1. 综述

(个人理解：真正的运动控制除了一些开关量，状态位，读写操作外，真正的涉及到运动的控制就必须控制电机，即轴的转动，在工控领域的直线、曲线、旋转各种运动其实都是由电机的转动产生的，所以本文叙述的运动控制分为单轴和多轴的运动控制功能块，主要都是对电机轴状态的控制，具体的实际运动类型可以有多轴配合运动而得到,标准提供的功能块是给厂商开发控制块进行的规范，即厂商可以给予不同的硬件来固化这些功能块，但由于标准接口的定义，用户只需学习标准块就可以进行编程了。也就是说不同厂商提供的功能块接口都是一样，这样用功能快变成的程序移植的时候只需要较少修改即可，因为他们对运动控制的定义、参数、接口都是一样的，也就是说，虽然具体功能块在具体的硬件上实现的方法可能不一样，但因为提供给用户的接口是一样的，所以程序可以在不同硬件之间移植，而开发这些功能块使用的就是 PLCopen 致力于推广的 IEC 61131-3)

运动控制市场提供了多种不兼容的系统和解决方案。在使用不同系统的企业，这种不兼容给最终的用户造成了相当大的损失，学习很困惑，设计变得很困难，并且减缓了市场的前进脚步。

标准化肯定会减少这些负面因素。标准化意味着不仅是编程语言本身（因为标准化已经实现了采用全球化的 IEC 61131-3 标准），而且要使面向不同运动控制解决方案的接口标准化。采用这种方法，这些运动控制解决方案的编程就会更少的依赖硬件。应用软件的可复用性增强了，并且减少了相关的培训和支持的费用 。

用户要求 PLCopen 帮助解决这个问题，PLCopen 发起建立了 Motion Control aSk Force。Task Force 通过使运动控制功能块的标准化来定义编程人员的接口。

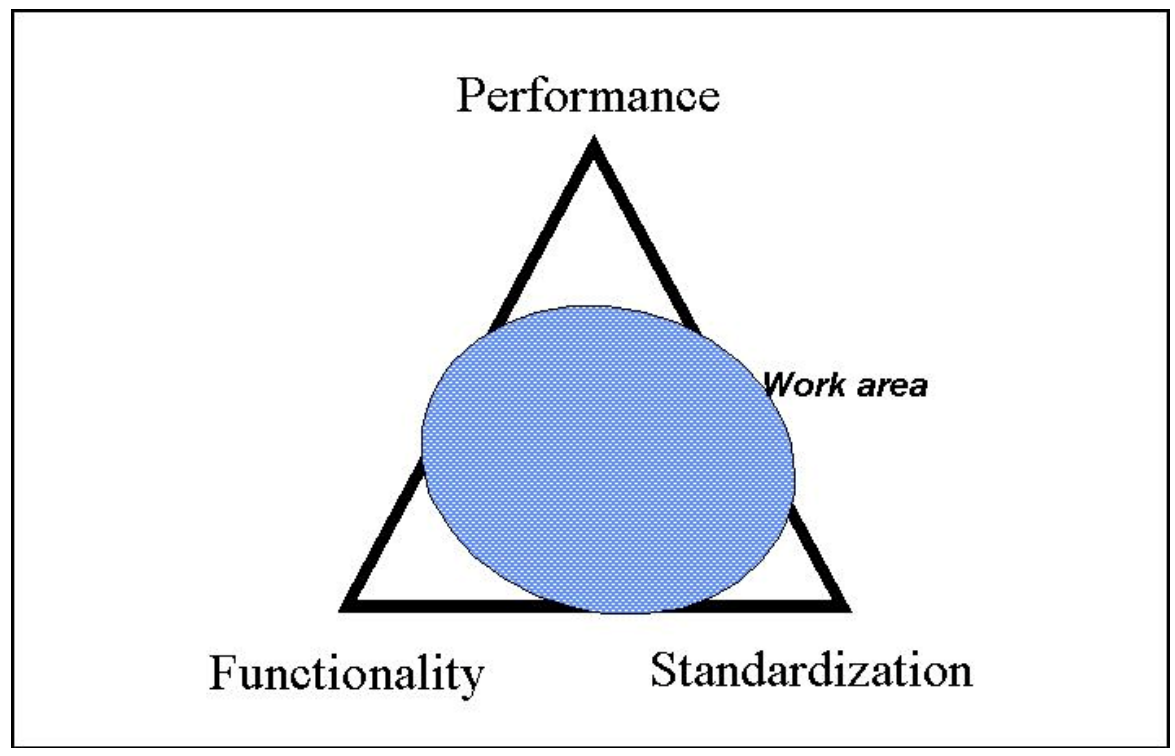


图 1：用户选择的三角关系

For the positioning of this action, 请检查图 1.这个三角关系在其角上有以下的用户选项：

- 性能
- 功能
- 标准化

实际上，用户编写它们的程序和具有专用功能的硬件紧密地结合，以尽可能获得由他们所处环境决定的最高的性能。这限制了用户在基于自己的硬件上的选择和控制软件的复用性，同时增加了培训的投资。

The second user option enables a very broad range of software functionality to be offered

这对用户非常有帮助，但是很少能获得很高的性能。也会使培训费用增加。
第三个角，标准化，最初的关注的是不同供应商的不同的系统的复用性，包括集成，分布式和网络系统，能很好的减少培训的投资。由于这个定义的一般特点， 基于不同架构的性能可能比不上硬件编码。由于这个原因，标准化不应该预期能提供最佳性能，但是可以非常接近于最佳性能，也就是说，三角型的底部很短。
已经发布的第一份说明书是一份独立的运动控制功能块的文库。它包括了单轴和多轴的运动功能，一些管理任务，和一份声明图表一样。这份说明书给用户提供了一套标准的指令集和独立于下层架构的结构。
这个结构可以在很多平台和架构上使用。这样一个人可以决定在开发循环中在哪一个阶段使用哪一个架构。对于机械制造商其优势在于，在其他人中，支持不同平台的花费更低并且可以自由的以更独立的方式开发应用软件，而且不会限制机器的生产率。除了这些好处之外，系统维护更加简单并且培训阶段更短。这是向前迈进的重要的一步，并且越来越多的被用户及供应商接受。

随着part1的发布，大家认识到需要增加额外的功能。Part 1为一套相互依存的说明书提供了基础：
Part 1-PLCopen 运动控制功能块
Part 2-PLCopen 运动控制-扩展，新发布的融合了Part 1的2.0
Part 3-PLCopen 运动控制-用户指南
Part 4-PLCopen 运动控制-调整运动
Part 5-PLCopen 运动控制-制导扩展
Part 6-PLCopen 运动控制-流体动力扩展

