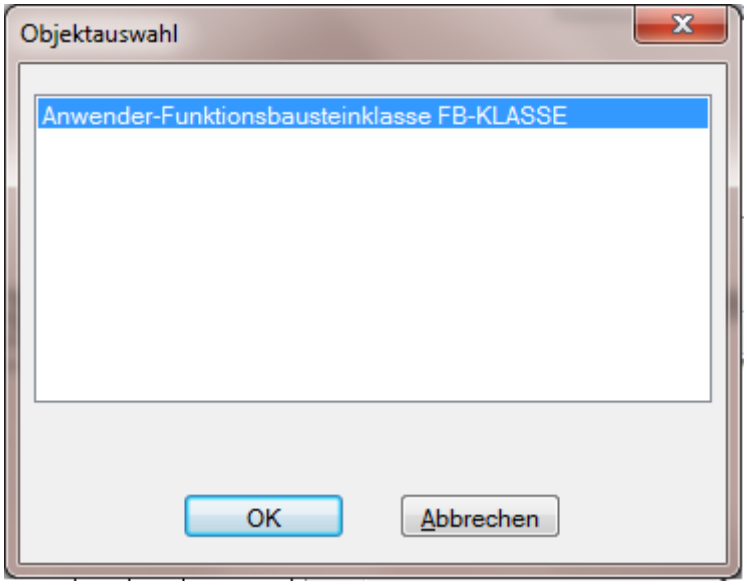


Unterhalb des Software (SW) Blocks muss in einer neuen Ebene ein Pool der anwenderdefinierten Funktionsbausteine P-FB eingefügt werden.

