参考手册

Allen-Bradley

Logix 5000 高级过程控制和驱动指令

1756 ControlLogix, 1756 GuardLogix, 1769 CompactLogix, 1769 Compact GuardLogix, 1789 SoftLogix, 5069 CompactLogix, Emulate 5570

Allen-Bradley • Rockwell Software

Rockwell Automation

重要用户信息

在安装、配置、操作或维护本产品前，请阅读本文档及其他资源部分列出的有关本设备安装、配置和操作的文档。除所有适用规范、法律和标准的要求外，用户还需要熟悉有关安装和接线的事项。

安装、调整、维修、使用、组装、拆卸和维护等活动需要由经相应培训的人员遵照适用操作规范执行。不遵照制造商指定的方式使用本设备，可能设备的保护措施失效。

在任何情况下，Rockwell Automation, Inc. 均不对因使用或应用本设备而造成的间接性或继发性损害承担责任。

本手册中的示例和图表仅供说明之用。由于具体安装场合的实际状况和要求千差万别，Rockwell Automation, Inc. 无法对基于这些示例和图表的实际使用承担责任或义务。

Rockwell Automation, Inc. 亦不对本手册所述之信息、电路、设备或软件的使用承担任何专利责任。

未经 Rockwell Automation, Inc. 的书面许可，禁止复制本手册的全部或部分内容。

我们根据需要在本手册中使用了注释，提醒您注意一些安全事项。

	**警告：**提示可能在危险环境中引发爆炸的操作或状况，这些操作或状况可能导致人身伤亡、财产损坏或经济损失。
	**注意：**提示可能导致人身伤亡、财产损坏或经济损失的操作或状况。这些提示可帮助您识别危险、避免危险、了解后果
重要事项：	提示一些对成功应用和理解产品至关重要的信息。

设备机身或内部可能贴有此类标签，用以提示特定的预防措施。

	**电击危险：**设备机身或内部（例如驱动器或电机）可能贴有此类标签，用以提醒人们可能存在危险电压。
	**灼伤危险：**设备机身或内部（例如驱动器或电机）可能贴有此类标签，用以提醒人们表面可能达到危险温度。
	**弧闪危险：**设备机身或内部（例如电机控制中心）可能贴有此类标签，用以提醒人们注意潜在弧闪。弧闪会导致严重人身伤亡。应穿戴适当的个人防护设备 (PPE)。遵循有关安全工作实践和个人防护设备 (PPE) 的所有监管要求。

本手册包含一些新增和更新的信息。使用这些参考表可找到发生变更的信息。

全局变更

本版本无全局变更。

新增或增强功能

主题	原因
通用属性 参考页数 589	增加了指向"基本数据类型"主题的链接
立即数 参考页数 591	增加了"整型立即数"和"浮点型立即数"两张表。
数据转换 参考页数 592	将最佳数据类型更改为立即数数据类型，并加入了 USINT、INT、UINT、UDINT、ULINT 和 LREAL 扩展数据类型。在"将 SINT 或 INT 型转换为 DINT 型"部分，增加了将 DINT 型转换为 LINT 型的内容。加入了 32 位和 64 位数据转换的内容。
基本数据类型 参考页数 596	将主题标题由"数据类型"更改为"基本数据类型"。增加了 LINT、USINT、UINT、UDINT、ULINT、REAL 和 LREAL 数据类型。
LINT 数据类型 参考页数 599	增加了适用控制器列表，列出支持指令中所用 LINT 数据类型的控制器。
浮点值 参考页数 600	增加了适用控制器列表。增加了 LREAL 标签说明。
数组索引编制 参考页数 602	新增两条提示信息，一个用于说明 Logix Designer 仅支持以扩展数据类型标签作为下标；另一个用于说明所有可用基本整型数据类型均可用作下标索引。
位寻址 参考页数 603	增加了新定义。
FOR DO 参考页数 576	更新了循环结束的说明。

此定位器用于在适用的 Logix5000 控制器指令手册中查查各个指令。

Logix5000 控制器通用指令参考手册（出版号 1756-RM003）	**Logix5000 控制器高级过程控制和驱动指令参考手册（出版号 1756-RM006）**	**Logix5000 Controllers Motion Instructions 参考手册（出版号 MOTION-RM002）**
绝对值 (ABS)	报警 (ALM)	主轴驱动协调控制 (MDCC)
加 (ADD)	连接到设备阶段 (PATT)	运动应用轴调谐 (MAAT)
模拟报警 (ALMA)	连接到设备顺序 (SATT)	运动应用连接诊断 (MAHD)
恒假 (AFI)	协调控制 (CC)	运动装备输出凸轮 (MAOC)
反余弦（ACS、ACOS）	D 触发器 (DFF)	运动装备记录 (MAR)
反正弦（ASN、ASIN）	死区时间 (DEDT)	运动装备监视 (MAW)
反正切（ATN、ATAN）	微分 (DERV)	运动轴故障复位 (MAFR)
缓冲区中的 ASCII 字符 (ACB)	从设备阶段断开 (PDET)	运动轴传动 (MAG)
ASCII 清空缓冲区 (ACL)	从设备顺序断开 (SDET)	运动轴归零 (MAH)
ASCII 握手线 (AHL)	离散三态设备 (D3SD)	运动轴点动 (MAJ)
ASCII 读取 (ARD)	离散 2 态设备 (D2SD)	运动轴运动 (MAM)
ASCII 读取线 (ARL)	增强型 PID (PIDE)	运动轴位置凸轮 (MAPC)
缓冲区行的 ASCII 测试 (ABL)	增强型选择 (ESEL)	运动轴停止 (MAS)
ASCII 写入 (AWT)	设备阶段清除故障 (PCLF)	运动轴时间凸轮 (MATC)
ASCII 写入附加 (AWA)	设备阶段命令 (PCMD)	运动轴关闭 (MASD)
位域分配 (BTD)	设备阶段外部请求 (PXRQ)	运动轴关闭复位 (MASR)
带目标的位域分配 (BTDT)	设备阶段故障 (PFL)	运动计算凸轮轮廓 (MCCP)
位左移 (BSL)	设备阶段新参数 (PRNP)	运动计算从轴值 (MCSV)
位右移 (BSR)	设备阶段超控命令 (POVR)	运动计算变换位置 (MCTP)
按位与 (AND)	设备阶段暂停 (PPD)	运动变化动态 (MCD)
按位非 (NOT)	设备顺序分配顺序标识符 (SASI)	运动协调变化动态 (MCCD)
按位或 (OR)	设备顺序清除故障 (SCLF)	运动协调圆弧运动 (MCCM)
布尔与 (BAND)	设备顺序命令 (SCMD)	运动协调直线运动 (MCLM)
布尔异或 (BXOR)	设备顺序超控 (SOVR)	运动协调关闭 (MCSD)
布尔非 (BNOT)	函数生成器 (FGEN)	运动协调关闭复位 (MCSR)
布尔或 (BOR)	高通滤波器 (HPF)	运动协调停止 (MCS)
中断 (BRK)	上限/下限 (HLL)	运动协调变换 (MCT)

Logix5000 控制器通用指令参考手册（出版号 1756-RM003）	Logix5000 控制器高级过程控制和驱动指令参考手册（出版号 1756-RM006）	Logix5000 Controllers Motion Instructions 参考手册（出版号 MOTION-RM002）
断点 (BPT)	积分器 (INTG)	运动直接驱动关闭 (MDF)
清零 (CLR)	内模控制 (IMC)	运动直接驱动开启 (MDO)
比较 (CMP)	JK 触发器 (JKFF)	运动直接启动 (MDS)
转换为 BCD (TOD)	超前-滞后 (LDLG)	运动解除输出凸轮 (MDOC)
转换为整数 (FRD)	低通滤波器 (LPF)	运动解除记录 (MDR)
复制文件 (COP)、同步复制文件 (CPS)	最大值捕捉 (MAXC)	运动解除监视 (MDW)
余弦 (COS)	最小值捕捉 (MINC)	运动组关闭 (MGSD)
计算 (CPT)	模块多变量控制 (MMC)	运动组关闭复位 (MGSR)
向下计数 (CTD)	移动平均值 (MAVE)	运动组停止 (MGS)
向上计数 (CTU)	移动标准偏差 (MSTD)	运动组抓拍位置 (MGSP)
向上/向下计数 CTUD	多路选择器 (MUX)	运动位置重设 (MRP)
数据转换 (DTR)	陷波滤波器 (NTCH)	运动运行轴调谐 (MRAT)
度数 (DEG)	阶段状态完成 (PSC)	运动运行连接诊断 (MRHD)
诊断检测 (DDT)	位置比例 (POSP)	运动伺服关闭 (MSF)
数字报警 (ALMD)	比例 + 积分 (PI)	运动伺服开启 (MSO)
DINT 转换为字符串 (DTOS)	脉冲乘法器 (PMUL)	
除 (DIV)	升温/持温 (RMPS)	
转换结束 (EOT)	变化率限制器 (RLIM)	
等于 (EQU)	优先复位 (RESD)	
文件算术和逻辑 (FAL)	标定 (SCL)	
位比较文件 (FBC)	S 曲线 (SCRV)	
FIFO 装载 (FFL)	二阶控制器 (SOC)	
FIFO 卸载 (FFU)	二阶超前滞后 (LDL2)	
文件平均值 (AVE)	选择 (SEL)	
文件标准偏差 (STD)	选择取反 (SNEG)	
文件填充 (FLL)	选择加法器 (SSUM)	
文件排序 (SRT)	优先置位 (SETD)	
查找字符串 (FIND)	分程时间比例 (SRTP)	
循环 (FOR)	累加器 (TOT)	

Logix5000 控制器通用指令参考手册（出版号 1756-RM003）	Logix5000 控制器高级过程控制和驱动指令参考手册（出版号 1756-RM006）	Logix5000 Controllers Motion Instructions 参考手册（出版号 MOTION-RM002）
文件搜索和比较 (FSC)	增/减累加器 (UPDN)	
获取系统值 (GSV) 和设置系统值 (SST)		
大于等于 (GEQ)		
大于 (GRT)		
插入字符串 (INSERT)		
即时输出 (IOT)		
跳转至标签 (JMP) 和标签 (LBL)		
跳转至子例程 (JSR)、子例程 (SBR) 和返回 (RET)		
跳转至外部例程 (JXR)		
小于 (LES)		
小于等于 (LEQ)		
LIFO 装载 (LFL)		
LIFO 卸载 (LFU)		
限制 (LIM)		
以 10 为底的对数 (LOG)		
小写 (LOWER)		
屏蔽移动 (MVM)		
带目标屏蔽移动 (MVMT)		
主控复位 (MCR)		
屏蔽码等于 (MEQ)		
消息 (MSG)		
中间字符串 (MID)		
取模 (MOD)		
移动 (MOV)		
乘 (MUL)		
自然对数 (LN)		
取反 (NEG)		
不等于 (NEQ)		
无操作 (NOP)		

Logix5000 控制器通用指令参考手册（出版号 1756-RM003）	Logix5000 控制器高级过程控制和驱动指令参考手册（出版号 1756-RM006）	Logix5000 Controllers Motion Instructions 参考手册（出版号 MOTION-RM002）
单脉冲触发 (ONS)		
下降沿单脉冲触发 (OSF)		
带输入的下降沿单脉冲触发 (OSFI)		
上升沿单脉冲触发 (OSR)		
带输入的上升沿单脉冲触发 (OSRI)		
输出接通 (OTE)		
输出闭锁 (OTL)		
输出解锁 (OTU)		
比例、积分和微分 (PID)		
弧度 (RAD)		
实数转换为字符串 (RTOS)		
复位 (RES)		
SFC 复位 (SFR)		
返回 (RET)		
保持型接通计时器 (RTO)		
带复位的保持型接通计时器 (RTOR)		
SFC 暂停 (SFP)		
以元素计的大小 (SIZE)		
定序程序输入 (SQI)		
定序程序加载 (SQL)		
定序程序输出 (SQO)		
正弦 (SIN)		
平方根 (SQR/SQRT)		
字符串串连 (CONCAT)		
字符串删除 (DELETE)		
字符串转换为 DINT (STOD)		
字符串转换为 REAL (STOR)		
交换字节 (SWPB)		
减 (SUB)		
正切 (TAN)		

Logix5000 控制器通用指令参考手册（出版号 1756-RM003）	Logix5000 控制器高级过程控制和驱动指令参考手册（出版号 1756-RM006）	Logix5000 Controllers Motion Instructions 参考手册（出版号 MOTION-RM002）
关断延时计时器 (TOF)		
带复位的关断延时计时器 (TOFR)		
接通延时计时器 (TON)		
带复位的接通延时计时器 (TONR)		
临时结束 (TND)		
追踪点 (TPT)		
触发事件任务 (EVENT)		
截断 (TRN)		
未知指令 (UNK)		
大写 (UPPER)		
禁止用户中断 (UID)/允许用户中断 (UIE)		
X 的 Y 次幂 (XPY)		
检查是否闭合 (XIC)		
检查是否断开 (XIO)		
按位异或 (XOR)		

目录

第 2 章

驱动

第 3 章

滤波器

第 4 章

选择_限制指令

第 5 章

统计指令

第 6 章

逻辑和移动

第 7 章

设备阶段指令

第 8 章

设备顺序

第 9 章

功能块属性

第 10 章

结构化文本编程

第 11 章

高级过程控制和驱动指令的通用属性

索引

本手册为编程人员详细介绍 Logix 型控制器可用的常规、运动控制、过程和驱动指令集。

若要对使用 GuardLogix 控制器的安全应用进行设计、编程或故障排除，请参见 GuardLogix Safety Application Instruction Set Safety 参考手册，出版号 1756-RM095。

本手册是介绍 LOGIX 5000 控制器通用编程与操作步骤的系列手册之一。

有关通用步骤手册的完整列表，请参见 LOGIX 5000 Controllers Common Procedures Programming Manual，出版号 1756-PM001。

文档中的 LOGIX 5000 控制器是指基于 LOGIX 5000 操作系统的所有控制器。

Studio 5000 环境

Studio 5000 Automation Engineering & Design Environment® 将工程和设计元素整合到一个公共环境中。第一个元素是 Studio 5000 Logix Designer® 应用程序。Logix Designer 应用程序由 RSLogix 5000® 软件更新换代而成，继续作为 LOGIX 5000™ 控制器的编程产品，用于编写离散、过程、批处理、运动、安全和基于驱动器的解决方案。

Studio 5000® 环境将成为 Rockwell Automation® 工程 设计工具和功能的基础。Studio 5000 环境将是设计工程师开发控制系统中所有元素的一体化环境。

其他资源

以下文档包含有关 Rockwell Automation 相关产品的其他信息。

资源	说明
Industrial Automation Wiring and Grounding Guidelines，出版号 1770-4.1	针对 Rockwell Automation 工业系统的安装，提供一些常规的指导。
产品认证网页 http://ab.rockwellautomation.com	提供符合性声明、证书及其他认证详情。

可以访问以下网址查看或下载上述出版物：http://www.rockwellautomation.com/literature。如需纸质版技术文档，请联系当地的 Rockwell Automation 分销商或销售代表。

本手册用途

本手册旨在为编程人员介绍有关 Logix 型控制器各可用指令的详细信息。本手册还将提供具体的用法指导和示例，帮助编程人员使用设备阶段指令来切换状态、处理故障、设置断点及完成其他操作。

法律声明

版权声明

最终用户许可协议（EULA）

您可以打开您的硬盘驱动器中产品安装文件夹内的 License.rtf 文件查看 Rockwell Automation 最终用户许可协议 ("EULA")。

开源许可证

本产品中包含的软件中，某些软件受版权保护但已获得一个或多个开源许可证授权许可。这些许可证的副本包含在软件内。本产品中开源软件包的相应源代码位于各自的网站上。

或者可以通过填写 Rockwell Automation 网站上的“联系”表格联系 Rockwell Automation，获取完整的相应源代码：http://www.rockwellautomation.com/global/about-us/contact/contact.page 请在请求文本中写明“开源”字样。

本产品中使用的所有开源软件及其相应许可证的完整列表位于产品版本说明的 OPENSOURCE 文件夹中。这些许可证的默认安装位置为 C:\Program Files (x86)\Common Files\Rockwell\Help\<*Product*>\ReleaseNotes\OPENSOURCE\index.htm。

商标声明

Allen-Bradley、ControlBus、ControlFLASH、Compact GuardLogix、Compact I/O、ControlLogix、CompactLogix、DCM、DH+、Data Highway Plus、DriveLogix、DPI、DriveTools、Explorer、FactoryTalk、FactoryTalk Administration Console、FactoryTalk Alarms and Events、FactoryTalk Batch、FactoryTalk Directory、FactoryTalk Security、FactoryTalk Services Platform、FactoryTalk View、FactoryTalk View SE、FLEX Ex、FlexLogix、FLEX I/O、Guard I/O、High Performance Drive、Integrated Architecture、Kinetix、Logix5000、LOGIX 5000、Logix5550、MicroLogix、DeviceNet、EtherNet/IP、PLC-2、PLC-3、PLC-5、PanelBuilder、PowerFlex、PhaseManager、POINT I/O、PowerFlex、Rockwell Automation、RSBizWare、Rockwell Software、RSEmulate、Historian、RSFieldbus、RSLinx、RSLogix、RSNetWorx for DeviceNet、RSNetWorx for EtherNet/IP、RSMACC、RSView、RSView32、Rockwell Software Studio 5000 Automation Engineering & Design Environment、Studio 5000 View Designer、SCANport、SLC、SoftLogix、SMC Flex、Studio 5000、Ultra 100、Ultra 200、VersaView、WINtelligent、XM、SequenceManager 是 Rockwell Automation, Inc. 的商标。

此处未提及的任何 Rockwell Automation 徽标、软件或硬件产品也是 Rockwell Automation, Inc. 的商标、注册商标或其他财产。

其他商标

CmFAS Assistant、CmDongle、CmStick、CodeMeter、CodeMeter Control Center 和 WIBU 是 WIBU-SYSTEMS AG 在美国和/或其他国家的注册商标。

所有其他商标均为其各自所有者的资产，特此声明。

担保

本产品根据产品许可担保。产品的性能可能受系统配置、执行的应用程序、操作员控制、维护和其他相关因素影响。对于这些干扰因素，Rockwell Automation 概不负责。本文档中的说明并未涵盖描述的设备、步骤或过程的所有详细信息或变体，也不提供满足安装、操作或维护期间每个可能意外情况的指引。本产品的实施可能在用户之间有所不同。

本文档是产品发布时的即时版本；然而，发布后附带的软件可能有所更改。Rockwell Automation, Inc. 保留随时更改本文档或软件中包含的任何信息的权利，恕不提前通知。在安装或使用本产品时，您负责从 Rockwell 获取可用的最新信息。

环境合规性

Rockwell Automation 在其网站 http://www.rockwellautomation.com/rockwellautomation/about-us/sustainability-ethics/product-environmental-compliance.page 上维护当前产品环境信息

联系 Rockwell

客户支持电话 — 1.440.646.3434

联机支持 — http://www.rockwellautomation.com/support/

过程控制指令

过程控制指令

过程控制指令包括以下指令：

可用指令

梯形图

不可用

功能块和结构化文本

ALM	SCL	PIDE	RMPS	POSP	SRTP	LDLG	FGEN

TOT	DEDT	D2SD	D3SD	IMC	CC	MMC

如果希望	使用此指令
针对任何模拟信号提供报警。	ALM
控制只有两种可能状态（如开/关、打开/关闭）的离散设备，如电磁阀、泵和电机等。	D2SD
控制有三种可能状态（例如快/慢/关，正向/停止/反向等）的离散设备，如快/慢/关进料系统。	D3SD
执行单路输入的延时。死区时间延时可以选择。	DEDT
基于分段线性函数对输入进行转换。	FGEN
为输入信号提供相位超前-滞后补偿。	LDLG
使用 PID 算法调节模拟量输出，使过程变量保持在特定的设置点。	PIDE
通过生成开关触点脉冲升/降或开/关设备（如电动阀）。	POSP
提供交替出现的升温和持温周期以遵循温度曲线。	RMPS

将未标定的输入值转换为采用工程单位的浮点值。	SCL
采用 PID 回路的 0-100% 输出，并通过周期脉冲驱动加热和冷却数字量输出触点。	SRTP
对一段时间内的模拟量输入值进行累加，例如体积流量等。	TOT
通过保持单个控制器输出来控制单个过程变量。	IMC
通过操作多达三个不同的控制变量控制单个过程变量。	CC
使用多达三个控制变量来控制两个过程变量，使其达到设置点。	MMC

另请参见

滤波指令 参考页数 365

逻辑和移动指令 参考页数 453

驱动指令 参考页数 309

选择/限制指令 参考页数 395

统计指令 参考页数 431

报警 (ALM)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

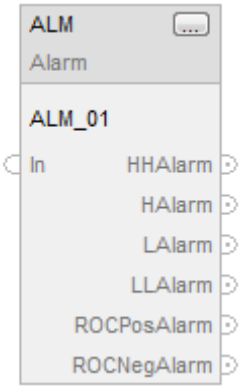
ALM 指令可针对任何模拟信号提供报警。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

ALM(ALM_tag)

操作数

功能块

操作数	类型	格式	说明
ALM tag	ALARM	结构	ALM 结构

结构化文本

操作数	类型	格式	说明
ALM tag	ALARM	结构	ALM 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

ALARM 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0

HHLimit	REAL	输入的上上限报警限值。 有效值 = 任意实数值 默认值 = 最大正值
HLimit	REAL	输入的上限报警限值。 有效值 = 任意实数值 默认值 = 最大正值
LLimit	REAL	输入的下限报警限值。 有效值 = 任意实数值 默认值 = 最大负值
LLLimit	REAL	输入的下下限报警限值。 有效值 = 任意实数值 默认值 = 最大负值
≤ Deadband	REAL	上上限到下下限的报警死区 有效值 = ≥ 0.0 的任意实数值 默认值 = 0.0
ROCPosLimit	REAL	输入正（增大）变化时的变化率报警限值（单位/秒）。若设置 ROCPosLimit = 0，可禁用正 ROC 报警。如果该值无效，该指令会假定该值等于 0.0，并将 Status 中的相应位置位。 有效值 = ≤ 0.0 的任意实数值 默认值 = 0.0
ROCNegLimit	REAL	输入负（减小）变化时的变化率报警限值（单位/秒）。若设置 ROCNegLimit = 0，可禁用负 ROC 报警。如果该值无效，该指令会假定该值等于 0.0，并将 Status 中的相应位置位。 有效值 = ≤ 0.0 的任意实数值 默认值 = 0.0
ROCPeriod	REAL	用于计算变化率值的时间周期（采样间隔），以秒为单位。每次采样时间间隔到期后，都将存储 In 中的新采样，并重新计算 ROC。变化率检测通过将 ROCPeriod 设为任意非零值的方式使能，而不像模拟报警中的其他条件那样使用使能位。 有效值 = 0.0 到 32767.0 默认值 = 0.0。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 ROC 溢出，则设置为假。

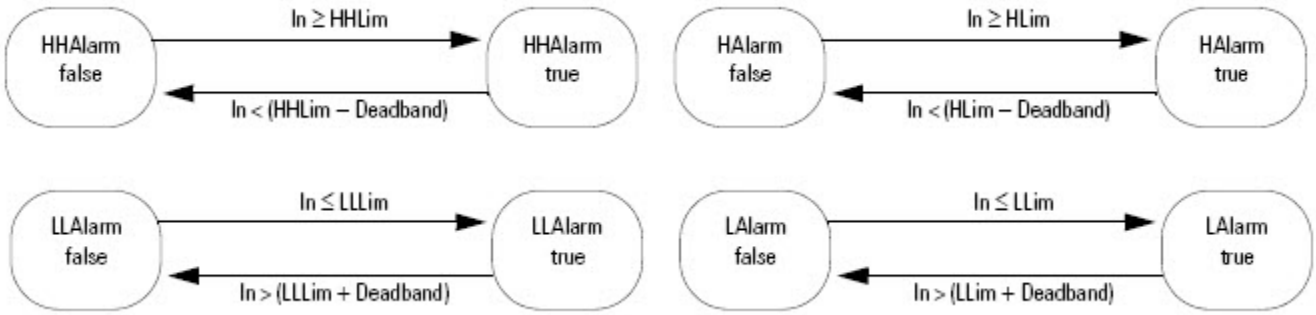
HHAlarm	BOOL	上上限报警指示器。 默认值 = 假
HAlarm	BOOL	上限报警指示器。 默认值 = 假
LAlarm	BOOL	下限报警指示器。 默认值 = 假
LLAlarm	BOOL	下下限报警指示器。 默认值 = 假
ROCPosAlarm	BOOL	正变化率报警指示器。 默认值 = 假
ROCNegAlarm	BOOL	负变化率报警指示器。 默认值 = 假
ROC	REAL	变化率输出。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
DeadbandInv (Status.1)	BOOL	Deadband 值无效。
ROCPosLimitInv (Status.2)	BOOL	ROCPosLimit 值无效。
ROCNegLimitInv (Status.3)	BOOL	ROCNegLimit 值无效。
ROCPeriodInv (Status.4)	BOOL	ROCPeriod 值无效。

说明

ALM 指令为上上限、上限、下限、下下限、正变化率、负变化率提供报警指示器。它可为上上限到下下限报警提供报警死区，还可提供用户定义的时间段用于执行变化率报警。

上上限到下下限报警

上上限和下下限报警算法会将输入与报警限值以及报警限值加减死区的结果进行比较。



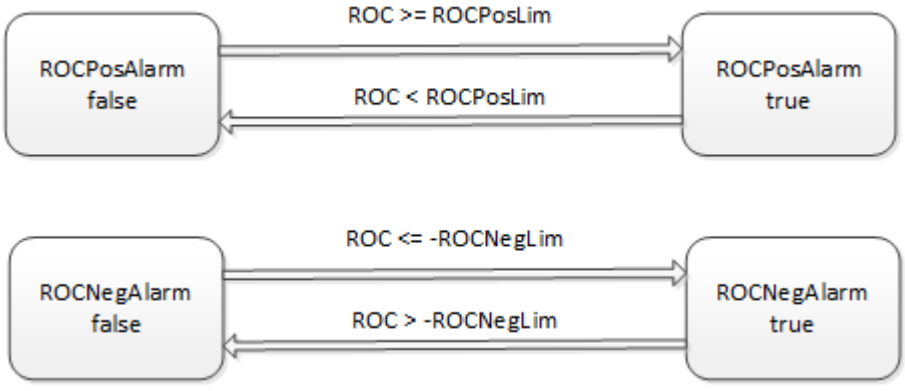
变化率报警

变化率 (ROC) 报警会将 ROCPeriod 内输入的变化与变化率限值进行比较。ROCPeriod 为变化率报警提供一种死区类型。例如，定义执行周期为 100 ms、大小为 2°F/s 的 ROC 报警限值。如果使用分辨率为 1°F 的模拟量输入模块，则输入值每次发生变化时，都会生成 ROC 报警，因为指令计算出的有效变化率为 10°F/s。然而，如果输入的 ROCPeriod 为 1 秒，则仅当变化率真正超过 2°F/s 的限值时，指令才会生成报警。

ROC 报警会按以下公式计算变化率：

$$ROC = \frac{In(Now) - In(EndofpreviousROCPeriod)}{ROCPeriod}$$

当 ROCPeriod 到期时，指令会执行此计算。指令计算出 ROC 后，按以下规则确定报警：



监视 ALM 指令

ALM 指令有相应的操作员面板。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	梯级输入条件位设置为假。
梯级输入条件为假	梯级输入条件位设置为假。
梯级输入条件为真	梯级输入条件设置为真。 指令执行。
后扫描	梯级输入条件位设置为假。

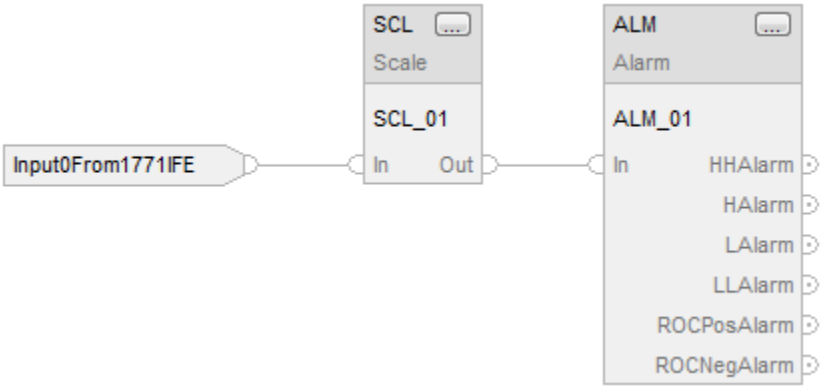
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“梯级输入条件为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

ALM 指令通常用于不支持板载报警的模拟量输入模块（例如 1771 I/O 模块），或用于生成与计算出的变量相关的报警。本例中，模拟量输入来自 1771-IFE 模块，首先使用 SCL 指令将其标定为工程单位。SCL 指令的 Out 输出是 ALM 指令的输入，用于确定是否设置报警。得出的报警输出参数随后可在程序中使用，并/或可在操作员界面显示画面上进行查看。

功能块



结构化文本

SCL_01.IN := Input0From1771IFE;

SCL(SCL_01);

ALM_01.IN := SCL_01.Out;

ALM(ALM_01);

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

功能块面板控件 参考页数 604

离散三态设备 (D3SD)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

D3SD 指令控制有三种可能状态（如快/慢/停或正向/停止/反向）的离散设备。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块

结构化文本

D3SD(D3SD_tag)

操作数

结构化文本

操作数	类型	格式	说明
D3SD tag	DISCRETE_3STATE	结构	D3SD 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

功能块

操作数	类型	格式	说明
D3SD tag	DISCRETE_3STATE	结构	D3SD 结构

DISCRETE_3STATE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
Prog0Command	BOOL	程序给定状态 0。此输入用于确定设备处于程序控制模式时的设备状态。如果为真，设备受控进入状态 0。 默认值为假。
Prog1Command	BOOL	程序给定状态 1。此输入用于确定设备处于程序控制模式时的设备状态。如果为真，设备受控进入状态 1。 默认值为假。
Prog2Command	BOOL	程序给定状态 2。此输入用于确定设备处于程序控制模式时的设备状态。如果为真，设备受控进入状态 2。 默认值为假。
Oper0Req	BOOL	操作员状态 0 请求。设备处于操作员控制模式时，通过操作员界面设置为真，将设备置为状态 0。 默认值为假。
Oper1Req	BOOL	操作员状态 1 请求。设备处于操作员控制模式时，通过操作员界面设置为真，将设备置为状态 1。 默认值为假。
Oper2Req	BOOL	操作员状态 2 请求。设备处于操作员控制模式时，通过操作员界面设置为真，将设备置为状态 2。 默认值为假。

输入参数	数据类型	说明
State0Perm	BOOL	状态 0 选通。除非在手控或超控模式下，否则此输入必须为真才能使设备进入状态 0。如果设备已经处于状态 0 ,则此输入无效。 默认值为真。
State1Perm	BOOL	状态 1 选通。除非在手控或超控模式下，否则此输入必须为真才能使设备进入状态 1。如果设备已经处于状态 1 ,则此输入无效。 默认值为真。
State2Perm	BOOL	状态 2 选通。除非在手控或超控模式下，否则此输入必须为真才能使设备进入状态 2。如果设备已经处于状态 2 ,则此输入无效。 默认值为真。
FB0	BOOL	可供指令使用的第一个反馈输入。 默认值为假。
FB1	BOOL	可供指令使用的第二个反馈输入。 默认值为假。
FB2	BOOL	可供指令使用的第三个反馈输入。 默认值为假。
FB3	BOOL	可供指令使用的第四个反馈输入。 默认值为假。
HandFB0	BOOL	手控反馈状态 0。该输入来自现场手控/关/自动站 , 用于显示所请求的现场设备状态。为真时，表示请求现场设备进入状态 0；为假时，表示请求现场设备进入其他某种状态。 默认值为假。

输入参数	数据类型	说明
HandFB1	BOOL	手控反馈状态 1。该输入来自现场手控/关/自动站，用于显示所请求的现场设备状态。为真时，表示请求现场设备进入状态 1；为假时，表示请求现场设备进入其他某种状态。 默认值为假。
HandFB2	BOOL	手控反馈状态 2。该输入来自现场手控/关/自动站，用于显示所请求的现场设备状态。为真时，表示请求现场设备进入状态 2；为假时，表示请求现场设备进入其他某种状态。 默认值为假。
FaultTime	REAL	故障时间值。以秒为单位配置时间值以使设备达到最新给定的状态。设置 FaultTime = 0 可禁用故障计时器。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 任意 ≥ 0.0 的浮点值 默认值 = 0.0
FaultAlarmLatch	BOOL	故障报警锁定输入。此参数和 FaultAlarm 同时为真时，将锁定 FaultAlarm。要启用 FaultAlarm，可将 FaultAlmUnlatch 设置为真或将 FaultAlarmLatch 设置为假。 默认值为假。
FaultAlmUnLatch	BOOL	故障报警启用输入。在 FaultAlarmLatch 置位时将此输入设置为真可启用 FaultAlarm。指令会将此输入设置为假。 默认值为假。
OverrideOnInit	BOOL	初始化时超控请求。如果该位为真，则在指令首次扫描期间，指令将进入操作员控制模式，且 Override 为真，Hand 为假。如果 ProgHandReq 为真，则 Override 设置为假，Hand 设置为真。 默认值为假。

输入参数	数据类型	说明
OverrideOnFault	BOOL	故障时超控请求。如果希望设备在发生故障报警时转为超控模式并进入超控状态，将此值设置为真。消除故障报警后，指令进入操作员控制模式。 默认值为假。
Out0State0	BOOL	输出 0 状态 0 输入。该值用于确定设备处于状态 0 时 Output0 的值。 默认值为假。
Out0State1	BOOL	输出 0 状态 1 输入。该值用于确定设备处于状态 1 时 Output0 的值。 默认值为假。
Out0State2	BOOL	输出 0 状态 2 输入。该值用于确定设备处于状态 2 时 Output0 的值。 默认值为假。
Out1State0	BOOL	输出 1 状态 0 输入。该值用于确定设备处于状态 0 时 Output1 的值。 默认值为假。
Out1State1	BOOL	输出 1 状态 1 输入。该值用于确定设备处于状态 1 时 Output1 的值。 默认值为假。
Out1State2	BOOL	输出 1 状态 2 输入。该值用于确定设备处于状态 2 时 Output1 的值。 默认值为假。
Out2State0	BOOL	输出 2 状态 0 输入。该值用于确定设备处于状态 0 时 Output2 的值。 默认值为假。
Out2State1	BOOL	输出 2 状态 1 输入。该值用于确定设备处于状态 1 时 Output2 的值。 默认值为假。

输入参数	数据类型	说明
Out2State2	BOOL	输出 2 状态 2 输入。该值用于确定设备处于状态 2 时 Output2 的值。 默认值为假。
OverrideState	DINT	超控状态输入。设置此值来指示处于超控模式时的设备状态。 2 = 设备进入状态 2 1 = 设备进入状态 1 0 = 设备进入状态 0 该值无效时会将 Status 中的相应位置位。 有效值 = 0 到 2 默认值 = 0
FB0State0	BOOL	反馈 0 状态 0 输入。该值用于确定设备处于状态 0 时 FB0 的预期值。 默认值为假。
FB0State1	BOOL	反馈 0 状态 1 输入。该值用于确定设备处于状态 1 时 FB0 的预期值。 默认值为假。
FB0State2	BOOL	反馈 0 状态 2 输入。该值用于确定设备处于状态 2 时 FB0 的预期值。 默认值为假。
FB1State0	BOOL	反馈 1 状态 0 输入。该值用于确定设备处于状态 0 时 FB1 的预期值。 默认值为假。
FB1State1	BOOL	反馈 1 状态 1 输入。该值用于确定设备处于状态 1 时 FB1 的预期值。 默认值为假。
FB1State2	BOOL	反馈 1 状态 2 输入。该值用于确定设备处于状态 2 时 FB1 的预期值。 默认值为假。

输入参数	数据类型	说明
FB2State0	BOOL	反馈 2 状态 0 输入。该值用于确定设备处于状态 0 时 FB2 的预期值。 默认值为假。
FB2State1	BOOL	反馈 2 状态 1 输入。该值用于确定设备处于状态 1 时 FB2 的预期值。 默认值为假。
FB2State2	BOOL	反馈 2 状态 2 输入。该值用于确定设备处于状态 2 时 FB2 的预期值。 默认值为假。
FB3State0	BOOL	反馈 3 状态 0 输入。该值用于确定设备处于状态 0 时 FB3 的预期值。 默认值为假。
FB3State1	BOOL	反馈 3 状态 1 输入。该值用于确定设备处于状态 1 时 FB3 的预期值。 默认值为假。
FB3State2	BOOL	反馈 3 状态 2 输入。该值用于确定设备处于状态 2 时 FB3 的预期值。 默认值为假。
ProgProgReq	BOOL	程序发出的程序控制请求。由用户程序设置为真可请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。保持该参数为真且保持 ProgOperReq 为假可将指令锁定为程序控制模式。 默认值为假。
ProgOperReq	BOOL	程序发出的操作员控制请求。由用户程序设置为真可请求操作员控制模式。保持该参数为真可将指令锁为操作员控制模式。 默认值为假。

输入参数	数据类型	说明
ProgOverrideReq	BOOL	程序超控模式请求。由用户程序设置为真可请求进入超控模式。如果 ProgHandReq 为真，则忽略此参数。 默认值为假。
ProgHandReq	BOOL	程序手控模式请求。由用户程序设置为真可请求设备进入手控模式。 默认值为假。
OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。指令会将此输入设置为假。 默认值为假。
OperOperReq	BOOL	操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。指令会将此输入设置为假。 默认值为假。
ProgValueReset	BOOL	将程序控制值复位。该参数为真时，每次执行指令时，所有程序请求输入都将设置为假。 默认值为假。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Out0	BOOL	指令的第一个输出。
Out1	BOOL	指令的第二个输出。
Out2	BOOL	指令的第三个输出。
Device0State	BOOL	设备状态 0 输出。设备受控进入状态 0 且反馈表明设备确实处于状态 0 时为真。
Device1State	BOOL	设备状态 1 输出。设备受控进入状态 1 且反馈表明设备确实处于状态 1 时为真。

Device2State	BOOL	设备状态 2 输出。设备受控进入状态 2 且反馈表明设备确实处于状态 2 时为真。
Command0Status	BOOL	设备状态 0 受控状态。设备受控进入状态 0 时为真 ;设备受控进入其他状态时为假。
Command1Status	BOOL	设备状态 1 受控状态。设备受控进入状态 1 时为真 ;设备受控进入其他状态时为假。
Command2Status	BOOL	设备状态 2 受控状态。设备受控进入状态 2 时为真 ;设备受控进入其他状态时为假。
FaultAlarm	BOOL	故障报警输出。如果设备受控进入新状态，但经过 FaultTime 后反馈没有表明设备确实达到新状态，则此参数为真。此外，如果在达到给定状态后，反馈突然表明设备不再处于给定状态，该参数也会设置为真。
ModeAlarm	BOOL	模式报警输出。如果设备处于操作员控制模式，ProgxCommand 输入请求了不同于操作员当前给定的状态，此参数为真。此报警旨在提醒已将设备保留为操作员控制模式。
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。处于操作员控制模式时 为假。
Override	BOOL	超控模式。当设备处于超控模式时为真。
Hand	BOOL	手控模式。当设备处于手控模式时为真。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
FaultTimeInv (Status.1)	BOOL	FaultTime 值无效。指令将设置 FaultTime = 0。

OverrideStateInv (Status.2)	BOOL	超控值超出范围。它可以防止指令进入超控状态。
ProgCommandInv (Status.3)	BOOL	多个程序状态命令位同时置位。请参见"程序控制模式下的给定状态"部分。
OperReqInv (Status.4)	BOOL	多个操作员状态请求位同时置位。请参见"程序控制模式下的给定状态"部分。
HandCommandInv (Status.5)	BOOL	多个手控反馈状态请求位同时置位。

说明

D3SD 指令控制有三种可能状态（如快/慢/停或正向/停止/反向）的离散设备。具备这种特性的典型离散设备包括进料系统和可逆电机。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行

指令首次运行	将 ProgOper 设为操作员模式。 将 Command0Status 设为真。 Command1Status 设为假。 Command2Status 设为假。
指令首次扫描	故障计时器清零。 ModeAlarm 设置为假。 将所有操作员请求输入清零。 如果 ProgValueReset 为真，则将所有程序请求输入清零。 当 OverrideOnInit 为真时，ProgOper 设置为假（操作员控制模式）。 如果 ProgHandReq 为假并且 OverrideOnInit 为真，则 Hand 设置为假且 Override 设为真（超控模式）。 如果 ProgHandReq 为真，则 Hand 设为真且 Override 设置为假（手控模式）。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。

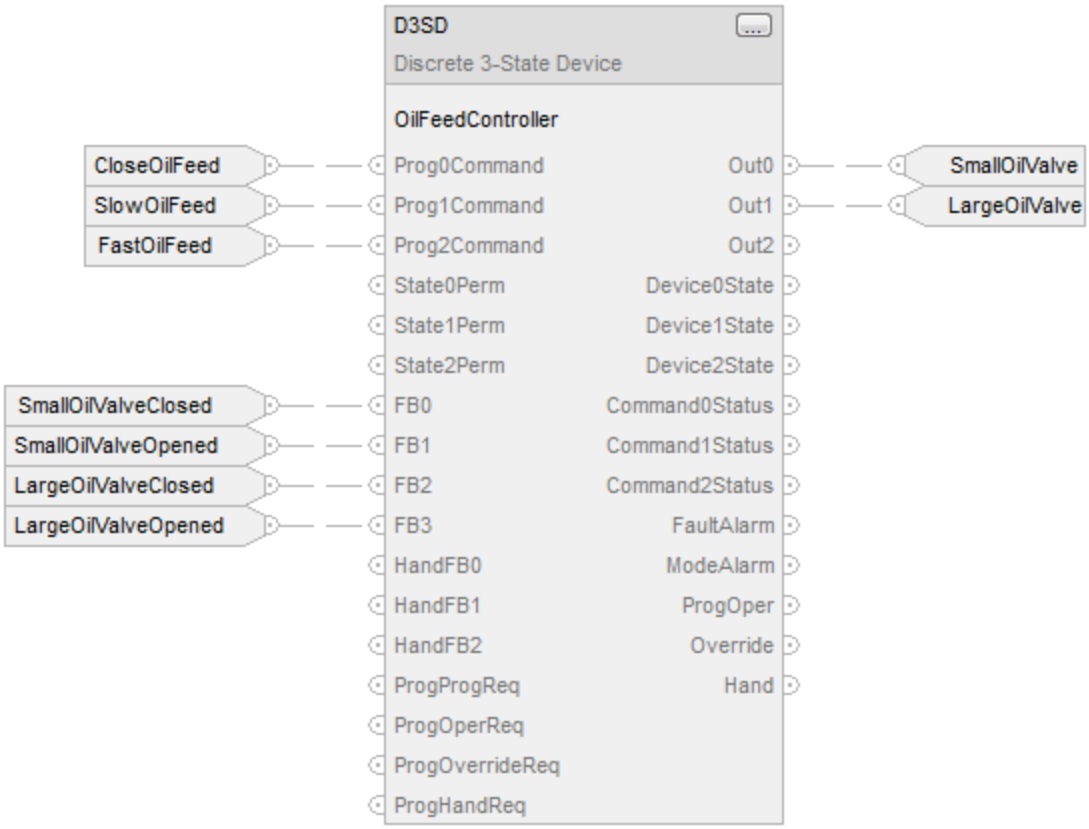
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

D3SD 指令通常用于控制三态设备，如快/慢/关进料系统。在本例中，D3SD 指令用于控制由一对电磁阀组成的进料系统，进而控制将植物油加入批处理槽这一过程。其中一个阀门位于连接到批处理槽中的大口径进料管上，另一个阀门则并排放置在小口径进料管上。最初添加植物油时，D3SD 指令受控进入快速进料状态（状态 2），此时两个阀门均打开。当添加的植物油接近目标量时，D3SD 指令受控进入慢速进料状态（状态 1），即“大阀门”关闭、“小阀门”保持打开转改。达到目标油量后，D3SD 指令受控进入停止状态（状态 0）并且关闭上述两个阀门。只要 D3SD 指令处于程序控制模式，这两个阀门就会根据 CloseOilFeed、SlowOilFeed 和 FastOilFeed 输入的值相应地打开。操作员也可以根据需要对进料系统采取操作员控制模式。本例中的电磁阀配有限位开关，用于指示阀门何时完全关闭或打开。这些开关与 FB0、FB1、FB2 和 FB3

反馈输入相连。因此，如果电磁阀未在组态的 FaultTime 内达到给定状态，D3SD 指令会生成 FaultAlarm。

功能块



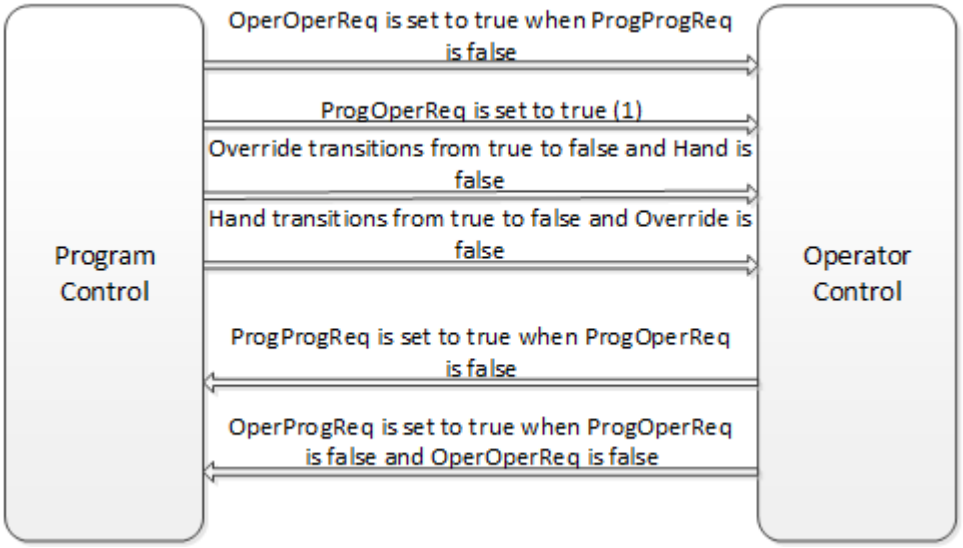
结构化文本

```
OilFeedController.Prog0Command := ClosedOilFeed;

OilFeedController.Prog1Command := SlowOilFeed;

OilFeedController.Prog1Command := FastOilFeed;

OilFeedController.FB0 := SmallOilValveClosed;

OilFeedController.FB1 := SmallOilValveOpened;

OilFeedController.FB2 := LargeOilValveClosed;

OilFeedController.FB3 := LargeOilValveOpened;

D3SD(OilFeedController);

SmallOilValve := OilFeedController.Out0;
```

LargeOilValve := OilFeedController.Out1;

在程序控制与操作员控制之间切换

下图显示了 D3SD 指令在程序控制与操作员控制模式之间进行切换的方式。



(1) 当 ProgOperReq 为真时，指令保持在操作员控制模式。

程序控制模式下的给定状态

下表所示为 D3SD 指令在程序控制模式下的工作情况。

Prog0 Command	Prog1 Command	Prog2 Command	State0 Perm	State1 Perm	State2 Perm	说明
假	假	真	真/假	真/假	真	Command0Status 设置为假 Command1Status 设置为假 Command2Status 设置为真
假	真	假	真/假	真	真/假	Command0Status 设置为假 Command1Status 设置为真 Command2Status 设置为假
真	假	假	真	真/假	真/假	Command0Status 设置为真 Command1Status 设置为假 Command2Status 设置为假

如果多个程序命令输入为真：

- 指令会将 Status 中的相应位置位

- 如果 Override 和 Hand 设置为假，指令将保持前一状态。

操作员控制模式下的给定状态

下表所示为 D3SD 指令在操作员控制模式下的工作情况。

Oper0Req	Oper1Req	Oper2Req	State0 Perm	State1 Perm	State2 Perm	说明
假	假	真	真/假	真/假	真	Command0Status 设置为假 Command1Status 设置为假 Command2Status 设置为真
假	真	假	真/假	真	真/假	Command0Status 设置为假 Command1Status 设置为真 Command2Status 设置为假
真	假	假	真	真/假	真/假	Command0Status 设置为真 Command1Status 设置为假 Command2Status 设置为假

如果多个操作员命令输入为真：

- 指令会将 Status 中的相应位置位
- 如果 Override 和 Hand 设置为假，指令将保持前一状态。

每次执行指令后，指令都会执行以下操作：

- 将所有操作员请求输入清零
- 如果 ProgValueReset 为真，则将所有的程序请求输入设置为假。

手控模式或超控模式

下表说明 D3SD 指令确定以手控模式还是超控模式工作的方式。

ProgHandReq	ProgOverrideReq	FaultAlarm 和 OverrideOnFault	说明
真	真/假	真/假	手控模式 Hand 设置为真 Override 设置为假
假	真	真/假	超控模式 Hand 设置为假 Override 设置为真

假	真/假	真	超控模式 Hand 设置为假 Override 设置为真

当 Override 置位时，超控模式的优先级将高于程序控制和操作员控制模式。下表说明了超控模式影响给定状态的方式。

Override	超控状态	说明
真	2	Command0Status 设置为假 Command1Status 设置为假 Command2Status 设置为真
真	1	Command0Status 设置为假 Command1Status 设置为真 Command2Status 设置为假
真	0	Command0Status 设置为真 Command1Status 设置为假 Command2Status 设置为假

如果 OverrideState 无效，指令会将 Status 中的相应位置位，并且不会进入超控状态。

当 Hand 为真时，手控模式的优先级将高于程序控制和操作员控制模式。下表说明了手控模式影响给定状态的方式。

Hand	HandFB0	HandFB1	HandFB2	说明
真	假	假	真	Command0Status 设置为假 Command1Status 设置为假 Command2Status 设置为真
真	假	真	假	Command0Status 设置为假 Command1Status 设置为真 Command2Status 设置为假
真	真	假	假	Command0Status 设置为真 Command1Status 设置为假 Command2Status 设置为假

如果多个 HandFB 输入为真，指令会将 Status 中的相应位置位；如果 Hand 为真，指令将保持前一状态。

输出状态

D3SD 的输出状态取决于受控状态的状态。

CommandStatus	输出状态
Command0Status 为真	Out0 = Out0State0 Out1 = Out1State0 Out2 = Out2State0
Command0Status 为真 FB0 = FB0State0 FB1 = FB1State0 FB2 = FB2State0 且 FB3 = FB3State0	停止故障计时器并将其清零。 Device0State 设置为真
Command1Status 为真	Out0 = Out0State1 Out1 = Out1State1 Out2 = Out2State1
Command1Status 为真 FB0 = FB0State1 FB1 = FB1State1 FB2 = FB2State1 且 FB3 = FB3State1	停止故障计时器并将其清零。 Device1State 设置为真
Command2Status 为真	Out0 = Out0State2 Out1 = Out1State2 Out2 = Out2State2
Command2Status 为真 FB0 = FB0State2 FB1 = FB1State2 FB2 = FB2State2 且 FB3 = FB3State2	停止故障计时器并将其清零。 Device2State 设置为真

故障报警条件

D3SD 指令将检查以下故障报警条件。

产生故障报警条件的原因	规则
设备状态受控进行更改，但反馈并未指示在 FaultTime 内确实达到目标状态。	当 $Command0Status_n \neq Command0Status_{n-1}$、或 $Command1Status_n \neq Command1Status_{n-1}$、或 $Command2Status_n \neq Command2Status_{n-1}$ 时，启动故障计时器 当故障计时器计时完成且 FaultTime > 0.0 时，将 FaultAlarm 置位
设备在未受到控制的情况下意外退出某个状态（根据反馈）。	当故障计时器未计时且满足以下任一条件时，将 FaultAlarm 设置为真： Command0Status 为真且 Device0State 为假 Command1Status 为真且 Device1State 为假 Command2Status 为真且 Device2State 为假

如果不存在任何故障，则满足以下任一条件后，FaultAlarm 将设置为假：

- Command0Status 为真且 Device0State 为真

- Command1Status 为真且 Device1State 为真
- Command2Status 为真且 Device2State 为真
- FaultTime ≤ 0

当 FaultAlarmLatch 为真时，无法将 FaultAlarm 设置为假，除非 FaultAlmUnlatch 为真且不存在任何故障。

模式报警条件

模式报警可提醒操作员已将设备置于操作员控制模式。仅当程序处于操作员控制下，尝试将设备状态从操作员的给定状态更改为其他状态时，模式报警才会启用。如果操作员将设备置于操作员控制模式并更改状态，报警不会启用。D3SD 指令将按照以下规则检查模式报警条件。

ModeAlarm 为	条件
真	Prog2Command ≠ $Prog2Command_{n-1}$ **且** Prog2Command ≠ Command2Status，**或** Prog1Command ≠ $Prog1Command_{n-1}$ **且** Prog1Command ≠ Command1Status，**或** Prog0Command ≠ $Prog1Command_{n-1}$ **且** Prog0Command ≠ Command0Status
假	Prog2Command = Command2Status 且 Prog1Command = Command1Status 且 Prog0Command = Command0Status 或 设备处于超控、手动或程序控制模式

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

离散 2 态设备 (D2SD)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

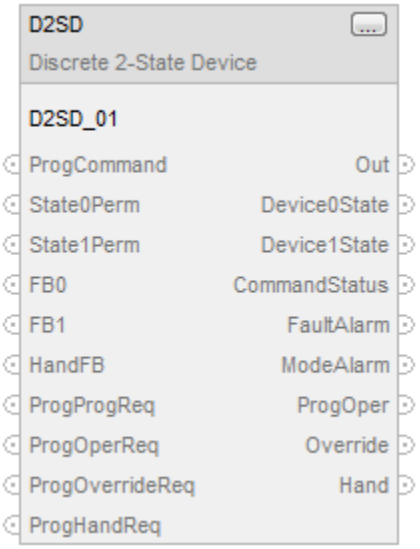
D2SD 指令控制仅有两个可能状态（如开/关或断开/闭合）的离散设备。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

D2SD(D2SD_tag)

操作数

对指令内的混合数据类型有相应的数据转换规则。请参见*数据转换*部分。

结构化文本

操作数	类型	格式	说明
D2SD tag	DISCRETE_2STATE	结构	D2SD 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

功能块

操作数	类型	格式	说明
D2SD tag	DISCRETE_2STATE	结构	D2SD 结构

DISCRETE_2STATE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
ProgCommand	BOOL	用于确定设备处于程序控制模式时的 CommandStatus。为真时，指示设备受控进入状态 1；为假时，指示设备受控进入状态 0。 默认值为假。
Oper0Req	BOOL	操作员状态 0 请求。设备处于操作员控制模式时，通过操作员界面置位，将设备置为状态 0。 默认值为假。
Oper1Req	BOOL	操作员状态 1 请求。设备处于操作员控制模式时，通过操作员界面置位，将设备置为状态 1。 默认值为假。
State0Perm	BOOL	状态 0 选通。除非在手控或超控模式下，否则必须将该输入置位才能使设备进入状态 0。该输入对于已经处于状态 0 的设备无效。 默认值为真。
State1Perm	BOOL	状态 1 选通。除非在手控或超控模式下，否则必须将该输入置位才能使设备进入状态 1。该输入对于已经处于状态 1 的设备无效。 默认值为真。
FB0	BOOL	D2SD 指令可用的第一个反馈输入。 默认值为假。
FB1	BOOL	D2SD 指令可用的第二个反馈输入。 默认值为假。
HandFB	BOOL	手控反馈输入。该输入来自现场手动/关/自动站，用于显示所请求的现场设备状态。为真时，请求现场设备进入状态 1；为假时，请求现场设备进入状态 0。 默认值为假。

FaultTime	REAL	故障时间值。以秒为单位配置时间值以使设备达到最新给定的状态。设置 FaultTime = 0 可禁用故障计时器。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 任意 ≥ 0.0 的浮点值 默认值 = 0.0
FaultAlarmLatch	BOOL	故障报警锁定输入。此参数和 FaultAlarm 同时为真时，将锁定 FaultAlarm。要启用 FaultAlarm，将 FaultAlmUnlatch 设置为真或将 FaultAlarmLatch 设置为假。 默认值为假。
FaultAlmUnLatch	BOOL	故障报警启用输入。在 FaultAlarmLatch 置位时设置 FaultAlmUnLatch 可启用 FaultAlarm。指令会将此输入设置为假。 默认值为假。
OverrideOnInit	BOOL	初始化时超控请求。如果该位为真，则在指令初次扫描期间，2 态设备进入操作员控制模式，Override 设置为真，Hand 设置为假。如果 ProgHandReq 为真，则 Override 设置为假，Hand 设置为真。 默认值为假。
OverrideOnFault	BOOL	故障时超控请求。如果希望设备在发生故障报警时转为超控模式并进入超控状态，将 OverrideOnFault 设置为真。消除故障报警后，2 态设备进入操作员控制模式。 默认值为假。
OutReverse	BOOL	反向默认输出状态。当受控进入状态 0 时，Out 的默认状态设置为假，当受控进入状态 1 时设置为真。若 OutReverse 为真，则在受控进入状态 0 时，Out 设置为真，在受控进入状态 1 时，Out 设置为假。 默认值为假。
OverrideState	BOOL	超控状态输入。配置此值来指定设备处于超控模式时的设备状态。真表示设备应进入状态 1；假表示设备应进入状态 0。 默认值为假。
FB0State0	BOOL	反馈 0 状态 0 输入。配置设备处于状态 0 时 FB0 的状态。 默认值为假。

FB0State1	BOOL	反馈 0 状态 1 输入。配置设备处于状态 1 时 FB0 的状态。 默认值为假。
FB1State0	BOOL	反馈 1 状态 0 输入。配置设备处于状态 0 时 FB1 的状态。 默认值为假。
FB1State1	BOOL	反馈 1 状态 1 输入。配置设备处于状态 1 时 FB1 的状态。 默认值为假。
ProgProgReq	BOOL	程序发出的程序控制请求。由用户程序设置为真可请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。保持该参数为真且 ProgOperReq 为假可将指令锁定为程序控制模式。 默认值为假。
ProgOperReq	BOOL	程序发出的操作员控制请求。由用户程序设置为真可请求操作员控制模式。保持该参数为真可将指令锁定为操作员控制模式。 默认值为假。
ProgOverrideReq	BOOL	程序超控模式请求。由用户程序设置为真可请求进入超控模式。如果 ProgHandReq 为真，则忽略此参数。 默认值为假。
ProgHandReq	BOOL	程序手控模式请求。由用户程序设置为真可请求设备进入手控模式。 默认值为假。
OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。指令会将此输入设置为假。 默认值为假。
OperOperReq	BOOL	操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。指令会将此输入设置为假。 默认值为假。
ProgValueReset	BOOL	将程序控制值复位。该参数为真时，每次执行指令时，所有程序请求输入都将设置为假。 默认值为假。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Out	BOOL	2 态指令的输出。
Device0State	BOOL	设备状态 0 输出。指示设备受控进入状态 0 且反馈表明设备确实处于状态 0 时设置为真。
Device1State	BOOL	设备状态 1 输出。当设备受控进入状态 1 且反馈表明设备确实处于状态 1 时设置为真。
CommandStatus	BOOL	命令状态输出。当设备受控进入状态 1 时设置为真，受控进入状态 0 时清零。
FaultAlarm	BOOL	故障报警输出。如果指示设备受控进入新状态，但经过 FaultTime 后反馈没有表明设备确实达到新状态，则此参数设置为真。此外，如果在达到给定状态后，反馈突然表明设备不再处于给定状态，该参数也会设置为真。
ModeAlarm	BOOL	模式报警输出。如果设备处于操作员控制，但程序命令切换为与操作员指示的当前状态不同的状态，则此参数设置为真。此报警旨在提醒已将设备保留为操作员控制模式。
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。操作员控制模式下为假。
Override	BOOL	超控模式。当设备处于超控模式时为真。
Hand	BOOL	手控模式。当设备处于手控模式时为真。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
FaultTimeInv (Status.1)	BOOL	FaultTime 值无效。指令将设置 FaultTime = 0 。
OperReqInv (Status.2)	BOOL	两个操作员状态请求位均为真。

说明

D2SD 指令控制仅有两个可能状态（如开/关或断开/闭合）的离散设备。具备这种特性的典型离散设备包括电机、泵和电磁阀。

监视 D2SD 指令

D2SD 指令有相应的操作员面板。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	将 ProgOper 设置为操作员模式。将 CommandStatus 设置为假。
指令首次扫描	将 EnableOut 设置为真。 ModeAlarm 和操作员请求输入设置为假， 如果 ProgValueReset 为真，则将所有程序请求输入清零。 当 OverrideOnInit 为真时，ProgOper 设置为假（操作员控制）。 如果 ProgHandReq 清零且 OverrideOnInit 置位，则清零 Hand 并置位 Override（超控模式）。 如果置位 ProgHandReq，请置位 Hand 并清零 Override（手动模式）。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

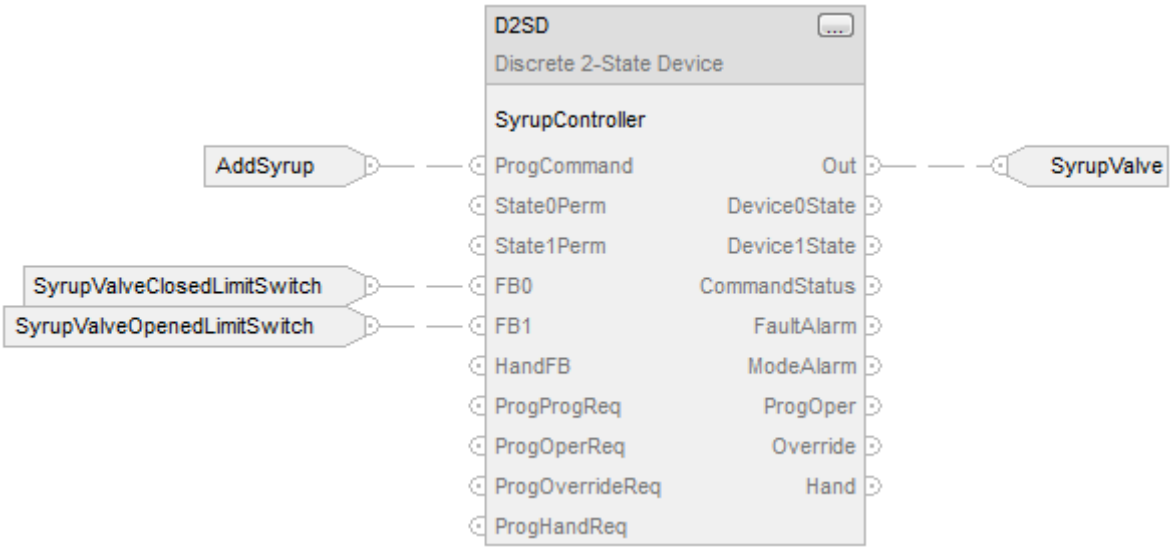
在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

SD 指令通常用于控制开/关或打开/关闭设备，如泵或电磁阀。在此示例中，D2SD 指令控制一个电磁阀，从而控制将玉米糖浆加入批量槽的过程。只要 D2SD 指令处于程序控制模式，阀门便会在 AddSyrup 输入置位时打开。操作员还可以在必要时通过操作员方式控制阀门的打开或关闭。本例中的电磁阀采用了限位开关来指示阀门完全关闭或打开的时刻。这些开关与 FB0 和 FB1 反馈输入相连。这样，如果电磁阀未在组态的 FaultTime 内达到给定状态，D2SD 指令便会生成 FaultAlarm。

功能块



结构化文本

SyrupController.ProgCommand := AddSyrup;

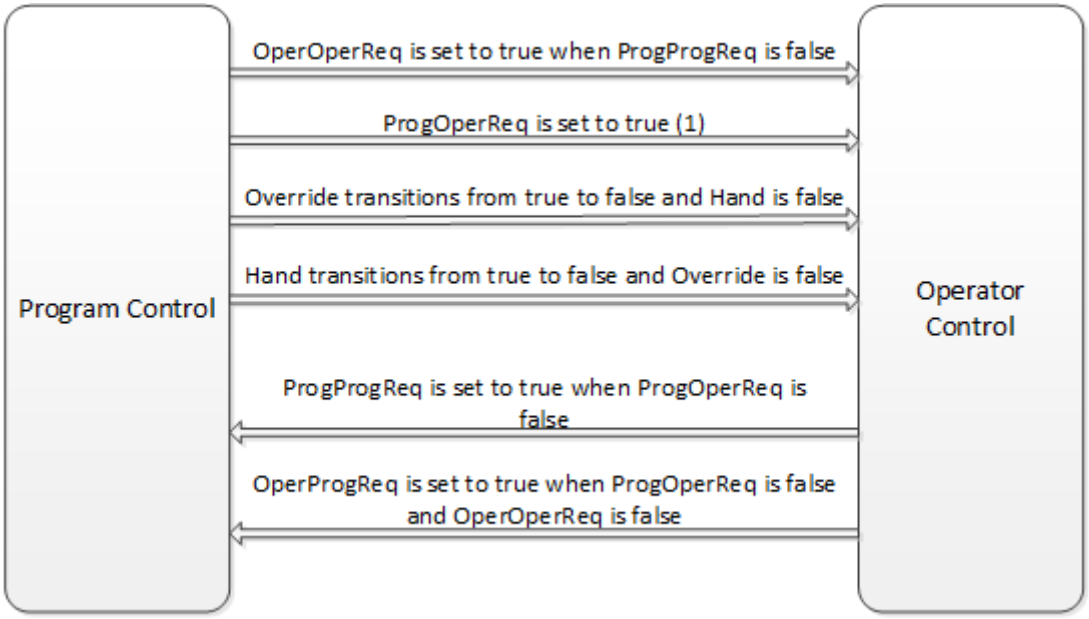
SyrupController.FB0 := SyrupValveClosedLimitSwitch;

SyrupController.FB1 := SyrupValveOpenedLimitSwitch;

```
D2SD(SyrupController);

SyrupValve := SyrupController.Out;
```

在程序控制与操作员控制之间切换

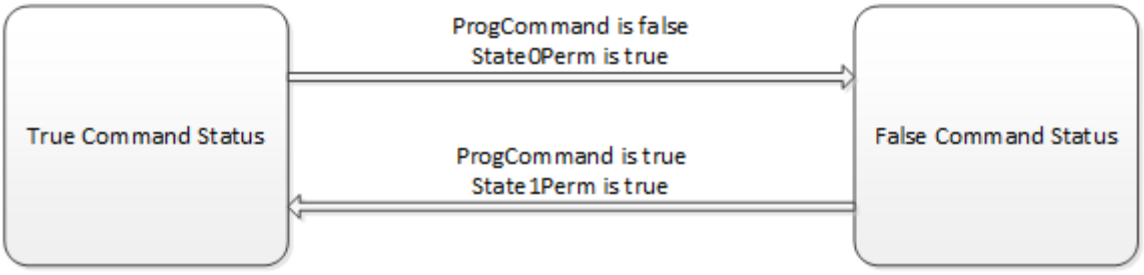
下图显示了 D2SD 指令如何在程序控制与操作员控制之间进行切换。



(1) 当 ProgOperReq 为真时，指令保持在操作员控制模式。

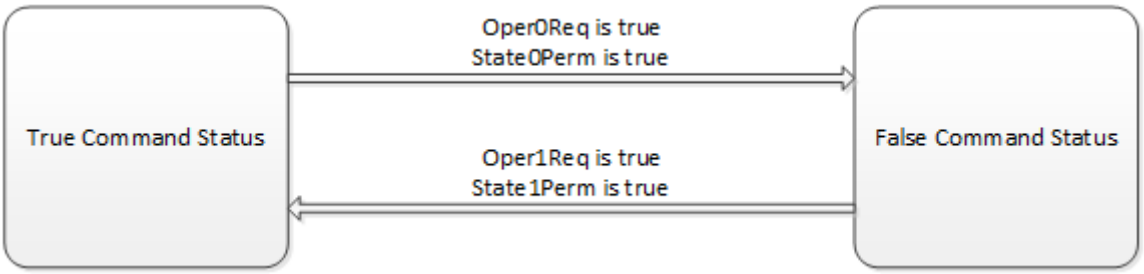
程序控制模式下的给定状态

下图说明了 D2SD 指令在程序控制模式下的工作情况。



操作员控制模式下的给定状态

下图说明了 D2SD 指令在操作员控制模式下的工作情况。



如果 Oper0Req 和 Oper1Req 均为真：

- 指令将 Status 中的相应位设置为真
- 如果 Override 和 Hand 为假，则指令保持前一状态。

每次执行指令后，指令都会执行以下操作：

- 将所有操作员请求输入设置为假
- 如果 ProgValueReset 为真，则将所有的程序请求输入设置为假。

手控模式或超控模式

下表说明 D2SD 指令确定以手控模式还是超控模式工作的方式。

ProgHandReq	ProgOverrideReq	FaultAlarm 和 OverrideOnFault	说明
真	真/假	真/假	手控模式 Hand 设置为真 Override 设置为假
假	真	真/假	超控模式 Hand 设置为假 Override 设置为真
假	真/假	真	超控模式 Hand 设置为假 Override 设置为真

指令处于超控模式时，CommandStatus = OverrideState。

指令处于手控模式时，CommandStatus = HandFB。

输出状态

D2SD 输出状态取决于命令状态的状态。

CommandStatus	输出状态
假	若 OutReverse 为假，则 Out 设置为假 若 OutReverse 为真，则 Out 设置为真
真	若 OutReverse 为假，则 Out 设置为真 若 OutReverse 为真，则 Out 设置为假
为假且 FB0 = FB0State0 且 FB1 = FB1State0	故障计时器停止并清除为 0 Device0State 设置为真
为真且 FB0 = FB0State1 且 FB1 = FB1State1	故障计时器停止并清除为 0 Device1State 设置为真

故障报警条件

D2SD 指令检查以下故障报警条件。

故障报警条件 的原因	规则
设备状态受控进行更改，但反馈并未指示在 FaultTime 内确实达到目标状态。	$CommandStatus_n \neq CommandStatus_{n-1}$ 时启动故障计时器 当故障计时器完成且 FaultTime > 0.0 时，将 FaultAlarm 置位
设备在未受到控制的情况下意外退出某个状态（根据反馈）。	当故障计时器未计时且满足以下任一条件时，将 FaultAlarm 设置为真： CommandStatus 为假且 Device0State 为假 CommandStatus 为真且 Device1State 为假

满足以下任一条件时，FaultAlarm 设置为假：

- CommandStatus 为假且 Device0State 为真
- CommandStatus 为真且 Device1State 为真
- FaultTime $\leq$ 0

当 FaultAlarmLatch 为真时，无法将 FaultAlarm 设置为假，除非 FaultAlmUnlatch 为真且不存在任何故障。

模式报警条件

模式报警可提醒操作员已将设备置于操作员控制模式。仅当程序处于操作员控制下，尝试将设备状态从操作员的给定状态更改为其他状态时，模式报警才会启用。如果操作员将设备置于操作员控制模式并更改状态，报警不会启用。D2SD 指令将按照以下规则检查模式报警条件。

ModeAlarm	条件
真	$ProgCommand_n \neq ProgCommandn\text{-}1$ 且 $ProgCommand_n \neq CommandStatus$
假	ProgCommand = CommandStatus 或 设备处于超控、手控或程序控制模式

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

数据转换 参考页数 592

死区时间 (DEDT)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

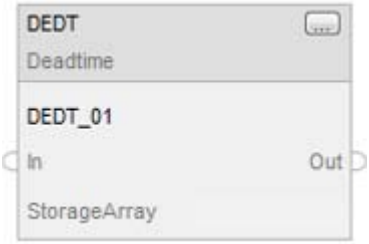
DEDT 指令用于执行单路输入的延时。死区时间延时可以选择。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

DEDT(DEDT_tag,storage);

操作数

结构化文本

操作数	类型	格式	说明
DEDT tag	DEADTIME	结构	DEDT 结构
storage	REAL	数组	死区时间缓冲区

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

功能块

操作数	类型	格式	说明
DEDT tag	DEADTIME	结构	DEDT 结构
storage	REAL	数组	死区时间缓冲区

DEADTIME 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
InFault	BOOL	输入不良状况指示器。如果输入值从模拟量输入读取，则 InFault 由模拟量输入的故障状态控制。InFault 为真时，说明输入信号存在错误，此指令会将 Status 中的相应位置位，控制算法不会执行且 Out 处于保持状态。 默认值为假。 假 = 状况良好

Deadtime	REAL	指令的死区时间输入。以秒为单位输入死区时间。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 0.0 到（StorageArray 的大小 * DeltaT） 默认值 = 0.0
Gain	REAL	指令的增益输入。In 的值将与该值相乘。这样，可以模拟过程增益。 有效值 = 任意浮点值 默认值 = 1.0
Bias	REAL	指令的偏置输入。In 的值将乘以 Gain，然后再与该值相加。这样，可以模拟环境条件。 有效值 = 任意浮点值 默认值 = 0.0
TimingMode	DINT	选择时序执行模式。 0 = 周期模式 1 = 过采样模式 2 = 实时采样模式 有效值 = 0 至 2 默认值 = 0 有关时序模式的更多信息，请参见“功能块属性”部分。
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。

Out	REAL	计算所得的死区时间算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InFaulted (Status.1)	BOOL	In 状况不良。
DeadtimeInv (Status.2)	BOOL	Deadtime 值无效。
TimingMode (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS(DeltaT - RTSTime) > 1 毫秒时置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

说明

DEDT 指令使用数据缓冲区存储延迟的数据，因此支持使用任意时长的所需死区时间。DEDT 指令设计为在扫描速率保持恒定的任务中执行。

要使用 DEDT 指令，需创建一个存储数组，存储用于保存 (In x Gain) + Bias 采样值的死区时间缓冲区。存储数组的大小应足以容纳所需的最长死区时间，计算公式如下：

所需 StorageArray 的大小 = 最大死区时间（秒）/DeltaT（秒）

提供死区时间缓冲区服务

运行期间，该指令将检查 Deadtime 是否有效。Deadtime 必须介于 0.0 和（StorageArray 的大小 x DeltaT）之间。

如果 Deadtime 无效，该指令会将 Status 中的相应位置位，并设置 Out = (In x Gain) + Bias。

死区时间缓冲区用作先进先出缓冲区。每次死区时间算法执行时，最先进入死区时间缓冲区的值都会移到 Out 中。缓冲区中的其余值下移，而值 ((Inx Gain) + Bias) 则移至死区时间缓冲区的开始位置。经过 Deadtime 秒后，放入死区时间缓冲区中的新值将显示在 Out 中。

执行所设延时所需的数组元素数按 Deadtime 除以 DeltaT 计算。如果 Deadtime 不能被 DeltaT 整除，则数组元素数将四舍五入为最接近的整数，所设延时则四舍五入为最接近的 DeltaT 增量。例如，给定 Deadtime = 4.25s 且 DeltaT = 0.50s，执行所设延时所需的数组元素数计算如下：

4.25s / 0.50s = 8.5
所需数组元素数为 9

本例中应用于输入的实际延时为：

数组元素数 x DeltaT = 所设延时，即
9 x 0.5s = 4.5s

若在运行期间更改 Deadtime 或 DeltaT，将改变各个值从缓冲区中移出时对应的时间点。执行所设延时所需的元素数既可以增大也可以减小。提供死区时间缓冲区服务之前，将执行以下更新：

如果所需元素数需要增大，则使用当前死区时间缓冲区中最先进入的值填充新的缓冲区元素。

如果所需元素数需要减小，则放弃当前死区时间缓冲区中最先进入的元素。

InFault 跳变时指令的行为

InFault 为真（状况不良）时，指令将暂停执行，保持上一个输出并将 Status 中的相应位置位。

当 InFault 由真跳变为假时，指令会将死区时间缓冲区中的所有值设置为 In x Gain + Bias。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	不适用 该指令不会执行，但确实验证输入参数。
后扫描	EnableIn 和 EnableOut 位设置为假。

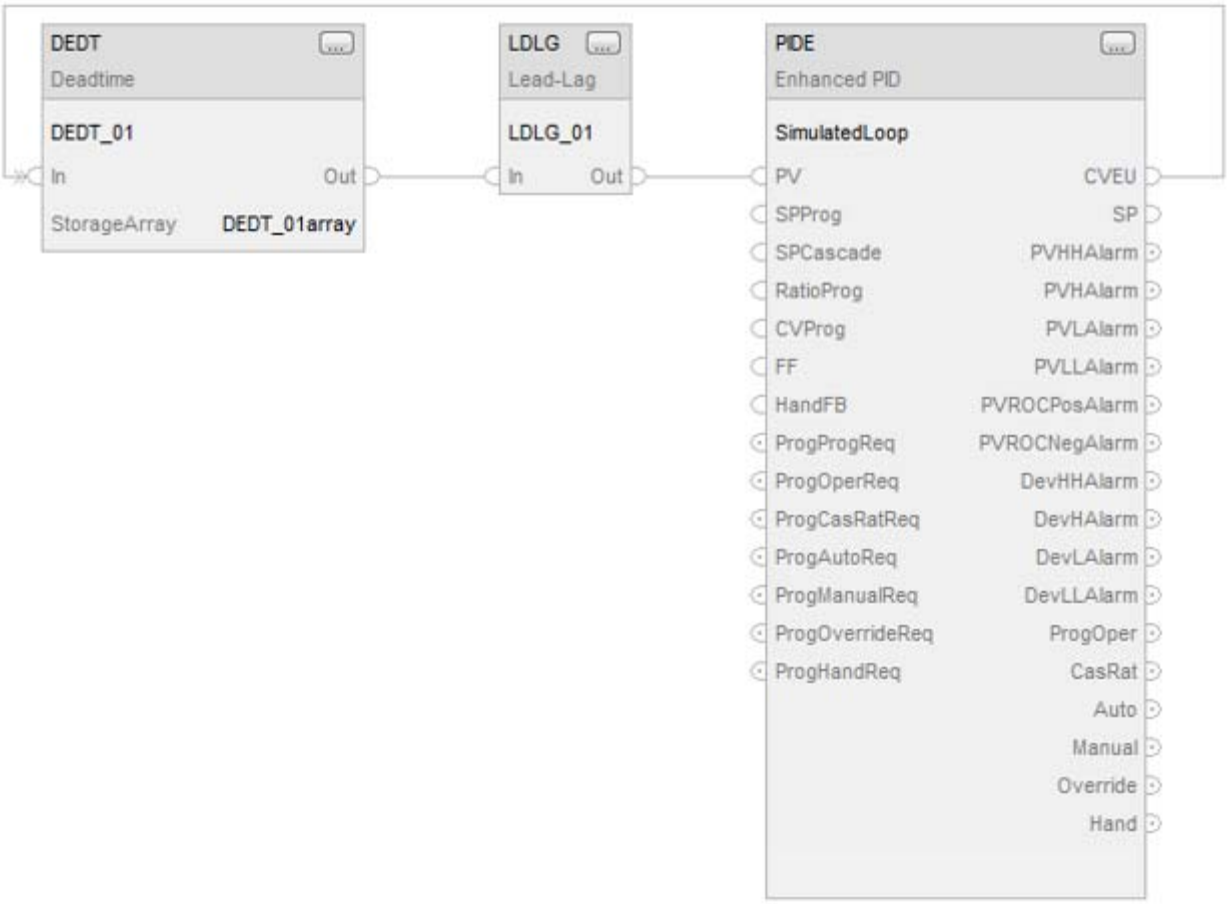
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

本例中，DEDT 指令将对所模拟过程中的死区时间延时进行模拟。为模拟此过程，PIDE 指令的输出将经过死区时间延时和一阶滞后。数组 DEDT_01array 是一个包含 100 个元素的 REAL 数组，最多可支持 100 个采样值的死区时间。例如，如果此例程每 100 ms 执行一次，则该数组最多可支持 10 s 的死区时间。

功能块



另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

函数生成器 (FGEN)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

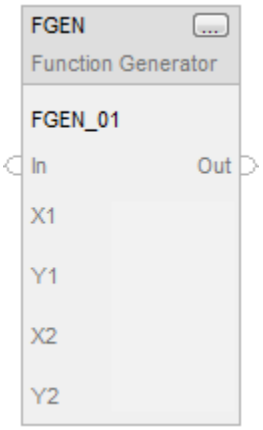
FGEN 指令用于基于分段线性函数对输入进行转换。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

FGEN(FGEN_tag,X1,Y1,X2,Y2);

操作数

功能块

操作数	类型	格式	说明
FGEN tag	FUNCTION_ GENERATOR	结构	FGEN 结构
X1	REAL	数组	X 轴数组，表 1。同 Y 轴数组结合使用，表 1 定义第一条分段线性曲线的点。 有效值 = 任意浮点值
Y1	REAL	数组	Y 轴数组，表 1。同 X 轴数组结合使用，表 1 定义第一条分段线性曲线的点。 有效值 = 任意浮点值
X2	REAL	数组	（可选） X 轴数组，表 2。同 Y 轴数组结合使用，表 2 定义第二条分段线性曲线的点。 有效值 = 任意浮点值

Y2	REAL	数组	（可选） Y 轴数组，表 2。同 X 轴数组结合使用，表 2 定义第二条分段线性曲线的点。 有效值 = 任意浮点值

结构化文本

操作数	类型	格式	说明
FGEN tag	FUNCTION_ GENERATOR	结构	FGEN 结构
X1	REAL	数组	X 轴数组，表 1。同 Y 轴数组结合使用，表 1 定义第一条分段线性曲线的点。 有效值 = 任意浮点值
Y1	REAL	数组	Y 轴数组，表 1。同 X 轴数组结合使用，表 1 定义第一条分段线性曲线的点。 有效值 = 任意浮点值
X2	REAL	数组	（可选） X 轴数组，表 2。同 Y 轴数组结合使用，表 2 定义第二条分段线性曲线的点。 有效值 = 任意浮点值
Y2	REAL	数组	（可选） Y 轴数组，表 2。同 X 轴数组结合使用，表 2 定义第二条分段线性曲线的点。 有效值 = 任意浮点值

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

FUNCTION_GENERATOR 结构

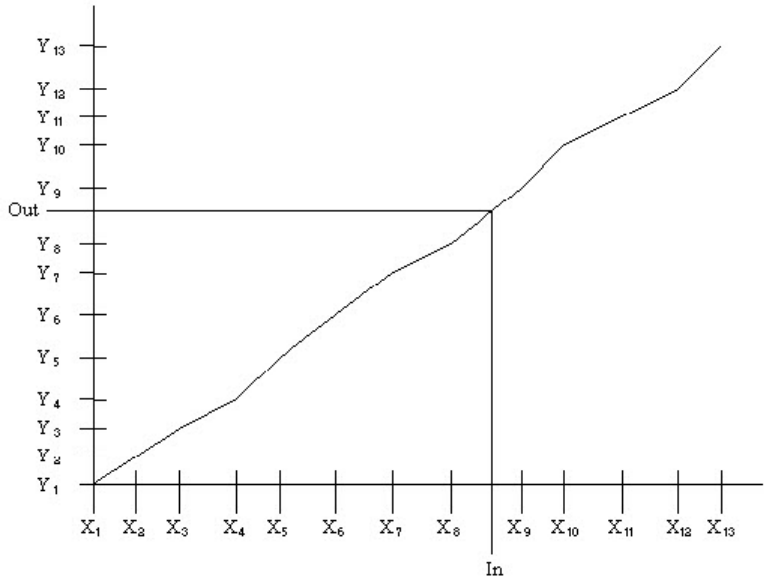
输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 =0.0