随着底层文档Part 1的发布—PLCopen 运动控制功能块2.0版，Part 2—PLCopen 运动控制扩展已经被整合到这个基础文档中。
PLCopen 运动控制用户指南，Part 3 是附加在PLCopen 运动控制功能块中的，不能被看做一份单独的文档。

1.1.目标（Objectives）

运动控制功能块适用IEC61131-3 语言是因为以下几点考虑：

1. 简易 —使用方便，面向应用程序的建立者、安装和维修
2. 效率 —功能块的数量，设计效率（和理解）
3. 一致性 —遵守IEC 61131-3标准
4. 普遍性 —硬件独立
5. 灵活性 —软件未来的升级/适用范围
6. 完备性 —非强制但完备

1.1.1. language context goals

- 关注功能块接口和行为的定义和符合IEC61131-3 规格的数据的类型
- 这些功能块和数据类型可以在所有的IEC61131-3 语言中使用
- 这份文档中的例子以文本和图解的IEC61131-3 语言的方式给出信息
- 功能块中的内容可用任意一种编程语言（例如IEC 61131-3 ST,C）或者甚至采用固件或硬件来实现。因此内容不应预计被跨平台移植（固件块不能移植）。
- 采用这些功能块和数据类型组成的可重复使用的应用程序简单的使用PLCopen 交换标

准

- 这份说明书应该被看成一份开放的无关硬件的框架。它基于不同的平台，如完全集成，集中或分布式系统在实现上提供了开放性。实际功能块的实现其本身已经超出这个标准的范围。

1.1.2. 一套功能块的定义

一个基本问题涉及到标准功能块的粒度和模块性。The extremes are one Function Block per axis versus a command level functional unit. 上述的目标可以很方便的由模块化设计的功能块实现。模块化创建了一个更高水平的可扩展性，灵活性和可重构性。可以由此创建大型块（派生功能块），例如：易于应用程序的构建和浏览的整个轴。
如果可行的话，这里指定的功能块可以以函数的方式实现（例如MC_ReadParameter).

1.1.3. 综述定义了的功能块

下面的表格给出了定义了的功能块的概览，分为管理（不是驱动运动）和运动相关的集和。

Administrative		Motion	
Single Axis	Multiple Axis	Single Axis	Multiple Axis
MC_Power	MC_HaltSuperimposed	MC_Home	MC_CamIn
MC_ReadStatus		MC_Stop	MC_CamOut
MC_ReadAxisError		MC_Halt	MC_GearIn
MC_ReadParameter		MC_MoveAbsolute	MC_GearOut
MC_ReadBoolParameter		MC_MoveRelative	MC_GearInPos
MC_WriteParameter		MC_MoveAdditive	MC_PhasingAbsolute
MC_WriteBoolParameter		MC_MoveSuperimposed	MC_PhasingRelative
MC_ReadDigitalInput		MC_MoveVelocity	MC_CombineAxis
MC_ReadDigitalOutput		MC_MoveContinuousAbsolute	
MC_WriteDigitalOutput		MC_MoveContinuousRelative	
MC_ReadActualPosition		MC_T orqueControl	
MC_ReadActualVelocity		MC_PositionProfile	
MC_ReadActualTorque		MC_VelocityProfile	
MC_ReadAxisInfo		MC_AccelerationProfile	
MC_ReadMotionState			
MC_SetPosition			
MC_SetOverride			
MC_T ouchProbe			
MC_DigitalCamSwitch			
MC_Reset			
MC_AbortTrigger			
MC_HaltSuperimposed			

表1

1.1.4. Compliance 和可移植性

这个工作的目的就是实现按单轴工作的运动控制功能块级的可移植,并且为用户提供这份文

档描述的相同的功能，尊重用户的接口，输入/输出变量，参数和使用单位。
在一个应用程序中将不同供应商的MC 库相结合的可能性留给系统集成商和最终用户解决。
一个声称符合PLCopen 规范的实现应该提供一套（即一个或多个）至少有基本输入和输出变量的运动控制功能块，在本文档的功能块定义中的定义表中以“B”标记。
对于更高级别的系统和未来的扩展，扩展的输入和输出变量的任意子集，在表中以“E”标记的可以被实现。
供应商特定的增加都标记为“V”
为了获取更多符合和使用PLCopen 运动控制标志的信息，请参阅附录B：

-Basic 输入/输出变量是强制的	在表中以字母“B”表示
-Extended 输入/输出变量是可选的	在表中以字母“E”表示
-Vendor Specific additions	在供应商的遵循文档以“V”表示

允许任何厂商增加厂商特定的参数到文档指定的功能块中任意一个中。
提示：根据IEC 61131-3规则，输入变量可能unconnected or not parameterized by the user，在这种情况下，功能块会使用功能块实例先前调用的值或避免使用第一次调用的初始值。
每个功能块的输入都有一个明确的初始值，通常是0.

如果不遵从标准，列在功能块和参数（例如：velocity, Accleration, distance等等）中的REAL 类型数据在不被看见的情况下被转换成SINT，INT，DINT 或者LREAL 类型，直到它和整套功能块和参数一致。
实现允许保存和基本数据类型一样长的扩展数据类型。例如：WORD 会被改为DWORD ，而不是REAL 。
在本规范新的版本中不再指定的FB 和输入可以存放在供应商的系统以保持兼容性，避免现有的FB 的不兼容的改变。

1.1.5. 名称的长度和缩短它们的方式

有些系统只支持名字中有有限数量的显著特征。 因为这些规则，下面提供了更短的名称。尽管必须在认证文档中被提及，这些名称仍然被视为标准，

缩短名称的规则表

Command	Cmd
Position	Pos
Velocity	Vel
Acceleration	Acc
Deceleration	Decel
Absolute	Abs
Relative	Rel
Actual	Act
Superimposed	SupImp
Additive	Add
Parameter	Par
Continuous	Cont
GearRatioDenominatorM1	RatioDenM1
GearRatioNumeratorM1	RatioNumM1

生成标准名称例子:

CommandAborted	CmdAborted
MC_MoveContinuousRelative	MC_MoveContRel
MC_ReadParameter	MC_ReadPar

1.1.6. 历史

第一份官方发布的Part 1产生于2001 年11 月。此后从用户和制定者收到了反馈。在2004 年决定发布一份新版本，Part 1版本1.1，其中包括规范纳入反馈后造成变化。这份更新在2005 年4月发行。在2005 年9月，第一份官方发布的Part 2-Extension发行。
此后，更正和附录被保存为两个部分。在2008 年这个提议被接受以合并part1和part 2成为一个新的part 1，发布成为版本2.0，就是你现在看到的文档。
基本上这两套功能块已经合并。此外，一些整体变化已经完成。这些变化包括(但不仅限于)；

