SIEMENS

SIMOTION 的任务执行机制及系统时钟

SIMOTION Task Execution System and System Clock

Getting-started	Edition (2010 年- 6 月)

摘　要 本文对 SIMOTION 系统中的任务执行系统和系统时钟的设置进行了介绍。

关键词 SIMOTION, 执行系统, 执行等级, 系统时钟

Key Words SIMOTION, Execution System, Execution Level

目　录

1 SIMOTION 执行系统 4
1.1 任务执行等级 4
1.2 任务（Task） 4
1.2.1 StartupTask 4
1.2.2 自由运行任务 5
1.2.3 时间驱动任务TimerInterruptTasks 5
1.2.4 同步任务SynchronousTasks 5
1.2.5 事件驱动任务 6
1.2.6 ShutdownTask 7
1.3 任务的执行 7
1.4 任务的优先级 11
2 执行系统的配置 12
2.1 分配程序到执行等级和任务中 12
2.2 选择时钟信号源 14
2.2.1 系统时钟周期 14
2.2.2 系统时钟周期的设置 15
2.3 给TO（工艺对象）分配系统时钟周期 16
2.4 任务运行时间 17
2.5 自由执行任务的时间分配 17
2.5.1 自由执行任务等级的时间分配 18
2.5.2 举例说明 19

1 SIMOTION 执行系统

SIMOTION 执行系统规定了 SIMOTION 系统中的程序是如何被执行的。

1.1 任务执行等级

任务执行等级定义了执行系统中的程序执行的时间顺序，每个执行等级包含一个或几个任务（Task）。执行等级包括：

- 同步执行等级（Synchronous execution levels）：与控制或者插补时钟周期同步
- 时间驱动的执行等级（Time-driven execution levels）：时间触发任务
- 事件驱动的执行等级（Event-driven execution levels）：事件触发任务
- 中断控制的执行等级（Interrupt-controlled execution levels）：中断触发任务
- 顺序执行等级（Sequential execution levels）： 顺序执行的任务
- 自由运行执行等级（Free-running execution levels）：自由循环执行的任务

1.2 任务（Task）

每个执行等级包括一个或多个任务（Task）。每个任务在满足一定的条件时执行。用户可以指定多个程序到某个任务，还可以调整程序在任务中的执行顺序。

除了用户程序任务外，还有系统任务，系统任务的执行内容和执行顺序不可改变。

SIMOTION 的任务包括：

- 系统任务（ System tasks ）

– 通讯

PROFIBUS, PROFINET IO 网络的连接及 IO 处理。非周期通讯，trace 等。

– 运动控制

包括 IPO/IPO_2, position control (servo)，当使用工艺包时，系统自动分配执行系统。用户 程序不会影响工艺程序的执行。

– 温度控制程序

- 用户程序任务 (user program tasks)

在用户程序任务中可以执行运动控制, 逻辑和工艺函数等。用户程序任务包括：

1.2.1 StartupTask

当 SIMOTION 运行模式从 STOP 或 STOPU 到 RUN 时触发。用于变量的初始化和工艺对象的复位。在这个任务中，由于工艺对象正在初始化，不能执行运动控制命令。当此任务执行时，除了 SystemInterruptTask 其他的程序都不执行。

此任务结束，达到 RUN 模式后，启动下面的任务：

- SynchronousTasks
- TimerInterruptTasks
- MotionTasks
- BackgroundTask

1.2.2 自由运行任务

自由运行任务在自由执行等级中执行, 包括 MotionTasks 和 BackgroundTask 。

- MotionTasks

MotionTasks 用于执行顺序执行的命令，例如运动控制的命令等。共有 32 个 MotionTasks(MotionTask_1 到 MotionTask_32)。MotionTasks 通常通过用户程序的任务控制命令例如_startTaskID，_stopTaskID 来启动或停止任务。也可以通过设置使之在达到 RUN 模式时自动启动。还可以通过_getStateOfTaskID 命令查询任务的状态。

MotionTasks 只执行一次，没有事件监控，也就是说 MotionTasks 中的程序可以无限期的执行。

MotionTasks 在执行完或者是系统达到 STOP 或 STOPU 模式时停止。如果有等待命令（Wait for condition /WAITFORCONDITION），任务将被挂起，设置的条件在 IPO 周期内被检查，当条件满足时任务被继续。WAITFORCONDITION 和 ENDWAITFORCONDITION 之间的程序有更高的优先级(在 SystemInterruptTasks 和 TimerInterruptTasks 之间)。

- BackgroundTask

BackgroundTask 用于非固定周期的循环程序的执行。Start-up 任务后开始执行，在程序结束时自动重新执行。适于执行后台程序或逻辑处理程序等。

BackgroundTask 的循环时间被监控，一旦超时，TimeFaultBackgroundTask 被触发，如果此任务中没有分配程序则会造成 CPU 进入 STOP 模式。

1.2.3 时间驱动任务 TimerInterruptTasks

- TimerInterruptTasks

用于执行有固定循环周期的任务。在程序执行结束后自动重新执行。

TimerInterruptTasks 用于周期性程序的执行。有 5 个 TimerInterruptTasks，TimerInterruptTask_1 到 TimerInterruptTask_5。

TimerInterruptTasks 在固定的周期内被循环触发，这个周期要设为插补周期的倍数。

在此任务中可以实现闭环控制或者监控功能程序。

1.2.4 同步任务 SynchronousTasks

SynchronousTasks 与指定的系统时钟同步。

包括下列的同步任务：

- ServoSynchronousTask (软件版本4.0以上)：与伺服周期同步，在此任务中可以运行对时间有严格要求的任务。例如对I/O的快速响应程序，PROFIBUS DP通讯数据的同步处理，伺服层面的设定值的修改。

- IPOSynchronousTask/IPOSynchronousTask_2：与IPO周期同步

在 IPOSynchronousTask 中，可以实现对时间有严格要求的功能，用户程序在插补之前运行。在此任务中可以执行一些对 TO 的操作。

1.2.5 事件驱动任务

当一个事件发生时，启动此类任务，执行一次后停止。

- SystemInterruptTasks

当一个系统事件发生时，SystemInterruptTasks被调用。

有下面的SystemInterruptTasks ：

- TimeFaultTask：TimerInterruptTask 超时时执行
- TimeFaultBackgroundTask：BackgroundTask 超时时执行
- TechnologicalFaultTask：TO发生故障时执行
- PeripheralFaultTask：发生 I/O错误时执行
- ExecutionFaultTask：执行程序错误时执行

