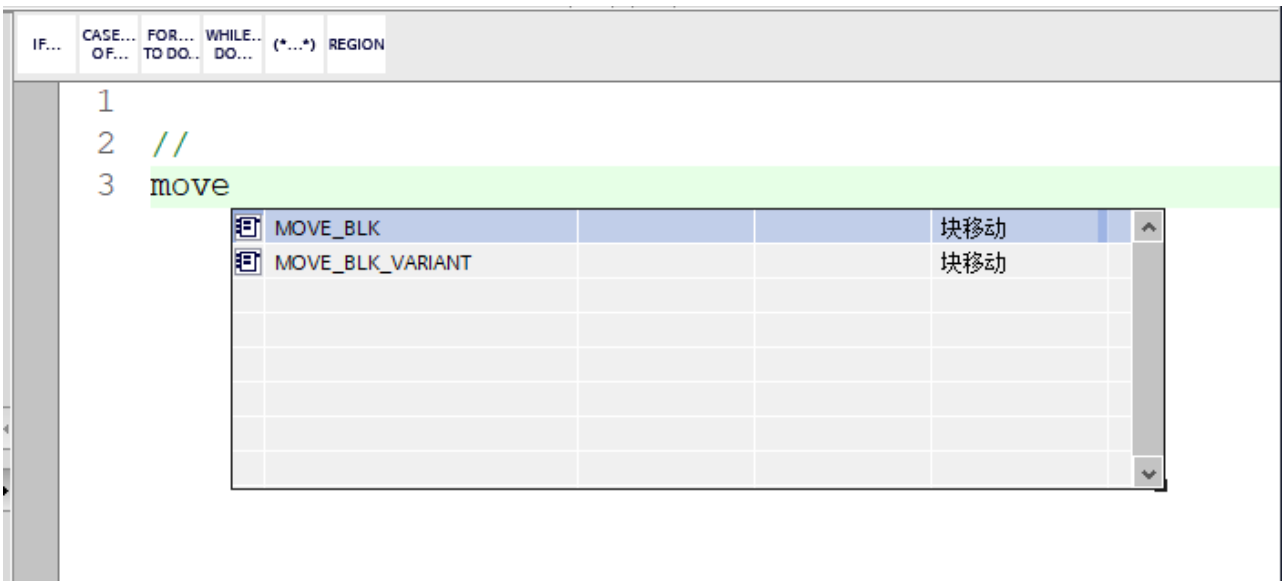
块移动指令包括两个：MOVE_BLK和MOVE_BLK_VARIANT，实际项目中都经常被用到。比如设备作业结果的存储及显示功能就需要用到块移动指令。通常采用这种逻辑：首先创建一个数据块，在其中创建一个定长数组（比如可存放100条记录）。每次设备作业流程结束后，将流程结果的数据作为一条记录存放到数组中。当需要查看某次作业结果的数据时，从数组中读取相关的数据拷贝到临时变量或某个特定变量中即可。这类存储功能一般使用MOVE_BLK_VARIANT指令，因为它可以拷贝结构体数据类型（设备流程结果一般内容比较多，比如包括压力、温度、时间等等，多数采用结构体变量存储），而MOVE_BLK指令只能拷贝基础数据类型（整数、实数等）。数据的写入及读取要设定相应的指针变量，实际应用中经常因为指针错误导致读写错误。今天这篇文章我们介绍一下MOVE_BLK_VARIANT指令在使用中应注意的问题。

MOVE_BLK_VARIANT指令位于【指令】任务卡的【基本指令】→【移动操作】中，使用梯形图或功能块图编程时可以在这个位置将其拖放到函数或函数块中。使用SCL语言编程时，可以直接输入"MOVE"编辑器会自动提示，如下图所示：



指令添加后的初始状态如下图所示：

```
1
2  //
3 MOVE_BLK_VARIANT(SRC:= variant_in ,
4                   COUNT:= udint_in ,
5                   SRC_INDEX:= dint_in ,
6                   DEST_INDEX:= dint_in ,
7                   DEST=> variant_out )
8
```

指令参数含义如下：

1. SRC：可变（Variant）数据类型。源数据，可以是数组或结构体复杂数据类型；
2. COUNT：无符号双整数（UDINT），要拷贝的记录条数；
3. SRC_INDEX：双整数（DINT），源数据的索引；
4. DEST_INDEX：双整数（DINT），目标数据的索引；
5. DEST：可变（Variant）数据类型。目标数据，即源数据要拷贝到的目标区域；