- 简化的状态图表示和移动过渡指令AO 。
- 新的输入“ContinuousUpdate ”扩展了相关运动有关的功能块的behaviour
- Adopted description resulting in a changed behaviour of the output ‘Active’
- Aborting mode deleted in some FBs
- Changes in the mcAborting enum
- 将MC_Phasing 和MC_MoveContinuous FB 分割为相对和绝对两个版本。
- 新的FB MC_ReadMotionState, MC_ReadAxisInfo, MC_CombineAxes 和 MC_HaltSuperimposed
- 对凸轮的描述
- MC_CamTableSelect 采用输入“ExecutionMode ”扩展了功能。描述为“Periodic”定义更精确。
- MC_CamIn 采用输入“MasterStartDistance”和“MasterSyncPosition”扩展了功能。
- 新的输入“MasterValueSource ”和 MC_CamIn ， MC_GearIn ，MC_GearInPos ， MC_ReadMotionState 和 MC_CombineAxes 中匹配的数据类型
- MC_SetPosotion 的输入“模式”现在叫做“相对”(在第四章)
- 所有PLCopen 运动控制规范的功能块、枚举元素、数据类型、结构、输入和输出都采用统一的命名
- 达到相应的设置值后，“InVelocity”，“InGear”，“ InTorque” 和“InSync”的输出会发生改变。
- FB MC_ReadAxisInfo, MC_PhasingRelative 和MC_PhasingAbsolute 被加到了功能块中，但没有列在状态图中。
- 改变了输入“Axis”，“Master”和“Slave”的描述
- 改变了输出“Busy ”，“Error”和“ErrorID”的描述。

2. 模型

以下功能块库是为了控制轴而设计的，采用了IEC61131-3 标准定义的语言元素。根据专家组的决定，它并不是将一个轴的所有方面都封装到一个功能块中。保留的解决方案是提供一组有一个参考轴的命令导向功能块，例如：提供能灵活性，易于使用和可复用性的抽象数据类型“Axis”。
基于IEC61131-3 （比如通过功能块和SFC ）的实现主要集中于功能块的接口（外观和感觉/代理）。本规范没有定义功能块的内部运作。
这导致本章所描述的一些后果。

2.1. 状态关系图

下面的图表规范的在一个较高的水平定义了当多运动控制功能块“同时”激活时轴的行为。这种运动框架的组合有助于组建更为复杂的框架或在同一个程序中处理异常状况（In real implementations there may be additional states at a lower level)defined
基本规则是，运动控制总是采取顺序控制，即使PLC 有能力进行真正的并行处理。这些命令作用于轴的状态图。
轴始终处在定义的状态之一（见下图）。任何会引起转变的动作指令都会改变轴的状态，并且会因此修改当前运动的计算方式。
状态图是轴的实际状态的一个抽象层，类似于在一个循环程序（PLC ）中的I/O点的图像。当发出相应的动作指令时，立即反应状态的变化。(注：响应时间“立即”是依赖与系统的，和轴的状态耦合或者是软件中的一个抽象层。)
这个图表专注于单轴控制。多轴功能块，MC_CamIn ，MC_GearIn 和MC_Phasing ，从状态关系图的观点来看，可以看成多个单轴都处于特殊的状态。例如：CAM_masterke 可以处于“ContinuousMotion”状态，相应的从动装置处于“SynchronizedMotion”状态。将从动装置的轴连接到主轴上对主轴没有影响。
在状态图中的箭头显示了在状态之间可能的状态转换。由发出命令引起的状态转换以全箭头显示。虚线箭头用于发生在轴的命令终止或系统相关的转变（如有关错误）时的状态装换。将轴转换为相应运动状态的动作指令被列在状态之上。These motion commands may also be issued when the axis is already in the according motion state
Remarks on states

Disabled	状态“Disabled”描述了轴的最初状态。在这个状态轴的运动是不受FB 影响的，电源关闭，轴也没有错误。当处于“Disabled”时，如果MC_Power FB 的输入“Enable”=TRUE ，那轴的状态就变为“停止”。在进入“停止”状态之前可以进行轴的反馈。在除了“ErrorStop”以外的任一状态下以“Enabled”

	=FALSE 来启用MC_Power 可以使轴变为“Disabled”状态，无论是直接或是通过其它状态。轴上任何正在运行的动作指令都会中止(CommandAborted)。
Errorstop	“ErrorStop”作为最高优先级，适用于错误的情况下。轴可以电源启用或禁用，并且可以通过MC_Power 改变状态。然而，只要错误一直存在，状态就保持为“ErrorStop”。“ErrorStop”状态的意图是如果可能的话就让轴停止。在“ErrorStop”状态一个重置完成之前，不接受进一步的动作指令。转变到“ErrorStop”指的是从轴和轴控制的错误，而不是从功能块实例。这些轴的错误也可能会反映在功能块的输出“FB 实例错误”。
Standstill	接通电源，轴没有错误，并且没有动作指令作用在轴上

