

TwinCAT 3: eXtended Automation



- IEC 61131-3 编程标准
第三版
- 详细介绍
- OOP编程的应用



- 1993年第一版
- 2003年第二版：错误修正和内容扩充
- 2011年第三版
 - 2008年着手编辑
 - 2009年7月第一份草案
 - 2010年发布

- LINT, ULINT, LWORD
 - 64位

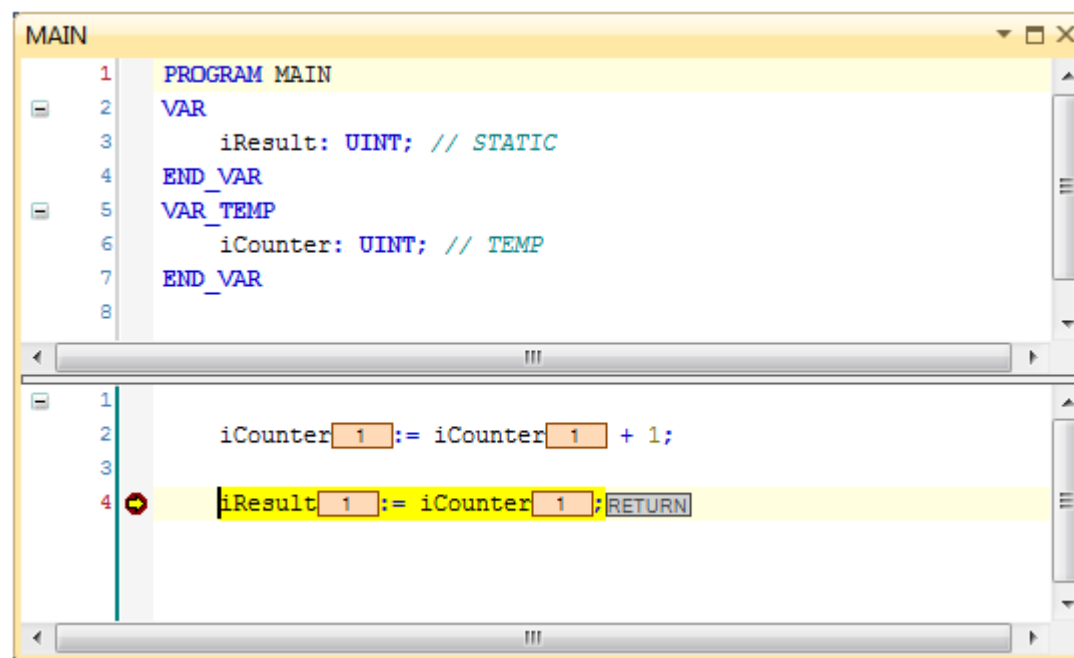
- LTIME, LDATE, LDATE_AND_TIME, LTIME_OF_DAY
 - tTime:= LTIME#12s34ms2us44ns
 - DT:= CONCAT_DATE_TOD(DATE, TOD);
 - DATE:= CONCAT_DATE(YEAR, MONTH, DAY);

- UNION
 - 多个变量共用同一段内存地址

- WSTRING
 - Unicode编码

```
sChinese: WSTRING:= "答复";
```

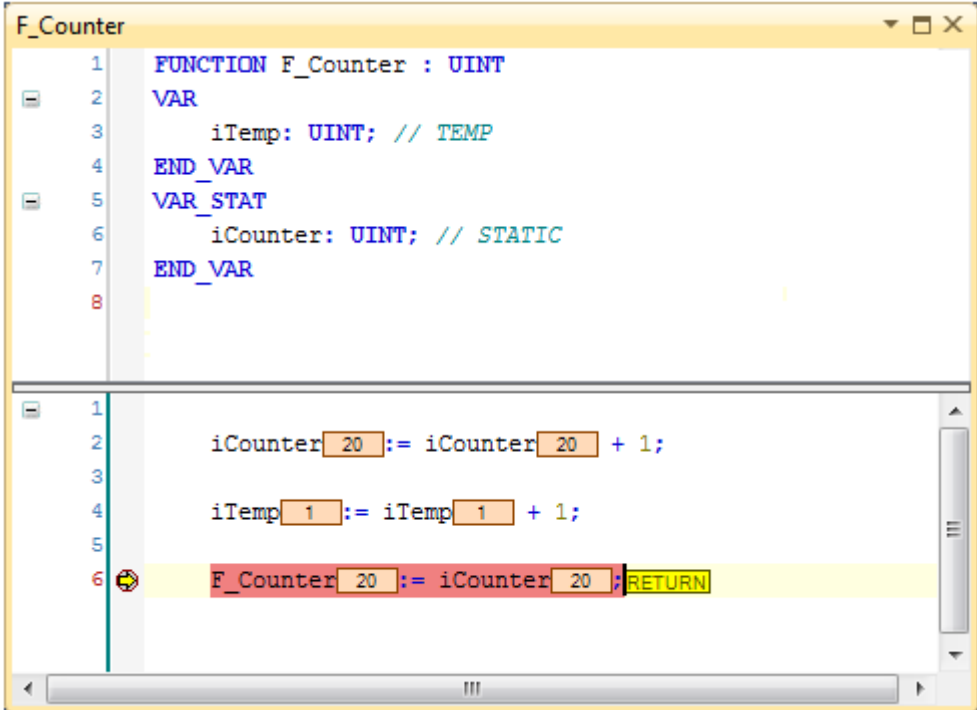
- 临时变量在每次调用POU时都会初始化。
- 用在程序和功能块中。
- 在方法和函数中的内部变量都是临时变量！



The screenshot shows a Beckhoff ladder logic editor window titled 'MAIN'. The editor contains two sections of code. The first section, starting at line 1, defines a program 'PROGRAM MAIN' and declares two variables: 'iResult: UINT; // STATIC' (line 3) and 'iCounter: UINT; // TEMP' (line 6). The second section, starting at line 1, shows the logic for these variables. Line 2 contains the statement 'iCounter[1] := iCounter[1] + 1;'. Line 4 contains the statement 'iResult[1] := iCounter[1]; RETURN', which is highlighted in yellow. The variable names 'iCounter' and 'iResult' are followed by a box containing the number '1', indicating an array or indexed access. The 'RETURN' statement is also highlighted in yellow.

```
1 PROGRAM MAIN
2 VAR
3   iResult: UINT; // STATIC
4 END_VAR
5 VAR_TEMP
6   iCounter: UINT; // TEMP
7 END_VAR
8
9
10 iCounter[1] := iCounter[1] + 1;
11
12 iResult[1] := iCounter[1]; RETURN
```

- 静态变量只在POU的第一次调用时被初始化。
- 静态变量声明赋值后，值不会丢失。
- 可用在方法和函数中。
- 在程序和功能块中声明的变量都是静态的！



The screenshot shows a Beckhoff Ladder Editor window titled 'F_Counter'. The editor is divided into two sections. The top section contains the function declaration and variable declarations:

```
1 FUNCTION F_Counter : UINT
2 VAR
3     iTemp: UINT; // TEMP
4 END_VAR
5 VAR_STAT
6     iCounter: UINT; // STATIC
7 END_VAR
8
```

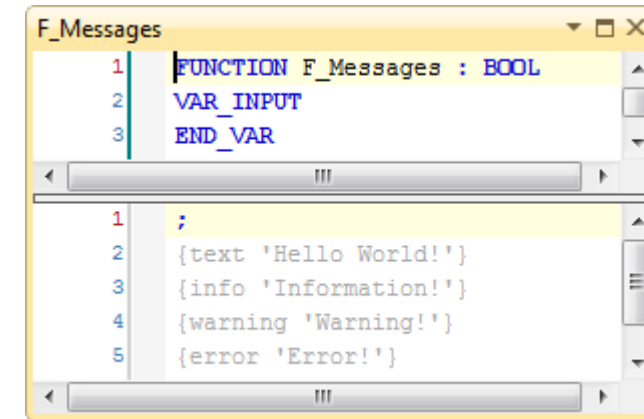
The bottom section shows the function body with three rungs:

```
1
2     iCounter[20] := iCounter[20] + 1;
3
4     iTemp[1] := iTemp[1] + 1;
5
6     F_Counter[20] := iCounter[20]; RETURN
```

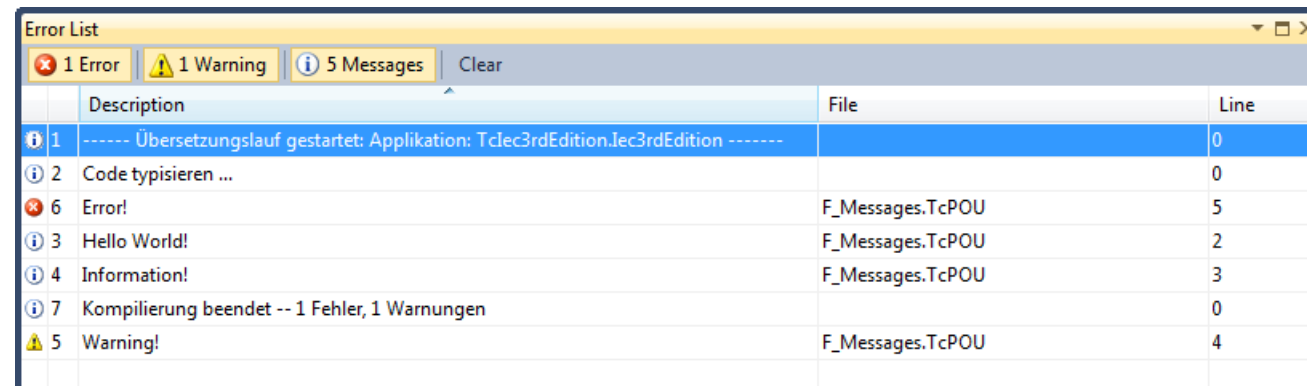
The variable names in the code are color-coded: 'FUNCTION' and 'VAR' are blue, 'END_VAR' is green, 'VAR_STAT' is blue, and 'iCounter' and 'iTemp' are green. The comments '// TEMP' and '// STATIC' are in red. The variable names in the rungs are also color-coded: 'iCounter' and 'iTemp' are green, and 'F_Counter' is blue. The rungs are numbered 1 through 6 on the left side.