说明：

1. 无论数组的上、下限是如何定义，源数据和目标数据的索引都是从0开始计算的；
2. 如果源数据或目标参数不是数组，则其相应的索引值应为0；

举个例子：

假设存储数组data的定义如下：

```
data[1..100] of typeResult//typeResult是用户自定义数据类型
```

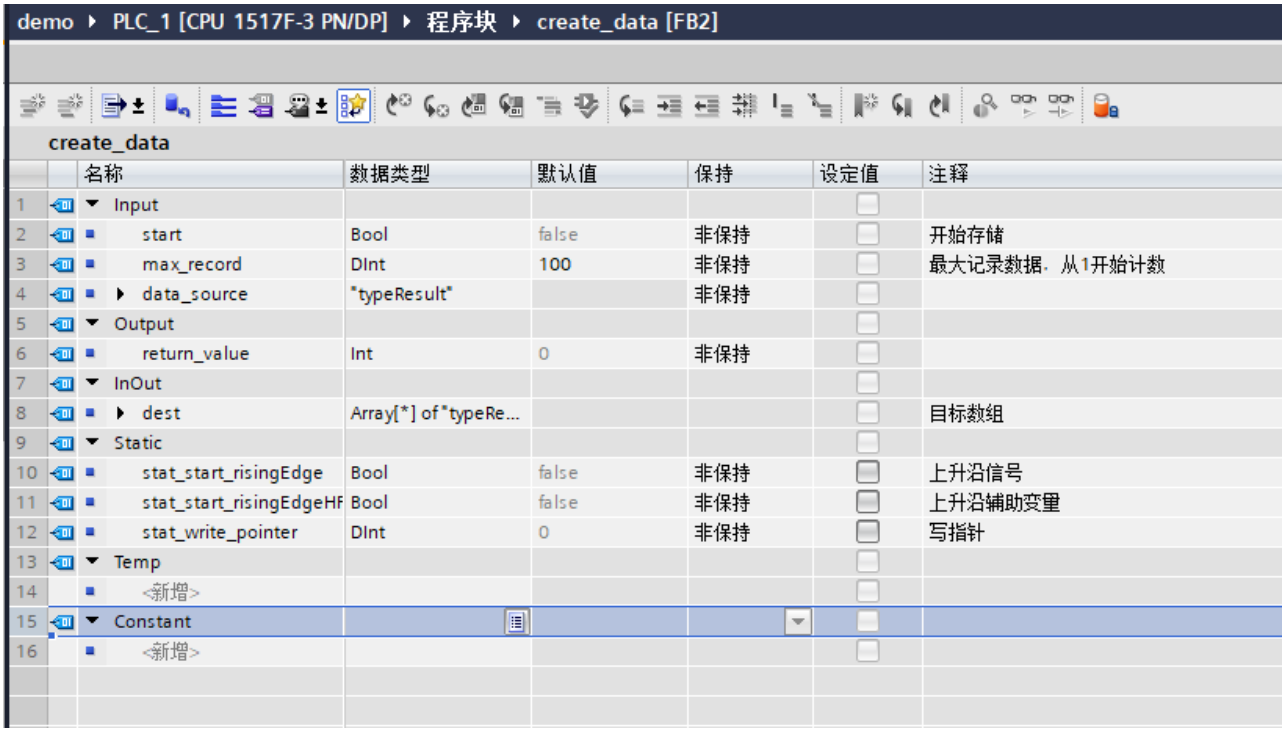
假设过程结果存放在变量tmpResult中。第一次将过程结果存放到存储区数组中的代码如下：

```
#return_value:=MOVE_BLK_VARIANT(SRC := #tmpResult,
                                COUNT := 1,
                                SRC_INDEX := 0,
                                DEST_INDEX := 0,
                                DEST => "db_storage".data);
```

注意这里“DEST_INDEX”的值为0，而数据则存放在数组的第一个元素中。由于该数组的下限为1，所以存放在data[1]中。假设定义的存放数据为data[10..110]，则存放第一条记录的“DEST_INDEX”的值仍然为0，由于该数组的下限为10，因此数据存放在data[10]中。