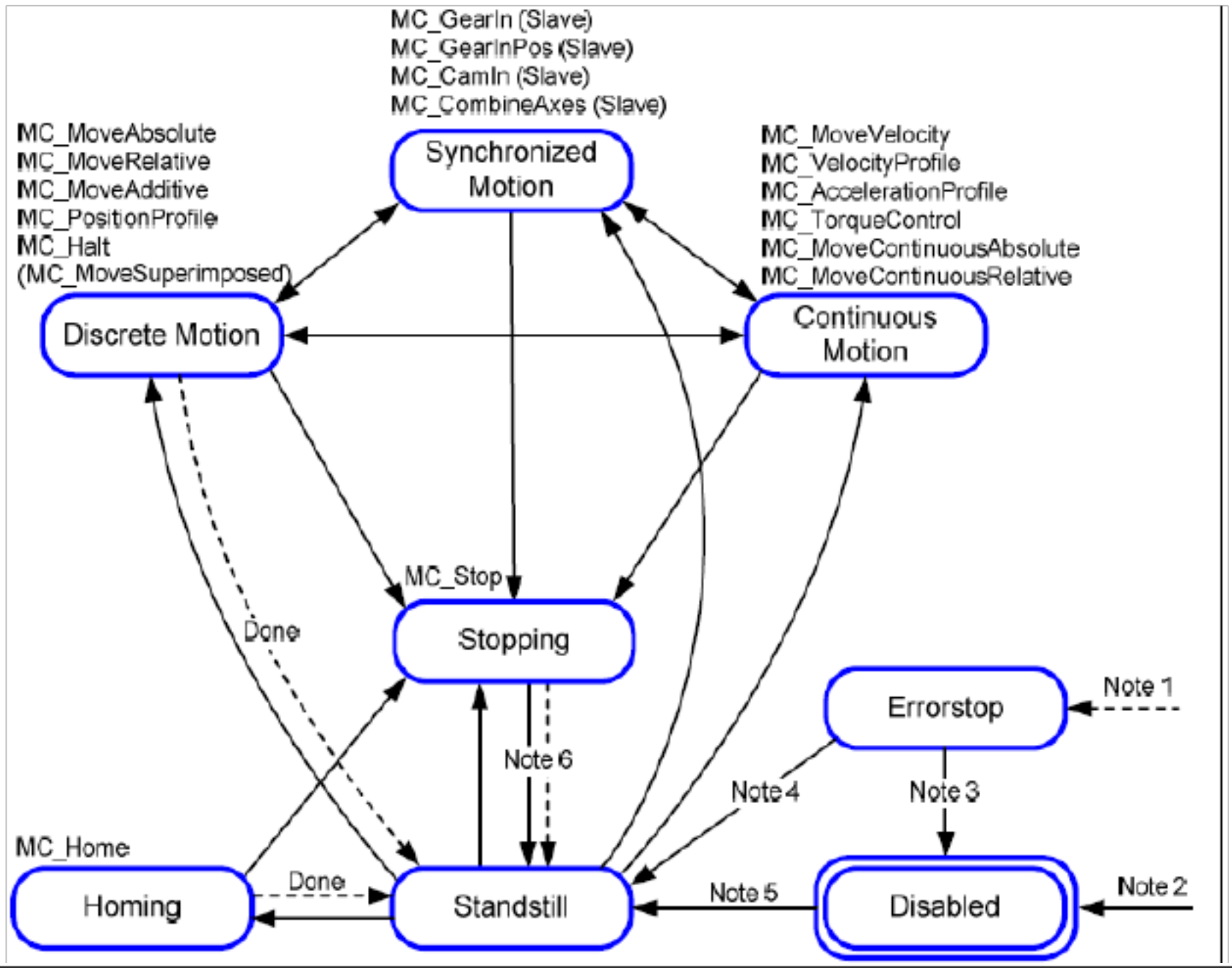
Remarks on command :

MC_Stop	在“Standstill”状态调用FB MC_Stop 将轴的状态改变为“Stopping”，当“Execute”=FALSE 时回到“Standstill”状态。只要输入“Execute”为true，轴就保持“Stopping”状态。当停止斜坡完成时“Done ”的输出被设置。
MC_MoveSuperimposed	MC_MoveSuperimposed 在“Standstill”状态issued会是州的状态变为“Discrete-Motion”(离散运动)。在其它任何状态issued轴的状态都不会受影响。
MC_GearOut, MC_CamOut	更改从动轴的状态“SynchronizedMotion”为“ContinuousMotion” 在其它任何状态issuing这些FB 中的一个会产生错误。

没有列在状态图中功能块不会影响状态图的状态，这意味着当它们被调用时，轴的状态并不会改变。它们是:

- MC_ReadStatus
- MC_ReadAxisError
- MC_ReadParameter
- MC_ReadBoolParameter
- MC_WriteParameter
- MC_WriteBoolParameter
- MC_ReadDigitalInput
- MC_ReadDigitalOutput
- MC_WriteDigitalOutput
- MC_ReadActualPosition
- MC_ReadActualVelocity
- MC_ReadActualTorque
- MC_ReadMotionState
- MC_SetPosition
- MC_SetOverride
- MC_AbortTrigger
- MC_T ouchProbe
- MC_DigitalCamSwitch
- MC_CamTableSelect
- MC_ReadAxisInfo

- MC_PhasingRelative
- MC_PhasingAbsolute
- MC_HaltSuperimposed



Note 1: 来自任何状态。轴的错误引起。
Note 2: 来自任何状态。MC_Power.Enable=FALSE 并且轴没有错误。
Note 3: MC_Reset 和MC_Power.Status=FALSE 。
Note 4: MC_Reset 和MC_Power.Status = TRUE 并且MC_Power.Enable=TRUE 。
Note 5: MC_Power.Enable = TRUE 并且 MC_Power.Status = TRUE
Note 6: MC_Stop.Done = TRUE 并且 MC_Stop.Execute = FALSE

FB 状态图

2.2. 错误处理

所有的驱动器/运动控制的访问都是通过功能块（从这一点可以看出对不同厂商的驱动器/运动控制，用户都可以使用具有相同接口的功能块去控制）。在这些功能块的内部提供了基本输入数据的基本错误的检查。究竟如何做到这一点是依赖于实现。例如：假如MaxVelocity设置为6000，而输入到FB 的Velocity设置为10000，系统就会变慢或者产生一个错误。In the case
where an intelligent drive is coupled via a network to the system, the MaxVelocity parameter is probably stored on the drive. FB要小心处理驱动内部产生的错误。采用另一种实现，MaxVelocity的值可以存储在本地。在这种情况下，FB 会产生本地错误。

