FAQ on SIMATIC

service & SUPPORT

STL 语言中的状态机编程

SIEMENS

FAQ

Entry-ID: 30362969

目录

这个文档来自于西门子 A&D 服务与技术支持。可以用下面的链接直接下载这个文档。

http://support.automation.siemens.com/WW/view/en/30362969

问：

在 STL 语言中如何创建一个 SIMATIC 状态机？

答：

用本文档中罗列的用法说明和注意事项来详细解答这个问题。

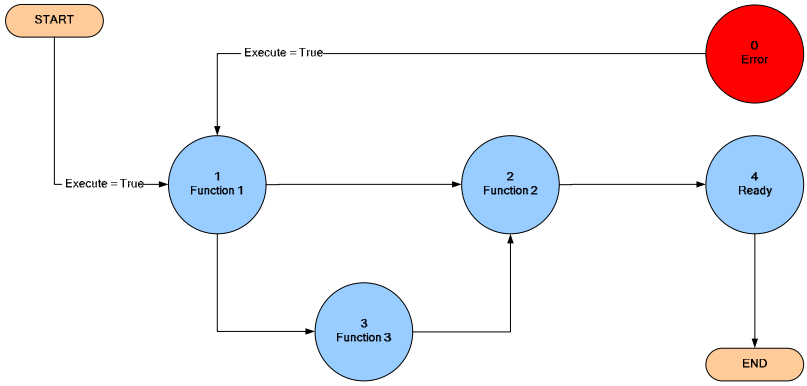
1 状态机

1.1 状态机组成

状态机由不同的状态组成, 包括不同的转换条件和启动条件来开始状态机

下面的插图做为一个状态机例子：

图 1-1 状态机结构



状态 1 到 3 每一个状态由编程来实现特定的功能。

利用输入信号 **Execute = True， 状态机开始第一个状态。**

任何状态下的引起的 **State 0 "Error" 也会通过 Execute = True 信号返回到状态 1**

在状态 1 可以设定到状态 2 或状态 3 的转换条件

1.2 状态机的优点

状态机有如下优点：

- 只有状态机中当前状态中已编程的程序代码是激活的。状态之间的程序影响是不可能的。
- 清楚的定义了状态之间的转换条件。 这就阻止了错误的状态传递。
- 在 STL 块中编程状态机结构清楚和简单。
- 改变程序代码只影响状态机中已经编程的状态。

1.3 状态机缺点

状态机有以下缺点：

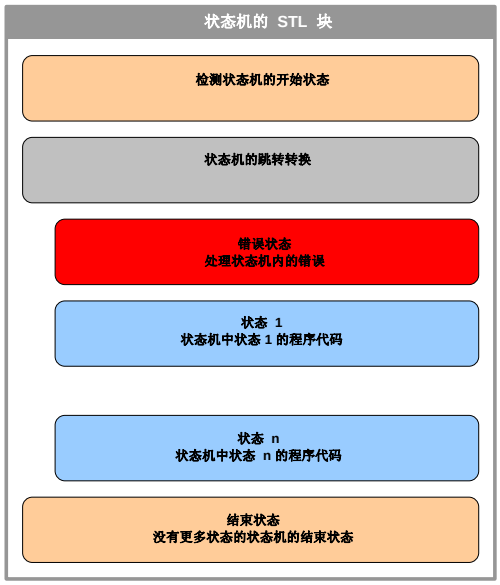
- 运行状态机要比用 STEP 语句编制相同的功能稍慢。

2 STL 状态机编程

2.1 程序的基本结构

STL 状态机编程的基本结构由如下部分组成：

图 2-1 基本结构



2.2 STL 中的状态机

请参考 FAQ 附带中的 step7 文档中所包含的 STL 块。

2.2.1 检测开始条件 （ 状态 1 ）

在网络 1 中检测布尔量 **Execute，如果上升沿** (False ⇨ True) **到来那么将执行这个网络中的程序代码。**

图 2-2 网络 1 – 检测开始条件

```
Network 1: Global: Start Execute

Comment:

      U     #Execute                    //Check positive edge
      FP    #FP_Execute
      SPBN  NoFP

//--- Start Sequencer ------------------------------------------------------

      L     1
      T     #Sequencer                  //Start Sequencer at Step 01

      SET
      R     #OK_Step01_02
      R     #OK_Step02_04
      R     #OK_Step03_02

//--- No positive edge -----------------------------------------------------

NoFP: NOP   0
```