如果 SystemInterruptTasks 被触发，如果没有分配程序会造成 CPU 停机。

最多可以有 8 个不同的中断被缓存，如果此时有另一个中断发生，缓存会溢出 CPU 停机。触发 SystemInterruptTasks 的事件可以由 TaskStartInfo 来查询。

启动 ExecutionFaultTask 的程序执行错误包括：

- 浮点数的错误操作，例如对负数取对数，错误数据格式等
- 除0
- 数组超限
- 访问系统变量错误

发生错误的任务将会被停止。

对于下面的任务发生了错误可以在 ExecutionFaultTask 中用命令重新启动：

- StartupTask
- ShutdownTask
- MotionTasks

对于下面的任务发生了错误，在ExecutionFaultTask结束后CPU达到STOP，启动ShutdownTask：

- BackgroundTask
- TimerInterruptTasks
- SynchronousTasks

ExecutionFaultTask 和 ShutdownTask 中的编程错误会立即导致系统停机。

- UserInterruptTasks

当一个用户自定义事件发生时，UserInterruptTasks被调用。

有两个用户中断任务：UserInterruptTask_1 和 UserInterruptTask_2。必须指定UserInterruptTask的条件，当条件满足时，UserInterruptTask被执行。如果同时被触发，UserInterruptTask_1 在 UserInterruptTask_2之前被执行。

如果使用 UserInterruptTask，那么也必须使用 IPOsynchronousTask。因为UserInterruptTask 的条件在 IPO 周期中被检查。

UserInterruptTask 在 StartupTask 和 ShutDownTask 期间不被执行。

1.2.6 ShutdownTask

ShutdownTask 在 CPU 从 RUN 模式到 STOP 或 STOPU 模式时被执行一次。可以执行例如设置输出点的状态或轴的停止命令等。此任务不会在系统失电时执行。

