FB块和FC块

FB和FC区别

FB--功能块，带背景数据块 FC--功能，相当于函数
FB，FC块均相当于子程序，既可以调用其它FB，FC块，也可以被OB，FB，FC块调用。

他们之间的主要区别是：

1. FB使用背景数据块作为存储区，FC没有独立的存储区，使用全局DB或M区
2. FB局部变量有STAT和TEMP，FC由于没有自己的存储区因此不具有STAT，TEMP本身不能设置初始值。
本质上，FB，FC的实现目的是相同的；无论何种逻辑要求，FB，FC均可实现。只是实现方式效率不同，这也和工程师个人编程习惯有关。

FB块优点：

1. 易于移植性，对于相同控制逻辑不同参数的被控对象，只要使用不同的背景DB，同一个FB块就可以方便
2. 多重背景，减少重复工作，提高效率
3. 多次调用时，参数修改方便
4. 有独立的存储区

FC块优点：

1. 小巧灵活，对于非多次调用的程序更易理解
2. 不占用额外的存储资源

FB,FC块管脚定义

IN---------变量是外部输入的，只能被本程序块读，不能被本程序块写;
OUT-------是本程序块输出的，他可以被本程序块读写，其他程序通过引脚只能读值不能写;
IN_OUT--- 输入输出变量 本程序块和其他程序都可以读写这个引脚的值。
TEMP -----临时变量，顾名思义是暂时存储数据的变量。这些临时的数据存储在CPU工作存储区的局部数据堆栈（L堆栈）中。
STAT-------在PLC运行期间始终被存储。S7 将静态变量定义在背景数据块（仅对FB而言，FC和OB无静态变量），当被调用块运行时，能读出或修改静态变量；被调用块结束后，静态变量保留在数据块中。

为何定义的FB，FC块，多次调用后程序混乱？

对于，多次调用的程序块，FB块建议更换调用不同的背景DB；FC则需要确保使用的存储地址不重复，即每次调用，块中调用的地址不重复。

为何含有定时器或计数器的FB或FC单次调用ok，多次调用时定时器或计数器混乱？

对于多次调用的FB，FC，如为S7定时器，计数器，则需要在IN接口中定义TIMER或Counter，每调用一次FB或FC，均赋不同的定时器或计数器号。
如为IEC定时器，计数器，则需要在IN接口定义Block_DB，每调用一次FB或FC，均赋不同的DB块给其中的IEC定时器或计数器。

临时变量引起的麻烦

临时变量可以在组织块OB、功能FC和功能块FB中使用，当块执行时它们被用来临时存储数据，一旦块执行结束，堆栈的地址将被重新分配用于其它程序块使用，此地址上的数据不会被清零，直到被其他程序块赋予新值。
需要遵循“先赋值，再使用”的原则。

因此，有常见的几种情况导致程序运行不正常：

1. 某个块程序运行时好时坏，其中某个数值或多个数值偶尔不正常
   此问题在于，一定遵循“先赋值，再使用”。否则，TEMP的数值在每个扫描周期开始未有明确的赋值，此地址的数值将是随机的。
2. 多个块使用TEMP，单独使用任意一个都正常，无法一起正常使用
   此问题在于，TEMP未能先赋值，再使用；程序块1的TEMP中的数值并没有清零，而是CPU运行机制调用此地址使用或直接分配给程序块2使用，导致这个TEMP地址并不为0，因此程序混乱。
   由于内存运行机制并不公开，因此，这一分配过程看起来是随机的。这可能导致，程序多次运行情况下正常，运行一段时间后出现问题。
   只要遵循“先赋值，再使用”的原则，就可避免。
3. TEMP无法实现自锁
   此问题在于，TEMP数值无法像M点或Q点一样保持上一个周期的数值；TEMP需要在每个扫描周期有一个明确的赋值，即先赋值（写），再使用（读写）。
   解决方式，FB可使用STAT静态变量；FC可使用M区或全局DB地址。

总结，在使用临时变量TEMP时:

1.不能先使用，再赋值
2.不适用于自锁线圈
3.不适用于上升，下降沿

遇到如上情况，FC块可采用M区或全局DB地址；FB块也可采用自身背景DB的STAT静态变量
在FB，FC中使用第一次调用的某个临时变量，必须先对其赋值即写指令，而不能是读指令。