2.2.1. 集中式VS 分散式

当使用运动控制功能块时，集中式和分散式的错误处理方法都是可行的。

集中式错误处理用于简化功能块的程序。在错误产生的实例中，Error-reaction一样是独立的。（集中式的错误处理，只在结尾读取轴的状态）

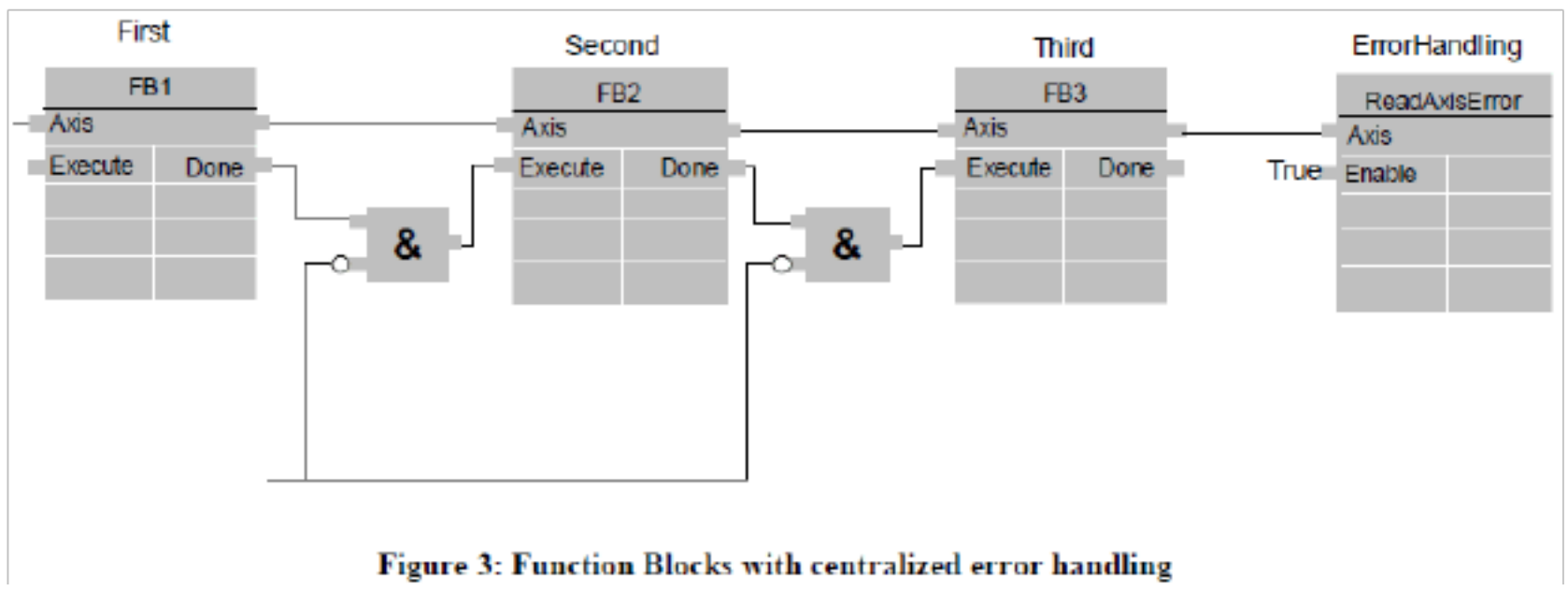


Figure 3: Function Blocks with centralized error handling

分散式的错误处理使基于错误产生的功能块的不同反应成为可能。

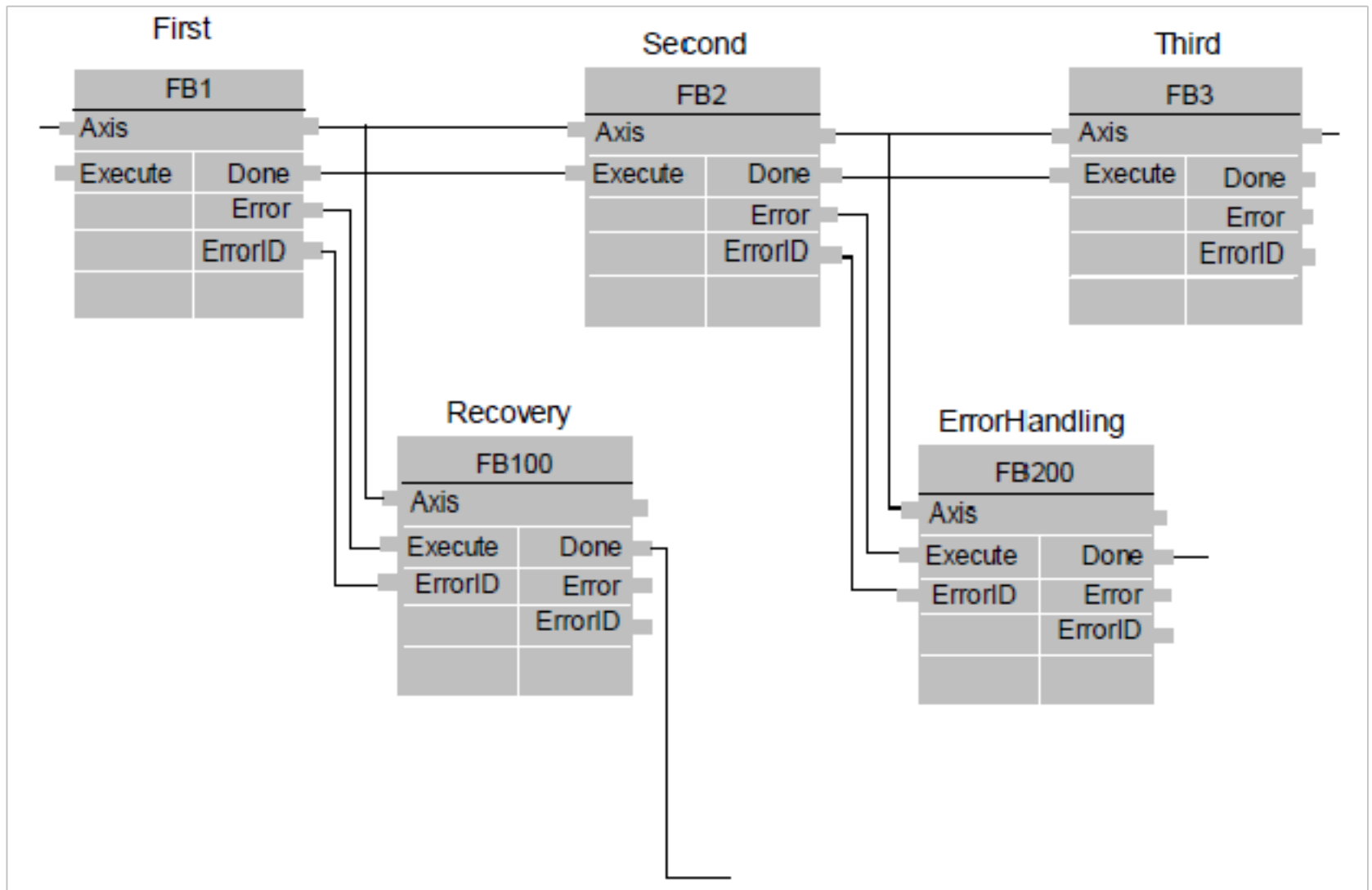


Figure 4: Function blocks with decentralized error handling

2.2.2 缓冲命令

如果适用的轴移到“ErrorStop”状态，所有的缓冲命令都会被中止。适用中止的FB 的“Error”输出是固定的。任何后续的命令都会被拒绝，并且“Error“输出是固定的（动作不允许-查看状态图）。

如果一个FB 有一个错误（例如由于一个错误的参数集），erro输出也设定好了，行为取决于应用程序。例如：有两个FB，第一个FB 实例FB1 在轴上执行任意动作指令。在缓冲模式下于同一个轴上在第二个FB 实例FB2 启动一个新的命令。这个命令处于缓冲状态并且等到FB 完成。在第一个实例FB1 完成它的命令时，会让下面的情形之一发生：

1. 轴变为“ErrorStop”状态（例如：由于下面的错误或超温）。FB1 设置其“Error”输出。FB2 （和其它任何等在轴上待执行缓冲命令的FB 实例一样）设置它的“Error”输出并且显示“ErrorID”的输出，它不能执行自己的工作，应为轴处在一个不允许其工作的状态。所有的缓冲命令都被清除。在轴的错误被MC_Reset 复位以后，它就可以再次被控制。（这种情况，FB2 不能执行）
2. FB1 设置其“Error”输出（例如：由于一个无效的参数）。FB2 变为active并且随后立即执行给定的命令，应用程序会处理这个错误的情况。