还需要设置 ShutdownTask 的监控时间，过了设置的时间后 CPU 会自动到 STOP 模式。

1.3 任务的执行

任务在执行等级中的分配如图 1。图 2 表示了 SIMOTION 中任务的执行顺序，由上而下。图 3 为 SIMOTION 任务在时间上的分配。

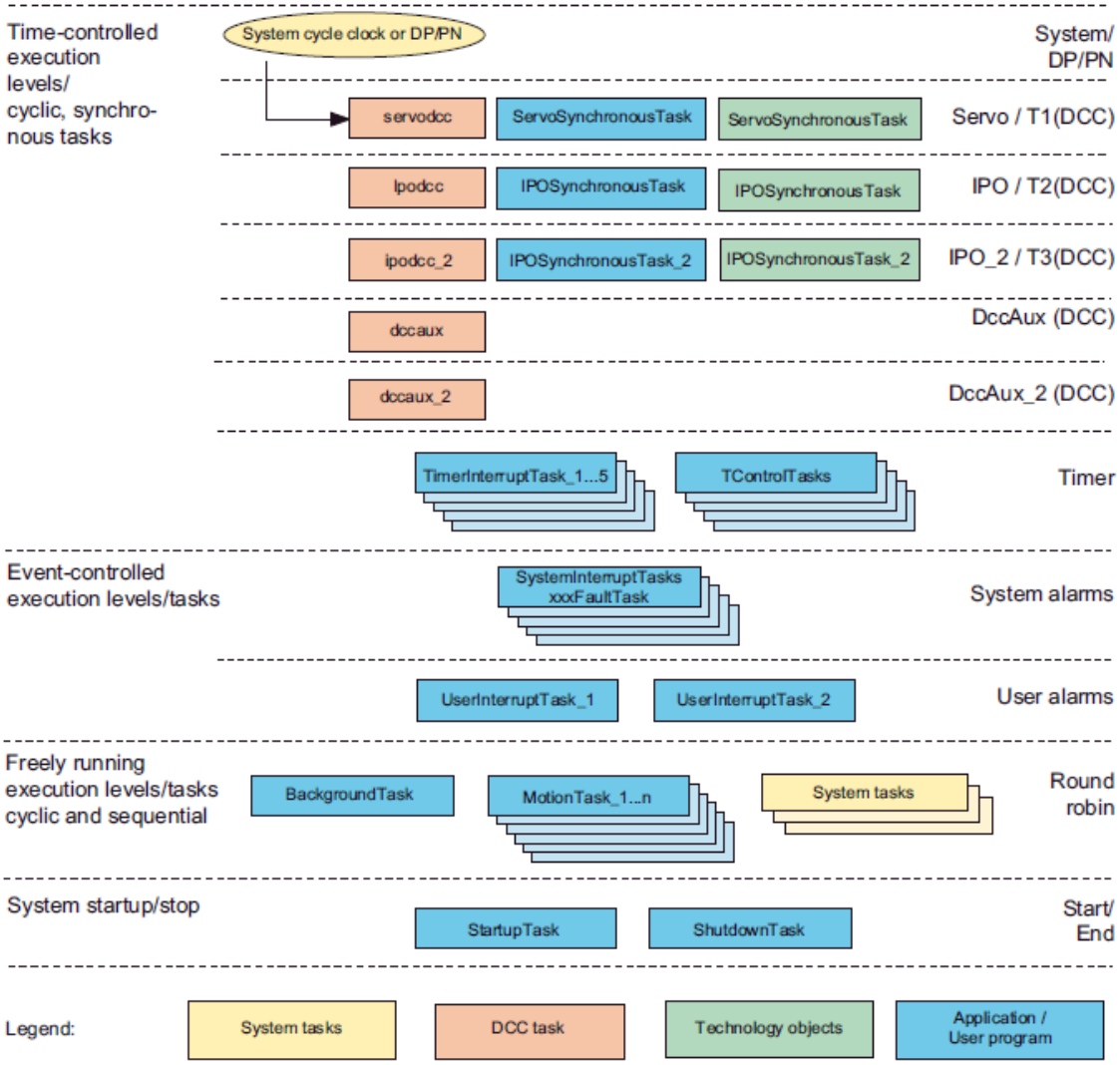


图 1　任务在执行等级中的分配

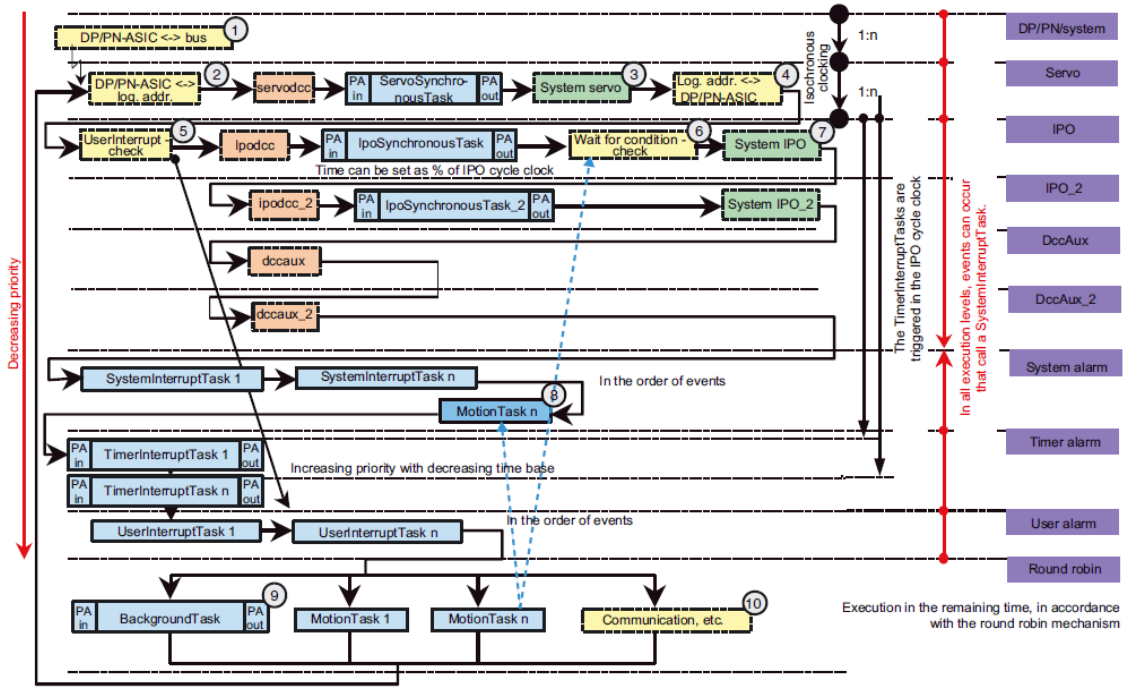


图 2 任务的执行顺序

对图 2 的注释：

DP, Servo 和 IPO 的比例是 1:1:1

蓝/绿：用户程序任务

黄：系统任务

1. DP/PN-ASIC <-> Bus

通讯的芯片和 PROFIBUS 或 PROFINET IO 进行数据交换。

2 DP/PN-ASIC -> log. addr

IO 输入数据从通讯芯片加载。

3. System servo

在伺服时钟周期内进行系统计算（位置控制器等），如果在此周期内程序不能计算完会造成溢出，进入 STOP 模式，禁止启动，并在诊断缓冲区中记录。只有在重新上电后或下载后才能再次启动。

4. Log. addr. -> DP/PN-ASIC:

I/O 输出写入到通讯芯片

5. UserInterrupt - Check:

检查用户中断程序的条件

6. Wait for condition - Check:

WAITFORCONDITION (等待轴，等待信号等) 命令的条件被检查。

7. System IPO/IPO_2:

IPO 中的系统程序（运动控制：定位曲线，同步操作等）

8. MotionTask n:

WAITFORCONDITION 等待中的 MotionTask 当条件满足时被优先执行(更高的优先级).

9. BackgroundTask:

Background 背景数据块 (PI) 在 BackgroundTask 开始时和结束后被刷新。如果程序的运行时间比较长，BackgroundTask 在运行时可能被高级的任务打断几次。

10. 通讯:

通讯功能 (HMI, PG/PC 等)

