SIEMENS西门子广西省贺州市（授权）一级代理商——西门子伺服电机华南总代理

产品名称	SIEMENS西门子广西省贺州市（授权）一级代理商——西门子伺服电机华南总代理
公司名称	广东湘恒智能科技有限公司
价格	.00/件
规格参数	西门子总代理:PLC 西门子一级代:驱动 西门子代理商:伺服电机
公司地址	惠州大亚湾澳头石化大道中480号太东天地花园2栋二单元9层01号房
联系电话	15915421161 15903418770

产品详情

赋值运算定义：通过赋值运算，可以将一个表达式的值分配给一个变量。赋值表达式的左侧为变量，右侧为表达式的值。函数名称也可以作为表达式。赋值运算将调用该函数，并返回其函数值，赋给左侧的变量。赋值运算的数据类型取决于左边变量的数据类型。右边表达式的数据类型必须与该数据类型一致。赋值运算的计算按照从右到左的顺序进行。可通过以下方式编程赋值运算：单赋值运算：执行单赋值运算时，仅将一个表达式或变量分配给单个变量：示例：a := b;多赋值运算：执行多赋值运算时，一个指令中可执行多个赋值运算。示例：a := b := c;此时，将执行以下操作：b := c;a := b;组合赋值运算：执行组合赋值运算时，可在赋值运算中组合使用操作符"+"、"-"、"*"和"/"：示例：a += b;此时，将执行以下操作：a := a + b;也可多次组合赋值运算：a += b += c *= d;此时，将按以下顺序执行赋值运算：c := c * d;b := b + c;a := a + b;示例下表举例说明了单赋值运算的操作："MyTag1" := "MyTag2";(* 变量赋值 *)"MyTag1" := "MyTag2" * "MyTag3";(* 表达式赋值 *)"MyTag" := "MyFC"();(* 调用一个函数，并将函数值赋给 "MyTag" 变量 *)#MyStruct.MyStructElement := "MyTag";(* 将一个变量赋值给一个结构元素 *)#MyArray[2] := "MyTag";(* 将一个变量赋值给一个 ARRAY 元素 *)"MyTag" := #MyArray[1,4];(* 将一个 ARRAY 元素赋值给一个变量 *)#MyString[2] := #MyOtherString[5];(* 将一个 STRING 元素赋给另一个 STRING 元素 *)下表举例说明了多赋值运算的操作："MyTag1" := "MyTag2" := "MyTag3";(* 变量赋值 *)"MyTag1" := "MyTag2" := "MyTag3" * "MyTag4";(* 表达式赋值 *)"MyTag1" := "MyTag2" := "MyTag3 := "MyFC"();(* 调用一个函数，并将函数值赋值给变量 "MyTag1"、"MyTag1" 和 "MyTag1" *)#MyStruct.MyStructElement1 := #MyStruct.MyStructElement2 := "MyTag";(* 将一个变量赋值给两个结构元素 *)#MyArray[2] := #MyArray[32] := "MyTag";(* 将一个变量赋值给两个数组元素 *)"MyTag1" := "MyTag2" := #MyArray[1,4];(* 将一个数组元素赋值给两个变量 *)#MyString[2] := #MyString[3]:= #MyOtherString[5];(* 将一个 STRING 元素赋值给两个 STRING 元素 *)下表举例说明了组合赋值运算的操作："MyTag1" += "MyTag2";(* "MyTag1" 和 "MyTag2" 相加，并将相加的结果赋值给 "MyTag1"。*)"MyTag1" -= "MyTag2" += "MyTag3";(* "MyTag2" 和 "MyTag3" 相加。将相加的结果赋值给操作数"MyTag2"，再从 "MyTag1"