■ Messages

- {text 'Hello World!'}
- {info 'Information!'}
- {warning 'Warning!'}
- {error 'Error!'}



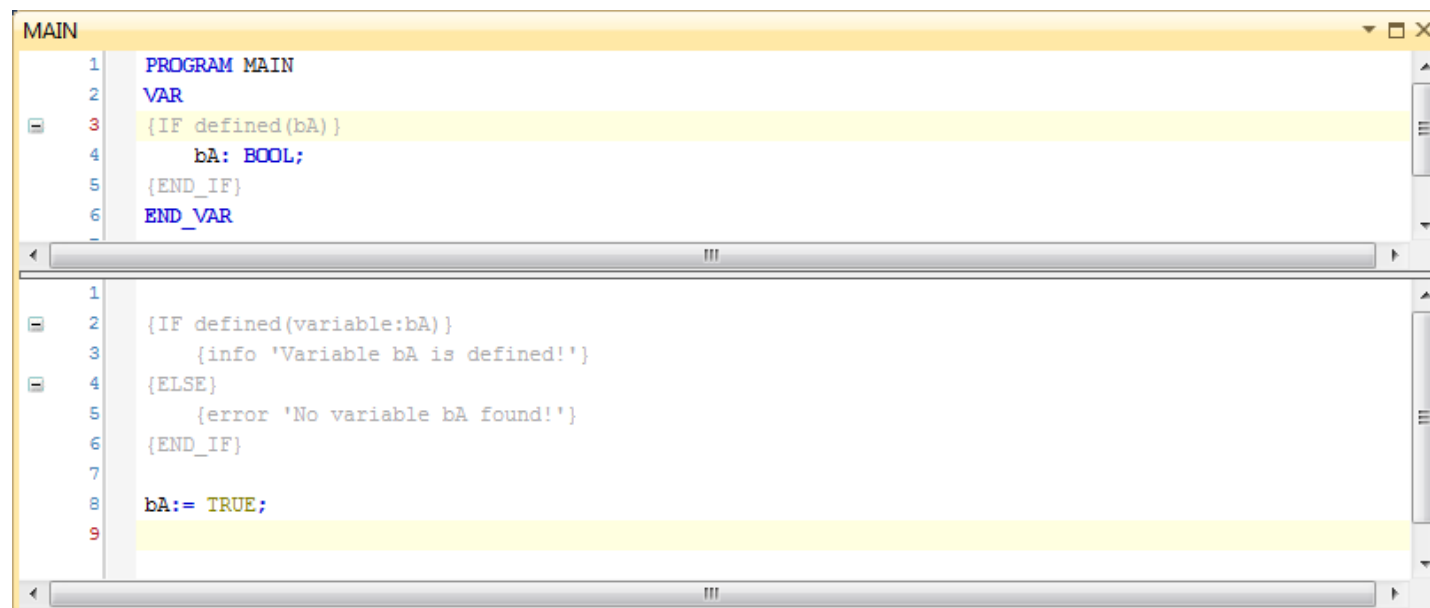
```
1 FUNCTION F_Messages : BOOL
2   VAR_INPUT
3   END_VAR
4
5   ;
6   {text 'Hello World!'}
7   {info 'Information!'}
8   {warning 'Warning!'}
9   {error 'Error!'}
```



	Description	File	Line
1	----- Übersetzungslauf gestartet: Applikation: Tclec3rdEdition.Iec3rdEdition -----		0
2	Code typisieren ...		0
6	Error!	F_Messages.TcPOU	5
3	Hello World!	F_Messages.TcPOU	2
4	Information!	F_Messages.TcPOU	3
7	Kompilierung beendet -- 1 Fehler, 1 Warnungen		0
5	Warning!	F_Messages.TcPOU	4

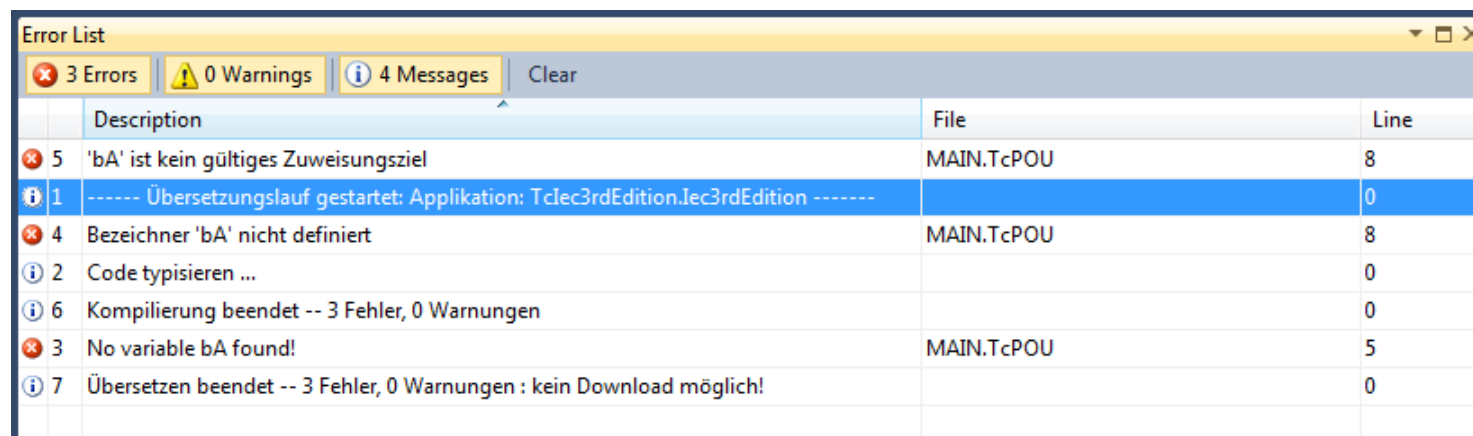
- {defined identifier}
- {undefined identifier}
- {IF expr}
- {ELSIF expr}
- {ELSE}
- {END_IF}

- {IF defined(variable:iInput)}
- {IF defined(pou:CheckBounds)}
- {IF defined(type:ST_DataType)}
- {IF hasattribute(variable:iInput, 'DoSomething')}
- {IF hastype(variable:iInput, UINT)}



```
1 PROGRAM MAIN
2 VAR
3 {IF defined(bA)}
4   bA: BOOL;
5 {END_IF}
6 END_VAR

1
2 {IF defined(variable;bA)}
3   {info 'Variable bA is defined!'}
4 {ELSE}
5   {error 'No variable bA found!'}
6 {END_IF}
7
8 bA:= TRUE;
9
```



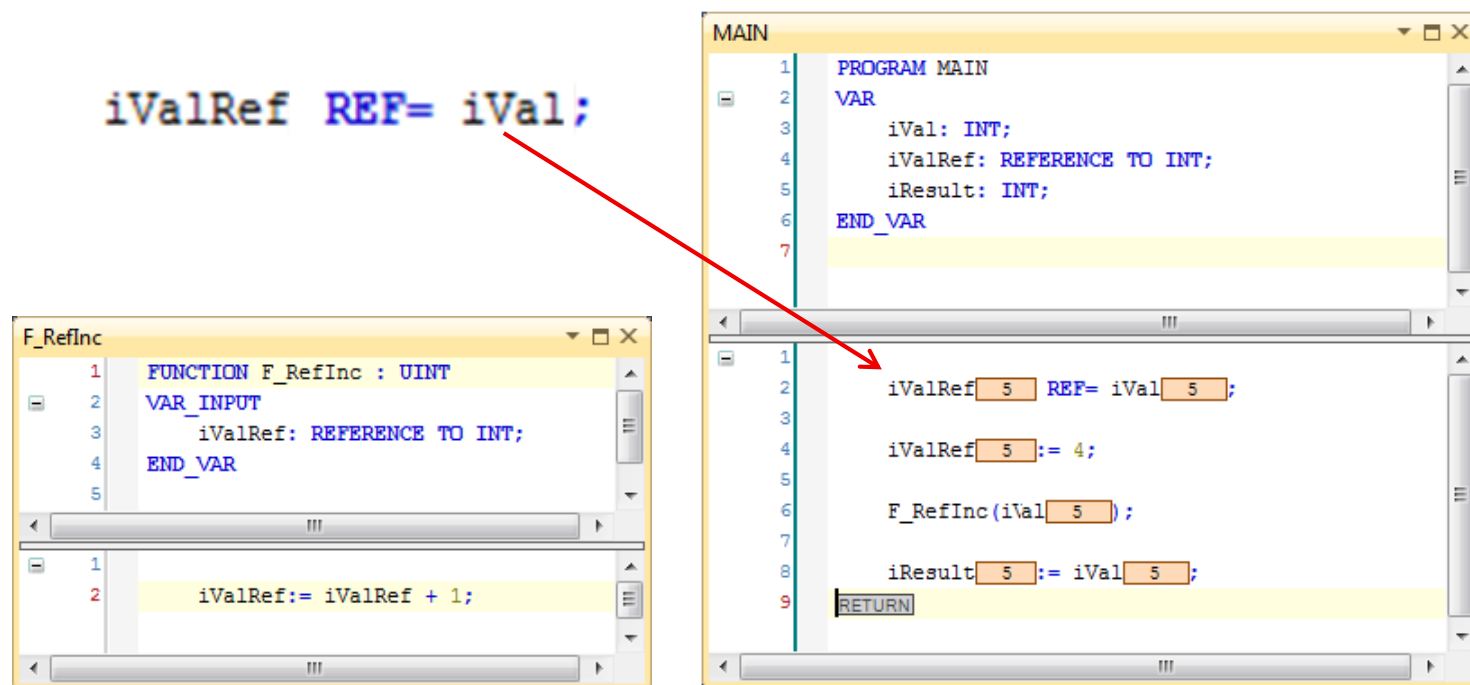
Error List			
3 Errors 0 Warnings 4 Messages Clear			
	Description	File	Line
5	'bA' ist kein gültiges Zuweisungsziel	MAIN.TcPOU	8
1	----- Übersetzungslauf gestartet: Applikation: Tlc3rdEdition.Iec3rdEdition -----		0
4	Bezeichner 'bA' nicht definiert	MAIN.TcPOU	8
2	Code typisieren ...		0
6	Kompilierung beendet -- 3 Fehler, 0 Warnungen		0
3	No variable bA found!	MAIN.TcPOU	5
7	Übersetzen beendet -- 3 Fehler, 0 Warnungen : kein Download möglich!		0

第三类：属性编译，需要我们添加属性

属性种类很多，具体都可以在帮助文档找到（搜索Pragmas）

如果想简单了解一下，可以看OOP的教学视频（8：09开始简单讲了一个hide隐藏属性）

- 引用类型的变量可以看做是一个对象的别名。
- 作用与指针型变量一样,可以代替指针型变量。



- TwinCAT 3是多功能的PLC开发平台（可以在线调试，修改等）
- 将面向对象的思想融入到PLC程序中（可以很简单地对已有项目进行升级改造）
- OOP并不是一种新的PLC编程语言，而是对IEC标准语言的扩充。
- OOP适用于所有IEC语言（不仅仅是ST）
- IEC标准组织已经将OOP融入IEC61131-3标准中。
- IEC61131-3第三版，2011年成为一个编程标准。

- IEC 61131-3 编程标准
第三版
- 详细内容介绍
- OOP编程的应用



关键词:

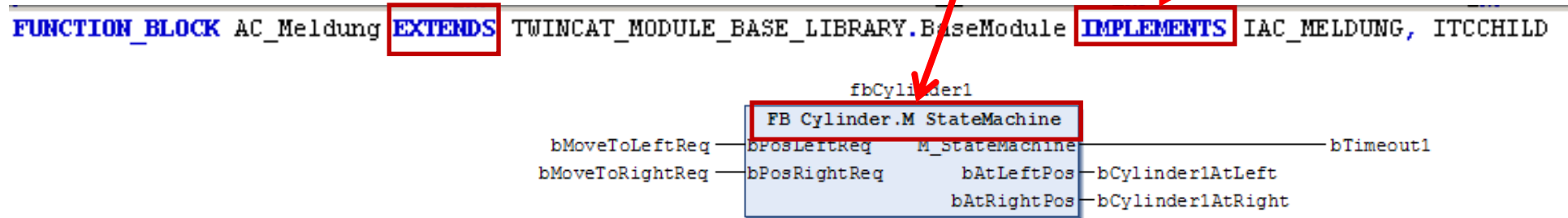
- **METHOD**: 方法，带自定义变量的**FB action**。
- **EXTENDS**: 继承，获得继承对象的方法，变量，属性等。继承的对象可以是**FB**或结构类型变量
- **THIS**: 一种指针，引用当前功能块的方法、变量或属性。
- **SUPER**: 一种指针，引用父类功能块的方法、变量或属性。
- **INTERFACE**: 接口，一种复杂的结构类型，包含变量，方法，属性。（类似于没有代码的**FB**）
- **IMPLEMENTS in the FB**: 功能块引用一个指定的接口类型。
- **PROPERTY**: 属性，使用**Set/Get**方法来访问变量的一种方式。

方法的调用:

- *对象.方法名 (...)*

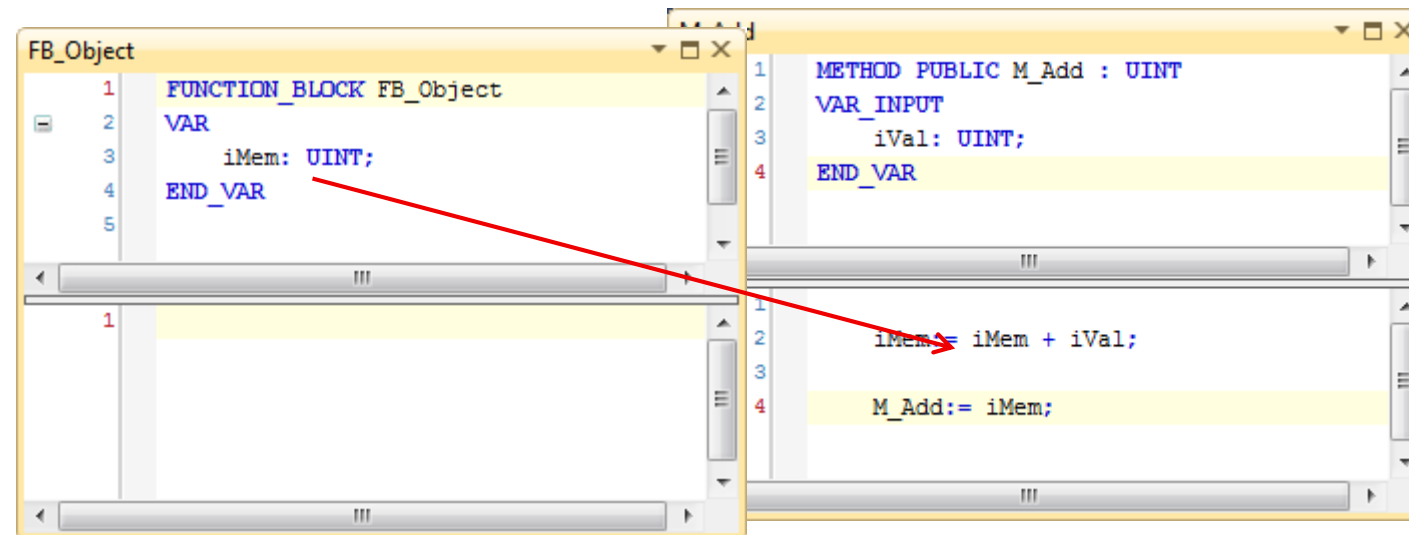
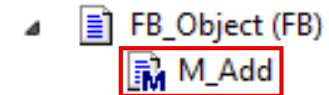
IEC 61131-3第三版中面向对象增加的概念：

- 概念：
 - Classes
 - Interfaces
 - Methods
 - Inheritance
 - Attributes
 - Keywords such as THIS, SUPER

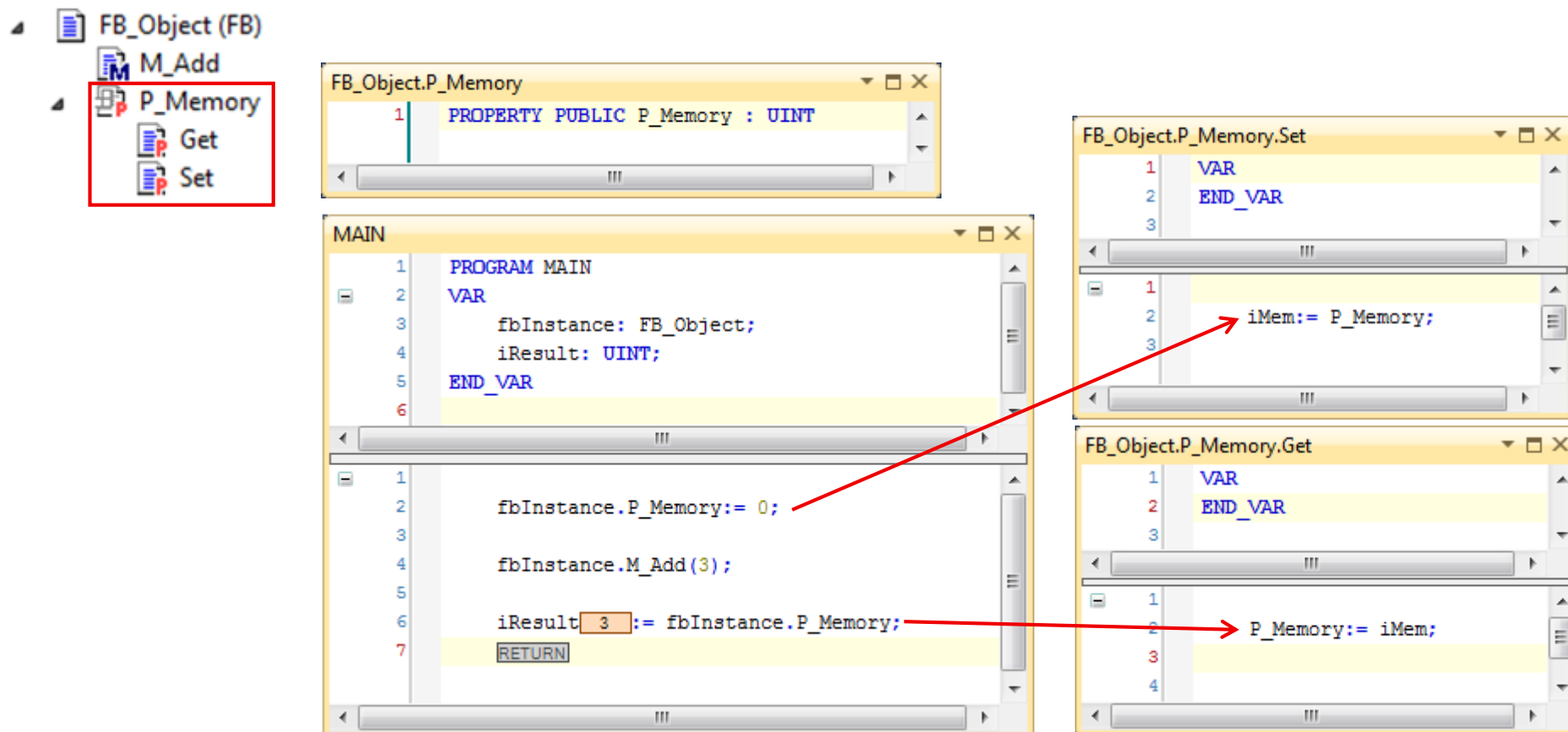


- 扩展的功能
 - 适用于所有的IEC语言
 - 与硬件无关
 - 可以不使用这些扩展的功能

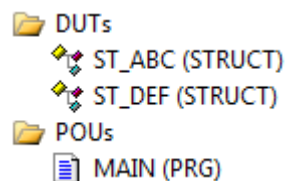
- 方法描述了一系列的动作指令,实现目的功能。
- 类似于函数,依附功能块。
- 将功能块的ACTION扩展以下内容。
 - VAR_INPUT, VAR_IN_OUT and VAR_OUTPUT
- 可以访问所属功能块中的变量。



- 访问功能块内部变量的状态或值。
- 属性提供了Get和Set方法对目标进行读取和设置。



- 结构体包含了扩展前的所有变量。



```
1 PROGRAM MAIN
2 VAR
3   stDEF: ST_DEF;
4 END_VAR
5
```

stDEF.

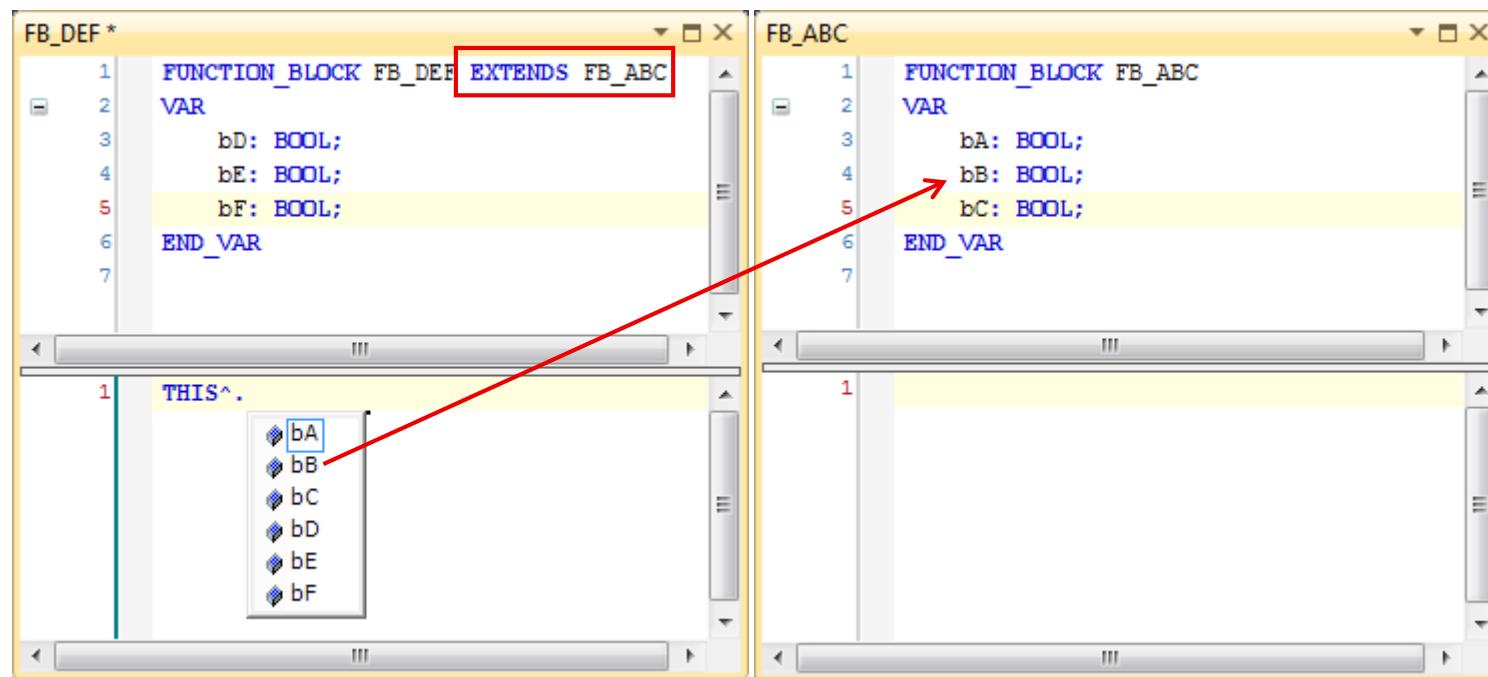
- bA
- bB
- bC
- bD
- bE
- bF

```
1 TYPE ST_ABC :
2   STRUCT
3     bA: BOOL;
4     bB: BOOL;
5     bC: BOOL;
6   END_STRUCT
7 END_TYPE
8
```