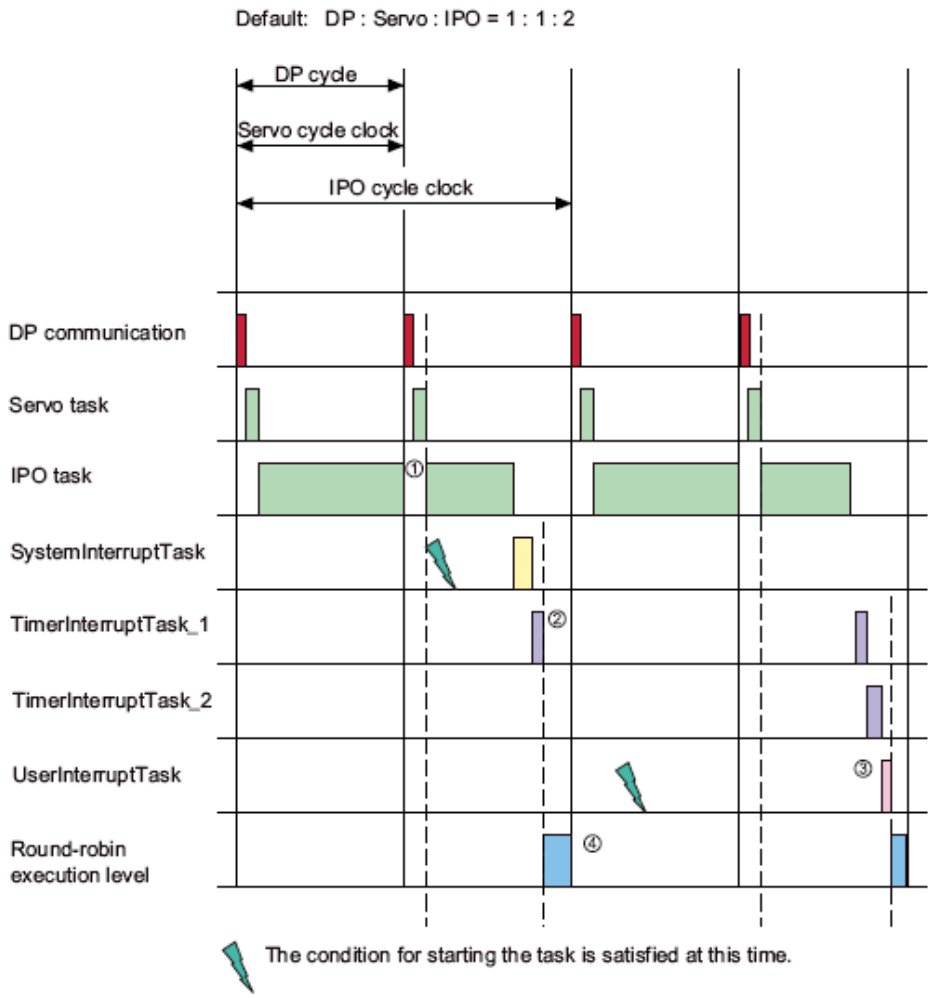


图 3　SIMOTION 任务的执行

1 IPO 任务中执行的程序比伺服周期时间长，此时，IPO 任务被中断然后执行 DP 通讯和伺服任务。然后 IPO 任务继续执行。

2 IPO 任务结束后, 执行 SystemInterruptTask ， 然后执行低优先级的 TimerInterruptTask 。

3 UserInterruptTask 即使在条件满足的条件下，也是在高优先级的任务结束后执行。

4 自由执行等级任务在剩余的时间执行。

1.4 任务的优先级

如果两个任务的程序在某个时刻同时执行，那么任务的优先级决定了哪个任务先执行。

优先级	执行等级	任务	周期(z)/顺序 (s)	注释
高	Servo, DP 通讯	-	z	
	Servo / T1(DCC)	Servodcc ServoSynchronousTask ServoTask	z	DCC 实时任务组中的任务 (T1) 伺服周期中的用户程序任务 系统任务
	IPO / T2(DCC)	Ipodcc	z	DCC 实时任务组中的任务 (T2)
		IPOSynchronousTask • 检查 UserInterrupt • 用户程序	z	插补周期的用户程序任务
		IPOTask • 检查 WAITFORCONDITION 的条件 (IPO,system)	z	IPO 周期中的系统任务
	IPO_2 / T3(DCC)	Ipodcc_2	z	DCC实时任务组中的任务(T3)
		IPOSynchronousTask_2	z	插补周期2的用户程序任务
		IPOTask2	z	IPO2 周期中的系统任务
	TControlTasks	PWMsynchronousTask	z	温度控制任务
		InputSynchronousTask_ 1	z	温度控制任务
		InputSynchronousTask_ 2	z	温度控制任务
		PostControlTask_1	z	温度控制任务
		PostControlTask_2	z	温度控制任务
		+ 附加的系统任务	z	
	DccAux (DCC)	Dccaux	z	DCC实时任务组中的任务(T4)
	DccAux_2 (DCC)	Dccaux_2	z	DCC实时任务组中的任务(T5)
低	SystemInterruptTasks	TimeFaultTask	s	系统报警 (按事件发生的顺序)
		TimeFaultBackgroundTask	s	系统报警 (按事件发生的顺序)
		TechnologicalFaultTask	s	系统报警 (按事件发生的顺序)
		PeripheralFaultTask	s	系统报警 (按事件发生的顺序)

		ExecutionFaultTask	s	系统报警 (按事件发生的顺序)
	TimerInterruptTasks	TimerInterruptTask1 ... TimerInterruptTask5	z ... z	定时器中断
	UserInterruptTasks	UserInterruptTask_1 UserInterruptTask_2	s s	用户中断(按事件的顺序)
	Round robin	BackgroundTask	z	用户程序任务(不能指定顺序)
		MotionTask_1 ...MotionTask_32	s ... s	用户程序任务(不能指定顺序)

表 1 任务的优先级

任务的优先级不能由用户改变。注意：

1 对于 TimerInterruptTasks: 设定的时间越短优先级越高

2 所有的 UserInterruptTasks 优先级是一样的，按照触发的顺序一一执行。

3 Wait for condition / WAITFORCONDITION 命令可以暂时提高 MotionTask 的优先级。

- 条件的检查优先级与 UserInterruptTasks 一样。

- 条件满足时，之前被挂起的 MotionTask 被重新使能

- WAITFORCONDITION 和 ENDWAITFORCONDITION 之间的命令在更高的优先级内执行(SystemInterruptTasks 和 TimerInterruptTasks 之间)。