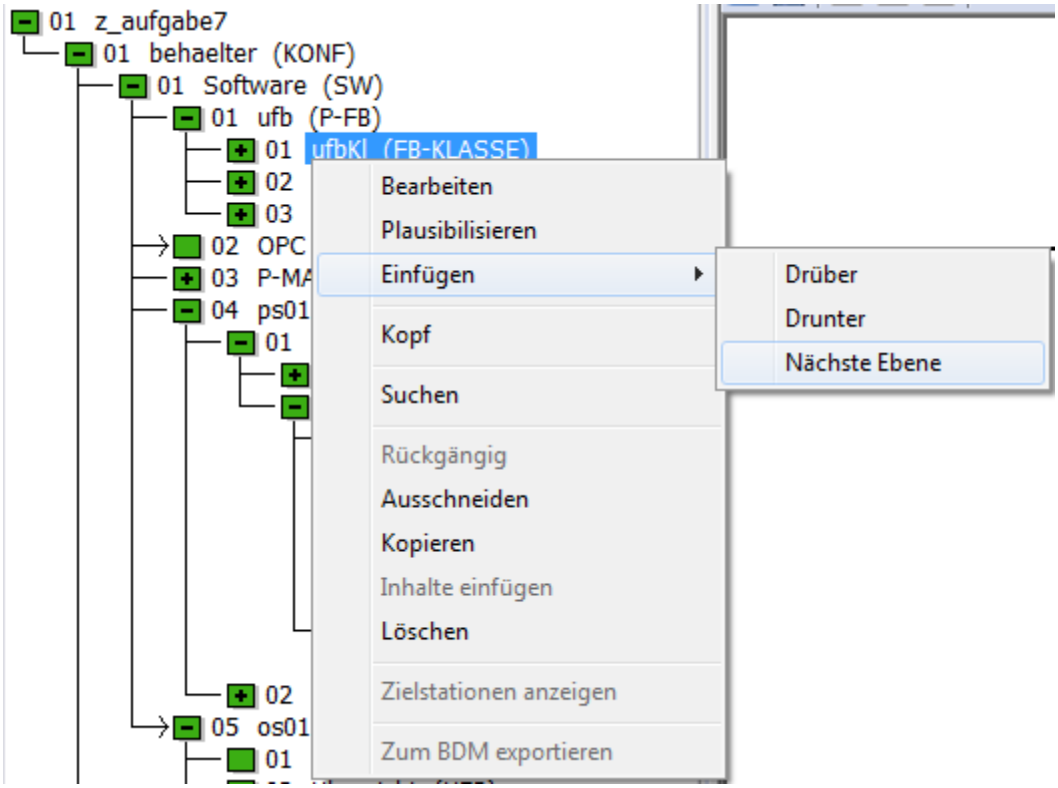




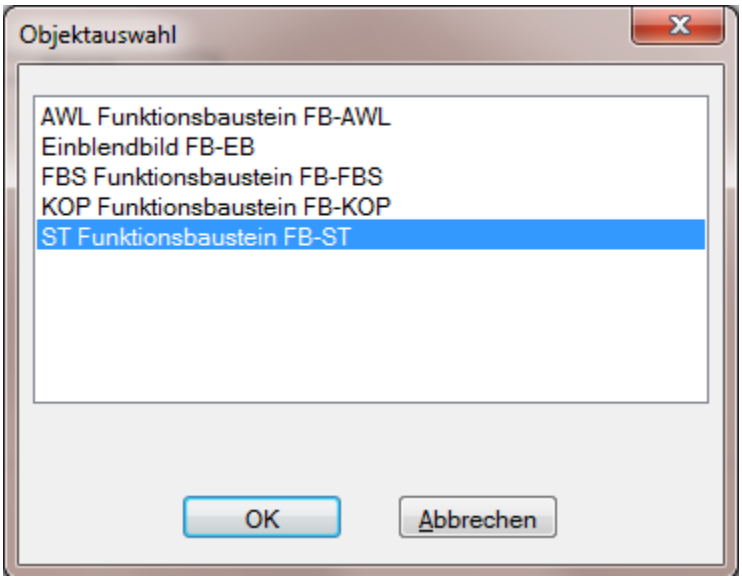
Unterhalb des neuen Pools (P-UB) muss in einer neuen Ebene eine Anwender-Funktionsbausteinklasse FB-KLASSE eingefügt werden. Diese Klasse repräsentiert einen Funktionsbaustein und kann mehrfach instanziiert werden. Werden später mehrere Bausteine erstellt, muss jeweils eine FB-KLASSE angelegt werden.



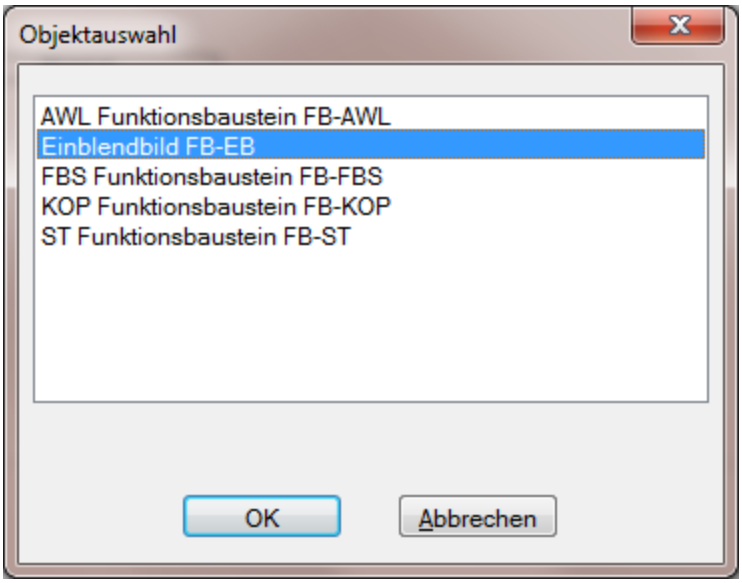
Unterhalb des neuen Pools (P-UB) muss in einer neuen Ebene ein Pool der anwenderdefinierten Funktionsbausteine P-FB eingefügt werden.



Unterhalb der neuen (FB-KLASSE) müssen sowohl ein ST Funktionsbaustein FB-ST Objekt als auch ein Einblendbild FB-EB eingefügt werden.



Beim Doppelklick auf das ST-Programm kann man den Programmcode bearbeiten. Dort fügt man das Hauptprogramm ein. In diesem Beispiel wird die Zuweisung ave := (a + b) / 2.0; eingefügt. Die Variablendeklaration ist ausgegraut und muss separat bearbeitet werden (s. unten).