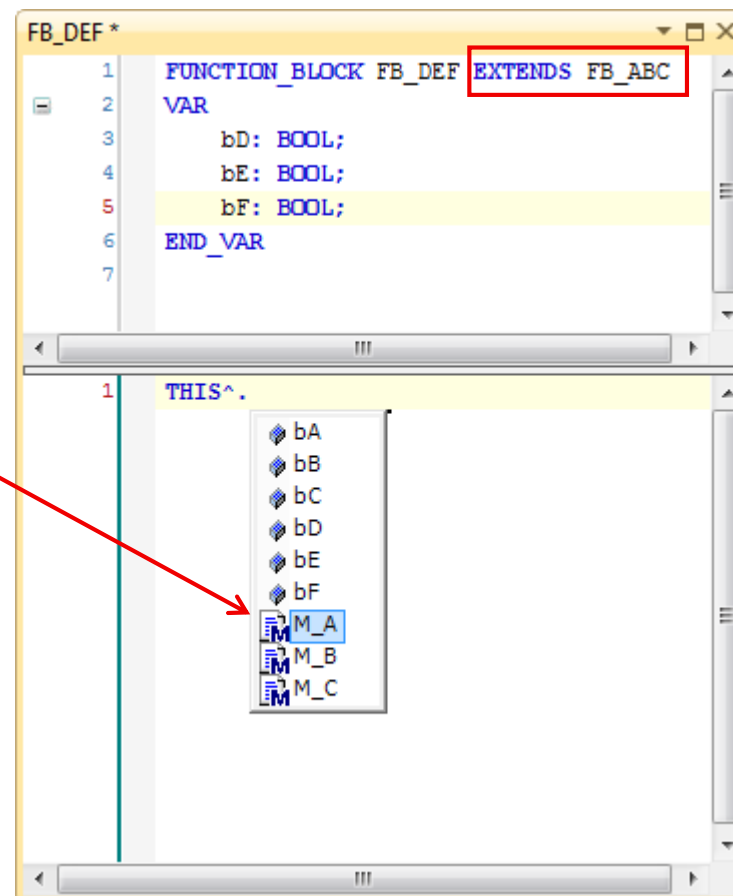
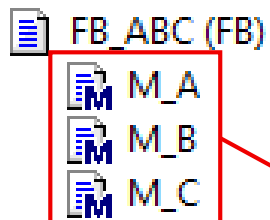
```
1 TYPE ST_DEF EXTENDS ST_ABC :
2   STRUCT
3     bD: BOOL;
4     bE: BOOL;
5     bF: BOOL;
6   END_STRUCT
7 END_TYPE
8
```

- 功能块包括了扩展前的所有变量。

FB_ABC (FB)



- 功能块包括了扩展前的所有方法。



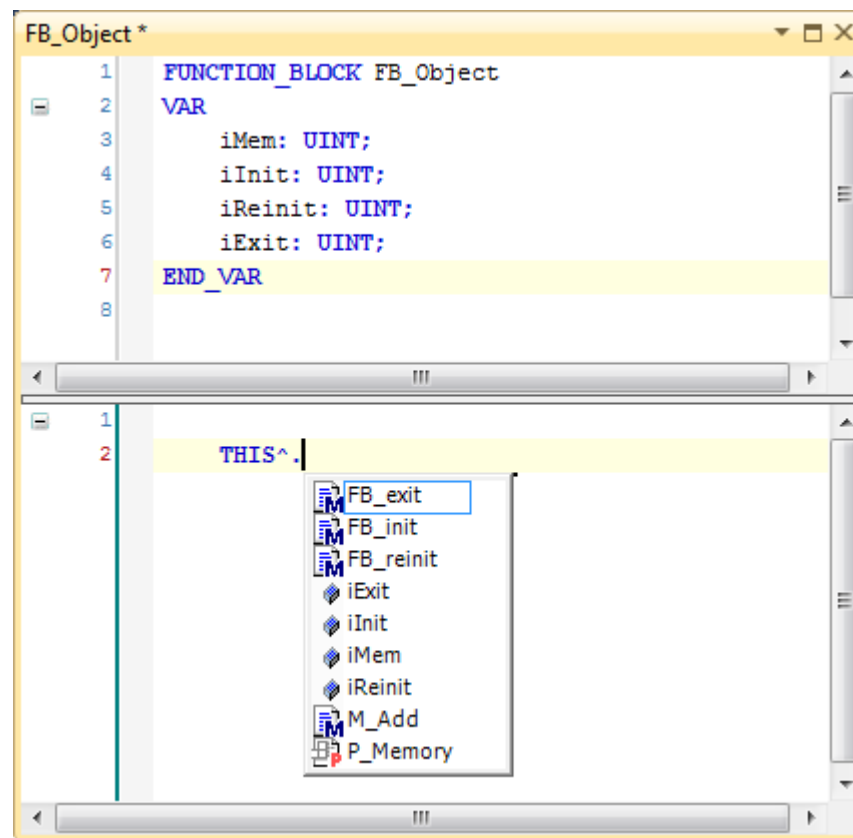
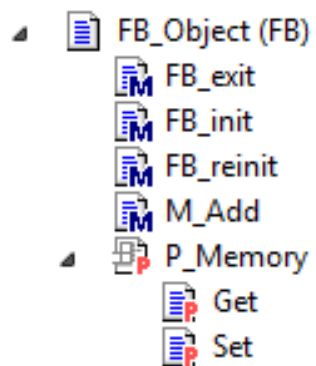
- SUPER 一种指针，引用父类功能块的方法、变量或属性。
- THIS 一种指针，引用当前功能块的方法、变量或属性。

```
FB_1
1  FUNCTION_BLOCK FB_1 EXTENDS FB_Base
2  VAR_INPUT
3      bLocal: BOOL;
4  END_VAR
5  VAR_OUTPUT
6      iBase: INT;
7  END_VAR
8  VAR
9  END_VAR
10
11 IF bLocal THEN
12     //Aufruf der Methoden aus FB_1
13     THIS^.METH_DoIt();
14     THIS^.METH_DoAlso();
15     iBase:=iCnt;
16 ELSE
17     //Aufruf der Methoden aus FB_Base
18     SUPER^.METH_DoIt();
19     SUPER^.METH_DoAlso();
20     iBase:=SUPER^.iCnt;
21 END_IF
```

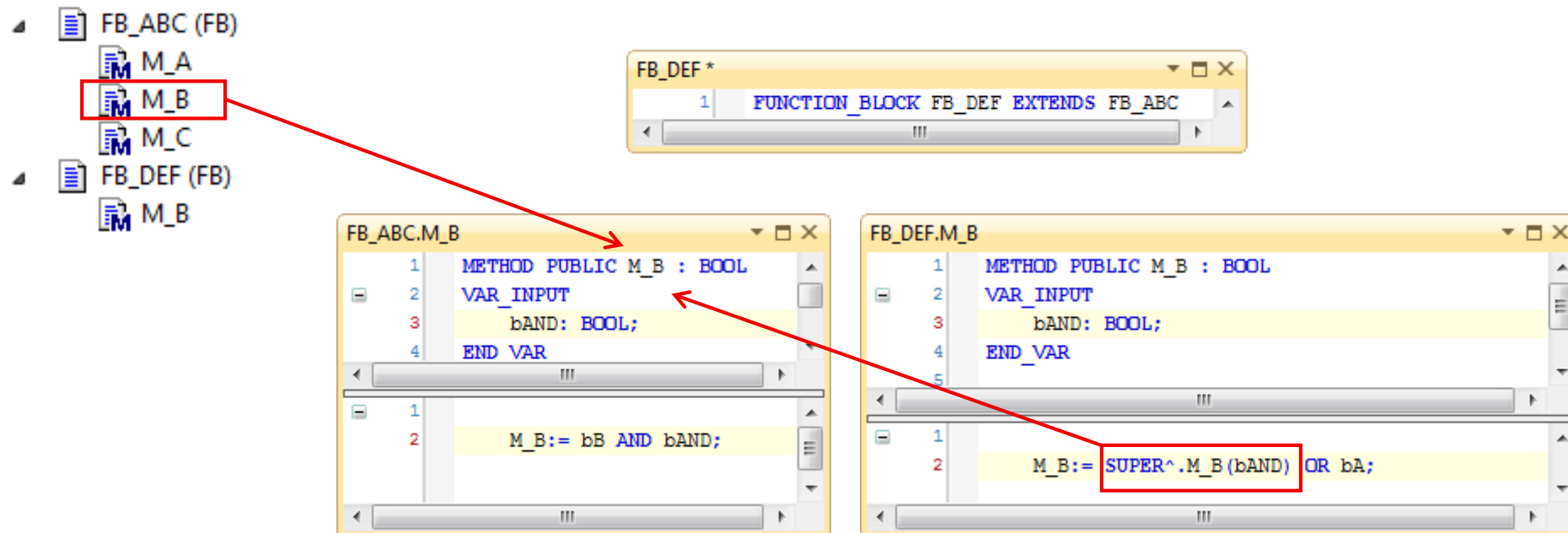
```
FB_Base.METH_DoAlso
1  METHOD PUBLIC METH_DoAlso : BOOL
2  VAR_INPUT
3  END_VAR
4
5  METH_DoAlso:=TRUE;
```

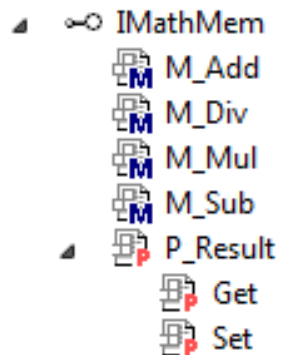
```
FB_Base.METH_DoIt
1  METHOD PUBLIC METH_DoIt : BOOL
2  VAR_INPUT
3  END_VAR
4
5  iCnt:=iCnt+1;
```