2 执行系统的配置

2.1 分配程序到执行等级和任务中

用户的程序必须分配到执行等级和任务中才能执行。可以分配 MCC，ST 或者 LAD/FBD 程序到一个或多个任务中。也可在一个任务中分配多个程序。

分配的程序按照列表中的顺序依次执行，此顺序可以在 SCOUT 中指定。后面的程序必须在前面的程序结束后才能执行。

可以将一个程序分配到几个任务中，此时他们在不同任务中独立运行。

当分配一个程序时，也就定义了该程序的执行优先级，执行模式是顺序执行还是循环执行以及程序变量的初始化。

分配一个程序到一个或多个任务时应注意：

- 被分配的程序必须被编译过且没有错误

- 在下载程序之前进行分配

- 当一个程序在执行时，有可能被另外一个任务调用。这时系统不能保证数据的一致性。

- 当分配了一个程序到任务后，即使重新编译程序也会保持分配状态。

在 Scout 项目浏览界面中选择 EXECUTION SYSTEM 打开任务配置界面。

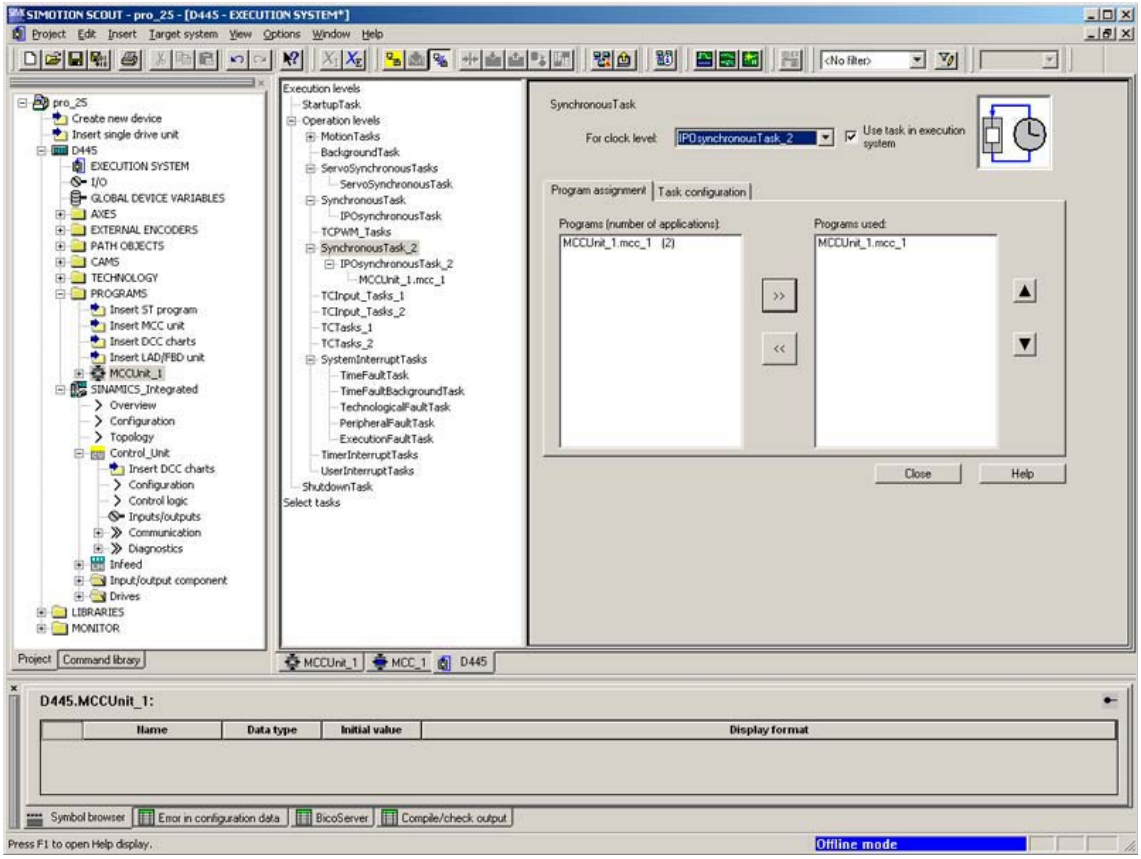


图 4 执行系统任务配置界面

在任务配置的左侧可以看到执行等级树，在每个执行等级下有分配的任务和程序的列表。

给任务分配程序的步骤：

1 首先选择要分配程序的任务

2 选择 Program assignment 标签项，在左侧的窗口会列出所有可分配的程序

3 在左侧的窗口选择要分配的程序

4 选择  ，选定的程序出现在右边的窗口中。

5 在右侧窗口中选中相应的任务，按上下按钮可以调整程序的运行顺序。

6 还可以在 Task configuration 标签中进行更多的设置，例如

- 程序出错的处理方式
- 周期任务的看门狗时间
- MotionTasks 的启动模式等

全部设置完成后，可以建立连接然后下载到目标系统中。

2.2 选择时钟信号源

在 SIMOTION 硬件配置时就确定了时钟的信号源。如果 SIMOTION 的某个接口被设置成了等时同步 DP/PN 模式，则时钟周期的设置被用做总线时钟周期。DP/PN 的通讯，伺服以及插补周期均与总线时钟周期同步。如果想要等时同步地访问 IO 变量，必须进行此设置。支持等时同步的驱动设备有：SIMODRIVE 611U, MASTERDRIVES MOTION CONTROL, SINAMICS。

也可以连接不支持等时同步模式的驱动，例如 Micromaster MM4 和 MASTERDRIVES VC。

如果未设置等时同步模式，就可以设置基本的系统时钟。伺服和插补周期与基本的时钟同步。

2.2.1 系统时钟周期

一旦选择了时钟周期的源，就可以定义各个同步的周期时间，他们是基本时钟周期的倍数。

- Bus cycle clock (DP cycle clock / PN cycle clock)：系统时钟周期
- Servo cycle clock (position control cycle clock) / T1(DCC) cycle clock

输入输出在此周期内被刷新。包括轴的位置控制以及集中式 IO 或分布式 IO 的处理。伺服时钟周期与总线周期的比例可以是 1：1 或 2：1。

设置此周期时要注意：

- 伺服周期要设置成 DP 的总线时间，因此所有的 DP 总线的时钟周期要一致。

- 对于 SIMOTION C P D，外部的 DP 总线周期至少为 1ms

- 对于等式同步的 PROFINET 来说，如果设置的时钟周期比 DP 的周期短，例如 0.5ms，那么伺服周期必须与 DP 的周期一致（对于 SIMOTION D 包括 PROFIBUS Integrated）