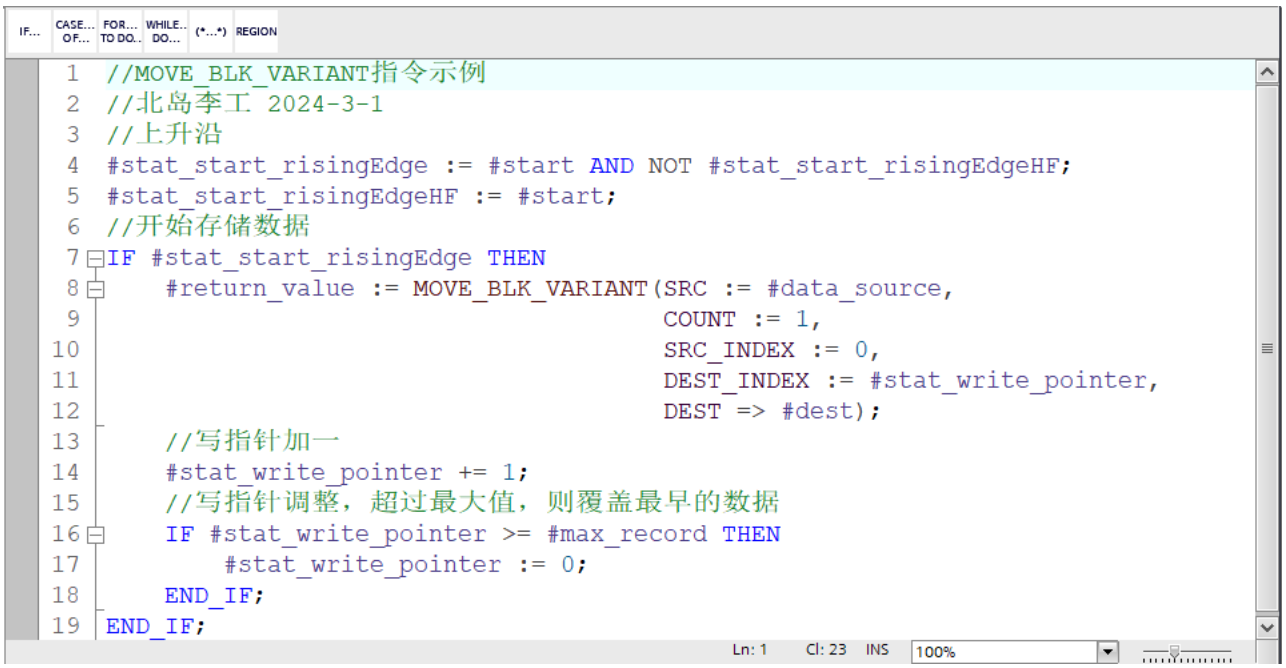
实际应用中需要定义写指针变量，随着存入次数的变化修改写指针的值，从而存放到数组的不同位置中。为了防止数据重复写入，存储应在沿信号时进行。

基于上述描述我写了一个简单的数据写入代码，变量定义如下图所示：



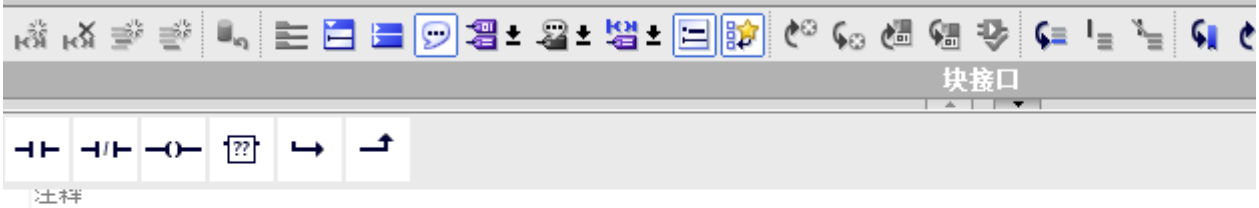
	名称	数据类型	默认值	保持	设定值	注释
1	Input					
2	start	Bool	false	非保持		开始存储
3	max_record	DInt	100	非保持		最大记录数据，从1开始计数
4	data_source	"typeResult"		非保持		
5	Output					
6	return_value	Int	0	非保持		
7	InOut					
8	dest	Array[*] of "typeRe...				目标数组
9	Static					
10	stat_start_risingEdge	Bool	false	非保持		上升沿信号
11	stat_start_risingEdgeHF	Bool	false	非保持		上升沿辅助变量
12	stat_write_pointer	DInt	0	非保持		写指针
13	Temp					
14	<新增>					
15	Constant					
16	<新增>					

代码如下图所示：



```
1  //MOVE_BLK_VARIANT指令示例
2  //北岛李工 2024-3-1
3  //上升沿
4  #stat_start_risingEdge := #start AND NOT #stat_start_risingEdgeHF;
5  #stat_start_risingEdgeHF := #start;
6  //开始存储数据
7  IF #stat_start_risingEdge THEN
8      #return_value := MOVE_BLK_VARIANT(SRC := #data_source,
9                                        COUNT := 1,
10                                       SRC_INDEX := 0,
11                                       DEST_INDEX := #stat_write_pointer,
12                                       DEST => #dest);
13     //写指针加一
14     #stat_write_pointer += 1;
15     //写指针调整，超过最大值，则覆盖最早的数据
16     IF #stat_write_pointer >= #max_record THEN
17         #stat_write_pointer := 0;
18     END_IF;
19 END_IF;
```