中减去"MyTag2"，计算结果将赋值给 "MyTag1"。*)#MyArray[2] += #MyArray[32] += "MyTag";(* 数组元素 "MyArray[32]" 加上 "MyTag"。计算结果将赋值给 "MyArray[32]"。之后这个数组元素 "MyArray[32]" 与数组中另一个元素"MyArray[2]"相加，然后将结果分配给数组元素 "MyArray[2]"。在该运算中，相应的数据类型必需兼容。*)#MyStruct.MyStructElement1 /= #MyStruct.MyStructElement2 *= "MyTag";(* 结构化元素 "MyStructElement2" 乘以 "MyTag"。计算结果将赋值给 "MyStructElement2"。之后，将结构化元素 "MyStructElement1" 除以 "MyStructElement2"，并将计算结果赋值给 "MyStructElement1"。在该运算中，相应的数据类型必需兼容。*)寻址与调用寻址SCL寻址分为符号寻址与地址寻址。符号寻址DB块变量："DB块名称"（."变量名称"）PLC变量：变量名称局部变量：#变量名称地址寻址DB块变量：%DB块号（.变量地址），TIA PORTAL软件会判断该地址有没有对应符号名称，如果有则立即转换为符号名称，没有则保留juedui地址PLC变量：%变量地址，TIA PORTAL软件会判断该地址有没有对应符号名称，如果有则立即转换为符号名称，没有则新建符号名称Temp变量：SCL不支持非优化FC/FB的Temp变量的地址寻址举例： 符号名说明符号寻址DB块变量"MyDB".Variable.Static_1"MyDB".Array[0]访问数组元素"MyDB"DB块名作为参数PLC变量"Start"局部变量#Input_1#Temp_1.x0变量名片段访问地址寻址DB块变量%DB2.DBB1%DB2DB块名作为参数，会立刻转换为DB块名PLC变量%M100.0会立刻转换为"符号名"%Q1.0:P会立刻转换为"符号名":P调用程序调用分为以下几类：FC调用FB调用FB多重背景调用调用可以从指令列表或者项目树程序块中拖拽入程序编辑区域，也可以直接输入。FC调用FC调用的格式是"FC块名称"(输入形参:=实参,输出形参=>实参,输入输出形参:=实参...)返回值:= “ FC块名称 ” (输入形参:=实参,输出形参=>实参,输入输出形参:=实参...)FC调用需要确保所有形参都有对应实参。如果没有参数的FC也需要有括号。如图所示的例子；图1 FC调用FB调用FB调用的格式是"背景数据块名称"(输入形参:=实参,输出形参=>实参,输入输出形参:=实参...)一般情况下，FB的简单数据类型形参可以没有对应实参，复杂数据类型的输入、输出也可以没有对应实参，所以FB可以隐藏或不隐藏不出现的形参。如果没有参数的FB也需要有括号。如图2所示，显示了一些FB调用的例子。图2 FB调用如图3所示，当FB的参数全部显示，在背景数据块右键可以激活"仅显示分配的参数"；当FB的参数只显示了分配的参数时，在背景数据块右键可以激活"显示所有参数"。图3 显示分配/所有参数FB多重背景调用FB多重背景调用的格式是#多重背景(输入形参:=实参,输出形参=>实参,输入输出形参:=实参...)#多重背景[索引](输入形参:=实参,输出形参=>实参,输入输出形参:=实参...)一般情况下，FB的简单数据类型形参可以没有对应实参，复杂数据类型的输入、输出也可以没有对应实参，所以FB可以隐藏或不隐藏不出现的形参。如果只有Static的FB也需要有括号。如图4所示，显示了一些FB多重背景调用的例子。图4 FB多重背景调用注意：对于定时器和计数器的SCL调用，有特殊的格式，请参考链接：定时器、计数器。新建SCL有两种方式新建SCL：第一种是在新建块，选择OB/FC/FB后，设置语言为SCL，如图5所示。第二种是在LAD、FBD中直接插入SCL语言段，这需要TIA PORTAL V14及其以上的版本，如图6所示。图5 新建SCL块①在项目树中，找到PLC，然后展开程序块，点击"添加新块"②在弹出对话框中，选择块类型，可以是OB/FB/FC，③选择语言为SCL图6 在LAD中插入SCL段区域与注释和LAD/FBD不同，LAD/FBD在程序编辑器是一段一段的，编辑器可以插入新的网络段，每一个网络段可以有各自的注释。而SCL是文本语言，不分网络段（LAD/FBD语言内增加SCL除外），需要用其他的方法来解决。区间从TIA PORTAL V14以后，增加区间功能，使用指令：REGION 区间名称程序文本END_REGION可以在指令中间增加需要编写的程序还不影响程序逻辑，并且支持嵌套。此外还可以像网络段一样收折叠来，如图7所示。图7 区域其中左边为区间总览，可以看出整体的结构①使得程序或总览全部展开②使得程序或总览全部折叠③全部展开/折叠是针对总览与程序还是只针对总览，图中为针对总览与程序④独立展开/折叠程序注释编辑器的空行，或者调用块的右侧均可以增加注释，如图8所示有两种方式注释：第一种是：
//注释内容第二种是：(/*注释内容*/)可以在工具栏中利用按钮整段注释或取消注释。此外从TIA PORTAL V16开始支持多语言注释，使用指令(*多语言注释内容*)，具体参考多语言文档。图8 注释①注释掉选中段落②对注释掉的段落取消注释