- 伺服周期与总线周期的比率也要在驱动设置里的应用周期中设置。 此设置对于生命周期的监控是必须的。

- Interpolator cycle clock (IPO cycle clock) / T2(DCC) cycle clock

轴运动是在 IPO 时钟周期内被计算。IPOSynchronousTask 也是在此周期内执行。IPO 周期与伺服周期的比率可以设置成 1:1 到 1:6。

- Interpolator cycle clock 2 (IPO_2 cycle clock) / T3(DCC) cycle clock

IPO_2 时钟周期可以运行低优先级的轴。此外 IPOSynchronousTask_2 和 PWM Task (TControl) 也在此周期内执行。

IPO_2 与 IPO 时钟周期的比率可以设置成 1:2 到 1:64。

● DccAux (DCC) cycle clock

DCCAux 时钟周期与 T3(DCC) 时钟周期的比率可以设置成 1:2 到 1:32。

● DCCAux_2(DCC) cycle clock

DCCAux_2 时钟周期与 DCCAux 时钟周期的比率可以设置成 1:2 到 1:32。

● PWM cycle clock: TControl 工艺应用相关的时钟设置

2.2.2 系统时钟周期的设置

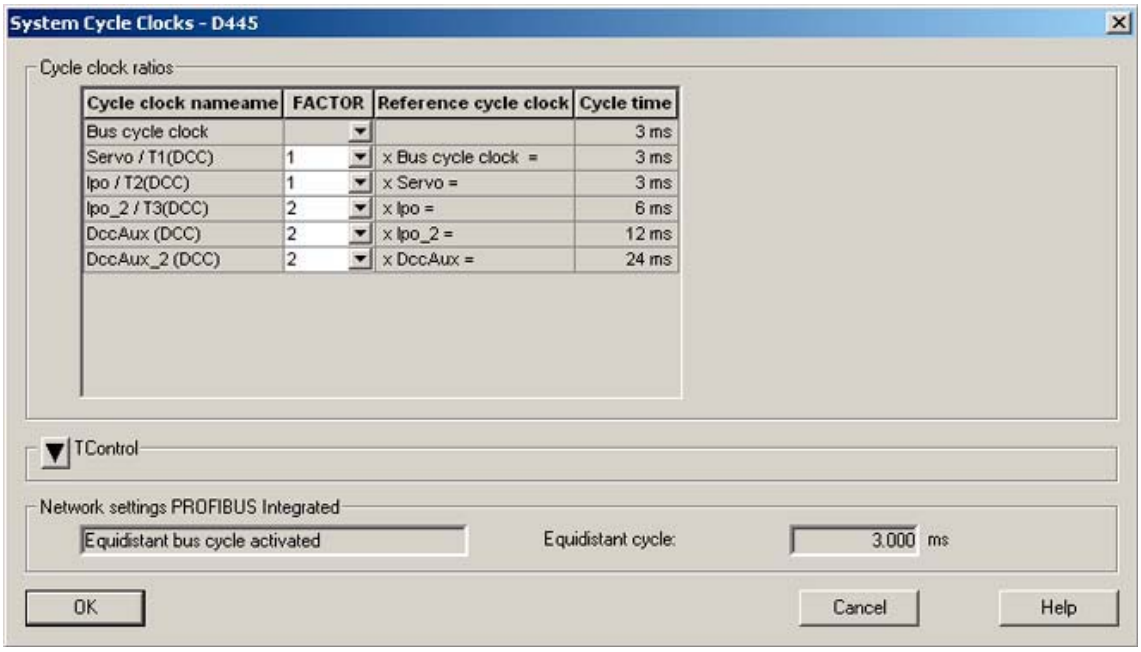


图 5 系统时钟周期的设置

系统时钟周期的设置步骤：

1 通过菜单 Target system > Expert > Set system cycle clocks...，或者右击 EXECUTION SYSTEM 选择 Expert > Set system cycle clocks 打开设置页面。

2 如果没有设置等时同步模式则可以指定基本的总线时间。

PROFIBUS 可能的设置：1.0…8.0

PROFINET 可能的设置：

- V4.0 开始带 PROFINET 的 SIMOTION P 和 SIMOTION D: 0.5 ... 4.0 ms

- V4.1 开始带 PROFINET 的 SIMOTION P: 0.25 ms 到 4 ms。

PROFIBUS 的总线时钟周期 是 0.125 ms 的倍数， C2xx 是 0.25 ms。PROFINET 是 0.125ms 的倍数。

3 设置各个时钟之间的比例关系。

4 如果使用了 TCONTROL 则需对 TCONTROL 系统任务以及时钟周期进行设置。

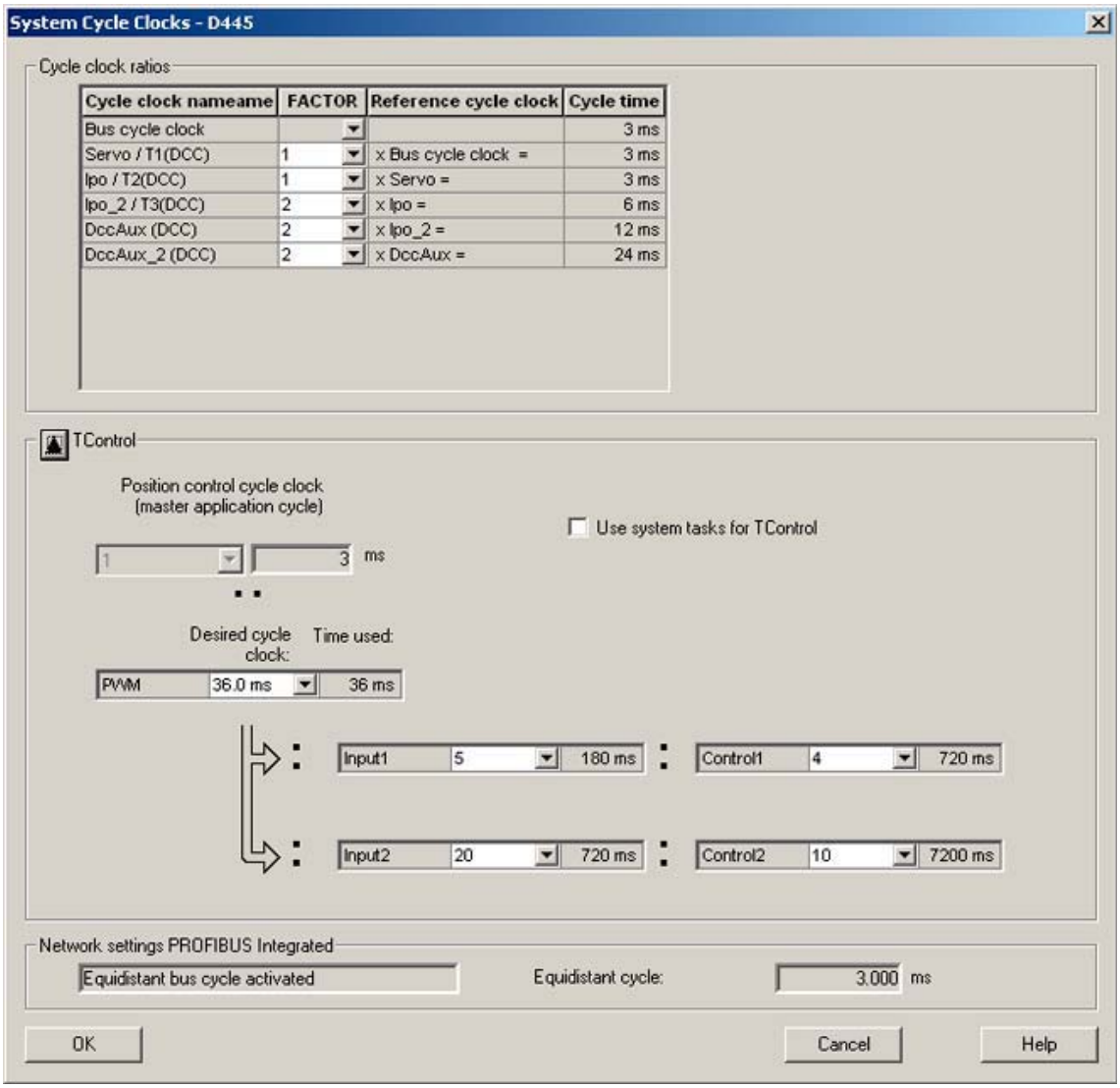


图 6 系统时钟设置

2.3 给 TO（工艺对象）分配系统时钟周期

可以分配 TO 的计算周期来改变 TO 的优先级以及优化系统的性能。一般情况下，可以分配低优先级的工艺对象到 IPO2，例如外部编码器。分配 IPO 或伺服时钟周期到高优先级的 TO。可以设置的时钟周期如下表。