2.2.3.“Enable”输入的时序例子

例1：图片的左边显示正常运行。在图片的右边显示在运行的过程中发生了一个错误。这个错误迫使“Valid”输出复位。“Busy”的输出保持为高电平。在错误复位后，可能会在一段时间后恢复正常运作程序。（虚线应该是表示延时）

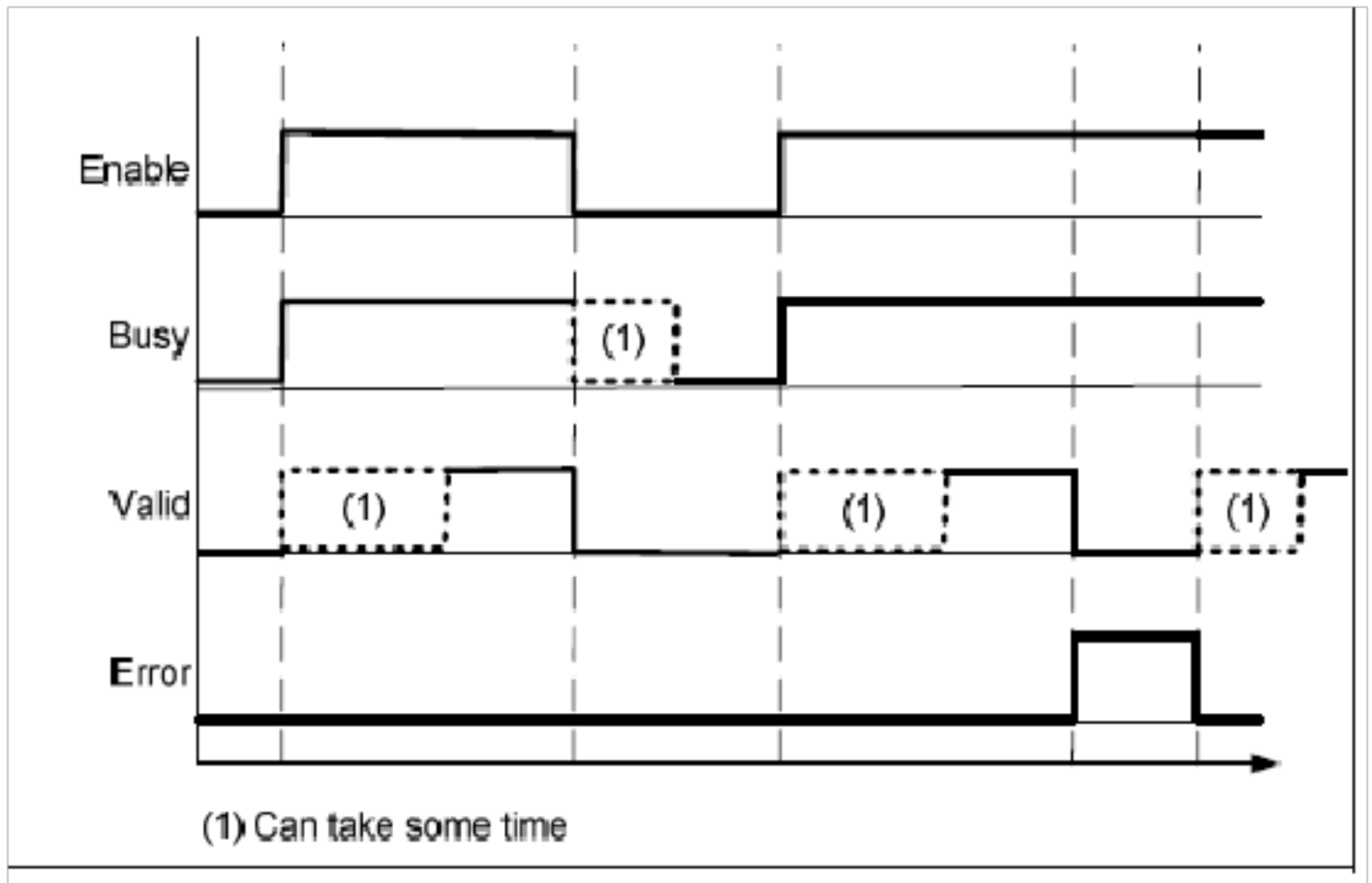


Figure 5: Example of error handling with ‘Enable’ input

第二个例子的右边显示了错误不能自动被清除。“Busy ”和“Valid”的输出在错误产生后被重置。FB 的“Enable”输入需要一个上升沿使其继续工作。

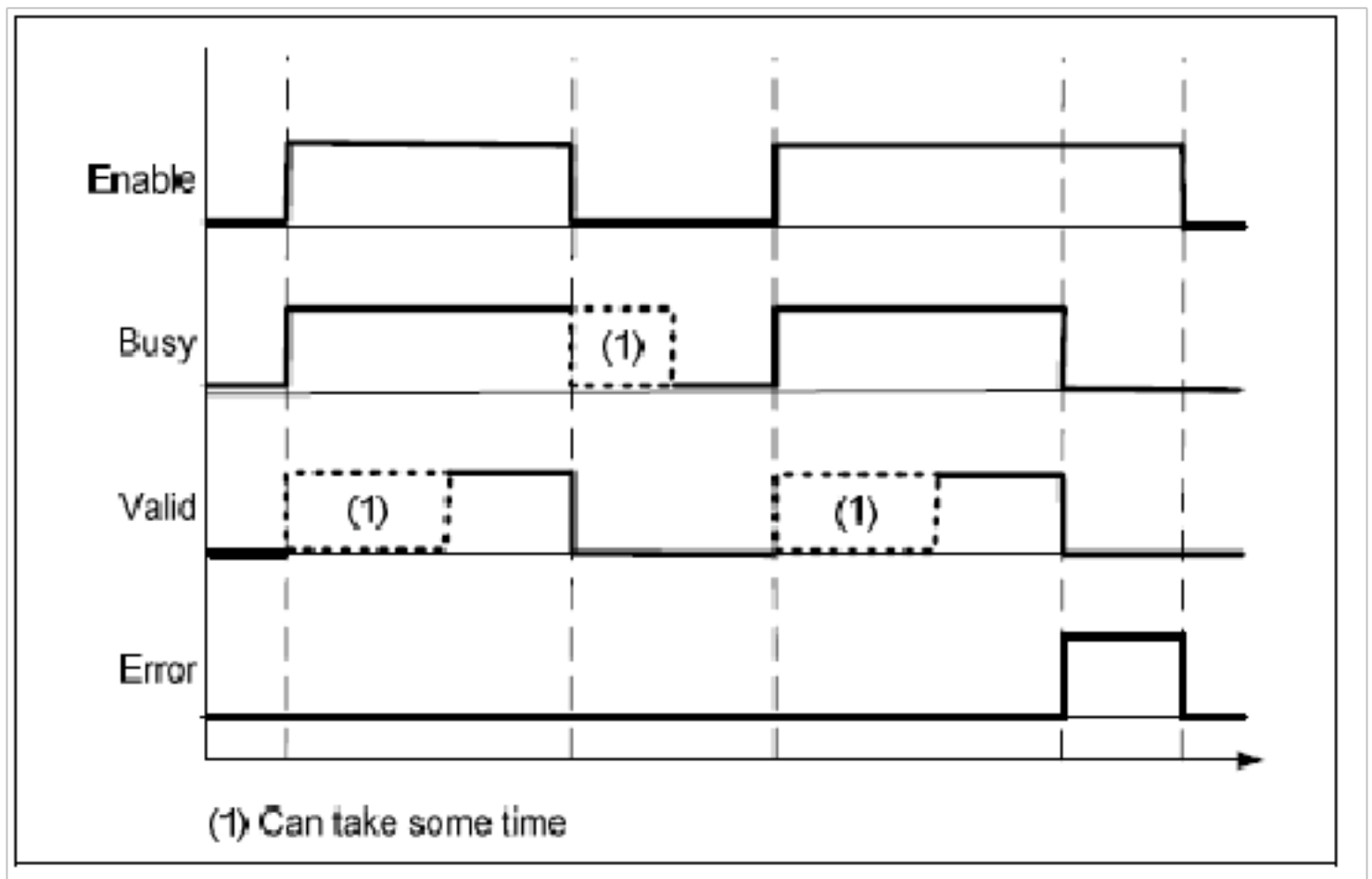


Figure 6: Second example of an error behavior with an ‘Enable’ input

2.3. 定义

本文中下面水平的值会被使用：Command/Set/Actual ：

- Command　值—是基于功能块输入的值，并且可以作为（其中一个）波形发生器的输入值。
- Set值—在一个“lower”水平，接近执行机构。它是最新的值，即将被发送到伺服回路（例如：执行机构），例如：执行机构即将使用的值。
- Actual值—从反馈系统中得到的在系统中可用的最新值。

2.4. FB 接口

2.4.1.一般规则

输入参数（Input parameters）	With“Execute”without“ContinuousUpdate”:输入参数和“Execute”的输入的上升沿一起使用。修改任何参数都有必要改变输入参数并且再次触发“Execute”输入。 With “Execute”combined with “ContinuousUpdate”: 此参数和“Execute”输入的上升沿一起使用。只要“ContinuousUpdate”是SET ，参数就可以被持续修改。 With“Enable”: 该参数和“Enable”输入的上升沿一起使用，并且可以持续被修改。
输入超过应用程序的限制	如果控制FB 的参数违反了应用程序的限制，输入会受限与系统或FB 实例产生一个错误。轴的错误引起的后果由特定的应用程序产生，因此应该由这个应用程序处理。
丢失输入参数	根据IEC 61131-3，如果丢失（开放）了一个功能块的任意输入参数，那么上次调用这个实例的值将被使用。在第一次调用时会使用初始值。
加速，减速和加速度变化率（Jerk）的输入	如果输入“Deceleration”,“Acceleration”或“Jerk”设置为0，结果依赖于实现。有几种可能的实现，像一个进入错误状态，一个信号警告（通过供应商特定的输出），一个将其抑制在编辑器，一个将这个值作为AxisRef或驱动器本身特定值，或者akes a maximum value 。即使输入值0被系统接受，请谨慎使用，尤其是针对兼容性。
输出的排它性	With“Execute”:输出“Busy”,“Done”,“Error”和“CommandAborted ”是相互排斥的：在FB 中它们只有一个可以为TRUE 。如果“Execute”为TRUE ，则它们其中一个输出必为TRUE 。 只有“Active”,“Error”,“Done ”和“CommandAborted ”其中的一个是在同一时间设置，除了MC_Stop 中“Valid”和“Error”可以在同一时间设置。 With“Enable”: 输出“Valid”和“Error”是相互排斥的：在FB 中它们中只有一个可以为TRUE.