XY1Size	DINT	要从表 1 中使用的分段线性曲线上的点的数量。如果该值小于 1 且 Select 已清零，此指令会将 Status 中的相应位置位且输出不发生更改。 有效值 = 1 到（X1 和 Y1 数组大小中的较小者） 默认值 = 1
XY2Size	DINT	要从表 2 中使用的分段线性曲线上的点的数量。如果该值小于 0 且 Select 已置位，此指令会将 Status 中的相应位置位且输出不发生更改。 有效值 = 0 到（X2 和 Y2 数组大小中的较小者） 默认值 = 0
Select	BOOL	此输入确定使用哪个表。清零时，指令会使用表 1；置位时，指令会使用表 2。 默认值为清零。

输出参数	**数据类型**	**说明**
EnableOut	BOOL	指示指令是否处于启用状态。溢出时设置为假
Out	REAL	指令的输出。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	指令生成了故障。
XY1SizeInv (Status.1)	BOOL	表 1 的大小无效或与数组大小不兼容。
XY2SizeInv (Status.2)	BOOL	表 2 的大小无效或与数组大小不兼容。
XisOutofOrder (Status.3)	BOOL	X 参数未进行排序。

说明

下图说明 FGEN 指令转换一条十二段曲线的方式。



X 轴参数必须遵循以下关系：

X[1] < X[2] < X[3] < ...< X[XY<n>Size],

其中 XY<n>Size > 1，表示分段线性曲线上的点数，其中 n 为 1 或 2，具体取决于所选的表。必须在 X 数组中创建经过排序的 X 轴元素。

Select 输入确定指令使用哪个表。当指令基于其中一个表执行时，可对另一个表中的值进行修改。更改 Select 的状态后，可基于另一个表执行指令。

计算 Out 之前，将扫描 X 轴参数。如果这些参数未按升序排序，会将 Status 中的相应位置位，Out 保持不变。同样，如果 XY1Size 或 XY2Size 无效，此指令也会将 Status 中的相应位置位且 Out 保持不变。

该指令会使用以下算法根据 In 来计算 Out：

- 当 In ≤ X[1] 时，设置 Out = Y[1]
- 当 In > X[XY<n>Size] 时，设置 Out = Y[XY<n>Size]
- 当 X[n] < In ≤ X[n+1] 时，计算 Out = ((Y[n+1]-Yn)/(X[n+1]-Xn))*(In-Xn)+Yn

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件	功能块操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。指令执行。
指令首次运行	不适用
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为真。指令执行。

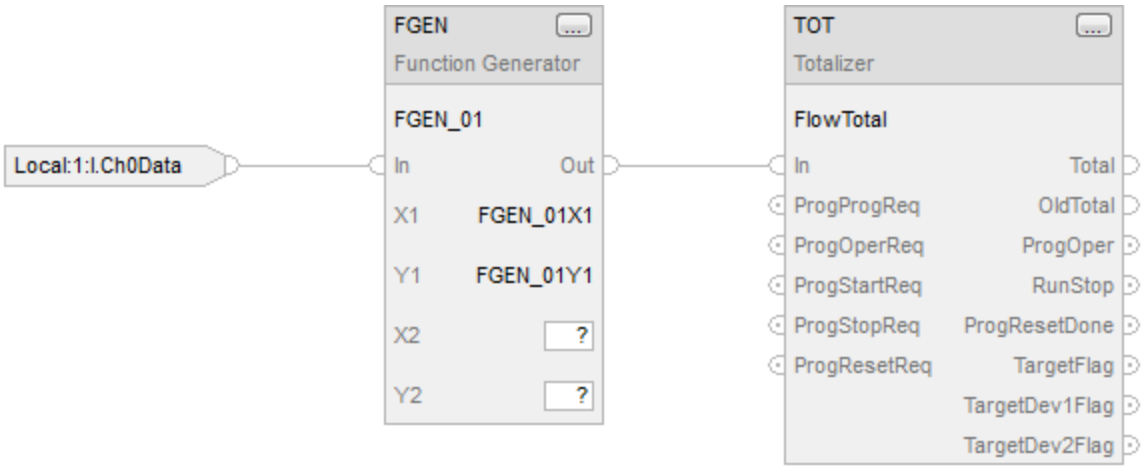
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

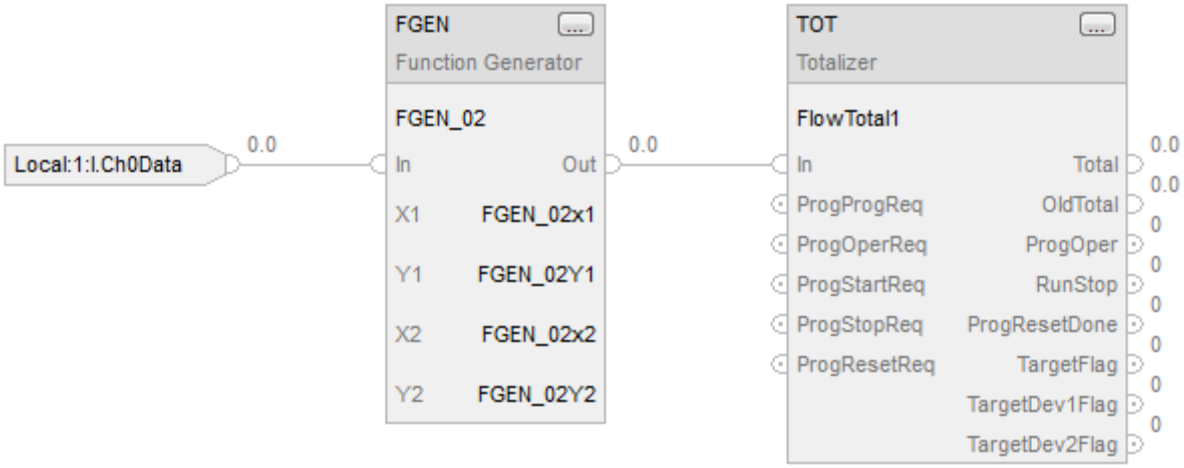
示例 1

在本例中，FGEN 指令采用流量信号形式，然后通过 TOT 指令累加流量信号。FGEN_01X1 和 FGEN_01Y1 数组均为具有 10 个元素的 REAL 数组，每个数组最多可支持 9 段曲线。可以使用任意大小的数组来支持需要任意段数的曲线。



示例 2

此示例将可选参数传递给 FGEN 指令。



另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

超前-滞后 (LDLG)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

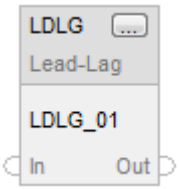
LDLG 指令用于为输入信号提供相位超前-滞后补偿。此指令通常用于前馈 PID 控制或过程仿真。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

LDLG(LDLG_tag);

操作数

功能块

操作数	类型	格式	说明
LDLG tag	LEAD_LAG	结构	LDLG 结构

结构化文本

操作数	类型	格式	说明
LDLG tag	LEAD_LAG	结构	LDLG 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

LEAD_LAG 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	初始化滤波控制算法的请求。当 Initialize 置位时，Out = (In x Gain) + Bias。 默认值 = 清零
Lead	REAL	超前时间，单位为秒。若设置 Lead = 0.0，会禁用超前控制算法。如果 Lead < 0.0，该指令会将 Status 中的相应位置位，并将 Lead 限制为 0.0。如果 Lead > 最大正浮点值，该指令会将 Status 中的相应位置位。 有效值 = 任意 ≥ 0.0 的浮点值 默认值 = 0.0
Lag	REAL	滞后时间，单位为秒。最短滞后时间为 DeltaT/2。如果 Lag < DeltaT/2，该指令会将 Status 中的相应位置位，并将 Lag 限制为 DeltaT/2。如果 Lag > 最大正浮点值，该指令会将 Status 中的相应位置位。 有效值 = 任意 ≥ DeltaT/2 的浮点值 默认值 = 0.0
Gain	REAL	过程增益乘数。该值用于模拟过程增益。In 信号将乘以该值。I = (In x Gain) + Bias 有效值 = 任意浮点值 默认值 = 1.0
Bias	REAL	过程偏移程度。该值用于模拟环境条件。该值将与 In 和 Gain 的乘积相加。I = (In x Gain) + Bias 有效值 = 任意浮点值 默认值 = 0.0

TimingMode	DINT	选择时序执行模式。 0 = 周期速率 1 = 过采样模式 2 = 实时采样模式 有效值 = 0 至 2 默认值 = 0 有关时序模式的更多信息，请参见"功能块属性"部分。
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	启用输出。
Out	REAL	计算所得的算法输出。数学状态标志用于此输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
LeadInv (Status.1)	BOOL	Lead < 最小值或 Lead > 最大值。
LagInv (Status.2)	BOOL	Lag < 最小值或 Lag > 最大值。
TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS|DeltaT - RTSTime| > 1（0.001 秒）时置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。

RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

说明

LDLG 指令支持一个超前和滞后串联。该指令还支持使用可组态的增益和偏置系数。LDLG 指令设计为在扫描速率保持恒定的任务中执行。

LDLG 指令采用以下公式：

$$H(s) = \left[\frac{1 + Lead \times s}{1 + Lag \times s}\right]$$

其中，参数限制如下：

参数	限制
Lead	LowLimit = 0.0 HighLimit = 最大正浮点值
Lag	LowLimit = DeltaT/2（DeltaT 以秒为单位） HighLimit = 最大正浮点值

一旦计算出的输出值无效（NAN 或 ± INF），指令会设置 Out = 无效值，并将数学溢出状态标志置位。当计算出的输出值有效后，该指令将初始化内部参数并设置 Out = (In x Gain) + Bias。

数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件	功能块操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。

指令首次运行	不适用
指令首次扫描	执行“Out = (In * Gain) + Bias”。 重新计算 Lead/Lag 系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

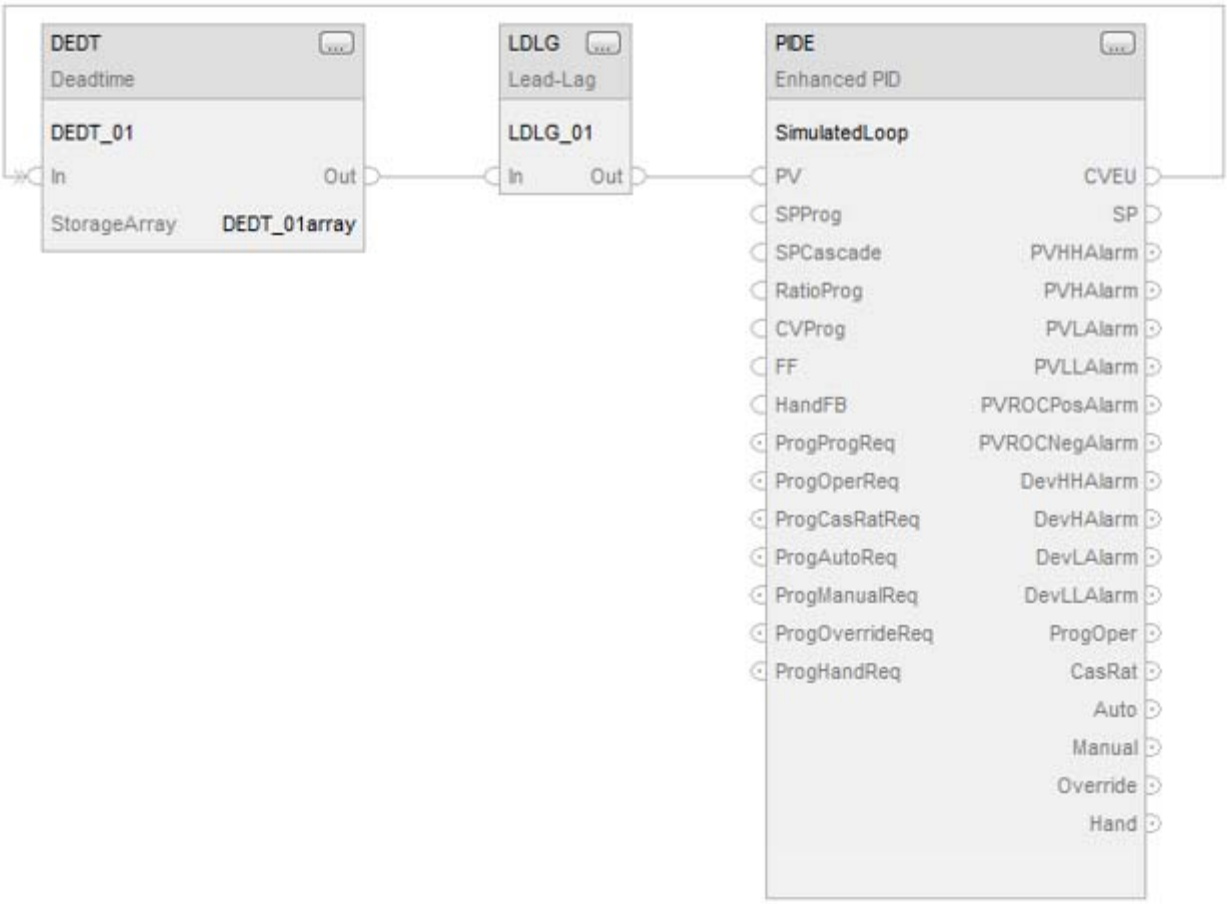
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

本例中，LDLG 指令将向模拟过程添加一阶滞后。用户也可以在 LDLG 指令中输入一个 Gain 来模拟过程增益，还可以输入一个 Bias 来模拟环境条件。

功能块



结构化文本

```
DEDT_01.In := SimulatedLoop.CVEU;

DEDT(DEDT_01,DEDT_01_array);

LDLG_01.In := DEDT_01.Out;

LDLG(LDLG_01);

SimulatedLoop.PV := LDLG_01.Out;

PIDE(SimulatedLoop);
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

增强型 PID (PIDE)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

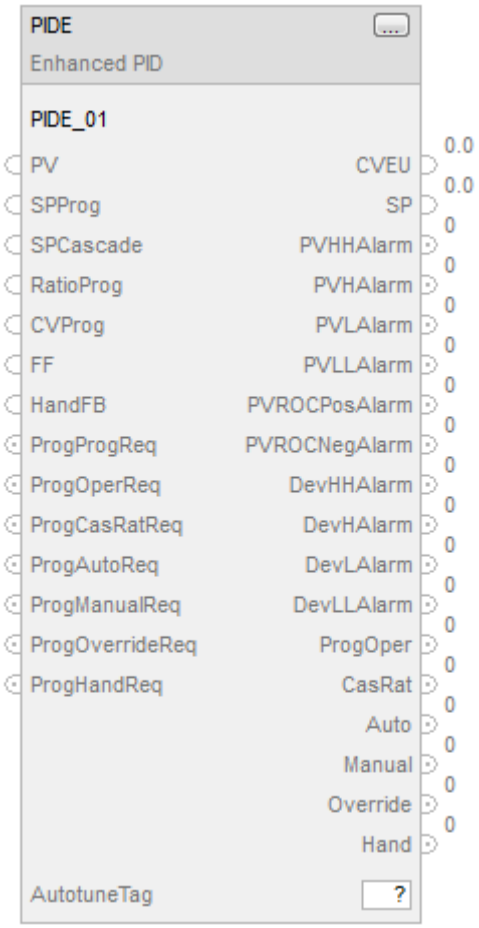
PIDE 指令在标准的 PID 指令基础上提供增强功能。该指令采用 PID 算法的速度形式。增益项应用于误差或 PV 值的变化，而不是误差或 PV 值。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

PIDE(PIDE_tag);

操作数

功能块

操作数	类型	格式	说明
PIDE tag	PID_ENHANCED	结构	PIDE 结构
autotune tag	PIDE_AUTOTUNE	结构	（可选） 自动调谐结构

结构化文本

操作数	类型	格式	说明
PIDE tag	PID_ENHANCED	结构	PIDE 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

PIDE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
PV	REAL	经过标定的过程变量输入。该值通常从模拟量输入模块读取。 有效值 = 任意浮点值 默认值 = 0.0
PVFault	BOOL	PV 不良状况指示器。如果 PV 从模拟量输入中读取，那么 PVFault 通常由模拟量输入故障状态控制。当 PVFault 为真时，此参数指示输入信号存在错误。 默认值为假 =“状况良好”
PVEUMax	REAL	PV 的最大标定值。对应于过程变量 100% 量程的 PV 和 SP 值。 有效值 = PVEUMin < PVEUMax ≤ 最大正浮点值 默认值 = 100.0
PVEUMin	REAL	PV 的最小标定值。对应于过程变量 0% 量程的 PV 和 SP 值。 有效值 = 最大负浮点值 ≤ PVEUMin < PVEUMax 默认值 = 0.0
SPProg	REAL	SP 程序值，以 PV 单位标定。SP 在处于程序控制模式且非级联/比率模式下时设置为此值。如果 SPProg 的值 < SPLLimit 或 > SPHLimit，该指令会将 Status 中的相应位置位，并限制 SP 的值。 有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPOper	REAL	SP 操作员值，以 PV 单位标定。SP 在处于操作员控制模式且非级联/比率模式下时设置为此值。如果 SPOper 的值 < SPLLimit 或 > SPHLimit，该指令会将 Status 中的相应位置位，并限制 SP 的值。 有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0

SPCascade	REAL	SP 级联值，以 PV 单位标定。如果 CascadeRatio 为真且 UseRatio 为假，则 SP = SPCascade。这通常为主回路的 CVEU。如果 CascadeRatio 和 UseRatio 均为真，则 SP = (SPCascade x Ratio)。如果 SPCascade 的值 < SPLLimit 或 > SPHLimit，则会将 Status 中的相应位置位，并限制 SP 的值。 有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPHLimit	REAL	SP 上限值，以 PV 单位标定。如果 SPHLimit > PVEUMax，该指令会将 Status 中的相应位置位。 有效值 = SPLLimit 至 PVEUMax 默认值 = 100.0
SPLLimit	REAL	SP 下限值，以 PV 单位标定。如果 SPLLimit < PVEUMin，该指令会将 Status 中的相应位置位。如果 SPHLimit < SPLLimit，该指令会将 Status 中的相应位置位，并使用 SPLLimit 的值限制 SP。 有效值 = PVEUMin 到 SPHLimit 默认值 = 0.0
UseRatio	BOOL	允许比率控制。若设置为真，可在级联/比率模式下启用比率控制。默认值为假。
RatioProg	REAL	程序控制模式下的比率乘数。Ratio 和 RatioOper 在程序控制模式下设置为等于此值。如果 RatioProg < RatioLLimit 或 > RatioHLimit，该指令会将 Status 中的相应位置位，并限制 Ratio 的值。 有效值 = RatioLLimit 至 RatioHLimit 默认值 = 1.0
RatioOper	REAL	操作员控制下的比率乘数。Ratio 在操作员控制模式下设置为等于此值。如果 RatioOper < RatioLLimit 或 > RatioHLimit，该指令会将 Status 中的相应位置位，并限制 Ratio 的值。 有效值 = RatioLLimit 至 RatioHLimit 默认值 = 1.0
RatioHLimit	REAL	比率上限值。限制从 RatioProg 或 RatioOper 获取的 Ratio 值。如果 RatioHLimit < RatioLLimit，该指令会将 Status 中的相应位置位，并使用 RatioLLimit 的值限制 Ratio。 有效值 = RatioLLimit 到最大正浮点值 默认值 = 1.0
RatioLLimit	REAL	比率下限值。限制从 RatioProg 或 RatioOper 获取的 Ratio 值。如果 RatioLLimit < 0，该指令会将 Status 中的相应位置位，并将值限制为零。如果 RatioHLimit < RatioLLimit，该指令会将 Status 中的相应位置位，并使用 RatioLLimit 的值限制 Ratio。 有效值 = 0.0 到 RatioHLimit 默认值 = 1.0

CVFault	BOOL	控制变量不良状况指示器。如果 CVEU 控制模拟量输出，则 CVFault 通常来自于模拟量输出的故障状态。当 CVFault 为真时，说明输出模块存在错误，该指令会将 Status 中的相应位置位。 默认值为假 ="状况良好"
CVInitReq	BOOL	CV 初始化请求。此信号通常由受 CVEU 控制的模拟量输出模块上的"In Hold"状态控制，或者来自次级 PID 回路的 InitPrimary 输出。 默认值为假。
CVInitValue	REAL	CVEU 初始化值，以 CVEU 单位标定。当 CVInitializing 为真时，CVEU = CVInitValue 且 CV 等于相应的百分比值。CVInitValue 来自于受 CVEU 控制的模拟量输出的反馈，或者次级回路的设置点。当 CVFaulted 或 CVEUSpanInv 为真时，会禁用指令初始化。 有效值 = 任意浮点值 默认值 = 0.0
CVProg	REAL	CV 程序手动值。处于程序手动模式时，CV 将等于该值。如果 CVProg < 0 或 > 100，或者在 CVManLimiting 为真时 < CVLLimit 或 > CVHLimit，该指令会将 Status 中的相应位置位，并限制 CV 值。 有效值 = 0.0 到 100.0 默认值 = 0.0
CVOper	REAL	CV 操作员手动值。处于操作员手动模式时，CV 将等于该值。如果未处于操作员手动模式，该指令会在每条指令执行结束时设置 CVOper = CV。如果 CVOper < 0 或 > 100，或者在 CVManLimiting 为真时 < CVLLimit 或 > CVHLimit ,该指令会将 Status 中的相应位置位 ,并限制 CV 值。 有效值 = 0.0 到 100.0 默认值 = 0.0
CVOverride	REAL	超控模式下的 CV 值。处于超控模式时，CV 等于此值。此值应对应于 PID 回路的某个安全状态输出。如果 CVOverride < 0 或 > 100，该指令会将 Status 中的相应位置位，并限制 CV 值。 有效值 = 0.0 到 100.0 默认值 = 0.0
CVPrevious	REAL	CV_{n-1} 值。如果 CVSetPrevious 置位，CV_{n-1} 等于此值。CV_{n-1} 是上次执行指令后的 CV 值。处于手动、超控或手控模式时 ,或者 CVInitializing 置位时，都会忽略 CVPrevious。如果 CVPrevious < 0 或 > 100，或者在处于自动或级联/比率模式时 < CVLLimit 或 > CVHLimit，该指令会将 Status 中的相应位置位，并限制 CVn-1 值。 有效值 = 0.0 到 100.0 默认值 = 0.0
CVSetPrevious	BOOL	使用 CVPrevious 的请求。如果为真，CVn-1 = CVPrevious 默认值为假。

CVManLimiting	BOOL	在手动模式下限制 CV 的请求。如果处于手动模式并且 CVManLimiting 为真，则 CV 由 CVHLimit 和 CVLLimit 值限制。 默认值为假。
CVEUMax	REAL	CVEU 的最大值。对应于 100% CV 的 CVEU 的值。如果 CVEUMax = CVEUMin，该指令会将 Status 中的相应位置位。 有效值 = 任意浮点值 默认值 = 100.0
CVEUMin	REAL	CVEU 的最小值。对应于 0% CV 的 CVEU 的值。如果 CVEUMax = CVEUMin，该指令会将 Status 中的相应位置位。 有效值 = 任意浮点值 默认值 = 0.0
CVHLimit	REAL	CV 上限值。该值用于设置 CVHAlarm 输出。处于自动或级联/比率模式时，或者处于手动模式并且 CVManLimiting 为真时，该值还用于对 CV 进行限制。如果 CVHLimit > 100 或 < CVLLimit，该指令会将 Status 中的相应位置位。如果 CVHLimit < CVLLimit，该指令会使用 CVLLimit 的值限制 CV。 有效值 = CVLLimit < CVHLimit ≤ 100.0 默认值 = 100.0
CVLLimit	REAL	CV 下限值。该值用于设置 CVLAlarm 输出。处于自动或级联/比率模式时，或者处于手动模式并且 CVManLimiting 为真时，该值还用于对 CV 进行限制。如果 CVLLimit < 0 或 CVHLimit < CVLLimit ,该指令会将 Status 中的相应位置位。如果 CVHLimit < CVLLimit，该指令会使用 CVLLimit 的值限制 CV。 有效值 = 0.0 ≤ CVLLimit < CVHLimit 默认值 = 0.0
CVROCLimit	REAL	CV 变化率限值 ,以百分比/秒表示。仅当处于自动或级联/比率模式，或者处于手动模式并且 CVManLimiting 为真时，才会使用变化率限值。若输入 0，会禁用 CV ROC 限制。如果 CVROCLimit < 0，该指令会将 Status 中的相应位置位，并禁用 CV ROC 限制。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
FF	REAL	前馈值。向 CV 应用过零死区限制后，前馈的值将与 CV 相加。因此，FF 的变化将始终反映在 CV 的最终输出值上。如果 FF < -100 或 > 100，该指令会将 Status 中的相应位置位，并限制 FF 的值。 有效值 = -100.0 到 100.0 默认值 = 0.0

FFPrevious	REAL	FF_{n-1} 值。如果 FF SetPrevous 置位，该指令会设置 FF_{n-1} = FFPrevious。FF_{n-1} 是上次执行指令后的 FF 值。如果 FFPrevious < -100 或 > 100，该指令会将 Status 中的相应位置位，并限制 FF_{n-1} 的值 有效值 = -100.0 到 100.0 默认值 - 0.0
FFSetPrevious	BOOL	使用 FFPrevious 的请求。如果为真，FFn-1 = FFPrevious。 默认值为假。
HandFB	REAL	CV 手控反馈值。处于手控模式且 HandFBFault 为假（状况良好）时，CV 等于该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。如果 HandFB < 0 或 > 100，该指令会将 Status 中的相应位置位，并限制 CV 的值。 有效值 = 0.0 到 100.0 默认值 = 0.0
HandFBFault	BOOL	HandFB 值不良状况指示器。如果 HandFB 值从模拟量输入读取，则 HandFBFault 通常由模拟量输入通道的状态控制。当 HandFBFault 为真时，说明输入模块存在错误，该指令会将 Status 中的相应位置位。 默认值为假 ="状况良好"
WindupHIn	BOOL	积分饱和上限请求。为真时，CV 无法沿正向积分。该信号通常从次级回路的 WindupHOut 输出获取。 默认值为假。
WindupLIn	BOOL	积分饱和下限请求。为真时，CV 无法沿负向积分。该信号通常从次级回路的 WindupLOut 输出获取。 默认值为假。
ControlAction	BOOL	控制操作请求。设置为真时，可按 E = PV - SP 计算误差；设置为假时，可按 E = SP - PV 计算误差。 默认值为假。
DependIndepend	BOOL	相关/独立控制请求。为真时，使用 PID 等式的相关形式；为假时，使用等时的独立形式。 默认值为假。
PGain	REAL	比例增益。选择 PID 算法的独立形式时，向该值输入无单位的比例增益。当选择 PID 算法的相关形式时，向该值输入无单位的控制器增益。若输入 0，会禁用比例控制。如果 PGain < 0，该指令会将 Status 中的相应位置位，并使用 PGain = 0 的值。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0

IGain	REAL	积分增益。选择 PID 算法的独立形式时，向该值输入单位为“1/分钟”的积分增益。选择 PID 算法的相关形式时，向该值输入单位为“分钟/循环”的积分时间常数。若输入 0，会禁用积分控制。如果 IGain < 0，该指令会将 Status 中的相应位置位，并使用 IGain = 0 的值。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
DGain	REAL	微分增益。选择 PID 算法的独立形式时，向该值输入单位为分钟的微分增益。使用 PID 算法的相关形式时，向该值输入单位为分钟的微分时间常数。若输入 0，会禁用微分控制。如果 DGain < 0，该指令会将 Status 中的相应位置位，并使用 DGain = 0 的值。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
PVEProportional	BOOL	PV 比例控制请求。为真时，使用过程变量的变化量 (PVPercent) 计算比例项 (DeltaPTerm)。为假时，使用误差的变化量 (EPercent)。 默认值为假。
PVEDerivative	BOOL	PV 微分控制请求。为真时，使用过程变量的变化量 (PVPercent) 计算微分项 (DeltaDTerm)。为假时，使用误差的变化量 (EPercent)。 默认值为真。
DSmoothing	BOOL	微分平滑请求。为真时，平滑微分项的变化。若平滑微分项，会减少噪声 PV 信号引起的输出“抖动”，但也会限制高微分增益的作用。 默认值为假。
PVTracking	BOOL	SP 跟踪 PV 请求。若处于手动模式时设置为真，会使 SP 跟踪 PV。处于级联/比率或自动模式时，会忽略此设置。 默认值为假。
ZCDeadband	REAL	过零死区范围，以 PV 单位标定。定义过零死区范围。若输入 0，会禁用过零死区检查。如果 ZCDeadband < 0，该指令会将 Status 中的相应位置位，并禁用过零死区检查。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
ZCOff	BOOL	过零禁用请求。若设置为真，会针对死区计算禁用过零检查。 默认值为假。
PVHHLimit	REAL	PV 上上限报警限值，以 PV 单位标定。 有效值 = 任意浮点值 默认值 = 最大正浮点值
PVHLimit	REAL	PV 上限报警限值，以 PV 单位标定。 有效值 = 任意浮点值 默认值 = 最大正浮点值

PVLLimit	REAL	PV 下限报警限值，以 PV 单位标定。 有效值 = 任意浮点值 默认值 = 最大负浮点值
PVLLLimit	REAL	PV 下下限报警限值，以 PV 单位标定。 有效值 = 任意浮点值 默认值 = 最大负浮点值
PVDeadband	REAL	PV 报警限值死区值，以 PV 单位标定。死区是每个 PV 报警限制的开启和关闭值之间的差值。如果 PVDeadband < 0.0，该指令会将 Status 中的相应位置位，并将 PVDeadband 限制为零。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
PVROCPosLimit	REAL	PV 正变化率报警限值。PV 正变化（增大）限值，以 PV 单位/秒标定。若输入 0.0，会禁用正 PVROC 报警检查。如果 PVROCPosLimit < 0.0，该指令会将 Status 中的相应位置位，并禁用正 PVROC 检查。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0 PV/s
PVROCNegLimit	REAL	PV 负变化率报警限值。PV 负变化（减小）限值，以 PV 单位/秒标定。若输入 0.0，会禁用负 PVROC 报警检查。如果 PVROCNegLimit < 0，该指令会将 Status 中的相应位置位，并禁用负 PVROC 检查。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
PVROCPeriod	REAL	PV 变化率采样周期。计算 PV 变化率时所对应的时间段，以秒为单位。输入 0 会禁用 PVROC 报警检查。如果 PVROCPeriod < 0.0，该指令会将 Status 中的相应位置位，并禁用正负 PVROC 检查。 有效值 = 任意 ≥ 0.0 的浮点值 默认值 = 0.0 秒
DevHHLimit	REAL	偏差上上限报警限值，以 PV 单位标定。偏差指过程变量 (PV) 值与设置点 (SP) 值之差。偏差报警提示操作员过程变量和设置点值存在差异。如果 DevHHLimit < 0.0，该指令会将 Status 中的相应位置位，并设置 DevHHLimit = 0.0。 有效值 = 0.0 到最大正浮点值 默认值 = 最大正浮点值
DevHLimit	REAL	偏差上限报警限值，以 PV 单位标定。偏差指过程变量 (PV) 值与设置点 (SP) 值之差。偏差报警提示操作员过程变量和设置点值存在差异。如果 DevHLimit < 0.0，该指令会将 Status 中的相应位置位，并设置 DevHLimit = 0.0。 有效值 = 0.0 到最大正浮点值 默认值 = 最大正浮点值

DevLLimit	REAL	偏差下限报警限值，以 PV 单位标定。偏差指过程变量 (PV) 值与设置点 (SP) 值之差。偏差报警提示操作员过程变量和设置点值存在差异。如果 DevLLimit < 0.0，该指令会将 Status 中的相应位置位，并设置 DevLLimit = 0.0。 有效值 = 0.0 到最大正浮点值 默认值 = 最大正浮点值
DevLLLimit	REAL	偏差下下限报警限值，以 PV 单位标定。偏差指过程变量 (PV) 值与设置点 (SP) 值之差。偏差报警提示操作员过程变量和设置点值存在差异。如果 DevLLLimit < 0.0，该指令会将 Status 中的相应位置位，并设置 DevLLLimit = 0.0。 有效值 = 0.0 到最大正浮点值 默认值 = 最大正浮点值
DevDeadband	REAL	偏差报警限值的死区值，以 PV 单位标定。死区是每个偏差报警限制的开启和关闭值之间的差值。如果 DevDeadband < 0.0，该指令会将 Status 中的相应位置位，并设置 DevDeadband = 0.0。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
AllowCasRat	BOOL	允许级联/比率模式。设置为真时，支持使用 ProgCasRatReq 或 OperCasRatReq 选择级联/比率模式。 默认值为假。
ManualAfterInit	BOOL	初始化后进入手动模式的请求。当此参数为真并且 CVInitializing 为真时，除非当前模式为“超控”或“手控”，否则会将该指令置于手动模式。当 ManualAfterInit 为假时，除非请求更改该指令的模式，否则不会更改该指令的模式。 默认值为假。
ProgProgReq	BOOL	程序发出的程序控制请求。由用户程序设置为真可请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。若此参数保持为真，并且 ProgOperReq 保持为假，会将指令锁定在程序控制模式。当 ProgValueReset 为真时 ,该指令在每次执行时都会将此输入设置为假。 默认值为假。
ProgOperReq	BOOL	程序发出的操作员控制请求。由用户程序设置为真可请求操作员控制模式。若将此参数保持为真，会将该指令锁定在操作员控制模式。当 ProgValueReset 为真时，该指令在每次执行时都会将此输入设置为假。 默认值为假。
ProgCasRatReq	BOOL	程序级联/比率模式请求。由用户程序设置为真可请求级联/比率模式。当 ProgValueReset 为真时，该指令在每次执行时都会将此输入设置为假。 默认值为假。

ProgAutoReq	BOOL	程序自动模式请求。由用户程序设置为真可请求自动模式。当 ProgValueReset 为真时 ,该指令在每次执行时都会将此输入设置为假。 默认值为假。
ProgManualReq	BOOL	程序手动模式请求。由用户程序设置为真可请求手动模式。如果 ProgValueReset 为真，指令会在每次执行时将该输入设置为假。 默认值为假。
ProgOverrideReq	BOOL	程序超控模式请求。由用户程序设置为真可请求超控模式。当 ProgValueReset 为真时 ,该指令在每次执行时都会将此输入设置为假。 默认值为假。
ProgHandReq	BOOL	程序手控模式请求。由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。当 ProgValueReset 为真时，该指令在每次执行时都会将此输入设置为假。 默认值为假。
OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。该指令在每次执行时都会将此输入设置为假。 默认值为假。
OperOperReq	BOOL	操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。该指令在每次执行时都会将此输入设置为假。 默认值为假。
OperCasRatReq	BOOL	操作员级联/比率模式请求。由操作员界面设置为真可请求级联/比率模式。该指令在每次执行时都会将此输入设置为假。 默认值为假。
OperAutoReq	BOOL	操作员自动模式请求。由操作员界面设置为真可请求自动模式。该指令在每次执行时都会将此输入设置为假。 默认值为假。
OperManualReq	BOOL	操作员手动模式请求。由操作员界面设置为真可请求手动模式。该指令在每次执行时都会将此输入设置为假。 默认值为假。
ProgValueReset	BOOL	将程序控制值复位。此参数为真时，该指令每次执行时都会将所有程序请求输入设置为假。此参数为真且处于操作员控制模式时，该指令会设置 SPProgram = SP 以及 CVProgram = CV。 默认值为假。
TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0

OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0
AtuneAcquire	BOOL	获取 PIDE 自动调谐数据的请求。 默认值为假。
AtuneStart	BOOL	启动自动调谐的请求。 默认值为假。
AtuneUseGains	BOOL	使用自动调谐增益的请求。 默认值为假。
AtuneAbort	BOOL	中止自动调谐的请求。 默认值为假。
AtuneUnacquire	BOOL	不获取 PIDE 自动调谐数据的请求。 默认值为假。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 CVEU 溢出，则设置为假。
CVEU	REAL	标定控制变量输出。使用 CVEUMax 和 CVEUMin 标定，其中 CVEUMax 对应于 100%，CVEUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CVEU = (CV x CVEUSpan / 100) + CVEUMin CVEU 量程计算如下：CVEUSpan = (CVEUMax - CVEUMin)
CV	REAL	控制变量输出。此值以 0% 到 100% 表示。当处于自动或级联/比率模式时，或者处于手动模式并且 CVManLimiting 设置为真时，CV 由 CVHLimit 和 CVLLimit 进行限制。否则，此值限制为 0% 到 100%。
CVInitializing	BOOL	初始化模式指示器。当 CVInitReq 设置为真时，在指令首次扫描期间，以及 CVHealth 由真跳变为假时（不良到良好），CVInitializing 会设置为真。在指令完成初始化以及 CVInitReq 为假时，CVInitializing 设置为假。

CVHAlarm	BOOL	CV 上限报警指示器。当 CV 的 计算值 > 100 或 CVHLimit 时，会设置为真。
CVLAlarm	BOOL	CV 下限报警指示器。当 CV 的 计算值 < 0 或 CVLLimit 时，会设置为真。
CVROCAlarm	BOOL	CV 变化率报警指示器。当计算的 CV 变化率超过 CVROCLimit 时，会设置为真。
SP	REAL	当前设置点的值。SP 值在处于级联/比率模式时用于控制 CV。
SPPercent	REAL	以 PV 量程的百分比形式表示的 SP 值。 SPPercent = ((SP - PVEUMin) x 100) / PVSpan PV 量程计算如下：PVSpan = (PVEUMax - PVEUMin)
SPHAlarm	BOOL	SP 上限报警指示器。 当 SP > SPHLimit 时设置为真。
SPLAlarm	BOOL	SP 下限报警指示器。 当 SP < SPLLimit 时设置为真。
PVPercent	REAL	以量程百分比的形式表示的 PV 值。 PVPercent = ((PV- PVEUMin) x 100) / PVSpan PV 量程计算如下：PVSpan = (PVEUMax - PVEUMin)
E	REAL	过程误差。SP 与 PV 之间的差值，以 PV 单位标定。
EPercent	REAL	以量程百分比形式表示的误差。
InitPrimary	BOOL	初始化主回路命令。当未处于级联/比率模式或者 CVInitializing 为真时设置为真。此信号通常由主 PID 回路的 CVInitReq 输入使用。
WindupHOut	BOOL	积分饱和上限指示器。当达到 CV 上下限（取决于控制操作）或者 SP 上限时设置为真。此信号通常由 WindupHIn 输入使用，用于防止主回路上 CV 输出积分饱和。
WindupLOut	BOOL	积分饱和下限指示器。当达到 CV 上下限（取决于控制操作）或者 SP 下限时设置为真。此信号通常由 WindupLIn 输入使用，用于防止主回路上 CV 输出积分饱和。
Ratio	REAL	当前比率乘数。
RatioHAlarm	BOOL	比率上限报警指示器。当 Ratio > RatioHLimit 时设置为真。
RatioLAlarm	BOOL	比率下限报警指示器。当 Ratio < RatioLLimit 时设置为真。
ZCDeadbandOn	BOOL	过零死区指示器。该值为真时，CV 值保持不变。如果 ZCOff 为真，当 |E| 处于 ZCDeadband 范围内时，ZCDeadbandOn 设置为真。如果 ZCOff 为假，当 |E| 过零且处于 ZCDeadband 范围内时，ZCDeadbandOn 设置为真。当 |E| 超出死区范围或者 ZCDeadband = 0 时，ZCDeadbandOn 设置为假。
PVHHAlarm	BOOL	PV 上上限报警指示器。当 PV ≥ PVHHLimit 时设置为真。当 PV < (PVHHLimit - PVDeadband) 时设置为假

PVHAlarm	BOOL	PV 上限报警指示器。当 PV ≥ PVHLimit 时设置为真。当 PV < (PVHLimit - PVDeadband) 时设置为假
PVLAlarm	BOOL	PV 下限报警指示器。当 PV ≤ PVLLimit 时设置为真。当 PV > (PVLLimit + PVDeadband) 时设置为假
PVLLAlarm	BOOL	PV 下下限报警指示器。当 PV ≤ PVLLLimit 时设置为真。当 PV > (PVLLLimit + PVDeadband) 时设置为假
PVROCPosAlarm	BOOL	PV 正变化率报警指示器。当计算的 PV 变化率 ≥ PVROCPosLimit 时设置为真。
PVROCNegAlarm	BOOL	PV 负变化率报警指示器。当计算的 PV 变化率 ≤ (PVROCNegLimit x -1) 时设置为真。
DevHHAlarm	BOOL	偏差上上限报警指示器。当 PV ≥ (SP + DevHHLimit) 时设置为真。当 PV < (SP + DevHHLimit - DevDeadband) 时设置为假
DevHAlarm	BOOL	偏差上限报警指示器。当 PV ≥ (SP + DevHLimit) 时设置为真。当 PV < (SP + DevHLimit - DevDeadband) 时设置为假
DevLAlarm	BOOL	偏差下限报警指示器。当 PV ≤ (SP - DevLLimit) 时设置为真。当 PV > (SP - DevLLimit + DevDeadband) 时设置为假
DevLLAlarm	BOOL	偏差下下限报警指示器。当 PV ≤ (SP - DevLLLimit) 时设置为真。当 PV > (SP - DevLLLimit + DevDeadband) 时设置为假
ProgOper	BOOL	程序/操作员控制指示器。在程序控制模式下设置为真。在操作员控制模式下设置为假。
CasRat	BOOL	级联/比率模式指示器。在级联/比率模式下设置为真。
Auto	BOOL	自动模式指示器。在自动模式下设置为真。
Manual	BOOL	手动模式指示器。在手动模式下设置为真。
超控 (Override)	BOOL	超控模式指示器。在超控模式下设置为真。
手控 (Hand)	BOOL	手控模式指示器。在手控模式下设置为真。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
AtuneReady	BOOL	当 PIDE 指令获取到指定的自动调谐数据时，设置为真。
AtuneOn	BOOL	当启动自动调谐后，设置为真。
AtuneDone	BOOL	当完成自动调谐后，设置为真。
AtuneAborted	BOOL	当自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。

AtuneBusy	BOOL	当由于指定的自动调谐数据正由其他 PIDE 指令获取而无法获取时，设置为真。
Status1	DINT	功能块的状态。
InstructFault (Status1.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
PVFaulted (Status1.1)	BOOL	过程变量 (PV) 状况不良。
CVFaulted (Status1.2)	BOOL	控制变量 (CV) 状况不良。
HandFBFaulted (Status1.3)	BOOL	HandFB 值状况不良。
PVSpanInv (Status1.4)	BOOL	PV 量程无效。PVEUMax £ PVEUMin。
SPProgInv (Status1.5)	BOOL	SPProg < SPLLimit 或 SPProg > SPHLimit。指令将限制 SP 的值。
SPOperInv (Status1.6)	BOOL	SPOper < SPLLimit 或 SPOper > SPHLimit。指令将限制 SP 的值。
SPCascadeInv (Status1.7)	BOOL	SPCascade < SPLLimit 或 SPCascade > SPHLimit。指令将限制 SP 的值。
SPLimitsInv (Status1.8)	BOOL	限值无效：SPLLimit < PVEUMin，SPHLimit > PVEUMax 或 SPHLimit < SPLLimit。如果 SPHLimit < SPLLimit，指令将使用 SPLLimit 限制该值
RatioProgInv (Status1.9)	BOOL	RatioProg < RatioLLimit 或 RatioProg > RatioHLimit。指令将限制 Ratio 的值。
RatioOperInv (Status1.10)	BOOL	RatioOper < RatioLLimit 或 RatioOper > RatioHLimit。指令将限制 Ratio 的值。
RatioLimitsInv (Status1.11)	BOOL	RatioLLimit < 0 或 RatioHLimit < RatioLLimit。
CVProgInv (Status1.12)	BOOL	CVProg < 0 或 CVProg > 100，或者当 CVManLimiting 为真时，CVProg < CVLLimit 或 CVProg > CVHLimit。指令将限制 CV 的值。
CVOperInv (Status1.13)	BOOL	CVOper < 0 或 CVOper > 100，或者当 CVManLimiting 为真时，CVOper < CVLLimit 或 CVOper > CVHLimit。指令将限制 CV 的值。
CVOverrideInv (Status1.14)	BOOL	CVOverride < 0 或 CVOverride > 100。指令将限制 CV 的值。
CVPreviousInv (Status1.15)	BOOL	CVPrevious < 0 或 CVPrevious > 100，或者在自动或级联/比率模式下，CVPrevious < CVLLimit 或 CVPrevious > CVHLimit。指令将限制 CVn-1 的值。
CVEUSpanInv (Status1.16)	BOOL	CVEU 量程无效。指令使用的 CVEUMax 值 = CVEUMin。
CVLimitsInv (Status1.17)	BOOL	CVLLimit < 0、CVHLimit > 100 或 CVHLimit < CVLLimit。如果 CVHLimit < CVLLimit，指令将使用 CVLLimit 限制 CV 值。
CVROCLimitInv (Status1.18)	BOOL	CVROCLimit < 0。指令禁用 ROC 限制。
FFInv (Status1.19)	BOOL	FF < -100 或 FF > 100。指令将限制 FF 的值。
FFPreviousInv (Status1.20)	BOOL	FFPrevious < - 100 或 FFPrevious > 100。指令将限制 FF_{n-1} 的值。

HandFBInv (Status1.21)	BOOL	HandFB < 0 或 HandFB > 100。指令将限制 CV 的值。
PGainInv (Status1.22)	BOOL	PGain < 0。指令使用 PGain = 0 值。
IGainInv (Status1.23)	BOOL	IGain < 0。指令使用 IGain = 0 值。
DGainInv (Status1.24)	BOOL	DGain < 0。指令使用 DGain = 0 值。
ZCDeadbandInv (Status1.25)	BOOL	ZCDeadband < 0。指令禁用过零死区。
PVDeadbandInv (Status1.26)	BOOL	PVDeadband < 0。指令将 PVDeadband 限制为零。
PVROCLimitsInv (Status1.27)	BOOL	PVROCPosLimit < 0、 PVROCNegLimit < 0 或 PVROCPeriod < 0。
DevHLLimitsInv (Status1.28)	BOOL	偏差上下限无效。下下限 < 0， 下限 < 0，上限 < 0 或者上上限 < 0。指令将无效限值限制为 0。
DevDeadbandInv (Status1.29)	BOOL	偏差死区 < 0。该指令使用 DevDeadband = 0 值。
Status2	DINT	功能块的时序状态。
TimingModeInv (Status2.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status2.28)	BOOL	仅用于实时采样模式。当 ABS|DeltaT - RTSTime| > 1（0.001 秒）时，设置为真。
RTSTimeInv (Status2.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status2.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status2.31)	BOOL	DeltaT 值无效。

说明

当指令级联/比率或自动模式下执行时，PID 算法会对 CV 输出进行相应调节，使 PV 保持在 SP 值。

ControlAction 置位后，需将计算的 EPercent 和 PVPIDPercent 值取反，然后再供控制算法使用。

下表说明指令计算各个 PID 项的方法。

PID 项	计算方法
比例	比例项的计算方法如下： • PVEProportional 置位时，使用 PV 值计算 • PVEProportional 清零时，使用误差值计算 设置 PGain = 0 时，禁用比例控制。
积分	积分项使用误差值计算。设置 IGain = 0 时，禁用积分控制。当 DependIndepend 置位时，设置 PGain = 0 也会禁用积分控制。
微分	微分项的计算方法如下： • PVEDerivative 置位时，使用 PV 值计算 • PVEDerivative 清零时，使用误差值计算 设置 DGain = 0 时，禁用微分控制。当 DependIndepend 置位时，设置 PGain = 0 也会禁用微分控制。 当 DSmoothing 置位时 ,启用微分平滑处理 ,当 DSmoothing 清零时，禁用微分平滑处理。微分平滑处理可减少噪声 PV 信号引起的 CV 输出"抖动"，但也会限制高微分增益的作用。

计算 CV

PID 控制算法通过将上次执行指令时的 Delta PTerm、Delta ITerm、Delta

DTerm 和 CV（例如 CV_{n-1}）相加，来计算 CV 值。当 CVsetPrevious 置位时，

CVn-1 设置为等于 CVPrevious。这样，可以在计算 CV 值之前，将 CV_{n-1} 预设为一个指定值。

$CalculatedCV = CV_{n-1} + D\Delta PTerm + DITerm + DDTerm$

PIDE 算法

PIDE 指令使用与大多数 DCS 系统类似的速度形式 PID 算法。速度形式算法的优点包括：

- 无扰动自适应增益变化 - 无需初始化算法即可实时完成增益变化。
- 多回路控制方案 - 可通过调节 CV_{n-1} 项来实现多回路间的交叉限制。

独立增益形式

$$CV_n = CV_{n-1} + K_P\Delta E + \frac{K_I}{60}E\Delta t + 60K_D\frac{E_n - 2E_{n-1} + E_{n-2}}{\Delta t}$$

在这种形式的算法中，每个算法项（比例、积分和微分）均采用单独的增益。更改一个增益只会影响相应的项，而不影响其他项，其中：

PIDE 项：	说明：
CV	Control variable
E	以量程百分比表示的误差
Δt	回路使用的更新时间（秒）
K_p	比例增益
K_I	积分增益（分钟）$^{-1}$ K_I 值越大，积分响应速度越快。
K_D	微分增益（分钟）

相关增益形式

$$CV_n = CV_{n-1} + K_C\left(\Delta E + \frac{1}{60T_I}E\Delta t + 60T_D\frac{E_n - 2E_{n-1} + E_{n-2}}{\Delta t}\right)$$

在这种形式的算法中，比例增益的更改会影响控制器增益。通过更改控制器增益，可以同时更改三个项（比例、积分和微分）的操作，其中：

PIDE 项：	说明：
CV	Control variable
E	以量程百分比表示的误差
Δt	回路使用的更新时间（秒）
K_c	控制器增益
T_I	积分时间常数（分钟）。T_I 值越大，积分响应速度越慢 为了响应误差阶跃变化，积分项需要花费 T_I 分钟来重复比例项的动作。
T_D	微分时间常数（分钟）

PIDE 项：	说明：
CV	Control variable
E	以量程百分比表示的误差
Δt	回路使用的更新时间（秒）
Kp	比例增益
K_I	积分增益（分钟）$^{-1}$ K_I 值越大，积分响应速度越快。
K_D	微分增益（分钟）

确定要使用的算法

- $K_P = K_C$
- $K_I = \frac{K_C}{T_I}$
- $K_D = K_C T_D$

上面的 PIDE 公式是 PIDE 指令使用的典型算法。对于比例项和微分项，可以通过调节 PVEProportional 和 PVEDerivative 参数来用误差变化量代替 PV 变化量（量程百分比）。默认情况下，PIDE 指令对于比例项使用误差变化量，而对于微分项则使用 PV 变化量。这样可以避免设置点的变化产生大幅的微分尖峰。

可使用以下公式转换不同 PIDE 算法形式下的增益：

- $K_P = K_C$
- $K_I = \frac{K_C}{T_I}$
- $K_D = K_C T_D$

每种算法通过相应的增益来实现相同的控制。有些用户喜欢使用独立增益形式，因为这样可以分别调节各个增益而不会影响其他项。而有些用户则喜欢相关增益形式，因为这样至少可以在某种程度上仅调节控制器增益，无需分别调节每个增益，即可使 PID 回路的作用发生整体性变化。

监视 PIDE 指令

PIDE 指令有相应的操作员面板。

自动调谐 PIDE 指令

Logix Designer 应用程序 PIDE 自动调谐器是一个内置在 PIDE 指令中的开环自动调谐器。可以通过 PanelView 终端或任何其他操作员接口设备以及 Logix Designer 应用程序完成自动调谐。PIDE 块有一个自动调谐标签（PIDE_AUTOTUNE 类型），可以为要自动调谐的 PIDE 块指定该标签。

PIDE 自动调谐器随应用程序一道安装，但要启用该调谐器，需要使用激活密钥。该自动调谐器仅能用于功能块编程，而不适用于梯形图或结构化文本编程。

可使用自动调谐标签来为 PIDE 块指定和配置自动调谐标签。

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。

指令首次运行	不适用
指令首次扫描	如果 CVFault 和 CVEUSpanInv 置位，请参见本指令后面的“处理故障”部分。 如果 CVFault 和 CVEUSpanInv 清零： 1. CVInitializing 置位 2. 如果 PVFault 置位，而 PVSpanInv 和 SPlimitsInv 清零。请参见本指令的“处理故障”部分。 3. 不执行 PID 控制算法。 4. 该指令设置 CVEU = CVInitValue 以及 CV = 相应百分比。 5. 当 CVInitializing 和 ManualAfterInit 置位时，指令将禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，指令会切换为手动模式。如果 ManualAfterInit 清零，则模式保持不变。 CVEu = CVInitValue $CV_{n-1} = CV = \frac{CVEU - CVEUMin}{CVEUMax - CVEUMin} \times 100$ CVOper = CV
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

当 CVInitReq 置位、指令的首次扫描期间或者 CVFault 由置位跳变为清零（不良变为良好）时，指令会将 CVEU 和 CV 输出初始化为 CVInitValue 值。如果时序模式未设置为过采样模式且 EnableIn 由清零跳变为置位，指令会初始化 CVEU 和 CV 值。在完成初始化且 CVInitReq 清零后，CVInitialization 清零。

CVInitValue 通常来自于模拟量输出的反馈值。CVInitReq 值通常来自于 CVEU 所控制的模拟量输出的“In Hold”状态位。执行初始化过程是为了避免在启动时发送到现场设备的输出信号出现扰动。

使用级联 PID 回路时，主 PID 回路可以在次级回路初始化后或者次级回路离开级联/比率模式后进行初始化。这种情况下，将来自次级回路的 InitPrimary 输出和 SP 输出状态送入主回路的 CVInitReq 输入和 CVInitValue 输入。

如果 CVFault 或 CVEUSpanInv 置位，指令不会初始化，CVEU 和 CV 值也不会更新。

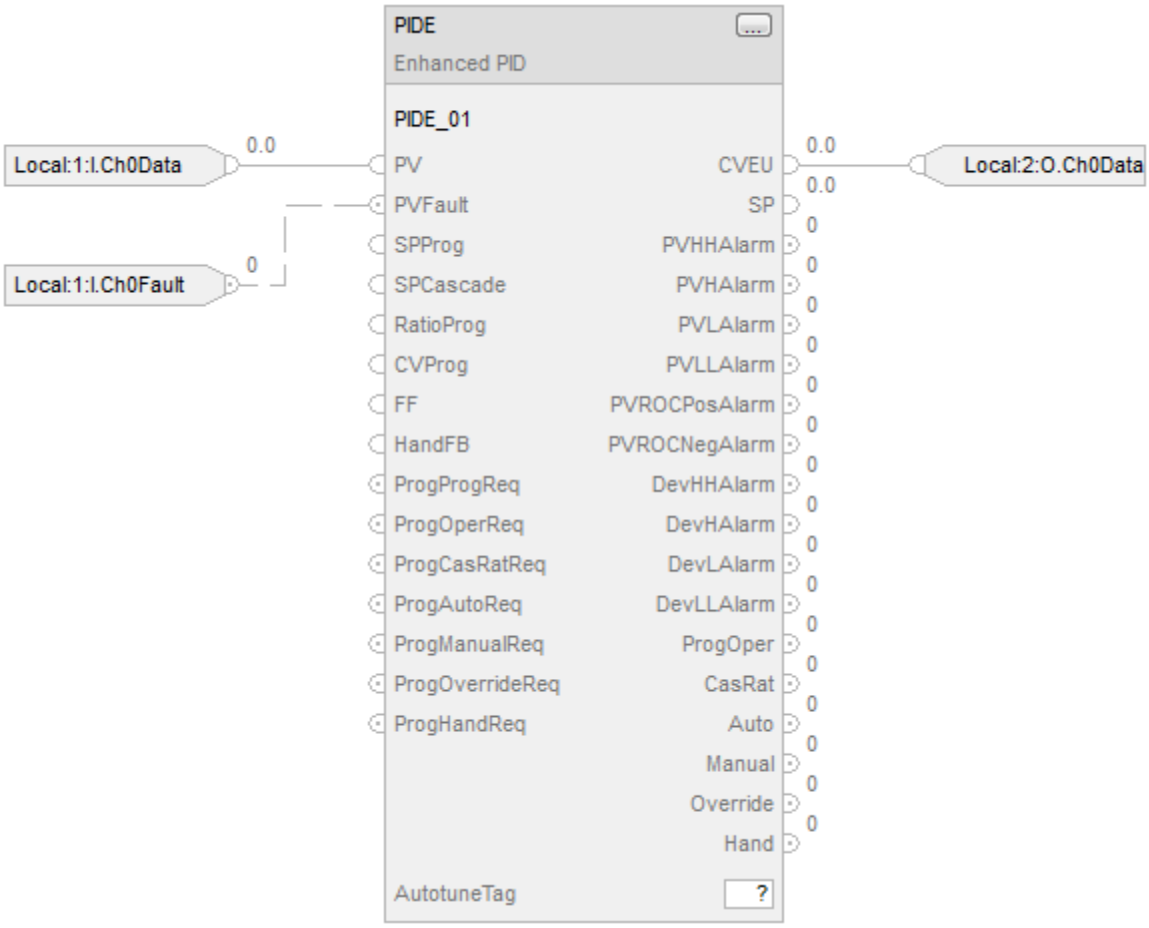
示例

示例 1

实施 PIDE 指令最简单的方法是在周期性任务的程序中创建一个功能块例程。PIDE 指令默认的时序模式为周期性模式。当 PIDE 指令用于周期性任务且处于周期性时序模式时，会自动使用周期性任务的更新速率作为 Delta t 更新时间。用户只需将过程变量模拟量输入接入 PIDE 指令的 PV 参数，将 PIDE 指令的 CVEU 输出接入受控变量模拟量输出。

也可以选择将模拟量输入的故障指示器（如果有）接入 PIDE 指令的 PVFault 参数。这样会强制 PIDE 在模拟量输入发生故障时进入手动模式，并在 PV 信号不可用时停止 PIDE CVEU 输出的积分饱和或解饱和。

功能块



结构化文本

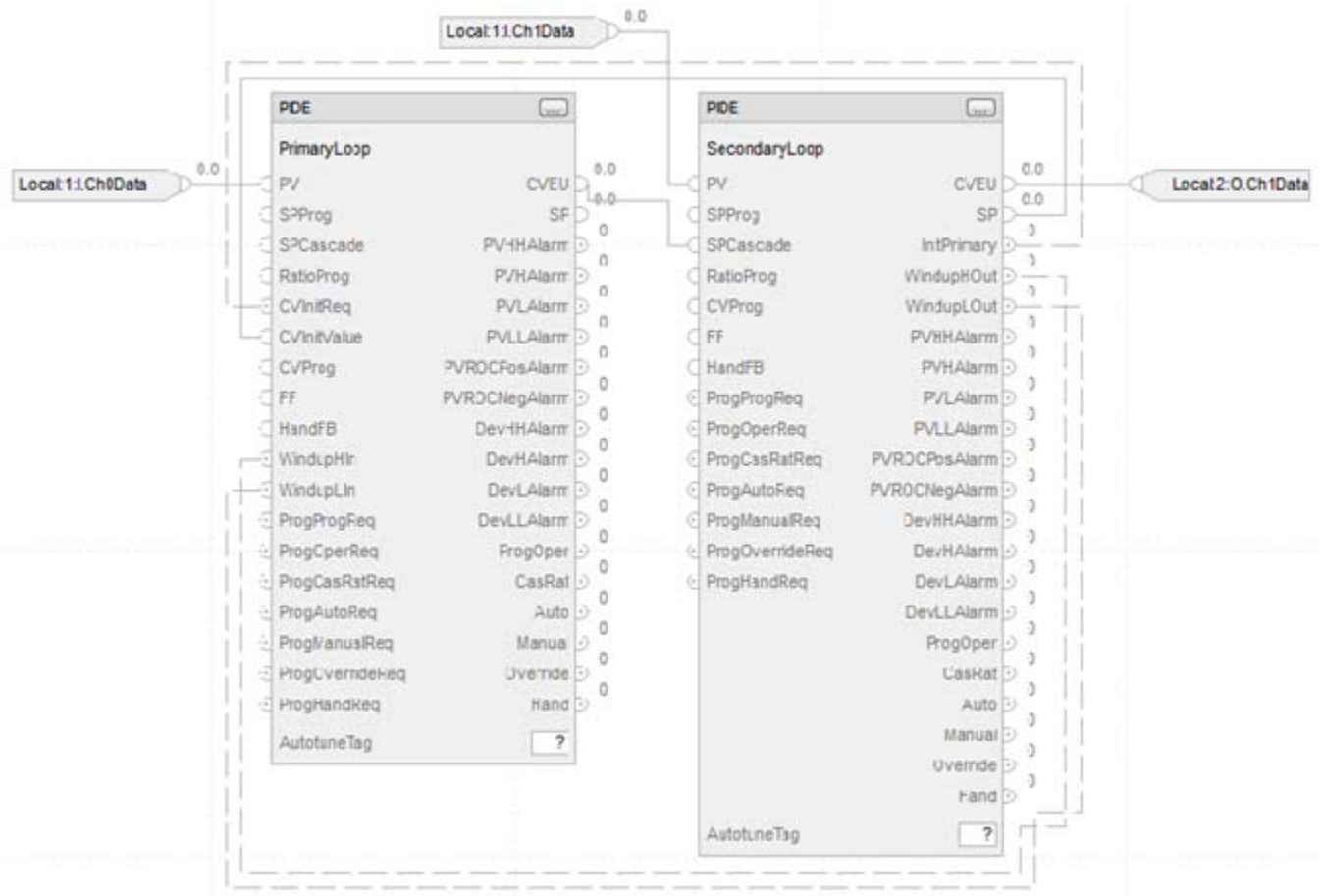
```
PIDE_01.PV := Local:1:I.Ch0Data;

PIDE_01.PVFault := Local:1:I.Ch0Fault;

PIDE(PIDE_01);

Local:2:)O.Ch0Data :=PIDE_01.CVEU;
```

示例 2

当受控变量经常遭受外部干扰，进而导致所控制的过程变量遭受干扰时，级联控制非常有用。例如，尝试通过调节送入容器周围加热套中的蒸汽量来控制容器中液体的温度。如果由于某个上游过程导致蒸汽流量骤降，容器中液体的温度最终也会下降，PIDE 指令随即打开蒸汽阀来补偿温度下降。

在本例中，级联回路可以在蒸汽流量下降导致容器液体温度下降前打开蒸汽阀，从而实现更好的控制。要实现级联回路，可使用 PIDE 指令来根据蒸汽流量变送器的过程变量信号控制蒸汽阀开关。这是级联对的次级回路。另一条 PIDE 指令（称为主回路）使用液体温度作为过程变量，并将其 CV 输出发送到次级回路的设置点。通过这种方法，主温度回路可以向次级蒸汽流量回路请求特定的蒸汽流量。随后，蒸汽流量回路负责提供温度回路所请求的蒸汽量，从而保持恒定的液体温度。

功能块



结构化文本

```
PrimaryLoop.PV := Local:1:I.CH0Data;
PrimaryLoop.CVInitReq := SecondaryLoop.InitPrimary;
PrimaryLoop.CVInitValue := SecondaryLoop.SP;
PrimaryLoop.WindupHIn := SecondaryLoop.WindupHOut;
PrimaryLoop.WindupLIn := SecondaryLoop.WindupLOut;

PIDE(PrimaryLoop);
```

```
SecondaryLoop.PV := Local:1:I.Ch1Data;
SecondaryLoop.SPCascade := PrimaryLoop.CVEU;

PIDE(SecondaryLoop);

Local:2:O.Ch0Data:= SecondaryLoop.CVEU;
```

要使一对级联回路正常工作，次级回路的过程响应速度必须比主回路快。原因在于，在任何干扰对主回路的过程造成影响之前，次级回路的过程必须能够先对这些干扰进行补偿。在本例中，如果蒸汽流量下降，次级控制器必须在液体温度受到影响之前动作，以便增加蒸汽流量。

要设立一对级联 PIDE 指令，请将次级回路中的 *AllowCasRat* 输入参数置位。这样可使次级回路进入级联/比率模式。接下来，将主回路的 *CVEU* 连接到次级回路的 *SPCascade* 参数。当次级回路进入级联/比率模式时，*SPCascade* 值将用作次级回路的 SP。主回路中 CVEU 的工程单位范围应当与次级回路中 PV 的工程单位范围一致。这样，主回路便可以将其 CV 的 0-100% 值标定为次级回路上设置点所用的相应工程单位。

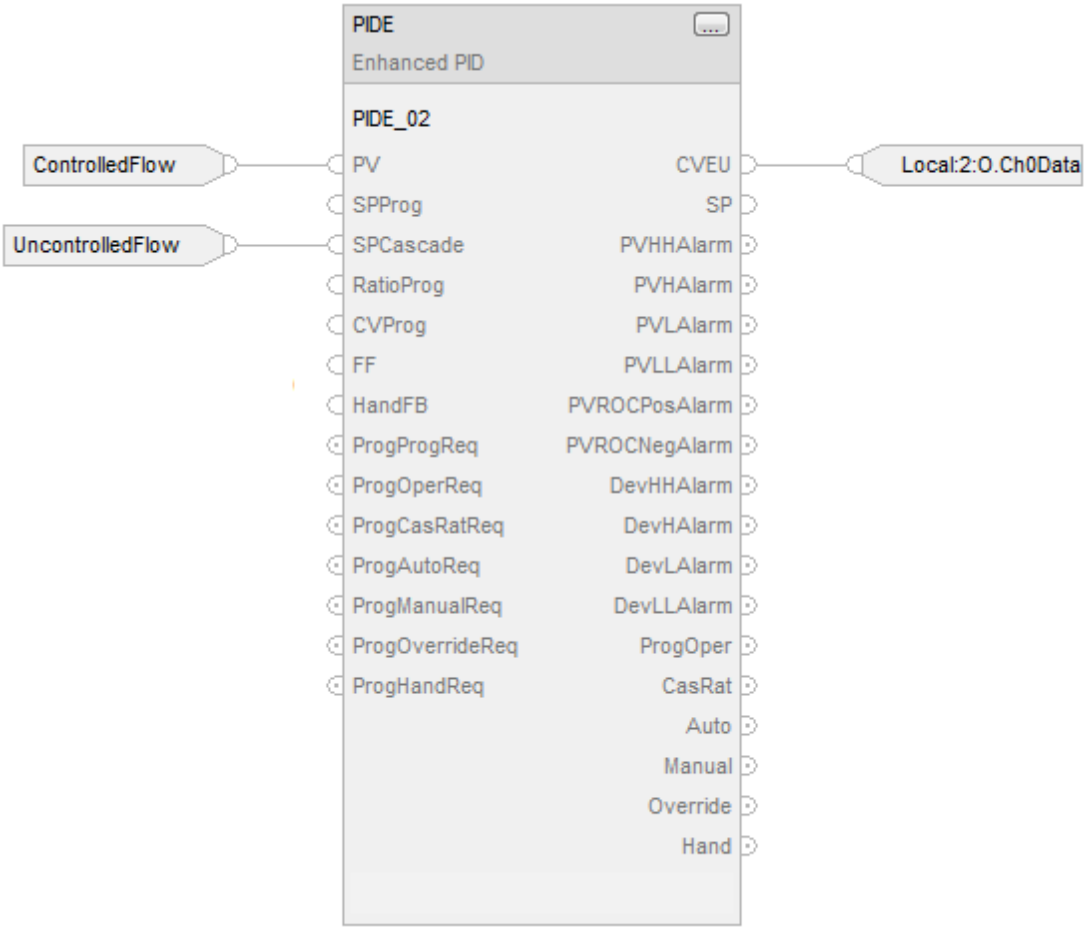
PIDE 指令还支持其他多种功能，从而更有效地支持级联控制。将次级回路的 *InitPrimary* 输出连接到主回路的 *CVInitReq* 输入，将次级回路的 SP 输出连接到主回路的 *CVInitValue* 输入。这样，当次级回路退出级联/比率模式时，会将主回路的 CVEU 值设为次级回路的 SP，从而在将次级回路重新切换为级联/比率模式时实现无扰动转换。同样，将次级回路的 *WindupHOut* 和 WindupLOut 输出连接到主回路的 *WindupHIn* 和 *WindupLIn* 输入。因此，主回路在次级回路达到 SP 限值或 CV 限值时会停止增大或减小（视情况而定）CVEU 值，并且在发生上述情况时消除主回路上的任何积分饱和。

示例 3

比率控制通常用于按设定比例将一种液体添加到另一种液体中。例如，如果要按恒定比率将两种反应物（例如 A 和 B）添加到容器中，而由于上游过程的某些干扰，反应物 A 的流速可能随时间发生变化，这时便可以使用比率控制器自动调节反应物 B 的添加速率。在本例中，反应物 A 通常称为“非可控”流，因为它不受 PIDE 指令控制。而反应物 B 则称为“受控”流。

要使用 PIDE 指令执行比率控制，需将 *AllowCasRat* 和 *UseRatio* 输入参数置位。将非可控流连接到 *SPCascade* 输入参数。处于级联/比率模式时，将非可控流乘以 *RatioOper*（在操作员控制模式下）或 RatioProg（在程序控制模式下），得出的乘积会由 PIDE 指令用作设置点。

功能块



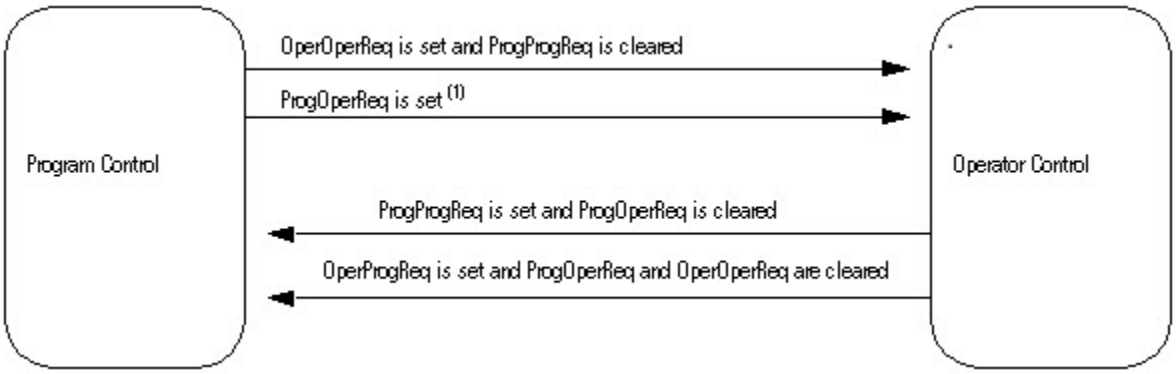
结构化文本

```
PIDE_01.PV := ControlledFlow;
PIDE_01.SPCascade := UncontrolledFlow;

PIDE(PIDE_01);