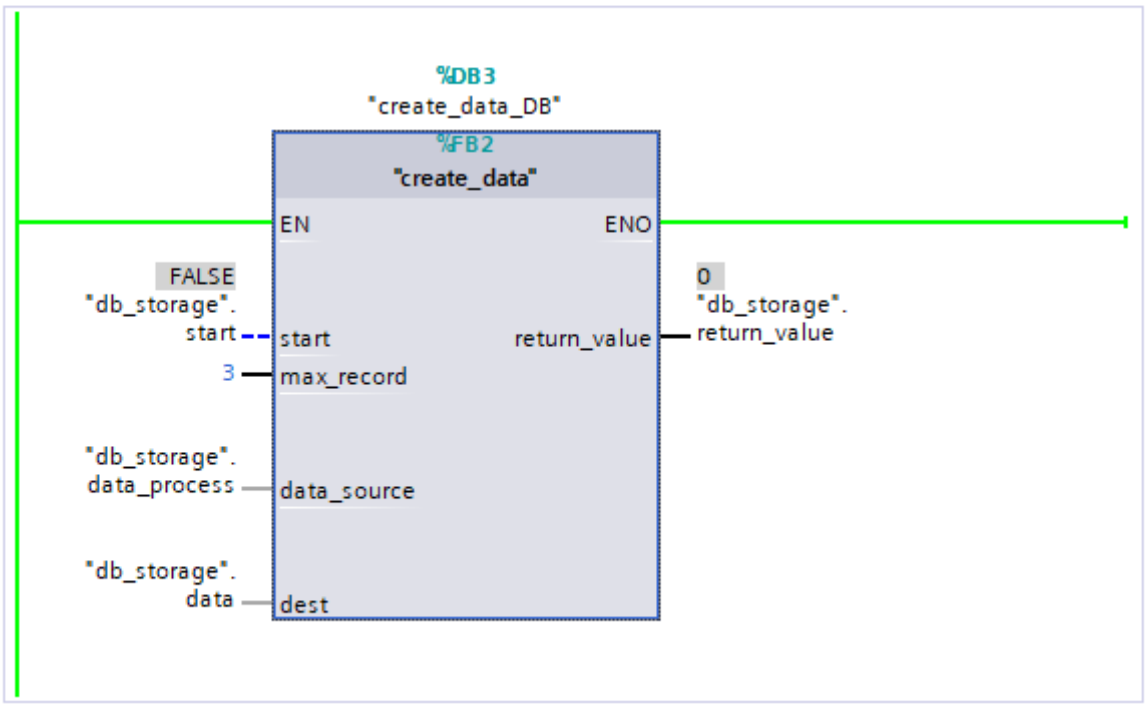
这段代码我已经测试过了，包括数据写满后的情况，如下图所示：



注释

程序段 1： test

注释



	名称	数据类型	起始值	监视值	保持
1	Static				
2	data	Array[1..3] of "type...			
3	data[1]	"typeResult"			
4	pressure	Real	0.0	33.0	
5	temperature	Real	0.0	63.0	
6	speed	Real	0.0	102.0	
7	humidity	Real	0.0	57.0	
8	data[2]	"typeResult"			
9	pressure	Real	0.0	31.0	
10	temperature	Real	0.0	61.0	
11	speed	Real	0.0	101.0	
12	humidity	Real	0.0	56.0	
13	data[3]	"typeResult"			
14	pressure	Real	0.0	32.0	
15	temperature	Real	0.0	62.0	
16	speed	Real	0.0	102.0	
17	humidity	Real	0.0	57.0	
18	data_process	"typeResult"			
19	pressure	Real	30.0	33.0	
20	temperature	Real	60.0	63.0	
21	speed	Real	100.0	102.0	
22	humidity	Real	55.0	57.0	
23	start	Bool	false	FALSE	
24	return_value	Int	0	0	

这篇文章最核心的内容就是大家在使用MOVE_BLK_VARIANT指令时要记住其SRC_INDEX和DEST_INDEX的计算是从0开始的。感兴趣的小伙伴可以把上述的代码进行完善，然后完成数据读取的代码。

好吧，就先聊到这里。

下面是西门子SCL编程的文章归档链接：

》》西门子SCL编程入门到精通文章归档《《

我的书《西门子S7-1200/1500 PLC SCL语言编程 ——从入门到精通》从硬件到软件，比较详细的介绍了SCL语言的编程，感兴趣的话可以扫描下面的二维码查看：


西门子 S7-1200/1500 PLC SCL语言编程 从入门到精通
西门子
S7-1200/1500 PLC
SCL语言编程
从入门到精通
北岛李工 编
SCL
SCL
全彩印刷
视频同步讲解

识别图中小程序码购买