- 每个功能块都提供指针THIS。
- THIS指向当前功能块。

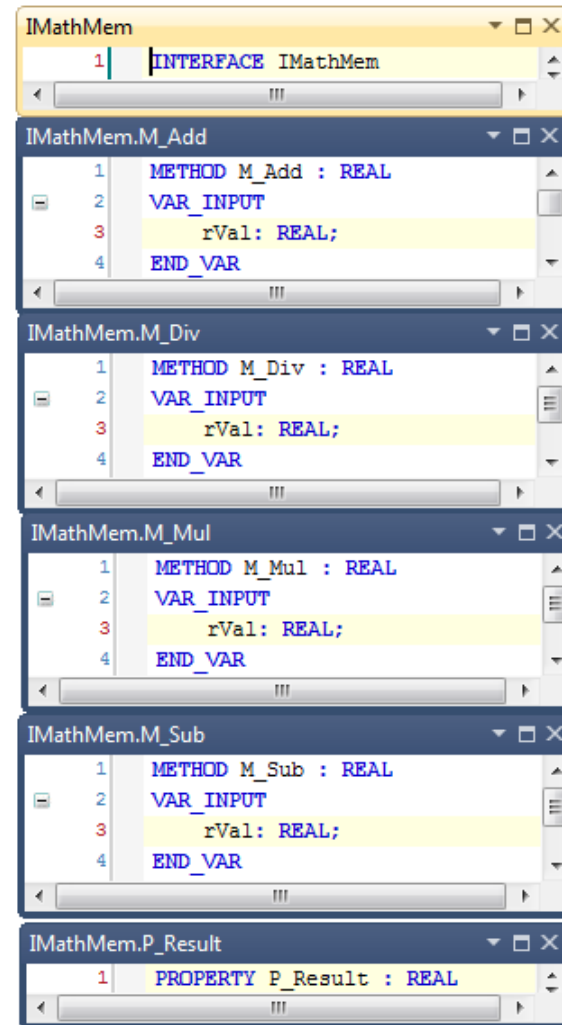


- 每一个功能块都提供了指针SUPER
- SUPER指向原有的功能块
 - 用原有的方法覆盖现有的方法
 - 改变了现有功能块的行为

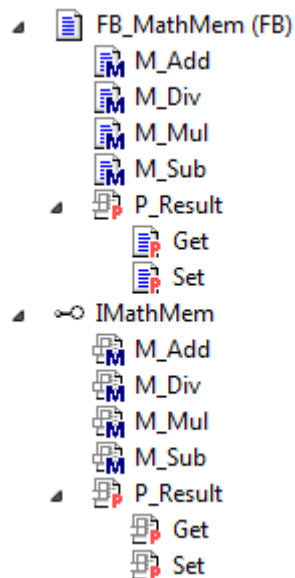




- 一种POU可以引用的复杂结构类型，包含变量、方法和属性。
- 只是声明，没有实际代码
- 只有输入，输出和输入输出变量
- 没有内部变量。



- 声明一个新的功能块去引用接口。



The screenshots show the following code:

```
FB_MathMem
1 FUNCTION_BLOCK FB_MathMem IMPLEMENTS IMathMem
2 VAR
3   rMem: REAL;
4 END_VAR

FB_MathMem.M_Add
1 METHOD M_Add : REAL
2 VAR_INPUT
3   rVal : REAL;
4 END_VAR
5
6   rMem := rMem + rVal;
7
8   M_Add := rMem;

FB_MathMem.M_Mul
1 METHOD M_Mul : REAL
2 VAR_INPUT
3   rVal : REAL;
4 END_VAR
5
6   rMem := rMem * rVal;
7
8   M_Add := rMem;

FB_MathMem.P_Result.Get *
1 VAR
2 END_VAR
3
4   P_Result := rMem;
```

- 实例化功能块，引用该接口。
- 声明接口的变量。
- 将接口指向功能块。

The screenshot displays the Beckhoff IEC 61131-3 editor interface. On the left, a project tree shows the structure of the program, including a function block 'FB_MathMem (FB)' and an interface 'IMathMem'. A red arrow points from the 'ipInterface' variable in the variable declaration table to the assignment statement in the ladder logic program.

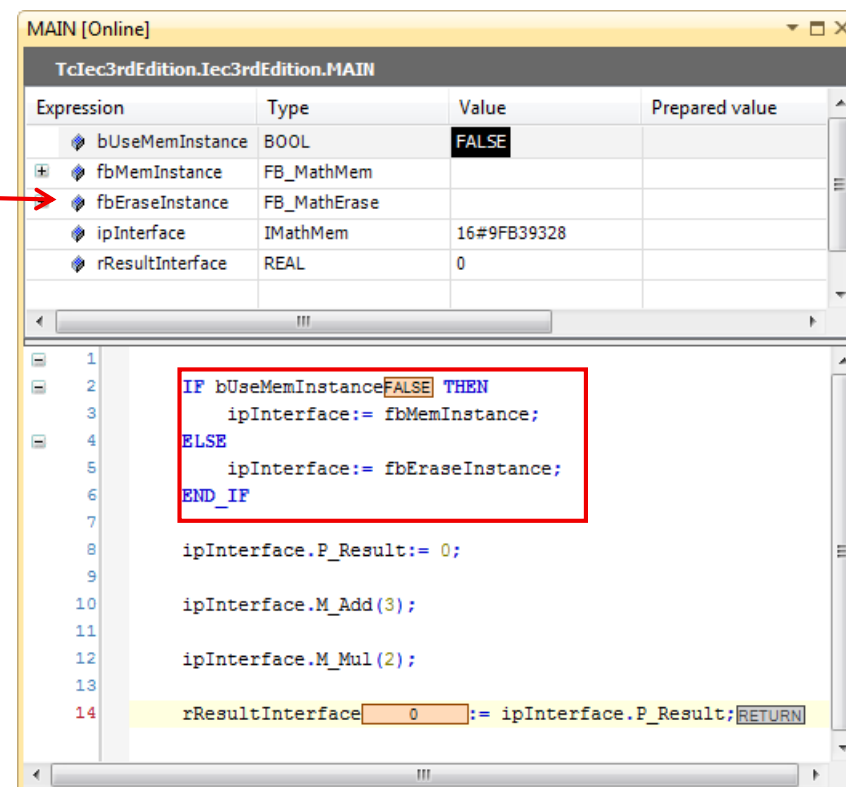
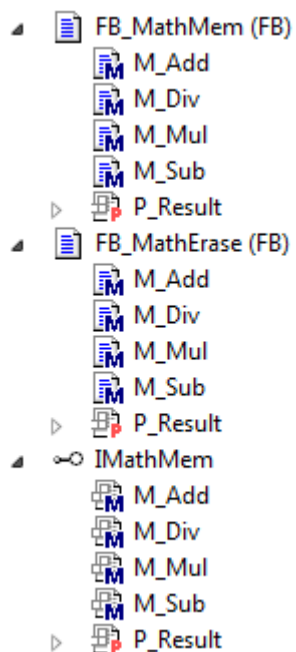
Variable Declaration Table:

Expression	Type	Value	Prepared value	Comm
fbInstance	FB_MathMem			
ipInterface	IMathMem	16#9FB392E8		
rResultInstance	REAL	6		
rResultInterface	REAL	6		

Ladder Logic Program:

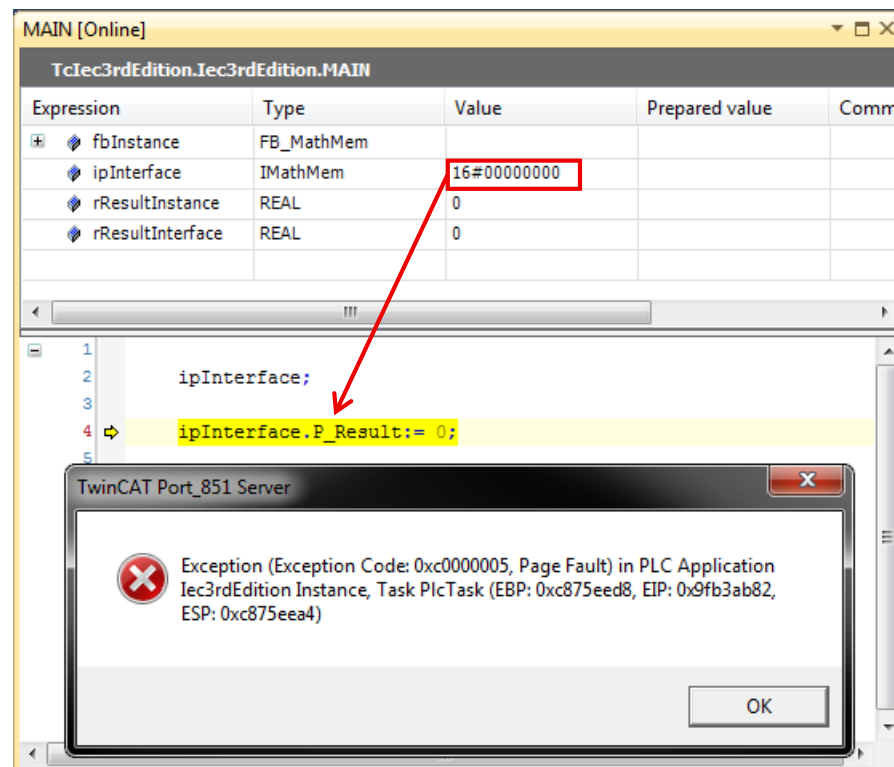
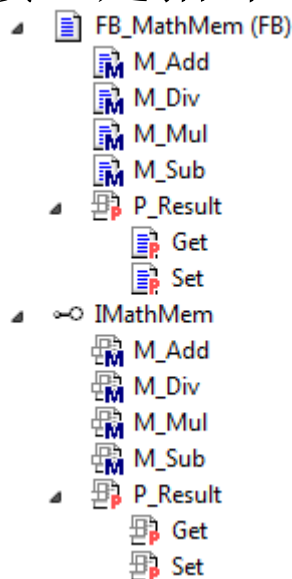
```
1  
2 ipInterface := fbInstance;  
3  
4 ipInterface.P_Result := 0;  
5  
6 ipInterface.M_Add(3);  
7  
8 ipInterface.M_Mul(2);  
9  
10 rResultInstance := fbInstance.P_Result;  
11 rResultInterface := ipInterface.P_Result; RETURN
```