输出状态	With“Execute”: 输出“Done ”,“Error”,“ErrorID”和“CommandAborted ”在“Execute”的下降沿复位。然而,“Execute”的下降沿并不会停止或者会影响实际FB 的执行。如果这种情况发生，它必须保证相应的输出设置位置少一个周期，即使在FB 完成之前“Execute”已被重置。 如果FB 的实例在完成前收到一个新的执行命令（在相同的实例作为一系列命令),FB 不会返回任何反馈,像“Done ”或“CommandAborted ”,为先前的动作。

	With“Enable”:“Valid”,“Enabled”,“Error”和“ErrorID”的输出会在“Enable”的下降沿被尽快的重置。
Done 输出的行为	当命令动作成功执行后“Done ”的输出被设置。 在顺序控制中，工作于相同轴上的多个功能块适用以下原则： 当轴上的一个运动在达成目标之前被另一个在相同轴上的运动打断，第一个FB 的“Done ”不会被设置。
Busy 输出的行为	With“Execute”：每个FB 都可以有“Busy ”输出，反映FB 还没有完成，并且可预期得到新的输出值。“Busy ”在“Execute”的上升沿被设置，并且当“Done ”,“Aborted”或者“Error”其中一个被设置时就被重置。 With“Enable”：每个FB 都可以有“Busy ”输出。反应FB 正在工作，并且可以预期得到新的输出值。“Busy ”在“Enable”的上升沿被设置，并且保持设置直到FB 执行任何动作。 根据建议，只要“Busy ”为true，FB 就应该保持在应用程序的active loop，因为输出仍然可能会改变
InVelocity, InGear, InTorque 和InSync的行为	“InVelocity”，“InGear”,“InTorque”和“InSync”（现在起称为“Inxxx”）的输出和“Done ”输出相比有不同的行为。 只要FB 是active，“Inxxx”is SET当设置值等于命令值，并且在稍后它们不相等的时候会被重置。例如：InVelocity的输出会在set velocity等于command velocity时SET 。这类似于在applicable FB中的“InGear”,“InTorque”和“InSync”的输出。 只要FB 控制轴（“Active”和“Busy ”are SET），即使“Execute”为低电平，“Inxxx”is updated。 当满足“Inxxx”的条件，在“Execute”重新设置后，“Inxxx”的行为是具体实施。 “Inxxx”的定义不是指实际轴的值，但是必须指内部瞬时设定值。
Active输出	“Active”输出需要缓冲功能块。其输出在功能块取得相应轴的运动控制权时被设置。对于un-buffered模式，“Active”和“Busy ”的输出可以拥有相同的值。 对于一个轴，可能有几个功能块处于busy状态，但是同时只能有一个处于active。例外是FB 打算并行工作，像MC_MoveSuperimposed 和MC_Phasing ，其中涉及到一个轴有超过一个FB 可以active。
CommandAborted 的输出行为	“CommandAborted ”在一个控制运动被另一个控制运动打断时被设置。 “CommandAborted ”的复位行为和“Done ”差不多。当 “CommandAborted ”发生时，其他输出信号如“InVelocity”被重置
Enable和Valid	“Enable”输入和一个“Valid”输出耦合。“Enable”是level敏感的，而“Valid”显示输出的有效设置在FB 中是可用的。 只要“Valid”:的输出值有效，其输出就是TRUE ，并且“Enable”输入为TRUE 。只要“Enable”输入为TRUE ，相关的输出值就可以被刷新。假如有一个FB 错误，输出就无效（“Valid”设置为FALSE ）。当错误条件消失时，值会再次出现并且“Valid”输出会被再次设置。
Position versus	“Position”是在一个坐标系统中定义的值。“Distance”是一个相对