描述当前激活状态的状态机的顺序变量 **Sequencer** 装载数值 1 （ 状态 1 ）开始运行状态机。在随后的状态中会复位决定状态转换的变量。

2.2.2 跳转分支（网络 2）

跳转指令是状态机的核心元素。

图 2-3 网络 2 – 跳转分支

```
Network 2: Global: Sequencer

State-Machine:

                +------> 02 --> 04 --> END
                |        |
  START --> 01 -+        |
                |        |
                +--> 03 -+

  Act. Step  :  Next Step      Act. Step  :  Next Step
  -----------------------      -----------------------
    01       :  02, 03           03       :  02
    02       :  04               04       :  END

      L     #Sequencer
      SPL   SQEr                        //Error  : Sequencer fault
      SPA   SQ00                        //Step 00: Error
      SPA   SQ01                        //Step 01: Function 1
      SPA   SQ02                        //Step 02: Function 2
      SPA   SQ03                        //Step 03: Function 3
      SPA   SQ04                        //Step 04: Ready

SQEr: CALL  "STP"                       //Incorrect sequencer-step selected
```

在跳转分支中，顺序变量决定了状态机进入到那个独立的状态。

跳转分支和基本的程序工艺设定了独立的状态，保证当前激活的状态的程序代码是唯一被激活的。

由跳转标号来对应跳转分支中所有的状态，因此他们可以由设定顺序变量的值来实现分支跳转。

顺序变量超过范围

当顺序变量超过跳转标号的值时，将由跳转标号中的 **SQEr 标号**转入错误状态。

在这种情况下 CPU 会由系统功能块 SFC46 转入 "STP" 状态。

2.2.3 状态机的状态 (网络 3-22)

这四个网络的主要组成是由状态机的不同状态组成。

状态启动 – 跳转目标

已经在自己的网络中清楚的编程了跳转分支的跳转目标和开始的状态。

图 2-4 跳转目标

```
Network 7: Step 01: Function 1
Comment:

SQ01: NOP   0
```

状态的程序代码

在下一个网络中编制当前状态所要执行的程序代码。

图 2-5 状态的程序代码

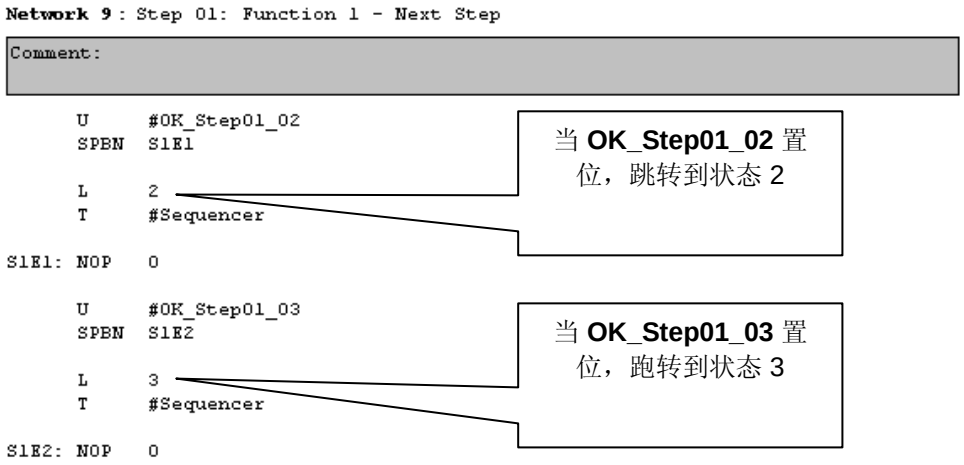
```
Network 8: Step 01: Function 1 - Your Program Code
Program Code of Step 01 - HERE
```

在这个网络当中可以插入状态所需要的程序代码。这个程序代码可以分解成几个连续的网络。

转换条件

在这个状态的最后要对的转换条件赋值。

图 2-6 转换条件

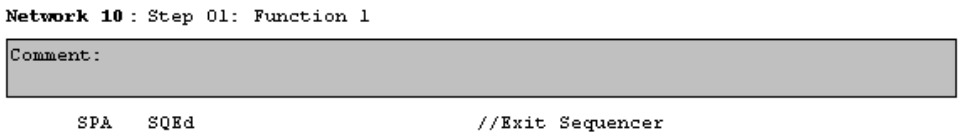


这的顺序变量是由程序设定的子分支状态编号来确定。(比如 **K_Step01_02** or **OK_Step01_03**)。如图所示，几个分支也可以按一定标准简单的编程