运动控制任务	高优先级	...	低优先级
工艺对象	伺服周期时间	插补时钟周期	插补时钟周期2
Drive axis		默认	X
Positioning axis		默认	X
Synchronous axis		默认	X
External encoder		默认	X
Output cam	X	X	X

Cam track	X	X	X
Measuring input	X	X	X

表 2　工艺对象可以分配的系统时钟周期

2.4 任务运行时间

可以用任务运行时间检查系统的性能是否可以满足要求。在设备的变量 Taskruntime 和 effectiveTaskruntime 中可以查看任务的运行时间。Taskruntime 是指任务运行的时间（不包括中断的时间），effectiveTaskruntime 显示实际执行等级的执行时间，从执行等级开始到最后的任务结束所需的时间。如图所示：

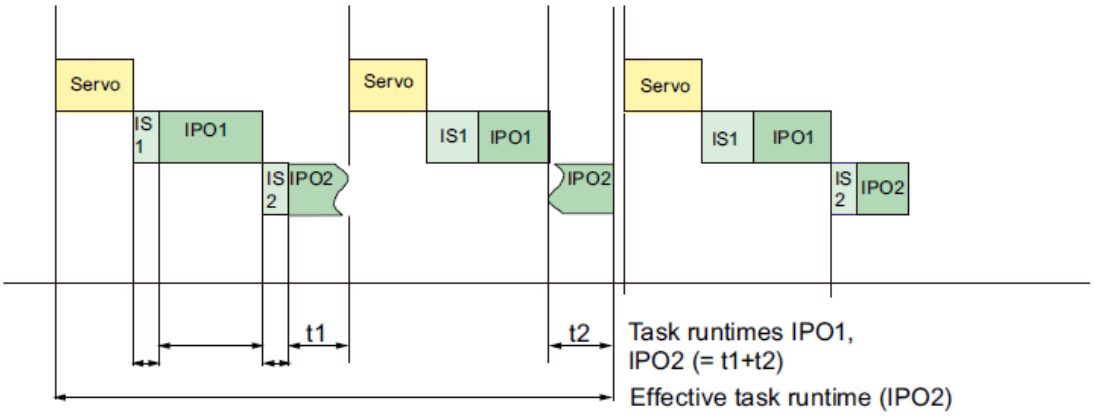


图 7　任务的运行时间

图中：

Servo: 伺服循环周期

IS1: IPOSynchronousTask

IS2: IPOSynchronousTask_2

IPO1: IPOTask

IPO2: IPOTask2

系统运行时可能会发生超时或溢出。可以用 SCOUT 的诊断功能来监控程序的运行。

2.5 自由执行任务的时间分配

除了高优先级的用户和系统任务外剩余的时间是留给 MotionTasks 和 BackgroundTask 的。在自由执行任务级别自由的顺序执行这些任务。由于是循环执行，因此执行时起始顺序是不确定的。在此循环中一个任务可以在指定的时间内连续执行，然后转到下一个任务，如果任务执行结束就会直接转到下一个任务，如图 8 所示。可以通过设置来决定 MotionTasks 和 BackgroundTask 在自由执行任务级别中执行时间的分配。

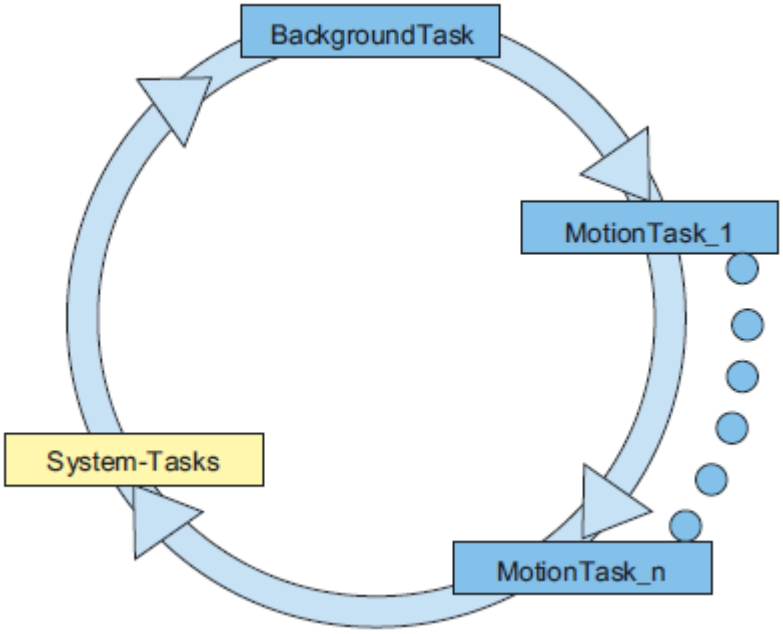
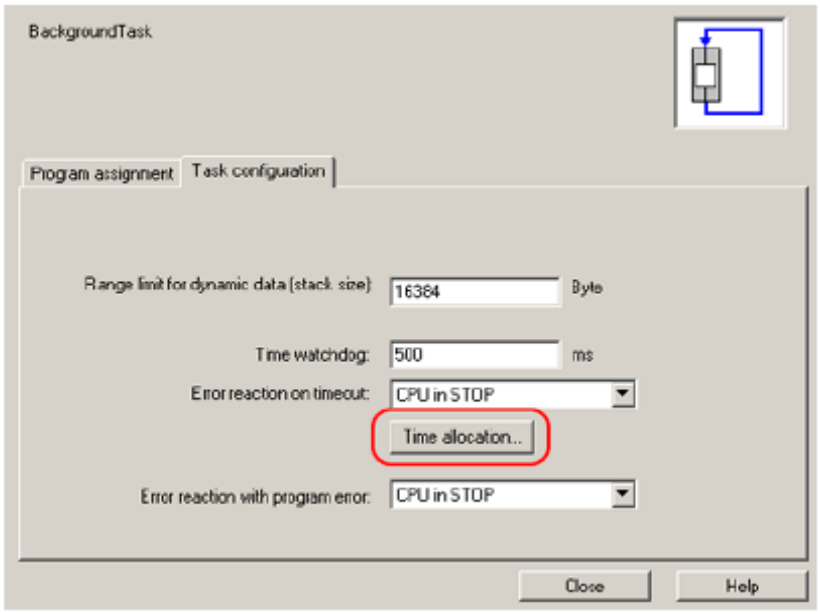