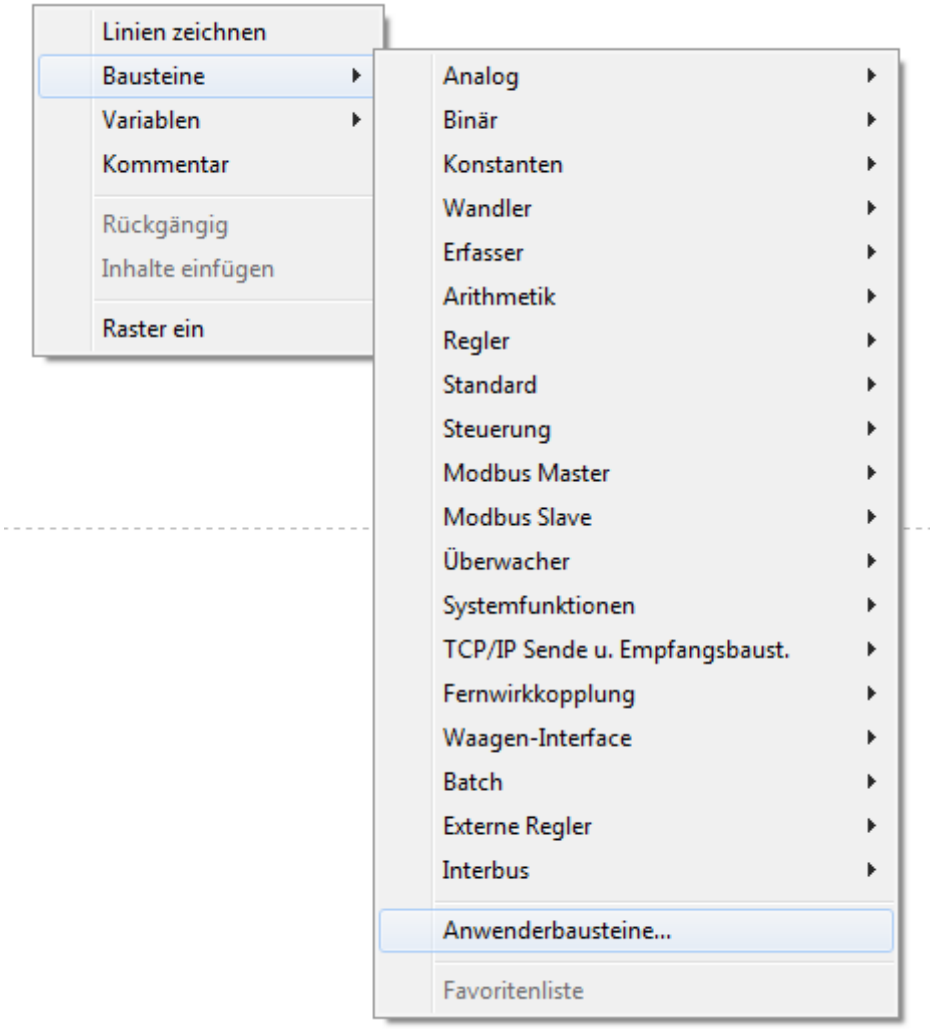
Beim Doppelklick auf das eingefügte Einblendbild kann man die Größe des Bausteins bestimmen. Weitere Einstellungen müssen hier nicht vorgenommen werden.

Name	Datentyp	Speichertyp	Initialwer	Min. Wert	Max. Wert	Ref-Parameter
a	REAL	VAR_IN				
b	REAL	VAR_IN				
ave	REAL	VAR_OUT				
ClassName	TEXT	PARA_VIS				
TagName	TEXT	PARA_VIS				
ShortText	TEXT	PARA_VIS				
LongText	TEXT	PARA_VIS				
SelState	BOOL	PARA_VIS				

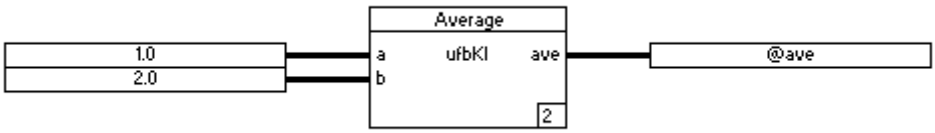
Als nächstes navigiert man mit einem Doppelklick auf die (FB-KLASSE) in die Variablendeklaration. Die Variablendeklaration in ST in Freelance kann nur über diese Oberfläche erledigt werden. In diesem Beispiel fügt man die Variablen a, b, und ave mit entsprechendem Datentyp und dem Speichertyp an. VAR_IN verwendet man für Eingangsvariablen, VAR_OUT verwendet man für Ausgangsvariablen. Lokale Variablen können mit dem Speichertyp VAR_DPS angelegt werden. Sie werden jedoch in diesem Beispiel nicht benötigt.

```
FUNCTION_BLOCK ufbKlST

VAR_INPUT
    a : REAL;
    b : REAL;
END_VAR

VAR_OUTPUT
    ave : REAL;
END_VAR

(*
PARA_VIS
    ClassName : TEXT;
    TagName : TEXT;
    ShortText : TEXT;
    LongText : TEXT;
    SelState : BOOL;
END_VAR
*)

ave := (a + b) / 2.0;

END_FUNCTION_BLOCK
```

Das fertige ST-Programm ist oben dargestellt.



Den neu erstellten Baustein kann man unter den Anwenderbausteinen finden und in ein FBS Blatt einfügen.



Zum Schluss verbindet man die Ein- und Ausgänge mit Variablen.