Local:2:O.Ch0Data := PIDE_01.CVEU;
```

在程序控制与操作员控制之间切换

PIDE 指令可通过用户程序或操作员界面控制。用户可以随时更改控制模式。程序控制和操作员控制都使用 ProgOper 这一输出。当 ProgOper 置位时，控制模式为程序控制；当 ProgOper 清零时，控制模式则为操作员控制。

下图显示了 PIDE 指令在程序控制与操作员控制之间进行切换的方式。



(1) 当 ProgOperReq 置位时，指令仍会处于操作员控制模式。

工作模式

PIDE 指令支持以下 PID 模式。

PID 工作模式	说明
级联/比率	处于级联/比率模式时，指令将计算 CV 的变化量。而且，会对 CV 进行相应的调节，使 PV 保持在 SPCascade 值或 SPCascade 值与 Ratio 值的乘积。SPCascade 来自级联控制中主 PID 回路的 CVEU 或比率控制回路的"非可控"流。 可使用 OperCasRatReq 或 ProgCasRatReq 选择级联/比率模式： 将 OperCasRatReq 置位可请求进入级联/比率模式。当 ProgOper、ProgOverrideReq、ProgHandReq、OperAutoReq 或 OperManualReq 置位时，或者 AllowCasRat 清零时，会被忽略。 将 ProgCasRatReq 置位可请求进入级联/比率模式。当 ProgOper 或 AllowCasRat 清零时，或者当 ProgOverrideReq、ProgHandReq、ProgAutoReq 或 ProgManualReq 置位时，会被忽略。
自动	处于自动模式时，指令将计算 CV 的变化量。而且，指令会对 CV 进行相应的调节以使 PV 保持在 SP 值。如果处于程序控制模式，则 SP = SPProg；如果处于操作员控制模式，则 SP = SPOper。 可使用 OperAutoReq 或 ProgAutoReq 选择自动模式： 将 OperAutoReq 置位可请求进入自动模式。当 ProgOper、ProgOverrideReq、ProgHandReq 或 OperManualReq 置位时，会被忽略。 将 ProgAutoReq 置位可请求进入自动模式。当 ProgOper 清零时，或者当 ProgOverrideReq、ProgHandReq 或 ProgManualReq 置位时，会被忽略。

手动	处于手动模式时，指令不会计算 CV 的变化量。CV 的值由控制模式确定。如果处于程序控制模式，则 CV = CVProg；如果处于操作员控制模式，则 CV = CVOper。 可使用 OperManualReq 或 ProgManualReq 选择手动模式： 将 OperManualReq 置位可请求进入手动模式。当 ProgOper、ProgOverrideReq 或 ProgHandReq 置位时，会被忽略。 将 ProgManualReq 置位可请求进入手动模式。当 ProgOper 清零时，或者当 ProgOverrideReq 或 ProgHandReq 置位时，会被忽略。
超控	处于超控模式时，指令不会计算 CV 的变化量。 CV = CVOverride，与控制模式无关。超控模式通常用于设置 PID 回路的"安全状态"。 可使用 ProgOverrideReq 选择超控模式： 将 ProgOverrideReq 置位可请求进入超控模式。当 ProgHandReq 清零时，会被忽略。
手控	处于手控模式时，PID 算法不计算 CV 的变化量。 CV = HandFB，与控制模式无关。手控模式通常用于指示最终控制元件的控制功能已由现场手控/自动站接管。 可使用 ProgHandReq 选择手控模式： 将 ProgHandReq 置位可请求进入手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。

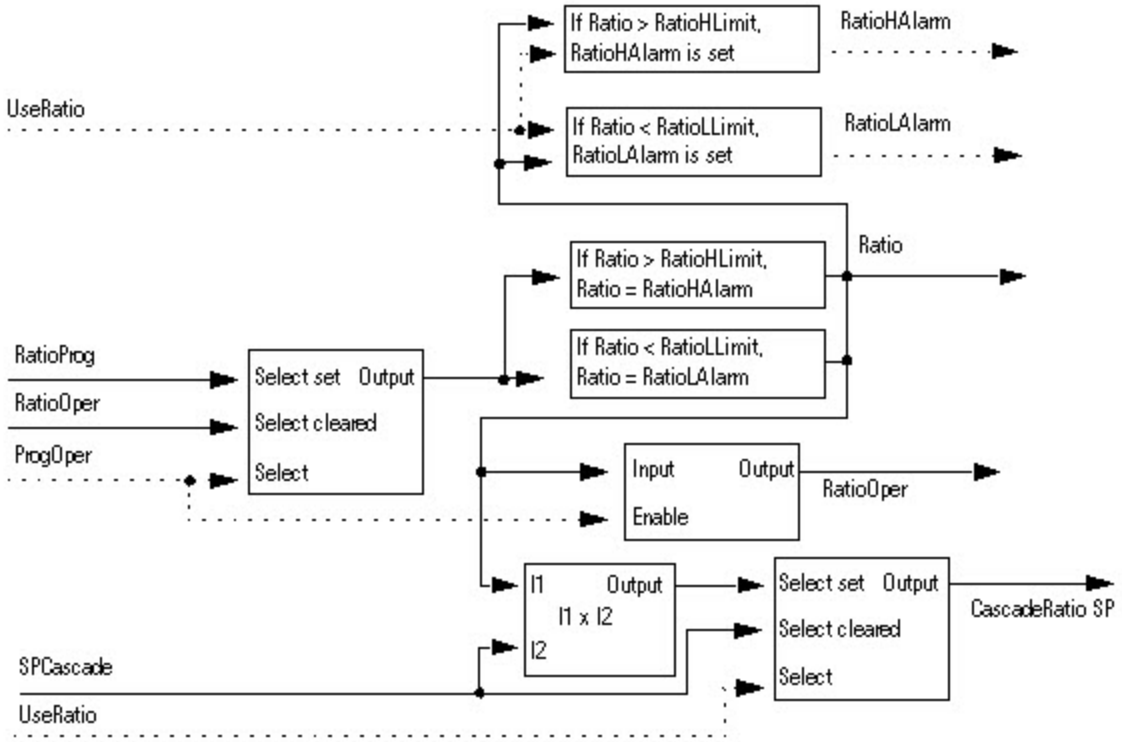
处于程序控制模式时，级联/比率、自动和手动模式可由用户程序控制；处于操作员控制模式下时则可通过操作员界面进行控制。超控和手控模式的模式请求输入只能由用户程序控制；这些输入可在程序控制和操作员控制两种模式下工作。

选择设置点

一旦确定处于程序控制还是操作员控制模式以及相应的 PID 模式，指令便可获取正确的 SP 值。可以选择级联/比率 SP 或当前 SP。

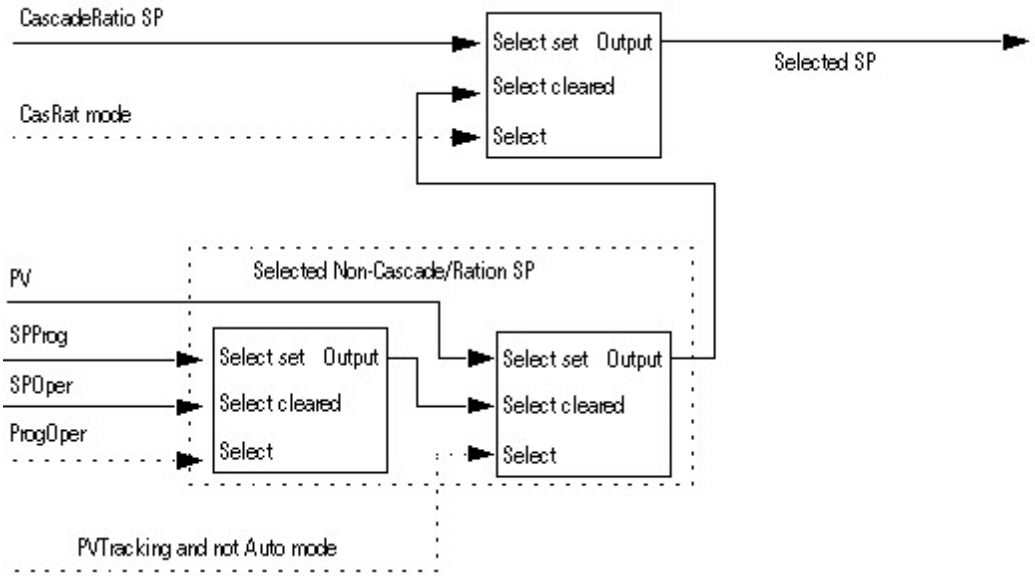
级联/比率 SP

级联/比率 SP 取决于 UseRatio 值和 ProgOper 值。



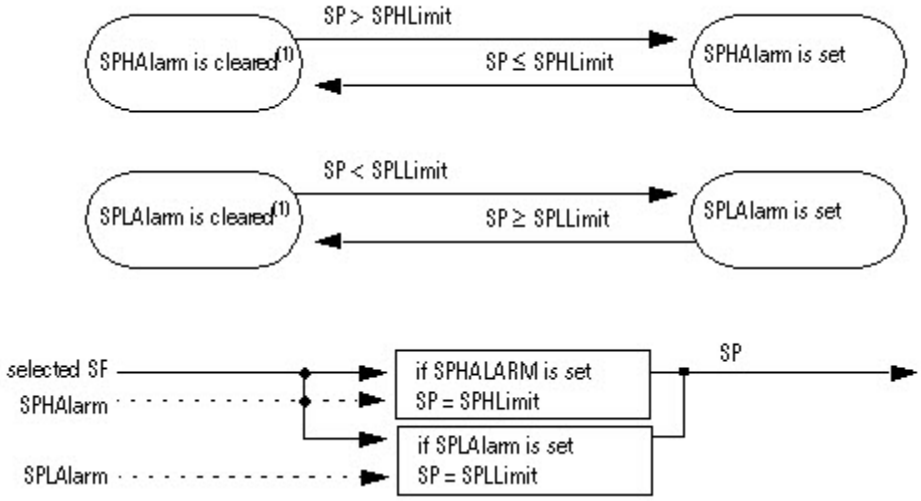
当前 SP

当前 SP 取决于级联/比率模式、PVTracking 值、自动模式以及 ProgOper 值。



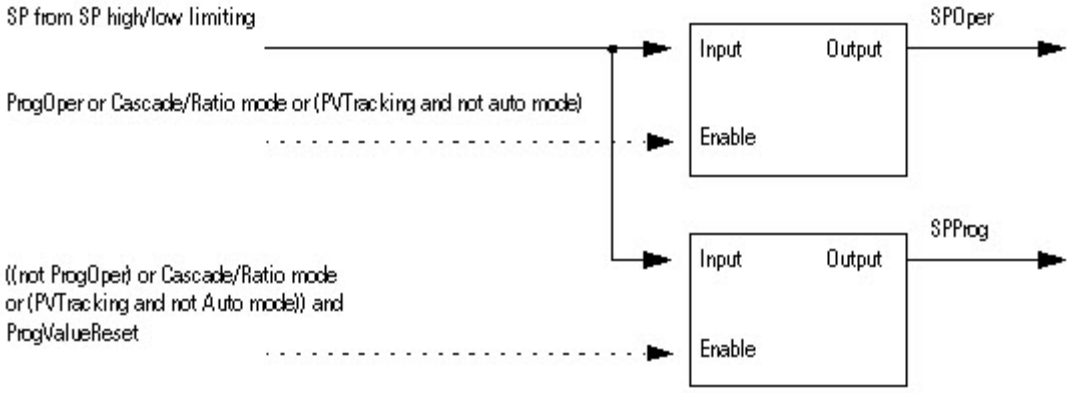
SP 上限/下限

上下限报警算法会将 SP 与 SPHLimit 和 SPLLimit 报警限值进行比较。SPHLimit 不能大于 PVEUMax 且 SPLLimit 不能小于 PVEUMin 。



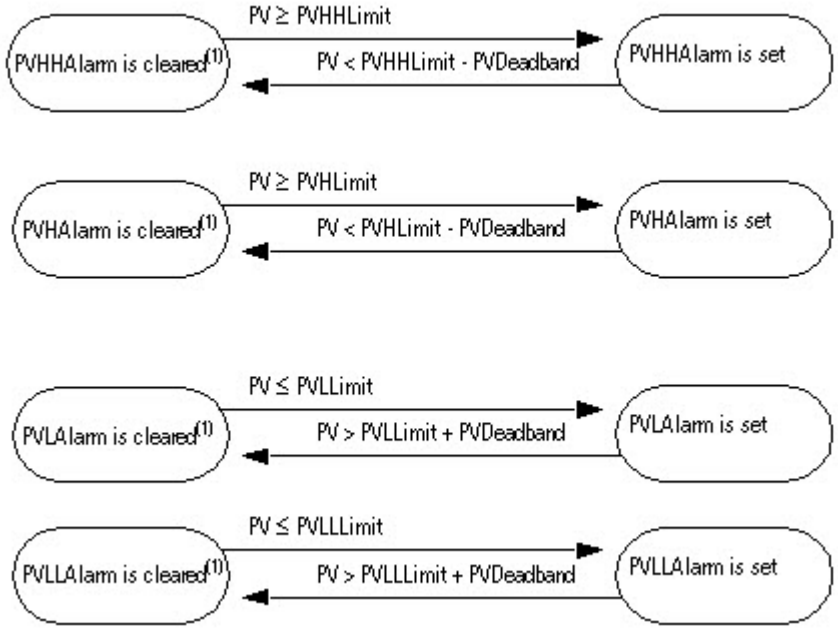
更新 SPOper 值和 SPProg 值

为了在程序控制和操作员控制之间实现无扰动切换，或者在从级联/比率模式切换为其他模式时实现无扰动控制，PIDE 指令会设置 SPOper = SP 或 SPProg = SP。



PV 上限/下限报警

上上限至下下限报警算法会将 PV 与 PV 报警限值以及 PV 报警限值加减 PV 报警死区的结果进行比较



(1) 在指令首次扫描期间，指令会将所有的 PV 报警输出清零。当 PVFaulted 置位时，指令也会将 PV 报警输出清零并禁用报警算法。

PV 变化率报警

PV 变化率 (ROC) 报警会将 PV 值在 PVROCPeriod 期间的变化量与 PV 正负变化率限值进行比较。PVROCPeriod 为变化率报警提供一种死区类型。例如，如果使用执行周期为 100 ms、大小为 2°F/s 的 ROC 报警限值，而模拟量输入模块的分辨率为 1°F，则输入值每次发生变化时，都会生成 ROC 报警，因为指令认为变化率为 10°F/s。然而，如果输入大于等于 1 秒的 PVROCPeriod，则仅当变化率确实超过 2°F/s 的限值时，才会生成 ROC 报警。

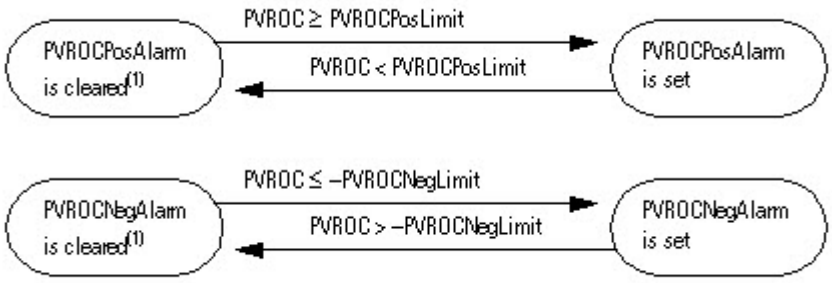
仅当 PVROCPeriod 到期时才会执行计算 ROC。变化率的计算方式如下：

ElapsedROCPeriod = ElapsedROCPeriod + ElapsedTimeSinceLastExecution

如果 ElapsedROCPeriod ≥ PVROCPeriod，则：

参数：	值：
PVROC	$\frac{PV_N - PVROC_{N-1}}{PVROCPeriod}$
$PVROC_{N-1}$	$PVROC_{N-1} = PV$
ElapsedROCPeriod	ElapsedROCPeriod = 0

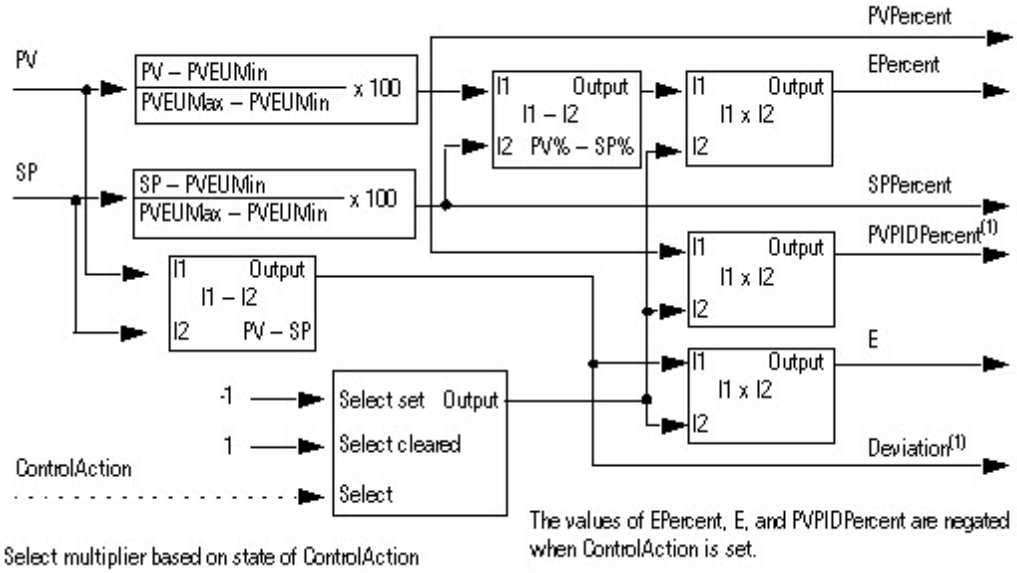
计算出 PVROC 后，即可按以下方式确定 PV ROC 报警：



(1) 在指令首次扫描期间，指令会将 PV ROC 报警输出清零。当 PVFaulted 置位时，指令还会将 PVROC 报警输出清零并禁用 PV ROC 报警算法。

将 PV 值和 SP 值转换为百分比

执行 PID 控制算法之前，指令会将 PV 和 SP 转换为百分比并计算误差。此误差为 PV 与 SP 的差值。ControlAction 置位时，需要将 EPercent、E 和 PVPIDPercent 的值取反，然后才能供 PID 算法使用。

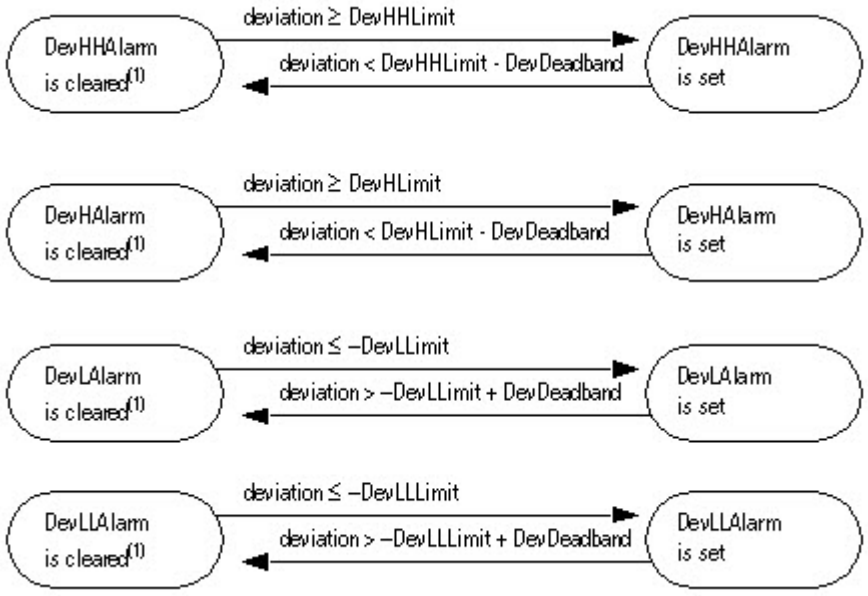


(1) PVPIDPercent 和 Deviation 是供 PID 控制算法使用的内部参数。

偏差上限/下限报警

偏差指过程变量（PV）与设置点（SP）之差。偏差报警提示操作员过程变量和设置点值存在差异。

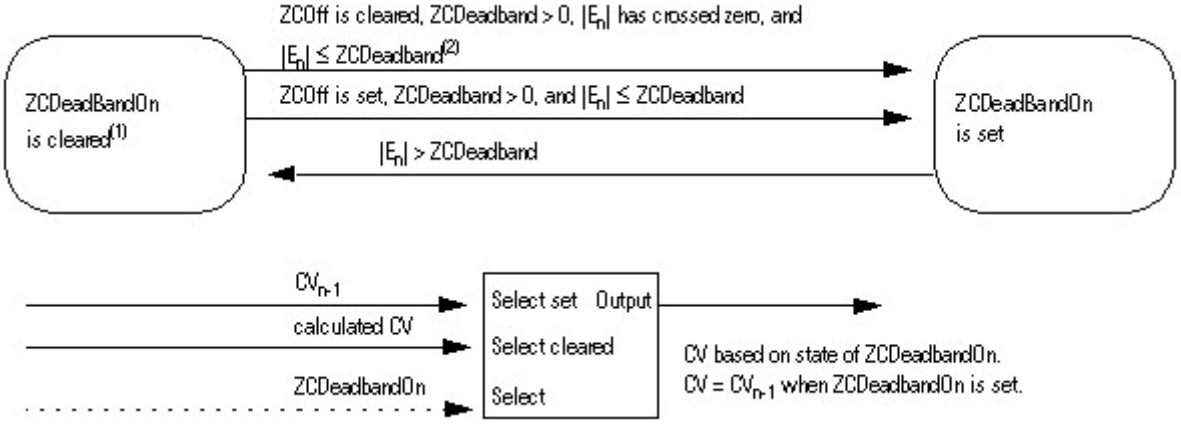
上上限至下下限报警算法会将偏差与偏差报警限值以及偏差报警限值加减死区的结果进行比较。



(1) 在指令首次扫描期间，指令会将偏差报警输出清零。当 PVFaulted 或 PVSpanInv 置位时，指令还会将偏差报警输出清零并禁用报警算法。

过零死区控制

可以限制 CV 值，使其值在误差处于 ZCDeadband 指定的范围内时 (|E|≤ZCDeadband) 保持不变。



(1) 当 ZCOff 清零、ZCDeadband > 0、误差首次过零（即 En >= 0 且 En-1 < 0，或者 En <= 0 且 En-1 > 0），且 |En| <= ZCDeadband 时，指令将 ZCDeadbandOn 置位。

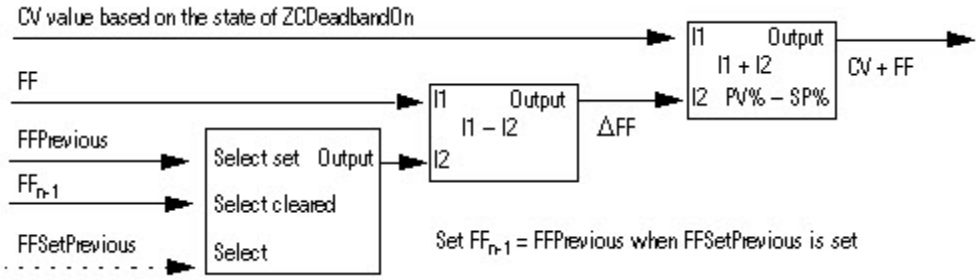
(2) 切换为自动或级联/比率模式时，指令会设置 En-1 = En。

满足以下条件时，指令将禁用过零算法并将 ZCDeadband 清零：

- 指令首次扫描期间
- ZCDeadband ≤ 0
- 当前模式不是自动或级联/比率模式
- PVFaulted 置位
- PVSpanInv 置位

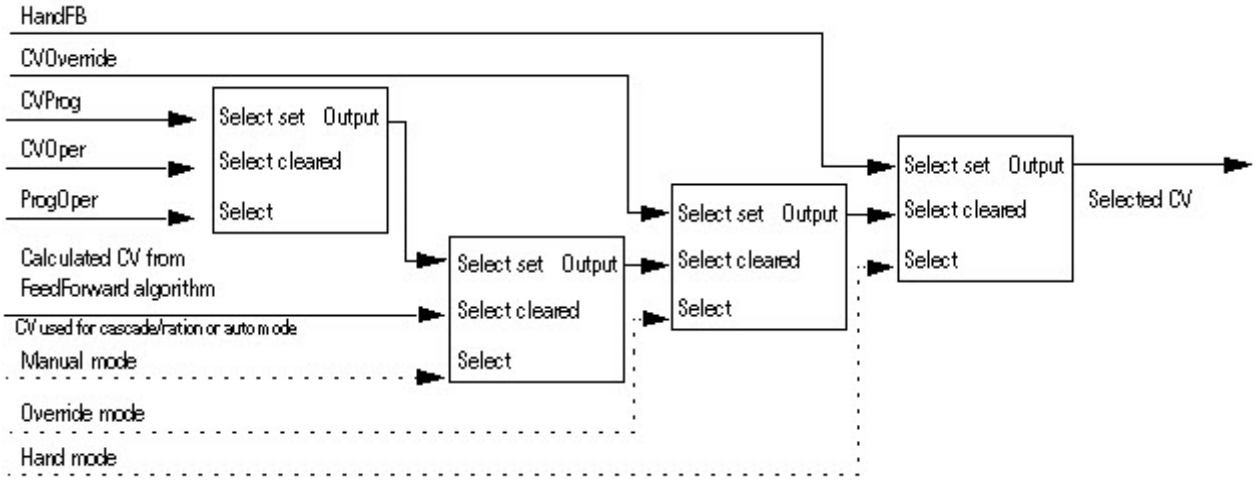
前馈控制

计算 CV 时，需将过零算法得出的 CV 与 ΔFF 相加。ΔFF 值 = FF – FFn-1。FFSetPrevious 置位时，FFn-1=FFPrevious。这样便可在指令计算 ΔFF 值之前将 FFn-1 预设为指定值。



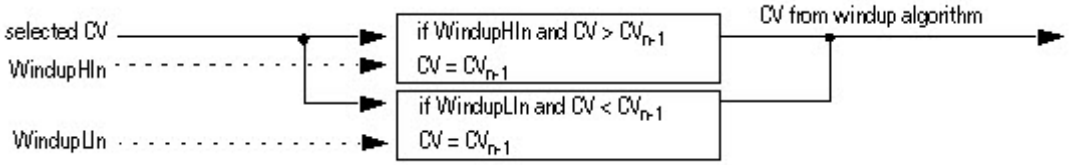
选择控制变量

执行 PID 算法后，根据程序控制或操作员控制模式以及当前 PID 模式选择 CV。



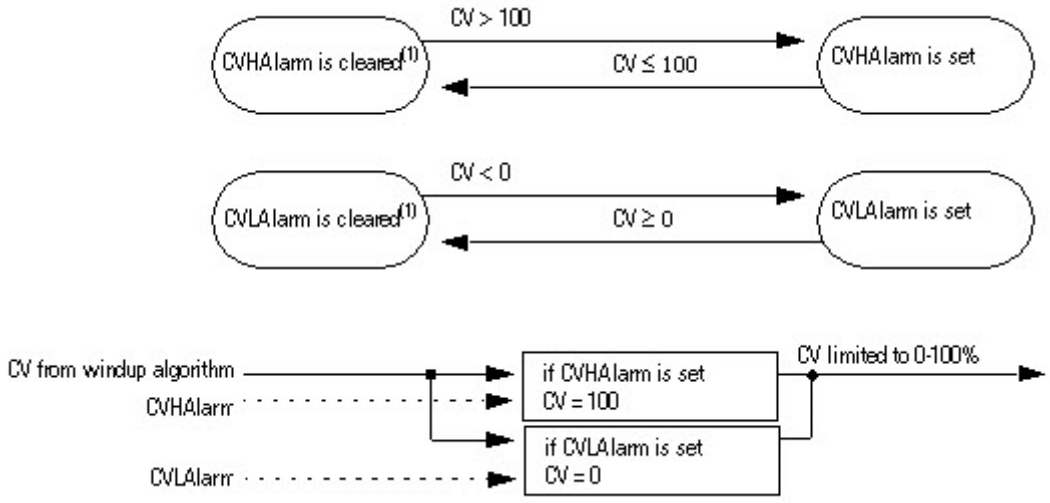
CV 积分饱和限制

可以对 CV 进行限制，使其值在 WindupHIn 置位时不会增大，或者在 WindupLIn 置位时不会减小。这些输入通常是来自次级回路的 WindupHOut 或 WindupLOut 输出。如果 CVInitializing、CVFault 或 CVEUSpanInv 置位，WindupHIn 和 WindupLIn 输入会被忽略。



CV 百分比限制

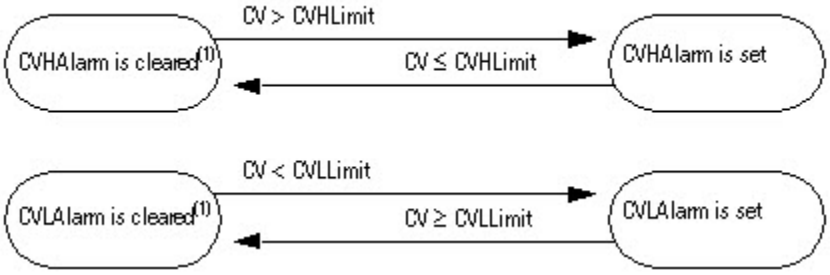
下图展示指令如何确定限制 CV 百分比的方式。

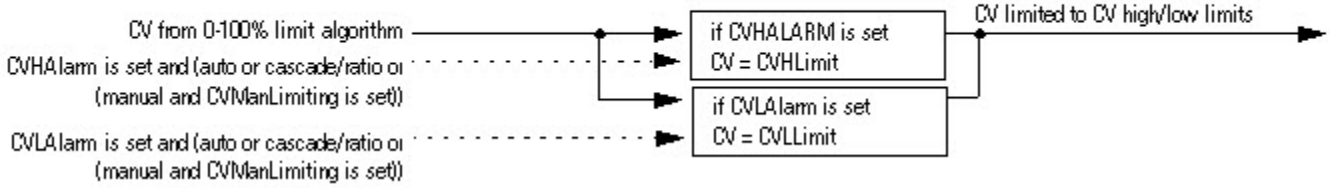


(1) 在指令首次扫描期间，指令会将报警输出清零。

CV 上限/下限

指令始终根据 CVHLimit 和 CVLLimit 来执行报警。处于自动或级联/比率模式时，CV 由 CVHLimit 和 CVLLimit 进行限制。处于手动模式时，如果 CVManLimiting 置位，则 CV 由 CVHLimit 和 CVLLimit 进行限制。否则，CV 由 0 和 100% 限制。





(1) 在指令首次扫描期间，指令会将报警输出清零。

CV 变化率限制

处于自动或级联/比率模式时，或者处于手动模式且 CVManLimiting 置位时，PIDE 指令会对 CV 变化率进行限制。使用零值会禁用 CV 变化率限制。

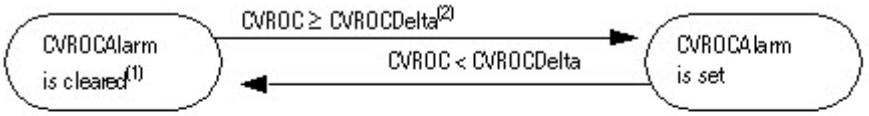
CV 变化率的计算方式如下：

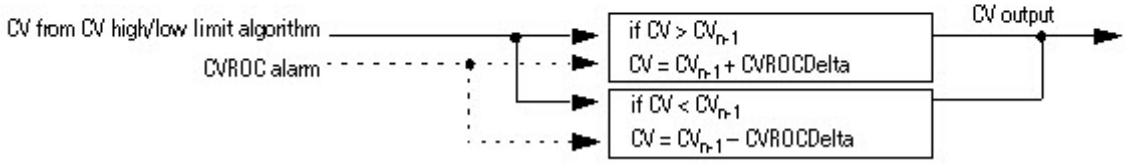
$$CVROC = |CV_n - CV_{n-1}|$$

$$CVROCDelta = CVROCLimit \times DeltaT$$

其中，DeltaT 以秒为单位。

计算出 CV 变化率后，即可按以下方式确定 CV 变化率报警：



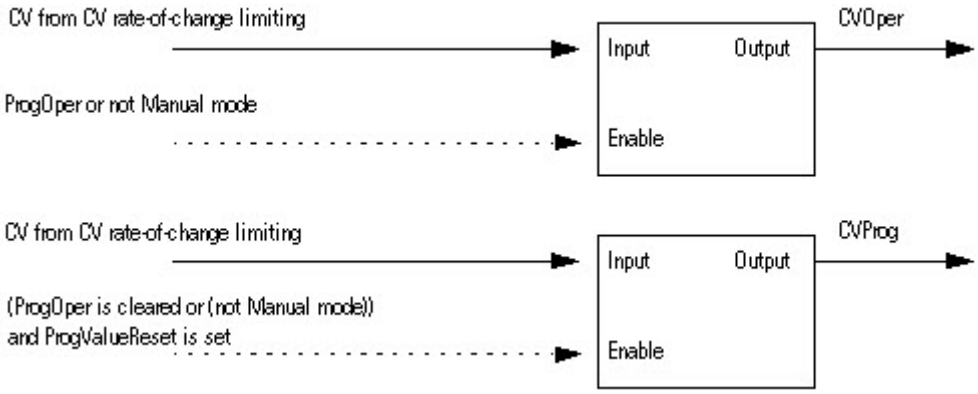


(1) 在指令首次扫描期间，指令会将报警输出清零。当 CVInitializing 置位时，指令还会将报警输出清零并禁用 CV 变化率算法。

(2) 处于自动或级联/比率模式时，或者处于手动模式且 CVManLimiting 置位时，指令会限制 CV 的变化。

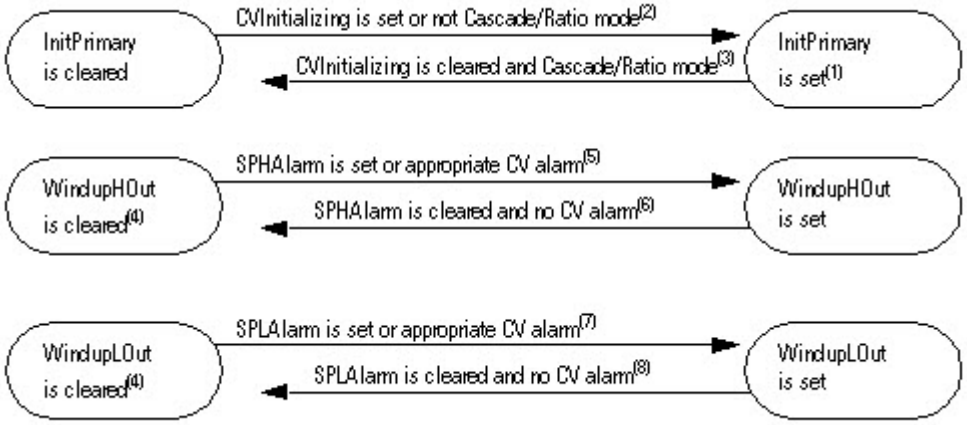
更新 CVOper 值和 CVProg 值

如果未处于操作员手动模式，PIDE 指令会设置 CVOper = CV。因此，从任何其他控制模式切换为操作员手动模式时都可实现无扰动切换。



主回路控制

主回路控制通常由主 PID 回路使用，用于在使用级联/比率模式时，实现无扰动切换和抗积分饱和。主回路控制包括初始化主回路输出和抗积分饱和输出。InitPrimary 输出通常供主 PID 回路的 CVInitReq 输入使用。积分饱和输出通常供主回路的积分饱和输入使用，用于限制其 CV 输出的积分饱和。



(1) 在指令首次扫描期间，指令会将 InitPrimary 置位。

(2) 当 CVInitializing 置位或处于非级联/比率模式时，指令会将 InitPrimary 置位。

(3) 当 CVInitializing 清零且处于级联/比率模式时，指令会将 InitPrimary 清零。

(4) 在指令首次扫描期间，指令会将积分饱和输出清零。当 CVInitializing 置位时，或者 CVFaulted 或 CVEUSpanInv 置位时，指令也会将积分饱和输出清零并禁用 CV 积分饱和算法。

(5) 当 SPHAlarm 置位时、ControlAction 清零且 CVHAlarm 置位时，或者 ControlAction 置位且 CVLAlarm 置位时，指令会将 WindupHOut 置位。

SP 限值和 CV 限值的作用相互独立。SP 上限不会限制 CV 值增大。同样，CV 上下限也不会限制 SP 值增大。

(6) 当 SPHAlarm 清零且未设置（ControlAction 清零且 CVHAlarm 置位）及（ControlAction 置位且 CVLAlarm 置位）时，指令会将 WindupHOut 清零。

(7) 当 SPLAlarm 置位时、ControlAction 清零且 CVLAlarm 置位时，或者 ControlAction 置位且 CVHAlarm 置位时，指令会将 WindupLOut 置位。

SP 限值和 CV 限值的作用相互独立。SP 下限不会限制 CV 值增大。同样，CV 上下限也不会限制 SP 值增大。

(8) 当 SPLAlarm 清零且未设置（ControlAction 清零且 CVLAlarm 置位）及（ControlAction 置位且 CVHAlarm 置位）时，指令会将 WindupLOut 清零。

处理故障

下表说明指令处理执行故障的方式：

故障条件	操作
CVFaulted 为真或 CVEUSpanInv 为真	指令不进行初始化，CVInitializing 设置为假 计算 PV 和 SP 百分比以及误差，更新 EPercent 和 PVPIDPercent 的内部参数 不执行 PID 控制算法 禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式。 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。
PVFaulted 为真	禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式 PV 上下限、PV 变化率以及偏差上下限报警输出均设置为假 不执行 PID 控制算法 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。
PVSpanInv 为真或 SPLimitsInv 为真	禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式 不计算 PV 和 SP 百分比 不执行 PID 控制算法 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。
RatioLimitsInv 为真， CasRat 为真，并且 UseRatio 为真	如果尚未处于手控或超控模式，则设置为手动模式 禁用级联/比率模式 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。

TimingModeInv 为真， RTSTimeStampInv 为真，或 DeltaTInv 为真	如果尚未处于手控或超控模式，则设置为手动模式

另请参见

功能块属性 参考页数 545

功能块面板控件 参考页数 604

通用属性 参考页数 589

结构化文本语法 参考页数 559

位置比例 (POSP)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

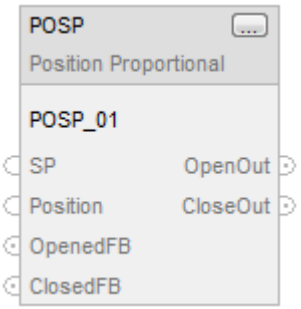
POSP 指令根据用户自定义的循环时间产生宽度与预期位置和实际位置之差成比例的脉冲，来开启或关断触点，从而打开或关闭设备。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

POSP(POSP_tag)

操作数

功能块

操作数	类型	格式	说明
POSP tag	POSITION_PROP	结构	POSP 结构

结构化文本

操作数	类型	格式	说明
block tag	POSITION_PROP	结构	POSP 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

POSITION_PROP 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
SP	REAL	设置点。即期望的位置值。该值必须与 Position 使用相同的工程单位。 有效值 = 任意浮点值 默认值 = 0.0
Position	REAL	位置反馈。该模拟量输入来自设备的位置反馈。 有效值 = 任意浮点值 默认值 = 0.0
OpenedFB	BOOL	已打开反馈。设备完全打开时，该输入将进行指示。输入值为真时，不允许接通打开输出。 默认值为假。
ClosedFB	BOOL	关闭反馈。设备完全关闭时，该输入将进行指示。输入值为真时，不允许接通关闭输出。 默认值为假。
PositionEUMax	REAL	Position 与 SP 的最大标定值。 有效值 = 任意浮点值 默认值 = 100.0

PositionEUMin	REAL	Position 与 SP 的最小标定值。 有效值 = 任意浮点值 默认值 = 0.0
CycleTime	REAL	输出脉冲的周期（秒）。该值为零时，OpenOut 和 CloseOut 清零。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 任意正浮点值 默认值= 0.0
OpenRate	REAL	设备打开速率（%/秒）。该值为零时，OpenOut 清零。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 任意正浮点值 默认值= 0.0
CloseRate	REAL	设备关闭速率（%/秒）。该值为零时，CloseOut 清零。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 任意正浮点值 默认值= 0.0
MaxOnTime	REAL	打开或关闭脉冲可持续的最长时间（秒）。如果计算出的 OpenTime 或 CloseTime 大于该值，将被限定为该值。如果该值无效，该指令会假定该值等于 CycleTime 并将 Status 中的相应位置位。 有效值 = 0.0 到 CycleTime 默认值 = CycleTime
MinOnTime	REAL	打开或关闭脉冲可持续的最短时间（秒）。如果计算出的 OpenTime 或 CloseTime 小于该值，将设置为零。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 0.0 到 MaxOnTime 默认值 = 0.0

Deadtime	REAL	用来克服设备摩擦力的附加脉冲时间（秒）。当设备改变方向或停止时，死区时间将与 OpenTime 或 CloseTime 值相加。如果该值无效，指令会将 Status 中的相应位置位，并设置 Deadtime = 0.0 有效值 = 0.0 至 MaxOnTime 默认值 = 0.0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 PositionPercent 溢出，该输出将清零。
OpenOut	BOOL	该输出产生用于打开设备的脉冲。
CloseOut	BOOL	该输出产生用于关闭设备的脉冲。
PositionPercent	REAL	位置反馈以 Position 范围的百分比形式表示。
SPPercent	REAL	设置点以 Position 范围的百分比形式表示。
OpenTime	REAL	当前循环中 OpenOutput 的脉冲时间（秒）。
CloseTime	REAL	当前周期的 CloseOutput 脉冲时间，单位为秒。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
CycleTimeInv (Status.1)	BOOL	CycleTime 值无效。指令将使用零值。
OpenRateInv (Status.2)	BOOL	OpenRate 值无效。指令将使用零值。
CloseRateInv (Status.3)	BOOL	CloseRate 值无效。指令将使用零值。
MaxOnTimeInv (Status.4)	BOOL	MaxOnTime 值无效。指令将使用 CycleTime 值。
MinOnTimeInv (Status.5)	BOOL	MinOnTime 值无效。指令将使用零值。
DeadtimeInv (Status.6)	BOOL	Deadtime 值无效。指令将使用零值。
PositionPctInv (Status.7)	BOOL	计算得出的 PositionPercent 值超出范围。
SPPercentInv (Status.8)	BOOL	计算得出的 SPPercent 值超出范围。

PositionSpanInv (Status.9)	BOOL	PositionEUMax = PositionEUMin。

说明

POSP 指令通常从 PID 指令输出接收所需的位置设置点。

标定位置和设置点的值。

每次执行指令时，PositionPercent 和 SPPercent 输出都会更新。如果其中任何一个值超出范围（小于 0% 或大于 100%），则将 Status 中的相应位置位，但不会限制这些值。该指令使用以下公式计算这些值是否处于范围内：

$$PositionPercent = \frac{Position - PositionEUMin}{PositionEUMax - PositionEUMin} \times 100$$

$$SPPercent = \frac{SP - PositionEUMin}{PositionEUMax - PositionEUMin} \times 100$$

POSP 指令如何使用内部周期计时器

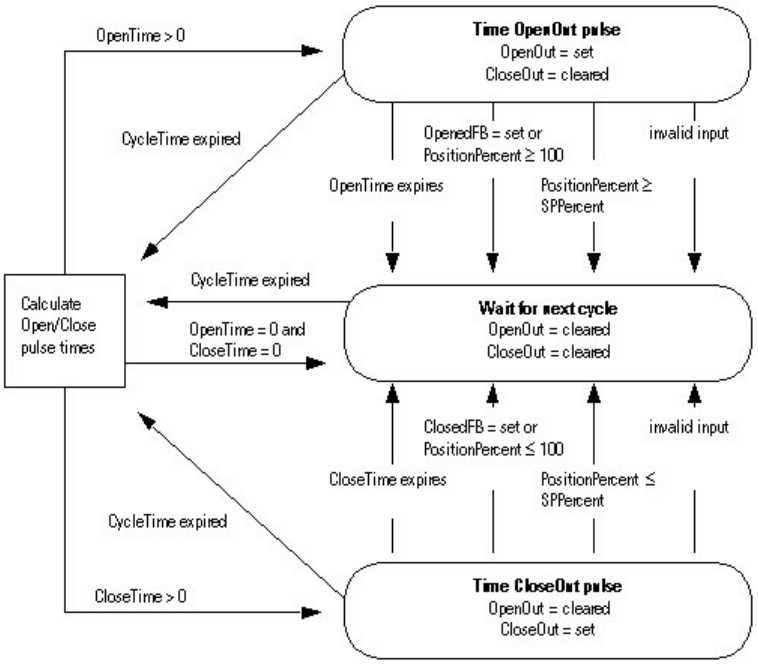
该指令使用 CycleTime 确定重新计算打开和关闭输出脉冲持续时间的频率。内部计时器通过 DeltaT 保持并更新。DeltaT 是自该指令上次执行后所经过的时间。只要内部计时器等于或超过设定的 CycleTime（周期时间到期），便会重新计算打开和关闭输出。

用户可随时更改 CycleTime。

如果 CycleTime = 0，内部计时器清零，OpenOut 和 CloseOut 设置为假。

生成输出脉冲

下图说明了 POSP 指令的三种主要状态。



计算打开和关闭脉冲时间

只要 SP > 位置反馈，就会生成 OpenOut 脉冲。此时，该指令将设置 CloseTime = 0，OpenOut 接通的持续时间计算如下：

$$OpenTime = \frac{SPPercent - PositionPercent}{OpenRate}$$

如果 OpenTimen-1 < CycleTime，则向 OpenTime 加上 Deadtime。

如果 OpenTime > MaxOnTime，则其值限制为 MaxOnTime。

如果 OpenTime < MinOnTime，则设置 OpenTime = 0。

如果满足以下任一条件，将不生成 OpenOut 脉冲且 OpenTime = 0。

OpenFB 为真或 PositionPercent ≥ 100

CycleTime = 0

OpenRate = 0

SPPercent 无效

只要 SP < 位置反馈，就会生成 CloseOut 脉冲。此时，该指令会设置 OpenTime = 0，CloseOut 接通的持续时间计算如下：

$$CloseTime = \frac{PositionPercent - SPPercent}{CloseRate}$$

如果 CloseTimen-1 < CycleTime，则向 CloseTime 加上 Deadtime。

如果 CloseTime > MaxOnTime，则其值限制为 MaxOnTime。

如果 CloseTime < MinOnTime，则将 CloseTime 设置为 0。

如果满足以下任一条件，将不会生成 CloseOut 脉冲，CloseTime 将清零 (0.0)。

ClosedFB 为真或 PositionPercent ≤ 0

CycleTime = 0

CloseRate = 0

SPPercent 无效

如果 SPPercent 等于 PositionPercent，将不生成 OpenOut 和 CloseOut 脉冲。OpenTime 和 CloseTime 都会设置为假。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 该指令执行。

指令首次运行	不适用
指令首次扫描	OpenTime 和 CloseTime 清零 (0.0)。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

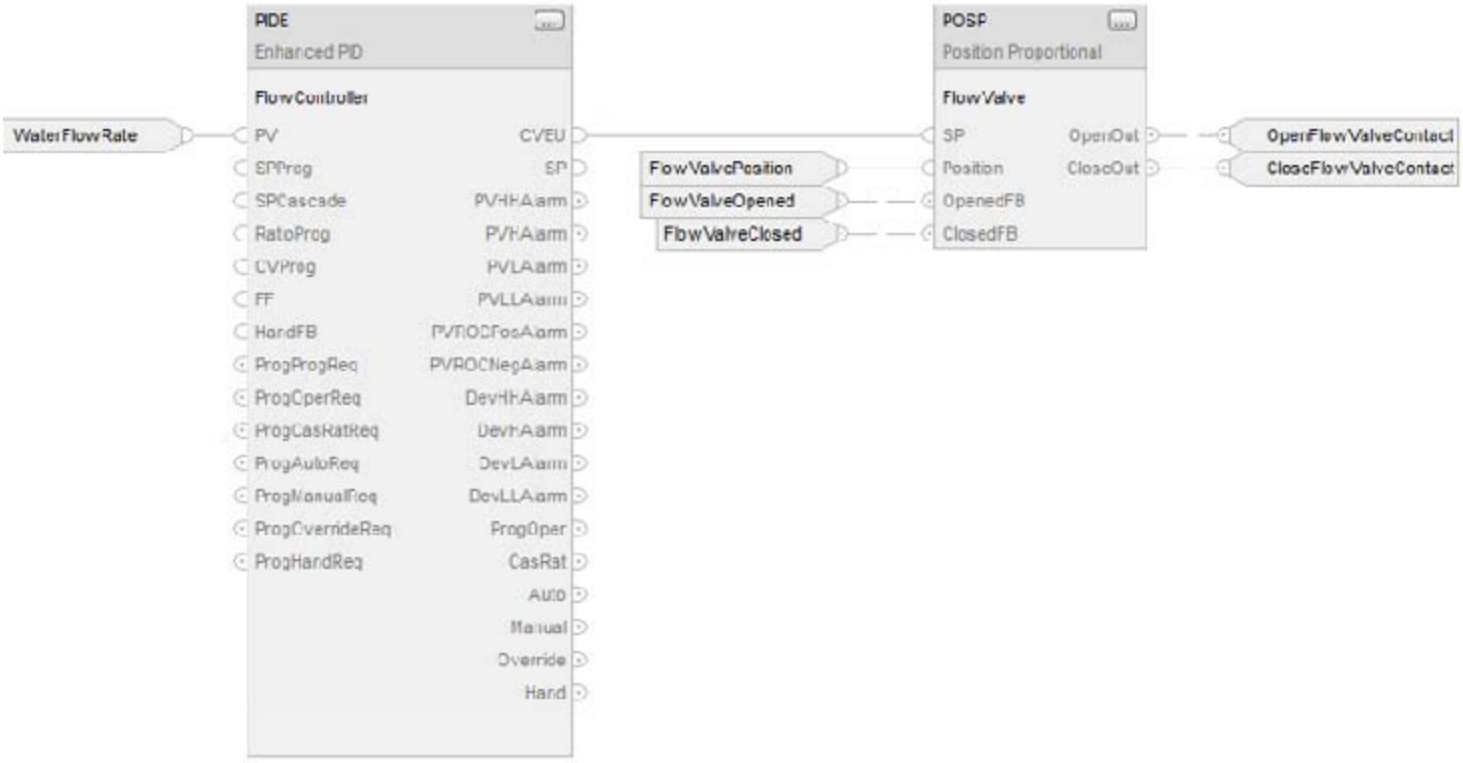
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

示例 1

本示例中，POSP 指令可根据 PIDE 指令的 CVEU 输出打开或关闭电动阀。阀门实际位置与 Position 输入相连，用于指示阀门是否完全打开或关闭的可选限位开关与 OpenedFB 和 ClosedFB 输入相连。OpenOut 和 CloseOut 输出连接到电动阀的打开触点和关闭触点。

功能块



结构化文本

```
FlowController.PV := WaterFlowRate;
PIDE(FlowController);

FlowValve.SP := FlowController.CVEU;
FlowValve.Position := FlowValvePosition;
FlowValve.OpenedFB := FlowValveOpened;
FlowValve.ClosedFB := FlowValveClosed;
POSP(FlowValve);

OpenFlowValveContact := FlowValve.OpenOut;
CloseFlowValveContact := FlowValve.CloseOut;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

升温/持温 (RMPS)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

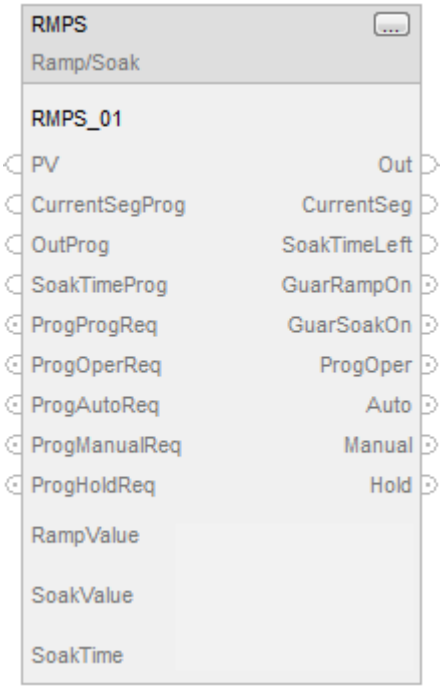
RMPS 指令提供一系列交替出现升温和持温周期的分段。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

RMPS(RMPS_tag,RampValue,SoakValue,SoakTime);

操作数

功能块

操作数	类型	格式	说明
RMPS tag	RAMP_SOAK	结构	RMPS 结构
RampValue	REAL	数组	升温值数组。输入每个分段(0 到 NumberOfSegs-1)的升温值。升温值可采用时间 (分钟) 或速率 (单位/分钟) 两种形式输入。TimeRate 参数指示升温值的指定方式。如果升温值无效，指令会将 Status 中的相应位置位 ,并切换到"操作员手动"或"程序保持"模式。该数组的大小至少应等于 NumberOfSegs。 有效值 = 0.0 到最大正浮点值
SoakValue	REAL	数组	持温值数组。输入每个分段(0 到 NumberOfSegs-1)的持温值。该数组的大小至少应等于 NumberOfSegs。 有效值 = 任意浮点值
SoakTime	REAL	数组	持温时间数组。输入每个分段(0 到 NumberOfSegs-1) 的持温时间。持温时间以分钟为单位输入。如果持温值无效，指令会将 Status 中的相应位置位 ,并切换到"操作员手动"或"程序保持"模式。该数组的大小至少应等于 NumberOfSegs。 有效值 = 0.0 到最大正浮点值

结构化文本

操作数	类型	格式	说明
RMPS tag	RAMP_SOAK	结构	RMPS 结构

RampValue	REAL	数组	升温值数组。输入每个分段(0 到 NumberOfSegs-1)的升温值。升温值可采用时间 (分钟) 或速率 (单位/分钟) 两种形式输入。TimeRate 参数指示升温值的指定方式。如果升温值无效，指令会将 Status 中的相应位置位 ,并切换到"操作员手动"或"程序保持"模式。该数组的大小至少应等于 NumberOfSegs。 有效值 = 0.0 到最大正浮点值
SoakValue	REAL	数组	持温值数组。输入每个分段(0 到 NumberOfSegs-1)的持温值。该数组的大小至少应等于 NumberOfSegs。 有效值 = 任意浮点值
SoakTime	REAL	数组	持温时间数组。输入每个分段(0 到 NumberOfSegs-1) 的持温时间。持温时间以分钟为单位输入。如果持温值无效，指令会将 Status 中的相应位置位 ,并切换到"操作员手动"或"程序保持"模式。该数组的大小至少应等于 NumberOfSegs。 有效值 = 0.0 到最大正浮点值

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

RMPS 结构

指定各指令的唯一 RMPS 结构。

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
PV	REAL	指令的模拟温度信号输入标定值。 有效值 = 任意浮点值 默认值 = 0.0

PVFault	BOOL	PV 状况不良指示器。如果该值为真，指令将进入“程序保持”模式或“操作员手动”模式，并将 Status 中的相应位置位。 默认值为假。
NumberOfSegs	DINT	分段数。指定指令使用的升温/持温分段数。RampValue、SoakValue 和 SoakTime 数组的大小至少应等于 NumberOfSegs。如果该值无效，指令将进入“操作员手动”或“程序保持”模式，并将 Status 中的相应位置位。 有效值 = 1 到（RampValue、SoakValue 或 SoakTime 数组中的最小数组大小） 默认值 = 1
ManHoldAftInit	BOOL	初始化后进入手动/保持模式。如果该值为真，升温/持温过程在初始化完成后将进入“操作员手动”或“程序保持”模式。否则，升温/持温过程在初始化完成后将保持其先前模式。 默认值为假。
CyclicSingle	BOOL	循环/单次执行。真值表示循环操作，假值表示单次操作。循环操作模式会不断重复升温/持温曲线。单次操作模式在执行一次升温/持温曲线后即停止。 默认值为假。
TimeRate	BOOL	以时间/速率配置升温值。如果以达到持温温度的时间（分钟）输入 RampValue 参数值，则该值为真。如果以速率（单位/分钟）输入 RampValue 参数值，则该值为假。 默认值为假。
GuarRamp	BOOL	保证升温。如果该值为真且指令处于自动模式，则当 PV 与输出值之差大于 RampDeadband 时，暂停升温。 默认清零。
RampDeadband	REAL	保证升温的死区值。指定在启用 GuarRamp 的情况下 PV 与输出值之间允许的差值（工程单位）。如果该值无效，指令将设置 RampDeadband = 0.0 并将 Status 中的相应位置位。 有效值 = 任意 ≥ 0.0 的浮点值 默认值 = 0.0

GuarSoak	BOOL	保证持温。如果该值为真且指令处于自动模式，则当 PV 与输出值之差大于 SoakDeadband 时，持温计时器清零。 默认值为假。
SoakDeadband	REAL	保证持温的死区值。指定在启用 GuarSoak 的情况下 PV 与输出值之间允许的差值（工程单位）。如果该值无效，指令将设置 SoakDeadband = 0.0 并将 Status 中的相应位置位。 有效值 = 任意 ≥ 0.0 的浮点值 默认值 = 0.0
CurrentSegProg	DINT	程序控制模式下的当前分段。用户程序将请求的 CurrentSeg 值写入此输入。当升温/持温过程处于“程序手动”模式时，使用该值。如果该值无效，指令将 Status 中的相应位置位。 有效值 = 0 到 NumberOfSegs-1 默认值 = 0
OutProg	REAL	程序控制模式下的输出。用户程序将请求的 Out 值写入此输入。当升温/持温过程处于“程序手动”模式时，将使用该值作为 Out 值。 有效值 = 任意浮点值 默认值 = 0.0
SoakTimeProg	REAL	程序控制模式下的持温时间。用户程序将请求的 SoakTimeLeft 值写入此输入。当升温/持温过程处于“程序手动”模式时，使用该值。如果该值无效，指令将 Status 中的相应位置位。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CurrentSegOper	DINT	操作员控制模式下的当前分段。操作员界面将请求的 CurrentSeg 值写入此输入。当升温/持温过程处于“操作员手动”模式时，使用该值。如果该值无效，指令将 Status 中的相应位置位。 有效值 = 0 到 NumberOfSegs-1 默认值 = 0
OutOper	REAL	操作员控制模式下的输出。操作员界面将请求的 Out 值写入此输入。当升温/持温过程处于“操作员手动”模式时，将使用该值作为 Out 值。 有效值 = 任意浮点值 默认值 = 0.0

SoakTimeOper	REAL	操作员控制模式下的持温时间。操作员界面将请求的 SoakTimeLeft 值写入此输入。当升温/持温过程处于"操作员手动"模式时，使用该值。如果该值无效，指令将 Status 中的相应位置位。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
ProgProgReq	BOOL	程序发出的程序控制请求。由用户程序设置为真可请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。保持该参数为真且保持 ProgOperReq 为假可将指令锁定为程序控制模式。 默认值为假。
ProgOperReq	BOOL	程序发出的操作员控制请求。由用户程序设置为真可请求操作员控制模式。保持该参数为真可将指令锁为操作员控制模式。 默认值为假。
ProgAutoReq	BOOL	程序自动模式请求。由用户程序设置为真可请求升温/持温过程进入自动模式。如果回路处于操作员控制模式、ProgManualReq 为真或者 ProgHoldReq 为真，则忽略该值。 默认值为假。
ProgManualReq	BOOL	程序手动模式请求。由用户程序设置为真可请求升温/持温过程进入手动模式。如果升温/持温过程处于操作员控制模式或者 ProgHoldReq 为真，则忽略该值。 默认值为假。
ProgHoldReq	BOOL	程序保持模式请求。由用户程序设置为真可请求停止升温/持温，同时不改变 Out、CurrentSeg 或 SoakTimeLeft 值。当从升温/持温获取设置点的 PID 回路退出级联时，该值也非常有用。操作员也可以通过将升温/持温设置为操作员手动模式来达到相同效果。 默认值为假。
OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。指令会将此输入设置为假。 默认值为假。

OperOperReq	BOOL	操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。如果 ProgProgReq 为真且 ProgOperReq 为假，则忽略该值。指令会将此输入设置为假。 默认值为假。
OperAutoReq	BOOL	操作员自动模式请求。由操作员界面设置为真可请求升温/持温过程进入自动模式。如果回路处于程序控制模式或 OperManualReq 为真 ,则忽略该值。指令会将此输入设置为假。 默认值为假。
OperManualReq	BOOL	操作员手动模式请求。由操作员界面设置为真可请求升温/持温过程进入手动模式。如果回路处于程序控制模式，则忽略该值。指令会将此输入设置为假。 默认值为假。
Initialize	BOOL	初始化程序和操作员值。如果该值为真且处于手动模式，则指令会设置 CurrentSegProg = 0、CurrentSegOper = 0、 SoakTimeProg = SoakTime[0] 以及 SoakTimeOper = SoakTime[0]。处于自动或保持模式时，将忽略 Initialize 值。指令将此参数设置为假。 默认值为假。
ProgValueReset	BOOL	将程序控制值复位。如果该值为真，则指令会将 ProgProgReq、ProgOperReq、ProgAutoReq、ProgHoldReq 和 ProgManualReq 设置为假。 默认值为假。

输出参数	**数据类型**	**说明**
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	升温/持温指令的输出。
CurrentSeg	DINT	当前分段号。显示升温/持温循环的当前分段号。分段编号从 0 开始。
SoakTimeLeft	REAL	剩余持温时间。显示当前持温时间。
GuarRampOn	BOOL	保证升温状态。如果正在使用保证升温功能，并且升温因 PV 与输出值之差大于 RampDeadband 而暂停，则该值设置为真。

GuarSoakOn	BOOL	保证持温状态。如果正在使用保证持温功能，并且持温计时器因 PV 与输出值之差大于 SoakDeadband 而清零，则该值设置为真。
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。操作员控制模式下为假。
Auto	BOOL	自动模式。当升温/持温处于“程序自动”模式或“操作员自动”模式时为真。
Manual	BOOL	手动模式。当升温/持温处于“程序手动”模式或“操作员手动”模式时为真。
保持 (Hold)	BOOL	保持模式。当升温/持温处于程序“保持”模式时为真。
状态 (Status)	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
PVFaulted (Status.1)	BOOL	PV 状况不良。
NumberOfSegsInv (Status.2)	BOOL	NumberOfSegs 值无效，或与数组大小不兼容。
RampDeadbandInv (Status.3)	BOOL	RampDeadband 值无效。
SoakDeadbandInv (Status.4)	BOOL	SoakDeadband 值无效。
CurrSegProgInv (Status.5)	BOOL	CurrSegProg 值无效。
SoakTimeProgInv (Status.6)	BOOL	SoakTimeProg 值无效。
CurrSegOperInv (Status.7)	BOOL	CurrSegOper 值无效。
SoakTimeOperInv (Status.8)	BOOL	SoakTimeOper 值无效。
RampValueInv (Status.9)	BOOL	RampValue 值无效。
SoakTimeInv (Status.10)	BOOL	SoakTime 值无效。

说明

RMPS 指令通常用于在批量加热过程中提供温度曲线。该指令的输出通常作为 PID 回路设置点的输入。

一旦计算出的输出值无效（NAN 或 ± INF），指令会设置 Out = 无效值，并将数学溢出状态标志置位。内部参数不会更新。在每次的后续扫描过程中，都会使用上一次扫描（输出有效）的内部参数计算输出。

监视 RMPS 指令

RMPS 指令有相应的操作员面板。

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	将 CurrentSeg 清零。 将模式设置为操作员手动模式。 如果 SoakTime[0] 有效，则将 SoakTimeProg 和 SoakTimeOper 设置为 SoakTime[0]。
指令首次扫描	将所有操作员请求输入清零。 如果 ProgValueReset 为真，则将所有程序请求输入清零。 如果当前处于保持模式，则将操作员控制模式设置为手动模式。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

指令首次扫描后的初始模式

下表所示为基于程序请求输入所获得的最终控制模式。

首次扫描开始时的控制模式	ProgOperReq	ProgProgReq	ProgValueReset	FirstRun	首次扫描结束时的控制模式
操作员控制	假	真	假	不适用	程序控制
	不适用	假	不适用	不适用	操作员控制
程序控制	真	不适用	假	假	操作员控制
	不适用	不适用	真	真	操作员控制
	假	假	假	真	操作员控制
	假	真	假	不适用	程序控制
	不适用	不适用	真	假	程序控制
	假	假	假	假	程序控制

下表所示为基于“手动”、“自动”和“保持”模式请求所获得的最终控制模式。

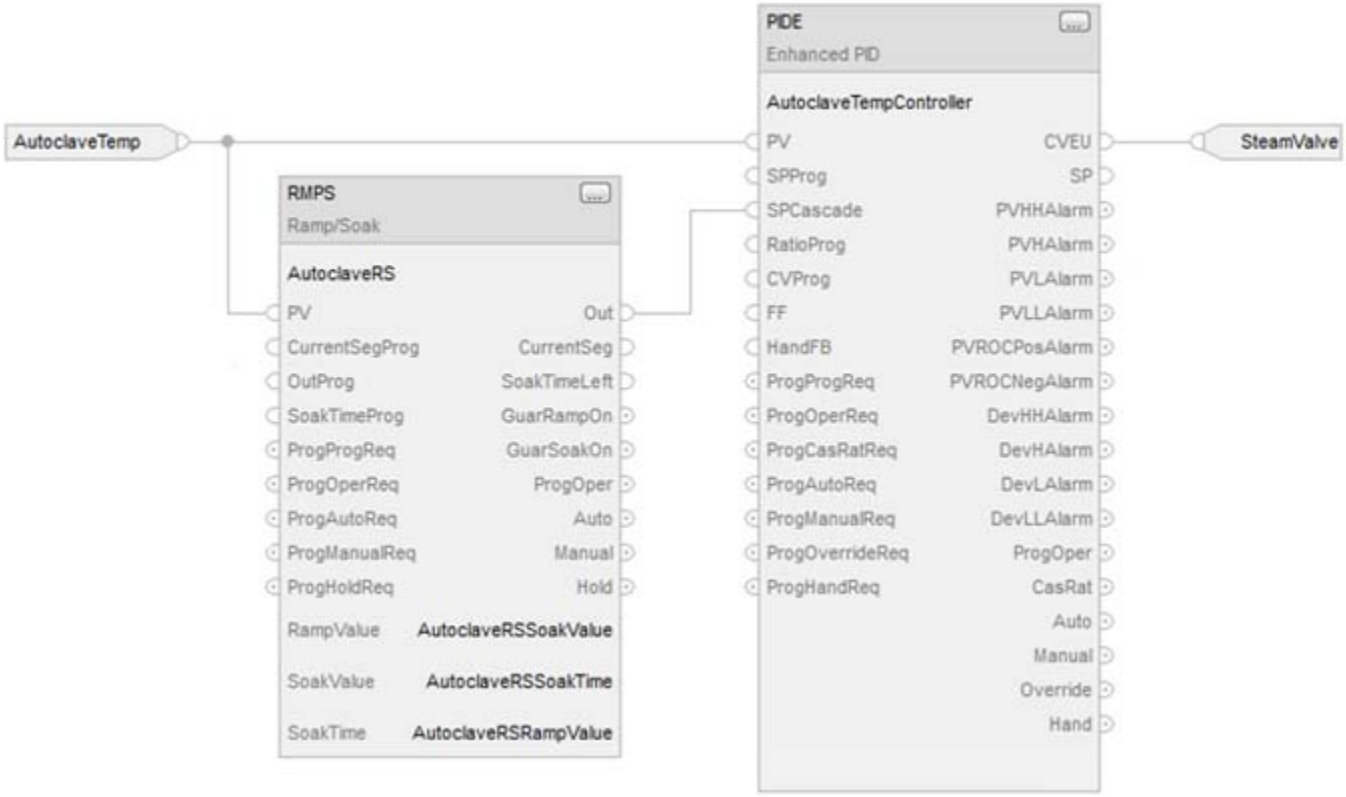
首次扫描开始时的控制模式	Oper Auto Req	Oper Man Req	Prog Auto Req	Prog Man Req	Prog Hold Req	Man Hold AftInit	Prog Value Reset	First Run	首次扫描结束时的控制模式
操作员控制	不适用	不适用	不适用	不适用	不适用	假	不适用	假	操作员当前模式
	不适用	不适用	不适用	不适用	不适用	不适用	不适用		操作员手动模式
	不适用	不适用	不适用	不适用	不适用	真	不适用	不适用	
程序控制	不适用	不适用	假	假	假	假	不适用	假	程序当前模式
	不适用	不适用	不适用	不适用	不适用	假	真	假	
	不适用	不适用	真	假	假	假	假	不适用	程序自动模式
	不适用	不适用	不适用	真	假	假	假	不适用	程序手动模式
	不适用	不适用	不适用	不适用	真	假	假	不适用	程序保持模式
	不适用	不适用	不适用	不适用	不适用	真	不适用	不适用	

示例

在此示例中，RMPS 指令将驱动 PIDE 指令的设置点。当 PIDE 指令处于“级联/比率”模式时，使用 RMPS 指令的输出作为设置点。如果要使用保证升温和/或保证持温功能，可选择将 PIDE 指令的 PV 送入 RMPS 指令的 PV 输入。

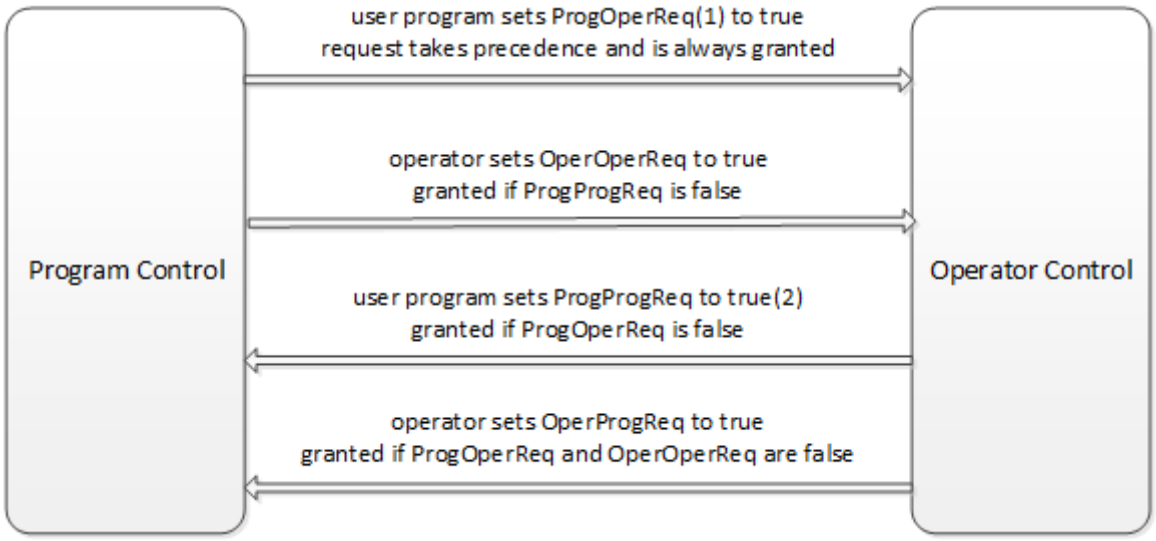
在此示例中，AutoclaveRSSoakValue、AutoclaveRSSoakTime 和 AutoclaveRSRampValue 数组均为包含 10 个元素的 REAL 数组，最多允许使用 10 段 RMPS 曲线。

功能块



在程序控制与操作员控制之间切换

RMPS 指令可以通过用户程序或操作员界面进行控制。可随时更改控制模式。



(1) ProgOperReq 保持为真时，指令锁定为操作员控制模式。

(2) ProgProgReq 保持为真且 ProgOperReq 为假时，指令锁定为程序控制模式。

当 ProgAutoReq、ProgManualReq 和 ProgHoldReq 输入均为假时，如果从操作员控制转换到程序控制模式，将按如下规则确定模式：

如果转换前指令处于“操作员自动”模式，则转换为“程序自动”模式。

如果转换前指令处于“操作员手动”模式，则转换为“程序手动”模式。

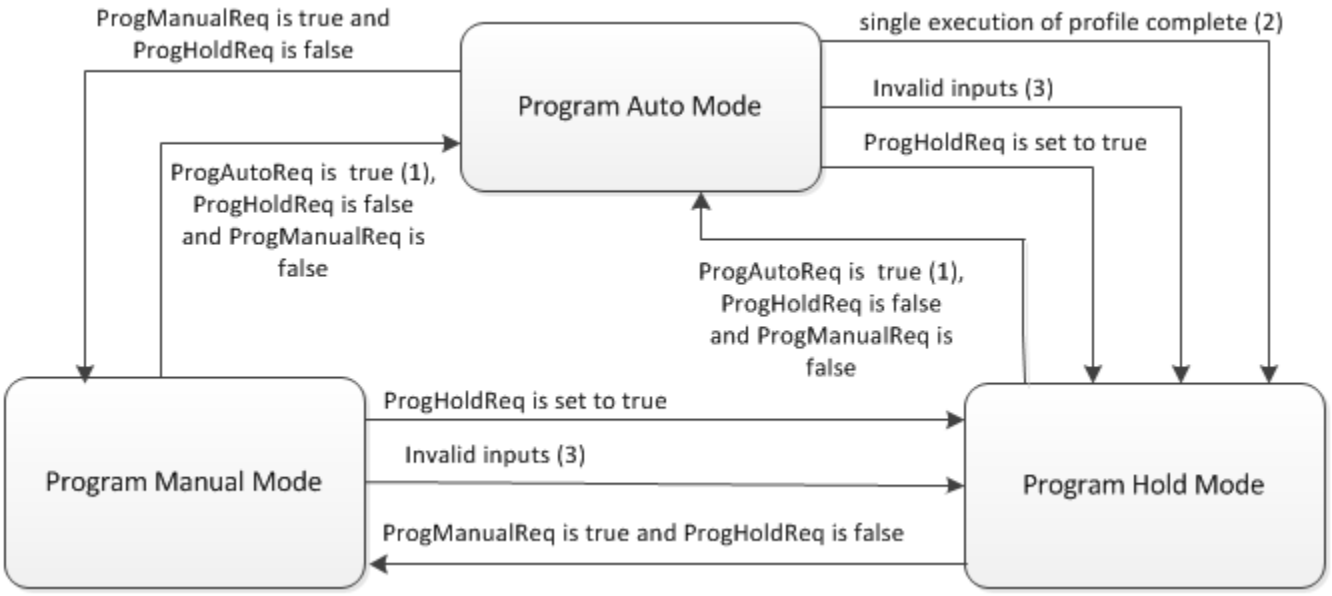
当 OperAutoReq 和 OperManualReq 输入均为假时，如果从程序控制转换到操作员控制模式，将按如下规则确定模式：

如果转换前指令处于“程序自动”模式，则转换为“操作员自动”模式。

如果转换前指令处于“程序手动”或“程序保持”模式，则转换为“操作员手动”模式。

程序控制

下图所示为 RMPS 指令在程序控制模式下的工作情况。



(1) 在单次（非循环）执行过程中，如果要在完成一次升温/持温曲线后再次执行升温/持温曲线，必须将 ProgAutoReq 从假切换为真。

(2) 如果将指令配置为单次执行，则在自动模式下完成升温/持温曲线时，指令将转换为保持模式。

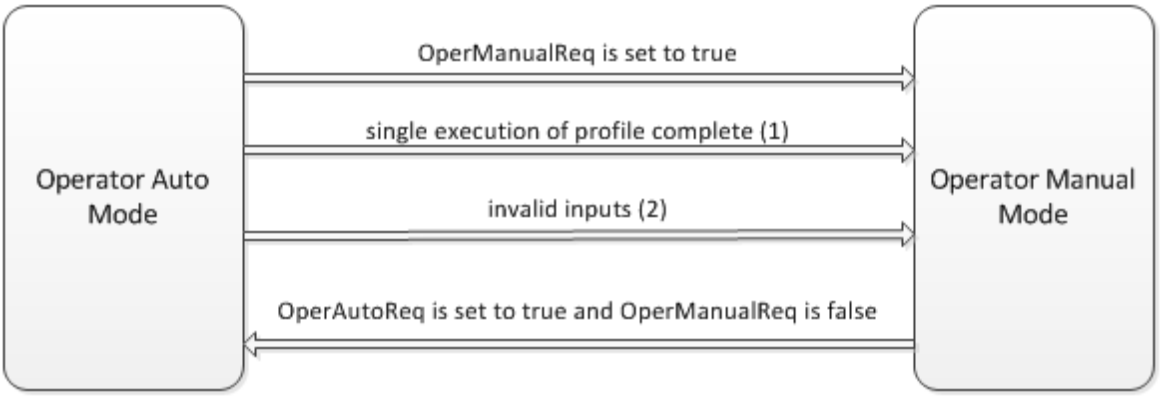
(3) 如果 PVFaulted 为真或以下任一输入无效，指令将进入保持模式：NumberOfSegs、CurrentSegProg 或 SoakTimeProg。

下表介绍各种可能的程序控制模式。

模式	说明
程序自动模式	在自动模式下，指令将按顺序执行升温/持温曲线。
程序手动模式	在手动模式下，由用户程序直接控制指令的 Out 输出。CurrentSegProg、SoakTimeProg 和 OutProg 输入将传送到 CurrentSeg、SoakTimeLeft 和 Out 输出。当指令进入自动模式时，升温/持温函数继续使用上次由用户程序输入的值。如果 CurrentSegProg 和 SoakTimeProg 无效，则不会传输到相应输出。 当 ProgValueReset 置位且指令不处于“程序手动”模式时，为“无扰动地”转换到手动模式，CurrentSegProg、SoakTimeProg 和 OutProg 输入会持续更新为当前的 CurrentSeg、SoakTimeLeft 和 Out 值。
程序保持模式	在保持模式下，指令的输出保持当前值不变。如果在此模式下通过将 ProgOperReq 置位来切换到操作员控制模式，指令将进入“操作员手动”模式。

操作员控制

下图所示为 RMPS 指令在操作员控制模式下的工作情况。



(1) 如果将指令配置为单次执行，则在自动模式下完成升温/持温曲线时，指令将转换为手动模式。

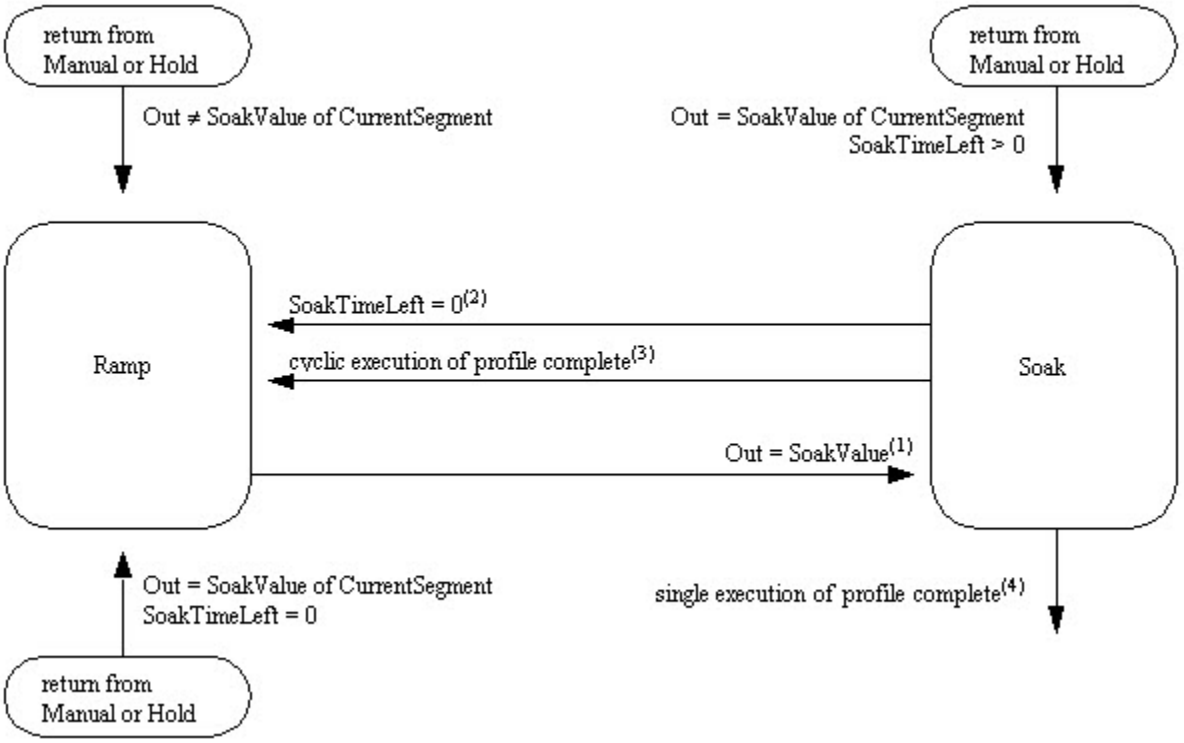
(2) 如果 PVFaulted 为真或以下任一输入无效，指令将进入手动模式：NumberOfSegs、CurrentSegOper 或 SoakTimeOper。

下表介绍各种可能的操作员控制模式。

模式	说明
操作员自动模式	在自动模式下，指令将按顺序执行升温/持温曲线。
操作员手动模式	在手动模式下，由操作员直接控制指令的 Out 输出。CurrentSegOper、SoakTimeOper 和 OutOper 输入将传送到 CurrentSeg、SoakTimeLeft 和 Out 输出。当指令进入自动模式时，升温/持温函数继续使用上次由操作员输入的值。如果 CurrentSegOper 和 SoakTimeOper 无效，则不会传输到相应输出。 当指令不处于"操作员手动"模式时，为"无扰动地"转换到手动模式，CurrentSegOper、SoakTimeOper 和 OutOper 输入会持续更新为当前的 CurrentSeg、SoakTimeLeft 和 Out 值。

执行升温/持温曲线

下图说明 RMPS 指令执行升温/持温曲线的过程。



(1) Out = SoakValue 时升温过程结束。如果在升温期间 Out > SoakValue，则将 Out 限定为 SoakValue。

(2) 当 Out 保持不变的时间达到当前分段的 SoakTime 所指定的时长时，持温过程结束。如果执行的分段并非最后一个分段，则 CurrentSeg 增加一。

(3) 最后一个设定分段的持温过程结束，指令配置为循环执行。指令设置 CurrentSeg = 0。

(4) 最后一个设定分段的持温过程结束，指令配置为单次执行。

(5) 返回自动模式时，指令将决定是否继续执行升温或持温。下一步动作取决于当前分段的 Out、SoakTimeLeft 和 SoakValue 值。如果当前分段的 Out=SoakValue 且 SoakTimeLeft =0，则表明当前分段完成，下一分段开始。

升温

在升温循环中，Out 将由前一分段的 SoakValue 值升温至当前分段的 SoakValue 值。升温经过的时间由 RampValue 参数定义。

如果 Out 值 < 当前分段的目标 SoakValue 值，则升温过程为正向。如果根据升温公式计算出的新 Out 值超过目标 SoakValue 值，则 Out 值将设置为目标 SoakValue 值。

如果 Out 值 > 当前分段的目标 SoakValue 值，则升温过程为负向。如果根据升温公式计算出的新 Out 值小于目标 SoakValue 值，则 Out 值将设置为目标 SoakValue 值。

每个分段都有相应的升温值。可选择以时间或速率为单位设定升温值。所有分段必须设定为相同的单位。下表介绍各个升温选项：

参数	说明
基于时间的升温	TimeRate 设置为真时，采用基于时间的升温（分钟） 将计算当前分段的变化率，并在 Out 值基础上相加或相减，直至 Out 值达到当前分段的持温值。在下面的公式中，DeltaT 是指自指令最后一次执行起经过的时间（分钟）。 $Out = Out \pm \frac{(SoakValue_{CurrentSeg} - RampStart)}{RampValue_{CurrentSeg}} \times \Delta t$ 其中，RampStart 是当前分段开始时的 Out 值。
基于速率的升温	TimeRate 清零时，采用基于速率的升温（单位/分钟） 在 Out 值的基础上加上或减去设定的变化率，直至 Out 值达到当前分段的持温值。在下面的公式中，DeltaT 是指自指令最后一次执行起经过的时间（分钟）。 $Out = Out \pm RampValue_{CurrentSeg} \times \Delta t$

保证升温

将 GuarRamp 输入设置为真时，启用保证升温。启用后，指令将监视 Out 与 PV 之间的差值。如果该差值超过设定的 RampDeadband 值，则输出保持不变，直至 PV 与 Out 的差值进入死区范围。当 Out 值由于保证升温的作用而保持不变时，输出 GuarRampOn 将设置为真。

持温

持温过程是指在下一个升温-持温分段开始前块输出保持不变的时间段。在持温循环期间，输出会在进入下一分段之前的设定时间内保持为 SoakValue 值。持温期间的输出保持时间在 SoakTime 参数中设定。

每个分段都有对应的 SoakValue 和 SoakTime 值。当 Out 升温至当前分段的 SoakValue 值时，持温过程开始。SoakTimeLeft 值表示剩余的输出保持时间（分钟）。在升温期间，SoakTimeLeft 将设置为当前分段的 SoakTime 值。升温过程完成后，SoakTimeLeft 值开始减小，以反映当前分段的剩余时间（分钟）。经过 SoakTime 时间后，SoakTimeLeft = 0。

保证持温

将 GuarSoak 输入设置为真时，启用保证持温。启用后，指令将监视 Out 与 PV 之间的差值。如果该差值超过 SoakDeadband 值，将暂停持温循环的计时，同时将内部持温计时器清零。当 Out 与 PV 之间的差值重新进入死区范围内时，继续计时。当计时由于保证持温的作用而暂停时，输出 GuarSoakOn 将设置为真。

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

标定 (SCL)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

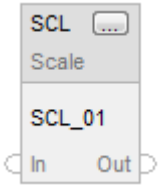
SCL 指令用于将未标定的输入值转换为采用工程单位的浮点值。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

SCL(SCL_tag)

操作数

功能块

操作数	类型	格式	说明
SCL tag	SCALE	结构	SCL 结构

结构化文本

操作数	类型	格式	说明
SCL tag	SCALE	结构	SCL 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

SCALE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
InRawMax	REAL	指令的输入可达到的最大值。如果 InRawMax ≤ InRawMin，则该指令会将 Status 中的相应位置位，并停止更新输出。 有效值 = InRawMax > InRawMin 默认值 = 0.0
InRawMin	REAL	指令的输入可达到的最小值。如果 InRawMin ≥ InRawMax，则指令会将 Status 中的相应位置位，并停止更新输出。 有效值 = InRawMin < InRawMax 默认值 = 0.0

InEUMax	REAL	对应于 InRawMax 的输入标定值。 有效值 = 任意实数值 默认值 = 0.0
InEUMin	REAL	对应于 InRawMin 的输入标定值。 有效值 = 任意实数值 默认值 = 0.0
限制	BOOL	限制选择器。如果为真，会将 Out 限制在 InEUMin 与 InEUMax 之间。 默认值为假。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	表示模拟量输入标定值的输出。 有效值 = 任意实数值 默认值 = InEUMin
MaxAlarm	BOOL	超出最大输入报警指示器。当 In> InRawMax 时，该值设置为真。
MinAlarm	BOOL	低于最小输入报警指示器。当 In < InRawMin 时，该值设置为真。
状态 (Status)	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InRawRangeInv (Status.1)	BOOL	InRawMin ≥ InRawMax。

说明

SCL 指令用于不支持标定为全分辨率浮点值的模拟量输入模块。

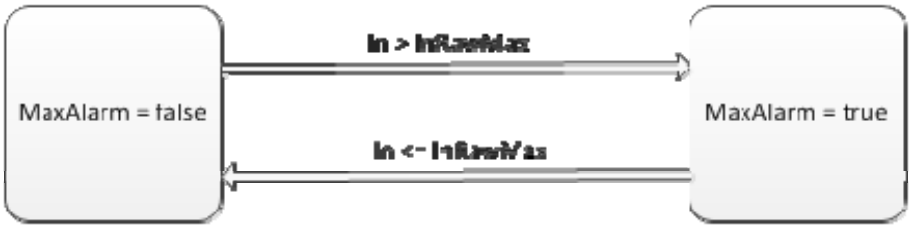
例如，1771-IFE 模块是一个仅支持以整数值进行标定的 12 位模拟量输入模块。如果使用 1771-IFE 模块读取 0-100 加仑/分钟 (gpm) 的流量，通常不会将模块标定为 0-100，否则会限制该模块的分辨率。相反，应使用 SCL 指令并将模块配置为返回未标定的 (0-4095) 值，其中 SCL 指令会将值转换为 0-100 gpm（浮点值）而不会损失分辨率。然后，此标定后的值可用作其他指令的输入。

SCL 指令采用以下算法将未标定的输入转换为标定值：

$$Out = (In - InRawMin) \times \left(\frac{InEUMax - InEUMin}{InRawMax - InRawMin}\right) + InEUMin$$

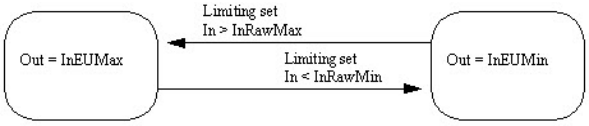
报警

指令计算出 Out 后，将按照以下方式确定 MaxAlarm 和 MinAlarm 的值：



限制

将 Limiting 置位后，会对 Out 进行限制。当 In > InRawMax 时，该指令会设置 Out = InEUMax。当 In < InRawMin 时，该指令会设置 Out = InEUMin。



影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。

Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真 指令执行。
指令首次运行	不适用
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

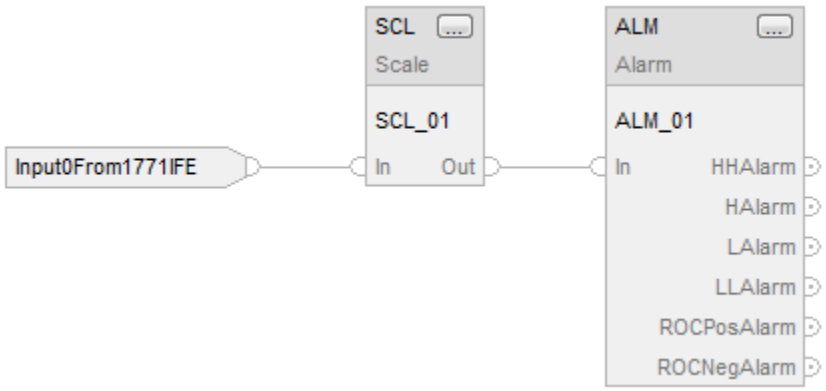
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

SCL 指令通常用于不支持板载标定为浮点工程单位的模拟量输入模块。在此示例中，SCL 指令对来自 1771-IFE 模块的模拟量输入进行标定。该指令将结果存储在 Out 中，供 ALM 指令使用。

功能块



结构化文本

```
SCL_01.In := Input0From1771IFE;
SCL(SCL_01);

ALM_01.In := SCL_01.Out;
ALM(ALM_01);
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

分程时间比例 (SRTP)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

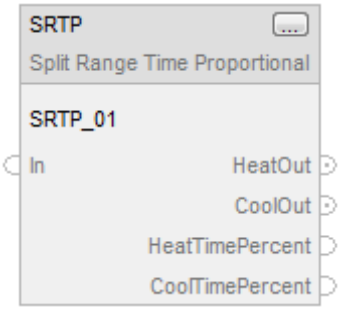
SRTP 指令采用 PID 回路的 0-100% 输出，并通过周期脉冲驱动加热和冷却数字量输出触点。例如，控制挤出机的桶温便是该指令的其中一种应用。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

SRTP(SRTP_tag)

操作数

功能块

操作数	类型	格式	说明
SRTP tag	SPLIT_RANGE	结构	SRTP 结构

结构化文本

操作数	类型	格式	说明
SRTP tag	SPLIT_RANGE	结构	SRTP 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

SPLIT_RANGE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	请求加热或冷却的模拟信号输入。该输入通常来自 PID 回路的 CVEU。 有效值 = 任意浮点值
CycleTime	REAL	输出脉冲周期，单位为秒。值为零时，将关闭加热和冷却输出。如果该值无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 任意正浮点值 默认值= 0.0
MaxHeatIn	REAL	最大加热输入。该值用于指定将引起最大加热量的 In 的百分比。对于加热/冷却回路，该值通常为 100%。 有效值 = 任意浮点值 默认值 = 100.0
MinHeatIn	REAL	最小加热输入。指定用于表示加热范围起始值并引起最小加热量的 In 的百分比。对于加热/冷却回路，该值通常为 50%。 有效值 = 任意浮点值 默认值 = 50.0

MaxCoolIn	REAL	最大冷却输入。指定引起最大冷却量的 In 的百分比。对于加热/冷却回路，该值通常为 0%。 有效值 = 任意浮点值 默认值 = 0.0
MinCoolIn	REAL	最小冷却输入。指定引起最小冷却量的 In 的百分比。对于加热/冷却回路，该值通常为 50%。 有效值 = 任意浮点值 默认值 = 50.0
MaxHeatTime	REAL	最长加热时间，单位为秒。指定加热脉冲可持续的最长时间（单位为秒）。如果指令计算出的 HeatTime 值大于该值，则将 HeatTime 限制为 MaxHeatTime。如果 MaxHeatTime 无效，该指令会假定该值等于 CycleTime 并将 Status 中的相应位置位。 有效值 = 0.0 到 CycleTime 默认值 = CycleTime
MinHeatTime	REAL	最短加热时间，单位为秒。指定加热脉冲可持续的最短时间（单位为秒）。如果指令计算出的 HeatTime 值小于该值，则将 HeatTime 设置为零。如果 MinHeatTime 无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 0.0 到 MaxHeatTime 默认值 = 0.0
MaxCoolTime	REAL	最长冷却时间，单位为秒。指定冷却脉冲可持续的最长时间（单位为秒）。如果指令计算出的 CoolTime 值大于该值，则将 CoolTime 限制为 MaxCoolTime。如果 MaxCoolTime 无效，该指令会假定该值等于 CycleTime 并将 Status 中的相应位置位。 有效值 = 0.0 到 CycleTime 默认值 = CycleTime

MinCoolTime	REAL	最短冷却时间，单位为秒。指定冷却脉冲可持续的最短时间（单位为秒）。如果指令计算出的 CoolTime 值小于该值，则将 CoolTime 设置为零。如果 MinCoolTime 无效，该指令会假定该值等于零并将 Status 中的相应位置位。 有效值 = 0.0 到 MaxCoolTime 默认值 = 0.0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 HeatTimePercent 或 CoolTimePercent 溢出，则设置为假。
HeatOut	BOOL	加热输出脉冲。指令为加热触点生成此输出脉冲。
CoolOut	BOOL	冷却输出脉冲。指令为冷却触点生成此输出脉冲。
HeatTimePercent	REAL	以百分比表示的加热输出脉冲时间。该值通过计算得出，是 HeatingOutput 处于接通状态的持续时间在当前周期中所占的百分比。因此，用户可根据需要将此指令与和模拟量输出配合使用以进行加热。
CoolTimePercent	REAL	以百分比表示的冷却输出脉冲时间。该值通过计算得出，是 CoolingOutput 处于接通状态的持续时间在当前周期中所占的百分比。因此，用户可根据需要将此指令与和模拟量输出配合使用以进行冷却。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
CycleTimeInv (Status.1)	BOOL	CycleTime 值无效。指令将使用零值。

MaxHeatTimeInv (Status.2)	BOOL	MaxHeatTime 值无效。指令将使用 CycleTime 值。
MinHeatTimeInv (Status.3)	BOOL	MinHeatTime 值无效。指令将使用零值。
MaxCoolTimeInv (Status.4)	BOOL	MaxCoolTime 值无效。指令将使用 CycleTime 值。
MinCoolTimeInv (Status.5)	BOOL	MinCoolTime 值无效。指令将使用零值。
HeatSpanInv (Status.6)	BOOL	MaxHeatIn = MinHeatIn。
CoolSpanInv (Status.7)	BOOL	MaxCoolIn = MinCoolIn。

说明

SRTP 脉冲长度与 PID 输出成正比。上述指令参数适用于加热和冷却应用。

使用内部周期计时器

该指令具有一个自由运行的周期计时器，可在零和设定的 CycleTime 之间循环。内部计时器通过 DeltaT 更新。DeltaT 是自该指令上次执行后所经过的时间。此计时器确定是否需要将输出接通。

用户可以随时更改 CycleTime。如果 CycleTime = 0，内部计时器清零，HeatOut 和 CoolOut 设置为假。

计算加热时间和冷却时间

每次执行指令时都会计算加热时间和冷却时间。

HeatTime 是指 CycleTime 内加热输出处于接通状态的持续时间。

$$HeatTime = \frac{In - MinHeatIn}{MaxHeatIn - MinHeatIn} \times CycleTime$$

如果 HeatTime < MinHeatTime，则设置 HeatTime = 0。

如果 HeatTime > MaxHeatTime，则限制 HeatTime = MaxHeatTime。

HeatTimePercent 是指 HeatOut 脉冲为真的持续时间在 CycleTime 中所占的百分比。

$$HeatTimePercent = \frac{HeatTime}{CycleTime} \times 100$$

CoolTime 是指 CycleTime 内冷却输出处于接通状态的持续时间。

$$CoolTime = \frac{In - MinCoolIn}{MaxCoolIn - MinCoolIn} \times CycleTime$$

如果 CoolTime < MinCoolTime，则设置 CoolTime = 0。

如果 CoolTime > MaxCoolTime，则限制 CoolTime = MaxCoolTime。

CoolTimePercent 是指 CoolOut 脉冲为真的持续时间在 CycleTime 中所占的百分比。

$$CoolTimePercent = \frac{CoolTime}{CycleTime} \times 100$$

指令按照以下规则控制加热输出和冷却输出：

- 如果 HeatTime ≥ 内部周期时间累加器的值，会将 HeatOut 设置为真。内部周期计时器的值 > HeatTime 时，会将 HeatOut 设置为假。
- 如果 CoolTime ≥ 内部周期时间累加器的值，会将 CoolOut 设置为真。如果内部周期计时器的值 > CoolTime，会将 CoolOut 设置为假。
- 如果 CycleTime = 0，会将 HeatOut 和 CoolOut 设置为假。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。

指令首次运行	不适用
指令首次扫描	HeatOut 和 CoolOut 设置为假。 HeatTimePercent 和 CoolTimePercent 清零 (0.0)。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

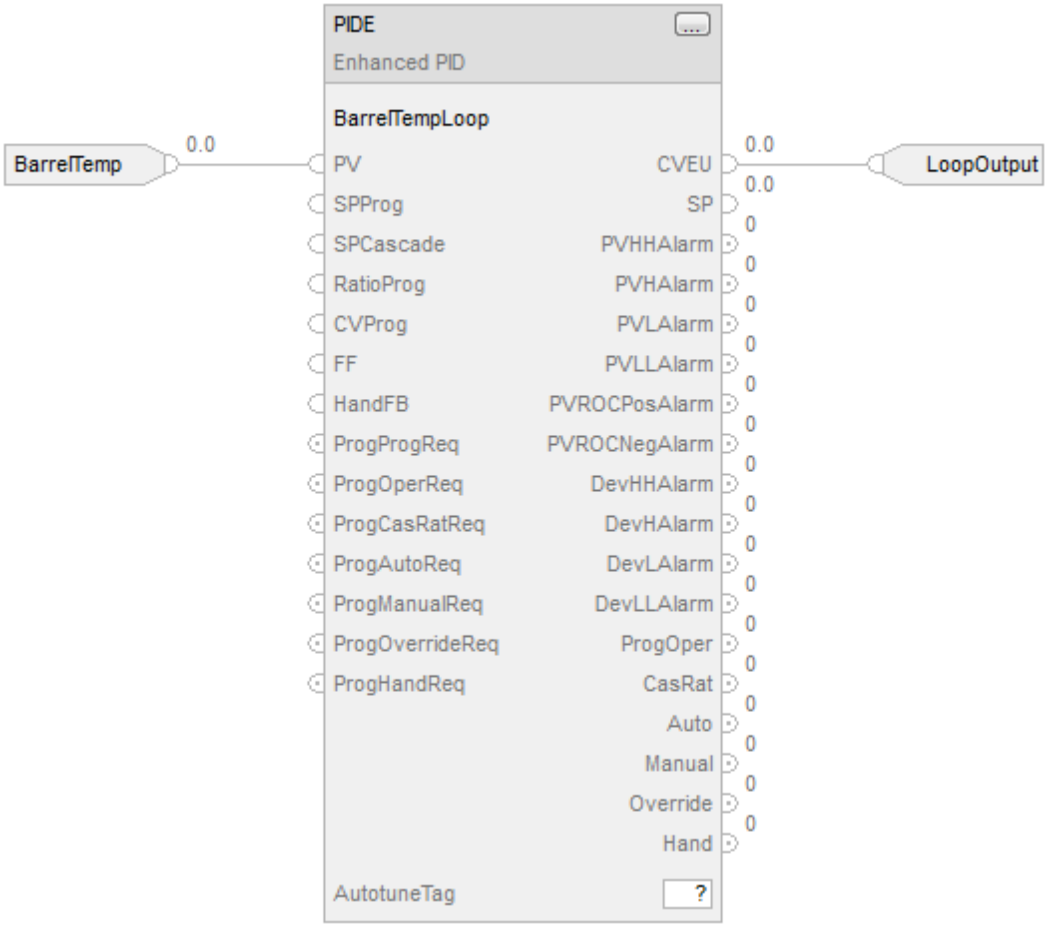
条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

示例

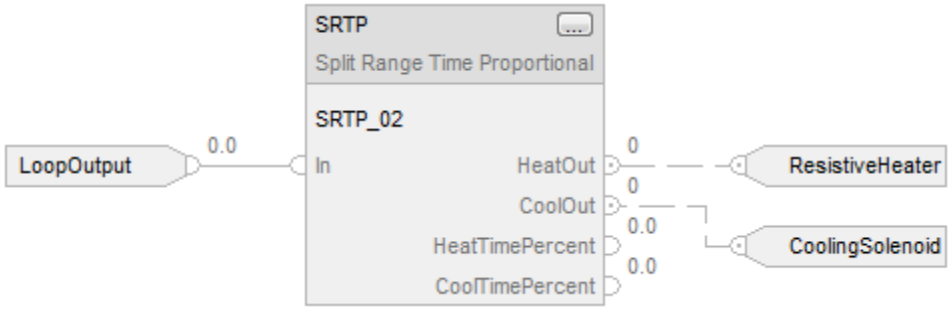
在本例中，PIDE 指令在速度缓慢且优先级较低的任务中执行，即，速度缓慢的温度回路。PIDE 指令的输出属于控制器作用域标签，因为该输出将作为 SRTP 指令的输入。SRTP 指令在速度较快且优先级较高的任务中执行，因此脉冲输出更加准确

功能块

将 PIDE 指令置于速度较慢且优先级较低的任务中



将 SRTP 指令置于速度较快且优先级较高的任务中。



结构化文本

将 PIDE 指令置于速度较慢且优先级较低的任务中。