图 8　自由时间分配

2.5.1 自由执行任务等级的时间分配

1. 在任务分配窗口中选择 MotionTask 或 BackgroundTask，选择任务配置标签。
2. 点击时间分配按钮 Time allocation.
3. 在打开的窗口中用滑动条设置 BackgroundTask 的时间分配。
4. 点击 OK 确定。



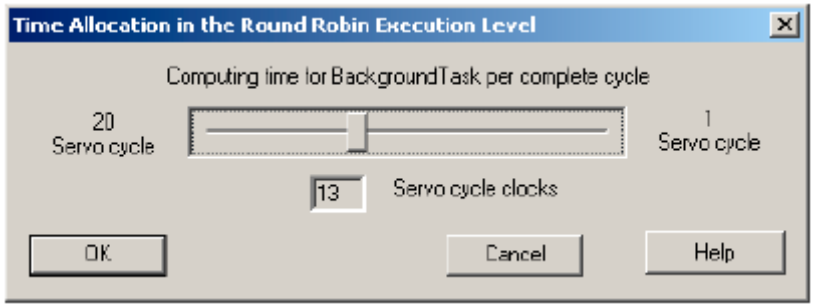
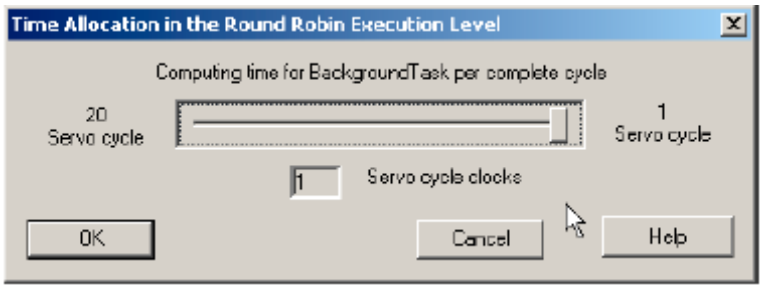


图 9　时间分配

2.5.2 举例说明

通过两个例子说明自由执行等级时间的设置。

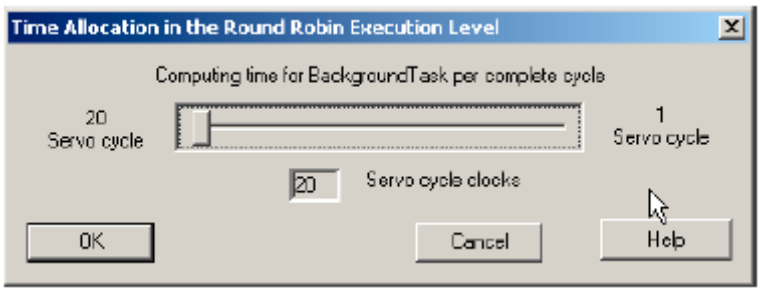
1.如果设置 BackgroundTask 的运行时间为 1 个伺服周期如下图所示。在此设置下，BackgroundTask 执行一个伺服周期，然后运行所有的 MotionTask，但每个 MotionTask 最多运行 2 个伺服周期。然后 BackgroundTask 再执行一个伺服周期。



例如有两个 MotionTask 的情况下，自由执行等级的执行情况。

Servo cycle clock 1	Servo cycle clock 2-3	Servo cycle clock 4-5	Servo cycle clock 6	Servo cycle clock 7-8
Background Task	MotionTask 1	MotionTask 2	Background Task	MotionTask 1

2. 如果设置 BackgroundTask 的运行时间为 20 个伺服周期如下图所示。在此设置下，BackgroundTask 执行 20 个伺服周期，然后运行所有的 MotionTask，但每个 MotionTask 最多运行 2 个伺服周期。然后 BackgroundTask 再执行 20 个伺服周期。



Servo cycle clock 1-20	Servo cycle clock 21-22	Servo cycle clock 23-24	Servo cycle clock 25-40	Servo cycle clock 41-42
Background Task	MotionTask 1	MotionTask 2	Background Task	MotionTask 1

如果您对该文档有任何建议，请将您的宝贵建议提交至下载中心留言板。

该文档的文档编号：**A0471**

附录一推荐网址

驱动技术
西门子（中国）有限公司
工业自动化与驱动技术集团 客户服务与支持中心
网站首页：www.4008104288.com.cn
驱动技术 **下载中心**：
http://www.ad.siemens.com.cn/download/DocList.aspx?TypeId=0&CatFirst=85
驱动技术 **全球技术资源**：
http://support.automation.siemens.com/CN/view/zh/10803928/130000
“找答案”驱动技术版区：
http://www.ad.siemens.com.cn/service/answer/category.asp?cid=1038

注意事项

应用示例与所示电路、设备及任何可能结果没有必然联系，并不完全相关。应用示例不表示客户的具体解决方案。它们仅对典型应用提供支持。用户负责确保所述产品的正确使用。这些应用示例不能免除用户在确保安全、专业使用、安装、操作和维护设备方面的责任。当使用这些应用示例时，应意识到西门子不对在所述责任条款范围之外的任何损坏/索赔承担责任。我们保留随时修改这些应用示例的权利，恕不另行通知。如果这些应用示例与其它西门子出版物(例如，目录)给出的建议不同，则以其它文档的内容为准。

声明

我们已核对过本手册的内容与所描述的硬件和软件相符。由于差错难以完全避免，我们不能保证完全一致。我们会经常对手册中的数据进行检查，并在后续的版本中进行必要的更正。欢迎您提出宝贵意见。