distance	指标的技术单位。“Distance”是两个位置之间的差异。
Sign rules	“Acceleration”,“Deceleration”和“Jerk”通常都是正值。“Velocity”,“Position”和“Distance”既可以是正值也可以是负值
错误处理行为	所有的块都可以有两个输出，处理在执行功能块时可能产生的错误。这些输出的定义如下：
	Error：“Error”的上升沿通知在功能块执行的过程中产生了错误。 ErrorID：错误识别（扩展参数）
	“Done ”,“InVelocity”,“InGear”,“InTorque”和“InSync”意味着成功的完成，所以这些信号对于“Error”在逻辑上是独立的。 错误类型： ● 功能块（例如：参数超出范围，state machine violation attempted） ● 通信 ● 驱动器 实例错误并不总是导致轴错误（使轴进入“ErrorStop”）。相关FB的“Error”输出会在“Execute”和“Enable”的下降沿复位。With“Enable”FB中的“Error”输出会在操作过程中复位（不用复位“Enable”）。
FB命名	在一个系统（支持多个驱动器/运动控制系统）中有多个库的情况下，FB的命名可能会改成“MC_FBname_SupplierID ”。
命名约定的枚举类型	由于IEC标准对变量名唯一性的命名约束，将“MC ”参照PLCopen 运动控制命名空间的方法用于枚举。 In this way we avoid the conflict that using the ENUM types ‘positive’ and ‘negative’ for instance with variables with these names throughout the rest of the project since they are called mcPositive and mcNegative resp.

“Execute”/“Done ”类型的FB的行为如下：

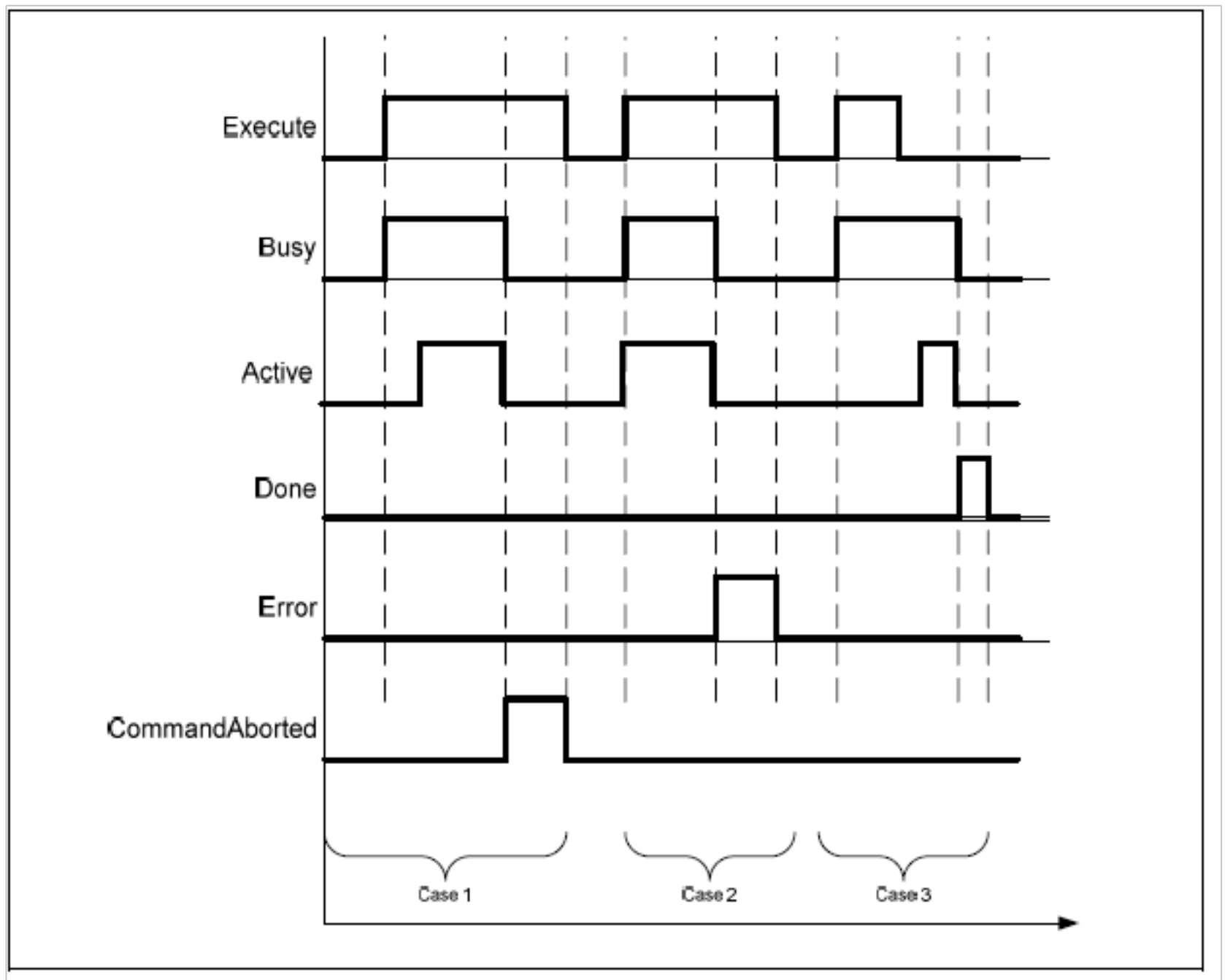


Figure 7: The behavior of the 'Execute' / 'Done' in relevant FBs

“Execute”/“Inxxx”类型的FB 的行为如下：