```
BarrelTempLoop.PV := BarrelTemp;

PIDE(BarrelTempLoop);

LoopOutput := BarrelTempLoop.CVEU;
```

将 SRTP 指令置于速度较快且优先级较高的任务中。

```
SRTP_02.In := LoopOutput;

SRTP(SRTP_02);

ResistiveHeater := SRTP_02.HeatOut;

CoolingSolenoid := SRTP_02.CoolOut;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

累加器 (TOT)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

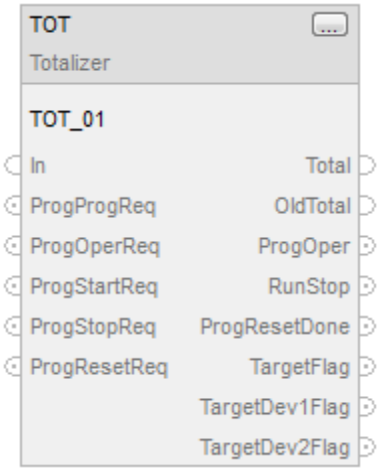
TOT 指令可对一段时间内的模拟量输入值进行累加。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

TOT(TOT_tag)

操作数

功能块

操作数	类型	格式	说明
TOT tag	TOTALIZER	结构	TOT 结构

结构化文本

操作数	类型	格式	说明
TOT tag	TOTALIZER	结构	TOT 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

TOTALIZER 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0

InFault	BOOL	In 不良状况指示器。如果为真，说明输入信号存在错误，此指令会将 Status 中的相应位置位，控制算法不会执行且不会更新 Total。 默认值为假。
TimeBase	DINT	时基输入。基于 In 工程单位的累加时基。 0 = 秒 1 = 分钟 2 = 小时 3 = 天 例如，如果 In 的单位为 gal/min，则使用 TimeBase = 分钟。如果该值无效，指令会将 Status 中的相应位置位，且不会更新 Total。 有关时序模式的详细信息，请参见“功能块属性”部分。 有效值 = 0 到 3 默认值 = 0
Gain	REAL	增量累加值的乘数。用户可以使用 Gain 来转换累加单位。例如，可以使用 Gain 将 gal/min 转换为以桶为单位的总计值。 有效值 = 任意浮点值 默认值 = 1.0
ResetValue	REAL	复位值输入。OperResetReq 或 ProgResetReq 由假跳变为真时的 Total 复位值。 有效值 = 任意浮点值 默认值 = 0.0
Target	REAL	累加 In 的目标值。 有效值 = 任意浮点值 默认值 = 0.0
TargetDev1	REAL	Total 预设目标值与 Target 值之间较大的偏差值。该值表示为与 Target 之间的偏差。 有效值 = 任意浮点值 默认值 = 0.0

TargetDev2	REAL	Total 预设目标值与 Target 值之间较小的偏差值。该值表示为与 Target 之间的偏差。 有效值 = 任意浮点值 默认值 = 0.0
LowInCutoff	REAL	指令输入下限输入。当 In 不大于 LowInCutoff 值时，累加停止。 有效值 = 任意浮点值 默认值 = 0.0
ProgProgReq	BOOL	程序发出的程序控制请求。设置为真时可请求程序控制。如果 ProgOperReq 为真，则忽略该值。保持该参数为真且保持 ProgOperReq 为假可将指令锁定为程序控制模式。 默认值为假。
ProgOperReq	BOOL	程序发出的操作员控制请求。设置为真时可请求操作员控制。保持该参数为真可将指令锁为操作员控制模式。 默认值为假。
ProgStartReq	BOOL	程序启动请求输入。设置为真时可请求累加启动。 默认值为假。
ProgStopReq	BOOL	程序停止请求输入。设置为真时可请求累加停止。 默认值为假。
ProgResetReq	BOOL	程序复位请求输入。设置为真时可请求 Total 复位为 ResetValue。 默认值为假。
OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。指令会将此输入设置为假。 默认值为假。
OperOperReq	BOOL	操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。指令会将此输入设置为假。 默认值为假。

OperStartReq	BOOL	操作员启动请求输入。由操作员界面设置为真可请求累加启动。指令会将此输入设置为假。 默认值为假。
OperStopReq	BOOL	操作员停止请求输入。由操作员界面设置为真可请求累加停止。指令会将此输入设置为假。 默认值为假。
OperResetReq	BOOL	操作员复位请求输入。由操作员界面设置为真可请求累加复位。指令将该输入设置为假。默认值为假。
ProgValueReset	BOOL	将程序控制值复位。该值为真时，每次执行指令时，所有程序请求输入都将设置为假。 默认值为假。
TimingMode	DINT	选择时序执行模式。 0 = 周期模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见“功能块属性”部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Total 值溢出，则设置为假。
Total	REAL	累加值（如果出现 In 值）。

OldTotal	REAL	复位前的总计值。可通过监视该值来读取最近一次复位前的确切总计值。
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。操作员控制模式下为假。
RunStop	BOOL	累加器运行状态指示器。TOT 指令运行时为真。TOT 指令停止时为假。
ProgResetDone	BOOL	该指示器用于指示 TOT 指令已完成程序复位请求。指令因 ProgResetReq 而复位时，设置为真。可通过监视此参数来确定是否已成功完成复位。ProgResetReq 为假时设置为假。
TargetFlag	BOOL	Total 值的标志。当 Total ≥ Target 时设置为真。
TargetDev1Flag	BOOL	TargetDev1 值的标志。当 Total ≥ Target - TargetDev1 时设置为真。
TargetDev2Flag	BOOL	TargetDev2 值的标志。当 Total ≥ Target - TargetDev2 时设置为真。
LowInCutoffFlag	BOOL	指令输入下限标志输出。当 In 值 £ LowInCutoff 值时， 设置为真。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InFaulted (Status.1)	BOOL	In 值有误。
TimeBaseInv (Status.2)	BOOL	TimeBase 值无效。
TimingModeInv (Status.27)	BOOL	TimingMode 值无效。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS(DeltaT – RTSTime) > 1 毫秒时， 设置为真。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。

RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。如果在过采样时序模式下 OversampleDT 值无效，则可能发生此情况。

说明

该指令通常根据流量信号对一定时间内添加的物料量进行累加。

TOT 指令支持以下功能：

- 将时基选择为秒、分、小时或天。
- 可指定一个目标值和最多两个预设目标值。预设目标值通常用于切换为更慢的给料速度。相应的数字量标志会指示是否达到目标值或预设目标 值。
- 可使用低流量输入下限值，来避免在进料流停止时因微小的流量计校准错误而发生负向累加。
- 可由操作员或程序控制启动/停止/复位功能。
- 可由用户自定义复位值。
- 支持梯形规则的数字积分，可提高准确度。
- 内部累加采用双精度运算方式，可提高准确度。

监视 TOT 指令

TOT 指令有相应的操作员面板。

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。

Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	Total 设置为 ResetValue。 OldTotal 清零为 0.0。 ProgOper 设置为假。
指令首次扫描	所有操作员请求输入设置为假。如果 ProgValueReset 为真 ,则所有程序请求输入设置为假。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

检查低流量输入下限

如果 (In <= LowInCutoff)，则指令将 LowInCutoffFlag 设置为真，并使 In(n-1) = 0.0。

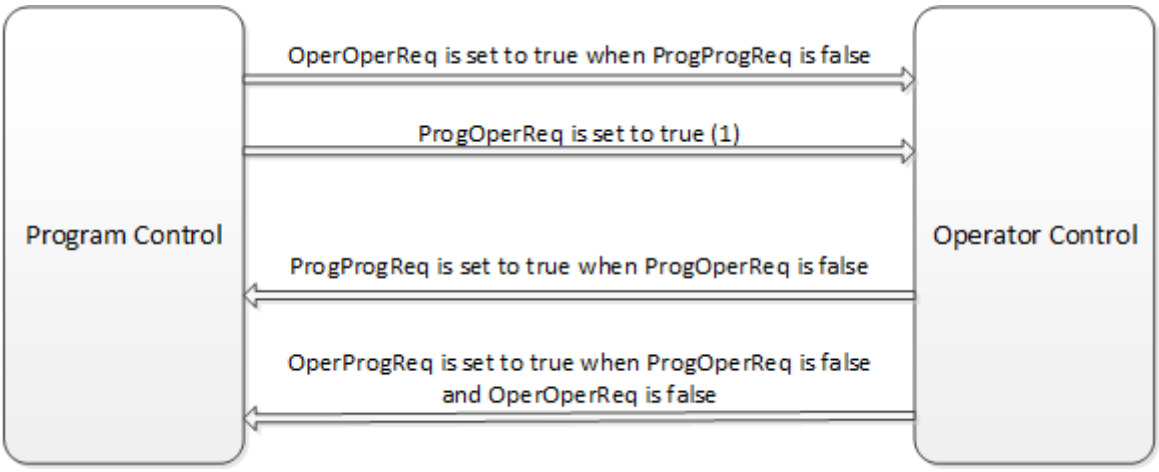
否则，指令将 LowInCutoffFlag 设置为假。

LowInCutoffFlag 为真时，会确定工作模式，但停止累加。

LowInCutoffFlag 为假时，该次扫描将继续累加。

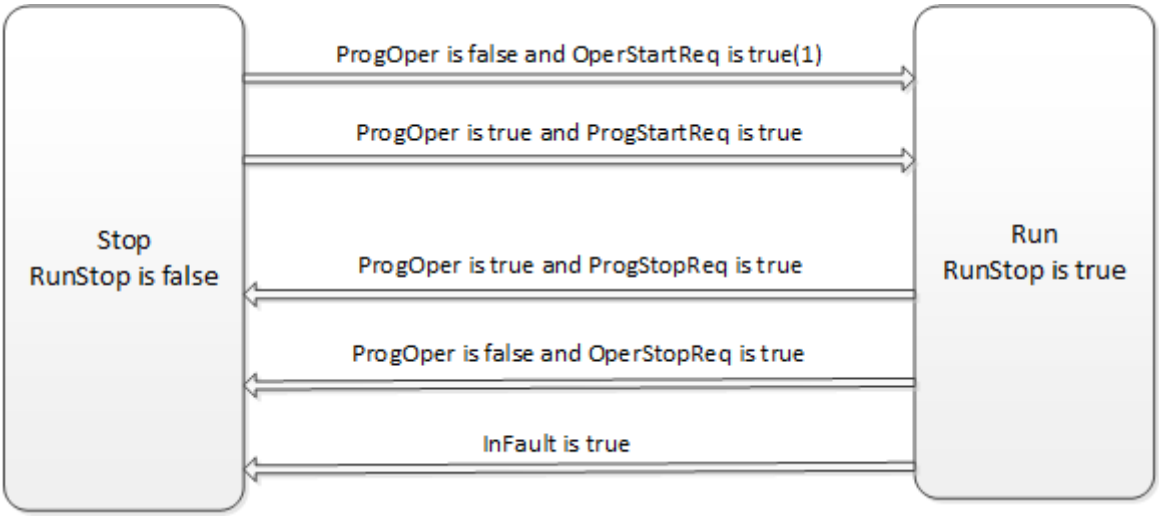
工作模式

下图显示了 TOT 指令在程序控制与操作员控制模式之间进行切换的方式。



(1) 当 ProgOperReq 为真时，指令保持在操作员控制模式。

下图显示了 TOT 指令在运行模式和停止模式之间进行切换的方式。



(1) 停止请求优先于启动请求。

(2) 停止后再次运行时的首次扫描期间，不计算累加值，但更新 In_{n-1}。

下次扫描时，继续进行累加。

每次扫描结束时，所有操作员请求输入都将设置为假。如果 ProgValueReset 为真，每次扫描结束时，所有程序请求输入都将设置为假。

复位 TOT 指令

如果 ProgOper 为真，当 ProgResetReq 跳变为真值时，将发生以下情况：

- OldTotal = Total
- Total = ResetValue
- ProgResetDone 设置为真

如果 ProgResetReq 为假且 ProgResetDone 为真，则 ProgResetDone 设置为假

如果 ProgOper 为假，当 OperResetReq 跳变为真值，将发生以下情况：

- OldTotal = Total
- Total = ResetValue

计算累加值

当 RunStop 为真且 LowInCutoffFlag 为假时，将根据以下公式进行累加计算。

$$Total_n = Total_{n-1} + Gain \times \frac{DeltaT}{2 \times TimeBase} \times (In_n + In_{n-1})$$

其中 TimeBase 为：

值	条件
1	TimeBase = 0（秒）
60	TimeBase = 1（分钟）
3600	TimeBase = 2（小时）
86400	TimeBase = 3（天）

确定是否达到目标值

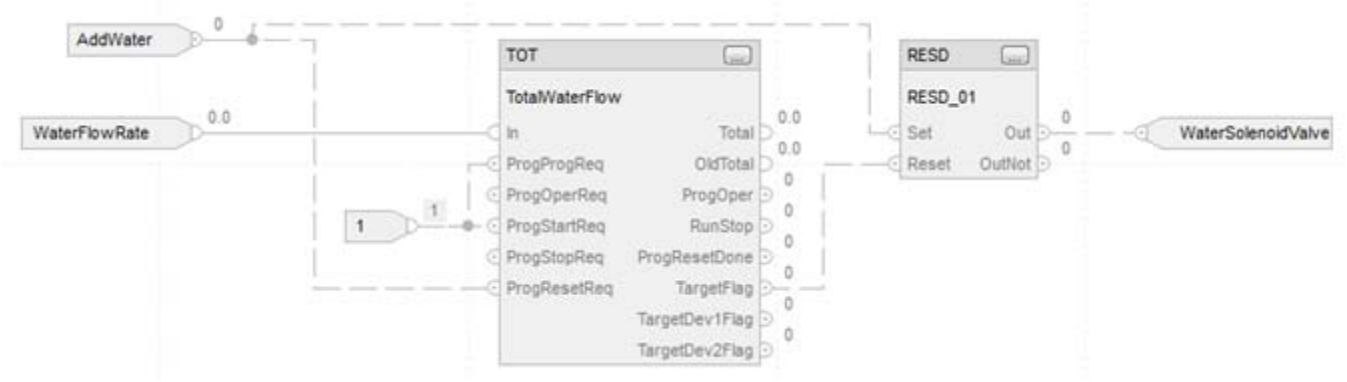
计算出累加值后，将根据以下规则确定是否达到目标值或预设目标值：

- Total ≥ Target 时，TargetFlag 为真
- Total ≥ (Target - TargetDev1) 时，TargetDev1Flag 为真
- Total ≥ (Target - TargetDev2) 时，TargetDev2Flag 为真

示例

在本示例中，使用 TOT 指令计量加入水箱中的水量，当添加量达到特定值后，水流关闭。按下 AddWater 按钮后，TOT 指令会复位并开始累加流入水箱的水量。达到 Target 值后，TOT 指令将 TargetFlag 输出置位，使电磁阀关闭。在此示例中，通过将 ProgProgReq 和 ProgStartReq 输入置位，将 TOT 指令“锁定”到程序运行模式。这样，操作员便无需再直接控制 TOT 指令。

功能块



结构化文本

```
TotalWaterFlow.In := WaterFlowRate;
TotalWaterFlow.ProgProgReq := 1;
TotalWaterFlow.ProgStartReq := 1;
TotalWaterFlow.ProgResetReq := AddWater;
TOT(TotalWaterFlow);

RESD_01.Set := AddWater;
RESD_01.Reset := TotalWaterFlow.TargetFlag;
RESD(RESD_01);

WaterSolenoidValve := RESD_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

协调控制 (CC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

CC 指令通过操作多达三个不同的控制变量控制单个过程变量。此外，也可以将三个输出中的任意一个用作输入，以在控制器中创建前馈操作。在自动模式下，CC 基于 PV -SP 偏差、内部模型和调节，计算控制变量（CV1、CV2 和 CV3）。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块

结构化文本

CC(CC_tag);

操作数

结构化文本

操作数	类型	格式	说明
CC tag	COORDINATED_ CONTROL	结构	CC 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

重要事项： 只要 APC 块检测到增量时间 (DeltaT) 发生变化，便执行 ModelInit。因此，这些块应该仅在 DeltaT 为常量的一种 TimingMode 下运行。

- TimingMode = 0（周期性），在周期性任务中执行这些功能块时
- TimingMode = 1（过采样）

无论在哪种情况下，如果周期性任务的时间动态变化，或者 OversampleDT 动态变化，该块便执行 ModelInit。

由于 DeltaT 中存在抖动，因此不推荐使用下面的 TimingMode 设置：

- TimingMode = 0（周期性），在连续任务或事件任务中执行这些功能块时
- TimingMode = 2（实时采样）

功能块

操作数	类型	格式	说明
CC tag	COORDINATED CONTROL	结构	CC 结构

结构

输入参数	数据类型	说明	有效值和默认值
EnableIn	BOOL	使能输入。如果为“假”，则功能块不执行，输出也不会更新。	默认值 = 真
PV	REAL	经过标定的过程变量输入。该值通常从模拟量输入模块中读取。	有效值 = 任意浮点值 默认值 = 0.0

PVFault	BOOL	PV 不良状况指示器。如果从模拟量输入读取 PV，则 PVFault 通常由模拟量输入故障状态控制。 如果 PVFault 为“真”,则表示输入模块发生错误，并且会将 Status 中的相应位置位。	假 = 状况良好 默认值 = 假
PVUEMax	REAL	PV 的最大标定值。对应于过程变量的 100% 量程的 PV 和 SP 的值。 如果 PVEUMax ≤ PVEUMin，则将 Status 中的相应位置位。	有效值 = PVEUMin < PVEUMax ≤ 最大正浮点值 默认值 = 100.0
PVUEMin	REAL	PV 的最小标定值。对应于过程变量的 0% 量程的 PV 和 SP 的值。 如果 PVEUMax ≤ PVEUMin，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 ≤ PVEUMin < PVEUMax 默认值 = 0.0
SPProg	REAL	SP 程序值，以 PV 单位标定。当指令处于程序控制模式时，SP 设置为此值。	有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPOper	REAL	SP 操作员值 ,以 PV 单位标定。SP 设置为此值， 前提是处于操作员控制模式时。 如果 SPProg 或 SPOper 的值 < SPLLimit 或 > SPHLimit，则将 Status 中的相应位置位，并限制 SP 的值。	有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPHLimit	REAL	SP 上限值，以 PV 单位标定。 如果 SPHLimit < SPLLimit 或 SPHLimit > PVEUMax，则将 Status 中的相应位置位。	有效值 = SPLLimit 到 PVEUMax 默认值 = 100.0

SPLLimit	REAL	SP 下限值，以 PV 单位标定。 如果 SPLLimit < PVEUMin 或 SPHLimit < SPLLimit，则将 Status 中的相应位置位，并使用 SPLLimit 的值限制 SP。	有效值 = PVEUMin 到 SPHLimit 默认值 = 0.0
CV1Fault	BOOL	控制变量 1 不良状况指示器。如果 CV1EU 控制模拟量输出，则 CV1Fault 通常由该模拟量输出的故障状态引起。 如果 CV1Fault 为“真”，则表示输出模块发生错误，并且将 Status 中的相应位置位。	默认值 = 假 假 = 状况良好
CV2Fault	BOOL	控制变量 2 不良状况指示器。如果 CV2EU 控制模拟量输出，则 CV2Fault 通常由该模拟量输出的故障状态引起。 如果 CV2Fault 为“真”，则表示输出模块发生错误，并且将 Status 中的相应位置位。	默认值 = 假 假 = 状况良好
CV3Fault	BOOL	控制变量 3 不良状况指示器。如果 CV3EU 控制模拟量输出，则 CV3Fault 通常由该模拟量输出的故障状态引起。 如果 CV3Fault 为“真”，则表示输出模块发生错误，并且将 Status 中的相应位置位。	默认值 = 假 假 = 状况良好
CV1InitReq	BOOL	CV1 初始化请求。为“真”时，将 CV1EU 设置为 CVInitValue 的值。该信号通常由受控于 CV1EU 的模拟量输出模块上的 In Hold 状态控制，或者从次级回路的 InitPrimary 输出获得。	默认值 = 假

CV2InitReq	BOOL	CV2 初始化请求。为“真”时，将 CV2EU 设置为 CVInitValue 的值。该信号通常由受控于 CV2EU 的模拟量输出模块上的 In Hold 状态控制，或者从次级回路的 InitPrimary 输出获得。	默认值 = 假
CV3InitReq	BOOL	CV3 初始化请求。为“真”时，将 CV3EU 设置为 CVInitValue 的值。该信号通常由受控于 CV32EU 的模拟量输出模块上的 In Hold 状态控制，或者从次级回路的 InitPrimary 输出获得。	默认值 = 假
CV1InitValue	REAL	CV1EU 初始化值，以 CV1EU 单位标定。CV1Initializing 为“真”时，CV1EU 设为等于 CV1InitValue 并且 CV1 设为相应的百分比值。CV1InitValue 通常从受 CV1EU 控制的模拟量输出的反馈获取，或者从次级回路的设置点得出。 当 CVFaulted 或 CVEUSpanInv 为“真”时，将禁用指令初始化。	有效值 = 任意浮点值 默认值 = 0.0
CV2InitValue	REAL	CV2EU 初始化值，以 CV2EU 单位标定。当 CV2Initializing 为“真”时，将 CV2EU 设为等于 CV2InitValue 并且 CV2 设为相应的百分比值。CV2InitValue 通常从受 CV2EU 控制的模拟量输出的反馈获取，或者从次级回路的设置点得出。 当 CVFaulted 或 CVEUSpanInv 为“真”时，将禁用指令初始化。	有效值 = 任意浮点值 默认值 = 0.0

CV3InitValue	REAL	CV3EU 初始化值，以 CV3EU 单位标定。当 CV3Initializing 为"真"时，将 CV3EU 设为等于 CV3InitValue 并且 CV3 设为相应的百分比值。 CV3InitValue 通常从受 CV3EU 控制的模拟量输出的反馈获取，或者从次级回路的设置点得出。 当 CVFaulted 或 CVEUSpanInv 为"真"时，将禁用指令初始化。	有效值 = 任意浮点值 默认值 = 0.0
CV1Prog	REAL	程序-手动模式下的 CV1 值。在程序控制和手动模式下，CV1 设置为此值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV2Prog	REAL	程序-手动模式下的 CV2 值。在程序控制和手动模式下，CV2 设置为此值。	有效值 = 0.0...100.0 默认值 = 0.0
CV3Prog	REAL	程序-手动模式下的 CV3 值。在程序控制和手动模式下，CV3 设置为此值。	有效值 = 0.0...100.0 默认值 = 0.0
CV1Oper	REAL	操作员-手动模式下的 CV1 值。在操作员控制和手动模式下，CV1 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CV1Oper 设置为 CV1 值。 当 CVManLimiting 为"真"时，如果 CV1Oper 的值 < 0 或 > 100，或者 < CV1LLimit 或 > CV1HLimit，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0

CV2Oper	REAL	操作员-手动模式下的 CV2 值。在操作员控制和手动模式下时，CV2 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CV2Oper 设置为 CV2 值。 当 CVManLimiting 为“真”时，如果 CV2Oper 的值 < 0 或 > 100，或者 < CV2LLimit 或 > CV2HLimit，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0
CV3Oper	REAL	操作员-手动模式下的 CV3 值。在操作员控制和手动模式下时，CV3 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CV3Oper 设置为 CV3 值。 当 CVManLimiting 为“真”时，如果 CV3Oper 的值 < 0 或 > 100，或者 < CV3LLimit 或 > CV3HLimit，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0
CV1OverrideValue	REAL	超控模式下的 CV1 值。在超控模式下，CV1 设置为此值。 此值应对应于回路的安全状态输出。 如果 CV1OverrideValue 的值 < 0 或 > 100，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0

CV2OverrideValue	REAL	超控模式下的 CV2 值。在超控模式下，CV2 设置为此值。 此值应对应于回路的安全状态输出。 如果 CV2OverrideValue 的值 < 0 或 > 100，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0
CV3OverrideValue	REAL	超控模式下的 CV3 值。在超控模式下，CV3 设置为此值。 此值应对应于回路的安全状态输出。 如果 CV3OverrideValue 的值 < 0 或 > 100，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0
CV1TrackValue	REAL	CV1 跟踪值。当启用 CVTrackReq 且 CC 功能块处于手动模式时，将忽略 CV1TrackValue，CC 内部模型将以 CV1Oper 或 CV1Prog 值更新其历史数据。当启用 CVTrackReq 且 CC 功能块处于自动模式时，内部模型将根据 CV1TrackValue 的值更新其历史数据。 这种情况下，CV1 值将可以正常变化，就如同 CC 功能块仍在控制该过程。在多回路选择方案中，如果希望 CC 功能块跟随不同控制算法的输出，则该功能将非常有用，可将控制算法的输出连接到 CV1TrackValue。	有效值 = 0.0...100.0 默认值 = 0.0

CV2TrackValue	REAL	CV2 跟踪值。当启用 CVTrackReq 且 CC 功能块处于手动模式时，将忽略 CV2TrackValue ,CC 内部模型将以 CV2Oper 或 CV2Prog 值更新其历史数据。当启用 CVTrackReq 且 CC 功能块处于自动模式时，内部模型将根据 CV2TrackValue 的值更新其历史数据。 这种情况下，CV2 值将可以正常变化，就如同 CC 功能块仍在控制该过程。在多回路选择方案中，如果希望 CC 功能块跟随不同控制算法的输出，则该功能将非常有用，可将控制算法的输出连接到 CV2TrackValue。	有效值 = 0.0...100.0 默认值 = 0.0
CV3TrackValue	REAL	CV3 跟踪值。当启用 CVTrackReq 且 CC 功能块处于手动模式时，将忽略 CV3TrackValue ,CC 内部模型将以 CV3Oper 或 CV3Prog 值更新其历史数据。当启用 CVTrackReq 且 CC 功能块处于自动模式时，内部模型将根据 CV3TrackValue 的值更新其历史数据。 这种情况下，CV3 值将可以正常变化，就如同 CC 功能块仍在控制该过程。在多回路选择方案中，如果希望 CC 功能块跟随不同控制算法的输出，则该功能将非常有用，可将控制算法的输出连接到 CV3TrackValue。	有效值 = 0.0...100.0 默认值 = 0.0

CVManLimiting	BOOL	限制手动模式请求中的 CV(n)，其中，(n) 可以是 1、2 或 3。如果处于手动模式并且 CVManLimiting 为“真”，则 CV 由 CV(n)HLimit 和 CV(n)LLimit 值限制。	默认值 = 假
CV1EUMax	REAL	CV1EU 的最大值。对应于 100% CV1 的 CV1EU 值。如果 CV1EUMax = CV1EUMin，会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CV2EUMax	REAL	CV2EU 的最大值。对应于 100% CV2 的 CV2EU 值。如果 CV2EUMax = CV2EUMin，会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CV3EUMax	REAL	CV3EU 的最大值。对应于 100% CV3 的 CV3EU 值。如果 CV3EUMax = CV3EUMin，会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CV1EUMin	REAL	CV1EU 的最小值。对应于 0% CV1 的 CV1EU 值。如果 CV1EUMax = CV1EUMin，会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0
CV2EUMin	REAL	CV2EU 的最小值。对应于 0% CV2 的 CV2EU 值。如果 CV2EUMax = CV2EUMin，会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0
CV3EUMin	REAL	CV3EU 的最小值。对应于 0% CV3 的 CV3EU 值。如果 CV3EUMax = CV3EUMin，会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0

CV1HLimit	REAL	CV1 上限值。该值用于设置 CV1HAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV1 进行限制。 如果 CV1HLimit > 100 或 CV1HLimit < CV1LLimit，则将 Status 中的相应位置位。 如果 CV1HLimit < CV1LLimit，则使用 CV1LLimit 的值限制 CV1。	有效值 = CV1LLimit < CV1HLimit ≤ 100.0 默认值 = 100.0
CV2HLimit	REAL	CV2 上限值。该值用于设置 CV2HAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV2 进行限制。 如果 CV2HLimit > 100 或 CV2HLimit < CV2LLimit，则将 Status 中的相应位置位。 如果 CV2HLimit < CV2LLimit，则使用 CV2LLimit 的值限制 CV2。	有效值 = CV2LLimit < CV2HLimit ≤ 100.0 默认值 = 100.0
CV3HLimit	REAL	CV3 上限值。该值用于设置 CV3HAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV3 进行限制。 如果 CV3HLimit > 100，或 CV3HLimit < CV3LLimit，则将 Status 中的相应位置位。 如果 CV3HLimit < CV3LLimit，则使用 CV3LLimit 的值限制 CV3。	有效值 = CV3LLimit < CV3HLimit ≤ 100.0 默认值 = 100.0

CV1LLimit	REAL	CV1 下限值。该值用于设置 CV1LAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV1 进行限制。 如果 CV1LLimit < 0，则将 Status 中的相应位置位。如果 CV1HLimit < CV1LLimit，则使用 CV1LLimit 的值限制 CV。	有效值 = 0.0 ≤ CV1LLimit < CV1HLimit 默认值 = 0.0
CV2LLimit	REAL	CV2 下限值。该值用于设置 CV2LAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV2 进行限制。 如果 CV2LLimit < 0，则将 Status 中的相应位置位。如果 CV2HLimit < CV2LLimit，则使用 CV2LLimit 的值限制 CV。	有效值 = 0.0 ≤ CV2LLimit < CV1HLimit 默认值 = 0.0
CV3LLimit	REAL	CV3 下限值。该值用于设置 CV3LAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 设置为“真”时，该值也用于对 CV3 进行限制。 如果 CV3LLimit < 0，则将 Status 中的相应位置位。如果 CV3HLimit < CV3LLimit，则使用 CVLLimit 的值限制 CV。	有效值 = 0.0 ≤ CV3LLimit < CV1HLimit 默认值 = 0.0

CV1ROCPosLimit	REAL	CV1 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV1 ROC 限制。如果 CV1ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV1 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2ROCPosLimit	REAL	CV2 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV2 ROC 限制。如果 CV2ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV2 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3ROCPosLimit	REAL	CV3 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV3 ROC 限制。如果 CV3ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV3 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1ROCNegLimit	REAL	CV1 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV1 ROC 限制。如果 CV1ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV1 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0

CV2ROCNegLimit	REAL	CV2 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV2 ROC 限制。如果 CV2ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV2 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3ROCNegLimit	REAL	CV3 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV3 ROC 限制。如果 CV3ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV3 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1HandFB	REAL	CV1 HandFeedback 值。当处于手动模式下且 CV1HandFBFault 为假（状况良好）时，将 CV1 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。如果 CV1HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV1 的值。	有效值 = 0.0...100.0 默认值 = 0.0
CV2HandFB	REAL	CV2 HandFeedback 值。当处于手动模式下且 CV2HandFBFault 为假（状况良好）时，将 CV2 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。如果 CV2HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV2 的值。	有效值 = 0.0...100.0 默认值 = 0.0

CV3HandFB	REAL	CV3 HandFeedback 值。当处于手动模式下且 CV3HandFBFault 为假（状况良好）时，将 CV3 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。如果 CV3HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV3 的值。	有效值 = 0.0...100.0 默认值 = 0.0
CV1HandFBFault	BOOL	CV1HandFB 值不良状况指示器。如果从模拟量输入中读取 CV1HandFB 值，则 CV1HandFBFault 通常由模拟量输入通道的状态控制。CV1HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。	假 = 状况良好 默认值 = 假
CV2HandFBFault	BOOL	CV2HandFB 值不良状况指示器。如果从模拟量输入中读取 CV2HandFB 值，则 CV2HandFBFault 通常由模拟量输入通道的状态控制。CV2HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。	假 = 状况良好 默认值 = 假
CV3HANDFBFault	BOOL	CV3HandFB 值不良状况指示器。如果从模拟量输入中读取 CV3HandFB 值，则 CV3HandFBFault 通常由模拟量输入通道的状态控制。CV3HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。	假 = 状况良好 默认值 = 假
CV1Target	REAL	CV1 的目标值。	有效值 = 0.0...100.0 默认值 = 0.0

CV2Target	REAL	CV2 的目标值。	有效值 = 0.0...100.0 默认值 = 0.0
CV3Target	REAL	CV3 的目标值。	有效值 = 0.0...100.0 默认值 = 0.0
CV1WindupHIn	BOOL	CV1 积分饱和上限请求。该参数为真时，不允许 CV1 值增加。该信号通常来自次级回路的 CV1WindupHOut 输出。	默认值 = 假
CV2WindupHIn	BOOL	CV2 积分饱和上限请求。该参数为真时，不允许 CV2 值增加。该信号通常来自次级回路的 CV2WindupHOut 输出。	默认值 = 假
CV3WindupHIn	BOOL	CV3 积分饱和上限请求。该参数为真时，不允许 CV3 值增加。该信号通常来自次级回路的 CV3WindupHOut 输出。	默认值 = 假
CV1WindupLIn	BOOL	CV1 积分饱和下限请求。该参数为真时，不允许 CV1 值减小。该信号通常来自次级回路的 CV1WindupLOut 输出。	默认值 = 假
CV2WindupLIn	BOOL	CV2 积分饱和下限请求。该参数为真时，不允许 CV2 值减小。该信号通常来自次级回路的 CV2WindupLOut 输出。	默认值 = 假
CV3WindupLIn	BOOL	CV3 积分饱和下限请求。该参数为真时，不允许 CV3 值减小。该信号通常来自次级回路的 CV3WindupLOut 输出。	默认值 = 假
GainEUSpan	BOOL	CVxModelGain 的单位可以表示为 EU 或量程百分数形式。置位时将 ModelGain 表示为 EU，复位时将 ModelGain 表示为量程百分数。	默认值 = 0

CV1ProcessGainSign	BOOL	仅用于自动调谐。过程增益符号 (Delta PV/Delta CV1)。 置位指示负过程增益（输出增大会导致 PV 减小）。 复位指示正过程增益（输出增大会导致 PV 增大）。	默认值 = 假
CV2ProcessGainSign	BOOL	仅用于自动调谐。过程增益符号 (Delta PV/Delta CV2)。 置位指示负过程增益（输出增大会导致 PV 减小）。 复位指示正过程增益（输出增大会导致 PV 增大）。	默认值 = 假
CV3ProcessGainSign	BOOL	仅用于自动调谐。过程增益符号 (Delta PV/Delta CV3)。 置位指示负过程增益（输出增大会导致 PV 减小）。 复位指示正过程增益（输出增大会导致 PV 增大）。	默认值 = 假
ProcessType	DINT	过程类型选项（1= 积分，0= 非积分）	默认值 =0
CV1ModelGain	REAL	CV1 的内模增益参数。根据过程方向输入正增益或负增益。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0
CV2ModelGain	REAL	CV2 的内模增益参数。根据过程方向输入正增益或负增益。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0
CV3ModelGain	REAL	CV3 的内模增益参数。根据过程方向输入正增益或负增益。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0
CV1ModelTC	REAL	CV1 的内模时间常数，以秒为单位。	有效值 =0.0 到最大正浮点值 默认值 =0.0
CV2ModelTC	REAL	CV2 的内模时间常数，以秒为单位。	有效值 =0.0 到最大正浮点值 默认值 =0.0
CV3ModelTC	REAL	CV3 的内模时间常数，以秒为单位。	有效值 =0.0 到最大正浮点值 默认值 =0.0

CV1ModelDT	REAL	CV1 的内模死区时间， 以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2ModelDT	REAL	CV2 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3ModelDT	REAL	CV3 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1RespTC	REAL	确定 CV1 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2RespTC	REAL	确定 CV2 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3RespTC	REAL	确定 CV3 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
Act1stCV	DINT	用于补偿 PV-SP 偏差的第一个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 1
Act2ndCV	DINT	用于补偿 PV-SP 偏差的第二个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 2
Act3rdCV	DINT	用于补偿 PV-SP 偏差的第三个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 3
Target1stCV	DINT	要按最高优先级驱动到目标值的 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 1
Target2ndCV	DINT	要按第二优先级驱动到目标值的 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 2
Target3rdCV	DINT	要按第三优先级驱动到目标值的 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 3

PVTracking	BOOL	SP 跟踪 PV 请求。设置为真可以使 SP 跟踪 PV。自动模式下忽略该参数。仅当全部的三个输出均处于手动模式时，SP 才跟踪 PV。只要有输出返回到自动模式，PVTracking 立即停止。	默认值 = 假
CVTrackReq	BOOL	CV 跟踪请求。若设置为真，则在自动调谐设置为"关"时启用 CV 跟踪。手控和超控模式下忽略该参数。	默认值 = 假
ManualAfterInit	BOOL	初始化后进入手动模式的请求。 该参数为真时，若 CV(n)Initializing 设置为真，除非当前模式为超控或手控模式，否则相应的 CV(n) 将切换为手动模式，其中 (n) 可以是 1、2 或 3。 当 ManualAfterInit 为假时，CV(n) 模式保持不变。	默认值 = 假
ProgProgReq	BOOL	程序发出的程序控制请求。 由用户程序设置为真请求程序控制模式。 如果 ProgOperReq 为真 ,则忽略该值。如果该值保持为真且 ProgOperReq 为假 ,则可将功能块锁定在程序控制模式。 当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假

ProgOperReq	BOOL	程序发出的操作员控制请求。 由用户程序设置为真可请求操作员控制模式。 如果该值保持为真，则可将功能块锁定在操作员控制模式。 如果 HandFB 值 < 0 或 > 100，指令将 Status 中的相应位置位，并限制 CV 的值。	默认值 = 假
ProgCV1AutoReq	BOOL	CV1 的程序-自动模式请求。 由用户程序设置为真可请求自动模式。 如果 CV1HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV1 的值。	默认值 = 假
ProgCV2AutoReq	BOOL	CV2 的程序-自动模式请求。 由用户程序设置为真可请求自动模式。 如果 CV2HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV2 的值。	默认值 = 假
ProgCV3AutoReq	BOOL	CV3 的程序-自动模式请求。 由用户程序设置为真可请求自动模式。 如果 CV3HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV3 的值。	默认值 = 假

ProgCV1ManualReq	BOOL	CV1 的程序-手动模式请求。 由用户程序设置为真可请求手动模式。 如果 CV1HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV1 的值。	默认值 = 假
ProgCV2ManualReq	BOOL	CV2 的程序-手动模式请求。 由用户程序设置为真可请求手动模式。 如果 CV2HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV2 的值。	默认值 = 假
ProgCV3ManualReq	BOOL	CV3 的程序-手动模式请求。 由用户程序设置为真可请求手动模式。 如果 CV3HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV3 的值。	默认值 = 假
ProgCV1OverrideReq	BOOL	CV1 的程序-超控模式请求。 由用户程序设置为真可请求超控模式。 如果 CV1HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV1 的值。	默认值 = 假
ProgCV2OverrideReq	BOOL	CV2 的程序-超控模式请求。 由用户程序设置为真可请求超控模式。 如果 CV2HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV2 的值。	默认值 = 假

ProgCV3OverrideReq	BOOL	CV3 的程序-超控模式请求。 由用户程序设置为真可请求超控模式。 如果 CV3HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV3 的值。	默认值 = 假
ProgCV1HandReq	BOOL	CV1 的程序-手控模式请求。 由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。	默认值 = 假
ProgCV2HandReq	BOOL	CV2 的程序-手控模式请求。 由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。	默认值 = 假
ProgCV3HandReq	BOOL	CV3 的程序-手控模式请求。 由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。	默认值 = 假
OperProgReq	BOOL	操作员发出的程序控制请求。 由操作员界面设置为真以请求程序控制模式。功能块将此参数复位（假）。	默认值 = 假
OperOperReq	BOOL	操作员发出的操作员控制请求。 由操作员界面设置为真可请求操作员控制模式。功能块将此参数复位（假）。	默认值 = 假

OperCV1AutoReq	BOOL	CV1 的操作员-自动模式请求。 由操作员界面设置为真可请求自动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV2AutoReq	BOOL	CV2 的操作员-自动模式请求。 由操作员界面设置为真可请求自动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV3AutoReq	BOOL	CV3 的操作员-自动模式请求。 由操作员界面设置为真可请求自动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV1ManualReq	BOOL	CV1 的操作员-手动模式请求。 由操作员界面设置为真可请求手动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV2ManualReq	BOOL	CV2 的操作员-手动模式请求。 由操作员界面设置为真可请求手动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV3ManualReq	BOOL	CV3 的操作员-手动模式请求。 由操作员界面设置为真可请求手动模式。功能块将此参数复位（假）。	默认值 = 假
ProgValueReset	BOOL	将程序控制值复位。 该值为真时，Prog_xxx_Req 输入复位（假）。 该值为真且处于程序控制模式下时，将 SPProg 设置为等于 SP，将 CVxProg 设置为等于 CVx。 当 ProgValueReset 为真时，指令将输入复位（假）。	默认值 = 假

TimingMode	DINT	选择时基执行模式。 值/说明 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的更多信息，请参见"功能块属性"部分。	有效值 = 0…2 默认值 = 0
OverSampleDT	REAL	过采样模式的执行时间。	有效值 = 0…4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期。	有效值 = 1…32,767 1 次计数 = 1 ms
RTSTimeStamp	DINT	实时采样模式的模块时戳值。	有效值 = 0…32,767 （从 32,767 跳回到 0） 1 次计数 = 1 ms
PVTuneLimit	REAL	以 PV 单位标定的 PV 调谐限值。当自动调谐正在运行并且预测 PV 值超过此限值时，调谐过程将中止。	有效值 = 任意浮点值 默认值 = 0
AtuneTimeLimit	REAL	CV 发生阶跃变化后完成自动调谐所需的最长时间。自动调谐时间超出此时间时，调谐将会中止。	有效值范围：任意 > 0 的浮点值。 默认值 = 60 分钟
NoiseLevel	DINT	PV 的噪声级别估计值，在调谐期间可对该值进行补偿。 可选项包括：0 = 低、1 = 中、2 = 高	范围：0…2 默认值 = 1
CV1StepSize	REAL	调谐阶跃测试所用的 CV1 步长（百分比）。步长将直接加到 CV1（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%
CV2StepSize	REAL	调谐阶跃测试所用的 CV2 步长（百分比）。步长将直接加到 CV2（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%

CV3StepSize	REAL	调谐阶跃测试所用的 CV3 步长（百分比）。步长将直接加到 CV3（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%
CV1ResponseSpeed	DINT	CV1 所需闭环响应速度。 慢速响应：ResponseSpeed=0 中速响应：ResponseSpeed=1 快速响应：ResponseSpeed=2 。 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0…2 默认值 = 1
CV2ResponseSpeed	DINT	CV2 所需闭环响应速度。 慢速响应：ResponseSpeed=0 中速响应：ResponseSpeed=1 快速响应：ResponseSpeed=2 。 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0…2 默认值 = 1
CV3ResponseSpeed	DINT	CV3 所需闭环响应速度。 慢速响应：ResponseSpeed=0 中速响应：ResponseSpeed=1 快速响应：ResponseSpeed=2 。 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0…2 默认值 = 1
CV1ModelInit	BOOL	CV1 内模初始化开关。请参考“CC 功能块调谐”中的“CC 功能块模型初始化”主题。	默认值 = 假
CV2ModelInit	BOOL	CV2 内模初始化开关。请参考“CC 功能块调谐”中的“CC 功能块模型初始化”主题。	默认值 = 假

CV3ModelInit	BOOL	CV3 内模初始化开关。请参考“CC 功能块调谐”中的“CC 功能块模型初始化”主题。	默认值 = 假
Factor	REAL	非积分模型近似因子。仅用于积分过程类型。	默认值 = 100
AtuneCV1Start	BOOL	CV1 的自动调谐启动请求。该值为真时，启动 CV1 输出的自动调谐。如果 CV1 未处于手动模式，则忽略该值。功能块将该输入复位（假）。	默认值 = 假
AtuneCV2Start	BOOL	CV2 的自动调谐启动请求。该值为真时，启动 CV2 输出的自动调谐。如果 CV2 未处于手动模式，则忽略该值。功能块将该输入复位（假）。	默认值 = 假
AtuneCV3Start	BOOL	CV3 的自动调谐启动请求。该值为真时，启动 CV3 输出的自动调谐。如果 CV3 未处于手动模式，则忽略该值。功能块将该输入复位（假）。	默认值 = 假
AtuneCV1UseModel	BOOL	CV1 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV2UseModel	BOOL	CV2 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假

AtuneCV3UseModel	BOOL	CV3 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV1Abort	BOOL	CV1 的自动调谐中止请求。该值为真时，中止 CV1 输出的自动调谐。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV2Abort	BOOL	CV2 的自动调谐中止请求。该值为真时，中止 CV2 输出的自动调谐。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV3Abort	BOOL	CV3 的自动调谐中止请求。该值为真时，中止 CV3 输出的自动调谐。功能块将该输入参数复位（假）。	默认值 = 假

输出参数	数据类型	说明	有效值和默认值
EnableOut	BOOL	指示指令是否处于启用状态。如果 CV1EU、CV2EU 或 CV3EU 溢出，则该值设置为假。	
CV1EU	REAL	CV1 的标定控制变量输出。使用 CV1EUMax 和 CV1EUMin 标定，其中 CV1EUMax 对应于 100%，CV1EUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CV1EU = (CV1 * CV1EUSpan / 100) + CV1EUMin CV1EU 量程计算 :CV1EUSpan = (CV1EUMax – CV1EUMin)	

CV2EU	REAL	CV2 的标定控制变量输出。使用 CV2EUMax 和 CV2EUMin 标定，其中 CV2EUMax 对应于 100%，CV2EUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CV2EU = (CV2 * CV2EUSpan / 100) + CV2EUMin CV2EU 量程计算：CV2EUSpan = (CV2EUMax – CV2EUMin)	
CV3EU	REAL	CV3 的标定控制变量输出。使用 CV3EUMax 和 CV3EUMin 标定，其中 CV3EUMax 对应于 100%，CV3EUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CV3EU = (CV3 * CV3EUSpan / 100) + CV3EUMin CV3EU 量程计算：CV3EUSpan = (CV3EUMax – CV3EUMin)	
CV1	REAL	控制变量 1 输出。此值始终以 0…100% 表示。当处于自动模式下或者处于手动模式下且 CVManLimiting 为真时，CV1 介于 CV1HLimit 和 CV1LLimit 之间；否则，介于 0 和 100% 之间。	
CV2	REAL	控制变量 2 输出。此值始终以 0…100% 表示。当处于自动模式下或者处于手动模式下且 CVManLimiting 为真时，CV2 介于 CV2HLimit 和 CV2LLimit 之间；否则，介于 0 和 100% 之间。	

CV3	REAL	控制变量 3 输出。此值始终以 0…100% 表示。当处于自动模式下或者处于手动模式下且 CVManLimiting 为真时，CV3 介于 CV3HLimit 和 CV3LLimit 之间；否则，介于 0 和 100% 之间。	
DeltaCV1	REAL	当前 CV1 与上一 CV1 之间的差值（当前 CV1 - 上一 CV1）。	
DeltaCV2	REAL	当前 CV2 与上一 CV2 之间的差值（当前 CV2 - 上一 CV2）。	
DeltaCV3	REAL	当前 CV3 与上一 CV3 之间的差值（当前 CV3 - 上一 CV3）。	
CV1Initializing	BOOL	CV1 的初始化模式指示器。 当 CV1InitReq 或功能块 blockFirstScan 为真，或者 CV1Fault 由真变为假（由不良转为良好）时，该值设置为真。当功能块初始化完毕且 CV1InitReq 不再为真后，CV1Initializing 将设置为假。	
CV2nitializing	BOOL	CV2 的初始化模式指示器。 当 CV2InitReq、功能块 blockFirstScan 或 OLCFirstRun 为真，或者 CV2Fault 由真变为假（由不良转为良好）时，该值设置为真。当功能块初始化完毕且 CV2InitReq 不再为真后，CV2Initializing 将设置为假。	

CV3initializing	BOOL	当 CV3InitReq、功能块 blockFirstScan 或 OLCFirstRun 为真，或者 CV3Fault 由真变为假（由不良转为良好）时，该值设置为真。当功能块初始化完毕且 CV3InitReq 不再为真后，CV3Initializing 将设置为假。	
CV1HAlarm	BOOL	CV1 上限报警指示器。计算的 CV1 值 > 100 或 CV1HLimit 时为真。	
CV12HAlarm	BOOL	CV2 上限报警指示器。计算的 CV2 值 > 100 或 CV2HLimit 时为真。	
CV3HAlarm	BOOL	CV3 上限报警指示器。计算的 CV3 值 > 100 或 CV3HLimit 时为真。	
CV1LAlarm	BOOL	CV1 下限报警指示器。计算的 CV1 值 < 0 或 CV1LLimit 时为真。	
CV2LAlarm	BOOL	CV2 下限报警指示器。计算的 CV2 值 < 0 或 CV2LLimit 时为真。	
CV3LAlarm	BOOL	CV3 下限报警指示器。计算的 CV3 值 < 0 或 CV3LLimit 时为真。	
CV1ROCPosAlarm	BOOL	CV1 变化率报警指示器。计算的 CV1 变化率超出 CV1ROCPosLimit 时为真。	
CV2ROCPosAlarm	BOOL	CV2 变化率报警指示器。计算的 CV2 变化率超出 CV2ROCPosLimit 时为真。	
CV3ROCPosAlarm	BOOL	CV3 变化率报警指示器。计算的 CV3 变化率超出 CV3ROCPosLimit 时为真。	
CV1ROCNegAlarm	BOOL	CV1 变化率报警指示器。计算的 CV1 变化率超出 CV1ROCNegLimit 时为真。	

CV2ROCNegAlarm	BOOL	CV2 变化率报警指示器。计算的 CV2 变化率超出 CV2ROCNegLimit 时为真。	
CV3ROCNegAlarm	BOOL	CV3 变化率报警指示器。计算的 CV3 变化率超出 CV3ROCNegLimit 时为真。	
SP	REAL	当前设置点的值。SP 值用于在自动模式或 PV 跟踪模式下控制 CV ,以 PV 单位标定。	
SPPercent	REAL	以 PV 量程百分比表示的 SP 值。 SPPercent = ((SP – PVEUMin) * 100) / PVSpan。 PV 量程计算：PVSpan = (PVEUMax – PVEUMin)	
SPHAlarm	BOOL	SP 上限报警指示器。当 SP ≥ SPHLimit 时为真。	
SPLAlarm	BOOL	SP 下限报警指示器。当 SP ≤ SPLLimit 时为真。	
PVPercent	REAL	以量程百分比表示的 PV。 PVPercent = ((PV – PVEUMin) * 100) / PVSpan PV 量程计算：PVSpan = (PVEUMax – PVEUMin)	
E	REAL	过程误差。SP 与 PV 之间的差值，以 PV 单位标定。	
EPercent	REAL	以量程百分比形式表示的误差。	
CV1WindupHOut	BOOL	CV1 积分饱和上限指示器。 当达到 SP 上限或 CV1 上限/下限时为真。此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV1 输出的积分饱和。	

CV2WindupHOut	BOOL	CV2 积分饱和上限指示器。当达到 SP 上限或 CV2 上限/下限时为真。此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV2 输出的积分饱和。	
CV3WindupHOut	BOOL	CV3 积分饱和上限指示器。当达到 SP 上限或 CV3 上限/下限时为真。此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV3 输出的积分饱和。	
CV1WindupLOut	BOOL	CV1 积分饱和下限指示器。当达到 SP 或 CV1 上限/下限时为真。此信号通常供 WindupLIn 输入使用 ,用以限制主回路上 CV1 输出的积分饱和。	
CV2WindupLOut	BOOL	CV2 积分饱和下限指示器。当达到 SP 或 CV2 上限/下限时为真。此信号通常供 WindupLIn 输入使用 ,用以限制主回路上 CV2 输出的积分饱和。	
CV3WindupLOut	BOOL	CV3 积分饱和下限指示器。当达到 SP 或 CV3 上限/下限时为真。此信号通常供 WindupLIn 输入使用 ,用以限制主回路上 CV3 输出的积分饱和。	
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。操作员控制模式下为假。	
CV1Auto	BOOL	CV1 的自动模式指示器。当 CV1 处于自动模式时为真。	
CV2Auto	BOOL	CV2 的自动模式指示器。当 CV2 处于自动模式时为真。	
CV2Auto	BOOL	CV3 的自动模式指示器。当 CV3 处于自动模式时为真。	

CV1Manual	BOOL	CV1 的手动模式指示器。当 CV1 处于手动模式时为真。	
CV2Manual	BOOL	CV2 的手动模式指示器。当 CV2 处于手动模式时为真。	
CV3Manual	BOOL	CV3 的手动模式指示器。当 CV3 处于手动模式时为真。	
CV1Override	BOOL	CV1 的超控模式指示器。当 CV1 处于超控模式时为真。	
CV2Override	BOOL	CV2 的超控模式指示器。当 CV2 处于超控模式时为真。	
CV3Override	BOOL	CV3 的超控模式指示器。当 CV3 处于超控模式时为真。	
CV1Hand	BOOL	CV1 的手控模式指示器。当 CV1 处于手控模式时为真。	
CV2Hand	BOOL	CV2 的手控模式指示器。当 CV2 处于手控模式时为真。	
CV3Hand	BOOL	CV3 的手控模式指示器。当 CV3 处于手控模式时为真。	
DeltaT	REAL	两次更新间隔的时间（秒）。	
CV1StepSizeUsed	REAL	调谐中使用的实际 CV1 步长。	
CV2StepSizeUsed	REAL	调谐中使用的实际 CV2 步长。	
CV3StepSizeUsed	REAL	调谐中使用的实际 CV3 步长。	
CV1GainTuned	REAL	调谐完毕后计算出的 CV1 内模增益值。	
CV2GainTuned	REAL	调谐完毕后计算出的 CV2 内模增益值。	
CV3GainTuned	REAL	调谐完毕后计算出的 CV3 内模增益值。	
CV1TCTuned	REAL	调谐完毕后计算出的 CV1 内模时间常数。	
CV2TCTuned	REAL	调谐完毕后计算出的 CV2 内模时间常数。	

CV3TCTuned	REAL	调谐完毕后计算出的 CV3 内模时间常数。	
CV1DTTuned	REAL	调谐完毕后计算出的 CV1 内模死区时间。	
CV2DTTuned	REAL	调谐完毕后计算出的 CV2 内模死区时间。	
CV3DTTuned	REAL	调谐完毕后计算出的 CV3 内模死区时间。	
CV1RespTCTunedS	REAL	调谐完毕后计算出的 CV1 慢速响应速度下的控制变量时间常数。	
CV2RespTCTunedS	REAL	调谐完毕后计算出的 CV2 慢速响应速度下的控制变量时间常数。	
CV3RespTCTunedS	REAL	调谐完毕后计算出的 CV3 慢速响应速度下的控制变量时间常数。	
CV1RespTCTunedM	REAL	调谐完毕后计算出的 CV1 中速响应速度下的控制变量时间常数。	
CV2RespTCTunedM	REAL	调谐完毕后计算出的 CV2 中速响应速度下的控制变量时间常数。	
CV3RespTCTunedM	REAL	调谐完毕后计算出的 CV3 中速响应速度下的控制变量时间常数。	
CV1RespTCTunedF	REAL	调谐完毕后计算出的 CV1 快速响应速度下的控制变量时间常数。	
CV2RespTCTunedF	REAL	调谐完毕后计算出的 CV2 快速响应速度下的控制变量时间常数。	
CV3RespTCTunedF	REAL	调谐完毕后计算出的 CV3 快速响应速度下的控制变量时间常数。	
AtuneCV1On	BOOL	启动 CV1 的自动调谐后设置为真。	

AtuneCV2On	BOOL	启动 CV2 的自动调谐后设置为真。	
AtuneCV3On	BOOL	启动 CV3 的自动调谐后设置为真。	
AtuneCV1Done	BOOL	CV1 的自动调谐成功完成后设置为真。	
AtuneCV2Done	BOOL	CV2 的自动调谐成功完成后设置为真。	
AtuneCV3Done	BOOL	CV3 的自动调谐成功完成后设置为真。	
AtuneCV1Aborted	BOOL	当 CV1 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV2Aborted	BOOL	当 CV2 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV3Aborted	BOOL	当 CV3 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV1Status	DINT	指示 CV1 的调谐状态。	
AtuneCV2Status	DINT	指示 CV2 的调谐状态。	
AtuneCV3Status	DINT	指示 CV3 的调谐状态。	
AtuneCV1Fault	BOOL	CV1 的自动调谐产生以下任一故障。	
AtuneCV2Fault	BOOL	CV2 的自动调谐产生以下任一故障。	
AtuneCV3Fault	BOOL	CV3 的自动调谐产生以下任一故障。	
AtuneCV1PVOutOfLimit	BOOL	在 CV1 自动调谐过程中，PV 或 PV 死区时间步提前预测值超过 PVTuneLimit。该值为真时，CV1 自动调谐过程将中止。	

AtuneCV2PVOutOfLimit	BOOL	在 CV2 自动调谐过程中，PV 或 PV 死区时间步提前预测值超过 PVTuneLimit。该值为真时，CV2 自动调谐过程将中止。	
AtuneCV3PVOutOfLimit	BOOL	在 CV3 自动调谐过程中，PV 或 PV 死区时间步提前预测值超过 PVTuneLimit。该值为真时，CV3 自动调谐过程将中止。	
AtuneCV1ModeInv	BOOL	CC 模式在自动调谐开始时并非手动模式，或者在 CV1 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV1 自动调谐不会启动或者将会中止。	
AtuneCV2ModeInv	BOOL	CC 模式在自动调谐开始时并非手动模式，或者在 CV2 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV2 自动调谐不会启动或者将会中止。	
AtuneCV3ModeInv	BOOL	CC 模式在自动调谐开始时并非手动模式，或者在 CV3 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV3 自动调谐不会启动或者将会中止。	
AtuneCV1WindupFault	BOOL	在 CV1 自动调谐开始时或 CV1 自动调谐期间，CV1WindupHIn 或 CV1WindupLIn 为真。该值为真时，CV1 自动调谐不会启动或者将会中止。	
AtuneCV2WindupFault	BOOL	在 CV1 自动调谐开始时或 CV2 自动调谐期间，CV2WindupHIn 或 CV2WindupLIn 为真。该值为真时，CV2 自动调谐不会启动或者将会中止。	

AtuneCV3WindupFault	BOOL	在 CV3 自动调谐开始时或 CV3 自动调谐期间，CV3WindupHIn 或 CV3WindupLIn 为真。该值为真时，CV3 自动调谐不会启动或者将会中止。	
AtuneCV1StepSize0	BOOL	CV1 自动调谐开始时 CV1StepSizeUsed = 0。该值为真时，CV1 自动调谐不会启动。	
AtuneCV2StepSize0	BOOL	CV2 自动调谐开始时 CV2StepSizeUsed = 0。该值为真时，CV2 自动调谐不会启动。	
AtuneCV3StepSize0	BOOL	CV3 自动调谐开始时 CV3StepSizeUsed = 0。该值为真时，CV3 自动调谐不会启动。	
AtuneCV1LimitsFault	BOOL	在 CV1 自动调谐开始时或 CV1 自动调谐期间，CV1LimitsInv 和 CVManLimiting 为真。该值为真时，CV1 自动调谐不会启动或者将会中止。	
AtuneCV2LimitsFault	BOOL	在 CV2 自动调谐开始时或 CV2 自动调谐期间，CV2LimitsInv 和 CVManLimiting 为真。该值为真时，CV2 自动调谐不会启动或者将会中止。	
AtuneCV3LimitsFault	BOOL	在 CV3 自动调谐开始时或 CV3 自动调谐期间，CV3LimitsInv 和 CVManLimiting 为真。该值为真时，CV3 自动调谐不会启动或者将会中止。	

AtuneCV1InitFault	BOOL	在 CV1 自动调谐开始时或 CV1 自动调谐期间，CV1Initializing 为真。该值为真时，CV1 自动调谐不会启动或者将会中止。	
AtuneCV2InitFault	BOOL	在 CV2 自动调谐开始时或 CV2 自动调谐期间，CV2Initializing 为真。该值为真时，CV2 自动调谐不会启动或者将会中止。	
AtuneCV3InitFault	BOOL	在 CV3 自动调谐开始时或 CV3 自动调谐期间，CV3Initializing 为真。该值为真时，CV3 自动调谐不会启动或者将会中止。	
AtuneCV1EUSpanChanged	BOOL	在 CV1 自动调谐期间，CV1EUSpan 或 PVEUSpan 发生改变。该值为真时，CV1 自动调谐过程将中止。	
AtuneCV2EUSpanChanged	BOOL	在 CV2 自动调谐期间，CV2EUSpan 或 PVEUSpan 发生改变。该值为真时，CV2 自动调谐过程将中止。	
AtuneCV3EUSpanChanged	BOOL	在 CV3 自动调谐期间，CV3EUSpan 或 PVEUSpan 发生改变。该值为真时，CV3 自动调谐过程将中止。	
AtuneCV1Changed	BOOL	CV1Oper（操作员控制模式）或 CV1Prog（程序控制模式）发生改变，或者 CV1 在 CV1 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV1 自动调谐过程将中止。	

AtuneCV2Changed	BOOL	CV2Oper（操作员控制模式）或 CV2Prog（程序控制模式）发生改变，或者 CV2 在 CV2 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV2 自动调谐过程将中止。	
AtuneCV3Changed	BOOL	CV3Oper（操作员控制模式）或 CV3Prog（程序控制模式）发生改变，或者 CV3 在 CV3 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV3 自动调谐过程将中止。	
AtuneCV1Timeout	BOOL	自阶跃测试开始起经过的时间长于 AtuneTimeLimit。该值为真时，CV1 自动调谐过程将中止。	
AtuneCV2Timeout	BOOL	自阶跃测试开始起经过的时间长于 AtuneTimeLimit。该值为真时，CV2 自动调谐过程将中止。	
AtuneCV3Timeout	BOOL	自阶跃测试开始起经过的时间长于 AtuneTimeLimit。该值为真时，CV3 自动调谐过程将中止。	
AtuneCV1PVNotSettled	BOOL	PV 变化过大而无法进行 CV1 自动调谐。该值为真时，CV1 自动调谐过程将中止。等待 PV 达到更稳定状态再进行 CV1 自动调谐。	
AtuneCV2PVNotSettled	BOOL	PV 变化过大而无法进行 CV2 自动调谐。该值为真时，CV2 自动调谐过程将中止。等待 PV 达到更稳定状态再进行 CV2 自动调谐。	

AtuneCV3PVNotSettled	BOOL	PV 变化过大而无法进行 CV3 自动调谐。该值为真时，CV3 自动调谐过程将中止。等待 PV 达到更稳定状态再进行 CV3 自动调谐。	
Status1	DINT	功能块的位映射状态。	
Status2	DINT	功能块的附加位映射状态。	
Status3CV1	DINT	功能块的附加位映射 CV1 状态。值为 0 时表示未发生故障。	
Status3CV2	DINT	功能块的附加位映射 CV2 状态。值为 0 时表示未发生故障。	
Status3CV3	DINT	功能块的附加位映射 CV3 状态。值为 0 时表示未发生故障。	
InstructFault	BOOL	功能块发生故障。指示 Status1、Status2 和 Status3CV(n) 相应位的状态，其中 (n) 可以是 1、2 或 3。 值为 0 时表示未发生故障。任何可能配置为无效值的参数都必须具有状态参数，来指示其无效状态。	
PVFaulted	BOOL	过程变量 PV 状况不良。	
PVSpanInv	BOOL	PV 量程无效，PVEUMax < PVEUMin。	
SPProgInv	BOOL	SPProg < SPLLimit 或 > SPHLimit。限制 SP 的值。	
SPOperInv	BOOL	SPOper < SPLLimit 或 > SPHLimit。限制 SP 的值。	
SPLimitsInv	BOOL	限值无效：SPLLimit < PVEUMin，SPHLimit > PVEUMax 或 SPHLimit < SPLLimit。如果 SPHLimit < SPLLimit，则将使用 SPLLimit 限制该值。	
SampleTimeTooSmall	BOOL	模型死区时间/DeltaT 必须小于或等于 200。	

FactorInv	BOOL	输入的 Factor 值 < 0。	
TimingModeInv	BOOL	输入的 TimingMode 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	
RTSMissed	BOOL	仅用于实时采样模式。ABS(DeltaT – RTSTime) > 1 毫秒时为真。	
RTSTimeInv	BOOL	输入的 RTSTime 无效。	
RTSTimeStampInv	BOOL	RTSTimeStamp 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	
DeltaTInv	BOOL	DeltaT 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	
CV1Faulted	BOOL	控制变量 CV1 状况不良。	
CV2Faulted	BOOL	控制变量 CV2 状况不良。	
CV3Faulted	BOOL	控制变量 CV3 状况不良。	
CV1HandFBFaulted	BOOL	CV1 HandFB 值状况不良。	
CV2HandFBFaulted	BOOL	CV2 HandFB 值状况不良。	
CV3HandFBFaulted	BOOL	CV3 HandFB 值状况不良。	
CV1ProgInv	BOOL	CV1Prog < 0 或 > 100，或者当 CVManLimiting 为真时，< CV1LLimit 或 > CV1HLimit。将限制 CV1 的值。	
CV2ProgInv	BOOL	CV2Prog < 0 或 > 100，或者当 CVManLimiting 为真时，< CV2LLimit 或 > CV2HLimit。将限制 CV2 的值。	
CV3ProgInv	BOOL	CV3Prog < 0 或 > 100，或者当 CVManLimiting 为真时，< CV3LLimit 或 > CV3HLimit。将限制 CV3 的值。	
CV1OperInv	BOOL	CV1Oper < 0 或 > 100，或者当 CVManLimiting 为真时，< CV1LLimit 或 > CV1HLimit。将限制 CV1 的值。	

CV2OperInv	BOOL	CV2Oper < 0 或 > 100，或者当 CVManLimiting 为真时，< CV2LLimit 或 > CV2HLimit。将限制 CV2 的值。	
CV3OperInv	BOOL	CV3Oper < 0 或 > 100，或者当 CVManLimiting 为真时，< CV3LLimit 或 > CV3HLimit。将限制 CV3 的值。	
CV1OverrideValueInv	BOOL	CV1OverrideValue < 0 或 > 100 。将限制 CV1 的值。	
CV2OverrideValueInv	BOOL	CV2OverrideValue < 0 或 > 100 。将限制 CV2 的值。	
CV3OverrideValueInv	BOOL	CV3OverrideValue < 0 或 > 100 。将限制 CV3 的值。	
CV1TrackValueInv	BOOL	输入的 CV1TrackValue < 0 或 > 100。将限制 CV1 的值。	
CV2TrackValueInv	BOOL	输入的 CV2TrackValue < 0 或 > 100。将限制 CV2 的值。	
CV3TrackValueInv	BOOL	输入的 CV3TrackValue < 0 或 > 100。将限制 CV3 的值。	
CV1EUSpanInv	BOOL	CV1EU 的量程无效，CV1EUMax 等于 CV1EUMin。	
CV2EUSpanInv	BOOL	CV2EU 的量程无效，CV2EUMax 等于 CV2EUMin。	
CV3EUSpanInv	BOOL	CV3EU 的量程无效，CV3EUMax 等于 CV3EUMin。	
CV1LimitsInv	BOOL	CV1LLimit < 0、CV1HLimit > 100 或 CV1HLimit <= CV1LLimit。如果 CV1HLimit <= CV1LLimit，则将使用 CV1LLimit 限制 CV1 值。	
CV2LimitsInv	BOOL	CV2LLimit < 0、CV2HLimit > 100 或 CV2HLimit <= CV2LLimit。如果 CV2HLimit <= CV2LLimit，则将使用 CV2LLimit 限制 CV2 值。	

CV3LimitsInv	BOOL	CV3LLimit < 0、CV3HLimit > 100 或 CV3HLimit <= CV3LLimit。如果 CV3HLimit <= CV3LLimit，则将使用 CV3LLimit 限制 CV3 值。	
CV1ROCLimitInv	BOOL	CV1ROCLimit < 0，禁用 CV1 ROC 限制。	
CV2ROCLimitInv	BOOL	CV2ROCLimit < 0，禁用 CV2 ROC 限制。	
CV3ROCLimitInv	BOOL	CV3ROCLimit < 0，禁用 CV3 ROC 限制。	
CV1HandFBInv	BOOL	CV1HandFB < 0 或 > 100。将限制 CV1 的值。	
CV2HandFBInv	BOOL	CV2HandFB < 0 或 > 100。将限制 CV2 的值。	
CV3HandFBInv	BOOL	CV3HandFB < 0 或 > 100。将限制 CV3 的值。	
CV1ModelGainInv	BOOL	CV1ModelGain 为 1.#QNAN 或 -1.#IND（非数字），或者 ± 1.$（无穷大 ∞）。	
CV2ModelGainInv	BOOL	CV2ModelGain 为 1.#QNAN 或 -1.#IND（非数字），或者 ± 1.$（无穷大 ∞）。	
CV3ModelGainInv	BOOL	CV3ModelGain 为 1.#QNAN 或 -1.#IND（非数字），或者 ± 1.$（无穷大 ∞）。	
CV1ModelTCInv	BOOL	CV1ModelTC < 0。	
CV2ModelTCInv	BOOL	CV2ModelTC < 0。	
CV3ModelTCInv	BOOL	CV3ModelTC < 0。	
CV1ModelDTInv	BOOL	CV1ModelDT < 0。	
CV2ModelDTInv	BOOL	CV2ModelDT < 0。	
CV3ModelDTInv	BOOL	CV3ModelDT < 0。	
CV1RespTCInv	BOOL	CV1RespTC < 0。	
CV2RespTCInv	BOOL	CV2RespTC < 0。	
CV3RespTCInv	BOOL	CV3RespTC < 0。	
CV1TargetInv	BOOL	CV1Target < 0 或 > 100。	

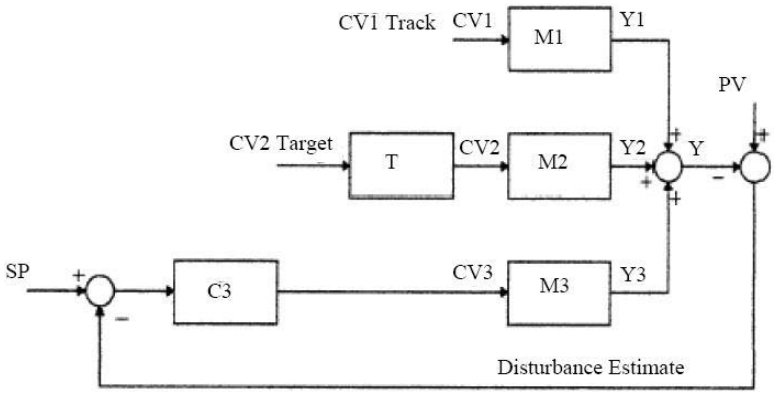
CV2TargetInv	BOOL	CV2Target < 0 或 > 100。	
CV3TargetInv	BOOL	CV3Target < 0 或 > 100。	

说明

协调控制是一种基于模型的灵活算法，可用在各种配置中，例如：

- 使用三个控制变量控制一个过程变量
- 热冷分程控制
- 前馈控制
- 区域温度控制
- 约束控制

下面是一个协调控制闭环配置的示例。



在该示例中，CV1 处于手动模式，CV2 驱动到其目标值，CV3 为主动控制。下表详细描述了该示例。

名称	说明
CV1	处于手动模式
CV2	驱动到其目标值 (CV2 = Target1stCV)
CV3	主动控制 (CV3 = Act1stCV)

该示例可以是一个具有前馈功能的冷热控制系统，其中：

- CV1 为前馈；
- CV2 为冷却；
- CV3 为加热。

由于 CV1 处于手动模式，因此无法实现作为最低优先级目标的 CV3 目标值。PV 将通过 CV3 维持在设置点，同时 CV2 将驱动到其目标值（第二优先级目标）。

如果操作员更改 CV1 手动值，则在计算新的 CV3 和 CV2 时，控制变量将考虑该变化量。

M1	CV1 - 具有死区时间模型的 PV 一阶滞后
M2	CV2 - 具有死区时间模型的 PV 一阶滞后
M3	CV3 - 具有死区时间模型的 PV 一阶滞后
T	目标响应
C3	基于模型的算法，通过 CV3 控制 PV
Y1、Y2 和 Y3	M1、M2 和 M3 的模型输出
Y	PV 预测值

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

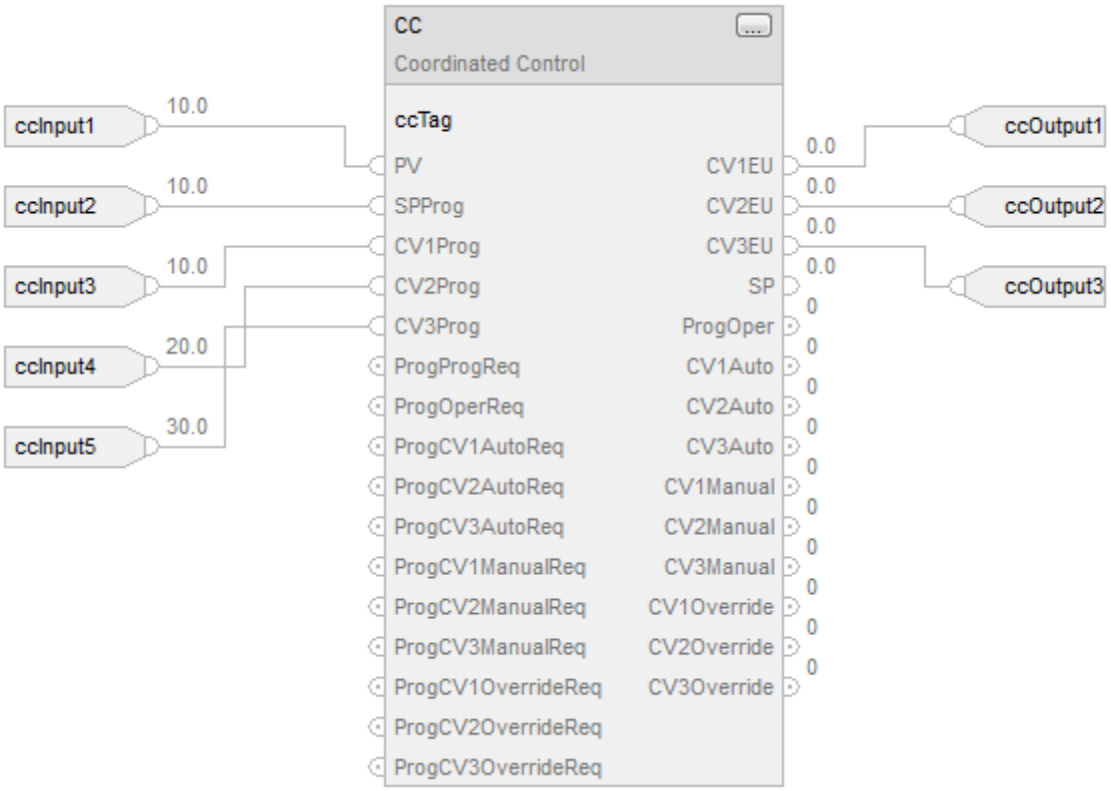
条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	N/A。
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

功能块



结构化文本