- 根据情况使用不同的FB。
 - 如果条件满足，那么就使用FB_MathMem。
 - 否则使用FB_MathErase。



- 功能块内部是具体的值。
- 接口存储功能块的地址。

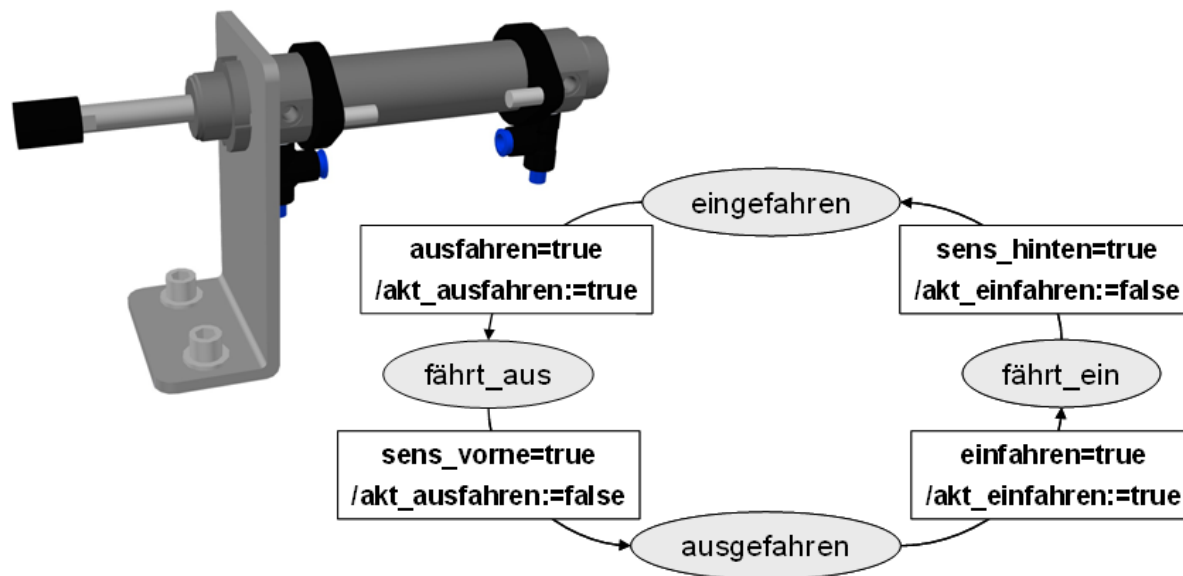
➔ 接口是指针！



- IEC 61131-3 第三版的面向对象编程
- 详细介绍介绍
- **OOP**的示例应用



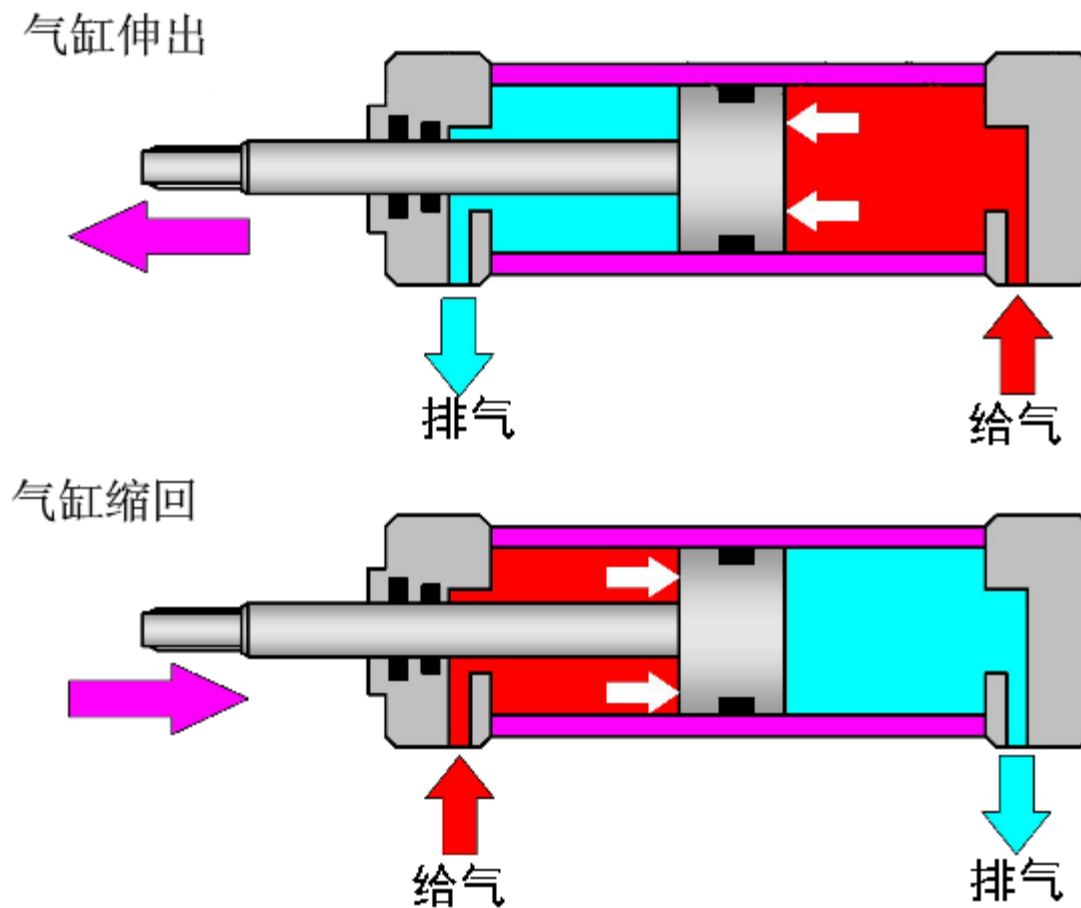
Example: 气缸

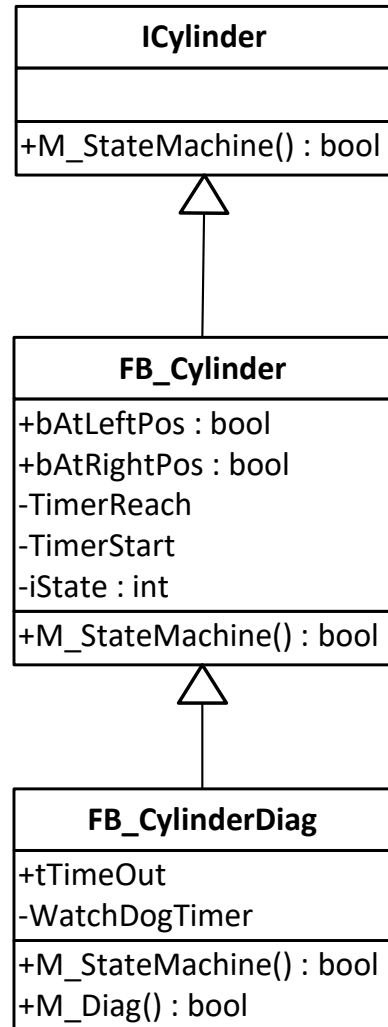


任务提示:

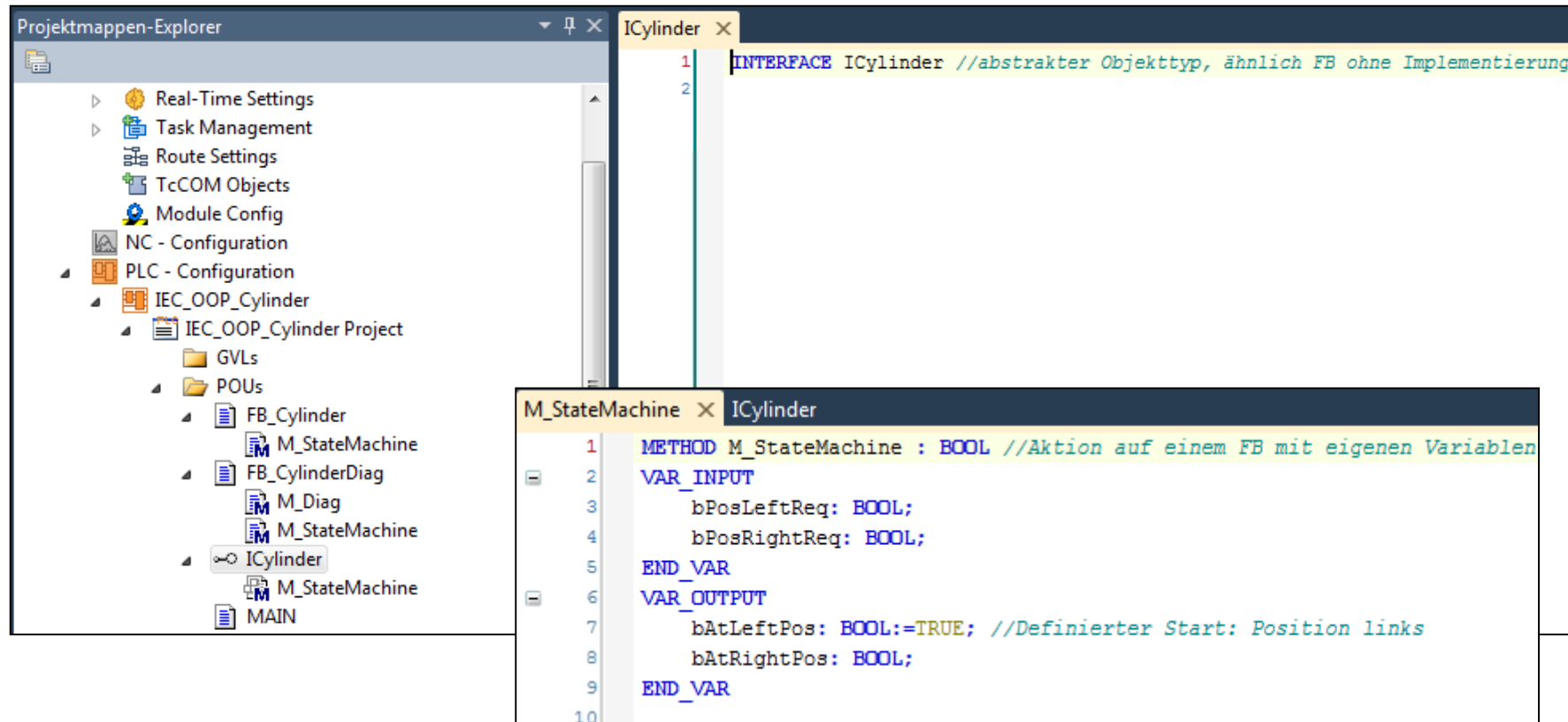
- 同一类型的气缸
- 一致的动作
- 有带诊断的气缸

■ 气缸的工作原理

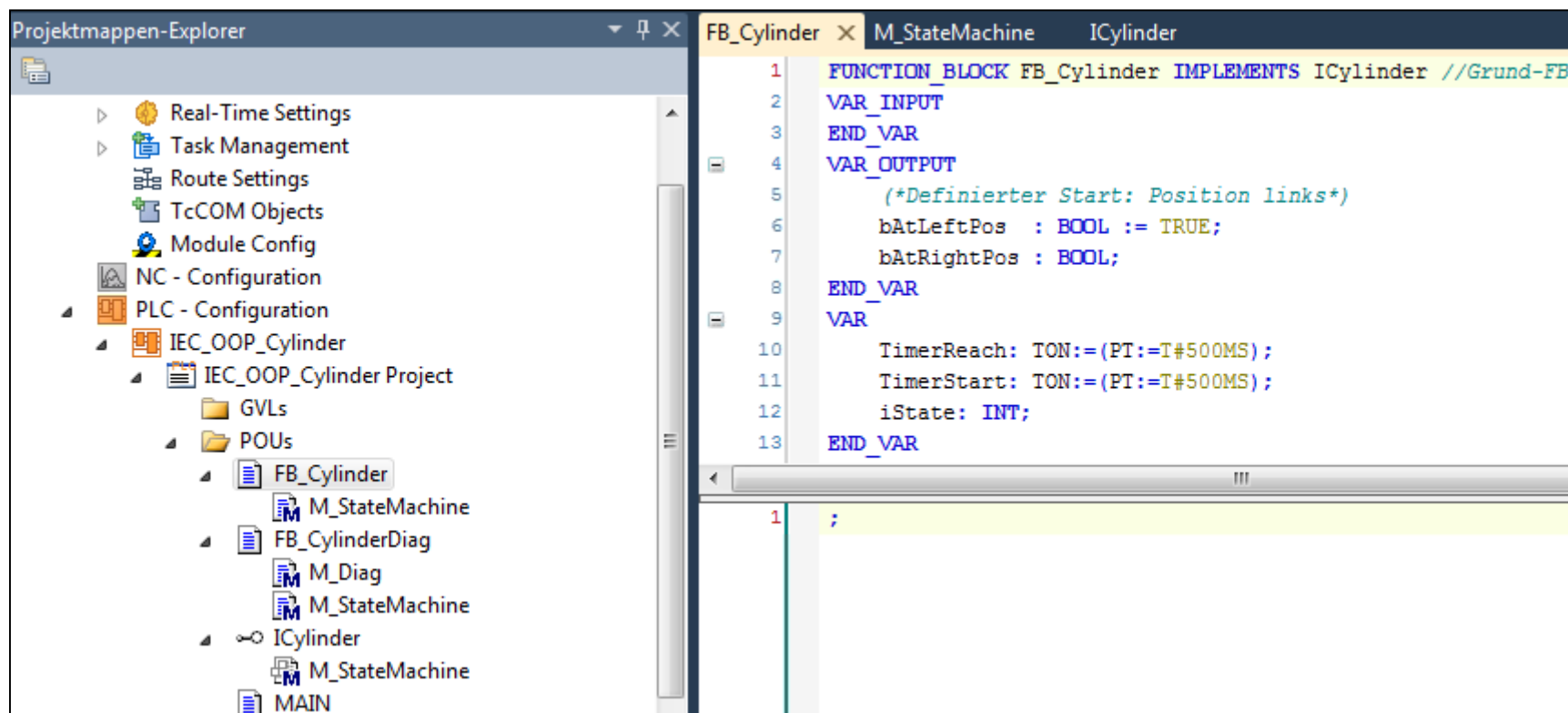




- 定义一个抽象的父类 (接口): I 气缸 ICylinder
- 相关的方法只是被创建, 相关的数据仅在实现时被定义。

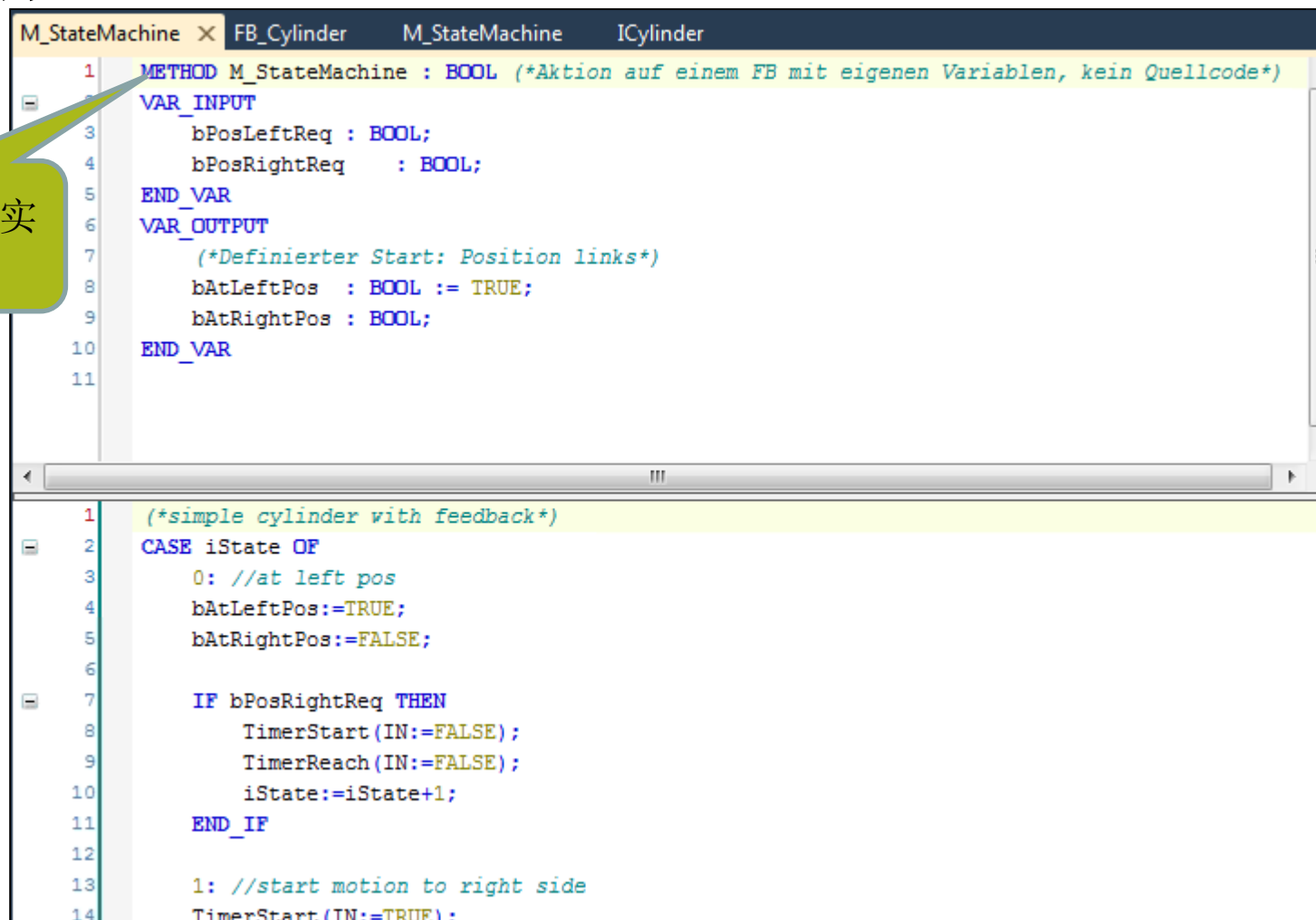


- 定义一个基本功能块FB_Cylinder来实现接口 I 气缸。
- 气缸的基本动作都在这个功能块中定义。



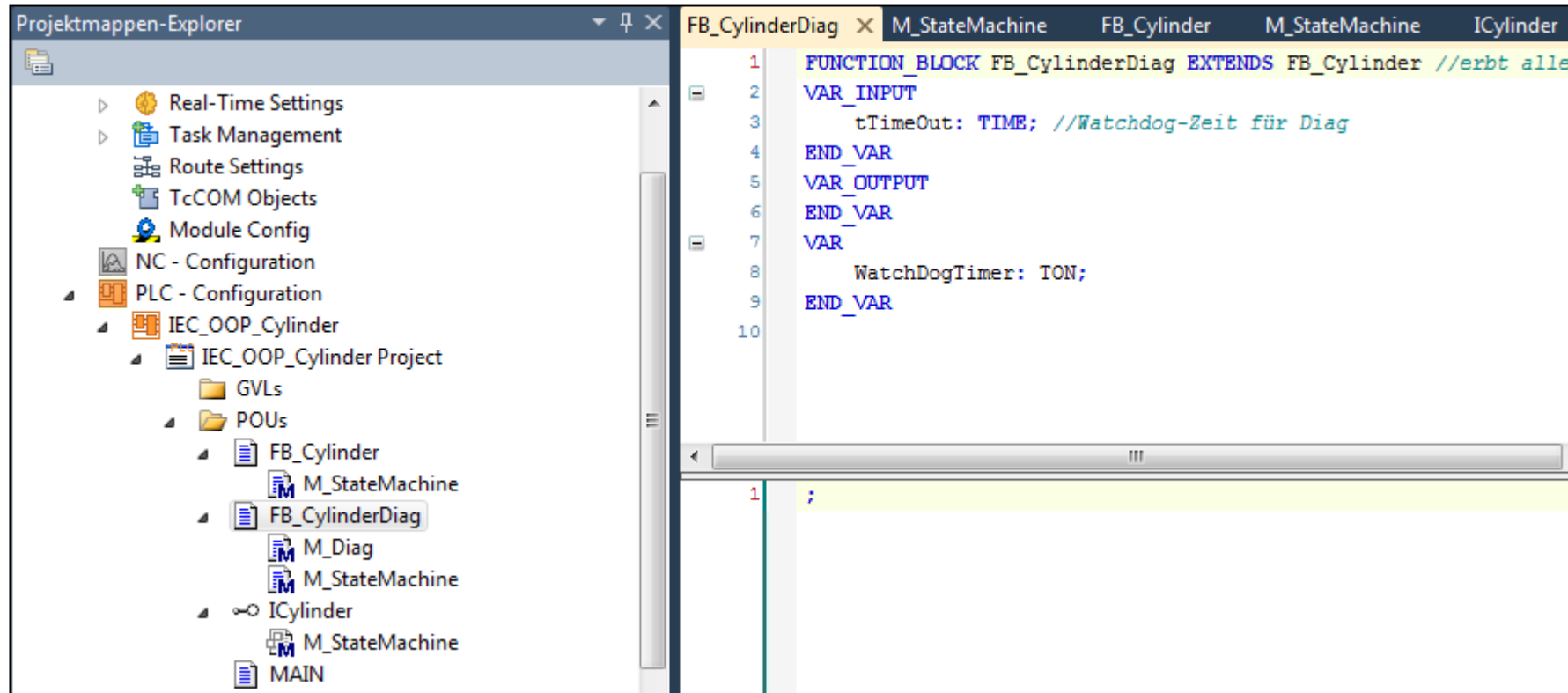
- 用于控制所有气缸的方法:

接口方法的实现



```
1 METHOD M_StateMachine : BOOL (*Aktion auf einem FB mit eigenen Variablen, kein Quellcode*)
2 VAR_INPUT
3     bPosLeftReq : BOOL;
4     bPosRightReq : BOOL;
5 END_VAR
6 VAR_OUTPUT
7     (*Definierter Start: Position links*)
8     bAtLeftPos : BOOL := TRUE;
9     bAtRightPos : BOOL;
10 END_VAR
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
280
281
282
283
284
285
286
287
288
289
290
291
292
293
294
295
296
297
298
299
300
301
302
303
304
305
306
307
308
309
310
311
312
313
314
315
316
317
318
319
320
321
322
323
324
325
326
327
328
329
330
331
332
333
334
335
336
337
338
339
340
341
342
343
344
345
346
347
348
349
350
351
352
353
354
355
356
357
358
359
360
361
362
363
364
365
366
367
368
369
370
371
372
373
374
375
376
377
378
379
380
381
382
383
384
385
386
387
388
389
390
391
392
393
394
395
396
397
398
399
400
401
402
403
404
405
406
407
408
409
410
411
412
413
414
415
416
417
418
419
420
421
422
423
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
590
591
592
593
594
595
596
597
598
599
600
601
602
603
604
605
606
607
608
609
610
611
612
613
614
615
616
617
618
619
620
621
622
623
624
625
626
627
628
629
630
631
632
633
634
635
636
637
638
639
640
641
642
643
644
645
646
647
648
649
650
651
652
653
654
655
656
657
658
659
660
661
662
663
664
665
666
667
668
669
670
671
672
673
674
675
676
677
678
679
680
681
682
683
684
685
686
687
688
689
690
691
692
693
694
695
696
697
698
699
700
701
702
703
704
705
706
707
708
709
710
711
712
713
714
715
716
717
718
719
720
721
722
723
724
725
726
727
728
729
730
731
732
733
734
735
736
737
738
739
740
741
742
743
744
745
746
747
748
749
750
751
752
753
754
755
756
757
758
759
760
761
762
763
764
765
766
767
768
769
770
771
772
773
774
775
776
777
778
779
780
781
782
783
784
785
786
787
788
789
790
791
792
793
794
795
796
797
798
799
800
801
802
803
804
805
806
807
808
809
810
811
812
813
814
815
816
817
818
819
820
821
822
823
824
825
826
827
828
829
830
831
832
833
834
835
836
837
838
839
840
841
842
843
844
845
846
847
848
849
850
851
852
853
854
855
856
857
858
859
860
861
862
863
864
865
866
867
868
869
870
871
872
873
874
875
876
877
878
879
880
881
882
883
884
885
886
887
888
889
890
891
892
893
894
895
896
897
898
899
900
901
902
903
904
905
906
907
908
909
910
911
912
913
914
915
916
917
918
919
920
921
922
923
924
925
926
927
928
929
930
931
932
933
934
935
936
937
938
939
940
941
942
943
944
945
946
947
948
949
950
951
952
953
954
955
956
957
958
959
960
961
962
963
964
965
966
967
968
969
970
971
972
973
974
975
976
977
978
979
980
981
982
983
984
985
986
987
988
989
990
991
992
993
994
995
996
997
998
999
1000
```

- 创建一个诊断功能块，它继承了基本型 (EXTENDS):



- 重载方法: M_StateMachine from FB_Cylinder

- 通过特殊类型的诊断，来扩展被继承的方法：

```
M_Diag  X  FB_CylinderDiag  M_StateMachine  FB_Cylinder

1  METHOD M_Diag : Bool
2  VAR_INPUT
3  END_VAR
4

1  IF (iState=0) OR (iState=3) THEN //iState
2      WatchDogTimer(IN:=FALSE , PT:=tTimeOut);
3  ELSE
4      WatchDogTimer(IN:=TRUE , PT:=tTimeOut);
5  END_IF
6
7  M_Diag:=WatchDogTimer.Q;
```

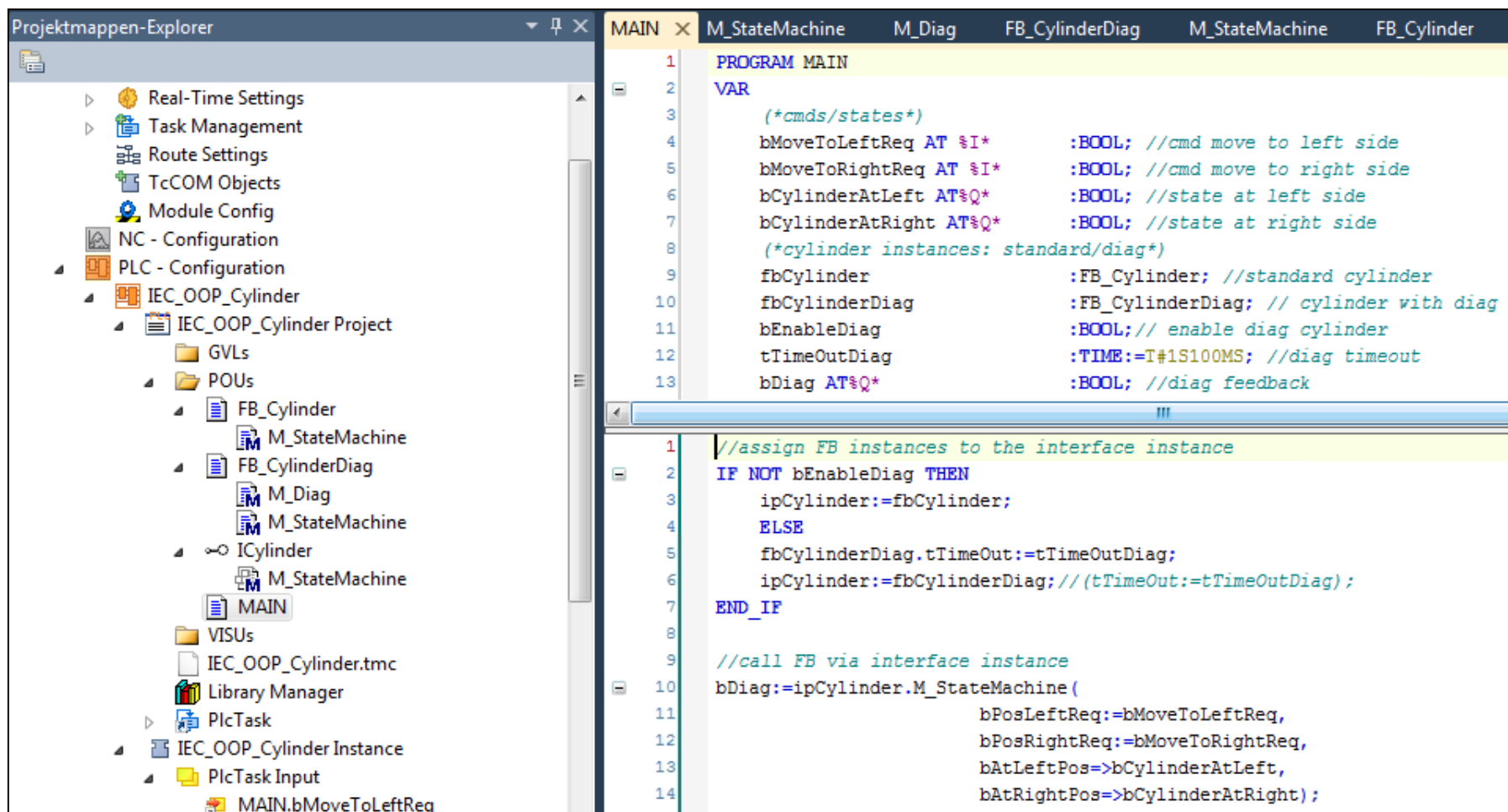
重载接口的方法

```
M_StateMachine  X  M_Diag  FB_CylinderDiag  M_Sta

1  METHOD M_StateMachine : Bool
2  VAR_INPUT
3      bPosLeftReq : BOOL;
4      bPosRightReq : BOOL;
5  END_VAR
6  VAR_OUTPUT
7      (*Definiierter Start: Position link
8      bAtLeftPos : BOOL := TRUE;
9      bAtRightPos : BOOL;
10 END_VAR

1  //Ruft die übergeordnete M_StateMachine
2
3  SUPER^.M_StateMachine(
4      bPosLeftReq:=bPosLeftReq,
5      bPosRightReq:=bPosRightReq,
6      bAtLeftPos=>bAtLeftPos,
7      bAtRightPos=>bAtRightPos);
8
9  M_StateMachine:=M_Diag(); //übergibt da
```

■ 调用功能块 - 类型独立 - 通过接口



The screenshot displays the TwinCAT 3 IDE interface. On the left, the 'Projektmappen-Explorer' (Project Explorer) shows the project structure for 'IEC_OOP_Cylinder'. The 'MAIN' program is selected. The main editor area shows the ladder logic for the 'MAIN' program, which is a single rungs (Rung 1) containing a 'PROGRAM MAIN' block. The program defines several variables and calls the 'fbCylinder' function block via an interface instance 'ipCylinder'.

```
1 PROGRAM MAIN
2 VAR
3   (*cmds/states*)
4   bMoveToLeftReq AT %I*      :BOOL; //cmd move to left side
5   bMoveToRightReq AT %I*     :BOOL; //cmd move to right side
6   bCylinderAtLeft AT %Q*     :BOOL; //state at left side
7   bCylinderAtRight AT %Q*    :BOOL; //state at right side
8   (*cylinder instances: standard/diag*)
9   fbCylinder                :FB_Cylinder; //standard cylinder
10  fbCylinderDiag             :FB_CylinderDiag; // cylinder with diag
11  bEnableDiag                :BOOL; // enable diag cylinder
12  tTimeOutDiag               :TIME:=T#1S100MS; //diag timeout
13  bDiag AT %Q*               :BOOL; //diag feedback
14
15  //assign FB instances to the interface instance
16  IF NOT bEnableDiag THEN
17    ipCylinder:=fbCylinder;
18  ELSE
19    fbCylinderDiag.tTimeOut:=tTimeOutDiag;
20    ipCylinder:=fbCylinderDiag; // (tTimeOut:=tTimeOutDiag);
21  END_IF
22
23  //call FB via interface instance
24  bDiag:=ipCylinder.M_StateMachine (
25    bPosLeftReq:=bMoveToLeftReq,
26    bPosRightReq:=bMoveToRightReq,
27    bAtLeftPos=>bCylinderAtLeft,
28    bAtRightPos=>bCylinderAtRight);
```

Language properties	2nd Edition IEC 61131-3	3rd Edition IEC 61131-3	C++	Java	C#
Multilingual capability	+	+	-	-	-
OOP/procedural mixed	-	+	+	-	-
Classes	~ (FB)	+	+	+	+
Methods	~ (Actions)	+	+	+	+
Interfaces	-	+	-	+	+
Partially abstract classes	-	-	+	+	+
Polymorphism	-	+	+/-	+	+
Reference semantics	-	+ (Interfaces)	-	+	+
Constructor/Destructor	-	+	+	+	+
Properties	-	+	-	-	+
Visibility	~ (Variables)	~ (Variablen)	+	+	+
Dyn. memory ('new')	-	- (in TC3)	+	+	+

德国倍福自动化有限公司

上海总部

上海市汶水路299弄9号（市北智汇园）

电话： 021-6631 2666

传真： 021-6631 5696

E-Mail: support@beckhoff.com.cn

Web: www.beckhoff.com.cn

虚拟学院: <https://tr.beckhoff.com.cn>



扫一扫，关注倍福官方微信！

© 德国倍福自动化有限公司

本 PowerPoint 演示文稿中的所有照片及图片均受版权保护。未经许可，任何用户不得擅自复制、使用、转载或将其提供给任何第三方。

Beckhoff®、TwinCAT®、EtherCAT®、Safety over EtherCAT®、TwinSAFE®、XFC® 和 XTS®是德国倍福自动化有限公司的注册商标。本 PowerPoint 演示文稿中所使用的其它名称可能是商标名称，任何第三方为其自身目的而引用，都可能触犯商标所有者的权利。

本PowerPoint 演示文稿中所包含的信息仅是一般描述或性能特征简介，在实际应用中并不总是与所述完全一致或者可能由于产品的进一步开发而不完全适用。仅在书面认同情况下，才提供相关特性信息。