结束状态 – 跳转到结束状态

最重要的是在每一个状态的最后都需要一个绝对的跳转到状态机的最后，以避免其他那些状态机中未被处理的状态运行。

图 2-7 结束状态



最终的状态机是由块中的最后的网络来完成的。

图 2-8 状态机的结束状态

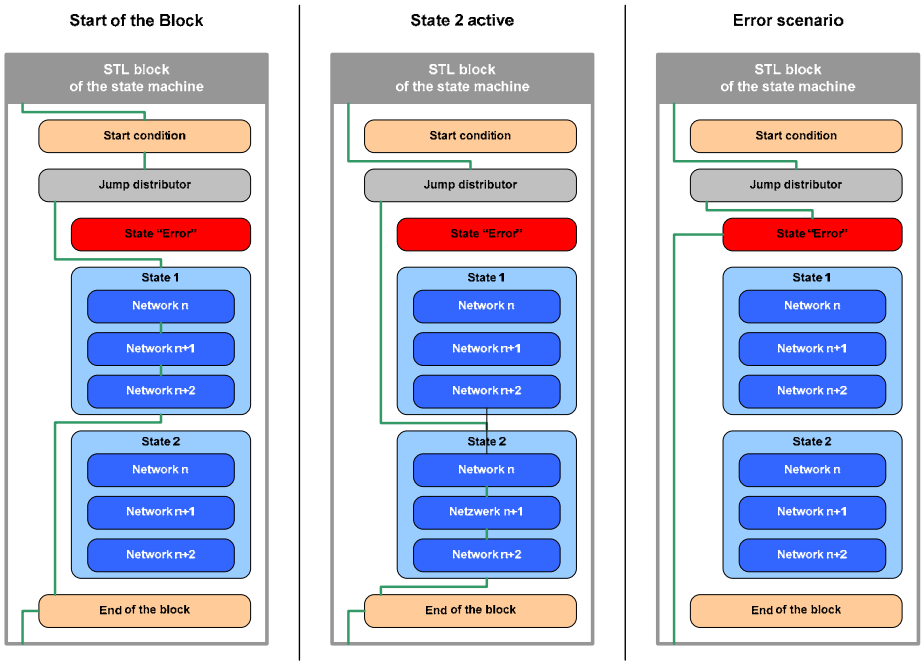
这保证了只有当前被激活的状态机中由顺序变量所选择的状态被处理。

3 概要

3.1 编制程序块的执行过程

总之，需要使用图形化编程来处理在由 STL 块编制的状态机。

图 3-1 所编制的状态机程序的处理过程



开始

当通过条件节 **Execute = True** 启动块时，启动条件只执行一次而且状态机设定到状态 1。

状态 2 激活

单独的状态间的转换只能通过顺序变量使用跳转分支实现。

错误发生时

万一发生错误时顺序变量可以设定到数值 0 。这时状态机会进入状态 0 **“Error”** 而且在通过变量 **Execute** 重启状态机之前保持。

注意

在跳转分支进入状态 0 “Error” 时，在当前顺序变量被 0 覆盖前，尽可能将当前的状态值保存到另外一个变量中，这样做可以更容易的诊断发生的错误。

3.2 扩展例子中更多的状态

为了扩充在 STEP 7 文档中包含的例子程序，按如下操作：

表 3-1 调整相同层次的转换结构

编号 No.	步骤说明	注
1	将状态 1 的网络 7 到 10 做为模板复制到网络 23 以前。复制你希望扩展的状态数目。	保证状态机的结束状态 (跳转标号 **SQEd**) 在插入网络的后面。
2	调整插入的状态前的跳转标号 (**SQxx**) 为新的状态标号。	
3	在网络的标题栏适当调整最新插入的状态 (**Step xx**)。	
4	在转换条件中调整新插入状态的跳转标号 (**SxxE**) 。	请考虑几个分支选项的跳转标号。
5	在转换条件中你可以调整后一个状态的数值（装载入顺序变量的值）。	
6	在状态机开始的跳转分支网络 2 中增加所插入的新的状态的标号。	
7	在网络 1 中必须尽可能的增加新状态的初始化程序。	

状态机已经可以使用和下载到控制器中，随后，可以开发状态机的不同状态的程序代码。

4 记录

表 4-1 记录

版本	日期	更正
V1.0	23.07.2008	第一次出版