```
ccTag.PV := ccInput1;

ccTag.SPProg := ccInput2;

ccTag.CV1Prog := ccInput3;

ccTag.CV2Prog := ccInput4;

ccTag.CV3Prog := ccInput5;
```

```
CC(ccTag);

ccOutput1 := ccTag.CV1EU;

ccOutput2 := ccTag.CV2EU;

ccOutput3 := ccTag.CV3EU;
```

另请参见

通用属性 参考页数 589

CC 功能块配置 参考页数 210

选择控制变量 参考页数 306

在程序控制与操作员控制之间切换 参考页数 301

结构化文本语法 参考页数 559

功能块属性 参考页数 545

CC 功能块配置

从默认配置开始，配置以下参数：

参数	说明
PVEUMax	PV 的最大标定值。
PVEUMin	PV 的最小标定值。
SPHLimit	SP 上限值，以 PV 单位标定。
PPLLimit	SP 下限值，以 PV 单位标定。
CV1InitValue	控制变量 CV1 输出的初始值。
CV2InitValue	控制变量 CV2 输出的初始值。
CV3InitValue	控制变量 CV3 输出的初始值。

如果已有过程模型可用，可输入以下参数的值，直观地对 CC 控制变量进行调谐。

参数	说明
ModelGains	非零数（负数表示直接作用的控制变量，正数表示反向作用的控制变量）
ModelTimeConstants	始终为正数

ModelDeadtimes	始终为正数
ResponseTimeConstants	始终为正数
第一个、第二个和第三个主动 CV	指定使用 CV 补偿 PV - SP 误差的顺序。
第一个、第二个和第三个目标 CV	指定将 CV 驱动至相应目标值采用的优先级。
CVTargetValues	指定控制变量驱动各 CV 达到的值。
TargetRespTC	指定 CV 接近目标值的速度

对于 CV 主动和目标列表与 CV 自动-手动模式的任意组合，功能块都以定义的方式进行工作。功能块尝试按照以下优先级顺序完成以下目标：

1. 控制 PV，使其达到 SP
2. 控制 Target1stCV，使其达到目标值
3. 控制 Target2ndCV，使其达到目标值

如果将任何 CV 置于手动模式，CC 功能块会放弃优先级为 3 的目标。如果两个 CV 处于手动模式，CC 功能块将降级为 IMC（单输入、单输出），控制变量会控制 PV，使其达到设置点。

但除此之外，控制变量还会从处于手动模式的 CV 中读取手动 CV 值作为前馈信号。随后，CC 功能块会使用适当的内部模型预测手动 CV 值对 PV 的影响，并计算仍处于自动模式的第三个 CV。

对于积分过程类型（例如液位控制和位置控制），使用内部非积分模型来近似模拟积分过程。使用 Factor 参数将确定的积分过程模型转换为用于 CV 计算的非积分内部模型。这对于稳定的功能块执行非常必要。

CC 功能块模型初始化

在以下情况下会进行模型初始化：

- 首次扫描功能块期间
- ModelInit 请求参数置位时
- DeltaT 发生变化时

您可能需要手动调整内模参数或响应时间常数。为此，可更改相应参数并设置适当的 ModelInit 位。控制变量的内部状态将初始化，相应位将自动复位。

例如，可修改 CV2 - PV 模型的模型增益。可将 ModelInit2 参数设置为真，以初始化 CV2 - PV 内模参数并使新的模型增益生效。

CC 功能块调谐

功能块配有内部调谐器（建模器）。调谐器的用途是识别过程模型参数并将这些参数用作内模参数（增益、时间常数和死区时间）。调谐器还将计算最佳的响应时间常数。设置调谐器时，可配置每个 CV - PV 过程的以下参数。

ProcessType	积分（液位、位置控制）或非积分（流量、压力控制）
ProcessGainSign	置位表示过程增益为负值（输出增大会导致 PV 减小）；复位表示过程增益为正值（输出增大会导致 PV 增大）。
ResponseSpeed	慢速、中速或快速，取决于控制目的。
NoiseLevel	PV 的估计噪声等级（低、中、高），使调谐器能够区分哪些 PV 变化是随机噪声，哪些是由 CV 阶跃变化引起的。
StepSize	非零正数或负数，分别定义正向或负向的 CV 阶跃变化幅值。
PVTuneLimit	（仅适用于积分过程类型）采用 PV 工程单位，定义容许的因 CV 变化导致的 PV 变化量，超出此限制时会中止调谐测试。

通过将 AtuneStart 位（例如 AtuneCV1Start）置位，可启动调谐器。要停止调谐，可将 AtuneAbort 位置位。

调谐成功完成后，相应的 GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数会根据调谐结果进行更新，而且 AtuneStatus 代码会置位，指示调谐完成。

可将 AtuneUseModel 位置位，通过这种方式分别将这些参数复制到 ModelGain、ModelTC 和 ResponseTC。控制变量会自动对内部变量进行初始化并继续正常工作。它会自动将 AtuneUseModel 位复位。

另请参见

CC 功能块调谐过程 参考页数 212

CC 功能块调谐错误

如果调谐过程中发生错误，调谐会中止，相应的 AtuneStatus 位会置位。要中止调谐，可将 AtuneAbort 参数置位。

中止后，CV 会采用阶跃变化之前的值，GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数不会更新。AtuneStatus 参数将指示中止的原因。

CC 功能块调谐过程

按照以下步骤配置调谐器。

1. 将全部三个 CV 参数设为手动模式。
2. 将 AtuneStart 参数置位。

   调谐器开始采集 PV 和 CV 数据进行噪声计算。

3. 采集 60 个样本 (60*DeltaT) 过后，调谐器会将 StepSize 与 CV 相加。

   成功采集 CV 步更改生成的 PV 数据后，CV 会采用阶跃变化之前的值，AtuneStatus、GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数会更新。

4. 将 AtuneUseModel 参数置位，从而将经过调谐的参数复制到模型参数中

   功能块随即会将 AtuneUseModel 参数复位。

   成功执行自动调谐后，Atune 参数会置 1。调谐成功完成。

要确定全部三个 CV-PV 过程的模型并计算响应时间常数，可将调谐器运行达三次，从而分别获取 CV1-PV、CV2-PV 和 CV3-PV 模型并实现调谐。

内模控制 (IMC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

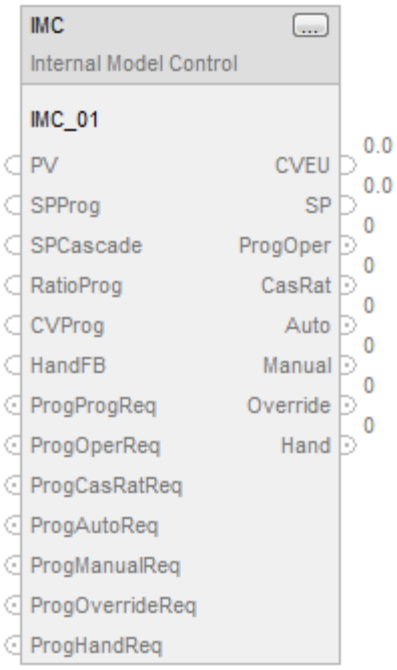
IMC 指令通过操纵单个控制变量输出来控制单个过程变量。该功能块执行一种算法，将实际误差信号与过程内部一阶加纯滞后模型的误差信号进行比较。在自动模式下，IMC 功能块基于 PV 与 SP 的偏差、内部模型和调谐来计算控制变量输出 (CV)。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

IMC(IMC_tag);

操作数

功能块

操作数：	类型：	格式	说明：
IMC tag	INTERNAL MODEL CONTROL	结构	IMC 结构

结构化文本

操作数：	类型：	格式	说明：
IMC tag	INTERNAL MODEL CONTROL	结构	IMC 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

重要事项： 只要 APC 块检测到增量时间 (DeltaT) 发生变化，便执行 ModelInit。因此，这些块应该仅在 DeltaT 为常量的一种 TimingMode 下运行。

- TimingMode = 0（周期性），在周期性任务中执行这些功能块时
- TimingMode = 1（过采样）

无论在哪种情况下，如果周期性任务的时间动态变化，或者 OversampleDT 动态变化，该块便执行 ModelInit。

由于 DeltaT 中存在抖动，因此不推荐使用下面的 TimingMode 设置：

- TimingMode = 0（周期性），在连续任务或事件任务中执行这些功能块时
- TimingMode = 2（实时采样）

结构

输入参数	数据类型	说明	有效值和默认值
EnableIn	BOOL	使能输入。如果为"假"，则功能块不执行，输出也不会更新。	默认值 = 真
PV	REAL	经过标定的过程变量输入。该值通常从模拟量输入模块中读取。	有效值 = 任意浮点值 默认值 = 0.0
PVFault	BOOL	PV 不良状况指示器。如果从模拟量输入读取 PV，则 PVFault 通常由模拟量输入故障状态控制。 如果 PVFault 为"真"，则表示输入模块发生错误，并且会将 Status 中的相应位置位。	默认值 = 假 假 = 状况良好
PVUEMax	REAL	PV 的最大标定值。对应于过程变量的 100% 量程的 PV 和 SP 的值。如果 PVEUMax ≤ PVEUMin，则将 Status 中的相应位置位。	有效值 = PVEUMin < PVEUMax ≤ 最大正浮点值 默认值 = 100.0
PVUEMin	REAL	PV 的最小标定值。对应于过程变量的 0% 量程的 PV 和 SP 的值。如果 PVEUMax ≤ PVEUMin，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 ≤ PVEUMin < PVEUMax 默认值 = 0.0

SPProg	REAL	SP 程序值，以 PV 单位标定。在程序控制模式下，SP 设置为此值。 如果 SPProg 或 SPOper 的值 < SPLLimit 或 > SPHLimit，则将 Status 中的相应位置位，并限制 SP 的值。	有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPOper	REAL	SP 操作员值，以 PV 单位标定。SP 设置为此值，前提是处于操作员控制模式时。 如果 SPProg 或 SPOper 的值 < SPLLimit 或 > SPHLimit，则将 Status 中的相应位置位，并限制 SP 的值。	有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPCascade	REAL	SP 级联值，以 PV 单位标定。如果级联/比率模式和 UseRatio 为假，则 SP 设置为此值，这通常是主回路的 CVEU。如果级联/比率模式和 UseRatio 为真，则 SP 设置为此值乘以 Ratio。 如果 SPCascade 值 < SPLLimit 或 > SPHLimit，则会将 Status 中的相应位置位，并限制 SP 的值。	有效值 = SPLLimit 到 SPHLimit 默认值 = 0.0
SPHLimit	REAL	SP 上限值，以 PV 单位标定。 如果 SPHLimit < SPLLimit 或 SPHLimit > PVEUMax，则将 Status 中的相应位置位。	有效值 = SPLLimit 到 PVEUMax 默认值 = 100.0
SPLLimit	REAL	SP 下限值，以 PV 单位标定。 如果 SPLLimit < PVEUMin 或 SPHLimit < SPLLimit，则将 Status 中的相应位置位，并使用 SPLLimit 的值限制 SP。	有效值 = PVEUMin 到 SPHLimit 默认值 = 0.0
UseRatio	BOOL	允许比率控制。该值为真时，可在级联/比率模式下启用比率控制。	默认值 = 假

RatioProg	REAL	程序控制模式下的比率乘数，无单位（例如标量）。在程序控制模式下 ,Ratio 和 RatioOper 设置为此值。 如果 RatioProg 或 RatioOper < RatioLLimit 或 > RatioHLimit ,则会将 Status 中的相应位置位，并限制 Ratio 值。	有效值 = RatioLLimit 到 RatioHLimit 默认值 = 1.0
RatioOper	REAL	操作员控制模式下的比率乘数，无单位（例如标量）。在操作员控制模式下，Ratio 设置为此值。 如果 RatioProg 或 RatioOper < RatioLLimit 或 > RatioHLimit，则会将 Status 中的相应位置位，并限制 Ratio 值。	有效值 = RatioLLimit 到 RatioHLimit 默认值 = 1.0
RatioHLimit	REAL	比率上限值，无单位（例如标量）。限制从 RatioProg 或 RatioOper 获取的 Ratio 值。 如果 RatioLLimit < 0 ,则会将 Status 中的相应位置位，并将值限定为零。如果 RatioHLimit < RatioLLimit，则会将 Status 中的相应位置位，并使用 RatioLLimit 值限制 Ratio 值。	有效值 = RatioLLimit 到最大正浮点值 默认值 = 1.0
RatioLLimit	REAL	比率下限值，无单位（例如标量）。限制从 RatioProg 或 RatioOper 获取的 Ratio 值。 如果 RatioLLimit < 0 ,则会将 Status 中的相应位置位，并将值限定为零。如果 RatioHLimit < RatioLLimit，则会将 Status 中的相应位置位，并使用 RatioLLimit 值限制 Ratio 值。	有效值 = 0.0 到 RatioHLimit 默认值 = 1.0
CVFault	BOOL	控制变量不良状况指示器。如果 CVEU 控制模拟量输出，则 CVFault 通常来自于模拟量输出的故障状态。 如果 CVFault 为真，则表示输出模块存在错误 ,会将 Status 中的相应位置位。	默认值 = 假 假 = 状况良好

CVInitReq	BOOL	CV 初始化请求。当为真时，会将 CVEU 设置为 CVInitValue 的值。此信号通常由受 CVEU 控制的模拟量输出模块的 In Hold 状态控制，或者来自次级 IMC 回路的 InitPrimary 输出。	默认值 = 假
CVInitValue	REAL	CVEU 初始化值，以 CVEU 单位标定。当 CVInitializing 为真时，将 CVEU 设置为 CVInitValue 且 CV 等于相应的百分比值。CVInitValue 通常来自于受 CVEU 控制的模拟量输出的反馈，或者次级回路的设置点。当 CVFaulted 或 CVEUSpanInv 为真（不良）时，会禁用功能块初始化。	有效值 = 任意浮点值 默认值 = 0.0
CVProg	REAL	程序-手动模式下的 CV 值。在程序控制和手动模式下，CV 设置为此值。 如果 CVProg 或 CVOper 的值 < 0 或 > 100，或者当 CVManLimiting 为真时 < CVLLimit 或 > CVHLimit，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CVOper	REAL	操作员-手动模式下的 CV 值。在操作员控制和手动模式下，CV 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CVOper 设置为 CV 值。 如果 CVProg 或 CVOper 的值 < 0 或 > 100，或者当 CVManLimiting 为真时 < CVLLimit 或 > CVHLimit，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CVOverrideValue	REAL	超控模式下的 CV 值。在超控模式下，CV 设置为此值。 此值应对应于 IMC 回路的安全状态输出。如果 CVOverrideValue 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0

CVTrackValue	REAL	CV 跟踪值。当启用 CVTrackReq 且 IMC 功能块处于手动模式时，将忽略 CVTrackValue 值，IMC 内部模型将以 CVOper 或 CVProg 值更新其历史数据。当启用 CVTrackReq 且 IMC 功能块处于自动模式时，内部模型将根据 CVTrackValue 的值更新其历史数据。这种情况下，CV 值将可以正常变化，就如同 IMC 功能块仍在控制该过程。在多回路选择方案中，如果希望 IMC 功能块跟随不同控制算法的输出，则该功能将非常有用，可将控制算法的输出连接到 CVTrackValue。	有效值 = 0.0 到 100.0 默认值 = 0.0
CVManLimiting	BOOL	在手动模式下限制 CV 的请求。在手动模式下，如果 CVManLimiting 为真，则 CV 由 CVHLimit 和 CVLLimit 值限制。	默认值 = 假
CVEUMax	REAL	CVEU 的最大值。对应于 100% CV 的 CVEU 值。如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CVEUMin	REAL	CVEU 的最小值。对应于 0% CV 的 CVEU 值。如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0
CVHLimit	REAL	CV 上限值。该值用于设置 CVHAlarm 输出。当处于自动或级联/比率模式，或者处于手动模式并且 CVManLimiting 设置为真时，该值也用于对 CV 值进行限制。 如果 CVLLimit < 0、CVHLimit > 100 或 CVHLimit < CVLLimit，则将 Status 中的相应位置位。如果 CVHLimit < CVLLimit，则使用 CVLLimit 值限制 CV 值。	有效值 = CVLLimit < CVHLimit ≤ 100.0 默认值 = 100.0

CVLLimit	REAL	CV 下限值。该值用于设置 CVLAlarm 输出。当处于自动或级联/比率模式，或者处于手动模式并且 CVManLimiting 设置为真时，该值也用于对 CV 值进行限制。 如果 CVLLimit < 0、CVHLimit > 100 或 CVHLimit < CVLLimit，则将 Status 中的相应位置位。 如果 CVHLimit < CVLLimit，则使用 CVLLimit 值限制 CV 值。	有效值 = 0.0 ≤ CVLLimit < CVHLimit 默认值 = 0.0
CVROCPosLimit	REAL	CV 递增变化率限值（百分比/秒）。 仅当处于自动或级联/比率模式，或者处于手动模式并且 CVManLimiting 设置为真时，才使用变化率限制。 该值为零值时，禁用 CV ROC 限制。 如果 CVROCPOSLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CVROCNegLimit	REAL	CV 递减变化率限值（百分比/秒）。 仅当处于自动或级联/比率模式，或者处于手动模式并且 CVManLimiting 设置为真时，才使用变化率限制。 该值为零值时，禁用 CV ROC 限制。 如果 CVROCNegLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0

HandFB	REAL	CV HandFeedback 值。当处于手动模式下且 HandFBFault 为假(状况良好) 时，将 CV 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。 如果 HandFB 值 < 0 或 > 100，指令将 Status 中的相应位置位，并限制 CV 的值。	有效值 = 0.0...100.0 默认值 = 0.0
HandFBFault	BOOL	HandFB 值不良状况指示器。如果从模拟量输入读取 HandFB 值，则 HandFBFault 通常由模拟量输入通道的状态控制。 HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。	默认值 = 假 假 = 状况良好
WindupHIn	BOOL	积分饱和上限请求。该值为真时，不允许 CV 值增大。该信号通常来自次级回路的 WindupHOut 输出。	默认值 = 假
WindupIIn	BOOL	积分饱和下限请求。该值为真时，不允许 CV 值减小。该信号通常来自次级回路的 WindupLOut 输出。	默认值 = 假
GainEUSpan	BOOL	ModelGain 单位 (EU 或量程百分比)。 CV ModelGain 单位 (EU 或量程百分比)。置位时将 ModelGain 表示为 EU，复位时将 ModelGain 表示为量程百分数。	默认值 = 假 真 = EU 形式的增益 假 = % 形式的增益
ProcessGainSign	BOOL	仅用于自动调谐。过程增益符号 (Delta PV/Delta CV)。 置位指示负过程增益 (输出增大会导致 PV 减小)。 复位指示正过程增益 (输出增大会导致 PV 增大)。	默认值 = 假
ProcessType	DINT	过程类型选项(1 = 积分 ,0 = 非积分)	默认值 = 0

ModelGain	REAL	内模增益参数。根据过程方向输入正增益或负增益。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 = 0.0
ModelTC	REAL	内模时间常数（秒）。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
ModelDT	REAL	内模死区时间（秒）。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
RespTC	REAL	用于确定控制变量操作速度的调谐参数（秒）。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
PVTracking	BOOL	SP 跟踪 PV 请求。设置为真可以使 SP 跟踪 PV。级联/比率模式或自动模式下忽略该值。	默认值 = 假
CVTrackReq	BOOL	CV 跟踪请求。若设置为真，则在自动调谐设置为“关”时启用 CV 跟踪。手控和超控模式下忽略该参数。	默认值 = 假
AllowCasRat	BOOL	允许级联/比率模式。该值为真时，允许通过 ProgCasRatReq 或 OperCasRatReq 选择级联/比率模式。	默认值 = 假
ManualAfterInit	BOOL	初始化后进入手动模式的请求。 该值为真时 如果 CVInitializing 设置为真，除非当前模式为超控或手控模式，否则功能块将切换为手动模式。 ManualAfterInit 设置为假时，功能块的模式保持不变。	默认值 = 假

ProgProgReq	BOOL	程序发出的程序控制请求。 由用户程序设置为真请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。如果该值保持为真且 ProgOperReq 为假 ,则可将功能块锁定在程序控制模式。 当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假
ProgOperReq	BOOL	程序发出的操作员控制请求。由用户程序设置为真可请求操作员控制模式。如果该值保持为真，则可将功能块锁定在操作员控制模式。当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCasRatReq	BOOL	程序-级联/比率模式请求。由用户程序设置为真可请求级联/比率模式。当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgAutoReq	BOOL	程序-自动模式请求。由用户程序设置为真可请求自动模式。当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假
ProgManualReq	BOOL	程序-手动模式请求。由用户程序设置为真可请求手动模式。当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假
ProgOverrideReq	BOOL	程序-超控模式请求。由用户程序设置为真可请求超控模式。当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假
ProgHandReq	BOOL	程序-手控模式请求。由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假

OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。功能块将此参数复位（假）。	默认值 = 假
OperOperReq		操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。功能块将此参数复位（假）。	默认值 = 假
OperCasRatReq	BOOL	操作员-级联/比率模式请求。由操作员界面设置为真可请求级联/比率模式。功能块将此参数复位（假）。	默认值 = 假
OperAutoReq	BOOL	操作员-自动模式请求。由操作员界面设置为真可请求自动模式。功能块将此参数复位(假)。	默认值 = 假
OperManualReq	BOOL	操作员-手动模式请求。由操作员界面设置为真可请求手动模式。功能块将此参数复位(假)。	默认值 = 假
ProgValueReset	BOOL	将程序控制值复位。该值为真时 ,Prog_xxx_Req 输入复位(假)。 该值为真且处于程序控制模式下时 ,将 SPProg 设置为等于 SP ,将 CVProg 设置为等于 CV。 当 ProgValueReset 为真时 ,功能块将此参数复位（假）。	默认值 = 假
TimingMode	DINT	选择时基执行模式。 值/说明 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的更多信息，请参见"功能块属性"部分。	有效值 = 0...2 默认值 = 0
OverSampleDT	REAL	过采样模式的执行时间。	有效值 = 0 到 TON_Timer 经过的最长时间（4194.303 秒） 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期。	有效值 = 1 到 32,767 1 次计数 = 1 ms

RTSTimeStamp	DINT	实时采样模式的模块时戳值。	有效值 = 0 到 32,767（从 32,767 跳回到 0） 1 次计数 = 1 ms
PVTuneLimit	REAL	以 PV 单位标定的 PV 调谐限值。当自动调谐正在运行并且预测 PV 值超过此限值时，调谐过程将中止。	范围：任意浮点值 默认值 = 0
AtuneTimeLimit	REAL	CV 发生阶跃变化后完成自动调谐所需的最长时间。自动调谐时间超出此时间时，调谐将会中止。	有效值范围：任意 > 0 的浮点值。 默认值 = 60 分钟
NoiseLevel	DINT	PV 的噪声级别估计值，在调谐期间可对该值进行补偿。 可选项包括：0 = 低、1 = 中、2 = 高	范围：0 到 2 默认值 = 1
CVStepSize	REAL	调谐阶跃测试所用的 CV 步长（百分比）。步长将直接加到 CV（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%
ResponseSpeed	DINT	所需的闭环响应速度。 慢速响应：ResponseSpeed=0 中速响应：ResponseSpeed=1 快速响应：ResponseSpeed=2。 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0 到 2 默认值 = 1
ModelInit	BOOL	内模初始化开关。	默认值 = 假
Factor	REAL	非积分模型近似因子。仅用于积分过程类型。	默认值 = 100
AtuneStart	BOOL	启动自动调谐的请求。该值为真时，启动功能块的自动调谐。如果 IMC 未处于手动模式，则忽略该值。功能块将此参数复位（假）。	默认值 = 假
AtuneUseModel	BOOL	使用自动调谐模型请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数设置为假。	默认值 = 假

AtuneAbort	BOOL	中止自动调谐的请求。该值为真时，中止 IMC 功能块的自动调谐。功能块将该输入参数设置为假。	默认值 = 假

输出参数	数据类型	说明	有效值和默认值
EnableOut	BOOL	指示指令是否处于启用状态。如果 CVEU 溢出，则设置为假。	
CVEU	REAL	标定控制变量输出。使用 CVEUMax 和 CVEUMin 标定，其中 CVEUMax 对应于 100%，CVEUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CVEU = (CV * CVEUSpan / 100) + CVEUMin CVEU 量程计算：CVEUSpan = (CVEUMax – CVEUMin)	
CV	REAL	控制变量输出。此值始终以 0…100% 表示。处于自动模式、级联/比率模式或手动模式且 CVManLimiting 为真时，CV 限定在 CVHLimit 和 CVLLimit 之间；否则，将限定在 0 和 100% 之间。	
DeltaCV	REAL	当前 CV 与上一 CV 之间的差值（当前 CV - 上一 CV）。	
CVInitializing	BOOL	初始化模式指示器。当 CVInitReq 或功能块 FirstScan 为真，或者 CVFault 由真变为假(由不良转为良好) 时，该值设置为真。当功能块初始化完毕且 CVInitReq 不再为真后，CVInitializing 将设置为假。	
CVHAlarm	BOOL	CV 上限报警指示器。CV 的计算值 > 100 或 CVHLimit 时 为真。	
CVLAlarm	BOOL	CV 下限报警指示器。计算的 CV 值 < 0 或 CVLLimit 时为真。	
CVROCPosAlarm	BOOL	CV 变化率报警指示器。计算的 CV 变化率超出 CVROCPosLimit 时为真。	

CVROCNegAlarm	REAL	CV 变化率报警指示器。计算的 CV 变化率超出 CVROCNegLimit 时为真。	
SP	REAL	当前设置点的值。SP 值用于在自动、级联/比率或 PV 跟踪模式下控制 CV，以 PV 单位标定。	
SPPercent	REAL	以 PV 量程百分比表示的 SP 值。 SPPercent = ((SP – PVEUMin) * 100) / PVSpan 其中 PVSpan = PVEUMax – PVEUMin	
SPHAlarm	BOOL	SP 上限报警指示器。当 SP ≥ SPHLimit 时为真。	
SPLAlarm	BOOL	SP 下限报警指示器。当 SP ≤ SPLLimit 时为真。	
PVPercent	REAL	以量程百分比表示的 PV。 PVPercent = ((PV – PVEUMin) * 100) / PVSpan PV 量程计算：PVSpan = (PVEUMax – PVEUMin)	
E	REAL	过程误差。SP 与 PV 之间的差值，以 PV 单位标定。	
EPercent	REAL	以量程百分比形式表示的误差。	
InitPrimary	BOOL	初始化主回路命令。当未处于级联/比率模式或 CVInitializing 为真时，为真。此信号通常供主回路的 CVInitReq 输入使用。	
WindupHOut	BOOL	积分饱和上限指示器。当达到 SP 上限或 CV 上限/下限时为真。此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV 输出的积分饱和。	
WindupLOut	BOOL	积分饱和下限指示器。当达到 SP 或 CV 上限/下限时为真。此信号通常供 WindupLIn 输入使用，用以限制主回路上 CV 输出的积分饱和。	
Ratio	REAL	当前比率乘数，无单位。	

RatioHAlarm	BOOL	比率上限报警指示器。当 Ratio > RatioHLimit 时为真。	
RatioLAlarm	BOOL	比率下限报警指示器。当 Ratio < RatioLLimit 时为真。	
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。操作员控制模式下为假。	
CasRat	BOOL	级联/比率模式指示器。处于级联/比率模式时为真。	
Auto	BOOL	自动模式指示器。处于自动模式时为真。	
Manual	BOOL	手动模式指示器。处于手动模式时为真。	
Override	BOOL	超控模式指示器。处于超控模式时为真。	
Hand	BOOL	手控模式指示器。处于手控模式时为真。	
DeltaT	REAL	两次更新间隔的时间（秒）。	
StepSizeUsed	REAL	调谐中使用的实际 CV 步长。	
GainTuned	REAL	调谐完毕后计算出的内模增益值。	
TCTuned	REAL	调谐完毕后计算出的内模时间常数。	
DTTuned	REAL	调谐完毕后计算出的内模死区时间值。	
RespTCTunedS	REAL	调谐完毕后计算出的慢速响应速度下的控制变量时间常数。	
RespTCTunedM	REAL	调谐完毕后计算出的中速响应速度下的控制变量时间常数。	
RespTCTunedF	REAL	调谐完毕后计算出的快速响应速度下的控制变量时间常数。	
AtuneOn	BOOL	启动自动调谐后设置为真。	
AtuneDone	BOOL	自动调谐成功完成后设置为真。	
AtuneAborted	BOOL	当自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	

AtuneStatus	DINT	指示功能块的调谐状态。	
AtuneFault	BOOL	自动调谐产生以下任一故障。	AtuneStatus 的位 0
AtunePVOutOfLimit	BOOL	在自动调谐过程中 ,PV 或 PV 死区时间步提前预测值超过 PVTuneLimit。该值为真时，自动调谐过程将中止。	AtuneStatus 的位 1
AtuneModeInv	BOOL	IMC 模式在自动调谐开始时并非手动模式，或者在自动调谐过程中由手动模式切换为其他模式。该值为真时，自动调谐不会启动或者将会中止。	AtuneStatus 的位 2
AtuneCVWindupFault	BOOL	在自动调谐开始时或自动调谐期间，WindupHIn 或 WindupLIn 为真。该值为真时，自动调谐不会启动或者将会中止。	AtuneStatus 的位 3
AtuneStepSize0	BOOL	自动调谐开始时，StepSizeUsed = 0。该值为真时，自动调谐不会启动。	AtuneStatus 的位 4
AtuneCVLimitsFault	BOOL	在自动调谐开始时或自动调谐期间，CVLimitsInv 和 CVManLimiting 为真。该值为真时，自动调谐不会启动或者将会中止。	AtuneStatus 的位 5
AtuneCVInitFault	BOOL	在自动调谐开始时或自动调谐期间，CVInitializing 为真。该值为真时，自动调谐不会启动或者将会中止。	AtuneStatus 的位 6
AtuneEUSpanChanged	BOOL	在自动调谐期间，CVEUSpan 或 PVEUSpan 发生改变。该值为真时，自动调谐过程将中止。	AtuneStatus 的位 7
AtuneCVChanged	BOOL	CVOper（操作员控制模式）或 CVProg（程序控制模式）发生改变，或者 CV 在自动调谐期间达到上/下限或 ROC 限制。该值为真时，自动调谐过程将中止。	AtuneStatus 的位 8
AtuneTimeout	BOOL	自阶跃测试开始起经过的时间长于 AtuneTimeLimit。该值为真时，自动调谐过程将中止。	AtuneStatus 的位 9

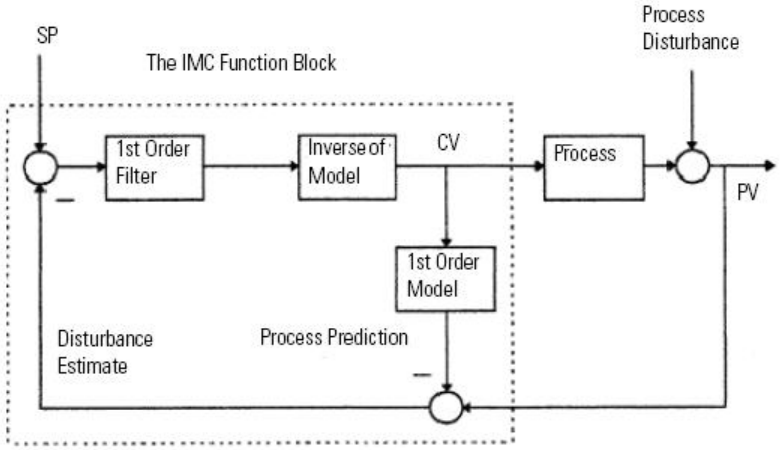
AtunePVNotSettled	BOOL	PV 变化过大而无法进行自动调谐。该值为真时，自动调谐过程将中止。等待 PV 达到更稳定状态再进行自动调谐。	AtuneStatus 的位 10
Status1	DINT	功能块的位映射状态。	
Status2	DINT	功能块的附加位映射状态。	
InstructFault	BOOL	功能块发生故障。指示 Status1 和 Status2 相应位的状态。 值为 0 时表示未发生故障。任何可能配置为无效值的参数都必须具有状态参数，来指示其无效状态。	Status1 的位 0
PVFaulted	BOOL	过程变量 PV 状况不良。	Status1 的位 1
CVFaulted	BOOL	控制变量 CV 故障	Status1 的位 2
HandFBFaulted	BOOL	HandFB 值状况不良	Status1 的位 3
PVSpanInv	BOOL	PV 量程无效，PVEUMax < PVEUMin。	Status1 的位 4
SPProgInv	BOOL	SPProg < SPLLimit 或 > SPHLimit。限制 SP 的值。	Status1 的位 5
SPOperInv	BOOL	SPOper < SPLLimit 或 > SPHLimit。限制 SP 的值。	Status1 的位 6
SPCascadeInv	BOOL	SPCascade < SPLLimit 或 > SPHLimit。限制 SP 的值。	Status1 的位 7
SPLimitsInv	BOOL	限值无效：SPLLimit < PVEUMin，SPHLimit > PVEUMax 或 SPHLimit < SPLLimit。如果 SPHLimit < SPLLimit，则使用 SPLLimit 对值进行限制。	Status1 的位 8
RatioLimitsInv	BOOL	比率上下限无效，下限 < 0 或上限 < 下限。	Status1 的位 9
RatioProgInv	BOOL	RatioProg < RatioLLimit 或 > RatioHLimit。将限制 Ratio 的值。	Status1 的位 10
RatioOperInv	BOOL	RatioOper < RatioLLimit 或 > RatioHLimit。将限制 Ratio 的值。	Status1 的位 11

CVProgInv	BOOL	CVProg < 0 或 > 100，或者当 CVManLimiting 为真时，< CVLLimit 或 > CVHLimit。将限制 CV 的值。	Status1 的位 12
CVOperInv	BOOL	CVOper < 0 或 > 100，或者当 CVManLimiting 为真时，< CVLLimit 或 > CVHLimit。将限制 CV 的值。	Status1 的位 13
CVOverrideValueInv	BOOL	CVOverrideValue < 0 或 > 100。将限制 CV 的值。	Status1 的位 14
CVTrackValueInv	BOOL	CVTrackValue < 0 或 > 100。将限制 CV 的值。	Status1 的位 15
CVEUSpanInv	BOOL	CVEU 的量程无效，CVEUMax 等于 CVEUMin。	Status1 的位 16
CVLimitsInv	BOOL	CVLLimit < 0、CVHLimit > 100 或 CVHLimit <= CVLLimit。如果 CVHLimit <= CVLLimit，则使用 CVLLimit 限制 CV 值。	Status1 的位 17
CVROCLimitInv	BOOL	CVROCLimit < 0，禁用 ROC 限制。	Status1 的位 18
HandFBInv	BOOL	HandFB < 0 或 > 100。将限制 CV 的值。	Status1 的位 19
SampleTimeTooSmall	BOOL	模型死区时间/DeltaT 必须小于或等于 200。	Status1 的位 20
FactorInv	BOOL	Factor 值 < 0。	Status1 的位 21
ModuleGainInv	BOOL	ModelGain 为 1.#QNAN 或 -1.#IND（非数字），或者 ±1.$（无穷大 ∞）	Status1 的位 22
ModelTCInv	BOOL	ModelTC < 0。	Status1 的位 23
ModelDTInv	BOOL	ModelDT < 0。	Status1 的位 24
RespTCInv	BOOL	RespTC < 0。	Status1 的位 25
TimingModelInv	BOOL	TimingMode 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	Status2 的位 27

RTSMissed	BOOL	仅用于实时采样模式。 ABS(DeltaT - RTSTime) > 1 毫秒时为真。	Status2 的位 28。
RTSTimeInv	BOOL	RTSTime 无效。	Status2 的位 29。
RTSTimeStampInv	BOOL	RTSTimeStamp 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	Status2 的位 30。
DeltaTInv	BOOL	DeltaT 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	Status2 的位 31。

说明

下图所示为 IMC 功能块的配置。



每次执行时，IMC 功能块都会将实际 PV 测量值与 PV 预测值进行比较。其结果称为扰动估计值，代表未测得过程扰动性与建模不精确性的综合效应。扰动估计值用作与控制变量设置点的偏差。在无扰动和完美建模的理想情况下，扰动估计值（反馈信号）变为零。

一阶模型	M = K/(T*s+1)*exp(-D*s)	
模型取反		Inv = (T*s+1)/K
一阶滤波器		F = 1/(e*s+1)

PV 预测值 = exp(-D*s)/(e*s+1) *（SP - 扰动估计值）

K...	模型增益
T...	模型时间常数
D...	模型死区时间
e...	响应时间常数
s...	拉普拉斯变量

功能块随后计算 CV 值（使用 CVHLimit、CVLLimit 和变化率限值）和 PV 预测值。

当控制死区时间较长的过程时，IMC 功能块比 PID 控制变量更具优势，因此可代替 PID 功能块。

对于积分过程类型（例如液位控制和位置控制），使用内部非积分模型来近似模拟积分过程。使用 Factor 参数将确定的积分过程模型转换为用于 CV 计算的非积分内部模型。这对于稳定的 IMC 执行非常必要。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

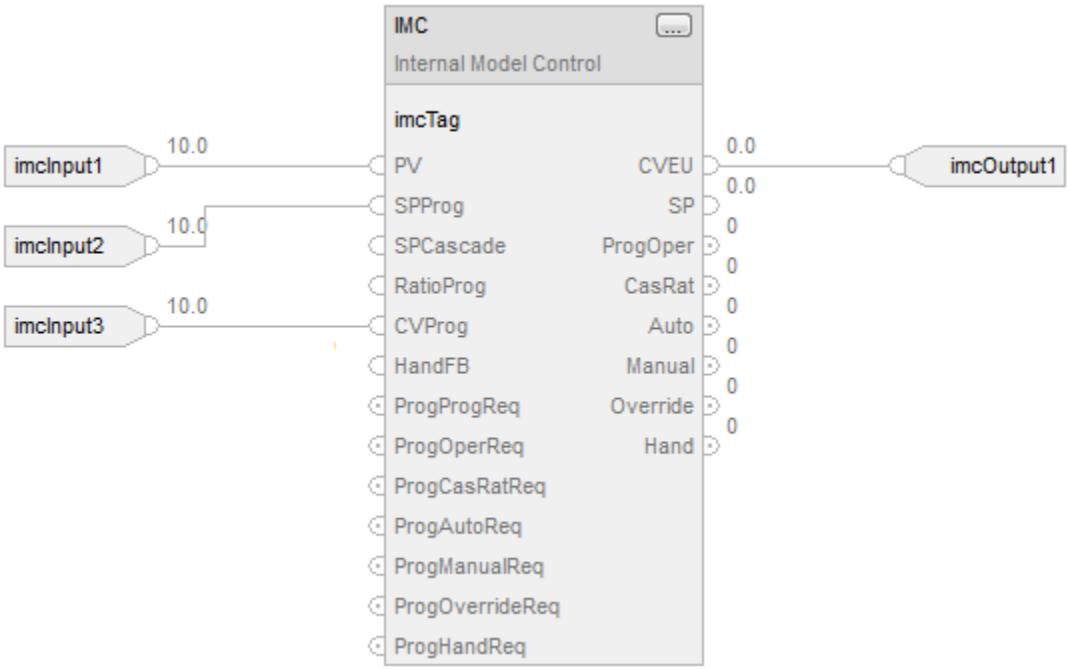
条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 .EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

功能块



结构化文本

```
imcTag.PV := imcInput1;

imcTag.SPProg := imcInput2;

imcTag.CVProg := imcInput3;

IMC(imcTag);

imcOutput1 := imcTag.CVEU;
```

另请参见

处理故障 参考页数 305

IMC 功能块调谐 参考页数 236

通用属性 参考页数 589

结构化文本语法 参考页数 559

功能块属性 参考页数 545

IMC 功能块配置

按照以下步骤创建基本的 IMC 配置。

1. 从默认配置开始，配置以下参数。

参数	说明
PVEUMax	PV 的最大标定值。
PVEUMin	PV 的最小标定值。
SPHLimit	SP 上限值，以 PV 单位标定。
SPLLimit	SP 下限值，以 PV 单位标定。
CVInitValue	控制变量输出的初始值。

2. 如果已有过程模型可用，可输入以下四个参数的值，直观地对 IMC 控制变量进行调谐。

参数	说明
模型增益	非零数（负数表示直接作用的控制变量，正数表示反向作用的控制变量）。
模型时间常数	始终为正数。
模型死区时间	始终为正数。
响应时间常数	始终为正数 - 用于调谐 IMC 控制变量的响应。数字越小，响应速度越快。

至此，已完成基本配置。未配置内置调谐器。控制变量可随时在自动或手动模式下置于在线状态。为进行调谐，可使用默认设置。请参见“IMC 功能块调谐”部分。

3. 如果不了解过程模型，则需要确定模型，并利用内置的调谐器（建模器）对控制变量进行调谐，以便于控制变量在自动模式下正常运行。

控制变量将一阶滞后与死区时间内部过程模型和一阶滤波器（共四个调谐参数）结合使用，计算 CV。计算 CV 时，确保过程变量 (PV) 在接近设置点的值时遵循一阶滞后轨迹。

响应速度取决于响应时间常数的值。响应时间常数的值越小，控制变量响应的速度越快。设置响应时间常数时，应确保 PV 可根据过程动态在合理的时间内达到设置点。响应时间常数的值越大，控制变量响应的速度越慢，但控制变量也会变得更加稳定。请参见“IMC 功能块调谐”部分。

在手动模式下，CV 设为等于操作员输入的或程序生成的 CVOper 或 CVProg 参数。

为实现从手动模式到自动模式的无扰动转换和控制变量的安全运算，实施了 CV 变化率限制器，以便 CV 从当前状态变化的速度不会超过指定的变化率限制参数。

4. 将 CVROCPosLimit 和 CVROCNegLimit 置位，对 CV 变化率进行限制。

   除非 CVManLimiting 已置位，否则控制变量处于手动模式时不会施加变化率限制。

另请参见

IMC 功能块调谐 参考页数 236

IMC 功能块模型初始化

在以下情况下会进行模型初始化：

- 首次扫描功能块期间
- ModelInit 请求参数置位时
- DeltaT 发生变化时

您可能需要手动调整内模参数或响应时间常数。为此，可更改相应参数并设置适当的 ModelInit 位。功能块的内部状态将进行初始化，相应的位将自动复位。

例如，如果修改 CV - PV 的 IMC 功能块模型增益，则将 ModelInit 参数设置为真，以初始化 CV - PV 内模参数，并使新模型增益生效。

IMC 功能块调谐

功能块配有内部调谐器（建模器）。调谐器的用途是识别过程模型参数并将这些参数用作内模参数（增益、时间常数和死区时间）。调谐器还会计算最佳响应时间常数。

配置以下参数，对调谐器进行设置。

ProcessType	积分（液位、位置控制）或非积分（流量、压力控制）
ProcessGainSign	置位表示过程增益为负值（输出增大会导致 PV 减小）；复位表示过程增益为正值（输出增大会导致 PV 增大）。
ResponseSpeed	慢速、中速或快速，取决于控制变量。
NoiseLevel	PV 的估计噪声等级（低、中、高），使调谐器能够区分哪些 PV 变化是随机噪声，哪些是由 CV 阶跃变化引起的。
StepSize	非零正数或负数，分别定义正向或负向的 CV 阶跃变化幅值。
PVTuneLimit	（仅适用于积分过程类型）采用 PV 工程单位，定义容许的因 CV 变化导致的 PV 变化量，超出此限制时会中止调谐测试。

将 AtuneStart 位置位，以此启动调谐器。要停止调谐，可将 AtuneAbort 位置位。调谐成功完成后，相应的 GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数会根据调谐结果进行更新，而且 AtuneStatus 代码会置位，指示调谐完成。

可将 AtuneUseModel 位置位，通过这种方式分别将这些参数复制到 ModelGain、ModelTC 和 ResponseTC。功能块会自动对内部变量进行初始化并继续正常工作。它会自动将 AtuneUseModel 位复位。

另请参见

IMC 功能块调谐过程 参考页数 237

IMC 功能块调谐错误 参考页数 237

IMC 功能块调谐错误

如果调谐过程中发生错误，调谐会中止，AtuneStatus 位会置位。要中止调谐，可将 AtuneAbort 位置位。

中止后，CV 会采用阶跃变化之前的值，GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数不会更新。AtuneStatus 参数将指示中止的原因。

IMC 功能块调谐过程

按照以下步骤配置调谐器。

1. 将 CV 设为手动模式。
2. 将 AtuneStart 参数置位。

   调谐器开始采集 PV 和 CV 数据进行噪声计算。
3. 采集 60 个样本 (60*DeltaT) 过后，调谐器会将 StepSize 与 CV 相加。

成功采集 CV 步更改生成的 PV 数据后，CV 会采用阶跃变化之前的值，AtuneStatus、GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数会更新。

4. 将 AtuneUseModel 参数置位，从而将经过调谐的参数复制到模型参数中。

   功能块随即会将 AtuneUseModel 参数复位。

   成功执行自动调谐后，Atune 参数会置 1。调谐成功完成。

模块多变量控制 (MMC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

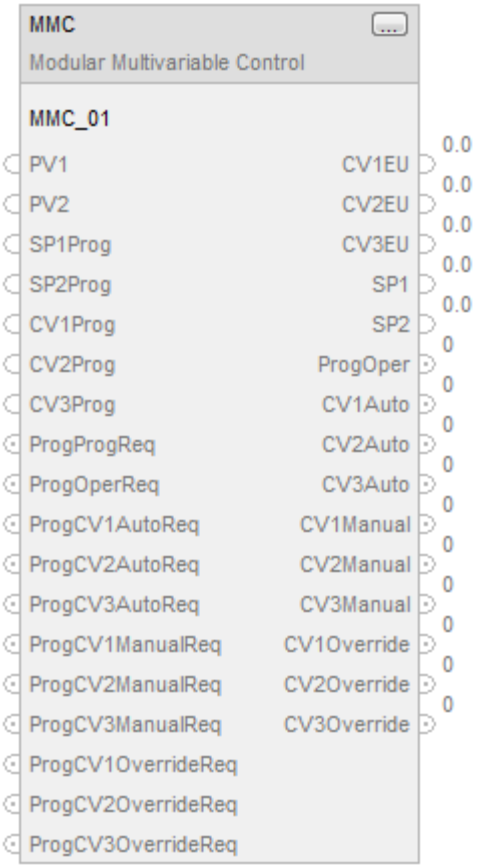
MMC 指令使用多达三个控制变量来控制两个过程变量，使其达到设置点。在自动模式下，MMC 基于 PV1 - SP1 偏差、PV2 - SP2 偏差、内部模型和调谐，计算控制变量（CV1、CV2 和 CV3）。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

MMC(MMC_tag);

操作数

功能块

操作数：	类型	格式	说明
MMC tag	MODULAR MULTIVARIABLE CONTROL	结构	MMC 结构

结构化文本

操作数：	类型	格式	说明
MMC tag	MODULAR MULTIVARIABLE CONTROL	结构	MMC 结构

结构

下表介绍 MMC 功能块的输入参数。

输入参数	数据类型	说明	值
EnableIn	BOOL	使能输入。如果为"假"，则功能块不执行，输出也不会更新。	默认值 = 真
PV1	REAL	经过标定的过程变量输入 1。该值通常从模拟量输入模块中读取。	有效值 = 任意浮点值 默认值 = 0.0
PV2	REAL	经过标定的过程变量输入 2。该值通常从模拟量输入模块中读取。	有效值 = 任意浮点值 默认值 = 0.0
PV1Fault	BOOL	PV1 不良状况指示器。如果 PV1 从模拟量输入中读取，则 PV1Fault 通常由模拟量输入故障状态控制。如果 PVFault 为"真"，则表示输入模块存在错误，会将 Status 中的相应位置位。 假 = 状况良好	默认值 = 假
PV2Fault	BOOL	PV2 不良状况指示器。如果 PV2 从模拟量输入中读取，则 PV2Fault 通常由模拟量输入故障状态控制。如果 PVFault 为"真"，则表示输入模块存在错误，会将 Status 中的相应位置位。 假 = 状况良好	默认值 = 假
PV1EUMax	REAL	PV1 的最大标定值。对应于过程变量 100% 量程的 PV1 和 SP1 的值。如果 PV1EUMax ≤ PV1EUMin，会将 Status 中的相应位置位。	有效值 = PV1EUMin < PV1EUMax <= 最大正浮点值 默认值 = 100.0
PV2EUMax	REAL	PV2 的最大标定值。对应于过程变量 100% 量程的 PV2 和 SP2 的值。如果 PV2EUMax ≤ PV2EUMin，会将 Status 中的相应位置位。	有效值 = PV2EUMin < PV2EUMax <= 最大正浮点值 默认值 = 100.0
PV1UEMin	REAL	PV1 的最小标定值。对应于过程变量 0% 量程的 PV1 和 SP1 的值。如果 PV1EUMax ≤ PV1EUMin，会将 Status 中的相应位置位。	有效值 = 最大负浮点值 <= PV1EUMin < PV1EUMax 默认值 = 0.0

输入参数	数据类型	说明	值
PV2UEMin	REAL	PV2 的最小标定值。对应于过程变量 0% 量程的 PV2 和 SP2 的值。如果 PV1EUMax ≤ PV1EUMin，会将 Status 中的相应位置位。	有效值 = 最大负浮点值 <= PV2EUMin < PV2EUMax 默认值 = 0.0
SP1Prog	REAL	SP1 程序值，以 PV 单位标定。SP1 在程序控制模式下设为此值。	有效值 = SP1LLimit 到 SP1HLimit 默认值 = 0.0
SP2Prog	REAL	SP2 程序值，以 PV 单位标定。SP2 在程序控制模式下设为此值。	有效值 = SP2LLimit 到 SP2HLimit 默认值 = 0.0
SP1Oper	REAL	SP1 操作员值，以 PV 单位标定。SP1 在操作员控制模式下设为此值。 如果 SP1Prog 或 SP1Oper 的值 < SP1LLimit 或 > SP1HLimit，则会将 Status 中的相应位置位，并限制 SP 的值。	有效值 = SP1LLimit 到 SP1HLimit 默认值 = 0.0
SP2Oper	REAL	SP2 操作员值，以 PV 单位标定。SP2 在操作员控制模式下设为此值。 如果 SP2Prog 或 SP2Oper 的值 < SP2LLimit 或 > SP2HLimit，则会将 Status 中的相应位置位，并限制 SP 的值。	有效值 = SP2LLimit 到 SP2HLimit 默认值 = 0.0
SP1HLimit	REAL	SP1 上限值，以 PV 单位标定。 • 如果 SP1LLimit < PV1EUMin 或 SP1HLimit > PV1EUMax，会将 Status 中的相应位置位。 • 如果 SP1HLimit < SP1LLimit，则会将 Status 中的相应位置位，并使用 S1PLLimit 的值限制 SP。	有效值 = SP1LLimit 到 PV1EUMax 默认值 = 100.0
SP2HLimit	REAL	SP2 上限值，以 PV 单位标定。 • 如果 SP2LLimit < PV2EUMin，或 SP2HLimit > PV2EUMax，则会将 Status 中的相应位置位。 • 如果 SP2HLimit < SP2LLimit，则会将 Status 中的相应位置位，并使用 SPLLimit 的值限制 SP。	有效值 = SP2LLimit 到 PV2EUMax 默认值 = 100.0

输入参数	数据类型	说明	值
SP1LLimit	REAL	SP1 下限值，以 PV 单位标定。 • 如果 SP1LLimit < PV1EUMin 或 SP1HLimit > PV1EUMax，会将 Status 中的相应位置位。 • 如果 SP1HLimit < SP1LLimit，则会将 Status 中的相应位置位，并使用 SPLLimit 的值限制 SP。	有效值 = PV1EUMin 到 SP1HLimit 默认值 = 0.0
SP2LLimit	REAL	SP2 下限值，以 PV 单位标定。 • 如果 SP2LLimit < PV2EUMin，或 SP2HLimit > PV2EUMax，则会将 Status 中的相应位置位。 • 如果 SP2HLimit < SP2LLimit，则会将 Status 中的相应位置位，并使用 SPLLimit 的值限制 SP。	有效值 = PV2EUMin 到 SP2HLimit 默认值 = 0.0
CV1Fault	BOOL	控制变量 1 不良状况指示器。如果 CV1EU 控制模拟量输出，则 CV1Fault 通常由该模拟量输出的故障状态引起。 如果 CV1Fault 为“真”，则表示输出模块存在错误，会将 Status 中的相应位置位。假 = 状况良好	默认值 = 假
CV2Fault	BOOL	控制变量 2 不良状况指示器。如果 CV2EU 控制模拟量输出，则 CV2Fault 通常由该模拟量输出的故障状态引起。 如果 CV2Fault 为“真”，则表示输出模块存在错误，会将 Status 中的相应位置位。假 = 状况良好	默认值 = 假
CV3Fault	BOOL	控制变量 3 不良状况指示器。如果 CV3EU 控制模拟量输出，则 CV3Fault 通常由该模拟量输出的故障状态引起。 如果 CV3Fault 为“真”，则表示输出模块存在错误，会将 Status 中的相应位置位。假 = 状况良好	默认值 = 假

输入参数	数据类型	说明	值
CV1InitReq	BOOL	CV1 初始化请求。当为“真”时，会将 CV1EU 设置为 CV1InitValue 的值。该信号通常由受控于 CV1EU 的模拟量输出模块上的 In Hold 状态控制，或者从次级回路的 InitPrimary 输出获得。 当 CV1Faulted 或 CV1EUSpanInv 为“真”时，会禁用指令初始化。	默认值 = 假
CV2InitReq	BOOL	CV2 初始化请求。当为“真”时，会将 CV2EU 设置为 CV2InitValue 的值。该信号通常由受控于 CV2EU 的模拟量输出模块上的 In Hold 状态控制，或者从次级回路的 InitPrimary 输出获得。 当 CV2Faulted 或 CV2EUSpanInv 为“真”时，会禁用指令初始化。	默认值 = 假
CV3InitReq	BOOL	CV3 初始化请求。当为“真”时，会将 CV3EU 设置为 CV3InitValue 的值。此信号通常由受 CV3EU 控制的模拟量输出模块的 In Hold 状态控制，或者来自次级回路的 InitPrimary 输出。 当 CV3Faulted 或 CV3EUSpanInv 为“真”时，会禁用指令初始化。	默认值 = 假
CV1InitValue	REAL	CV1EU 初始化值，以 CV1EU 单位标定。CV1Initializing 为“真”时，CV1EU 设为等于 CV1InitValue 并且 CV1 设为相应的百分比值。CV1InitValue 通常从受 CV1EU 控制的模拟量输出的反馈获取，或者从次级回路的设置点得出。 当 CV1Faulted 或 CV1EUSpanInv 为“真”时，会禁用指令初始化。	有效值 = 任意浮点值 默认值 = 0.0
CV2InitValue	REAL	CV2EU 初始化值，以 CV2EU 单位标定。当 CV2Initializing 为“真”时，将 CV2EU 设为等于 CV2InitValue 并且 CV2 设为相应的百分比值。CV2InitValue 通常从受 CV2EU 控制的模拟量输出的反馈获取，或者从次级回路的设置点得出。 当 CV2Faulted 或 CV2EUSpanInv 为“真”时，会禁用指令初始化。	有效值 = 任意浮点值 默认值 = 0.0

输入参数	数据类型	说明	值
CV3InitValue	REAL	CV3EU 初始化值，以 CV3EU 单位标定。当 CV3Initializing 为“真”时，将 CV3EU 设为等于 CV3InitValue 并且 CV3 设为相应的百分比值。CV3InitValue 通常从受 CV3EU 控制的模拟量输出的反馈获取，或者从次级回路的设置点得出。 当 CV3Faulted 或 CV3EUSpanInv 为“真”时，会禁用指令初始化。	有效值 = 任意浮点值 默认值 = 0.0
CV1Prog	REAL	程序-手动模式下的 CV1 值。在程序控制和手动模式下，CV1 设置为此值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV2Prog	REAL	程序-手动模式下的 CV2 值。在程序控制和手动模式下，CV2 设置为此值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV3Prog	REAL	程序-手动模式下的 CV3 值。在程序控制和手动模式下，CV3 设置为此值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV1Oper	REAL	操作员-手动模式下的 CV1 值。 • 在操作员控制和手动模式下，CV1 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CV1Oper 设置为 CV1 值。 • 如果当 CVManLimiting 为“真”时，CV1Prog 或 CV1Oper 的值 < 0 或 > 100，或 < CV1LLimit 或 > CV1HLimit，则将 Status 中的相应位置位，并限制 CV1 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0

输入参数	数据类型	说明	值
CV2Oper	REAL	操作员-手动模式下的 CV2 值。 • 在操作员控制和手动模式下时，CV2 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CV2Oper 设置为 CV2 值。 • 如果当 CVManLimiting 为“真”时，CV2Prog 或 CV2Oper 的值 < 0 或 > 100，或 < CV2LLimit 或 > CV2HLimit，则将 Status 中的相应位置位，并限制 CV2 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV3Oper	REAL	操作员-手动模式下的 CV3 值。 • 在操作员控制和手动模式下时，CV3 设置为此值。如果未处于操作员-手动模式，则在每个功能块执行结束时，将 CV3Oper 设置为 CV3 值。 • 如果当 CVManLimiting 为“真”时，CV3Prog 或 CV3Oper 的值 < 0 或 > 100，或 < CV3LLimit 或 > CV3HLimit，则将 Status 中的相应位置位，并限制 CV3 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV1OverrideValue	REAL	超控模式下的 CV1 值。 • 在超控模式下，CV1 设置为此值。此值应对应于回路的安全状态输出。 • 如果 CV1OverrideValue 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV1 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV2OverrideValue	REAL	超控模式下的 CV2 值。 • 在超控模式下，CV2 设置为此值。此值应对应于回路的安全状态输出。 • 如果 CV2OverrideValue 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV2 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0

输入参数	数据类型	说明	值
CV3OverrideValue	REAL	超控模式下的 CV3 值。在超控模式下，CV3 设置为此值。 此值应对应于回路的安全状态输出。 如果 CV3OverrideValue 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV3 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CVManLimiting	BOOL	限制手动模式下的 CV(n)，其中，(n) 可以是 1、2 或 3。如果处于手动模式并且 CVManLimiting 为“真”，则 CV(n) 由 CV(n)HLimit 和 CV(n)LLimit 的值限制。	默认值 = 假
CV1EUMax	REAL	CV1EU 的最大值。对应于 100% CV1 的 CV1EU 值。 如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CV2EUMax	REAL	CV2EU 的最大值。对应于 100% CV2 的 CV2EU 值。 如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CV3EUMax	REAL	CV3EU 的最大值。对应于 100% CV3 的 CV3EU 值。 如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 100.0
CV1EUMin	REAL	CV1EU 的最小值。对应于 0% CV1 的 CV1EU 值。 如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0
CV2EUMin	REAL	CV2EU 的最小值。对应于 0% CV2 的 CV2EU 值。 如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0
CV3EUMin	REAL	CV3EU 的最小值。对应于 0% CV3 的 CV3EU 值。 如果 CVEUMax = CVEUMin，则会将 Status 中的相应位置位。	有效值 = 任意浮点值 默认值 = 0.0

输入参数	数据类型	说明	值
CV1HLimit	REAL	CV1 上限值。该值用于设置 CV1HAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV1 进行限制。 • 如果 CV1LLimit < 0、CV1HLimit > 100 或 CV1HLimit < CV1LLimit，则将 Status 中的相应位置位。 • 如果 CV1HLimit < CV1LLimit，则使用 CV1LLimit 的值限制 CV1。	有效值 = CV1LLimit < CV1HLimit ≤ 100.0 默认值 = 100.0
CV2HLimit	REAL	CV2 上限值。该值用于设置 CV2HAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV2 进行限制。 • 如果 CV2LLimit < 0、CV2HLimit > 100 或 CV2HLimit < CV2LLimit，则将 Status 中的相应位置位。 • 如果 CV2HLimit < CV2LLimit，则使用 CV2LLimit 的值限制 CV2。	有效值 = CV2LLimit < CV2HLimit ≤ 100.0 默认值 = 100.0
CV3HLimit	REAL	CV3 上限值。该值用于设置 CV3HAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV3 进行限制。 • 如果 CV3LLimit < 0、CV3HLimit > 100 或 CV3HLimit < CV3LLimit，则将 Status 中的相应位置位。 • 如果 CV3HLimit < CV3LLimit，则使用 CV3LLimit 的值限制 CV3。	有效值 = CV3LLimit < CV3HLimit ≤ 100.0 默认值 = 100.0
CV1LLimit	REAL	CV1 下限值。该值用于设置 CV1LAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV1 进行限制。 • 如果 CV1LLimit < 0、CV1HLimit > 100 或 CV1HLimit < CV1LLimit，则将 Status 中的相应位置位。 • 如果 CV1HLimit < CV1LLimit，则使用 CV1LLimit 的值限制 CV1。	有效值 = 0.0 ≤ CV1LLimit < CV1HLimit 默认值 = 0.0

输入参数	数据类型	说明	值
CV2LLimit	REAL	CV2 下限值。该值用于设置 CV2LAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，该值也用于对 CV2 进行限制。 • 如果 CV2LLimit < 0、CV2HLimit > 100 或 CV2HLimit < CV2LLimit，则将 Status 中的相应位置位。 • 如果 CV2HLimit < CV2LLimit，则使用 CV2LLimit 的值限制 CV2。	有效值 = 0.0 ≤ CV2LLimit < CV1HLimit 默认值 = 0.0
CV3LLimit	REAL	CV3 下限值。该值用于设置 CV3LAlarm 输出。当处于自动模式，或者处于手动模式并且 CVManLimiting 设置为“真”时，该值也用于对 CV3 进行限制。 • 如果 CV3LLimit < 0、CV3HLimit > 100 或 CV3HLimit < CV3LLimit，则将 Status 中的相应位置位。 • 如果 CV3HLimit < CV3LLimit，则使用 CV3LLimit 的值限制 CV。	有效值 = 0.0 ≤ CV3LLimit < CV1HLimit 默认值 = 0.0
CV1ROCPosLimit	REAL	CV1 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV1 ROC 限制。 如果 CV1ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV1 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2ROCPosLimit	REAL	CV2 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV2 ROC 限制。 如果 CV2ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV2 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0

输入参数	数据类型	说明	值
CV3ROCPosLimit	REAL	CV3 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV3 ROC 限制。 如果 CV3ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV3 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1ROCNegLimit	REAL	CV1 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV1 ROC 限制。 如果 CV1ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV1 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2ROCNegLimit	REAL	CV2 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV2 ROC 限制。 如果 CV2ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV2 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3ROCNegLimit	REAL	CV3 变化率限值，以百分比/秒表示。仅当处于自动模式，或者处于手动模式并且 CVManLimiting 为“真”时，才使用变化率限制。使用零值将会禁用 CV3 ROC 限制。 如果 CV3ROCLimit 的值 < 0，则会将 Status 中的相应位置位，并禁用 CV3 ROC 限制。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0

输入参数	数据类型	说明	值
CV1HandFB	REAL	CV1 HandFeedback 值。当处于手动模式下且 CV1HandFBFault 为假（状况良好）时，将 CV1 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。 如果 CV1HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV1 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV2HandFB	REAL	CV2 HandFeedback 值。当处于手动模式下且 CV2HandFBFault 为假（状况良好）时，将 CV2 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。 如果 CV2HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV2 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV3HandFB	REAL	CV3 HandFeedback 值。当处于手动模式下且 CV3HandFBFault 为假（状况良好）时，将 CV3 设置为该值。该值通常来自现场安装的手控/自动站的输出，用于在手控模式下进行无扰动转换。 如果 CV3HandFB 的值 < 0 或 >100，则将 Status 中的相应位置位，并限制 CV3 的值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV1HandFBFault	BOOL	CV1HandFB 值不良状况指示器。如果从模拟量输入中读取 CV1HandFB 值，则 CV1HandFBFault 通常由模拟量输入通道的状态控制。 CV1HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。 假 = 状况良好	默认值 = 假

输入参数	数据类型	说明	值
CV2HandFBFault	BOOL	CV2HandFB 值不良状况指示器。如果从模拟量输入中读取 CV2HandFB 值，则 CV2HandFBFault 通常由模拟量输入通道的状态控制。 CV2HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。 假 = 状况良好	默认值 = 假
CV3HANDFBFault	BOOL	CV3HandFB 值不良状况指示器。如果从模拟量输入中读取 CV3HandFB 值，则 CV3HandFBFault 通常由模拟量输入通道的状态控制。 CV3HandFBFault 为真时，指示该输入模块存在错误，并且会将 Status 中的相应位置位。 假 = 状况良好	默认值 = 假
CV1Target	REAL	控制变量输出 1 的目标值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV2Target	REAL	控制变量输出 2 的目标值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV3Target	REAL	控制变量输出 3 的目标值。	有效值 = 0.0 到 100.0 默认值 = 0.0
CV1WindupHIn	BOOL	CV1 积分饱和上限请求。该参数为真时，不允许 CV1 值增加。该信号通常来自次级回路的 CV1WindupHOut 输出。	默认值 = 假
CV2WindupHIn	BOOL	CV2 积分饱和上限请求。该参数为真时，不允许 CV2 值增加。该信号通常来自次级回路的 CV2WindupHOut 输出。	默认值 = 假
CV3WindupHIn	BOOL	CV3 积分饱和上限请求。该参数为真时，不允许 CV3 值增加。该信号通常来自次级回路的 CV3WindupHOut 输出。	默认值 = 假

输入参数	数据类型	说明	值
CV1WindupLIn	BOOL	CV1 积分饱和下限请求。该参数为真时，不允许 CV1 值减小。该信号通常来自次级回路的 CV1WindupLOut 输出。	默认值 = 假
CV2WindupLIn	BOOL	CV2 积分饱和下限请求。该参数为真时，不允许 CV2 值减小。该信号通常来自次级回路的 CV2WindupLOut 输出。	默认值 = 假
CV3WindupLIn	BOOL	CV3 积分饱和下限请求。该参数为真时，不允许 CV3 值减小。该信号通常来自次级回路的 CV3WindupLOut 输出。	默认值 = 假
GainEUSpan	BOOL	ModelGain 单位（EU 或量程百分比）。	默认值 = 假 假 = % 形式的增益
CV1PV1ProcessGainSign	BOOL	仅用于自动调谐。过程增益的符号 (Delta PV1/Delta CV1)。 • 置位指示负过程增益（输出增大会导致 PV1 减小）。 • 复位指示正过程增益（输出增大会导致 PV1 增大）。	默认值 = 假
CV2PV1ProcessGainSign	BOOL	仅用于自动调谐。过程增益的符号 (Delta PV1/Delta CV2)。 • 置位指示负过程增益（输出增大会导致 PV1 减小）。 • 复位指示正过程增益（输出增大会导致 PV1 增大）。	默认值 = 假
CV3PV1ProcessGainSign	BOOL	仅用于自动调谐。过程增益的符号 (Delta PV1/Delta CV3)。 • 置位指示负过程增益（输出增大会导致 PV1 减小）。 • 复位指示正过程增益（输出增大会导致 PV1 增大）。	默认值 = 假

输入参数	数据类型	说明	值
CV1PV2ProcessGainSign	BOOL	仅用于自动调谐。过程增益的符号 (Delta PV2/Delta CV1)。 • 置位指示负过程增益（输出增大会导致 PV2 减小）。 • 复位指示正过程增益（输出增大会导致 PV2 增大）。	默认值 = 假
CV1PV2ProcessGainSign	BOOL	仅用于自动调谐。过程增益的符号 (Delta PV2/Delta CV2)。 • 置位指示负过程增益（输出增大会导致 PV2 减小）。 • 复位指示正过程增益（输出增大会导致 PV2 增大）。	默认值 = 假
CV1PV2ProcessGainSign	BOOL	仅用于自动调谐。过程增益的符号 (Delta PV2/Delta CV3)。 • 置位指示负过程增益（输出增大会导致 PV2 减小）。 • 复位指示正过程增益（输出增大会导致 PV2 增大）。	默认值 = 假
ProcessType	DINT	PV1 和 PV2 的过程类型选择（1= 积分，0= 非积分）	默认值 =0
CV1PV1ModelGain	REAL	CV1 - PV1 的内模增益参数。根据过程方向输入正增益或负增益。 如果 CV1PV1ModelGain = INF 或 NAN，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0
CV2PV1ModelGain	REAL	CV2 - PV1 的内模增益参数。根据过程方向输入正增益或负增益。 如果 CV2PV1ModelGain = INF 或 NAN，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0
CV3PV1ModelGain	REAL	CV3 - PV1 的内模增益参数。根据过程方向输入正增益或负增益。 如果 CV3PV1ModelGain = INF 或 NAN，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0
CV1PV2ModelGain	REAL	CV1 - PV2 的内模增益参数。根据过程方向输入正增益或负增益。 如果 CV1PV2ModelGain = INF 或 NAN，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 =0.0

输入参数	数据类型	说明	值
CV2PV2ModelGain	REAL	CV2 - PV2 的内模增益参数。根据过程方向输入正增益或负增益。 如果 CV2PV2ModelGain = INF 或 NAN，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 = 0.0
CV3PV2ModelGain	REAL	CV3 - PV2 的内模增益参数。根据过程方向输入正增益或负增益。 如果 CV3PV2ModelGain = INF 或 NAN，则将 Status 中的相应位置位。	有效值 = 最大负浮点值 –> 最大正浮点值 默认值 = 0.0
CV1PV1ModelTC	REAL	CV1 - PV1 的内模时间常数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2PV1ModelTC	REAL	CV2 - PV1 的内模时间常数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3PV1ModelTC	REAL	CV3 - PV1 的内模时间常数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1PV2ModelTC	REAL	CV1 - PV2 的内模时间常数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2PV2ModelTC	REAL	CV2 - PV2 的内模时间常数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3PV2ModelTC	REAL	CV3 - PV2 的内模时间常数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1PV1ModelDT	REAL	CV1 - PV1 的内模死区时间。以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2PV1ModelDT	REAL	CV2 - PV1 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3PV1ModelDT	REAL	CV3 - PV1 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0

输入参数	数据类型	说明	值
CV1PV2ModelDT	REAL	CV1 - PV2 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2PV2ModelDT	REAL	CV2 - PV2 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3PV2ModelDT	REAL	CV3 - PV2 的内模死区时间，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1PV1RespTC	REAL	确定 CV1 - PV1 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2PV1RespTC	REAL	确定 CV2 - PV1 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3PV1RespTC	REAL	确定 CV3 - PV1 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV1PV2RespTC	REAL	确定 CV1 - PV2 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV2PV2RespTC	REAL	确定 CV2 - PV2 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
CV3PV2RespTC	REAL	确定 CV3 - PV2 的控制变量操作速度的调谐参数，以秒为单位。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
PV1Act1stCV	DINT	用于补偿 PV1-SP1 偏差的第一个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 1
PV1Act2ndCV	DINT	用于补偿 PV1-SP1 偏差的第二个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 2
PV1Act3rdCV	DINT	用于补偿 PV1-SP1 偏差的第三个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 3
PV2Act1stCV	DINT	用于补偿 PV2-SP2 偏差的第一个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 1

输入参数	数据类型	说明	值
PV2Act2ndCV	DINT	用于补偿 PV2-SP2 偏差的第二个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 2
PV2Act3rdCV	DINT	用于补偿 PV2-SP2 偏差的第三个 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 3
TargetCV	DINT	要驱动到其目标值的 CV。 1=CV1、 2=CV2、 3=CV3	有效值 = 1-3 默认值 = 3
TargetRespTC	REAL	确定控制变量接近目标值的速度。	有效值 = 0.0 到最大正浮点值 默认值 = 0.0
PVTracking	BOOL	SP 跟踪 PV 请求。 设置为真可以使 SP 跟踪 PV。自动模式下忽略该参数。仅当全部的三个输出均处于手动模式时，SP 才跟踪 PV。只要有输出返回到自动模式，PVTracking 立即停止。	默认值 = 假
ManualAfterInit	BOOL	初始化后进入手动模式的请求。 • 该参数为真时 ,若 CV(n)Initializing 设置为真，除非当前模式为超控或手控模式，否则相应的 CV(n) 将切换为手动模式，其中 (n) 可以是 1、2 或 3。 • 当 ManualAfterInit 为假时，CV(n) 模式保持不变。	默认值 = 假
ProgProgReq	BOOL	程序发出的程序控制请求。 • 由用户程序设置为真请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。如果该值保持为真且 ProgOperReq 为假，则可将功能块锁定在程序控制模式。 • 当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假

输入参数	数据类型	说明	值
ProgOperReq	BOOL	程序发出的操作员控制请求。 • 由用户程序设置为真可请求操作员控制模式。如果该值保持为真，则可将功能块锁定在操作员控制模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV1AutoReq	BOOL	CV1 的程序-自动模式请求。 • 由用户程序设置为真可请求自动模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV2AutoReq	BOOL	CV2 的程序-自动模式请求。 • 由用户程序设置为真可请求自动模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV3AutoReq	BOOL	CV3 的程序-自动模式请求。 • 由用户程序设置为真可请求自动模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV1ManualReq	BOOL	CV1 的程序-手动模式请求。 • 由用户程序设置为真可请求手动模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV2ManualReq	BOOL	CV2 的程序-手动模式请求。 • 由用户程序设置为真可请求手动模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假

输入参数	数据类型	说明	值
ProgCV3ManualReq	BOOL	CV3 的程序-手动模式请求。 • 由用户程序设置为真可请求手动模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV1OverrideReq	BOOL	CV1 的程序-超控模式请求。 • 由用户程序设置为真可请求超控模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV2OverrideReq	BOOL	CV2 的程序-超控模式请求。 • 由用户程序设置为真可请求超控模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV3OverrideReq	BOOL	CV3 的程序-超控模式请求。 • 由用户程序设置为真可请求超控模式。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV1HandReq	BOOL	CV1 的程序-手控模式请求。 • 由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假
ProgCV2HandReq	BOOL	CV2 的程序-手控模式请求。 • 由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。 • 当 ProgValueReset 为真时，功能块将输入复位（假）。	默认值 = 假

输入参数	数据类型	说明	值
ProgCV3HandReq	BOOL	CV3 的程序-手控模式请求。 • 由用户程序设置为真可请求手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。 • 当 ProgValueReset 为真时 ,功能块将输入复位（假）。	默认值 = 假
OperProgReq	BOOL	操作员发出的程序控制请求。 • 由操作员界面设置为真以请求程序控制模式。功能块将此参数复位（假）。	默认值 = 假
OperOperReq	BOOL	操作员发出的操作员控制请求。 • 由操作员界面设置为真可请求操作员控制模式。功能块将此参数复位（假）。	默认值 = 假
OperCV1AutoReq	BOOL	CV1 的操作员-自动模式请求。 由操作员界面设置为真可请求自动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV2AutoReq	BOOL	CV2 的操作员-自动模式请求。 由操作员界面设置为真可请求自动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV3AutoReq	BOOL	CV3 的操作员-自动模式请求。 由操作员界面设置为真可请求自动模式。功能块将此参数复位（假）。	默认值 = 假
OperCV1ManualReq	BOOL	CV1 的操作员-手动模式请求。 由操作员界面设置为真可请求手动模式。功能块将此参数设置为“假”。	默认值 = 假
OperCV2ManualReq	BOOL	CV2 的操作员-手动模式请求。 由操作员界面设置为真可请求手动模式。功能块将此参数设置为“假”。	默认值 = 假
OperCV3ManualReq	BOOL	CV3 的操作员-手动模式请求。 由操作员界面设置为真可请求手动模式。功能块将此参数设置为“假”。	默认值 = 假

输入参数	数据类型	说明	值
ProgValueReset	BOOL	将程序控制值复位。 该值为真时，Prog_xxx_Req 输入复位（假）。当此参数为"真"且处于程序控制模式时，设置 SP(x)Prog = SP(x) 且 CV(y)Prog = CV(y)，其中 x = 1，2；y = 1，2，3	默认值 = 假
TimingMode	DINT	选择时基执行模式。 值/说明 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的更多信息，请参见"功能块属性"部分。	有效值 = 0 到 2 默认值 = 0
OverSampleDT	REAL	过采样模式的执行时间。	有效值 = 0 到 TON_Timer 经过的最长时间（4194.303 秒） 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期。	有效值 = 0 到 32,767 1 次计数 = 1 ms
RTSTimeStamp	DINT	实时采样模式的模块时戳值。	有效值 = 0 到 32,767（从 32,767 跳回到 0） 1 次计数 = 1 ms
PV1TuneLimit	REAL	以 PV1 单位标定的 PV1 调谐限值。当自动调谐正在运行并且预测 PV1 值超过此限值时，调谐过程将中止。	有效值 = 任意浮点值 默认值 = 0
PV2TuneLimit	REAL	以 PV2 单位标定的 PV2 调谐限值。当自动调谐正在运行并且预测 PV2 值超过此限值时，调谐过程将中止。	有效值 = 任意浮点值 默认值 = 0
PV1AtuneTimeLimit	REAL	CV1 发生阶跃变化后 PV1 完成自动调谐所需的最长时间，以分钟为单位。PV1 自动调谐时间超出此时间时，调谐将会中止。	有效值范围：任意 > 0 的浮点值。 默认值 = 60 分钟
PV2AtuneTimeLimit	REAL	CV2 发生阶跃变化后 PV2 完成自动调谐所需的最长时间，以分钟为单位。PV2 自动调谐时间超出此时间时，调谐将会中止。	有效值范围：任意 > 0 的浮点值。 默认值 = 60 分钟

输入参数	数据类型	说明	值
PV1NoiseLevel	DINT	PV1 的噪声级别估计值，在调谐期间可对该值进行补偿。 可选项包括：0 = 低、1 = 中、2 = 高	范围：0 到 2 默认值 = 1
PV2NoiseLevel	DINT	PV2 的噪声级别估计值，在调谐期间可对该值进行补偿。 可选项包括：0 = 低、1 = 中、2 = 高	范围：0 到 2 默认值 = 1
CV1StepSize	REAL	调谐阶跃测试所用的 CV1 步长（百分比）。步长将直接加到 CV1（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%
CV2StepSize	REAL	调谐阶跃测试所用的 CV2 步长（百分比）。步长将直接加到 CV2（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%
CV3StepSize	REAL	调谐阶跃测试所用的 CV3 步长（百分比）。步长将直接加到 CV3（不超出上限/下限范围）。	范围：-100% … 100% 默认值 = 10%
CV1PV1ResponseSpeed	DINT	CV1 - PV1 所需闭环响应速度。 • 慢速响应：ResponseSpeed=0 • 中速响应：ResponseSpeed=1 • 快速响应：ResponseSpeed=2 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0…2 默认值 = 1
CV2PV1ResponseSpeed	DINT	CV2 - PV1 所需闭环响应速度。 • 慢速响应：ResponseSpeed=0 • 中速响应：ResponseSpeed=1 • 快速响应：ResponseSpeed=2 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0…2 默认值 = 1
CV3PV1ResponseSpeed	DINT	CV3 - PV1 所需闭环响应速度。 • 慢速响应：ResponseSpeed=0 • 中速响应：ResponseSpeed=1 • 快速响应：ResponseSpeed=2 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0 到 2 默认值 = 1

输入参数	数据类型	说明	值
CV1PV2ResponseSpeed	DINT	CV1 - PV2 所需闭环响应速度。 • 慢速响应：ResponseSpeed=0 • 中速响应：ResponseSpeed=1 • 快速响应：ResponseSpeed=2 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0 到 2 默认值 = 1
CV2PV2ResponseSpeed	DINT	CV2 - PV2 所需闭环响应速度。 • 慢速响应：ResponseSpeed=0 • 中速响应：ResponseSpeed=1 • 快速响应：ResponseSpeed=2 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0 到 2 默认值 = 1
CV3PV2ResponseSpeed	DINT	CV3 - PV2 所需闭环响应速度。 • 慢速响应：ResponseSpeed=0 • 中速响应：ResponseSpeed=1 • 快速响应：ResponseSpeed=2 如果 ResponseSpeed 小于 0，则使用慢速响应。如果 ResponseSpeed 大于 2，则使用快速响应。	范围：0 到 2 默认值 = 1
CV1PV1ModelInit	BOOL	CV1 - PV1 内模初始化开关。请参见“功能块属性”部分。	默认值 = 假
CV2PV1ModelInit	BOOL	CV2 - PV1 内模初始化开关。请参见“功能块属性”部分。	默认值 = 假
CV3PV1ModelInit	BOOL	CV3 - PV1 内模初始化开关。请参见“功能块属性”部分。	默认值 = 假
CV1PV2ModelInit	BOOL	CV1 - PV2 内模初始化开关。请参见“功能块属性”部分。	默认值 = 假
CV2PV2ModelInit	BOOL	CV2 - PV2 内模初始化开关。请参见“功能块属性”部分。	默认值 = 假
CV3PV2ModelInit	BOOL	CV3 - PV2 内模初始化开关。请参见“功能块属性”部分。	默认值 = 假
PV1Factor	REAL	PV1 的非积分模型近似因子。仅用于积分过程类型。	默认值 = 100

输入参数	数据类型	说明	值
PV2Factor		PV2 的非积分模型近似因子。仅用于积分过程类型。	默认值 = 100
AtuneCV1Start	BOOL	CV1 的自动调谐启动请求。该参数为真时，启动 PV1 和 PV2 的 CV1 输出的自动调谐。如果 CV1 未处于手动模式，则忽略该值。功能块将该输入复位（假）。	默认值 = 假
AtuneCV2Start	BOOL	CV2 的自动调谐启动请求。该参数为真时，启动 PV1 和 PV2 的 CV2 输出的自动调谐。如果 CV2 未处于手动模式，则忽略该值。功能块将该输入复位（假）。	默认值 = 假
AtuneCV3Start	BOOL	CV3 的自动调谐启动请求。该参数为真时，启动 PV1 和 PV2 的 CV3 输出的自动调谐。如果 CV3 未处于手动模式，则忽略该值。功能块将该输入复位（假）。	默认值 = 假
AtuneCV1PV1UseModel	BOOL	CV1 - PV1 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV2PV1UseModel	BOOL	CV2 - PV1 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV3PV1UseModel	BOOL	CV3 - PV1 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV1PV2UseModel	BOOL	CV1 - PV2 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV2PV2UseModel	BOOL	CV2 - PV2 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假

输入参数	数据类型	说明	值
AtuneCV3PV2UseModel	BOOL	CV3 - PV2 的自动调谐模型使用请求。该值为真时，以计算出的自动调谐模型参数替换当前模型参数。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV1Abort	BOOL	CV1 的自动调谐中止请求。该参数为真时 ,会中止 PV1 和 PV2 的 CV1 输出的自动调谐。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV2Abort	BOOL	CV2 的自动调谐中止请求。该参数为真时 ,会中止 PV1 和 PV2 的 CV2 输出的自动调谐。功能块将该输入参数复位（假）。	默认值 = 假
AtuneCV3Abort	BOOL	CV3 的自动调谐中止请求。该参数为真时 ,会中止 PV1 和 PV2 的 CV3 输出的自动调谐。功能块将该输入参数复位（假）。	默认值 = 假

下表介绍 MMC 功能块的输出参数。

输出参数	数据类型	说明	值
EnableOut	BOOL	指示指令是否处于启用状态。如果 CV1EU、CV2EU 或 CV3EU 中的任何一个参数溢出，则设置为假。	
CV1EU	REAL	CV1 的标定控制变量输出。使用 CV1EUMax 和 CV1EUMin 标定，其中 CV1EUMax 对应于 100%，CV1EUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CV1EU = (CV1 * CV1EUSpan / 100) + CV1EUMin CV1EU 量程计算：CV1EUSpan = (CV1EUMax – CV1EUMin)	

输出参数	数据类型	说明	值
CV2EU	REAL	CV2 的标定控制变量输出。使用 CV2EUMax 和 CV2EUMin 标定，其中 CV2EUMax 对应于 100%，CV2EUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CV2EU = (CV2 * CV2EUSpan / 100) + CV2EUMin CV2EU 量程计算：CV2EUSpan = (CV2EUMax – CV2EUMin)	
CV3EU	REAL	CV3 的标定控制变量输出。使用 CV3EUMax 和 CV3EUMin 标定，其中 CV3EUMax 对应于 100%，CV3EUMin 对应于 0%。此输出通常用于控制模拟量输出模块或次级回路。 CV3EU = (CV3 * CV3EUSpan / 100) + CV3EUMin CV3EU 量程计算：CV3EUSpan = (CV3EUMax – CV3EUMin)	
CV1	REAL	CV1 的控制变量输出。此值始终以 0…100% 表示。当处于自动模式下或者处于手动模式下且 CVManLimiting 为真时，CV1 介于 CV1HLimit 和 CV1LLimit 之间；否则，介于 0 和 100% 之间。	
CV2	REAL	CV2 的控制变量输出。此值始终以 0…100% 表示。当处于自动模式下或者处于手动模式下且 CVManLimiting 为真时，CV2 介于 CV2HLimit 和 CV2LLimit 之间；否则，介于 0 和 100% 之间。	
CV3	REAL	CV3 的控制变量输出。此值始终以 0…100% 表示。当处于自动模式下或者处于手动模式下且 CVManLimiting 为真时，CV3 介于 CV3HLimit 和 CV3LLimit 之间；否则，介于 0 和 100% 之间。	
CV1Initializing	BOOL	CV1 的初始化模式指示器。当 CV1InitReq、功能块 blockFirstScan 或 OLCFirstRun 为真，或者 CV1Fault 由真变为假（由不良转为良好）时，该值设置为真。当功能块初始化完毕且 CV1InitReq 不再为真后，CV1Initializing 将设置为假。	

输出参数	数据类型	说明	值
CV2nitializing	BOOL	CV2 的初始化模式指示器。当 CV2InitReq、功能块 blockFirstScan 或 OLCFirstRun 为真，或者 CV2Fault 由真变为假（由不良转为良好）时，该值设置为真。当功能块初始化完毕且 CV2InitReq 不再为真后，CV2Initializing 将设置为假。	
CV3nitializing	BOOL	CV3 的初始化模式指示器。当 CV3InitReq、功能块 blockFirstScan 或 OLCFirstRun 为真，或者 CV3Fault 由真变为假（由不良转为良好）时，该值设置为真。当功能块初始化完毕且 CV3InitReq 不再为真后，CV3Initializing 将设置为假。	
CV1HAlarm	BOOL	CV1 上限报警指示器。计算的 CV1 值 > 100 或 CV1HLimit 时为真。	
CV2HAlarm	BOOL	CV2 上限报警指示器。计算的 CV2 值 > 100 或 CV2HLimit 时为真。	
CV3HAlarm	BOOL	CV3 上限报警指示器。计算的 CV3 值 > 100 或 CV3HLimit 时为真。	
CV1LAlarm	BOOL	CV1 下限报警指示器。计算的 CV1 值 < 0 或 CV1LLimit 时为真。	
CV2LAlarm	BOOL	CV2 下限报警指示器。计算的 CV2 值 < 0 或 CV2LLimit 时为真。	
CV3LAlarm	BOOL	CV3 下限报警指示器。计算的 CV3 值 < 0 或 CV3LLimit 时为真。	
CV1ROCPosAlarm	BOOL	CV1 变化率报警指示器。计算的 CV1 变化率超出 CV1ROCPosLimit 时为真。	
CV2ROCPosAlarm	BOOL	CV2 变化率报警指示器。计算的 CV2 变化率超出 CV2ROCPosLimit 时为真。	
CV3ROCPosAlarm	BOOL	CV3 变化率报警指示器。计算的 CV3 变化率超出 CV3ROCPosLimit 时为真。	
CV1ROCNegAlarm	BOOL	CV1 变化率报警指示器。计算的 CV1 变化率超出 CV1ROCNegLimit 时为真。	
CV2ROCNegAlarm	BOOL	CV2 变化率报警指示器。计算的 CV2 变化率超出 CV2ROCNegLimit 时为真。	

输出参数	数据类型	说明	值
CV3ROCNegAlarm	BOOL	CV3 变化率报警指示器。计算的 CV3 变化率超出 CV3ROCNegLimit 时为真。	
SP1	REAL	当前设置点 1 的值。SP1 值用于在自动模式或 PV1 跟踪模式下控制 CV，以 PV1 单位标定。	
SP2	REAL	当前设置点 2 的值。SP2 值用于在自动模式或 PV2 跟踪模式下控制 CV，以 PV2 单位标定。	
SP1Percent	REAL	以 PV1 量程百分比表示的 SP1 值。 SP1Percent = ((SP1 – PV1EUMin) * 100) / PV1Span 其中 PV1Span = PV1EUMax – PV1EUMin	
SP2Percent		以 PV2 量程百分比表示的 SP2 值。 SP2Percent = ((SP2 – PV2EUMin) * 100) / PV2Span 其中 PV2Span = PV2EUMax – PV1EUMin	
SP1HAlarm	BOOL	SP1 上限报警指示器。当 SP1 >= SP1HLimit 时为“真”。	
SP2HAlarm	BOOL	SP2 上限报警指示器。当 SP2 >= SP2HLimit 时为“真”。	
SP1LAlarm	BOOL	SP1 下限报警指示器。当 SP1 <= SP1LLimit 时为真。	
SP2LAlarm	BOOL	SP2 下限报警指示器。当 SP2 <= SP2LLimit 时为真。	
PV1Percent	REAL	以量程百分比表示的 PV1。 PV1Percent = ((PV1 – PV1EUMin) * 100) / PV1Span PV1 量程计算：PV1Span = (PV1EUMax - PV1EUMin)	
PV2Percent	REAL	以量程百分比表示的 PV2。 PV2Percent = ((PV2 – PV2EUMin) * 100) / PV2Span PV2 量程计算：PV2Span = (PV2EUMax – PV2EUMin)	
E1	REAL	过程 1 误差。SP1 与 PV1 之间的差值，以 PV1 单位标定。	
E2	REAL	过程 2 误差。SP2 与 PV2 之间的差值，以 PV2 单位标定。	

输出参数	数据类型	说明	值
E1Percent	REAL	以过程 1 量程百分比表示的误差。	
E2Percent	REAL	以过程 2 量程百分比表示的误差。	
CV1WindupHOut	BOOL	CV1 积分饱和上限指示器。 以下情况下，CV1WindupHOut 为真： • SP1HAlarm 或 SP2HAlarm 为真，或者 • ModelGain 为正且 CV1HAlarm 为真，或者 • ModelGain 为负且 CV1LAlarm 为真。 此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV1 输出的积分饱和。	
CV2WindupHOut	BOOL	CV2 积分饱和上限指示器。 以下情况下，CV2WindupHOut 为真： • SP1HAlarm 或 SP2HAlarm 为真，或者 • ModelGain 为正且 CV2HAlarm 为真，或者 • ModelGain 为负且 CV2LAlarm 为真。 此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV2 输出的积分饱和。	
CV3WindupHOut	BOOL	CV3 积分饱和上限指示器。 以下情况下，CV3WindupHOut 为真： • SP1HAlarm 或 SP2HAlarm 为真，或者 • ModelGain 为正且 CV3HAlarm 为真，或者 • ModelGain 为负且 CV3LAlarm 为真。 此信号通常供 WindupHIn 输入使用，用以限制主回路上 CV3 输出的积分饱和。	

输出参数	数据类型	说明	值
CV1WindupLOut	BOOL	CV1 积分饱和下限指示器。 以下情况下，CV1WindupLOut 为真： • SP1LAlarm 或 SP2LAlarm 为真，或者 • ModelGain 为正且 CV1LAlarm 为真，或者 • ModelGain 为负且 CV1HAlarm 为真。 此信号通常供 WindupLIn 输入使用，用以限制主回路上 CV1 输出的积分饱和。	
CV2WindupLOut	BOOL	CV2 积分饱和下限指示器。 以下情况下，CV2WindupLOut 为真： • SP1LAlarm 或 SP2LAlarm 为真，或者 • ModelGain 为正且 CV2LAlarm 为真，或者 • ModelGain 为负且 CV2HAlarm 为真。 此信号通常供 WindupLIn 输入使用，用以限制主回路上 CV2 输出的积分饱和。	
CV3WindupLOut	BOOL	CV3 积分饱和下限指示器。 以下情况下，CV3WindupLOut 为真： • SP1LAlarm 或 SP2LAlarm 为真，或者 • ModelGain 为正且 CV3LAlarm 为真，或者 • ModelGain 为负且 CV3HAlarm 为真。 此信号通常供 WindupLIn 输入使用，用以限制主回路上 CV3 输出的积分饱和。	
ProgOper	BOOL	程序/操作员控制指示器。程序控制模式下为真。操作员控制模式下为假。	
CV1Auto	BOOL	CV1 的自动模式指示器。当 CV1 处于自动模式时为真。	
CV2Auto	BOOL	CV2 的自动模式指示器。当 CV2 处于自动模式时为真。	
CV2Auto	BOOL	CV3 的自动模式指示器。当 CV3 处于自动模式时为真。	

输出参数	数据类型	说明	值
CV1Manual	BOOL	CV1 的手动模式指示器。当 CV1 处于手动模式时为真。	
CV2Manual	BOOL	CV2 的手动模式指示器。当 CV2 处于手动模式时为真。	
CV3Manual	BOOL	CV3 的手动模式指示器。当 CV3 处于手动模式时为真。	
CV1Override	BOOL	CV1 的超控模式指示器。当 CV1 处于超控模式时为真。	
CV2Override	BOOL	CV2 的超控模式指示器。当 CV2 处于超控模式时为真。	
CV3Override	BOOL	CV3 的超控模式指示器。当 CV3 处于超控模式时为真。	
CV1Hand	BOOL	CV1 的手控模式指示器。当 CV1 处于手控模式时为真。	
CV2Hand	BOOL	CV2 的手控模式指示器。当 CV2 处于手控模式时为真。	
CV3Hand	BOOL	CV3 的手控模式指示器。当 CV3 处于手控模式时为真。	
DeltaT	REAL	两次更新间隔的时间（秒）。	
CV1StepSizeUsed	REAL	调谐中使用的实际 CV1 步长。	
CV2StepSizeUsed	REAL	调谐中使用的实际 CV2 步长。	
CV3StepSizeUsed	REAL	调谐中使用的实际 CV3 步长。	
CV1PV1GainTuned	REAL	调谐完毕后计算出的 CV1 - PV1 内模增益值。	
CV2PV1GainTuned	REAL	调谐完毕后计算出的 CV2 - PV1 内模增益值。	
CV3PV1GainTuned	REAL	调谐完毕后计算出的 CV3 - PV1 内模增益值。	
CV1PV2GainTuned	REA：	调谐完毕后计算出的 CV1 - PV2 内模增益值。	
CV2PV2GainTuned	REAL	调谐完毕后计算出的 CV2 - PV2 内模增益值。	
CV3PV2GainTuned	REA：	调谐完毕后计算出的 CV3 - PV2 内模增益值。	

输出参数	数据类型	说明	值
CV1PV1TCTuned	REAL	调谐完毕后计算出的 CV1 - PV1 内模时间常数。	
CV2PV1TCTuned	REAL	调谐完毕后计算出的 CV2 - PV1 内模时间常数。	
CV3PV1TCTuned	REAL	调谐完毕后计算出的 CV3 - PV1 内模时间常数。	
CV1PV2TCTuned	REAL	调谐完毕后计算出的 CV1 - PV2 内模时间常数。	
CV2PV2TCTuned	REAL	调谐完毕后计算出的 CV2 - PV2 内模时间常数。	
CV3PV2TCTuned	REAL	调谐完毕后计算出的 CV3 - PV2 内模时间常数。	
CV1PV1DTTuned	REAL	调谐完毕后计算出的 CV1 - PV1 内模死区时间。	
CV2PV1DTTuned	REAL	调谐完毕后计算出的 CV2 - PV1 内模死区时间。	
CV3PV1DTTuned	REAL	调谐完毕后计算出的 CV3 - PV1 内模死区时间。	
CV1PV2DTTuned	REAL	调谐完毕后计算出的 CV1 - PV2 内模死区时间。	
CV2PV2DTTuned	REAL	调谐完毕后计算出的 CV2 - PV2 内模死区时间。	
CV3PV2DTTuned	REAL	调谐完毕后计算出的 CV3 - PV2 内模死区时间。	
CV1PV1RespTCTunedS	REAL	调谐完毕后计算出的 CV1 - PV1 慢速响应速度下的控制变量时间常数。	
CV2PV1RespTCTunedS	REAL	调谐完毕后计算出的 CV2 - PV1 慢速响应速度下的控制变量时间常数。	
CV3PV1RespTCTunedS	REAL	调谐完毕后计算出的 CV3 - PV1 慢速响应速度下的控制变量时间常数。	
CV1PV2RespTCTunedS	REAL	调谐完毕后计算出的 CV1 - PV2 慢速响应速度下的控制变量时间常数。	
CV2PV2RespTCTunedS	REAL	调谐完毕后计算出的 CV2 - PV2 慢速响应速度下的控制变量时间常数。	

输出参数	数据类型	说明	值
CV3PV2RespTCTunedS	REAL	调谐完毕后计算出的 CV3 - PV2 慢速响应速度下的控制变量时间常数。	
CV1PV1RespTCTunedM	REAL	调谐完毕后计算出的 CV1 - PV1 中速响应速度下的控制变量时间常数。	
CV2PV1RespTCTunedM	REAL	调谐完毕后计算出的 CV2 - PV1 中速响应速度下的控制变量时间常数。	
CV3PV1RespTCTunedM	REAL	调谐完毕后计算出的 CV3 - PV1 中速响应速度下的控制变量时间常数。	
CV1PV2RespTCTunedM	REAL	调谐完毕后计算出的 CV1 - PV2 中速响应速度下的控制变量时间常数。	
CV2PV2RespTCTunedM	REAL	调谐完毕后计算出的 CV2 - PV2 中速响应速度下的控制变量时间常数。	
CV3PV2RespTCTunedM	REAL	调谐完毕后计算出的 CV3 - PV2 中速响应速度下的控制变量时间常数。	
CV1PV1RespTCTunedF	REAL	调谐完毕后计算出的 CV1 - PV1 快速响应速度下的控制变量时间常数。	
CV2PV1RespTCTunedF	REAL	调谐完毕后计算出的 CV2 - PV1 快速响应速度下的控制变量时间常数。	
CV3PV1RespTCTunedF	REAL	调谐完毕后计算出的 CV3 - PV1 快速响应速度下的控制变量时间常数。	
CV1PV2RespTCTunedF	REAL	调谐完毕后计算出的 CV1 - PV2 快速响应速度下的控制变量时间常数。	
CV2PV2RespTCTunedF	REAL	调谐完毕后计算出的 CV2 - PV2 快速响应速度下的控制变量时间常数。	
CV3PV2RespTCTunedF	REAL	调谐完毕后计算出的 CV3 - PV2 快速响应速度下的控制变量时间常数。	
AtuneCV1PV1On	BOOL	启动 CV1 - PV1 的自动调谐后设置为真。	
AtuneCV2PV1On	BOOL	启动 CV2 - PV1 的自动调谐后设置为真。	
AtuneCV3PV1On	BOOL	启动 CV3 - PV1 的自动调谐后设置为真。	
AtuneCV1PV1Done	BOOL	CV1 - PV1 的自动调谐成功完成后设置为真。	

输出参数	数据类型	说明	值
AtuneCV2PV1Done	BOOL	CV2 - PV1 的自动调谐成功完成后设置为真。	
AtuneCV3PV1Done	BOOL	CV3 - PV1 的自动调谐成功完成后设置为真。	
AtuneCV1PV1Aborted	BOOL	当 CV1 - PV1 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV2PV1Aborted	BOOL	当 CV2 - PV1 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV3PV1Aborted	BOOL	当 CV3 - PV1 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV1PV2On	BOOL	启动 CV1 - PV2 的自动调谐后设置为真。	
AtuneCV2PV2On	BOOL	启动 CV2 - PV2 的自动调谐后设置为真。	
AtuneCV3PV2On	BOOL	启动 CV3 - PV2 的自动调谐后设置为真。	
AtuneCV1PV2Done	BOOL	CV1 - PV2 的自动调谐成功完成后设置为真。	
AtuneCV2PV2Done	BOOL	CV2 - PV2 的自动调谐成功完成后设置为真。	
AtuneCV3PV2Done	BOOL	CV3 - PV2 的自动调谐成功完成后设置为真。	
ATuneCV1PV2Aborted	BOOL	当 CV1-PV2 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
ATuneCV2PV2Aborted	BOOL	当 CV2-PV2 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
ATuneCV3PV2Aborted	BOOL	当 CV3-PV2 的自动调谐由用户中止，或者因自动调谐过程中出错而中止时，设置为真。	
AtuneCV1PV1Status	DINT	CV1 - PV1 的位映射状态。值为 0 时表示未发生故障。	

输出参数	数据类型	说明	值
AtuneCV2PV1Status	DINT	CV2 - PV1 的位映射状态。值为 0 时表示未发生故障。	
AtuneCV3PV1Status	DINT	CV3 - PV1 的位映射状态。 值为 0 时表示未发生故障。	
AtuneCV1PV2Status	DINT	CV1 - PV2 的位映射状态。值为 0 时表示未发生故障。	
AtuneCV2PV2Status	DINT	CV2 - PV2 的位映射状态。值为 0 时表示未发生故障。	
AtuneCV3PV2Status	DINT	CV3 - PV2 的位映射状态。值为 0 时表示未发生故障。	
AtuneCV1PV1Fault	BOOL	CV1 - PV1 的自动调谐产生以下任一故障。	AtuneCV1PV1Status 的位 0
AtuneCV2PV1Fault	BOOL	CV2 - PV1 的自动调谐产生以下任一故障。	AtuneCV2PV1Status 的位 0
AtuneCV3PV1Fault	BOOL	CV3 - PV1 的自动调谐产生以下任一故障。	AtuneCV3PV1Status 的位 0
AtuneCV1PV1OutOfLimit	BOOL	在 CV1 - PV1 自动调谐过程中，PV1 或 PV1 死区时间步提前预测值超过 PV1TuneLimit。该值为真时，CV1 - PV1 自动调谐过程将中止。	AtuneCV1PV1Status 的位 1
AtuneCV2PV1OutOfLimit	BOOL	在 CV2 - PV1 自动调谐过程中，PV1 或 PV1 死区时间步提前预测值超过 PV1TuneLimit。该值为真时，CV2 - PV1 自动调谐过程将中止。	AtuneCV2PV1Status 的位 1
AtuneCV3PV1OutOfLimit	BOOL	在 CV3 - PV1 自动调谐过程中，PV1 或 PV1 死区时间步提前预测值超过 PV1TuneLimit。该值为真时，CV3 - PV1 自动调谐过程将中止。	AtuneCV3PV1Status 的位 1
AtuneCV1PV1ModeInv	BOOL	MMC 模式在自动调谐开始时并非手动模式，或者在 CV1 - PV1 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV1 - PV1 自动调谐不会启动或者将会中止。	AtuneCV1PV1Status 的位 2

输出参数	数据类型	说明	值
AtuneCV2PV1ModeInv	BOOL	MMC 模式在自动调谐开始时并非手动模式，或者在 CV2 - PV1 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV2 - PV1 自动调谐不会启动或者将会中止。	AtuneCV2PV1Status 的位 2
AtuneCV3PV1ModeInv	BOOL	MMC 模式在自动调谐开始时并非手动模式，或者在 CV3 - PV1 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV3 - PV1 自动调谐不会启动或者将会中止。	AtuneCV3PV1Status 的位 2
AtuneCV1PV1WindupFault	BOOL	在 CV1 - PV1 自动调谐开始时或 CV1 - PV1 自动调谐期间，CV1WindupHIn 或 CV1WindupLIn 为真。该值为真时，CV1 - PV1 自动调谐不会启动或者将会中止。	AtuneCV1PV1Status 的位 3
AtuneCV2PV1WindupFault	BOOL	在 CV2 - PV1 自动调谐开始时或 CV2 - PV1 自动调谐期间，CV2WindupHIn 或 CV2WindupLIn 为真。该值为真时，CV2 - PV1 自动调谐不会启动或者将会中止。	AtuneCV2PV1Status 的位 3
AtuneCV3PV1WindupFault	BOOL	在 CV3 - PV1 自动调谐开始时或 CV3 - PV1 自动调谐期间，CV3WindupHIn 或 CV3WindupLIn 为真。该值为真时，CV3 - PV1 自动调谐不会启动或者将会中止。	AtuneCV3PV1Status 的位 3
AtuneCV1PV1StepSize0	BOOL	CV1 - PV1 自动调谐开始时 CV1StepSizeUsed = 0。该值为真时，CV1 - PV1 自动调谐不会启动。	AtuneCV1PV1Status 的位 4
AtuneCV2PV1StepSize0	BOOL	CV2 - PV1 自动调谐开始时 CV2StepSizeUsed = 0。该值为真时，CV2 - PV1 自动调谐不会启动。	AtuneCV2PV1Status 的位 4
AtuneCV3PV1StepSize0	BOOL	CV3 - PV1 自动调谐开始时 CV3StepSizeUsed = 0。该值为真时，CV3 - PV1 自动调谐不会启动。	AtuneCV3PV1Status 的位 4

输出参数	数据类型	说明	值
AtuneCV1PV1LimitsFault	BOOL	在 CV1 - PV1 自动调谐开始时或 CV1 - PV1 自动调谐期间，CV1LimitsInv 和 CVManLimiting 为真。该值为真时，CV1 - PV1 自动调谐不会启动或者将会中止。	AtuneCV1PV1Status 的位 5
AtuneCV2PV1LimitsFault	BOOL	在 CV2 - PV1 自动调谐开始时或 CV2 - PV1 自动调谐期间，CV2LimitsInv 和 CVManLimiting 为真。该值为真时，CV2 - PV1 自动调谐不会启动或者将会中止。	AtuneCV2PV1Status 的位 5
AtuneCV3PV1LimitsFault	BOOL	在 CV3 - PV1 自动调谐开始时或 CV3 - PV1 自动调谐期间，CV3LimitsInv 和 CVManLimiting 为真。该值为真时，CV3 - PV1 自动调谐不会启动或者将会中止。	AtuneCV3PV1Status 的位 5
AtuneCV1PV1InitFault	BOOL	在 CV1 - PV1 自动调谐开始时或 CV1 - PV1 自动调谐期间，CV1Initializing 为真。该值为真时，CV1 - PV1 自动调谐不会启动或者将会中止。	AtuneCV1PV1Status 的位 6
AtuneCV2PV1InitFault	BOOL	在 CV2 - PV1 自动调谐开始时或 CV2 - PV1 自动调谐期间，CV2Initializing 为真。该值为真时，CV2 - PV1 自动调谐不会启动或者将会中止。	AtuneCV2PV1Status 的位 6
AtuneCV3PV1InitFault	BOOL	在 CV3 - PV1 自动调谐开始时或 CV3 - PV1 自动调谐期间，CV3Initializing 为真。该值为真时，CV3 - PV1 自动调谐不会启动或者将会中止。	AtuneCV3PV1Status 的位 6
AtuneCV1PV1EUSpanChanged	BOOL	在 CV1 - PV1 自动调谐期间，CV1EUSpan 或 PV1EUSpan 发生改变。该值为真时，CV1 - PV1 自动调谐过程将中止。	AtuneCV1PV1Status 的位 7
AtuneCV2PV1EUSpanChanged	BOOL	在 CV2 - PV1 自动调谐期间，CV2EUSpan 或 PV1EUSpan 发生改变。该值为真时，CV2 - PV1 自动调谐过程将中止。	AtuneCV2PV1Status 的位 7
AtuneCV3PV1EUSpanChanged	BOOL	在 CV3 - PV1 自动调谐期间，CV3EUSpan 或 PV1EUSpan 发生改变。该值为真时，CV3 - PV1 自动调谐过程将中止。	AtuneCV3PV1Status 的位 7

输出参数	数据类型	说明	值
AtuneCV1PV1Changed	BOOL	CV1Oper（操作员控制模式）或 CV1Prog（程序控制模式）发生改变，或者 CV1 在 CV1 - PV1 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV1 - PV1 自动调谐过程将中止。	AtuneCV1PV1Status 的位 8
AtuneCV2PV1Changed	BOOL	CV2Oper（操作员控制模式）或 CV2Prog（程序控制模式）发生改变，或者 CV2 在 CV2 - PV1 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV2 - PV1 自动调谐过程将中止。	AtuneCV2PV1Status 的位 8
AtuneCV3PV1Changed	BOOL	CV3Oper（操作员控制模式）或 CV3Prog（程序控制模式）发生改变，或者 CV3 在 CV3 - PV1 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV3 - PV1 自动调谐过程将中止。	AtuneCV3PV1Status 的位 8
AtuneCV1PV1Timeout	BOOL	自阶跃测试开始起经过的时间长于 PV1AtuneTimeLimit。该值为真时，CV1 - PV1 自动调谐过程将中止。	AtuneCV1PV1Status 的位 9
AtuneCV2PV1Timeout	BOOL	自阶跃测试开始起经过的时间长于 PV1AtuneTimeLimit。该值为真时，CV2 - PV1 自动调谐过程将中止。	AtuneCV2PV1Status 的位 9
AtuneCV3PV1Timeout	BOOL	自阶跃测试开始起经过的时间长于 PV1AtuneTimeLimit。该值为真时，CV3 - PV1 自动调谐过程将中止。	AtuneCV3PV1Status 的位 9
AtuneCV1PV1NotSettled	BOOL	PV1 变化过大而无法进行 CV1 - PV1 自动调谐。该值为真时，CV1 - PV1 自动调谐过程将中止。等待 PV1 达到更稳定状态再进行 CV1 - PV1 自动调谐。	AtuneCV1PV1Status 的位 10
AtuneCV2PV1NotSettled	BOOL	PV1 变化过大而无法进行 CV2 - PV1 自动调谐。该值为真时，CV2 - PV1 自动调谐过程将中止。等待 PV1 达到更稳定状态再进行 CV2 - PV1 自动调谐。	AtuneCV2PV1Status 的位 10
AtuneCV3PV1NotSettled	BOOL	PV1 变化过大而无法进行 CV3 - PV1 自动调谐。该值为真时，CV3 - PV1 自动调谐过程将中止。等待 PV1 达到更稳定状态再进行 CV3 - PV1 自动调谐。	AtuneCV3PV1Status 的位 10

输出参数	数据类型	说明	值
AtuneCV1PV2Fault	BOOL	CV1 - PV2 的自动调谐产生以下任一故障。	AtuneCV1PV2Status 的位 0
AtuneCV2PV2Faul	BOOL	CV2 - PV2 的自动调谐产生以下任一故障。	AtuneCV2PV2Status 的位 0
AtuneCV3PV2Fault	BOOL	CV3 - PV2 的自动调谐产生以下任一故障。	AtuneCV3PV2Status 的位 0
AtuneCV1PV2OutOfLimit	BOOL	在 CV1 - PV2 自动调谐过程中，PV2 或 PV2 死区时间步提前预测值超过 PV2TuneLimit。该值为真时，CV1 - PV2 自动调谐过程将中止。	AtuneCV1PV2Status 的位 1
AtuneCV2PV2OutOfLimit	BOOL	在 CV2 - PV2 自动调谐过程中，PV2 或 PV2 死区时间步提前预测值超过 PV2TuneLimit。该值为真时，CV2 - PV2 自动调谐过程将中止。	AtuneCV2PV2Status 的位 1
AtuneCV3PV2OutOfLimit	BOOL	在 CV3 - PV2 自动调谐过程中，PV2 或 PV2 死区时间步提前预测值超过 PV2TuneLimit。该值为真时，CV3 - PV2 自动调谐过程将中止。	AtuneCV3PV2Status 的位 1
AtuneCV1PV2ModeInv	BOOL	MMC 模式在自动调谐开始时并非手动模式，或者在 CV1-PV2 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV1-PV2 自动调谐不会启动或者将会中止。	AtuneCV1PV2Status 的位 2
AtuneCV2PV2ModeInv	BOOL	MMC 模式在自动调谐开始时并非手动模式，或者在 CV2-PV2 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV2-PV2 自动调谐不会启动或者将会中止。	AtuneCV2PV2Status 的位 2
AtuneCV3PV2ModeInv	BOOL	MMC 模式在自动调谐开始时并非手动模式，或者在 CV3-PV2 自动调谐过程中由手动模式切换为其他模式。该值为真时，CV3-PV2 自动调谐不会启动或者将会中止。	AtuneCV3PV2Status 的位 2

输出参数	数据类型	说明	值
AtuneCV1PV2WindupFault	BOOL	在 CV1 - PV2 自动调谐开始时或 CV1 - PV2 自动调谐期间，CV1WindupHIn 或 CV1WindupLIn 为真。该值为真时，CV1 - PV2 自动调谐不会启动或者将会中止。	AtuneCV1PV2Status 的位 3
AtuneCV2PV2WindupFault	BOOL	在 CV2 - PV2 自动调谐开始时或 CV2 - PV2 自动调谐期间，CV2WindupHIn 或 CV2WindupLIn 为真。该值为真时，CV2 - PV2 自动调谐不会启动或者将会中止。	AtuneCV2PV2Status 的位 3
AtuneCV3PV2WindupFault	BOOL	在 CV3 - PV2 自动调谐开始时或 CV3 - PV2 自动调谐期间，CV3WindupHIn 或 CV3WindupLIn 为真。该值为真时，CV3 - PV2 自动调谐不会启动或者将会中止。	AtuneCV3PV2Status 的位 3
AtuneCV1PV2StepSize0	BOOL	CV1 - PV2 自动调谐开始时 CV1StepSizeUsed = 0。该值为真时，CV1 - PV2 自动调谐不会启动。	AtuneCV1PV2Status 的位 4
AtuneCV2PV2StepSize0	BOOL	CV2 - PV2 自动调谐开始时 CV2StepSizeUsed = 0。该值为真时，CV2 - PV2 自动调谐不会启动。	AtuneCV2PV2Status 的位 4
AtuneCV3PV2StepSize0	BOOL	CV3 - PV2 自动调谐开始时 CV3StepSizeUsed = 0。该值为真时，CV3 - PV2 自动调谐不会启动。	AtuneC3PV2Status 的位 4
AtuneCV1PV2LimitsFault	BOOL	在 CV1 - PV2 自动调谐开始时或 CV1 - PV2 自动调谐期间，CV1LimitsInv 和 CVManLimiting 为真。该值为真时，CV1 - PV2 自动调谐不会启动或者将会中止。	AtuneCV1PV2Status 的位 5
AtuneCV2PV2LimitsFault	BOOL	在 CV2 - PV2 自动调谐开始时或 CV2 - PV2 自动调谐期间，CV2LimitsInv 和 CVManLimiting 为真。该值为真时，CV2 - PV2 自动调谐不会启动或者将会中止。	AtuneCV2PV2Status 的位 5

输出参数	数据类型	说明	值
AtuneCV3PV2LimitsFault	BOOL	在 CV3 - PV2 自动调谐开始时或 CV3 - PV2 自动调谐期间，CV3LimitsInv 和 CVManLimiting 为真。该值为真时，CV3 - PV2 自动调谐不会启动或者将会中止。	AtuneCV3PV2Status 的位 5
AtuneCV1PV2InitFault	BOOL	在 CV1 - PV2 自动调谐开始时或 CV1 - PV2 自动调谐期间，CV1Initializing 为真。该值为真时，CV1 - PV2 自动调谐不会启动或者将会中止。	AtuneCV1PV2Status 的位 6
AtuneCV2PV2InitFault	BOOL	在 CV2 - PV2 自动调谐开始时或 CV2 - PV2 自动调谐期间，CV2Initializing 为真。该值为真时，CV2 - PV2 自动调谐不会启动或者将会中止。	AtuneCV2PV2Status 的位 6
AtuneCV3PV2InitFault	BOOL	在 CV3 - PV2 自动调谐开始时或 CV3 - PV2 自动调谐期间，CV3Initializing 为真。该值为真时，CV3 - PV2 自动调谐不会启动或者将会中止。	AtuneCV3PV2Status 的位 6
AtuneCV1PV2EUSpanChanged	BOOL	在 CV1 - PV2 自动调谐期间，CV1EUSpan 或 PV2EUSpan 发生改变。该值为真时，CV1 - PV2 自动调谐过程将中止。	AtuneCV1PV2Status 的位 7
AtuneCV2PV2EUSpanChanged	BOOL	在 CV2 - PV2 自动调谐期间，CV2EUSpan 或 PV2EUSpan 发生改变。该值为真时，CV2 - PV2 自动调谐过程将中止。	AtuneCV2PV2Status 的位 7
AtuneCV3PV2EUSpanChanged	BOOL	在 CV2 - PV3 自动调谐期间，CV3EUSpan 或 PV2EUSpan 发生改变。该值为真时，CV3 - PV2 自动调谐过程将中止。	AtuneCV3PV2Status 的位 7
AtuneCV1PV2Changed	BOOL	CV1Oper（操作员控制模式）或 CV1Prog（程序控制模式）发生改变，或者 CV1 在 CV1 - PV2 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV1 - PV2 自动调谐过程将中止。	AtuneCV1PV2Status 的位 8
AtuneCV2PV2Changed	BOOL	CV2Oper（操作员控制模式）或 CV2Prog（程序控制模式）发生改变，或者 CV2 在 CV2 - PV2 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV2 - PV2 自动调谐过程将中止。	AtuneCV2PV2Status 的位 8

输出参数	数据类型	说明	值
AtuneCV3PV2Changed	BOOL	CV3Oper（操作员控制模式）或 CV3Prog（程序控制模式）发生改变，或者 CV3 在 CV3 - PV2 自动调谐期间达到上/下限或 ROC 限制。该值为真时，CV3 - PV2 自动调谐过程将中止。	AtuneCV3PV2Status 的位 8
AtuneCV1PV2Timeout	BOOL	自阶跃测试开始起经过的时间长于 PV2AtuneTimeLimit。该值为真时，CV1 - PV2 自动调谐过程将中止。	AtuneCV1PV2Status 的位 9
AtuneCV2PV2Timeout	BOOL	自阶跃测试开始起经过的时间长于 PV2AtuneTimeLimit。该值为真时，CV2 - PV2 自动调谐过程将中止。	AtuneCV2PV2Status 的位 9
AtuneCV3PV2Timeout	BOOL	自阶跃测试开始起经过的时间长于 PV2AtuneTimeLimit。该值为真时，CV3 - PV2 自动调谐过程将中止。	AtuneCV3PV2Status 的位 9
AtuneCV1PV2NotSettled	BOOL	PV2 变化过大而无法进行 CV1-PV2 自动调谐。该值为真时，CV1-PV2 自动调谐过程将中止。等待 PV2 达到更稳定状态再进行 CV1-PV2 自动调谐。	AtuneCV1PV2Status 的位 10
AtuneCV2PV2NotSettled	BOOL	PV2 变化过大而无法进行 CV2-PV2 自动调谐。该值为真时，CV2-PV2 自动调谐过程将中止。等待 PV2 达到更稳定状态再进行 CV2-PV2 自动调谐。	AtuneCV2PV2Status 的位 10
AtuneCV3PV2NotSettled	BOOL	PV2 变化过大而无法进行 CV3-PV2 自动调谐。该值为真时，CV3-PV2 自动调谐过程将中止。等待 PV2 达到更稳定状态再进行 CV3-PV2 自动调谐。	AtuneCV3PV2Status 的位 10
Status1	DINT	功能块的位映射状态。值为 0 时表示未发生故障。任何可能配置为无效值的参数都必须具有状态参数位，来指示其无效状态。	
Status2	DINT	功能块的附加位映射状态。值为 0 时表示未发生故障。任何可能配置为无效值的参数都必须具有状态参数位，来指示其无效状态。	
Status3CV1	DINT	功能块的附加位映射 CV1 状态。值为 0 时表示未发生故障。	

输出参数	数据类型	说明	值
Status3CV2	DINT	功能块的附加位映射 CV2 状态。值为 0 时表示未发生故障。	
Status3CV3	DINT	功能块的附加位映射 CV3 状态。值为 0 时表示未发生故障。	
InstructFault	BOOL	功能块发生故障。指示 Status1、Status2 和 Status3CV(n) 相应位的状态，其中 (n) 可以是 1、2 或 3。	Status1 的位 0
PV1Faulted	BOOL	过程变量 PV1 状况不良。	Status1 的位 1
PV2Faulted	BOOL	过程变量 PV2 状况不良。	Status1 的位 2
PV1SpanInv	BOOL	PV1 量程无效，PV1EUMax < PV1EUMin。	Status1 的位 3
PV2SpanInv	BOOL	PV2 量程无效，PV2EUMax < PV2EUMin。	Status1 的位 4
SP1ProgInv	BOOL	SP1Prog < SP1LLimit 或 > SP1HLimit。将限制 SP1 的值。	Status1 的位 5
SP2ProgInv	BOOL	SP2Prog < SP2LLimit 或 > SP2HLimit。将限制 SP2 的值。	Status1 的位 6
SP1OperInv	BOOL	SP1Oper < SP1LLimit 或 > SP1HLimit。将限制 SP1 的值。	Status1 的位 7
SP2OperInv	BOOL	SP2Oper < SP2LLimit 或 > SP2HLimit。将限制 SP2 的值。	Status1 的位 8
SP1LimitsInv	BOOL	限值无效：SP1LLimit < PV1EUMin、SP1HLimit > PV1EUMax 或 SP1HLimit < SP1LLimit。如果 SP1HLimit < SP1LLimit，则使用 SP1LLimit 对值进行限制。	Status1 的位 9
SP2LimitsInv	BOOL	限值无效：SP2LLimit < PV2EUMin、SP2HLimit > PV2EUMax 或 SP2HLimit < SP2LLimit。如果 SP2HLimit < SP2LLimit，则使用 SP2LLimit 对值进行限制。	Status1 的位 10
SampleTimeTooSmall	BOOL	模型死区时间/DeltaT 必须小于或等于 200。	Status1 的位 11
PV1FactorInv	BOOL	输入的 Factor1 值 < 0。	Status1 的位 12
PV2FactorInv	BOOL	输入的 Factor2 值 < 0。	Status1 的位 13
TimingModeInv	BOOL	输入的 TimingMode 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	Status2 的位 27

输出参数	数据类型	说明	值
RTSMissed	BOOL	仅用于实时采样模式。ABS(DeltaT – RTSTime) > 1 毫秒时为真。	Status2 的位 28。
RTSTimeInv	BOOL	输入的 RTSTime 无效。	Status2 的位 29。
RTSTimeStampInv	BOOL	RTSTimeStamp 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	Status2 的位 30。
DeltaTInv	BOOL	DeltaT 无效。如果当前模式不是超控或手控模式，则设置为手动模式。	Status2 的位 31。
CV1Faulted	BOOL	控制变量 CV1 状况不良。	Status3CV1 的位 0
CV2Faulted	BOOL	控制变量 CV2 状况不良。	Status3CV2 的位 0
CV3Faulted	BOOL	控制变量 CV3 状况不良。	Status3CV3 的位 0
CV1HandFBFaulted	BOOL	CV1 HandFB 值状况不良。	Status3CV1 的位 1
CV2HandFBFaulted	BOOL	CV2 HandFB 值状况不良。	Status3CV2 的位 1
CV3HandFBFaulted	BOOL	CV3 HandFB 值状况不良。	Status3CV3 的位 1
CV1ProgInv	BOOL	CV1Prog < 0 或 > 100，或者当 CVManLimiting 为真时，< CV1LLimit 或 > CV1HLimit。将限制 CV1 的值。	Status3CV1 的位 2
CV2ProgInv	BOOL	CV2Prog < 0 或 > 100，或者当 CVManLimiting 为真时，< CV2LLimit 或 > CV2HLimit。将限制 CV2 的值。	Status3CV2 的位 2
CV3ProgInv	BOOL	CV3Prog < 0 或 > 100，或者当 CVManLimiting 为真时，< CV3LLimit 或 > CV3HLimit。将限制 CV3 的值。	Status3CV3 的位 2
CV1OperInv	BOOL	CV1Oper < 0 或 > 100，或者当 CVManLimiting 为真时，< CV1LLimit 或 > CV1HLimit。将限制 CV1 的值。	Status3CV1 的位 3
CV2OperInv	BOOL	CV2Oper < 0 或 > 100，或者当 CVManLimiting 为真时，< CV2LLimit 或 > CV2HLimit。将限制 CV2 的值。	Status3CV2 的位 3
CV3OperInv	BOOL	CV3Oper < 0 或 > 100，或者当 CVManLimiting 为真时，< CV3LLimit 或 > CV3HLimit。将限制 CV3 的值。	Status3CV3 的位 3
CV1OverrideValueInv	BOOL	CV1OverrideValue < 0 或 > 100。将限制 CV1 的值。	Status3CV1 的位 4

输出参数	数据类型	说明	值
CV2OverrideValueInv	BOOL	CV2OverrideValue < 0 或 > 100。将限制 CV2 的值。	Status3CV2 的位 4
CV3OverrideValueInv	BOOL	CV3OverrideValue < 0 或 > 100。将限制 CV3 的值。	Status3CV3 的位 4
CV1EUSpanInv	BOOL	CV1EU 的量程无效，CV1EUMax 等于 CV1EUMin。	Status3CV1 的位 5
CV2EUSpanInv	BOOL	CV2EU 的量程无效，CV2EUMax 等于 CV2EUMin。	Status3CV2 的位 5
CV3EUSpanInv	BOOL	CV3EU 的量程无效，CV3EUMax 等于 CV3EUMin。	Status3CV3 的位 5
CV1LimitsInv	BOOL	CV1LLimit < 0、CV1HLimit > 100 或 CV1HLimit <= CV1LLimit。如果 CV1HLimit <= CV1LLimit，则将使用 CV1LLimit 限制 CV1 值。	Status3CV1 的位 6
CV2LimitsInv	BOOL	CV2LLimit < 0、CV2HLimit > 100 或 CV2HLimit <= CV2LLimit。如果 CV2HLimit <= CV2LLimit，则将使用 CV2LLimit 限制 CV2 值。	Status3CV2 的位 6
CV3LimitsInv	BOOL	CV3LLimit < 0、CV3HLimit > 100 或 CV3HLimit <= CV3LLimit。如果 CV3HLimit <= CV3LLimit，则将使用 CV3LLimit 限制 CV3 值。	Status3CV3 的位 6
CV1ROCLimitInv	BOOL	输入的 CV1ROCLimit 值 < 0，禁用 CV1 ROC 限制。	Status3CV1 的位 7
CV2ROCLimitInv	BOOL	输入的 CV2ROCLimit 值 < 0，禁用 CV2 ROC 限制。	Status3CV2 的位 7
CV3ROCLimitInv	BOOL	输入的 CV3ROCLimit 值 < 0，禁用 CV3 ROC 限制。	Status3CV3 的位 7
CV1HandFBInv	BOOL	CV1 HandFB < 0 或 > 100。将限制 CV1 的值。	Status3CV1 的位 8
CV2HandFBInv	BOOL	CV2 HandFB < 0 或 > 100。将限制 CV2 的值。	Status3CV2 的位 8
CV3HandFBInv	BOOL	CV3 HandFB < 0 或 > 100。将限制 CV3 的值。	Status3CV3 的位 8
CV1PV1ModelGainInv	BOOL	CV1PV1ModelGain 为 1.#QNAN 或 -1.#IND。（非数字），或者 ±1.$（无穷大 ∞）。	Status3CV1 的位 9

输出参数	数据类型	说明	值
CV2PV1ModelGainInv	BOOL	输入的 CV2 - PV1 模型增益为 1.#QNAN 或 -1.#IND。（非数字），或者 ±1.$（无穷大 ∞）。	Status3CV2 的位 9
CV3PV1ModelGainInv	BOOL	输入的 CV3 - PV1 模型增益为 1.#QNAN 或 -1.#IND。（非数字），或者 ±1.$（无穷大 ∞）。	Status3CV3 的位 9
CV1PV2ModelGainInv	BOOL	CV1PV2ModelGain 为 1.#QNAN 或 -1.#IND。（非数字），或者 ±1.$（无穷大 ∞）。	Status3CV1 的位 10
CV2PV2ModelGainInv	BOOL	输入的 CV2 - PV2 模型增益为 1.#QNAN 或 -1.#IND。（非数字），或者 ±1.$（无穷大 ∞）。	Status3CV2 的位 10
CV3PV2ModelGainInv	BOOL	输入的 CV3 - PV2 模型增益为 1.#QNAN 或 -1.#IND。（非数字），或者 ±1.$（无穷大 ∞）。	Status3CV3 的位 10
CV1PV1ModelTCInv	BOOL	输入的 CV1 - PV1 模型时间常数值 < 0。	Status3CV1 的位 11
CV2PV1ModelTCInv	BOOL	输入的 CV2 - PV1 模型时间常数值 < 0。	Status3CV2 的位 11
CV3PV1ModelTCInv	BOOL	输入的 CV3 - PV1 模型时间常数值 < 0。	Status3CV3 的位 11
CV1PV2ModelTCInv	BOOL	输入的 CV1 - PV2 模型时间常数值 < 0。	Status3CV1 的位 12
CV2PV2ModelTCInv	BOOL	输入的 CV2 - PV2 模型时间常数值 < 0。	Status3CV2 的位 12
CV3PV2ModelTCInv	BOOL	输入的 CV3 - PV2 模型时间常数值 < 0。	Status3CV3 的位 12
CV1PV1ModelDTInv	BOOL	输入的 CV1 - PV1 模型死区时间值 < 0。	Status3CV1 的位 13
CV2PV1ModelDTInv	BOOL	输入的 CV2 - PV1 模型死区时间值 < 0。	Status3CV2 的位 13
CV3PV1ModelDTInv	BOOL	输入的 CV3 - PV1 模型死区时间值 < 0。	Status3CV3 的位 13
CV1PV2ModelDTInv	BOOL	输入的 CV1 - PV2 模型死区时间值 < 0。	Status3CV1 的位 14
CV2PV2ModelDTInv	BOOL	输入的 CV2 - PV2 模型死区时间值 < 0。	Status3CV2 的位 14
CV3PV2ModelDTInv	BOOL	输入的 CV3 - PV2 模型死区时间值 < 0。	Status3CV3 的位 14
CV1PV1RespTCInv	BOOL	输入的 CV1 - PV1 响应时间常数值 < 0。	Status3CV1 的位 15
CV2PV1RespTCInv	BOOL	输入的 CV2 - PV1 响应时间常数值 < 0。	Status3CV2 的位 15

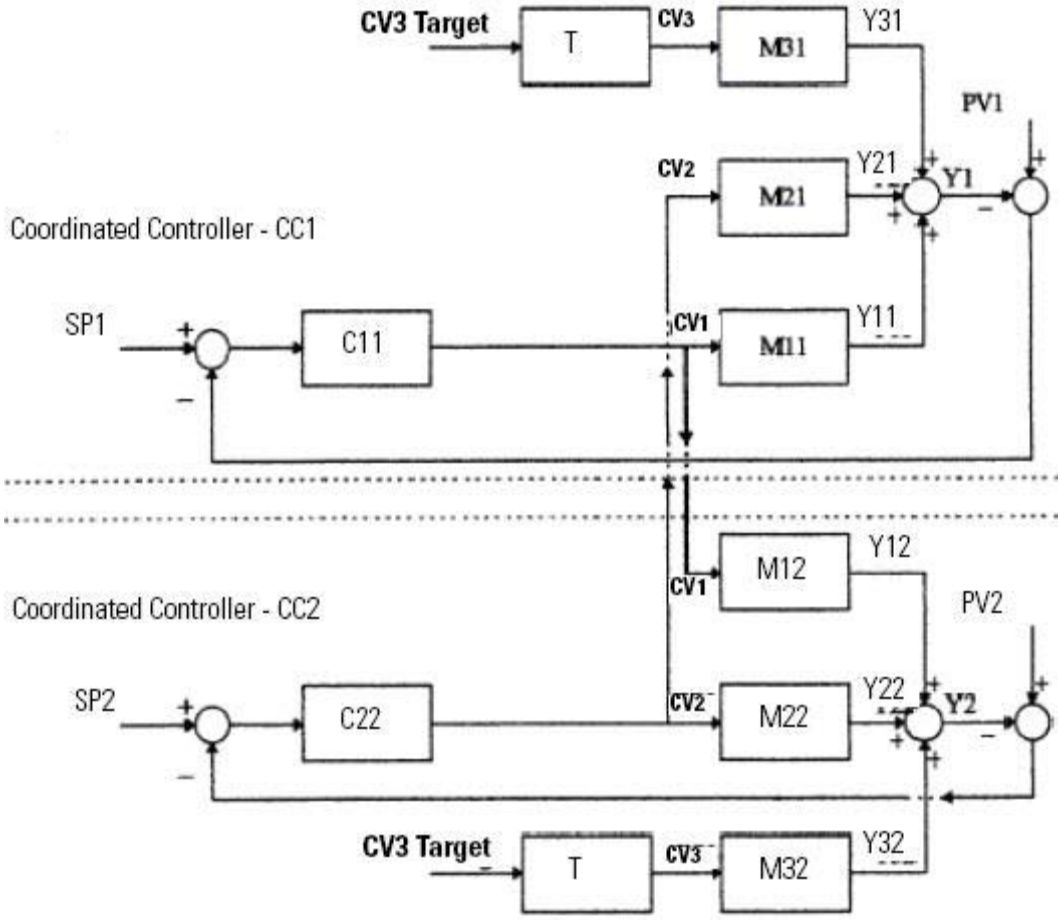
输出参数	数据类型	说明	值
CV3PV1RespTCInv	BOOL	输入的 CV3 - PV1 响应时间常数值 < 0。	Status3CV3 的位 15
CV1PV2RespTCInv	BOOL	输入的 CV1 - PV2 响应时间常数值 < 0。	Status3CV1 的位 16
CV2PV2RespTCInv	BOOL	输入的 CV2 - PV2 响应时间常数值 < 0。	Status3CV2 的位 16
CV3PV2RespTCInv	BOOL	输入的 CV3 - PV2 响应时间常数值 < 0。	Status3CV3 的位 16
CV1TargetInv	BOOL	输入的 CV1 目标值 < 0 或 > 100。	Status3CV1 的位 17
CV2TargetInv	BOOL	输入的 CV2 目标值 < 0 或 > 100。	Status3CV2 的位 17
CV3TargetInv	BOOL	输入的 CV3 目标值 < 0 或 > 100。	Status3CV3 的位 17

说明

MMC 是一种基于模型的灵活算法，可用在两种基本配置模式中：

- 使用三个控制变量控制两个交互作用的过程变量
- 使用两个控制变量控制两个交互作用的过程变量

以下是 MMC 功能块分离器配置示例。



项目	说明
M11	内部模型 CV1 - PV1
M21	内部模型 CV2 - PV1
M31	内部模型 CV3 - PV1
M12	内部模型 CV1 - PV2
M22	内部模型 CV2 - PV2
M32	内部模型 CV3 - PV2
T	目标响应
C11、C22	模型预测功能块 (IMC)，目前分别将控制 PV1 和 PV2，使其达到 SP1 和 SP2
Y11、Y21、Y31、Y12、Y22、Y32	M11、M21、M31、M12、M22、M32 的模型输出

Y1	PV1 预测值
Y2	PV2 预测值
CV1（回流比）	使用协调控制 (CC1) 控制 PV1（塔顶组成）。
CV2（蒸汽流量）	使用协调控制 (CC2) 控制 PV2（塔底组成）。
CV3	驱动目标值达到目标响应。

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。请参见“数组索引编制”部分，了解关于数组索引故障的信息

执行

请注意，在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。

有关更多详细信息，包括所有功能块指令的定义和常规行为，请参见“功能块属性”部分。

阴影区域下方的所有状况仅能在普通扫描模式期间出现。

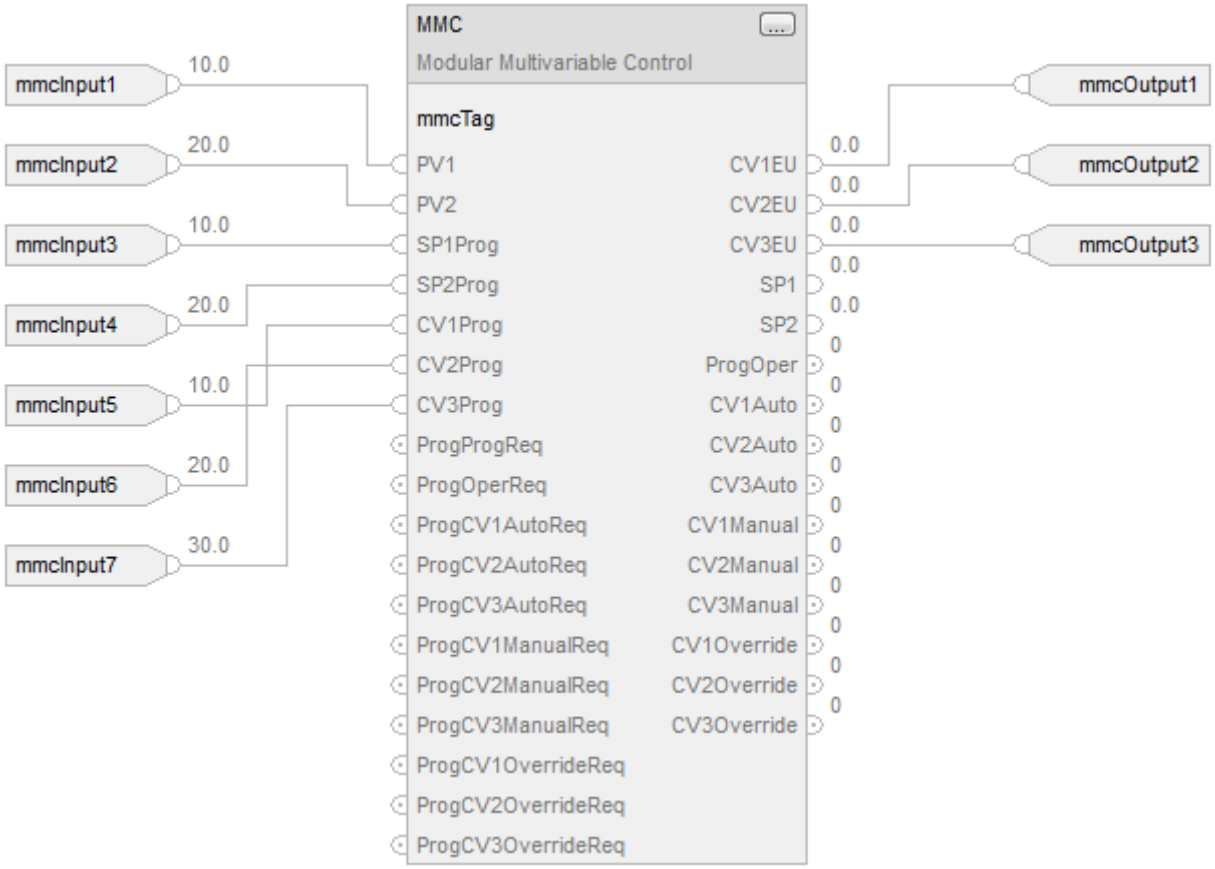
条件/状态	执行的操作
预扫描	.EnableIn 和 .EnableOut 位设置为假。
后扫描	.EnableIn 和 .EnableOut 位设置为假。
EnableIn 为假	.EnableIn 和 .EnableOut 位设置为假。
指令首次运行	不执行任何特定状态下的操作。 不执行主算法，但验证输入参数。
指令首次扫描	不执行任何特定状态下的操作。 不执行主算法，但验证输入参数。
EnableIn 为真	.EnableIn 和 .EnableOut 位设置为真。 将执行指令的主算法，并更新输出。

本地执行

平台	内部/主要功能
ABRisc / ARM	ABRisc 汇编代码 void FB_ModularMultivariableControl(UINT32 *pulArg0Ptr)
RCA	MMC(instance) void FB_ModularMultivariableControl(UINT32 *pulArg0Ptr)
SoftLogix (X86)	void rts$MMC(UINT32 *pFbdBlock)

示例

功能块



结构化文本

```
mmcTag.PV1 := mmcInput1;

mmcTag.PV2 := mmcInput2;

mmcTag.SP1Prog := mmcInput3;
```

```
mmcTag.SP2Prog := mmcInput4;

mmcTag.CV1Prog := mmcInput5;

mmcTag.CV2Prog := mmcInput6;

mmcTag.CV3Prog := mmcInput7;

MMC(mmcTag);

mmcOutput1 := mmcTag.CV1EU;

mmcOutput2 := mmcTag.CV2EU;

mmcOutput3 := mmcTag.CV3EU;
```

另请参见

将 PV 值和 SP 值转换为百分比 参考页数 303

指令首次扫描 参考页数 300

数组索引编制 参考页数 602

结构化文本语法 参考页数 559

功能块属性 参考页数 545

MMC 功能块配置

从默认配置开始，配置以下参数。

参数	说明
PV1EUMax	PV1 的最大标定值。
PV1EUMin	PV1 的最小标定值。
PV2EUMax	PV2 的最大标定值。
PV2EUMin	PV2 的最小标定值。
SP1HLimit	SP1 上限值，以 PV 单位标定。
SP1LLimit	SP1 下限值，以 PV 单位标定。
SP2HLimit	SP2 上限值，以 PV 单位标定。
SP2LLimit	SP2 下限值，以 PV 单位标定。
CV1InitValue	控制变量 CV1 输出的初始值。
CV2InitValue	控制变量 CV2 输出的初始值。
CV3InitValue	控制变量 CV3 输出的初始值。

如果已有过程模型可用，可输入以下参数的值，直观地对 MMC 功能块进行调谐。至此，已完成基本配置。未配置内置调谐器。功能块变量可随时在自动或手动模式下置于在线状态。为进行调谐，将使用默认设置。

如果不了解过程模型，则需要确认这些模型，并利用内置的调谐器（建模器）对功能块进行调谐，以便于功能块在自动模式下正常运行。

参数	说明
ModelGains	非零数（负数表示直接作用的控制变量，正数表示反向作用的控制变量）
ModelTimeConstants	始终为正数
ModelDeadtimes	始终为正数
RespTimeConstants	始终为正数
PV1 和 PV2 的第一个、第二个和第三个主动 CV	指定使用 CV 补偿 PV - SP 误差的顺序。
TargetCV	指定要将哪个 CV 驱动到目标值。
CVTargetValues	指定控制变量驱动各 CV 达到的值（如果选作 TargetCV）。
TargetRespTC	指定 CV 接近目标值的速度。

对于积分过程类型（例如液位控制和位置控制），使用内部非积分模型来近似模拟积分过程。使用 Factor 参数将确定的积分过程模型转换为用于 CV 计算的非积分内部模型。这对于稳定的 MMC 执行非常必要。MMC 功能块可处理积分或非积分过程类型的 PV1 和 PV2 的任何组合。

功能块将一阶滞后与死区时间内部过程模型和一阶滤波器（最多共 24 个调谐参数 - 6 个模型，每个模型 4 个参数）结合使用，计算 CV。计算各个 CV 时，确保各过程变量 (PV) 在接近设置点的值时均遵循一阶滞后轨迹。

响应速度取决于响应时间常数的值。响应时间常数的值越小，控制变量响应的速度越快。设置响应时间常数时，应确保 PV 可根据过程动态在合理的时间内达到设置点。响应时间常数的值越大，控制变量响应的速度越慢，但控制变量也会变得更加稳定。

在手动模式下，将控制变量 (CV) 设为等于操作员输入的手动 CV 参数。为实现从手动模式到自动模式的无扰动转换和控制变量的安全运算，实施了 CV 变化率限制器，以便 CV 在每次扫描时从当前状态移动的幅度不会超过指定的 CV 单位。

将 CVROCPosLimit 和 CVROCNegLimit 置位，对 CV 变化率进行限制。除非 CVManLimiting 已置位，否则控制变量处于手动模式时不会施加变化率限制。

MMC 功能块模型初始化

在以下情况下会进行模型初始化：

- 首次扫描功能块期间
- ModelInit 请求参数置位时
- DeltaT 发生变化时

您可能需要手动调整内模参数或响应时间常数。为此，可更改相应参数并设置适当的 ModelInit 位。功能块的内部状态将进行初始化，相应的位将自动复位。

例如，如果修改 CV2 - PV1 模型的模型，则将 CV2PV1ModelInit 参数设置为真，以初始化 CV2 - PV1 内模参数，并使新模型生效。

MMC 功能块调谐

MMC 功能块配有内部调谐器（建模器）。调谐器的用途是识别过程模型参数并将这些参数用作内模参数（增益、时间常数和死区时间）。调谐器还将计算最佳的响应时间常数。

设置调谐器时，可配置每个 CV - PV 过程的以下参数。

ProcessType	积分（液位、位置控制）或非积分（流量、压力控制）
ProcessGainSign	置位表示过程增益为负值（输出增大会导致 PV 减小）；复位表示过程增益为正值（输出增大会导致 PV 增大）。
ResponseSpeed	慢速、中速或快速，取决于控制目的
NoiseLevel	PV 的估计噪声等级（低、中、高），使调谐器能够区分哪些 PV 变化是随机噪声，哪些是由 CV 阶跃变化引起的
StepSize	非零正数或负数，分别定义正向或负向的 CV 阶跃变化幅值。
PVTuneLimit	（仅适用于积分过程类型）采用 PV 工程单位，定义容许的因 CV 变化导致的 PV 变化量，超出此限制时会中止调谐测试

将 AtuneStart 位（例如 AtuneCV1Start）置位，以此启动调谐器。要停止调谐，可将 AtuneAbort 位置位。

调谐成功完成后，相应的 GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数会根据调谐结果进行更新，而且 AtuneStatus 代码会置位，指示调谐完成。

可将 AtuneUseModel 位置位，通过这种方式分别将这些参数复制到 ModelGain、ModelTC、ModelDT 和 RespTC。MMC 功能块会自动对

内部变量进行初始化并继续正常工作。它会自动将 AtuneUseModel 位复位。

另请参见

MMC 功能块调谐过程 参考页数 294

MMC 功能块调谐错误 参考页数 294

使用 MMC 功能块实现分离器控制

以下示例说明了使用 MMC 功能块控制分离器的方法。请参见“模块多变量控制 (MMC)”部分的 MMC 功能块分离器配置示例。

项目	说明
PV1	塔顶（较为重要）
PV2	塔底（较不重要）
PV1 的第一个主动 CV	CV1（回流比）
PV1 的第二个主动 CV	CV3（压力设置点）
PV1 的第三个主动 CV	CV2（蒸汽流量）
PV2 的第一个主动 CV	CV2
PV2 的第二个主动 CV	CV3
PV2 的第三个主动 CV	CV1
TargetCV	CV3（如有可能，压力应保持恒定）
CV3Target	60%（压力范围的 60%）

MMC 会计算 CV1、CV2 以及 CV3，以确保按照以下重要性顺序达成控制目标：

1. 控制 PV1，使其达到 SP1（始终认为 PV1 比 PV2 重要）
2. 控制 PV2，使其达到 SP2
3. 控制 CV3，使其达到目标值

将 CV1 选为 PV1 的最主动的控制，将 CV2 选为 PV2 的最主动的控制。如果 CV1 或 CV2 饱和，或者进入手动模式，控制变量会使用 CV3 将 PV1 和 PV2 保持在设置点。

另请参见

模块多变量控制 (MMC) 参考页数 238

MMC 功能块调谐错误

如果调谐过程中发生错误，调谐会中止，相应的 AtuneStatus 位会置位。要中止调谐，可将 AtuneAbort 参数置位。

中止后，CV 会采用阶跃变化之前的值，GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数不会更新。AtuneStatus 参数将指示中止的原因。

MMC 功能块调谐过程

按照以下步骤配置调谐器。

1. 将全部三个 CV 参数设为手动模式。
2. 将相应的 AtuneStart 参数置位。

   调谐器开始采集 PV 和 CV 数据进行噪声计算。
3. 采集 60 个样本 (60*DeltaT) 过后，调谐器会将 StepSize 与 CV 相加。

   成功采集 CV 步更改生成的 PV 数据后，CV 会采用阶跃变化之前的值，AtuneStatus、GainTuned、TCTuned、DTTuned 和 RespTCTuned 参数会更新。
4. 将相应的 AtuneUseModel 参数置位，从而将经过调谐的参数复制到模型参数中

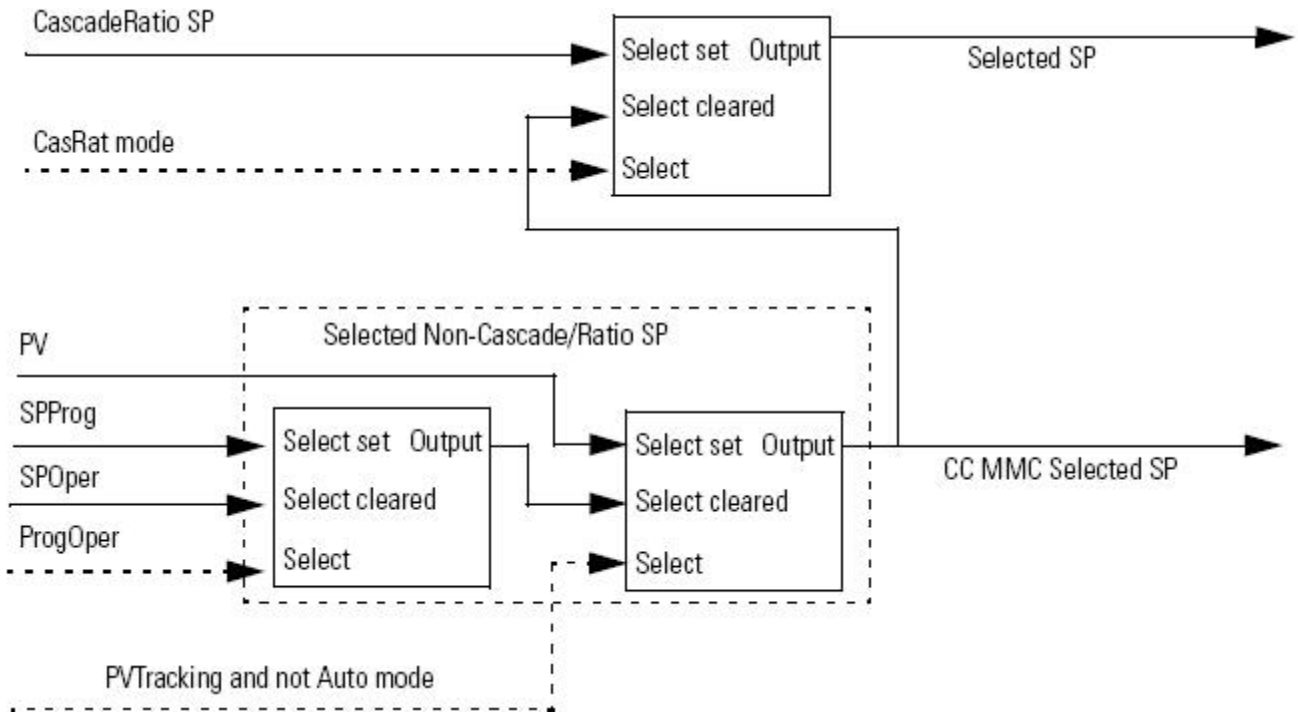
   功能块随即会将 AtuneUseModel 参数复位。

   成功执行自动调谐后，Atune 参数会置 1。调谐成功完成。

要确定全部六个 CV - PV 过程的模型并计算响应时间常数，可将调谐器运行达三次，从而分别获取 CV1 - PV2、CV2 - PV2 和 CV3 - PV2 模型并实现调谐。每次运行后，都会确定两个过程模型：CV - PV1 和 CV - PV2（一次 CV 阶跃变化引起的两个过程变量响应）

当前 SP

当前 SP 取决于级联/比率模式、PVTracking 值、自动模式以及 ProgOper 值。



使用协调控制功能块进行控制

以下示例说明使用协调控制功能块对过程中的温度进行控制的方法。

名称	说明
PV	温度
Act1stCV	CV3（高温蒸汽）
Act2ndCV	CV2（冷却）
Act3rdCV	CV1（低压蒸汽）
Target1stCV	CV2
Target2ndCV	CV3
Target3rdCV	CV1
CV1Target	0% 此值无关紧要，因为在目标列表设置中 CV1 的优先级最低，并且将采用稳态加载来使 PV 保持在设置点。
CV2Target	0%
CV3Target	10%

温度示例说明

对 PV 进行控制使其保持在设置点的优先级最高。高压蒸汽和冷却被选为最为主动的执行器。在稳态下，两个相同的控制应采用其目标值：CV3 为 10%，CV2 为 0%。CV1 将采用可使 PV 保持在设置点的任意值；因此，其目标值无关紧要，这是因为对 PV 进行控制使其保持在设置点是优先级更高的控制目标。目标 CV 优先级和目标 CV 值可在线更改。

CC 功能块会计算 CV1、CV2 以及 CV3，以确保按照以下重要性顺序达到控制目标：

1. 控制 PV，使其达到 SP
2. 控制 CV2，使其达到目标值
3. 控制 CV3，使其达到目标值

至此，已完成基本配置。未配置内置调谐器。控制变量可随时在自动或手动模式下置于在线状态。为进行调谐，将使用默认设置。请参见“CC 功能块调谐”部分。

如果不了解过程模型，则需要确认这些模型，并利用内置的调谐器（建模器）对功能块进行调谐，以便于功能块在自动模式下正常运行。

功能块将一阶滞后与死区时间内部过程模型和一阶滤波器（最多共十二个调谐参数）结合使用，计算 CV。计算各 CV 时，确保过程变量 (PV) 在接近设置点的值时遵循一阶滞后轨迹。

响应速度取决于响应时间常数的值。响应时间常数的值越小，控制变量响应的速度越快。设置响应时间常数时，应确保 PV 可根据过程动态在合理的时间内达到设置点。响应时间常数的值越大，控制变量响应的速度越慢，但控制变量也会变得更加稳定。有关详细信息，请参见调谐部分。

在手动模式下，控制变量 (CV) 设置为等于操作员输入的或程序生成的 CVnOper 或 CVnProg 参数。为实现从手动模式到自动模式的无扰动转换和控制变量的安全运算，实施了 CV 变化率限制器，以便 CV 在每次扫描时从当前状态移动的幅度不会超过指定的 CV 单位。

若要限制 CV 变化率：

- 将 CVnROCPosLimit 和 CVnROCNegLimit 置位

除非 CVManLimiting 已置位，否则控制变量处于手动模式时不会施加变化率限制。

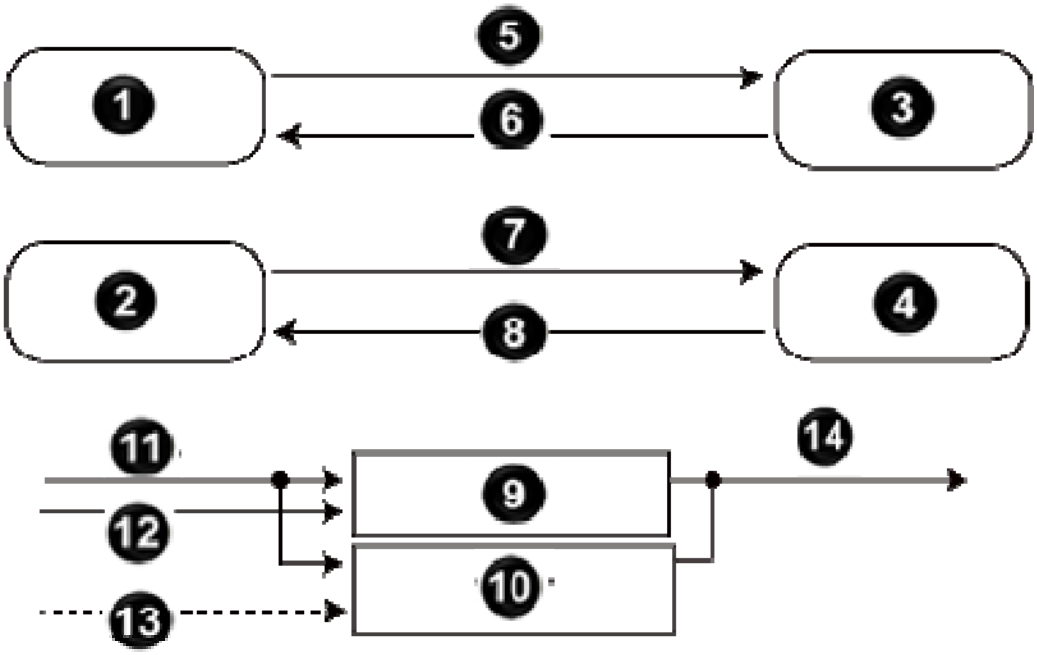
另请参见

CC 功能块调谐 参考页数 212

CV 上限/下限

指令始终根据 CVHLimit 和 CVLLimit 来执行报警。处于自动或级联/比率模式时，CV 由 CVHLimit 和 CVLLimit 进行限制。处于手动模式时，如果 CVManLimiting 置位，则 CV 由 CVHLimit 和 CVLLimit 进行限制。否则，使用 0 和 100% 限制 CV。

请遵循下图所述的原则：



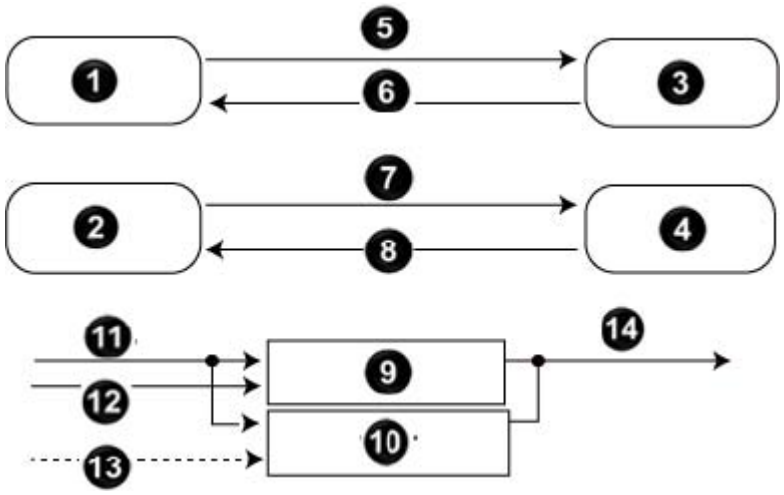
项目	说明
1	CVHAlarm 清零(1)
2	CVLAlarm 清零(1)
3	CVHAlarm 置位
4	CVLAlarm 置位
5	CV 大于 CVHLimit
6	CV 小于或等于 CVHLimit
7	CV 小于 CVLLimit
8	CV 大于或等于 CVLLimit
9	如果 CVHAlarm 置位 CV = CVHLimit

项目	说明
⑩	如果 CVLAlarm 置位 CV = CVLLimit
⑪	0 到 100% 限制算法得出的 CV
⑫	CVHAlarm 置位且处于自动或级联/比率模式，或者 （手动模式且 CVManLimiting 置位）
⑬	CVLAlarm 置位且处于自动或级联/比率模式，或 （手动模式且 CVManLimiting 置位）。
⑭	CV 在 CV 上下限之间。

(1) 在指令首次扫描期间，指令会将报警输出清零。

CV 百分比限制

下图展示指令如何确定限制 CV 百分比的方式。



项目	说明
①	CVHAlarm 清零(1)
②	CVLAlarm 清零(1)
③	CVHAlarm 置位
④	CVLAlarm 置位
⑤	CV 大于 100
⑥	CV 小于或等于 0

项目	说明
❼	CV 小于 0
❽	CV 大于或等于 0
❾	如果 CVHAlarm 置位 CV = 100
❿	如果 CVLAlarm 置位 CV = 100
⓫	积分饱和算法得出的 CV
⓬	CVHAlarm
⓭	CVLAlarm
⓮	CV 在 0 至 100% 之间

(1) 在指令首次扫描期间，指令会将报警输出清零。

CV 变化率限制

处于自动或级联/比率模式时，或者处于手动模式且 CVManLimiting 置位时，PIDE 指令会对 CV 变化率进行限制。使用零值会禁用 CV 变化率限制。

CV 变化率的计算方式如下：

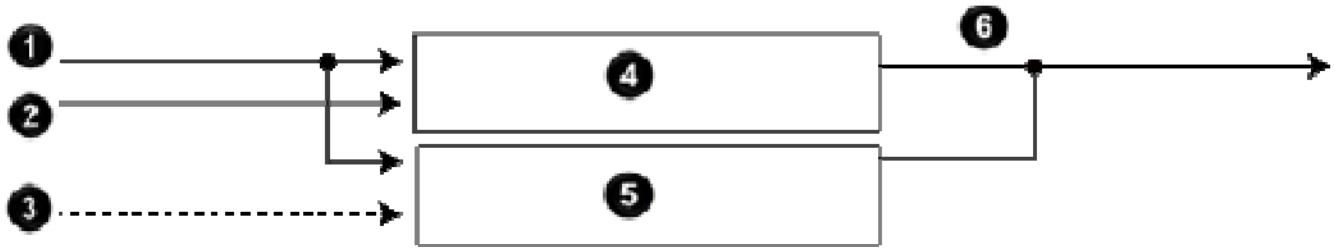
$$CVROC = \left| CV_n - CV_{n-1} \right|$$

$$CVROCDelta = CVROCLimit \times DeltaT$$

其中，DeltaT 以秒为单位。

CV 积分饱和限制

可以对 CV 进行限制，使其值在 WindupHIn 置位时不会增大，或者在 WindupLIn 置位时不会减小。这些输入通常是来自次级回路的 WindupHOut 或 WindupLOut 输出。如果 CVInitializing、CVFault 或 CVEUSpanInv 置位，WindupHIn 和 WindupLIn 输入会被忽略。



项目	说明
❶	选择的 CV
❷	WindupHIn
❸	WindupLIn
❹	如果 WindupHIn 置位且 CV 大于 CV_{n-1} $CV = CV_{n-1}$
❺	如果 WindupHIn 置位且 CV 小于 CV_{n-1} $CV = CV_{n-1}$
❻	积分饱和算法得出的 CV

执行

为 CV 输出设置数学状态标志。

条件	功能块操作	结构化文本操作
预扫描	将 InstructionFirstScan 置位	将 InstructionFirstScan 置位
指令首次扫描	如果将 CVFault 和 CVEUSpanInv 置位，请参见"处理故障"部分。如果将 CVFault 和 CVEUSpanInv 清零： 1. 将 CVInitializing 置位。 2. 如果 PVFault 置位，则将 PVSpanInv 和 SPLimitsInv 清零。请参见"处理故障"部分。 3. 不执行 PID 控制算法。 4. 该指令设置 CVEU = CVInitValue 以及 CV = 相应百分比。 CVInitValue 不受 CVEUmaximum 或 CVEUMin 的限制。当指令以相应的百分比形式计算 CV 时，其范围限制为 0-100。 $CVEU = CVInitValue$ $CV_{n-1} = CV = \frac{CVEU - CVEUMin}{CVEUMax - CVEUMin} \times 100$ $CVOper = CV$ 5. 当 CVInitializing 和 ManualAfterInit 置位时，指令将禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，指令会切换为手动模式。如果 ManualAfterInit 清零，则模式保持不变。 6. 所有操作员请求输入都清零。 7. 如果 ProgValueReset 已置位，则将所有程序请求输入清零 8. 所有 PV 上下限、PV 变化率和偏差上下限报警输出都会清零。 9. 如果 CVInitReq 清零，会将 CVInitializing 清零。	
指令首次运行	将 ProgOper 清零。 指令会切换为手动模式。	将 ProgOper 清零。 指令会切换为手动模式。
EnableIn 清零	将 EnableOut 清零，指令不会执行任何操作，并且输出不会更新。	不适用

EnableIn 置位	指令执行。 将 EnableOut 置位。	始终将 EnableIn 置位。 指令执行。
后扫描	不执行任何操作。	不执行任何操作。

另请参见

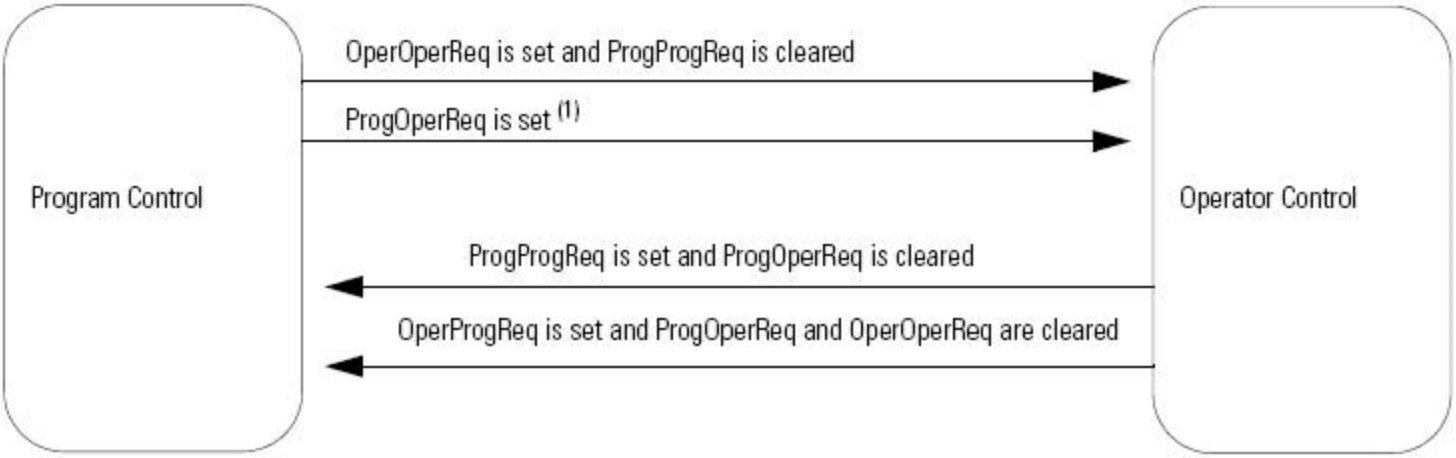
处理故障 参考页数 305

在程序控制与操作员控制之间切换

PIDE 指令可通过用户程序或操作员界面控制。用户可以随时更改控制模式。程序控制和

操作员控制使用相同的 ProgOper 输出。当 ProgOper 置位时，控制模式为程序控制；当 ProgOper 清零时，控制模式则为操作员控制。

下图显示了 PIDE 指令在程序控制与操作员控制之间进行切换的方式。



(1) The instruction remains in Operator control mode when ProgOperReq is set.

有关程序控制和操作员控制的详细信息，请参见“程序/操作员控制”部分。

另请参见

程序/操作员控制 参考页数 555

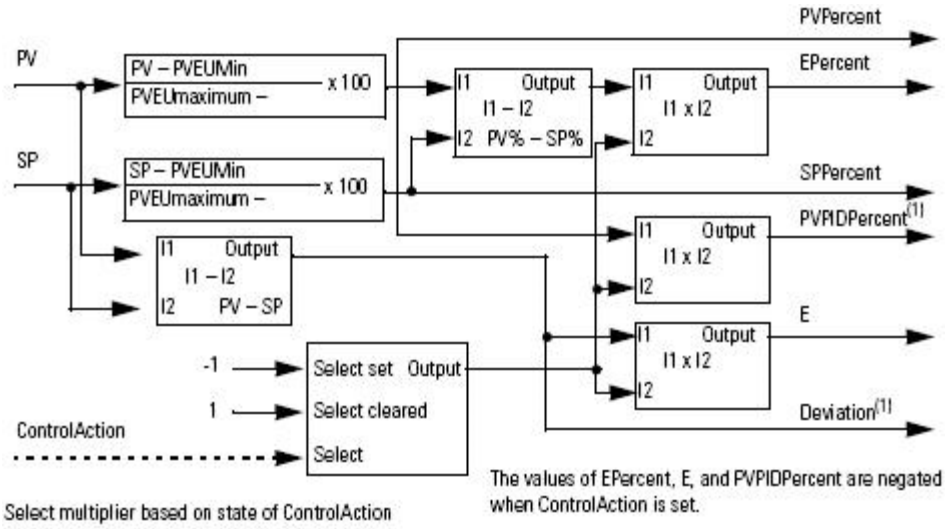
工作模式

处于程序控制模式时，级联/比率、自动和手动模式可由用户程序控制；处于操作员控制模式下时则可通过操作员界面进行控制。超控和手控模式的模式请求输入只能由用户程序控制；这些输入可在程序控制和操作员控制两种模式下工作。

工作模式	说明
级联/比率	处于级联/比率模式时，指令将计算 CV 的变化量。而且，会对 CV 进行相应的调节，使 PV 保持在 SPCascade 值或 SPCascade 值与 Ratio 值的乘积。SPCascade 来自级联控制中主 PID 回路的 CVEU 或比率控制回路的“非可控”流。 可使用 OperCasRatReq 或 ProgCasRatReq 选择级联/比率模式： 将 OperCasRatReq 置位可请求进入级联/比率模式。当 ProgOper、ProgOverrideReq、ProgHandReq、OperAutoReq 或 OperManualReq 置位时，或者 AllowCasRat 清零时，会被忽略。 将 ProgCasRatReq 置位可请求进入级联/比率模式。当 ProgOper 或 AllowCasRat 清零时，或者当 ProgOverrideReq、ProgHandReq、ProgAutoReq 或 ProgManualReq 置位时，会被忽略。
自动	处于自动模式时，指令将计算 CV 的变化量。而且，指令会对 CV 进行相应的调节以使 PV 保持在 SP 值。如果处于程序控制模式，则 SP = SPProg；如果处于操作员控制模式，则 SP = SPOper。 可使用 OperAutoReq 或 ProgAutoReq 选择自动模式： 将 OperAutoReq 置位可请求进入自动模式。当 ProgOper、ProgOverrideReq、ProgHandReq 或 OperManualReq 置位时，会被忽略。 将 ProgAutoReq 置位可请求进入自动模式。当 ProgOper 清零时，或者当 ProgOverrideReq、ProgHandReq 或 ProgManualReq 置位时，会被忽略。
手动	处于手动模式时，指令不会计算 CV 的变化量。CV 的值由控制模式确定。如果处于程序控制模式，则 CV = CVProg；如果处于操作员控制模式，则 CV = CVOper。 可使用 OperManualReq 或 ProgManualReq 选择手动模式： 将 OperManualReq 置位可请求进入手动模式。当 ProgOper、ProgOverrideReq 或 ProgHandReq 置位时，会被忽略。 将 ProgManualReq 置位可请求进入手动模式。当 ProgOper 清零时，或者当 ProgOverrideReq 或 ProgHandReq 置位时，会被忽略。
超控	处于超控模式时，指令不会计算 CV 的变化量。CV = CVOverride，与控制模式无关。超控模式通常用于设置 PID 回路的“安全状态”。 可使用 ProgOverrideReq 选择超控模式： 将 ProgOverrideReq 置位可请求进入超控模式。当 ProgHandReq 清零时，会被忽略。
手控	处于手控模式时，PID 算法不会计算 CV 的变化量。CV = HandFB，与控制模式无关。手控模式通常用于指示最终控制元件的控制功能已由现场手控/自动站接管。 可使用 ProgHandReq 选择手控模式： 将 ProgHandReq 置位可请求进入手控模式。该值通常以数字量输入的形式从手控/自动工作站读取。

将 PV 值和 SP 值转换为百分比

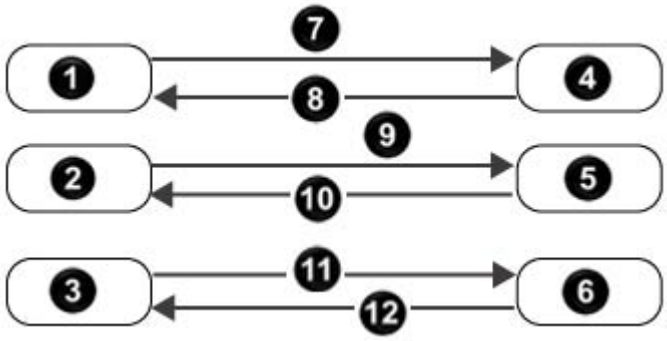
执行 PID 控制算法之前，指令会将 PV 和 SP 转换为百分比并计算误差。此误差为 PV 与 SP 的差值。ControlAction 置位时，需要将 EPercent、E 和 PVPIDPercent 的值取反，然后才能供 PID 算法使用。



(1) PVPIDPercent and Deviation are internal parameters used by the PID control algorithm.

主回路控制

主回路控制通常由主 PID 回路使用，用于在使用级联/比率模式时，实现无扰动切换和抗积分饱和。主回路控制包括初始化主回路输出和抗积分饱和输出。InitPrimary 输出通常供主 PID 回路的 CVInitReq 输入使用。积分饱和输出通常供主回路的积分饱和输入使用，用于限制其 CV 输出的积分饱和。



项目	说明
❶	InitPrimary 清零
❷	WindupHOut 清零(4)
❸	WindupLOut 清零(4)

项目	说明
❹	InitPrimary 置位[1]
❺	WindupHOut 置位
❻	WindupLOut 置位
❼	CVInitializing 置位或未处于级联/比率模式[2]
❽	CVInitializing 清零且处于级联/比率模式[3]
❾	SHPAlarm 置位或出现相应 Cv 报警[5]
❿	SHPAlarm 清零或无 CV 报警[6]
⓫	SPAlarm 置位或出现相应 CV 报警[7]
⓬	SPAlarm 清零或无 CV 报警[8]

1. 首次扫描期间，指令会将 InitPrimary 置位。
2. 当 CVInitializing 置位或处于非级联/比率模式时，指令会将 InitPrimary 置位。
3. 当 CVInitializing 清零且处于级联/比率模式时，指令会将 InitPrimary 清零。
4. 在指令首次扫描期间，指令会将积分饱和输出清零。当 CVInitializing 置位时，或者 CVFaulted 或 CVEUSpanlnv 置位时，指令也会将积分饱和输出清零并禁用 CV 积分饱和算法。
5. 当 SPHAlarm 置位时、ControlAction 清零且 CVHAlarm 置位时，或者 ControlAction 置位且 CVLAlarm 置位时，指令会将 WindupHOut 置位。
   SP 限值和 CV 限值的作用相互独立。SP 上限不会限制 CV 值增大。同样，CV 上下限也不会限制 SP 值增大。
6. 当 SPHAlarm 清零且未设置（ControlAction 清零且 CVHAlarm 置位）及（ControlAction 置位且 CVLAlarm 置位）时，指令会将 WindupHOut 清零。
7. 当 SPLAlarm 置位时、ControlAction 清零且 CVLAlarm 置位时，或者 ControlAction 置位且 CVHAlarm 置位时，指令会将 WindupLOut 置位。
   SP 限值和 CV 限值的作用相互独立。SP 下限不会限制 CV 值增大；同样，CV 上下限也不会限制 SP 值增大。

8. 当 SPLAlarm 清零且未设置（ControlAction 清零且 CVLAlarm 置位）及（ControlAction 置位且 CVHAlarm 置位）时，指令会将 WindupLOut 清零。

处理故障

下表说明指令处理执行故障的方式：

故障条件	操作
CVFaulted 或 CVEUSpanInv 置位	• 指令不进行初始化，将 CVInitializing 清零。 • 计算 PV 和 SP 百分比以及误差，更新 EPercent 和 PVPIDPercent 的内部参数 • 不执行 PID 控制算法。 • 禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式。 • 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。
CVinitRequest	请参见"执行"部分。
PV 状况不良	• 禁用自动模式和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式。 • 将 PV 上下限、PV 变化率和偏差上下限报警输出设置为假 • 不执行 PID 控制算法。 • 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或受控）确定的值。
PVFaulted 置位	• 禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式。 • 将 PV 上下限、PV 变化率以及偏差上下限报警输出清零 • 不执行 PID 控制算法。 • 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。
PVSpanInv 置位或 SPLimitsInv 置位	• 禁用自动和级联/比率模式。如果当前模式不是超控或手控模式，则设置为手动模式。 • 不计算 PV 和 SP 百分比。 • 不执行 PID 控制算法。 • 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。
RatioLimitsInv 置位、 CasRat 置位并且 UseRatio 置位	• 如果尚未处于手控或超控模式，则设置为手动模式。 • 禁用级联/比率模式。 • 将 CV 设置为由程序控制或操作员控制以及模式（手动、超控或手控）确定的值。

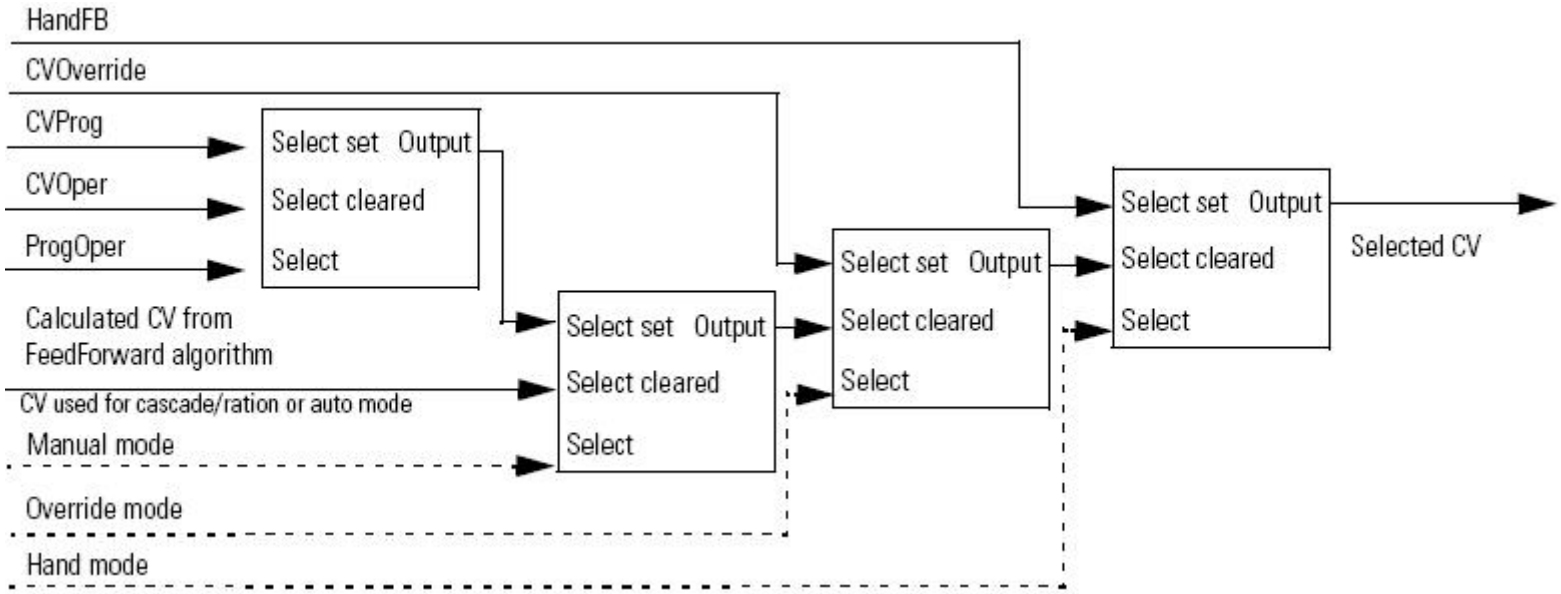
TimingModeInv 置位、RTSTimeStampInv 置位、或 DeltaTInv 置位	• 如果尚未处于手控或超控模式，则设置为手动模式。

另请参见

执行 参考页数 300

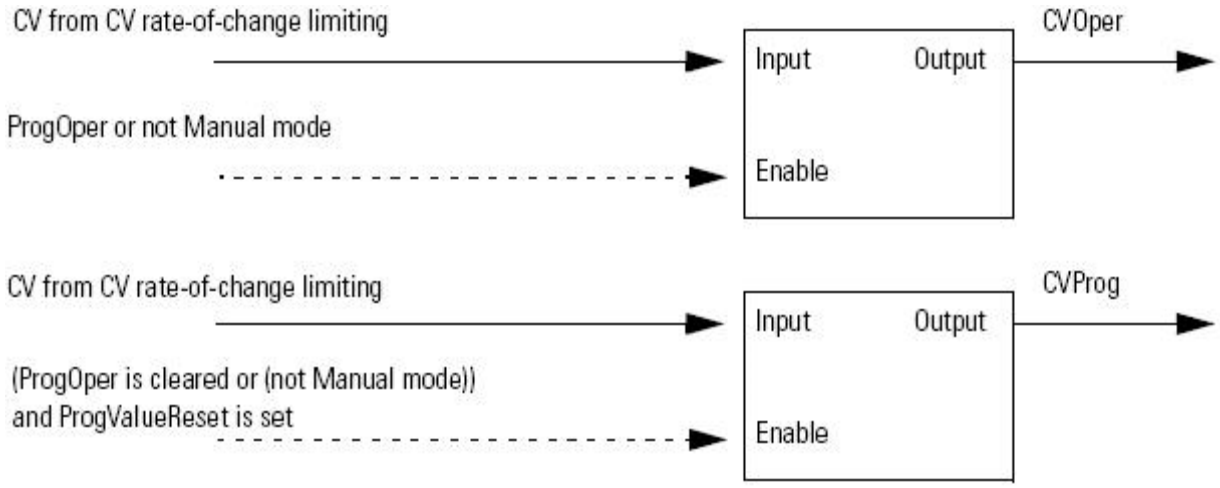
选择控制变量

执行 PID 算法后，根据程序控制或操作员控制模式以及当前 PID 模式选择 CV。



更新 CVOper 和 CVProg 值

如果未处于操作员手动模式，PIDE 指令会设置 CVOper = CV。因此，从任何其他控制模式切换为操作员手动模式时都可实现无扰动切换。



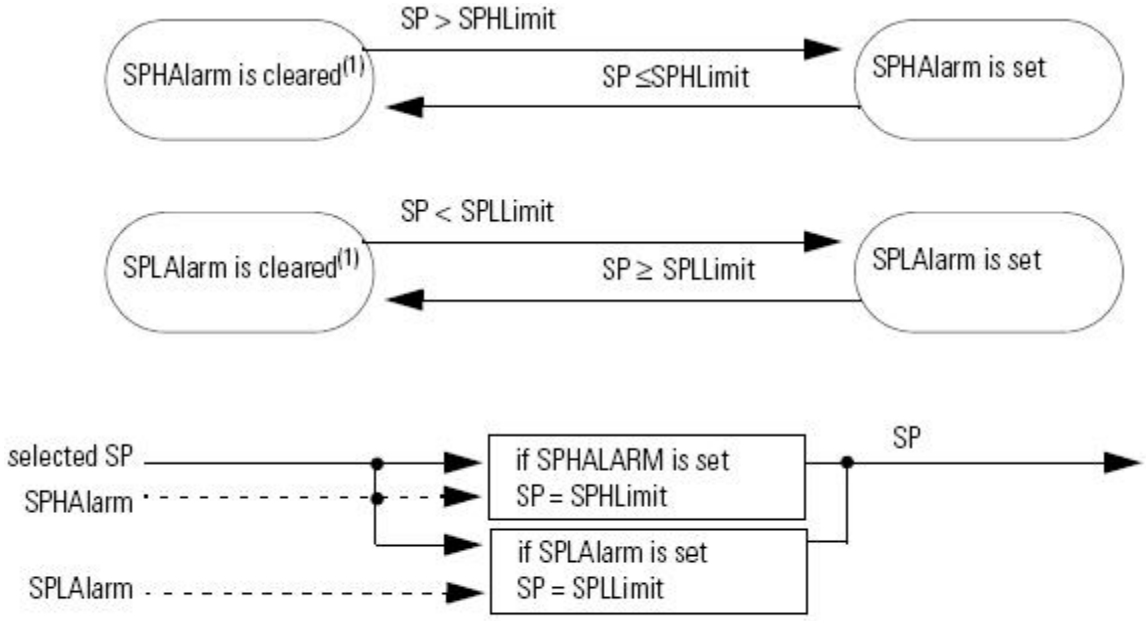
选择设置点

一旦确定处于程序控制还是操作员控制模式以及相应的 PID 模式，指令便可获取正确的 SP 值。

- 增强型 PID 参考页数 75
- 当前 SP 参考页数 295
- SP 上限/下限 参考页数 307

SP 上限/下限

上下限报警算法会将 SP 与 SPHLimit 和 SPLLimit 报警限值进行比较。SPHLimit 不能大于 PVEUmaximum，且 SPLLimit 不能小于 PVEUMin。



(1) During instruction first scan, the instruction clears the SP alarm outputs. The instruction also clears the SP alarm limits and disables the alarming algorithm when PVSpanInv is set.

驱动

驱动指令

驱动指令包含以下信息。

可用指令

梯形图

不可用

功能块和结构化文本

INTG	PI	PMUL	SCRV	SOC	UPDN	HMIBC

如果希望	使用此指令
执行积分运算。	INTG
执行 PI 算法。	PI
通过计算两次扫描之间的输入变化，提供从位置输入模块（如旋转变压器或编码器反馈模块）到数字系统的接口。	PMUL
执行具有附加急动度的斜坡函数。	SCRV
使用增益项、一阶滞后和二阶超前。	SOC
将一个输入加到累加值中，将另一个输入从累加值中减掉。	UPDN
使操作员能够以高准确度和高确定性启动机器控制操作，例如使电机点动或启用阀门等。	HMIBC

另请参见

滤波指令 参考页数 365

逻辑和移动指令 参考页数 453

过程控制指令 参考页数 21

选择/限制指令 参考页数 395

统计指令 参考页数 431

积分器 (INTG)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

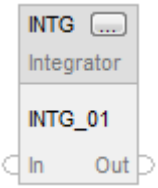
INTG 指令用于实施积分运算。该指令专用于恒速扫描的任务中。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

INTG(INTG_tag);

操作数

功能块

操作数	类型	格式	说明
INTG tag	INTEGRATOR	结构	INTG 结构

INTEGRATOR 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。默认值为置位。

In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	控制算法初始化请求。一旦 Initialize 置位，Output = InitialValue。 有效值 = 任意浮点值 默认值 = 0.0
InitialValue	REAL	指令的初始值。一旦 Initialize 置位，Output = InitialValue。 有效值 = 任意浮点值 默认值 = 0.0
IGain	REAL	积分增益乘数。如果 IGain < 0，该指令会设置 IGain = 0.0，将 Status 中的相应位置位，并将 Output 保持不变。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
HighLimit	REAL	Out 的上限值。如果 HighLimit ≤ LowLimit，指令会将 HighAlarm 和 LowAlarm 置位，将 Status 中的相应位置位，并设置 Out = LowLimit。 有效值 = 任意浮点值 默认值 = 最大正浮点值
LowLimit	REAL	Out 的下限值。如果 HighLimit ≤ LowLimit，指令会将 HighAlarm 和 LowAlarm 置位，将 Status 中的相应位置位，并设置 Out = LowLimit。 有效值 = 任意浮点值 默认值 = 最大负浮点值
HoldHigh	BOOL	输出保持高位请求。置位时，不允许 Out 值增大。 默认清零。
HoldLow	BOOL	输出保持低位请求。置位时，不允许 Out 值减小。 默认清零。

TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
HighAlarm	BOOL	上限报警指示器。当 Out ≥ HighLimit 时，会将 HighAlarm 置位，并将输出限制为 HighLimit 的值。
LowAlarm	BOOL	下限报警指示器。当 Out ≤ LowLimit 时，会将 LowAlarm 置位，并将输出限制为 LowLimit 的值。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
IGainInv (Status.1)	BOOL	IGain > 最大值或 IGain < 最小值。

HighLowLimsInv (Status.2)	BOOL	HighLimit ≤ LowLimit。
TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS|DeltaT - RTSTime|> 1 (0.001 秒) 时置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
INTG tag	INTEGRATOR	结构	INTG 结构

有关结构化文本中表达式语法的详细信息，请参见"结构化文本语法"部分。

说明

INTG 指令专用于恒速扫描的任务中。

当 Initialize 清零且 DeltaT > 0 时，INTG 指令将执行该控制算法。

$$Out = IGain \times \frac{In + In_{n-1}}{2} \times DeltaT + Out_{n-1}$$

只要计算出的输出值无效（NAN 或 ± INF），指令就会将 Out 设为无效值。内部参数不会更新。在每次的后续扫描过程中，都会使用上一次扫描（输出有效）的内部参数计算输出。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见"通用属性"部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	内部参数和 Out 设置为 0。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

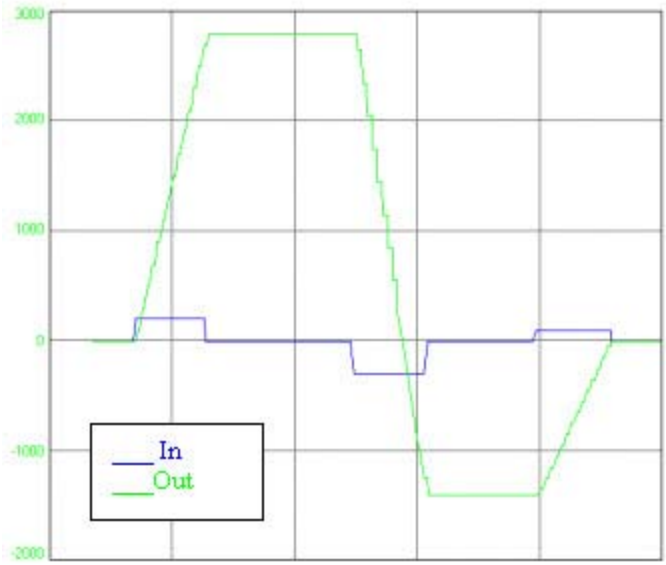
示例

在许多应用中，闭环调节器设计中都会包含积分增益分量，用以消除或最大限度减少所调节系统中的误差。纯比例调节器并不能完全消除系统中的误差。而使用比例增益和积分增益的调节器则可以将一定时间内的误差信号完全消除。INTG 指令使用以下公式计算输出值。

$$Out = IGain \times \frac{In + In_{n-1}}{2} \times DeltaT + Out_{n-1}$$

下图中，功能块的输入由 0 变为 +200 个单位。在此期间，功能块输出经积分运算后达到 2800 个单位。当 In 由 +200 个单位变为 0 个单位后，Out 保持为 2800 个单位。当 In 由 0 变为 -300 个单位时，Out 的积分值缓慢地减小为 -1400 个单位，直至 In 再变回 0。最后，随着 In 由 0 变为 +100，Out 的积分值再次达到 0，随着 Out 到达 0，In 也设置为 0，二者在此重合。

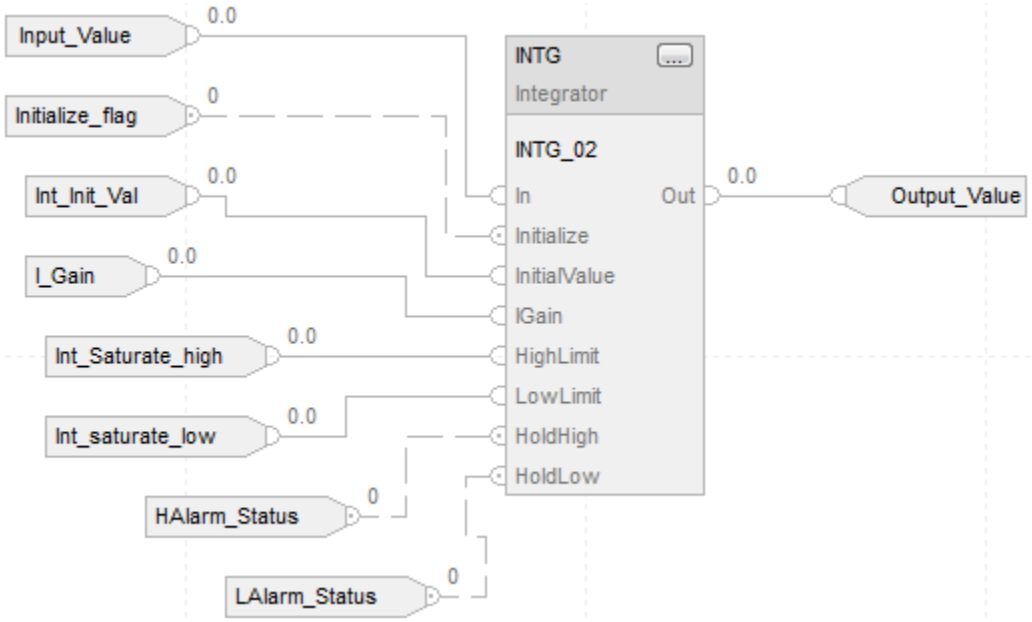
这种积分器的特点是：在函数存在任何输入时沿特定方向连续积分，并在输入为零时保持在任意水平。这正是促使调节器使用积分增益在一定时间范围内完全消除误差的原因所在。



下面的示例将说明 INTG 指令的使用方法。在许多实例中，HighLimit 和 LowLimit 输入会限制积分增益分量在调节器总输出中所占的总控制百分比。另一方面，HoldHigh 和 HoldLow 输入也可用来阻止输出向正方向或负方面继续变化。HoldHigh 和 HoldLow 输入会阻止 INTG 指令在已超出受控变量限值的方向达到“积分饱和”。

功能块

本示例是 INTG 功能块最基本的合法编程方法，仅用于展示该指令的纯文本内容和生成的代码。本示例仅供内部使用，并非可测试的用例。



结构化文本

```
INTG_01.In := Input_Value;

INTG_01.Initialize := Initialize_flag;

INTG_01.InitialValue := Int_Init_Val;

INTG_01.IGain := I_Gain;

INTG_01.HighLimit := Int_saturate_high;

INTG_01.LowLimit := Int_saturate_low;

INTG_01.HoldHigh :=HAlarm_Status;

INTG_01.HoldLow := LAlarm_Status;

INTG(INTG_01);
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

比例 + 积分 (PI)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

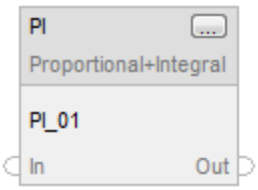
PI 指令提供两种运算方法。第一种方法采用传统 PI 算法，其中比例和积分增益在整个输入信号（误差）范围内保持恒定。第二种方法采用非线性算法，其中比例和积分增益在整个输入信号范围内会发生变化。输入信号是过程设置点与反馈之差。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

PI(PI_tag);

操作数

功能块

操作数	类型	格式	说明
PI tag	PROP_INT	结构	PI 结构

结构化文本

操作数	类型	格式	说明
PI tag	PROP_INT	结构	PI 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

PROP_INT 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	过程误差信号输入。这是设置点与反馈之差。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	指令初始化命令。置位时，Out 和内部积分器设为等于 InitialValue 的值。 默认清零。
InitialValue	REAL	输入初始值。Initialize 置位时，Out 和积分器设为 InitialValue 的值。InitialValue 值由 HighLimit 和 LowLimit 进行限制。 有效值 = 任意浮点值 默认值 = 0
Kp	REAL	比例增益。该值会影响比例和积分控制算法的计算值。如果该值无效，指令会将 Kp 强制设为限值，并将 Status 中的相应位置位。 有效值 = 任意 > 0.0 的浮点值 默认值 = 最小正浮点值
Wld	REAL	超前频率 (弧度/秒)。该值会影响积分控制算法的计算值。如果该值无效，指令会将 Wld 强制设为限值，并将 Status 中的相应位置位。 有效值 = 有效值范围请参见下文的"说明"部分 默认值 = 0.0
HighLimit	REAL	上限值。这是 Out 的最大值。如果 HighLimit ≤ LowLimit，指令会将 HighAlarm 和 LowAlarm 置位，将 Status 中的相应位置位，并设置 Out = LowLimit。 有效值 = LowLimit < HighLimit ≤ 最大正浮点值 默认值 = 最大正浮点值
LowLimit	REAL	下限值。这是 Out 的最小值。如果 HighLimit ≤ LowLimit，指令会将 HighAlarm 和 LowAlarm 置位，将 Status 中的相应位置位，并设置 Out = LowLimit。 有效值 = 最大负浮点值 ≤ LowLimit < HighLimit 默认值 = 最大负浮点值
HoldHigh	BOOL	保持高位命令。置位时，不允许内部积分器的值增大。 默认清零。

输入参数	数据类型	说明
HoldLow	BOOL	保持低位命令。置位时，不允许内部积分器的值减小。 默认清零。
ShapeKpPlus	REAL	正 Kp 修正增益乘数。In ≥ 0 时使用。如果该值无效，指令会将 ShapeKpPlus 强制设为限值，并将 Status 中的相应位置位。NonLinearMode 清零时不使用。 有效值 = 0.1 到 10.0 默认值 = 1.0
ShapeKpMinus	REAL	负 Kp 修正增益乘数。In < 0 时使用。如果该值无效，指令会将 ShapeKpMinus 强制设为限值，并将 Status 中的相应位置位。NonLinearMode 清零时不使用。 有效值 = 0.1 到 10.0 默认值 = 1.0
KpInRange	REAL	比例增益修正范围。定义一个 In（误差）范围，比例增益在该范围内根据 |In| / KpInRange 之比增大或减小。当 |In| > KpInRange 时，指令使用输入的 Kp 修正增益 x (In - KpInRange) 计算比例误差变化。如果该值无效，指令会将 KpInRange 强制设为限值，并将 Status 中的相应位置位。NonLinearMode 清零时不使用。 有效值 = 任意 > 0.0 的浮点值 默认值 = 最大正浮点值
ShapeWldPlus	REAL	正 Wld 修正增益乘数。In ≥ 0 时使用。如果该值无效，指令会将 ShapeWldPlus 强制设为限值，并将 Status 中的相应位置位。NonLinearMode 清零时不使用。 有效值 = 0.0 到 10.0 默认值 = 1.0
ShapeWldMinus	REAL	负 Wld 修正增益乘数。In < 0 时使用。如果该值无效，指令会将 ShapeWldMinus 强制设为限值，并将 Status 中的相应位置位。NonLinearMode 清零时不使用。 有效值 = 0.0 到 10.0 默认值 = 1.0
WldInRange	REAL	积分增益修正范围。定义一个 In（误差）范围，积分增益在该范围内根据 |In| / WldInRange 之比增大或减小。 |In| > WldInRange 时，计算积分误差时，指令会将 In 限制为 WldInRange。如果该值无效，指令会将 WldInRange 强制设为限值，并将 Status 中的相应位置位。NonLinearMode 清零时不使用。 有效值 = 任意 > 0.0 的浮点值 默认值 = 最大正浮点值

输入参数	数据类型	说明
NonLinearMode	BOOL	启用非线性增益模式。置位时，指令会使用由 ParabolicLinear 选择的非线性增益模式来计算实际的比例增益和积分增益。清零时 ,指令会禁用非线性增益模式 ,并将 Kp 和 Wld 值用作比例增益和积分增益。 默认清零。
ParabolicLinear	BOOL	选择非线性增益模式。模式可以是线性模式或抛物线模式。置位时 ,指令使用抛物线增益方法 $y=a*x^2+b$ 来计算实际的比例增益和积分增益。清零时，指令使用线性增益方法 $y=a*x+b$。 默认清零。
TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令的执行状态。
Out	REAL	计算出的 PI 算法输出。该输出的数学状态标志置位。
HighAlarm	BOOL	上限报警指示器。当 Out 的计算值 ≥ HighLimit 且输出和积分器强制设为 HighLimit 时置位。
LowAlarm	BOOL	下限报警指示器。当 Out 的计算值 ≤ LowLimit 且输出和积分器强制设为 LowLimit 时置位。

输出参数	数据类型	说明
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
KpInv (Status.1)	BOOL	Kp < 最小值或 Kp > 最大值。
WldInv (Status.2)	BOOL	Wld < 最小值或 Wld > 最大值。
HighLowLimsInv (Status.3)	BOOL	HighLimit ≤ LowLimit。
ShapeKpPlusInv (Status.4)	BOOL	ShapeKpPlus < 最小值或 ShapeKpPlus > 最大值。
ShapeKpMinusInv (Status.5)	BOOL	ShapeKpMinus < 最小值或 ShapeKpMinus > 最大值。
KpInRangeInv (Status.6)	BOOL	KpInRange < 最小值或 KpInRange > 最大值。
ShapeWldPlusInv (Status.7)	BOOL	ShapeWldPlus < 最小值或 ShapeWldPlus > 最大值。
ShapeWldMinusInv (Status.8)	BOOL	ShapeWldMinus < 最小值或 ShapeWldMinus > 最大值。
WldInRangeInv (Status.9)	BOOL	WldInRange < 最小值或 WldInRange > 最大值。
TimingModeInv (Status.27)	BOOL	TimingMode 无效。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS | DeltaT - RTSTime | > 1（0.001 秒）时置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaT (Status.31)	BOOL	DeltaT 值无效。

说明

PI 指令采用位置式 PI 算法。这意味着，增益项直接应用于输入信号，而非输入信号的变化。PI 指令专用于恒速扫描的任务中。

在非线性算法中，比例增益和积分增益随着输入信号幅值的变化而变化。PI 指令支持两种非线性增益模式：线性和抛物线增益模式。在线性算法中，增益随输入幅值的变化呈线性变化。在抛物线算法中，增益随输入幅值的变化呈抛物线变化。

PI 指令使用以下公式计算 Out：

$$Kp \times \frac{s + Wld}{s}$$

一旦计算出的输出值无效（NAN 或 ± INF），指令会设置 Out = 无效值，并将数学溢出状态标志置位。内部参数不会更新。在每次的后续扫描过程中，都会使用上一次扫描（输出有效）的内部参数计算输出。

在线性模式下工作

在线性模式下，非线性增益模式将被禁用。指令使用 Kp 和 Wld 值作为比例增益和积分增益。计算 Out 值的公式如下：

值	公式
ITerm	$Kp \times Wld \times \frac{WldInput + WldInput_{n-1}}{2} \times DeltaT + ITerm_{n-1}$ 其中，DeltaT 以秒为单位。
PTerm	*Kp x In*
Out	*ITerm + PTerm*

Wld 的限值如下：

- 下限 > 0.0
- High Limit = 0.7π/DeltaT
- WldInput = In

在非线性模式下工作

在非线性模式下，指令使用由 ParabolicLinear 选择的非线性增益模式来计算实际的比例增益和积分增益。

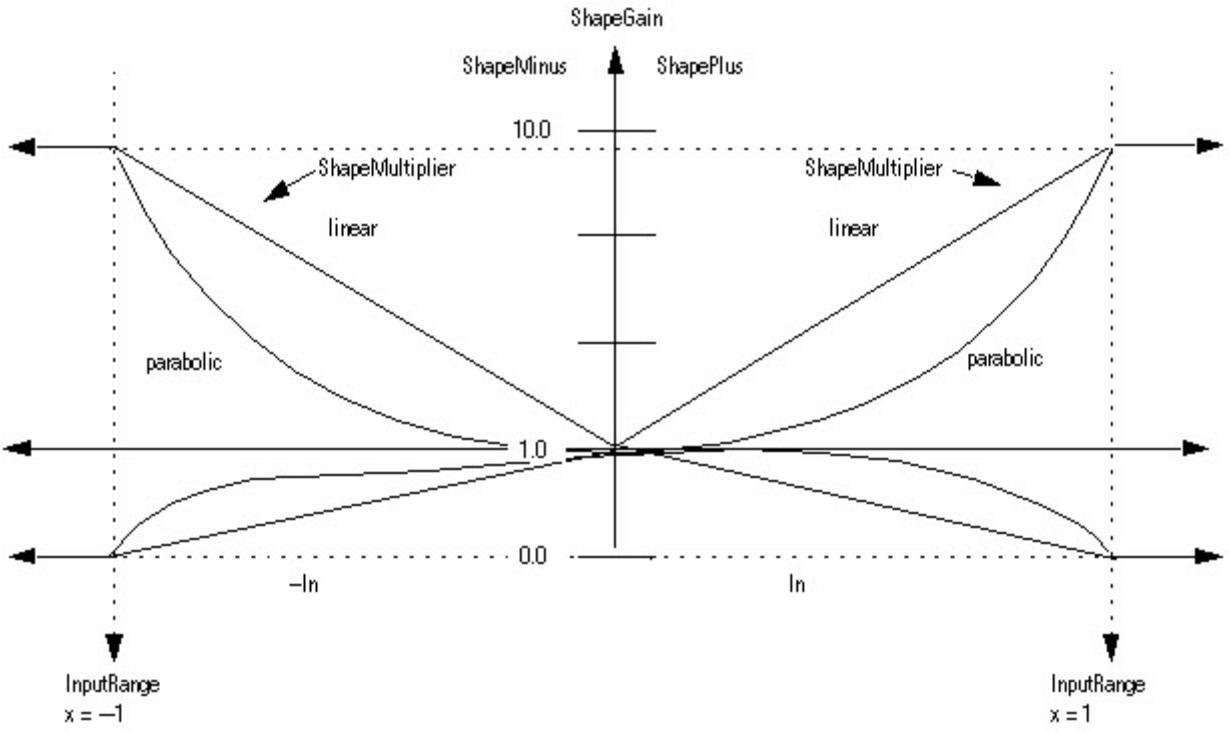
当 In = 0 时，Kp 和 Wld 指定的增益将乘以 1.0。随着误差幅值的变化，相应的比例和积分算法也会分别增大或减小比例或积分增益。这些算法使用输入范围和修正增益参数计算实际的比例增益和积分增益。输入范围将定义进行增益修正的 In（即误差）范围。输入范围由 KpInRange 和 WldInRange 这两个参数设置。修正增益定义在修正增益参数所控制的象限内的增益乘数。修正增益由 ShapeKpPlus、ShapeKpMinus、ShapeWldPlus 和 ShapeWldMinus 设置。

ParabolicLinear 输入用于选择非线性增益模式。如果 ParabolicLinear 清零，则选择线性模式。如果 ParabolicLinear 置位，则选择抛物线模式。

要配置特定的修正增益曲线，可输入 0.0-10.0 的积分修正增益和 0.1-10.0 的比例修正增益，并输入要进行修正的输入范围。将 Kp 和 Wld 与计算出的 ShapeMultiplier 相乘即可得出实际的比例增益和积分增益。如果输入的修正增益为 1.0，则禁用用于为象限计算比例增益或积分增益的非线性算法。

当 In（误差）幅值大于 InRange 时，ShapeMultiplier 等于 |In| 与 InRange 相等时计算出的值。

下图显示了表示抛物线增益和线性增益公式的最大和最小增益曲线。



计算 Out 值的公式如下：

值	公式
Kp 修正增益乘数	If In ≥ 0 then: $KpShapeGain = ShapeKpPlus$ $KpRange = KpInRange$ Else: $KpShapeGain = ShapeKpMinus$ $KpRange = -KpInRange$

Kp 输入比率	If |In| ≤ KpInRange: $KpInputRatio = \|In\| \times \frac{1}{KpInRange}$ Else: $KpInputRatio = 1$
Kp 比率	If not parabolic mode: $KpRatio = KpInputRatio \times 0.5$ If parabolic mode: $KpRatio = KpInputRatio^2 \times 0.333$
Kps 修正增益	$Kps = Kp \times (((KpShapeGain - 1) \times KpRatio) + 1)$
比例输出	If |In| ≤ KpInRange: $PTerm = Kps \times In$ Else, limit gain: $PTerm = Kps \times KpRange + (In - KpRange) \times KpShape$
Wld 修正增益	If In ≥ 0 then: $WldShapeGain = ShapeWldPlus$ Else: $WldShapeGain = ShapeWldMinus$
Wld 输入	If In > WldRange then: $WldInput = WldInRange$ Else if In < –WldInRange then: $WldInput = -WldInRange$ Else: $WldInput = In$
Wld 输入比率	If |In| ≤ WldInRange: $WldInputRange = \|In\| \times \frac{1}{WldInRange}$ Else: $WldInputRange = 1$
Wld 比率	If not parabolic mode: $WldRatio = WldInputRatio$ If parabolic mode: $WldRatio = WldInputRatio^2$
Wlds 修正增益	$Wlds = Wld \times (((WldShapeGain - 1) \times WldRatio) + 1)$
Wlds 限值	$LowLimit > 0$ $HighLimit = \frac{0.7\pi}{DeltaT}$
积分输出	$ITerm = Kps \times Wlds \times \frac{(WldInput + WldInput_{n-1})}{2} \times DeltaT + ITerm_{n-1}$
输出	$Out = PTerm + ITerm$

限制

指令根据保持输入的状态停止 ITerm 的积分饱和。

条件	操作
If HoldHigh is set **and** ITerm > $ITerm_{n-1}$	ITerm = $ITerm_{n-1}$
If HoldLow is set **and** ITerm < $ITerm_{n-1}$	ITerm = $ITerm_{n-1}$

指令还会根据 HighLimit 和 LowLimit 值停止积分器的积分饱和。

条件	操作
Integrator > HighLimit	Integrator = HighLimit
Integrator > LowLimit	Integrator = LowLimit

指令会根据 HighLimit 和 LowLimit 值限制 Out 值。

条件	操作
HighLimit ≤ LowLimit	Out = LowLimit ITerm = LowLimit HighLowLimsInv 置位 HighAlarm 置位 LowAlarm 置位 WldInput = 0
Out ≥ HighLimit	Out = HighLimit ITerm = $ITerm_{n-1}$ HighAlarm 置位
ITerm > HighLimit	ITerm = HighLimit
Out ≤ LowLimit	Out = LowLimit ITerm = $ITerm_{n-1}$ LowAlarm 置位
ITerm < LowLimit	ITerm = LowLimit

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件	执行的操作
预扫描	EnableIn 和 EnableOut 设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	Out n-1 = 0 将不执行用于计算 Out 的算法。
指令首次扫描	Out n-1 = 0 将不执行用于计算 Out 的算法。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

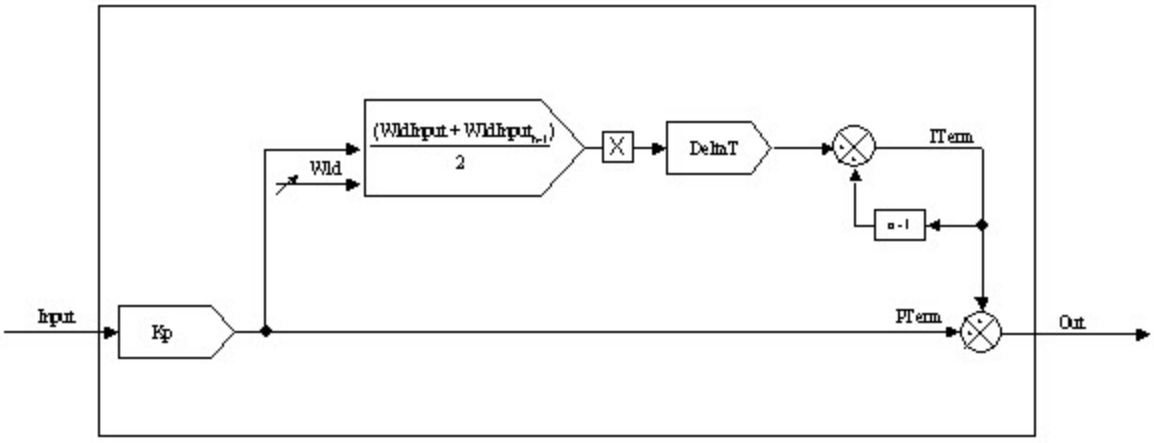
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

PI 指令是包含比例增益分量和积分增益分量的调节指令。积分增益分量由用户设置（弧度/秒），用于设置 PI 调节器的基本频率响应。比例增益用于设置功能块的总增益，包括功能块的比例增益和积分增益。

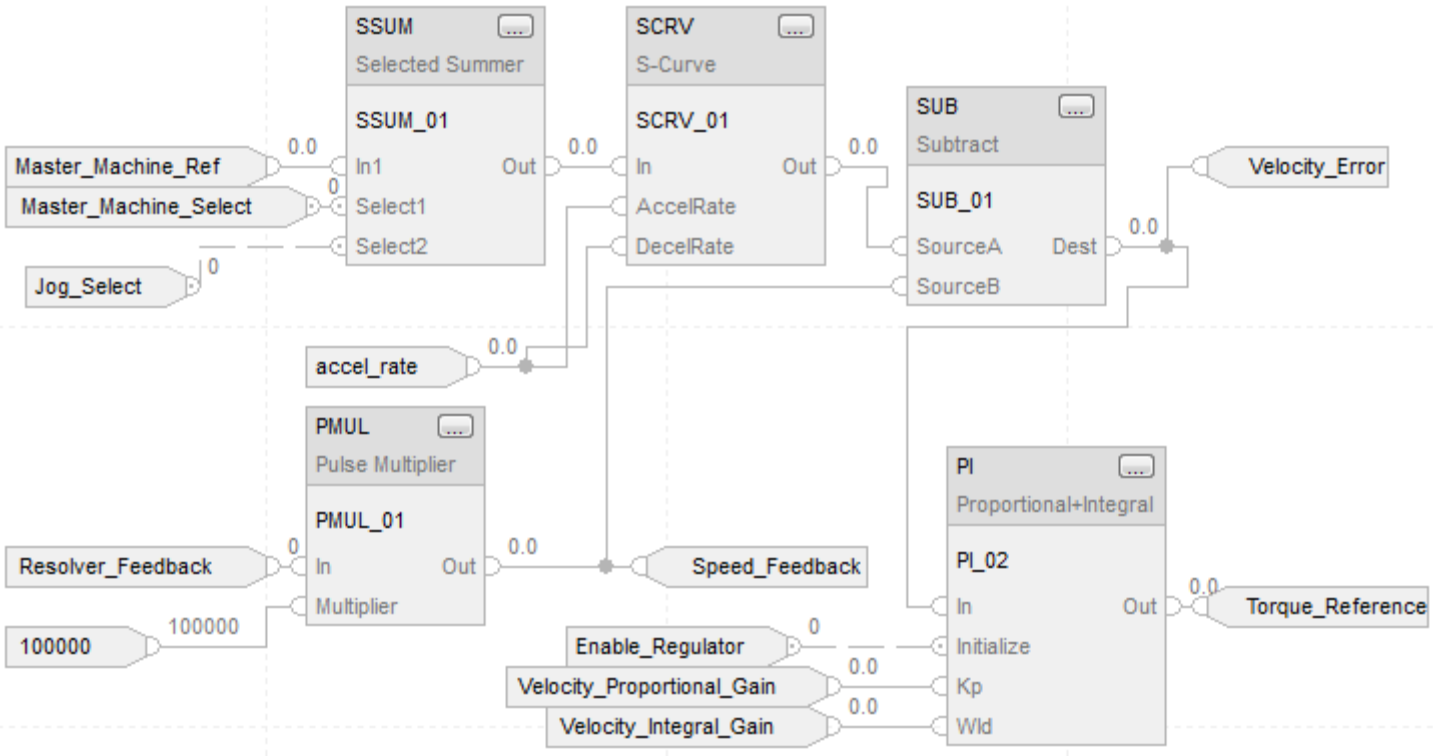
下图显示了 PI 功能块在线性模式下的基本调节回路，其中不包括初始化和保持/钳位功能。

PI 指令：线性模式



在下面的示例中，PI 指令作为速度调节器使用。本例中，速度误差通过将系统速度参考值与速度反馈信号相减（通过 SCRV 指令）得出（参见 PMUL 指令示例）。速度误差直接送入 PI 指令，该指令将根据上图所示的函数对该信号进行处理。

功能块



结构化文本

```
Reference_Select.In1 := Master_Machine_Ref;
Reference_Select.Select1 := Master_Machine_Select;
Reference_Select.In2 := Section_Jog;
Reference_Select.Select2 := Jog_Select;
SSUM(Reference_Select);

S_Curve.In := Reference_Select.Out;
S_Curve.AccelRate := accel_rate;
S_Curve.DecelRate := accel_rate;
SCRV(S_Curve);

PMUL_01.In := Resolver_Feedback;
PMUL_01.WordSize := 12;
PMUL_01.Multiplier := 100000;
PMUL(PMUL_01);

Speed_Feedback := PMUL_01.Out;
Velocity_Error := S_Curve.Out - Speed_Feedback;

PI_01.In := Velocity_Error;
PI_01.Initialize := Enable_Regulator;
PI_01.Kp := Velocity_Proportional_Gain;
PI_01.Wld := Velocity_Integral_Gain;
PI(PI_01);

Torque_Reference := PI_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

脉冲乘法器 (PMUL)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

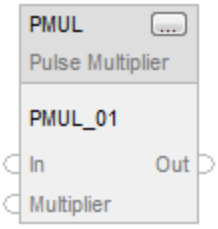
PMUL 指令通过计算两次扫描之间的输入变化，提供从位置输入模块（如旋转变压器或编码器反馈模块）到数字系统的接口。通过选择指定的字长，可将 PMUL 指令配置为采用连续线性的方式通过反转界限进行区分。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

PMUL(PMUL_tag);

操作数

功能块

操作数	类型	格式	说明
PMUL tag	PULSE_MULTIPLIER	结构	PMUL 结构

结构化文本

操作数	类型	格式	说明
PMUL tag	PULSE_MULTIPLIER	结构	PMUL 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

PULSE_MULTIPLIER 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认置位。
In	DINT	指令的模拟量输入信号。 有效值 = 任意 DINT 值 默认值 =0

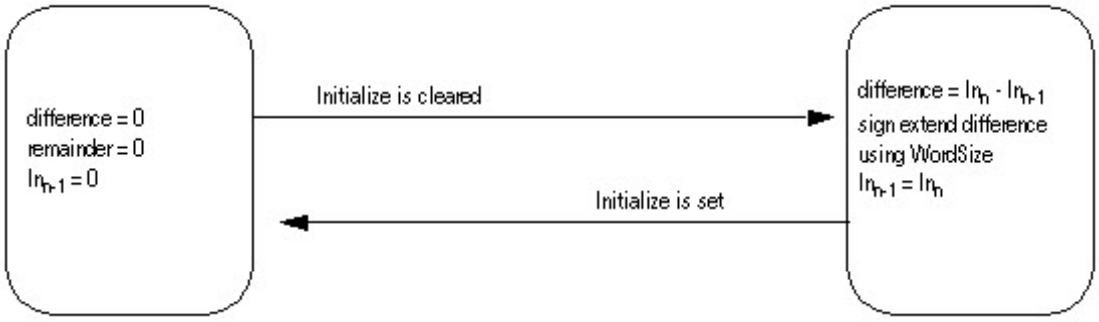
Initialize	BOOL	初始化输入。此参数置位时，Out 将保持为 0.0，并且所有内部寄存器均设为 0。此参数由置位跳变为清零时，In_{n-1} = InitialValue（对绝对模式无效）。此参数清零时，指令会正常执行。
InitialValue	DINT	输入初始值。Initialize 由置位跳变为清零时，In_{n-1} = InitialValue 有效值 = 任意 DINT 值 默认值 = 0。
模式	BOOL	模式输入。此参数置位时可启用相对模式。清零时可启用绝对模式。 默认置位。
WordSize	DINT	字长，以位数计。指定在相对模式下计算 (In_n-In_{n-1}) 时使用的位数。 绝对模式下不使用 WordSize。 如果 In 的变化量大于 $1/2 * 2^{(Wordsize-1)}$，Out 的符号会发生变化。 WordSize 无效时，Out 保持不变，指令会将 Status 中的相应位置位。 有效值 = 2 到 32 默认值 = 14
Multiplier	DINT	乘数。将该值除以 100,000 可控制 In 与 Out 之比。如果该值无效，指令会对值进行限制，并将 Status 中的相应位置位。 有效值 = -1,000,000 到 1,000,000 默认值 = 100,000

输出参数	数据类型	说明
EnableOut	BOOL	使能输出。
Out	REAL	指令的 Out 输出。如果 Out 的计算值溢出，会将 Out 强制设为 +/- 🕶，并将 Status 中的相应位置位。该输出的数学状态标志置位。
状态	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
WordSizeInv (Status.1)	BOOL	WordSize 值无效。
OutOverflow (Status.2)	BOOL	内部输出计算值溢出。
LostPrecision (Status.3)	BOOL	$Out < -2^{24}$ 或 $Out > 2^{24}$。当指令将 Out 由整数值转换为实数值时，如果结果大于 $\|2^{24}\|$，数据会丢失，因为 REAL 数据类型限制在 2^{24}。
MultiplierInv (Status.4)	BOOL	Multiplier 值无效。

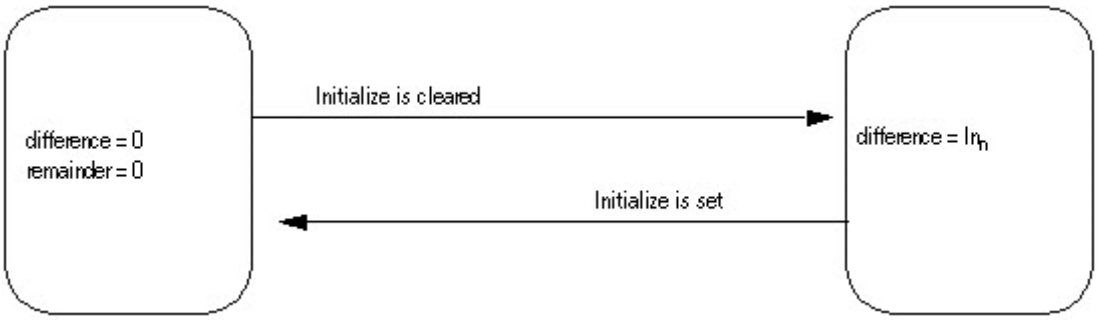
说明

PMUL 指令在相对或绝对模式下工作。

在相对模式下，指令的输出等于两次扫描之间的输入差值乘以 (Multiplier/100,000)。在相对模式下，指令将保存扫描中执行除法运算后得到的任何余数，并在下一次扫描期间重新将其加上。这样，运算过程中便不会丢失位置信息。



在绝对模式下，指令可以在两次扫描之间不丢失任何信息的情况下，对输入（如位置）进行变换。



计算输出和余数

PMUL 指令使用以下公式计算相对模式或绝对模式下的 Out 输出：

Ans = ((DiffInput x Multiplier) + INT_Remainder)

INT_Out = Ans / 100,000

INT_Remainder= Ans - (INT_Out * 100,000)

Out = INT_Out

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件	功能块操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	$In_{n-1} = In$。$Out_{n-1} = 0$。余数 = 0。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

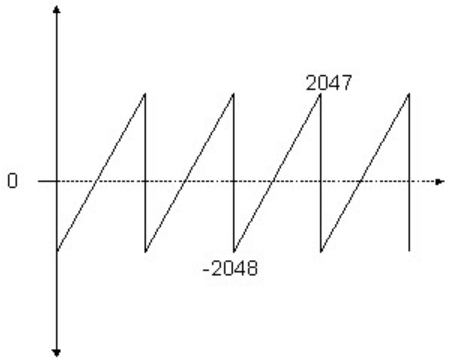
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

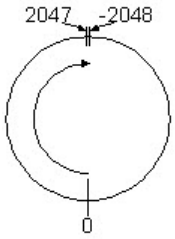
示例 1

PMUL 指令最常用于相对工作模式下。在该模式下，PMUL 指令具有多种用途。首先，在相对模式下，PMUL 指令可以确定两次扫描期间输入端接收信息的差异。在接收到数据后，该指令将输出两次扫描的输入值之差。也就是说，如果第“n”次扫描时 In = 500，第“n+1”次扫描时 In = 600，则第“n+1”次扫描时 Out = 100。

其次，在该工作模式下，PMUL 指令还会对反馈模块提供的二进制数据的“回滚”值进行补偿。例如，旋转变压器反馈模块可能具有 12 位分辨率，以带符号的二进制值表示，介于 –2048 到 2047 之间。根据来源于反馈模块的原始数据，反馈设备的旋转可表示如下：



在本例中，当反馈数据的值从 2047 变为 -2048 时，有效位置变化相当于位置跳变 4095 个计数。但实际上，就旋转变压器反馈设备的旋转而言，该位置变化仅占 1/4096。通过获取由反馈模块输入的数据的真实字长，PMUL 指令将以旋转的方式处理输入数据（如下图所示）：



知晓该功能块输入数据的字长后，当功能块输入由 2047 变为 -2048 时，PMUL 指令会确定两者相差 1 个计数并输出，而不会通过数学运算得出 4095 个计数。

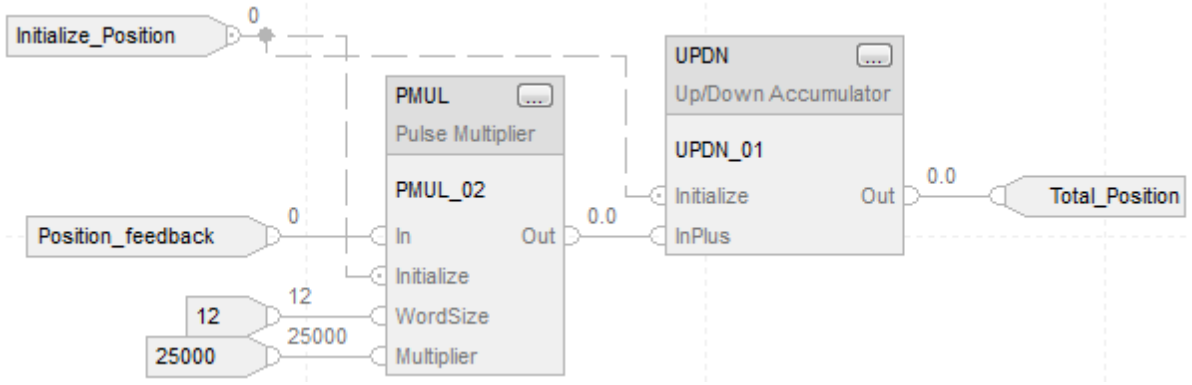
提示： 应用该功能块时，务必要注意的是，如果要正确区分旋转方向，反馈数据在两次扫描之间的变化不应超过字长的一半。

在上例中，如果反馈设备沿顺时针方向移动，当扫描“A”的读数为 0，扫描“B”的读数为 -2000 时，则实际位置变化相当于顺时针方向的 +2096 个计数。但由于这两个值大于字长的一半（大于物理设备旋转周期的一半），因此 PMUL 指令按照反馈设备反向旋转进行计算，返回值 -2000 而不是 +2096。

脉冲乘法器功能块的第三个属性是，它会保留每次扫描期间因 Multiplier/100,000 变换因子而产生的任何余数部分。每次执行该功能块后，上一次扫描得到的余数会加回到当前总值中，这样一来，所有计数

或“脉冲”最终都会计算到，因而系统中的任何数据都不会丢失。功能块的输出 Out 始终产生一个浮点数据类型的整数。

功能块



假定 Initial_Position = 0 且 Multiplier = 2500 => (25,000/100,000)

扫描	Position_Feedback	PMUL_02.Out	Total_Position
n	0	0	0
n + 1	1	0	0
n + 2	2	0	0
n + 3	3	0	0
n + 4	4	1	1
n + 5	5	0	1

结构化文本

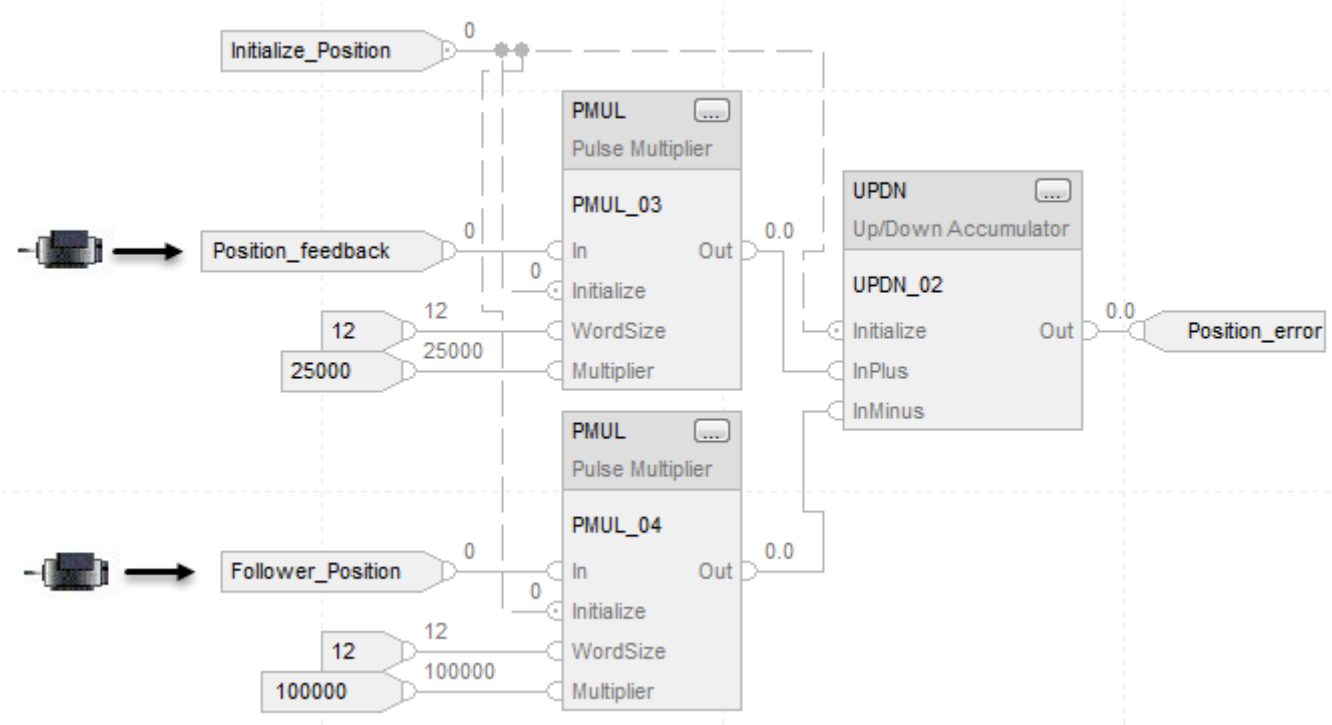
```
MUL_02.In := Position_feedback;
PMUL_02.Initalize := Initialize_Position;
PMUL_02.WordSize := 12;
PMUL_02.Multiplier := 25000;
PMUL(PMUL_02);

UPDN_02.Initialize := Initialize_Position;
UPDN_02.InPlus := PMUL_02.Out;
UPDN(UPDN_02);

Total_Position := UPDN_02.Out;
```

示例 2

在此电子传动轴应用中，电机 A 的反馈将作为电机 B 的主要参考。电机 A 的反馈又称“Position_feedback”。电机 B 的反馈又称“Follower_Position”。由于两条指令的乘数之比为 1/4，因此电机 A 每转

4 圈，电机 B 就需要旋转 1 圈，这样才能使 UPDN 累加器的累加值保持为零。UPDN 指令任何不为零的输出值都将视为 Position_error，可对该值进行调节并送回电机 B，以便在两个电机之间保持锁相。

功能块



结构化文本

```
PMUL_02.In := Position_feedback;
PMUL_02.Initalize := Initialize_Position;
PMUL_02.WordSize := 12;
PMUL_02.Multiplier := 25000;
PMUL(PMUL_02);

PMUL_03.In := Follower_Position;
PMUL_03.Initalize := Initialize_Position;
PMUL_03.WordSize := 12;
PMUL_03.Multiplier := 100000;
PMUL(PMUL_03);

UPDN_02.Initialize := Initialize_Position;
UPDN_02.InPlus := PMUL_02.Out;
UPDN_02.InMinus := PMUL_03.Out;
UPDN(UPDN_02);

Position_error := UPDN_02.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

S 曲线 (SCRV)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

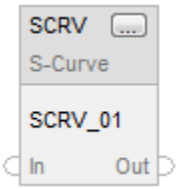
SCRV 指令执行具有附加急动度的斜坡函数。急动度是将输出经过斜坡处理为输入时使用的变化率的最大变化率。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

SCRV(SCRV_tag);

操作数

功能块

操作数	类型	格式	说明
SCRV tag	S_CURVE	结构	SCRV 结构

结构化文本

操作数	类型	格式	说明
SCRV tag	S_CURVE	结构	SCRV 结构

有关结构化文本中表达式语法的更多信息，请参见“结构化文本语法”部分。

S_CURVE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	指令的初始化输入。该参数置位时，指令保持 Out = InitialValue 默认清零。
InitialValue	REAL	S 曲线的初始值。当 Initialize 置位时，Out = InitialValue。 有效值 = 任意浮点值 默认值 = 0.0
AbsAlgRamp	BOOL	斜坡类型。置位时，指令将用作绝对值斜坡。清零时，指令将用作代数斜坡。 默认置位
AccelRate	REAL	加速度，单位为输入单位/秒^2。零值可阻止 Out 加速。当 AccelRate < 0 时，该指令会假定 AccelRate = 0，并将 Status 中的相应位置位。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
DecelRate	REAL	减速度，单位为输入单位/秒^2。零值可阻止 Out 减速。当 DecelRate < 0 时，该指令会假定 DecelRate = 0，并将 Status 中的相应位置位。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
JerkRate	REAL	减速度，单位为输入单位/秒^2。零值可阻止 Out 减速。当 DecelRate < 0 时，该指令会假定 DecelRate = 0，并将 Status 中的相应位置位。 有效值 = 0.0 到最大正浮点值 默认值 = 0.0
HoldMode	BOOL	S 曲线保持模式参数。该参数与 HoldEnable 参数一起使用。如果在 HoldEnable 置位且 Rate = 0 时 HoldMode 置位，指令将使 Out 保持不变。在这种情况下，只要 HoldEnable 置位指令就会使 Out 保持不变，JerkRate 会被忽略，而且 Out 曲线中将生成“拐点”。如果 HoldMode 在 HoldEnable 置位时清零，指令将使用 JerkRate 将 Out 变为常数值。当 Rate = 0 时，Out 保持不变。一旦 HoldEnable 置位，请勿更改 HoldMode，因为指令将忽略此更改。 默认清零。
HoldEnable	BOOL	S 曲线保持启用参数。置位时，Out 保持不变。清零时，Out 从当前值开始变化，直到等于 In。 默认清零。

TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	启用输出。
S_Mode	BOOL	S_Mode 输出。当 (Jerk * DeltaT) ≤ Rate 且 Rate < Accel 或 Decel 时，S_Mode 置位。否则，S_Mode 清零。
Out	REAL	S 曲线指令的输出。该输出的数学状态标志置位。
Rate	REAL	Out 的内部变化，以单位/秒表示。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
AcceRateInv (Status.1)	BOOL	AccelRate 为负数。
DecelRateInv (Status.2)	BOOL	DecelRate 为负数。
JerkRateInv (Status.3)	BOOL	JerkRate 为负数。
TimingModeInv (Status.27)	BOOL	时序模式无效。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS | DeltaT - RTSTime | > 1（0.001 秒）时置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。

RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaT (Status.31)	BOOL	DeltaT 值无效。

说明

SCRV 指令的主要要求是确保变化率的变化决不超过指定的急动度。

用户可以将 SCRV 指令配置为针对阶跃输入生成 S 曲线轨迹或斜坡曲线。

S 曲线轨迹

要生成 S 曲线轨迹，可对 JerkRate 进行设置，使 (JerkRate * DeltaT) < AccelRate 和/或 DecelRate。

在 S 曲线轨迹模式下，SCRV 指令可确保变化率的变化决不超过指定的 JerkRate。用于生成 S 曲线轨迹的算法旨在针对阶跃输入生成一条光滑且对称的 S 曲线。为简化此过程，算法中引入了 Out 的梯形积分。因此，曲线各个部分的 Rate 变化都将小于 JerkRate。

当输入发生阶跃变化时，变化率会增加到设定的 AccelRate 或 DecelRate。AccelRate 或 DecelRate 保持不变，直到为了在变化率到达零时使输出等于输入而必须降低变化率。

某些情况下，根据加速度、减速度和急动度值的不同，在变化率必须按急动度开始降低之前，可能还没有达到加速度或减速度。

对于非常小的阶跃变化，SCRV 指令不会尝试生成“S”曲线。在此模式下，将输出整个阶跃，并且 Rate 会反映输出的变化。如果 Out = In 并且 In 的下一个阶跃变化能够以小于或等于所设定 JerkRate 的变化率输出，则会出现这种情况。

SCRV 指令支持代数斜坡和绝对值斜坡。对于代数斜坡，加速条件由朝正向增大的输入定义，而减速条件则由朝负向减小的输入定义。对于绝对值斜坡，加速条件由逐渐远离零的输入定义，而减速条件则由逐渐接近零的输入定义。

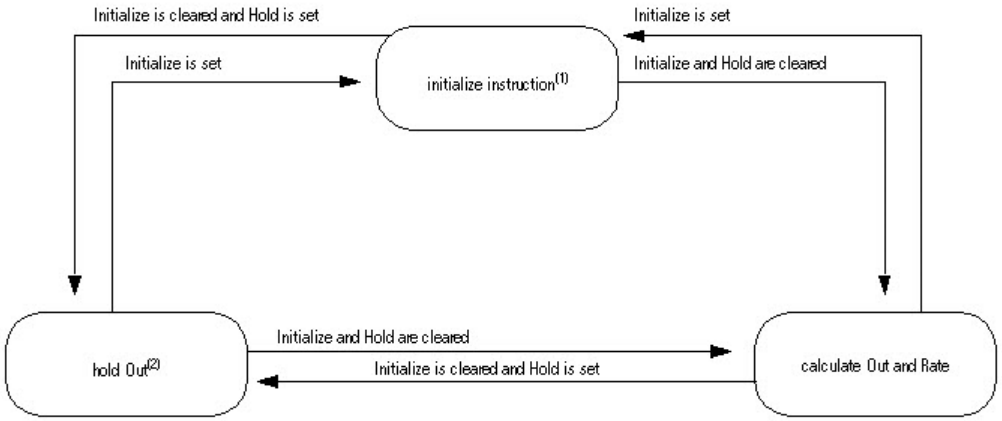
斜坡曲线

要生成斜坡曲线，可对 JerkRate 进行设置，使 (JerkRate * DeltaT) ≤ AccelRate 和/或 DecelRate。

在斜坡曲线模式下，SCRV 指令始终生成与设定的 AccelRate 或 DecelRate 相等的变化率，直到为了到达终点 Out 和 In 的差值需要小于 AccelRate 或 DecelRate。

HoldMode = 0 时的工作方式与 HoldMode = 1 相同。当 HoldEnable 置位时，Out 立即保持不变，Rate 将变为零。

下图展示指令修改 Out 的方式。



(1) 当 Initialize 置位时，指令将设置以下各项：

Outn = InitialValue

Outn-1 = Outn

Raten = 0

Raten-1 = 0

(2) 当 HoldMode 清零时，Out 朝 In 变化并且 HoldEnable 置位，变化率开始按急动度逐渐减小为零。由于 JerkRate 的作用，Out 会保持在变化率达到零时对应的值。当 Out 最终保持不变时，其值与 HoldEnable 置位时的瞬时值不同。

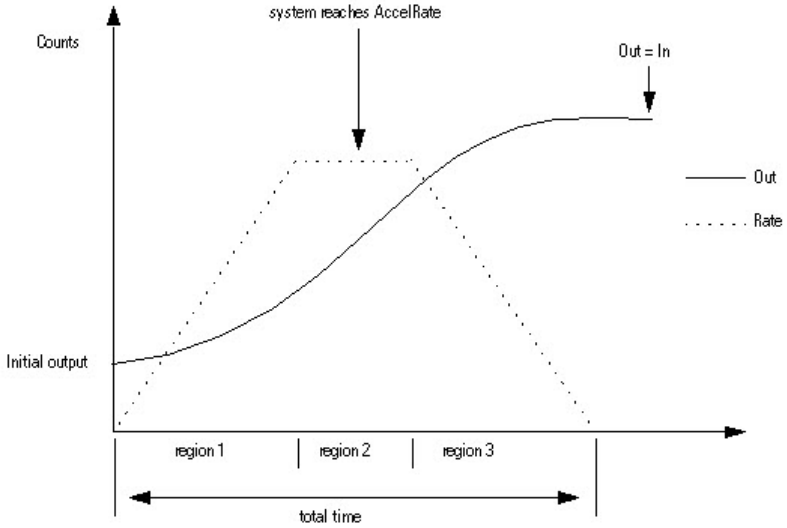
当 HoldMode 置位时，Out 朝 In 变化，HoldEnable 置位，变化率立即设置为 0。Out 保持在 HoldEnable 置位时对应的值。

在转换期间减小 JerkRate 可能会使 Out 超出 In。如果出现这种情况，是因强制使用输入的 JerkRate 引起的。若在调谐过程中以小步长减小 JerkRate，或者在 Out = In（非转换期间）时更改 JerkRate，可避免出现这种情况。

Out 达到与输入变化相等所需的时间与 AccelRate、JerkRate 以及 In 和 Out 之间差值相关。

计算输出值和变化率值

从初始值转换为最终值的过程中，Out 共经过三个区域。在区域 1 和区域 3 中，Out 的变化率取决于 JerkRate。在区域 2 中，Out 的变化率取决于 AccelRate 或 DecelRate。



各个区域的 Out 值计算如下：

$$TotalTime = \frac{FinalOutput - InitialOutput}{AccelRate} + \frac{AccelRate}{JerkRate}$$

各个区域对应的公式如下：

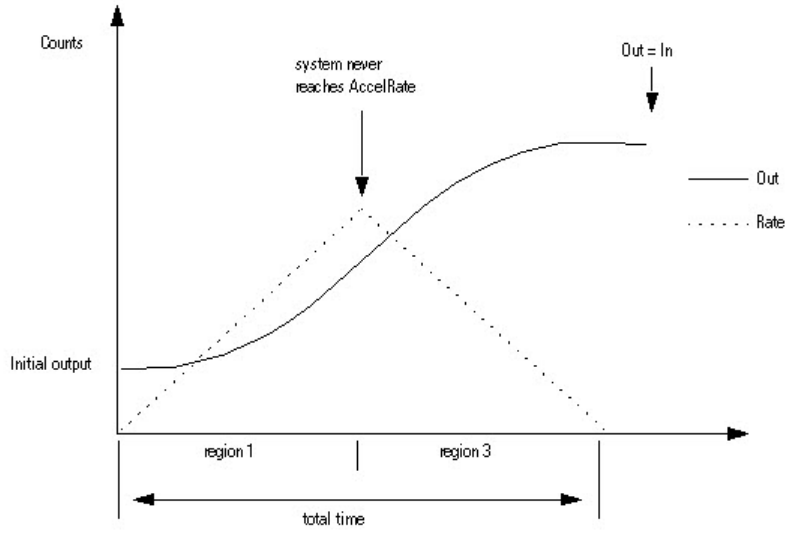
区域	公式
区域 1	$Time_1 = \frac{AccelRate}{JerkRate}$ $Y(Time) = InitialOutput + \frac{1}{2}(JerkRate) \times Time^2$
区域 2	$Time_2 = \frac{JerkRate \times (FinalOutput - InitialOutput) - AccelRate^2}{JerkRate \times AccelRate}$ $Y(Time) = InitialOutput + (AccelRate \times Time) - \frac{AccelRate^2}{2 \times JerkRate}$

区域 3	$Time_3 = \frac{AccelRate}{JerkRate}$ $Y(Time) = FinalOutput - \frac{1}{2}(JerkRate) \times \left(Time - \frac{FinalOutput - InitiaOutput}{AccelRate} - \frac{AccelRate}{JerkRate}\right)^2$

以下情况下：

$$|InitialOutput - FinalOutput| < \frac{AccelRate^2}{JerkRate}$$

SCRV 功能块未达到 AccelRate 或 DecelRate。Out 如下所示：



其中：

$$TotalTime = 2 \times \sqrt{\frac{|InitialOutput - FinalOutput|}{JerkRate}}$$

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件	功能块操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真 指令执行。
指令首次运行	不适用
指令首次扫描	清除之前的扫描数据。
后扫描	EnableIn 和 EnableOut 位设置为假

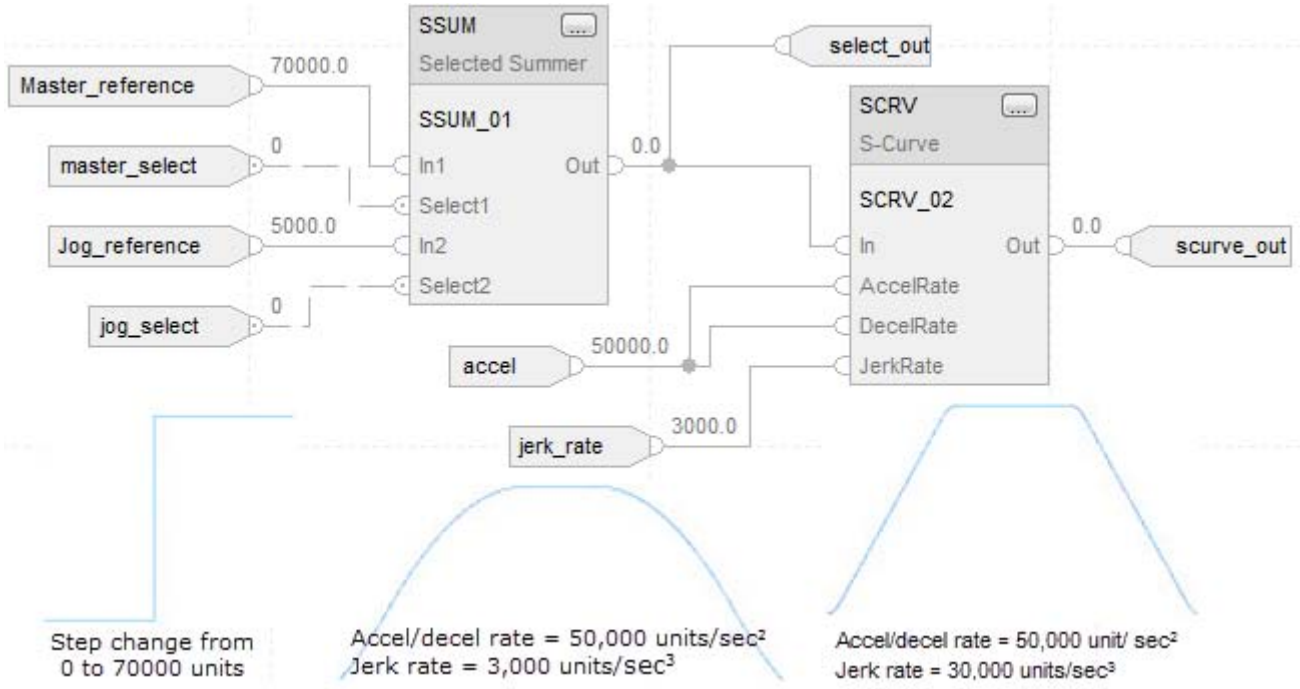
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

在大多数协调驱动应用中，整个驱动器组的线速度都由一个主基准值控制。选择不同基准值后，驱动器的速度基准值不能出现“阶跃”变化，因为负载惯量、电机转矩和调谐存在差异时不允许各个驱动器部分以协调一致的方式工作。SCRV 指令专用于对各驱动器部分的基准信号进行斜坡处理，从而使加速度、减速度和急动度（加速度的导数）得到控制。该指令提供了一种机制，可以使驱动器的基准值达到指定的基准值设置点，同时在此过程中可避免对所连接的机器和设备产生过度压力和影响。

功能块



结构化文本

```
SSUM_01.In1 := Master_reference;
SSUM_01.Select1 := master_select;
SSUM_01.In2 := Jog_reference;
SSUM_01.Select2 := jog_select;
SSUM(SSUM_01);

select_out := SSUM_01.Out;

SCRV_01.In := select_out;
SCRV_01.AccelRate := accel;
SCRV_01.DecelRate := accel;
SCRV_01.JerkRate := jerk_rate;
SCRV(SCRV_01);

scurve_out := SCRV_01.Out
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

二阶控制器 (SOC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

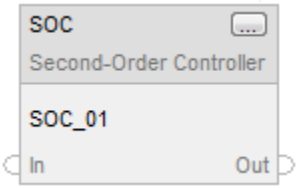
SOC 指令专用于闭环控制系统中，与 PI 指令的工作方式相似。SOC 指令具有增益项、一阶滞后和二阶超前。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

SOC(SOC_tag);

操作数

功能块

操作数	类型	格式	说明
SOC tag	SEC_ORDER_CONTROLLER	结构	SOC 结构

SEC_ORDER_CONTROLLER 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0

输入参数	数据类型	说明
Initialize	BOOL	指令初始化命令。置位时，Out 和内部积分器设为等于 InitialValue 的值。 默认清零。
InitialValue	REAL	输入初始值。Initialize 置位时，Out 和积分器设为 InitialValue 的值。InitialValue 值由 HighLimit 和 LowLimit 进行限制。 有效值 = 任意浮点值 默认值 = 0.0
Gain	REAL	指令的比例增益。如果值超出范围，则指令将限制该值，并将 Status 中的相应位置位。 有效值 = 任何 > 0.0 的浮点值 默认值 = 最小正浮点值
WLag	REAL	一阶滞后截止频率，单位为弧度/秒。如果值超出范围，指令会限制该值，并将 Status 中的相应位置位。 有效值 = 有效值范围请参见下文的“说明”部分 默认值 = 0.0
WLead	REAL	二阶超前截止频率，单位为弧度/秒。如果值超出范围，指令会限制该值，并将 Status 中的相应位置位。 有效值 = 有效值范围请参见下文的“说明”部分 默认值 = 0.0
ZetaLead	REAL	二阶超前阻尼系数。如果值超出范围，指令会限制该值，并将 Status 中的相应位置位。 有效值 = 0.0 至 10.0 默认值 = 0.0
HighLimit	REAL	上限值。这是 Out 的最大值。如果 HighLimit ≤ LowLimit，指令会将 HighAlarm 和 LowAlarm 置位，将 Status 中的相应位置位，并设置 Out = LowLimit。 有效值 = LowLimit < HighLimit ≤ 最大正浮点值 默认值 = 最大正浮点值
LowLimit	REAL	下限值。这是 Out 的最小值。如果 HighLimit ≤ LowLimit，指令会将 HighAlarm 和 LowAlarm 置位，将 Status 中的相应位置位，并设置 Out = LowLimit。 有效值 = 最大负浮点值 ≤ LowLimit < HighLimit 默认值 = 最大负浮点值

输入参数	数据类型	说明
HoldHigh	BOOL	保持高位命令。置位时，不允许内部积分器的值增大。 默认清零。
HoldLow	BOOL	保持低位命令。置位时，不允许内部积分器的值减小。 默认清零。
TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
HighAlarm	BOOL	上限报警指示器。当 Out 的计算值 ≥HighLimit 且输出强制设为 HighLimit 时置位。
LowAlarm	BOOL	下限报警指示器。当 Out 的计算值 ≤ LowLimit 且输出强制设为 LowLimit 时置位。

输出参数	数据类型	说明
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
状态 (Status)	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
GainInv (Status.1)	BOOL	增益 < 最小正浮点值。
WLagInv (Status.2)	BOOL	WLag > 最大值或 WLag < 最小值。
WLeadInv (Status.3)	BOOL	WLead > 最大值或 WLead < 最小值。
ZetaLeadInv (Status.4)	BOOL	ZetaLead > 最大值或 ZetaLead < 最小值。
HighLowLimsInv (Status.5)	BOOL	HighLimit ≤ LowLimit。
TimingModeInv (Status.27)	BOOL	时序模式无效。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS(DeltaT - RTSTime) > 1（0.001 秒）时置位。
RTSTimeInv (Satus.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaT (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
SOC tag	SEC_ORDER_CONTROLLER	结构	SOC 结构

有关结构化文本中表达式语法的更多信息，请参见"结构化文本语法"部分。

说明

SOC 指令具有增益项、一阶滞后和二阶超前。滞后的频率可以调整，并且超前的频率和阻尼也可以调整。二阶超前零点对可以是复数（阻尼小于 1）或实数（阻尼 ≥ 1）。SOC 指令专用于恒速扫描的任务中。

SOC 指令使用以下拉普拉斯变换方程。

$$H(s) = \frac{K\left(\frac{s^2}{\omega_{Lead}^2} + \frac{2 \times \xi_{Lead} \times s}{\omega_{Lead}} + 1\right)}{s\left(\frac{s}{\omega_{Lag}} + 1\right)}$$

参数限制

以下 SOC 参数对有效值具有以下限值。

参数	限制
WLead	$LowLimit = \frac{0.00001}{DeltaT}$ $HighLimit = \frac{0.07\pi}{DeltaT}$ 其中，DeltaT 以秒为单位。
WLag	$LowLimit = \frac{0.0000001}{DeltaT}$ $HighLimit = \frac{0.07\pi}{DeltaT}$ 其中，DeltaT 以秒为单位。
ZetaLead	LowLimit = 0.0 HighLimit = 10.0

一旦计算出的输出值无效 (NAN)，指令会将 Out 设为无效值。内部参数不会更新。在每次的后续扫描过程中，都会使用上一次扫描（输出有效）的内部参数计算输出。

限制

该指令基于 Hold 输入的状态停止积分饱和。

若：	则：
HoldHigh 置位且 Integrator > $Integrator_{n-1}$	Integrator = $Integrator_{n-1}$
HoldLow 置位且 Integrator < $Integrator_{n-1}$	Integrator = $Integrator_{n-1}$

指令还会根据 HighLimit 和 LowLimit 值停止积分器的积分饱和。

若：	则：
Integrator > IntegratorHighLimit	Integrator = IntegratorHighLimit
Integrator < IntegratorLowLimit	Integrator = IntegratorLowLimit

其中：

$$IntegratorHighLimit = HighLimit \times \frac{Gain \times WLag}{WLead^2}$$

$$IntegratorLowLimit = LowLimit \times \frac{Gain \times WLag}{WLead^2}$$

指令还会根据 HighLimit 和 LowLimit 值限制 Out 值。

若：	则：
HighLimit ≤ LowLimit	Out = LowLimit Integrator = IntegratorLowLimit HighLowLimsInv 置位 HighAlarm 置位 LowAlarm 置位
Out ≥ HighLimit	Out = HighLimit $IntegratorLowLimit_{n-1}$ HighAlarm 置位
Out ≤ LowLimit	Out = LowLimit Integrator = $Integrator_{n-1}$ LowAlarm 置位

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。

Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	内部参数和 Out 设置为 0。强制重新计算公式系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

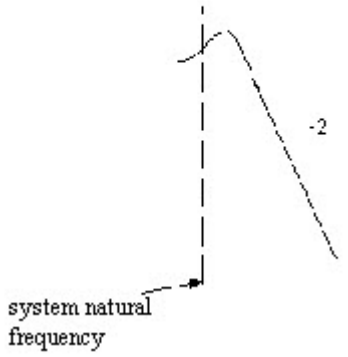
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

SOC 指令是一个专用的功能块，用于在两个区域之间通过弹簧质点系统传递能量的应用中。在这类应用中，过程本身的频率响应通常表示为如下所示的波德图（图 A）：

Diagram A: Process characteristics

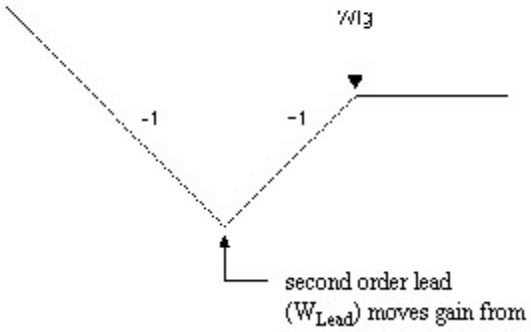


SOC 指令用于实现一阶滞后滤波器，后跟 PID 控制器，从而实现包含积分、二阶零点（超前）和一阶极点（滞后）的传递函数。使用该指令可简化 PID 调谐，因为调节项的组织方式可以确保 WLead 和 ZLead（而非 Kp、Ki 和 Kd 值）作为 SOC 指令的输入。SOC 指令的传递函数如下：

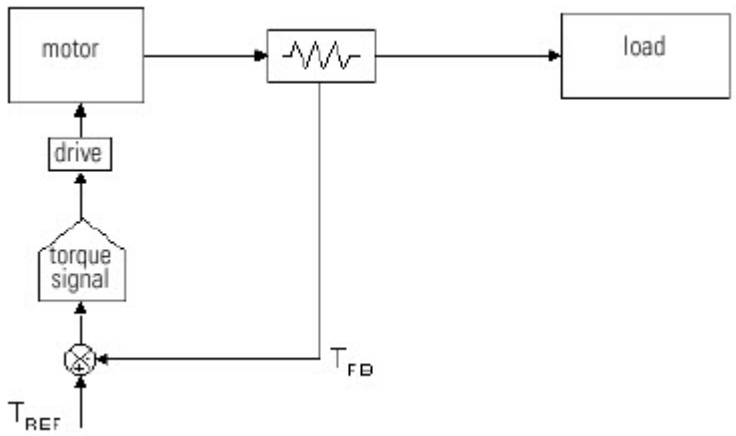
$$H(s) = \frac{K\left(\frac{s^2}{\omega_{Lead}^2} + \frac{2 \times \xi_{Lead} \times s}{\omega_{Lead}} + 1\right)}{s\left(\frac{s}{\omega_{Lag}} + 1\right)}$$

其相应的波特图如图 B 所示。

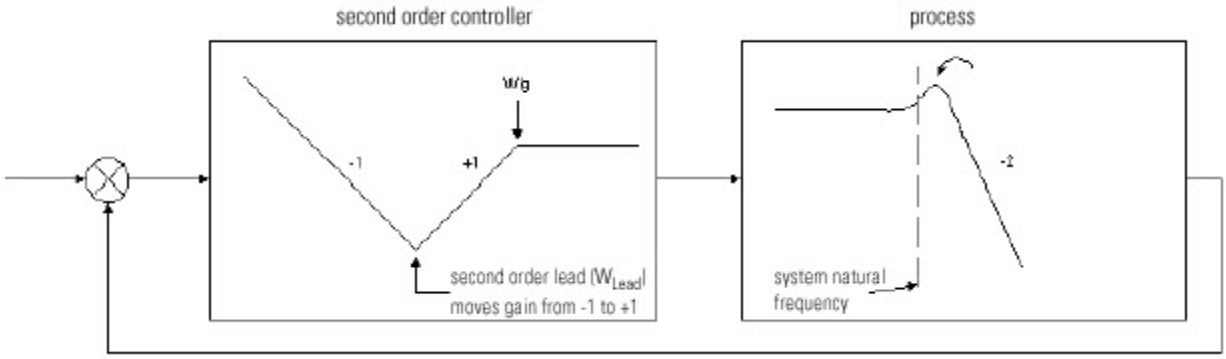
Diagram B: Second order controller



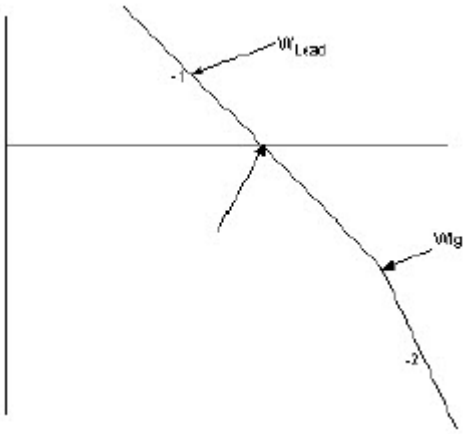
SOC 指令可用于扭矩或张力调节应用中，其中采用测力传感器作为反馈，调节系统的输出直接作用于驱动器的扭矩（电流）次回路。在诸多此类应用中，从机械角度而言受控系统都是欠阻尼的，并且具有一个固有频率，由于系统本身会通过反馈设备发生反射，因此难以稳定。



使用 SOC 指令可简化 PID 调节，因为调节项的组织方式可以确保 WLead 和 ZLead （而非 Kp、Ki 和 Kd 值）作为 SOC 指令的输入。通过这种方式，可以更轻松地根据实际过程调整和设置控制器/调节器的截止频率。启动期间，可根据经验预估或在现场测量系统的固有频率和阻尼系数。然后，可以根据过程的特征调整调节器的参数，从而对最终过程进行更多增益以及更加稳定的控制。



在上述系统中，如果将 Wlead 设置为与系统固有频率相等，将 Wlag 设置为远大于所需分频频率（> 5 倍的分频频率），则产生的系统响应显示如下：



在实际应用中，使用和设置此指令的步骤包括：

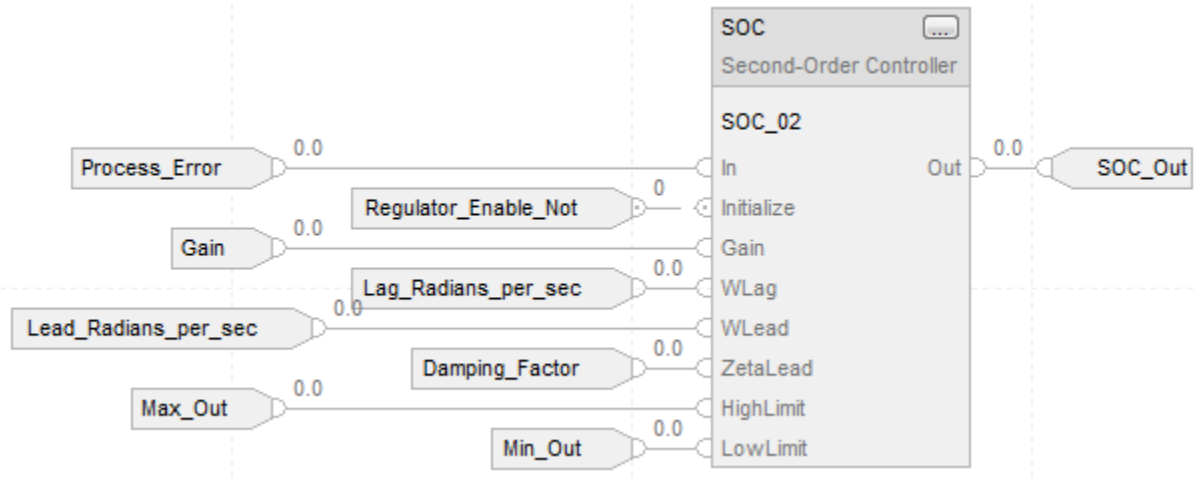
识别受控过程的类型。如果系统对阶跃函数的响应导致高度振铃，或者该响应的特征可以通过上述过程曲线来表示，则该块可以提供稳定控制所需的调节特征。

确定系统/过程的固有频率。既可以根据经验预估，也可以进行现场测量。调整 WLead，使其与过程本身的固有频率相对应或稍微超前。

调整阻尼系数 Zlead，使其消除系统中存在的任何超调。

将 WLag 调整为远大于系统分频频率（> 5 倍），开始增大总增益以达到

功能块



结构化文本

```
SOC_01.In := Process_Error;
SOC_01.Initialize := Regulator_Enable_Not;
SOC_01.Gain := Gain;
SOC_01.WLag := Lag_Radians_per_sec;
SOC_01.WLead := Lead_radians_per_sec;
SOC_01.ZetaLead := Damping_Factor;
SOC_01.HighLimit := Max_Out;
SOC_01.LowLimit := Min_Out;
SOC(SOC_01);

SOC_Out := SOC_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

增/减累加器 (UPDN)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

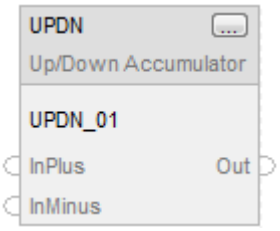
UPDN 指令用于将一个输入加到累加值中，将另一个输入从累加值中减掉。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

UPDN(UPDN_tag)

操作数

功能块

操作数	类型	格式	说明
UPDN tag	UP_DOWN_ACCUM	结构	UPDN 结构

UPDN 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
Initialize	BOOL	指令的初始化输入请求。当 Initialize 置位时，指令会将 Out 和内部累加器设置为 InitialValue。 默认清零。

InitialValue	REAL	指令的初始值。 有效值 = 任意浮点值 默认值 = 0.0
InPlus	REAL	加到累加器的输入。 有效值 = 任意浮点值 默认值 = 0.0
InMinus	REAL	从累加器减掉的输入。 有效值 = 任意浮点值 默认值 = 0.0
Hold	BOOL	指令的保持输入请求。当 Hold 置位并且 Initialize 清零时，Out 保持不变。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	指令的输出。

结构化文本

操作数	类型	格式	说明
UPDN tag	UP_DOWN_ACCUM	结构	UPDN 结构

有关结构化文本中表达式语法的更多信息，请参见“结构化文本语法”部分。

说明

UPDN 指令按以下算法执行。

条件	操作
Hold 清零且 Initialize 清零	$AccumValue_n = AccumValue_{n-1} + InPlus - InMinus$ $Out = AccumValue_n$
Hold 置位且 Initialize 清零	$AccumValue_n = AccumValue_{n-1}$ $Out = AccumValue_n$
Initialize 置位	$AccumValue_n = InitialValue$ $Out = AccumValue_n$

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

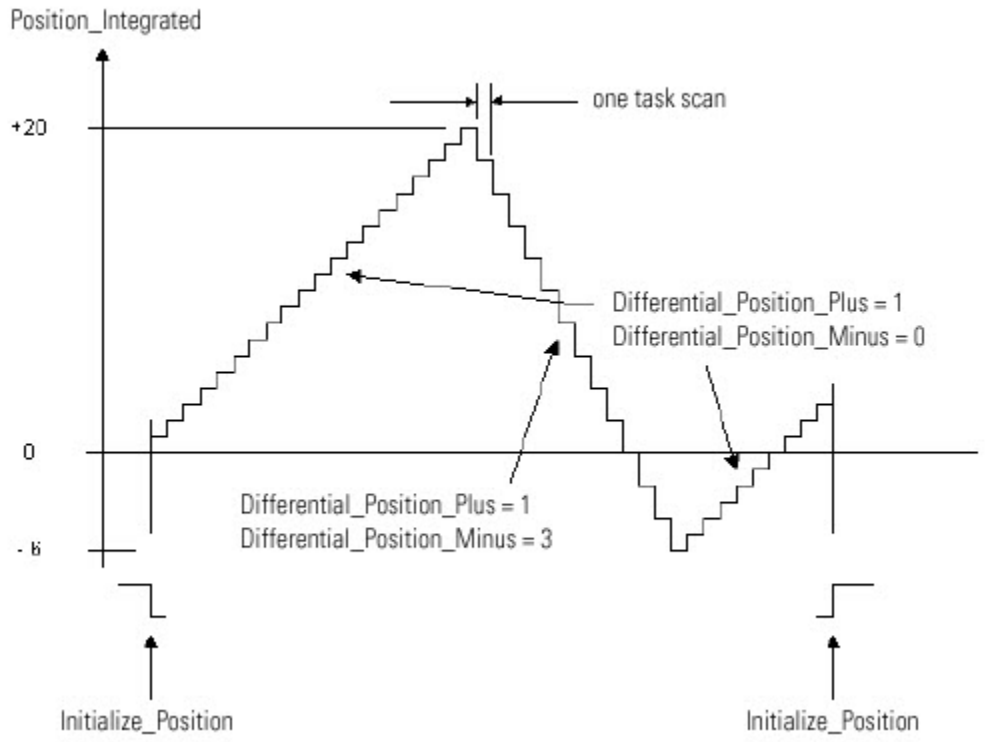
条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	将内部累加器设置为零。
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

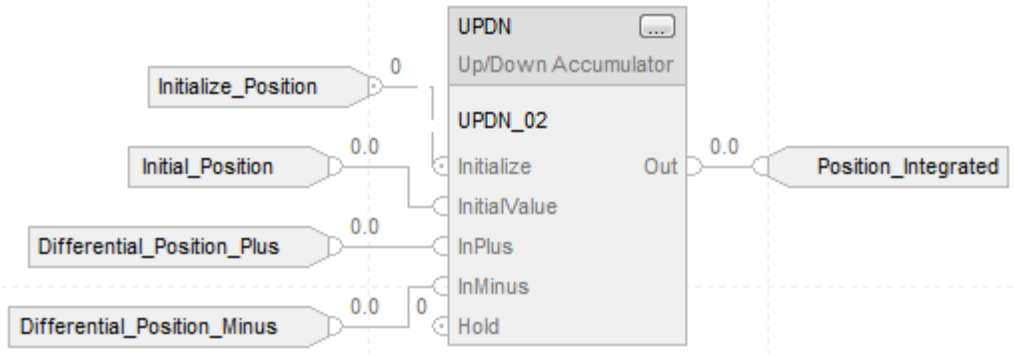
示例

UPDN 指令会整合两次扫描之间的计数。该指令既可用于简单的定位应用，也可用于需要简单积分来根据过程不同反馈信号生成累加值的其他应用类型。在以下示例中，Initial_Position 设置为零，而 Differential_Position_Plus 和 Differential_Position_Minus 的值在一段时间内不断变化。若使用此指令，InPlus 和 InMinus 还可接受负值。



功能块

微分指令用于计算信号随时间的变化量，以单位/秒表示。该指令通常用于闭环控制，从而在调节器中创建前馈路径，对具有大惯量的过程进行补偿。



结构化文本

```
UPDN_01.Initialize := Initialize_Position;
UPDN_01.InitialValue := Initial_Position;
UPDN_01.InPlus := Differential_Position_Plus;
UPDN_01.InMinus := Differential_Position_Minus;
UPDN(UPDN_01);

Position_Integrated := UPDN_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

HMI 按钮控件 (HMIBC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

将 HMI 按钮控件 (HMIBC) 指令与 PanelView 5500 人机界面 (HMI) 配合使用，操作员能够以高准确度和高确定性启动机器控制操作，例如使电机点动或启用阀门等。HMIBC 指令还具备内置的通信诊断功能，允许指令在与控制 HMI 的通信不可用时自动复位。

每个 Logix 控制器最多可支持 256 个 HMIBC 标签以及 32 个 PanelView 5500 HMI 同时通信和控制指令。当 PanelView 5500 HMI 设备启动与指令的实例标签相关联的按钮控件操作时，HMIBC 指令将变为激活状态并启用其输出。

重要事项： 要使用 HMIBC 指令，需要使用 PanelView 5500 模块。

若要正常运行，Logix 控制器 I/O 配置必须包括需要与 HMIBC 指令进行交互的所有 PanelView 5500 HMI。此外，为各 PanelView 5500 HMI 创建的应用程序必须包括一些按钮操作，要求这些按钮操作已配置为引用与 HMIBC 指令相关联的各个标签。

每次扫描至少执行一次此指令，不要跳过。

HMIBC 数据类型：

- 在控制器和程序作用域内可用。

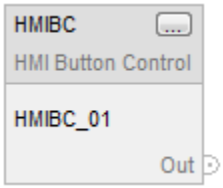
- 在 Add-On 自定义指令作用域内不可用。
- 用于跳转到子例程 (JSR) 指令。
- 不能与输入和输出程序参数一起使用
- 在安全程序中不可用。
- 必须具有外部访问值“读/写”。用户无法选择其他外部访问值。

HMIBC 标签具有 .L5K、.L5X 和 .CSV 的导入和导出格式。

可用语言

梯形图

功能块



提示： 对于 HMIBC 标签，仅使用 Out 参数，也可以使用功能块图中的 ProgFB 参数。

结构化文本

HMIBC (HMIBC tag)

操作数

以下操作数位于指令上。

操作数	类型	格式	说明
HMIBC tag	HMIBC	标签	当数据位置位时变为激活状态

HMIBC 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果此参数为假，则指令不会执行。
Prog FB	BOOL	程序反馈。此值不由指令处理，但会传输至所有注册的 HMI 设备。该值的用途或意义由用户定义。例如，此值可用于确定在按下相应按钮后预期操作是否真正执行以及是否在 HMI 设备上显示相应状态。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Button State	BOOL	在无任何已注册 HMI 设备按钮按下时设置为假。 在按下至少一个已注册 HMI 按钮时设置为真。 默认值为假。
Out	BOOL	当 EnableIn 为真时： 在无任何已注册 HMI 设备按钮按下时设置为假。 在按下至少一个已注册 HMI 按钮时设置为真。 当 EnableIn 为假时： 设置为假 默认值为假。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。请参见“数组索引编制”部分，了解关于数组索引故障的信息。

执行

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	如果按下与指令实例标签相关联的任何 HMI 设备按钮控件，梯级输出条件设置为真。否则，梯级输出条件设置为假。
后扫描	梯级输出条件设置为假。

功能块

条件/状态	执行的操作
预扫描	不适用
Tag.EnableIn 为假	指令不会执行。
Tag.EnableIn 为真	指令不会执行。
指令首次扫描	不适用
指令首次运行	不适用
后扫描	不适用

结构化文本

条件/状态	执行的操作
预扫描	指令执行。
正常执行	指令执行。
后扫描	指令执行。

示例

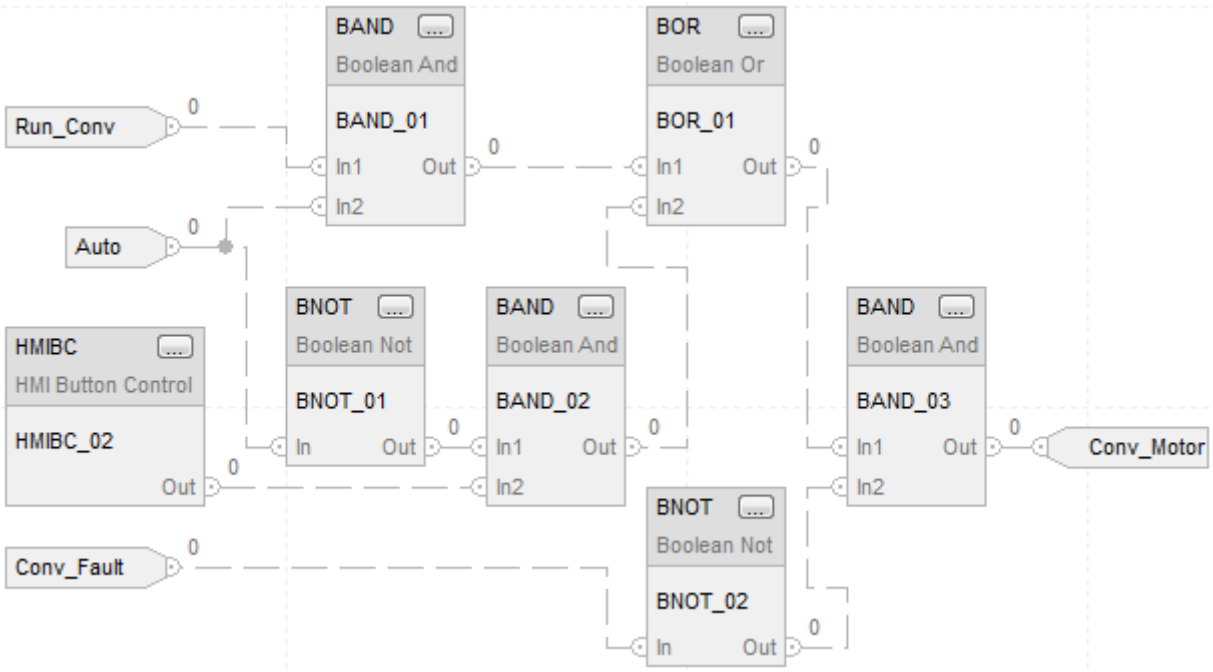
梯形图

- HMIBC 指令是一种输入指令，不能自行放置在梯级上。

- HMIBC 指令处于激活状态时会突出显示。

功能块

以下示例展示功能块图中的 HMIBC 指令。



结构化文本

HMIBC (HMIBC_Conv);

IF(((Auto AND Run_Conv) Or (NOT Auto AND HMIBC_Conv.Out)) AND NOT Conv_Fault)

THEN Conv_Motor: = 1;

ELSE Conv_Motor : = 0;

END_IF;

另请参见

数组索引编制 参考页数 602

滤波器

滤波指令

滤波指令包括以下指令：

可用指令

梯形图

此指令不可用于梯形图中

功能块和结构化文本

DERV	HPF	LDL2	LPF	NTCH

如果希望	使用此指令
计算信号随时间的变化量， 以单位/秒表示。	DERV
滤除低于截止频率的输入频率。	HPF
具有极点对和零点对的滤波器。	LDL2
滤除高于截止频率的输入频率。	LPF
滤除等于陷波频率的输入频率。	NTCH

另请参见

驱动指令 参考页数 309

逻辑和移动指令 参考页数 453

过程控制指令 参考页数 21

选择/限制指令 参考页数 395

统计指令 参考页数 431

微分 (DERV)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

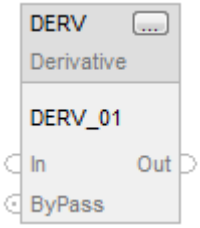
DERV 指令用于计算信号随时间的变化量，以单位/秒表示。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

DERV(DERV_tag);

操作数

功能块

操作数	类型	格式	说明
DERV tag	DERIVATIVE	结构	DERV 结构

DERIVATIVE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0

Gain	REAL	微分乘法器 有效值 = 任意浮点值 默认值 = 1.0
ByPass	BOOL	绕过算法请求。ByPass 为真时，指令会设置 Out = In。 默认值为真。
TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	**数据类型**	**说明**
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间(秒)。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。

TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见“功能块属性”部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS (DeltaT - RTSTime) > 1 毫秒时，设置为真。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
DERV tag	DERIVATIVE	结构	DERV 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

DERV 指令支持旁路输入，从而停止计算微分并将信号直接传递到输出。

Bypass 为以下条件时	指令使用此公式
清零且 DeltaT > 0	$Out = Gain\frac{In_n - In_{n-1}}{DeltaT}$ $In_{n-1} = In_n$ 其中，DeltaT 以秒为单位。
置位	$Out = In_n$ $In_{n-1} = In_n$

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。 结构化文本：不适用
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	重新计算系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

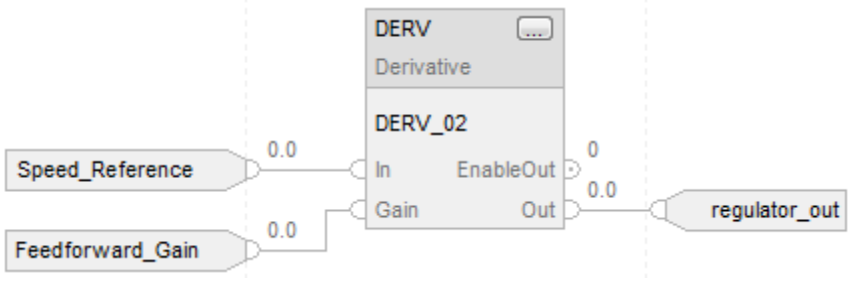
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

本示例是 DERV 功能块最基本的合法编程方法，仅用于展示该指令的纯文本内容和生成的代码。本示例仅供内部使用，并非可测试的用例。

功能块



结构化文本

```
DERV_01.In := Speed_Reference;
DERV_01.Gain := Feedforward_Gain;
DERV(DERV_01);

PI_01.In := Speed_Reference - Speed_feedback;
PI_01.Kp := Proportional_Gain;
PI_01.Wld := Integral_Gain;
PI(PI_01);

regulator_out := DERV_01.Out + PI_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

高通滤波器 (HPF)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

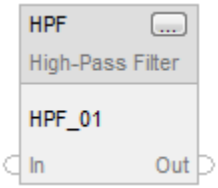
HPF 指令提供一个滤波器，用来对低于截止频率的输入频率进行衰减。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

```
HPF(HPF_tag);
```

操作数

功能块

操作数	类型	格式	说明
HPF tag	FILTER_HIGH_PASS	结构	HPF 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	初始化滤波控制算法的请求。该值为真时，指令会设置 Out = In。 默认值为假。
WLead	REAL	超前频率（弧度/秒）。如果 WLead < 最小值或 WLead > 最大值，指令会将 Status 中的相应位置位，并限制 WLead 值。 有效值 = 有效值范围请参见下文的"说明"部分。 默认值 = 0.0
Order	REAL	滤波器阶数。Order 值用于控制截止区的锐度。如果 Order 值无效，指令会将 Status 中的相应位置位，并使用 Order = 1。 有效值 = 1 到 3 默认值 = 1
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1

RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
WLeadInv (Status.1)	BOOL	WLead < 最小值或 WLead > 最大值。
OrderInv (Status.2)	BOOL	Order 值无效。
TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见“功能块属性”部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS (DeltaT - RTSTime) > 1 毫秒时， 设置为真。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
HPF tag	FILTER_HIGH_PASS	结构	HPF 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

HPF 指令使用 Order 参数控制截止区的锐度。HPF 指令专用于恒速扫描的任务中。

HPF 指令使用以下公式：

以下情况下：	指令使用此传递函数：
Order = 1	$\frac{s}{s+\omega}$
Order = 2	$\frac{s^2}{s^2+\sqrt{2}\times s\times \omega+\omega^2}$
Order = 3	$\frac{s^3}{s^3+(2\times s^2\times \omega)+2\times s\times \omega^2+\omega^3}$

各参数的限制如下（其中，DeltaT 单位为秒）：

参数	限制
一阶 WLead LowLimit	$\frac{0.0000001}{DeltaT}$
二阶 WLead LowLimit	$\frac{0.00001}{DeltaT}$
三阶 WLead LowLimit	$\frac{0.001}{DeltaT}$
HighLimit	$\frac{0.7\pi}{DeltaT}$

只要计算出的输出值无效（NAN 或 ± INF），指令就会将 Out 设为无效值。当计算出的输出值有效时，该指令将初始化内部参数并设置 Out = In。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。

Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	重新计算系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

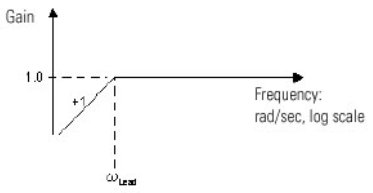
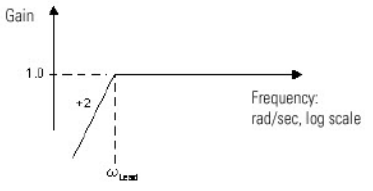
条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

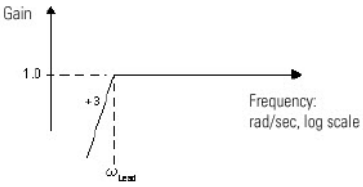
示例

HPF 指令将对低于所配置截止频率的信号进行衰减。该指令通常用于过滤来自电气或机械装置的低频"噪声"或干扰。用户可以选择特定的滤波器阶数，以实现不同程度的衰减。请注意，阶数越大，滤波器指令的执行时间越长。

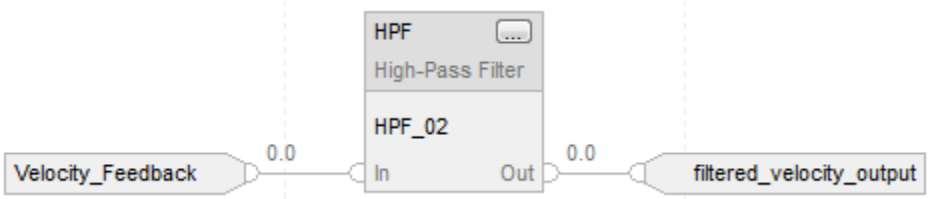
下图显示了不同滤波器阶数对给定截止频率的影响。每张图中的理想渐进逼近曲线均增益对数和频率对数为坐标。滤波器的实际响应逼近这些曲线，但并不完全贴合。

本示例是 HPF 功能块最基本的合法编程方法，仅用于展示该指令的纯文本内容和生成的代码。本示例仅供内部使用，并非可测试的用例。

滤波器	图形
一阶滤波器	
二阶滤波器	

三阶滤波器	

功能块



结构化文本

```
HPF_01.In := Velocity_Feedback;
HPF_01.WLead := Cutoff_frequency;
HPF_01.Order := 2;

HPF(HPF_01);

filtered_velocity_output := HPF_01.Out
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

低通滤波器 (LPF)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

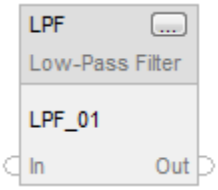
LPF 指令提供一个滤波器，用来对高于截止频率的输入频率进行衰减。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

LPF(LPF_tag);

操作数

功能块

操作数	类型	格式	说明
LPF tag	FILTER_LOW_PASS	结构	LPF 结构

FILTER_LOW_PASS 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	初始化滤波控制算法的请求。该值为真时，指令会设置 Out = In。 默认值为假。
WLag	REAL	滞后频率（弧度/秒）。如果 WLag < 最小值或 WLag > 最大值，指令会将 Status 中的相应位置位，并限制 WLag 值。 有效值 = 有效值范围请参见下文的"说明"部分 默认值 = 0.0

Order	REAL	滤波器阶数。Order 值用于控制截止区的锐度。如果 Order 值无效，指令会将 Status 中的相应位置位，并使用 Order = 1。 有效值 = 1 到 3 默认值 = 1
TimingMode	DINT	选择时序执行模式。 0 = 周期模式 1 = 过采样模式 2 = 实时采样模式 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否已使能，Out 溢出时设置为假。
Out	REAL	计算所得的算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
WLagInv (Status.1)	BOOL	WLag < 最小值或 WLag > 最大值。
OrderInv (Status.2)	BOOL	Order 值无效。

TimingModeInv (Status.27)	BOOL	TimingMode 值无效。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS(DeltaT – RTSTime) > 1 毫秒时，设置为真。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
LPF tag	FILTER_LOW_PASS	结构	LPF 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

LPF 指令使用 Order 参数控制截止区的锐度。LPF 指令专用于恒速扫描的任务中。

LPF 指令使用以下公式：

以下情况下：	指令使用此拉普拉斯传递函数：
Order = 1	$\frac{\omega}{s+\omega}$
Order = 2	$\frac{\omega^2}{s^2+\sqrt{2}\times s\times\omega+\omega^2}$
Order = 3	$\frac{\omega^3}{s^3+(2\times s^2\times\omega)+(2\times s\times\omega^2)\times\omega^3}$

各参数的限制如下（其中，DeltaT 单位为秒）：

参数	限制
一阶 WLag LowLimit	$\frac{0.0000001}{DeltaT}$
二阶 WLag LowLimit	$\frac{0.00001}{DeltaT}$
三阶 WLag LowLimit	$\frac{0.001}{DeltaT}$

HighLimit	$\frac{0.7\pi}{DeltaT}$

只要计算出的输出值无效（NAN 或 ± INF），指令就会将 Out 设为无效值。当计算出的输出值有效时，该指令将初始化内部参数并设置 Out = In。

影响数学状态标志

控制器	影响数学状态标志
ControlLogix 5580	否
CompactLogix 5370、 ControlLogix 5570	对输出而言影响

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	重新计算系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

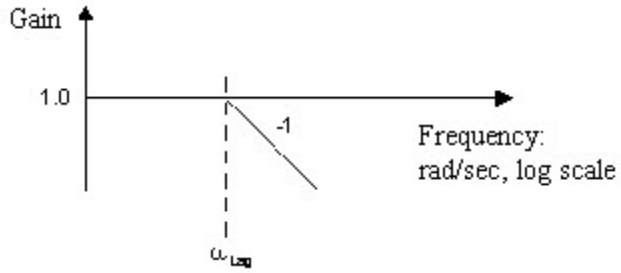
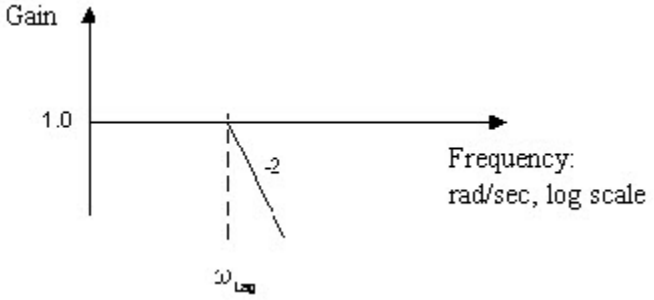
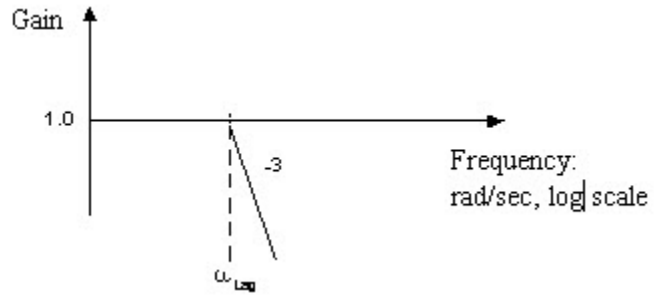
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

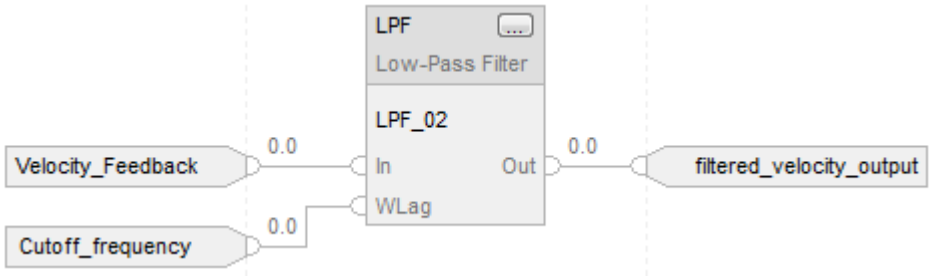
LPF 指令将对高于所配置截止频率的信号进行衰减。该指令通常用于过滤来自电气或机械装置的高频“噪声”或干扰。用户可以选择特定的滤波器阶数，以实现不同程度的衰减。请注意，阶数越大，指令的执行时间越长。

下图显示了不同滤波器阶数对给定截止频率的影响。每张图中的理想渐进逼近曲线均增益对数和频率对数为坐标。滤波器的实际响应逼近这些曲线，但并不完全贴合。

本示例是 LPF 功能块最基本的合法编程方法，仅用于展示该指令的纯文本内容和生成的代码。本示例仅供内部使用，并非可测试的用例。

滤波器	图形
一阶滤波器	
二阶滤波器	
三阶滤波器	

功能块



结构化文本

```
LPF_01.In := Velocity_Feedback;
LPF_01.WLag := Cutoff_frequency;

LPF(LPF_01);

filtered_velocity_output := LPF_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

陷波滤波器 (NTCH)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

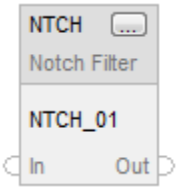
NTCH 指令提供一个滤波器，用来对等于险波频率的输入频率进行衰减。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

NTCH(NTCH_tag);

操作数

功能块

操作数	类型	格式	说明
NTCH tag	FILTER_NOTCH	结构	NTCH 结构

FILTER_NOTCH 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	初始化滤波控制算法的请求。该值为真时，指令会设置 Out = In。 默认值为假。
WNotch	REAL	滤波器中心频率（弧度/秒）。如果 WNotch < 最小值或 WNotch > 最大值，指令会将 Status 中的相应位置位，并限制 WNotch。 有效值 = 有效值范围请参见下文的“说明”部分。 默认值 = 最大正浮点值

QFactor	REAL	控制宽深比。设置 QFactor = 1 / (2*理想阻尼系数)。如果 QFactor < 最小值或 QFactor > 最大值，指令会将 Status 中的相应位置位，并限制 QFactor。 有效值 = 0.5 到 100.0 默认值 = 0.5
Order	REAL	滤波器阶数。Order 值用于控制截止区的锐度。如果 Order 值无效，指令会将 Status 中的相应位置位，并使用 Order = 2。 有效值 = 2 或 4 默认值 = 2
TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 2 有关时序模式的详细信息，请参见"功能块属性"部分。 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
WNotchInv (Status.1)	BOOL	WNotch < 最小值或 WNotch > 最大值
QFactorInv (Status.2)	BOOL	QFactor < 最小值或 QFactor > 最大值
OrderInv (Status.3)	BOOL	Order 值无效。
TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS (DeltaT - RTSTime) > 1 毫秒时， 设置为真。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
NTCH tag	FILTER_NOTCH	结构	NTCH 结构

有关结构化文本中表达式语法的详细信息，请参见"结构化文本语法"部分。

说明

NTCH 指令使用 Order 参数控制截止区的锐度。QFactor 参数可控制陷波区的宽深比。NTCH 指令专用于恒速扫描的任务中。

NTCH 指令使用以下公式：

$$\frac{\left(s^2+\omega^2\right)^i}{\left(s^2+s\times\frac{\omega}{Q}+\omega^2\right)^i}$$

其中，i 为 Order 参数，各参数的限制如下（其中，DeltaT 单位为秒）：

参数	限制
二阶 WNotch LowLimit	$\frac{0.0000001}{DeltaT}$
四阶 WNotch LowLimit	$\frac{0.001}{DeltaT}$
HighLimit	$\frac{0.7\pi}{DeltaT}$
QFactor	LowLimit = 0.5 HighLimit = 100.0

只要计算出的输出值无效（NAN 或 ± INF），指令就会将 Out 设为无效值。当计算出的输出值有效时，该指令将初始化内部参数并设置 Out = In。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。

Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	重新计算系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

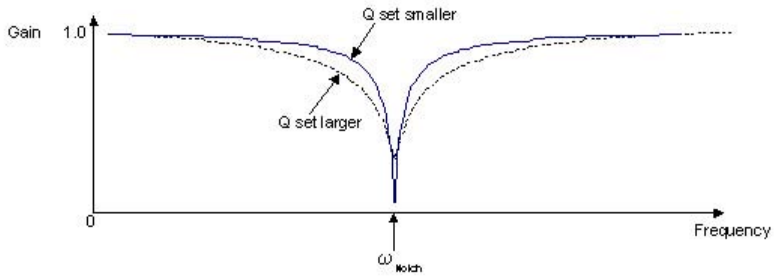
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

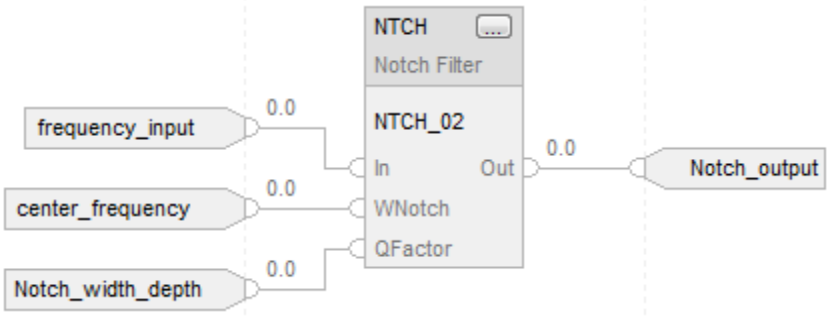
NTCH 指令用于衰减特定的共振频率。通常情况下，这些共振频率直接处于由闭环控制系统调节的响应范围内。共振频率通常由于机械联动装置松动而产生，会造成系统中出现回差和振动。尽管最佳的解决方法是校正机械装置的机械柔性，但也可使用陷波滤波器来减弱这些信号在闭环调节方案中的影响。

下图显示了特定中心频率和 Q 系数在某一频率范围内的理想增益曲线。Q 系数越大，陷波区越宽越浅。Q 系数越小，陷波区越窄越深。指令可设置为二阶或四阶。阶数越高，执行时间越长。

本示例是 NTCH 功能块最基本的合法编程方法，仅用于展示该指令的纯文本内容和生成的代码。本示例仅供内部使用，并非可测试的用例。



功能块



结构化文本

```
NTCH_01.In := frequency_input;
NTCH_01.WNotch := center_frequency;
NTCH_01.QFactor := Notch_width_depth;
NTCH(NTCH_01);

Notch_output := NTCH_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

二阶超前滞后 (LDL2)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

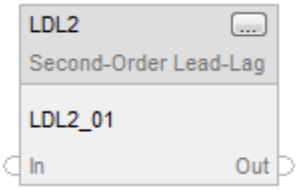
LDL2 指令提供具有极点对和零点对的滤波器。极点和零点对的频率和阻尼值均可调。极点或零点对可以是复数（阻尼系数小于 1）或实数（阻尼系数大于或等于 1）。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

LDL2(LDL2_tag);

操作数

功能块

操作数	类型	格式	说明
LDL2 tag	LEAD_LAG_SEC_ORDER	结构	LDL2 结构

LEAD_LAG_SEC_ORDER 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Initialize	BOOL	初始化滤波控制算法的请求。该值为真时，指令会设置 Out = In。 默认清零。
WLead	REAL	超前拐角频率（弧度/秒）。如果 WLead < 最小值或 WLead > 最大值，指令会将 Status 中的相应位置位，并限制 WLead 值。如果 WLag:WLead 之比 > 最大比率，指令会将 Status 中的相应位置位，并限制 WLag 值 有效值 = 有效值范围请参见下文的“说明”部分。 默认值 = 0.0

WLag	REAL	滞后拐角频率 (弧度/秒) 。如果 WLag < 最小值或 WLag > 最大值，指令会将 Status 中的相应位置位，并限制 WLag 值。 如果 WLag:WLead 之比 > 最大比率，指令会将 Status 中的相应位置位，并限制 WLag 值。 有效值 = 有效值范围请参见下文的"说明"部分 默认值 = 0.0
ZetaLead	REAL	二阶超前阻尼系数。仅在 Order = 2 时使用。如果 ZetaLead < 最小值或 ZetaLead > 最大值，指令会将 Status 中的相应位置位，并限制 ZetaLead 值。 有效值 = 0.0 到 4.0 默认值 = 0.0
ZetaLag	REAL	二阶滞后阻尼系数。仅在 Order = 2 时使用。如果 ZetaLag < 最小值或 ZetaLag > 最大值，指令会将 Status 中的相应位置位，并限制 ZetaLag 值。 有效值 = 0.05 到 4.0 默认值 = 0.05
Order	REAL	滤波器阶数。选择一阶或二阶滤波器算法。如果该值无效，指令会将 Status 中的相应位置位，并使用 Order = 2。 有效值 = 1 到 2 默认值 = 2
TimingMode	DINT	选择时序执行模式。 0 = 周期性模式 1 = 过采样模式 2 = 实时采样模式 有效值 = 0 到 2 默认值 = 0
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
WLeadInv (Status.1)	BOOL	WLead < 最小值或 WLead > 最大值。
WLagInv (Status.2)	BOOL	WLag < 最小值或 WLag > 最大值。
ZetaLeadInv (Status.3)	BOOL	超前阻尼系数 < 最小值，或超前阻尼系数 > 最大值。
ZetaLagInv (Status.4)	BOOL	滞后阻尼系数 < 最小值，或滞后阻尼系数 > 最大值。
OrderInv (Status.5)	BOOL	Order 值无效。
WLagRatioInv (Status.6)	BOOL	WLag:WLead 之比大于最大值。
TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。 ABS (DeltaT - RTSTime) > 1 毫秒时 置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。

DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

结构化文本

操作数	类型	格式	说明
LDL2 tag	LEAD_LAG_SEC_ORDER	结构	LDL2 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

LDL2 指令滤波器用在参考强制和反馈强制控制方法中。LDL2 指令专用于恒速扫描的任务中。

LDL2 指令使用以下公式：

以下情况下：	指令使用此拉普拉斯传递函数：
Order = 1	$H(s) = \frac{\frac{s}{\omega_{Lead}} + 1}{\frac{s}{\omega_{Lag}} + 1}$
Order = 2	$H(s) = \frac{\frac{s^2}{\omega_{Lead}^2} + \frac{2 \times \xi_{Lead} \times s}{\omega_{Lead}} + 1}{\frac{s^2}{\omega_{Lag}^2} + \frac{2 \times \xi_{Lag} \times s}{\omega_{Lag}} + 1}$ Normalize the filter such that $\omega_{Lead} = 1$ $H(s) = \frac{s^2 + 2 \times \xi_{Lead} \times s + 1}{\frac{s^2}{\omega_{Lag}^2} + \frac{2 \times \xi_{Lag} \times s}{\omega_{Lag}} + 1}$

各参数的限制如下（其中，DeltaT 单位为秒）：

参数	限制
一阶 WLead LowLimit	$\frac{0.0000001}{DeltaT}$
二阶 WLead LowLimit	$\frac{0.00001}{DeltaT}$
HighLimit	$\frac{0.7\pi}{DeltaT}$

WLead:WLag 之比	如果 WLead > WLag，无限制 如果 WLag > WLead： • WLag:WLead 无最小值限制 • WLag:WLead 的一阶最大值 = 40:1，指令会限制 WLag，以强制达到该比率 • WLag:WLead 的二阶最大值 = 10:1，指令会限制 WLag，以强制达到该比率
仅限二阶 ZetaLead	LowLimit = 0.0 HighLimit = 4.0
仅限二阶 ZetaLag	LowLimit = 0.05 HighLimit = 4.0

只要计算出的输出值无效（NAN 或 ± INF），指令就会将 Out 设为无效值。当计算出的输出值有效时，该指令将初始化内部参数并设置 Out = In。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	重新计算系数。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

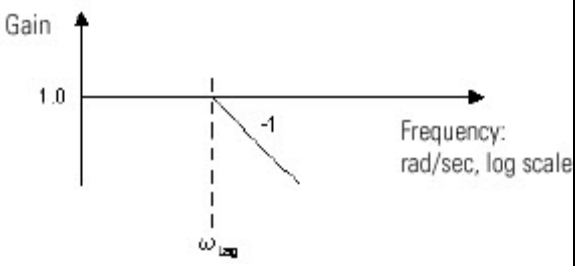
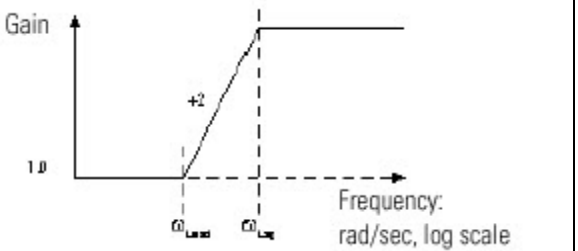
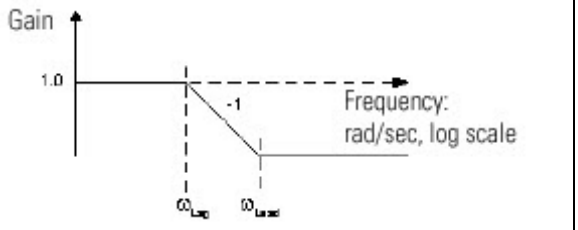
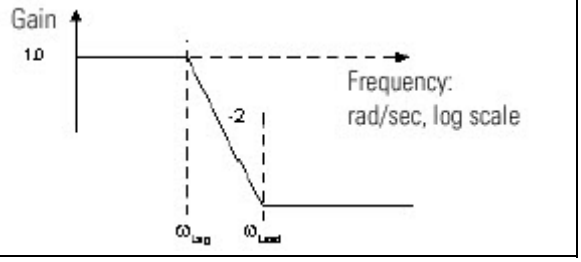
条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。

正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

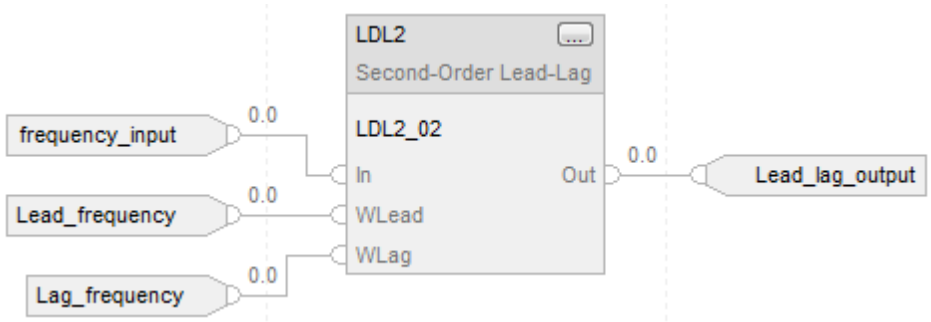
示例

LDL2 指令可以在两个频率之间衰减或放大，具体视指令配置方式而定。设置时，超前频率可以大于滞后频率，也可以小于滞后频率，因此指令可以作为超前-滞后功能块或滞后-超前功能块，具体取决于先配置哪个频率。请注意，阶数越大，滤波器指令的执行时间越长。

本示例是 LDL2 功能块最基本的合法编程方法，仅用于展示该指令的纯文本内容和生成的代码。本示例仅供内部使用，并非可测试的用例。

滤波器	图形
一阶超前-滞后 (wLead < wLag)	
二阶超前-滞后 (wLead < wLag)	
一阶超前-滞后 (wLag < wLead)	
二阶超前-滞后 (wLag < wLead)	

功能块



结构化文本

```
LDL2_01.In := frequency_input;
LDL2_01.WLead :=
Lead_frequency;
LDL2_01.WLag := Lag_frequency;
LDL2(LDL2_01);
Lead_lag_output := LDL2_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

选择_限制指令

选择/限制指令

选择/限制指令包括以下指令：

可用指令

梯形图

此指令不可用于梯形图中。

功能块和结构化文本

ESEL	HLL	MUX	RLIM	SEL	SNEG	SSUM

如果希望	使用此指令
从六路输入中任选其一。	ESEL
将模拟量输入限制在两个值之间。	HLL
选择八路输入其中之一。	MUX
限制信号随时间的变化量。	RLIM
选择两路输入其中之一。	SEL
在输入值与输入取反值之间进行选择。	SNEG
选择要进行求和的实数输入。	SSUM

另请参见

滤波指令 参考页数 365

逻辑和移动指令 参考页数 453

过程控制指令 参考页数 21

驱动指令 参考页数 309

统计指令 参考页数 431

增强型选择 (ESEL)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

ESEL 指令用于在多达六路输入中选择一路输入。选项包括：

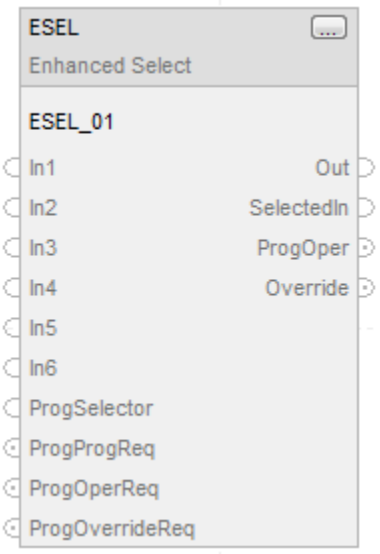
- 手动选择（由操作员或程序选择）
- 最大值选择
- 最小值选择
- 中位数选择
- 平均值（平均数）选择

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

ESEL(ESEL_tag);

操作数

功能块

操作数	类型	格式	说明
ESEL tag	SELECT_ENHANCED	结构	ESEL 结构

SELECT_ENHANCED 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In1	REAL	指令的第一个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In2	REAL	指令的第二个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In3	REAL	指令的第三个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In4	REAL	指令的第四个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In5	REAL	指令的第五个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In6	REAL	指令的第六个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In1Fault	BOOL	In1 不良状况指示器。如果 In1 从模拟量输入读取，则 In1Fault 通常由模拟量输入的故障状态控制。如果所有 InnFault 输入均为真，指令会将 Status 中的相应位置位，不执行控制算法，并且不会更新 Out。 默认值 = 假

In2Fault	BOOL	In2 不良状况指示器。如果 In2 从模拟量输入读取，则 In2Fault 通常由模拟量输入的故障状态控制。如果所有 InnFault 输入均为真，指令会将 Status 中的相应位置位，不执行控制算法，并且不会更新 Out。 默认值 = 假
In3Fault	BOOL	In3 不良状况指示器。如果 In3 从模拟量输入读取，则 In3Fault 通常由模拟量输入的故障状态控制。如果所有 InnFault 输入均为真，指令会将 Status 中的相应位置位，不执行控制算法，并且不会更新 Out。 默认值 = 假
In4Fault	BOOL	In4 不良状况指示器。如果 In4 从模拟量输入读取，则 In4Fault 通常由模拟量输入的故障状态控制。如果所有 InnFault 输入均为真，指令会将 Status 中的相应位置位，不执行控制算法，并且不会更新 Out。 默认值 = 假
In5Fault	BOOL	In5 不良状况指示器。如果 In5 从模拟量输入读取，则 In5Fault 通常由模拟量输入的故障状态控制。如果所有 InnFault 输入均为真，指令会将 Status 中的相应位置位，不执行控制算法，并且不会更新 Out。 默认值 = 假
In6Fault	BOOL	In6 不良状况指示器。如果 In6 从模拟量输入读取，则 In6Fault 通常由模拟量输入的故障状态控制。如果所有 InnFault 输入均为真，指令会将 Status 中的相应位置位，不执行控制算法，并且不会更新 Out。 默认值 = 假

InsUsed	DINT	使用的输入数。此参数定义指令使用的输入数。在最大值选择、最小值选择、中位数选择和平均值选择模式下，指令仅考虑 In1 到 $In_{InsUsed}$。如果该值无效，指令会将 Status 中的相应位置位。如果 InsUsed 无效、指令未处于手动选择模式并且 Override 清零，指令不会更新 Out。 有效值 =1 到 6 默认值 =1
Selector Mode	DINT	选择器模式输入。该值将决定指令的操作。 0 = 手动选择 1 = 最大值选择 2 = 最小值选择 3 = 中位数选择 4 = 平均值选择 如果该值无效，指令会将 Status 中的相应位置位，并且不更新 Out。 有效值 =0 到 4 默认值 =0
ProgSelector	DINT	程序模式下的选择器输入。选择器模式为手动选择并且指令处于程序控制模式时，由 ProgSelector 决定将哪路输入 (In1-In6) 送入 Out。如果 ProgSelector = 0，指令将不更新 Out。如果 ProgSelector 无效，指令会将 Status 中的相应位置位。如果该参数无效且指令处于程序控制模式，并且选择模式为手动选择或者 Override 置位，则指令不会更新 Out。 有效值 =0 到 6 默认值 =0

OperSelector	DINT	操作员模式下的选择器输入。选择器模式为手动选择并且指令处于操作员控制模式时，由 OperSelector 决定将哪路输入 (In1-In6) 送入 Out。如果 OperSelector = 0，指令将不更新 Out。如果 OperSelector 无效，指令会将 Status 中的相应位置位。如果该参数无效且指令处于操作员控制模式 ,并且选择模式为手动选择或者 Override 置位，则指令不会更新 Out。 有效值 = 0 到 6 默认值 = 0
ProgProgReq	BOOL	程序发出的程序控制请求。由用户程序设置为真可请求程序控制模式。如果 ProgOperReq 为真，则忽略该值。保持该参数为真且 ProgOperReq 为假可将指令锁定为程序控制模式。 默认值为假。
ProgOperReq	BOOL	程序发出的操作员控制请求。由用户程序设置为真可请求操作员控制模式。保持该参数为真可将指令锁定为操作员控制模式。 默认值为假。
ProgOverrideReq	BOOL	程序超控模式请求。由用户程序设置为真可请求进入超控模式。在超控模式下 ,指令将起到手动选择的作用。 默认值为假。
OperProgReq	BOOL	操作员发出的程序控制请求。由操作员界面设置为真以请求程序控制模式。 指令会将此输入设置为假。 默认值为假。
OperOperReq	BOOL	操作员发出的操作员控制请求。由操作员界面设置为真可请求操作员控制模式。指令会将此输入设置为假。 默认值为假。

ProgValueReset	BOOL	将程序控制值复位。该值为真时，每次执行指令时 ,所有程序请求输入都将设置为假。 默认值为假。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
SelectedIn	DINT	所选输入的编号。指令使用该值指示当前送入输出的是哪路输入。如果选择器模式为平均值选择，指令会设置 SelectedIn = 0。
ProgOper	BOOL	程序/操作员控制指示器。在程序控制模式下设置为真。在操作员控制模式下设置为假 为假。
Override	BOOL	超控模式。指令处于超控模式时设置为真。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InsFaulted (Status.1)	BOOL	所有使用的 Inn 输入的 InnFault 输入均为真。
InsUsedInv (Status.2)	BOOL	InsUsed 值无效。
SelectorModeInv (Status.3)	BOOL	SelectorMode 值无效。
ProgSelectorInv (Status.4)	BOOL	ProgSelector 值无效。
OperSelectorInv (Status.5)	BOOL	OperSelector 值无效。

结构化文本

操作数	类型	格式	说明
ESEL tag	SELECT_ENHANCED	结构	ESEL 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

ESEL 指令的工作方式如下：

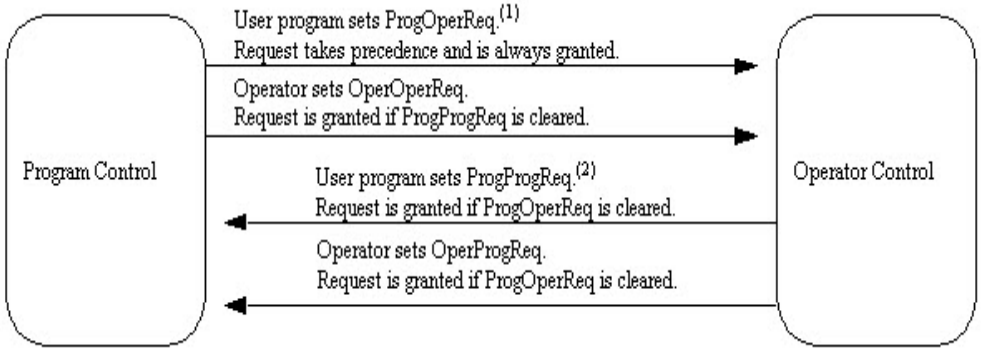
条件	操作
SelectorMode = 0（手动选择）或 Override 为真、ProgOper 为假，且 OperSelector 不等于 0	Out = In[OperSelector] SelectedIn = OperSelector
SelectorMode = 0（手动选择）或 Override 为真、ProgOper 为真，且 ProgSelector 不等于 0	Out = In[ProgSelector] SelectedIn = ProgSelector
SelectorMode = 1（最大值选择）且 Override 为假	Out = [InsUsed] 的最大值 SelectedIn = 最大输入值对应的索引值
SelectorMode = 2（最小值选择）且 Override 为假	Out = [InsUsed] 的最小值 SelectedIn = 最小输入值对应的索引值
SelectorMode = 3（中位数选择）且 Override 为假	Out = [InsUsed] 的中位数 SelectedIn = 输入值中位数对应的索引值
SelectorMode = 4（平均值选择）且 Override 为假	Out = In[InsUsed] 的平均值 SelectedIn = 0

当 SelectorMode 为 1 到 4 之间的值时，如果任何输入出现不良状况，选择时将会忽略该不良输入。例如，如果 SelectorMode = 1（最大值选择），并且 In6 的值最大但状况不良，则指令会将状况良好的次最大输入值送入输出。

对于最大值或最小值选择模式，如果两路输入相等且同为最大值或最小值，则指令会输出找到的第一路输入。对于中位数选择模式，中位数始终代表从可用输入中选择的值。如果多路输入的值为中位数，则指令会输出找到的第一路输入。

在程序控制与操作员控制之间切换

下图显示了 ESEL 指令在程序控制与操作员控制模式之间进行切换的方式。



(1) ProgOperReq 保持为真时，指令锁定为操作员控制模式。

(2) ProgProgReq 保持为真且 ProgOperReq 为假时，指令锁定为程序控制模式。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行
指令首次运行	指令设置为操作员控制模式。
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

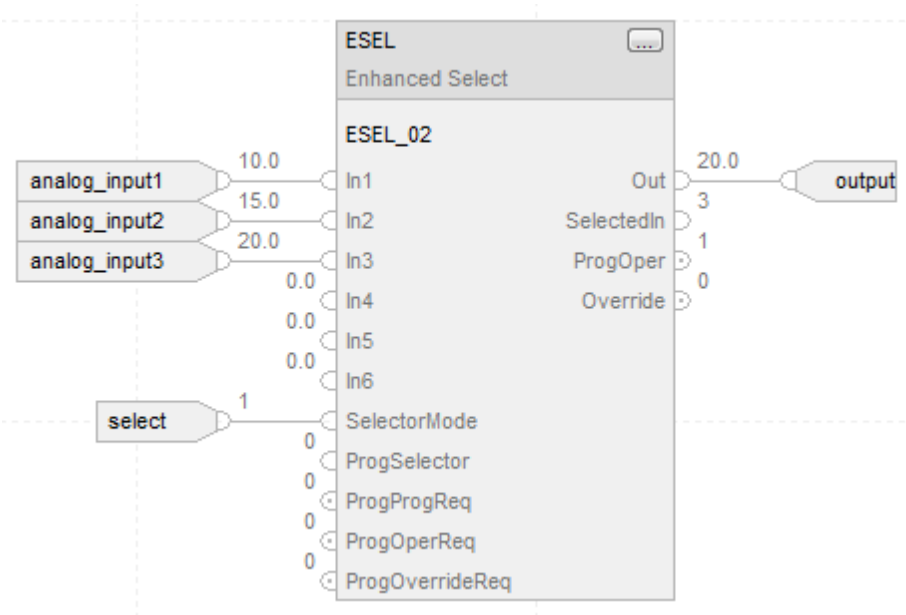
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

该 ESEL 指令根据 SelectorMode 选择 In1、In2 或 In3。在本示例中，SelectorMode = 1，表示最大值选择。指令将找到最大的输入值，并将 Out 值设置为最大 In 值。

功能块



结构化文本

```
ESEL_01.In1 := analog_input1;
ESEL_01.In2 := analog_input2;
ESEL_01.In3 := analog_input3;
ESEL_01.SelectorMode := 1;
ESEL(ESEL_01);

selected_value := ESEL_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

功能块面板控件 参考页数 604

上限/下限 (HLL)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

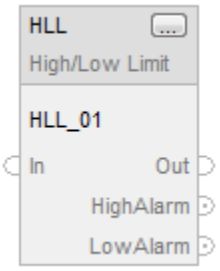
HLL 指令用于将模拟量输入限制在两个值之间。可以选择上限/下限、上限或下限三种限制方式。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

HLL(HLL_tag);

操作数

功能块

操作数	类型	格式	说明
HLL tag	HL_LIMIT	结构	HLL 结构

HL_LIMIT 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
HighLimit	REAL	输入上限。如果 SelectLimit = 0 且 HighLimit ≤ LowLimit，指令会将 Status 中的相应位置位，并会设置 Out = LowLimit。 有效值 = HighLimit > LowLimit 默认值 = 0.0
LowLimit	REAL	输入下限。如果 SelectLimit = 0 且 LowLimit ≥ HighLimit，指令会将 Status 中的相应位置位，并会设置 Out = LowLimit。 有效值 = LowLimit < HighLimit 默认值 = 0.0
SelectLimit	DINT	选择限值输入。此输入有三种设置： 0 = 同时使用两个限值 1 = 使用上限 2 = 使用下限 如果 SelectLimit 无效，指令会假定 SelectLimit = 0，并会将 Status 中的相应位置位。 有效值 = 0 到 2 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。 如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。

HighAlarm	BOOL	上限报警指示器。当 In ≥ HighLimit 时设置为真。将 SelectLimit 设为 2 时，HighAlarm 禁用。
LowAlarm	BOOL	下限报警指示器。当 In ≤ LowLimit 时设置为真。将 SelectLimit 设为 1 时，LowAlarm 禁用。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
LimitsInv (Status.1)	BOOL	HighLimit ≤ LowLimit。
SelectLimitInv (Status.2)	BOOL	SelectLimit 的值不是 0、1 或 2。

结构化文本

操作数	类型	格式	说明
HLL tag	HL_LIMIT	结构	HLL 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

HLL 指令使用以下规则确定 Out 的值：

选择	条件	操作
SelectLimit = 0 （使用上限和下限）	In < HighLimit 且 In > LowLimit	Out = In
	In ≥ HighLimit	Out = HighLimit
	In ≤ LowLimit	Out = LowLimit
	HighLimit ≤ LowLimit	Out = LowLimit
SelectLimit = 1 （仅使用上限）	In < HighLimit	Out = In
	In≥HighLimit	Out = HighLimit
SelectLimit = 2 （仅使用下限）	In > LowLimit	Out = In
	In ≤ LowLimit	Out = LowLimit

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

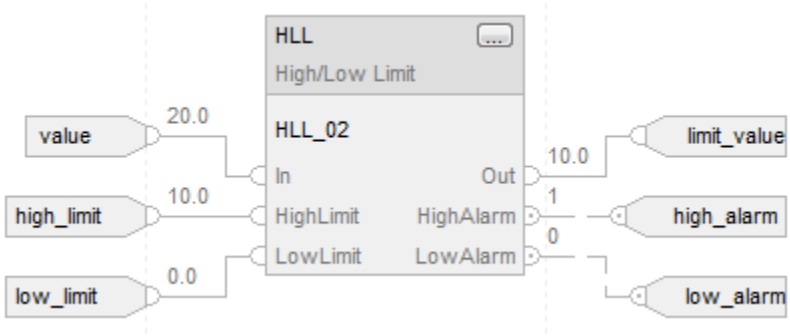
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

此 HLL 指令会将 In 的值限制在两个值之间，当 In 超出限值时，可根据需要将 HighAlarm 或 LowAlarm 置位。指令将 Out 设为限制的 In 值。

功能块



结构化文本

```
HLL_01.In := value;

HLL_01.HighLimit := high_limit;

HLL_01.LowLimit := low_limit;

HLL(HLL_01);

limited_value := HLL_01.Out;

high_alarm := HLL_01.HighAlarm;

low_alarm := HLL_01.LowAlarm;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

多路选择器 (MUX)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

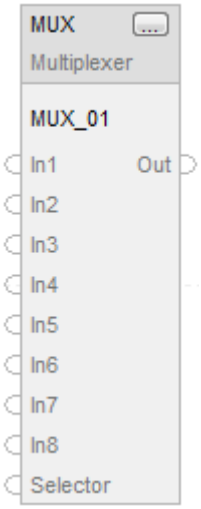
MUX 指令根据选择器输入选择八路输入其中之一。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

此指令不可用于梯形图中。

操作数

功能块

操作数	类型	格式	说明
Block tag	MULTIPLEXER	结构	MUX 结构

MUX 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In1	REAL	指令的第一个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0

In2	REAL	指令的第二个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In3	REAL	指令的第三个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In4	REAL	指令的第四个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In5	REAL	指令的第五个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In6	REAL	指令的第六个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In7	REAL	指令的第七个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In8	REAL	指令的第八个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Selector	DINT	指令的选择器输入。此输入用于确定将哪一输入 (1-8) 移入 Out。如果该值无效（其中包括 0），指令会将 Status 中的相应位置位，并使 Out 保持其当前值。 有效值 = 1 到 8 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。溢出时清零。
Out	REAL	已选择的算法输出。该输出的数学状态标志置位。
Status	DINT	功能块的状态。

InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
SelectorInv (Status.1)	BOOL	Selector 值无效。

说明

根据 Selector 值，MUX 指令会将 Out 设为八路输入之一。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

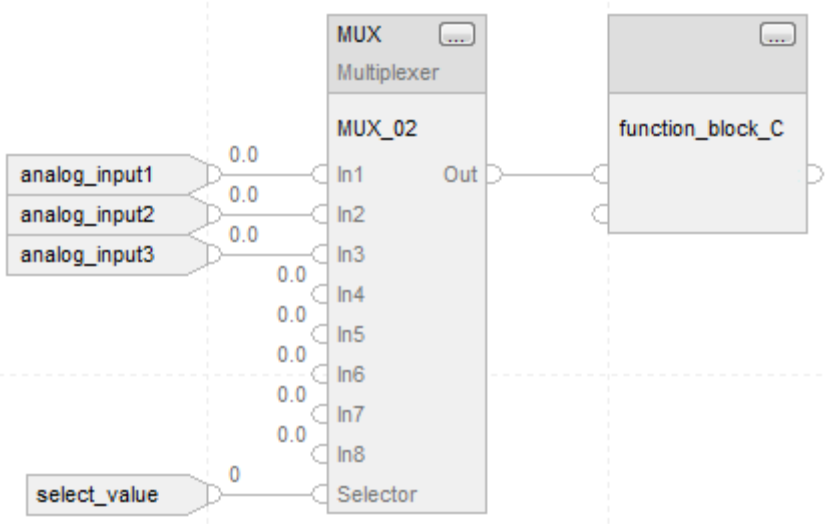
执行

功能块

条件	操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	将 Out 的内部值设置为零。
后扫描	EnableIn 和 EnableOut 位设置为假。

示例

功能块



此 MUX 指令会根据 Selector 的值选择 In1、In2、In3、In4、In5、In6、In7 或 In8。指令会设置 Out = In_n，此输出将成为 function_block_C 的输入参数。例如，如果 select_value = 2，指令会设置 Out = analog_input2。

另请参见

通用属性 参考页数 589

变化率限制器 (RLIM)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

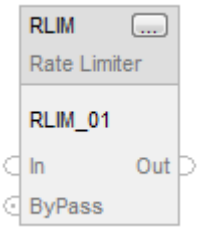
RLIM 指令用于限制信号随时间的变化量。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

RLIM(RLIM_tag);

操作数

功能块

操作数	类型	格式	说明
RLIM tag	RATE_LIMITER	结构	RLIM 结构

RATE_LIMITER 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
IncRate	REAL	最大输出增量变化率，以单位/秒表示。如果该值无效，指令将设置 IncRate = 0.0 并将 Status 中的相应位置位。 有效值 = 任何 ≥ 0.0 的浮点值 默认值 = 0.0
DecRate	REAL	最大输出减量变化率，以单位/秒表示。如果该值无效，指令将设置 DecRate = 0.0 并将 Status 中的相应位置位。 有效值 = 任何 ≥ 0.0 的浮点值 默认值 = 0.0

ByPass	BOOL	绕过算法请求。该参数为真时，Out = In。 默认值为假。
TimingMode	DINT	选择时序执行模式。 0 = 周期模式 1 = 过采样模式 2 = 实时采样模式 有效值 = 0 到 2 默认值 = 0
OversampleDT	REAL	过采样模式的执行时间。 有效值 = 0 到 4194.303 秒 默认值 = 0
RTSTime	DINT	实时采样模式的模块更新周期 有效值 = 1 到 32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样模式的模块时戳值。 有效值 = 0 到 32,767 ms 默认值 = 0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
IncRateInv (Status.1)	BOOL	IncRate < 0。指令使用值 0。
DecRateInv (Status.2)	BOOL	DecRate < 0。指令使用值 0。

TimingModeInv (Status.27)	BOOL	无效的 TimingMode 值。 有关时序模式的更多信息，请参见"功能块属性"部分。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS(DeltaT – RTSTime) > 1 毫秒时，设置为真。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	在过采样模式下，如果 DeltaT <= 0 或 DeltaT > 4194.303，此参数设置为真。

结构化文本

操作数	类型	格式	说明
RLIM tag	RATE_LIMITER	结构	RLIM 结构

有关结构化文本中表达式语法的详细信息，请参见"结构化文本语法"部分。

说明

RLIM 指令提供独立的增量变化率和减量变化率，以单位/秒表示。ByPass 输入用于停止变化率限制，将信号直接传送至输出。

条件	操作
ByPass 为真	$Out_n = In_n$ $Out_{n-1} = In_n$
ByPass 为假，且 DeltaT > 0	$Slope = \frac{In_n - Out_{n-1}}{DeltaT}$ If Slope ≤ –DecRate then YSlope = –DecRate If –DecRate ≤ Slope ≤ IncRate then YSlope = Slope If IncRate ≤ Slope then YSlope = IncRate $Out_n = Out_{n-1}$ + DeltaT x YSlope $Out_{n-1} = Out_n$ where DeltaT is in seconds

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真 指令执行。
指令首次运行	不适用
指令首次扫描	将 Out 初始化为 In 的值。
后扫描	EnableIn 和 EnableOut 位设置为假。

结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

功能块

RLIM 指令通过 IncRate 对 In 进行限制。如果 analog_input1 的变化率大于 IncRate 的值，指令会对 In 进行限制。指令将 Out 设为 In 的变化率限值。

结构化文本

```
RLIM_01.In := analog_input1;
RLIM_01.IncRate := value;
RLIM(RLIM_01);

rate_limited := RLIM_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

选择 (SEL)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

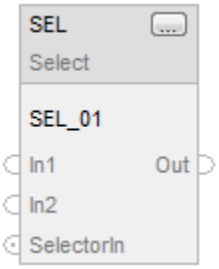
SEL 指令使用数字量输入选择两路输入其中之一。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

此指令不可用于结构化文本中。

操作数

功能块

操作数	类型	格式	说明
SEL tag	SELECT	结构	SEL 结构

SELECT 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In1	REAL	指令的第一个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
In2	REAL	指令的第二个模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
SelectorIn	BOOL	用于在 In1 与 In2 之间进行选择的输入。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。溢出时清零。
Out	REAL	计算所得的算法输出。

说明

SEL 指令的工作方式如下：

条件	操作
SelectorIn 置位	Out = In2
SelectorIn 清零	Out = In1

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

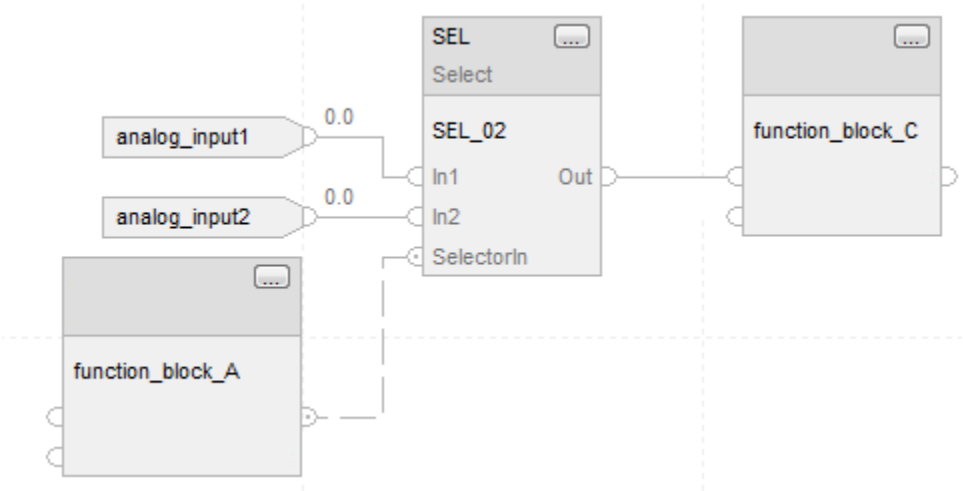
功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	将 Out n-1 设为 0。
后扫描	不适用

示例

SEL 指令会根据 SelectorIn 的值选择 In1 或 In2。如果 SelectorIn 置位，指令将设置 Out = In2。如果 SelectorIn 清零，指令将设置 Out = In1。Out 将成为 function_block_C 的输入参数。

功能块



另请参见

通用属性 参考页数 589

选择取反 (SNEG)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

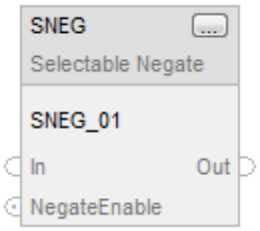
SNEG 指令使用数字量输入在输入值与输入取反值之间进行选择。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

SNEG(SNEG_tag);

操作数

功能块

操作数	类型	格式	说明
SNEG tag	SELECTABLE_NEGATE	结构	SNEG 结构

SNEG 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。 结构化文本： 无影响。指令执行。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
NegateEnable	BOOL	取反使能。当 NegateEnable 为真时，指令会将 Out 设为 In 的取反值。 默认值为真。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。

结构化文本

操作数	类型	格式	说明
SNEG tag	SELECTABLE_NEGATE	结构	SNEG 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

SNEG 指令的工作方式如下：

条件	操作
NegateEnable 为真	Out = - In
NegateEnable 为假	Out = In

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

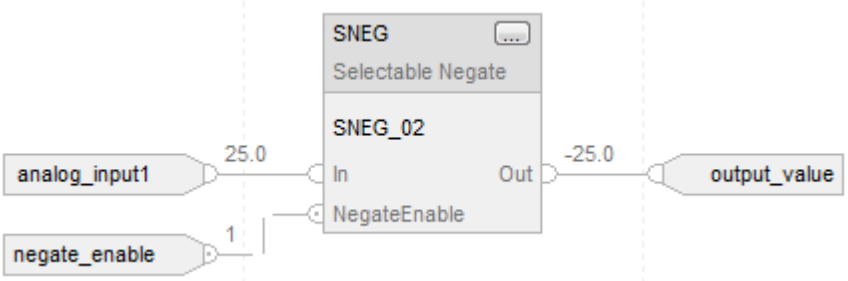
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

negate_enable 确定是否对 In 取反。如果 NegateEnable 为假，指令将设置 Out = In。如果 NegateEnable 为真，指令将设置 Out = -In。

功能块



结构化文本

```
SNEG_01.In := analog_input1;
SNEG_01.NegateEnable := negate_enable;
SNEG(SNEG_01);

output_value := SNEG_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

选择加法器 (SSUM)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

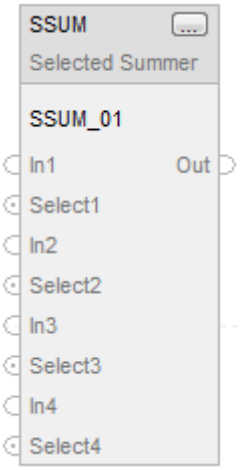
SSUM 指令使用布尔型输入选择要进行代数求和的实数输入。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

SSUM(SSUM_tag);

操作数

功能块

操作数	类型	格式	说明
SSUM tag	SELECTED_SUMMER	结构	SSUM 结构

SELECTABLE_SUMMER 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果为假，该指令不会执行，也不会更新输出。 默认值为真。
In1	REAL	要进行求和的第一路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain1	REAL	第一路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select1	BOOL	第一路输入的选择信号。 默认值为假。

In2	REAL	要进行求和的第二路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain2	REAL	第二路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select2	BOOL	第二路输入的选择信号。 默认值为假。
In3	REAL	要进行求和的第三路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain3	REAL	第三路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select3	BOOL	第三路输入的选择信号。 默认值为假。
In4	REAL	要进行求和的第四路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain4	REAL	第四路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select4	BOOL	第四路输入的选择信号。 默认值为假。
In5	REAL	要进行求和的第五路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain5	REAL	第五路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select5	BOOL	第五路输入的选择信号。 默认值为假。
In6	REAL	要进行求和的第六路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain6	REAL	第六路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0

Select6	BOOL	第六路输入的选择信号。 默认值为假。
In7	REAL	要进行求和的第七路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain7	REAL	第七路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select7	BOOL	第七路输入的选择信号。 默认值为假。
In8	REAL	要进行求和的第八路输入。 有效值 = 任意浮点值 默认值 = 0.0
Gain8	REAL	第八路输入的增益。 有效值 = 任意浮点值 默认值 = 1.0
Select8	BOOL	第八路输入的选择信号。 默认值为假。
Bias	REAL	偏置信号输入。指令会将 Bias 与输入之和相加。 有效值 = 任意浮点值 默认值 = 0.0

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。 如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。

结构化文本

操作数	类型	格式	说明
SSUM tag	SELECTED_SUMMER	结构	SSUM 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

SSUM 指令的工作方式如下：

条件	操作
未选择任何 In	Out = Bias
已选择一个或多个 In	对于所有 n，Selectn 为真时 $Out = \sum (In_n \times Gain_n) + Bias$

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

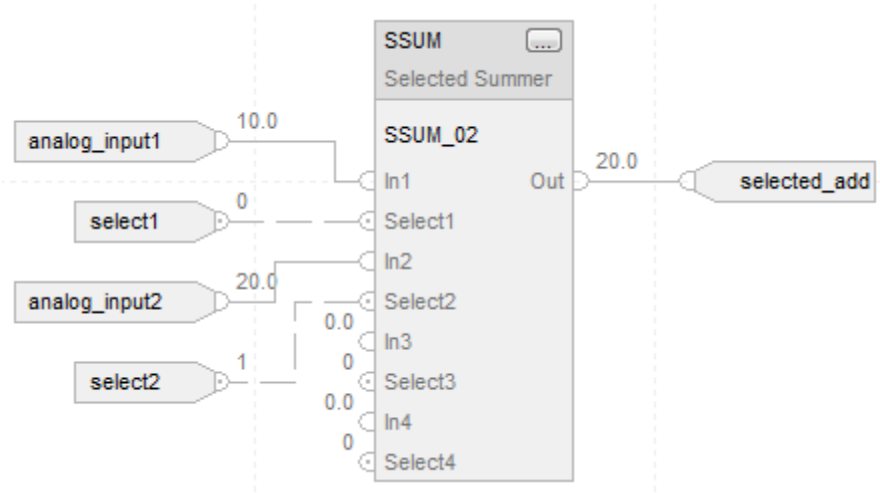
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

select1 和 select 2 的值分别确定是否选择 analog_input1 和 analog_input2。指令随后会将所选输入相加，并将结果放入 Out。

功能块



结构化文本

```
SSUM_01.In1 := analog_input1;
SSUM_01.Select1 := select1;
SSUM_01.In2 := analog_input2;
SSUM_01.Select2 := select2;
SSUM(SSUM_01);

selected_add := SSUM_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

统计指令

统计指令

统计指令包括以下指令：

可用指令

梯形图

不可用

功能块和结构化文本

MAVE	MAXC	MINC	MSTD

如果希望	使用此指令
计算时间平均值。	MAVE
查找最大时间信号。	MAXC
查找最小时间信号。	MINC
计算移动标准偏差。	MSTD

另请参见

滤波指令 参考页数 365

逻辑和移动指令 参考页数 453

驱动指令 参考页数 309

选择/限制指令 参考页数 395

过程控制指令 参考页数 21

移动平均值 (MAVE)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

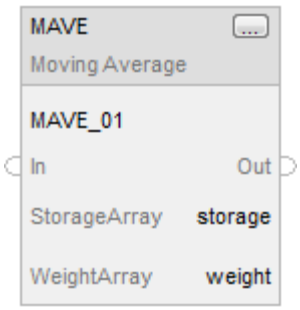
MAVE 指令用于计算 In 信号的时间平均值。该指令有选择地支持用户指定的权重。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

MAVE(MAVE_tag,storage,weight);

操作数

功能块

操作数	类型	格式	说明
MAVE tag	MOVING_AVERAGE	结构	MAVE 结构
storage	REAL	数组	存储移动平均值采样。此数组的大小至少应与 NumberOfSamples 相当。

weight	REAL	数组	（可选） 用于加权平均值。此数组的大小至少应与 NumberOfSamples 相当。 元素 [0] 用于最新的采样；元素 [n] 用于最早的采样。

MOVING_AVERAGE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
InFault	BOOL	输入不良状况指示器。如果 In 是从模拟量输入读取的，则 InFault 通常由模拟量输入的故障状态控制。置位时，InFault 指示输入信号存在错误，指令会将 Status 中的相应位置位，并且指令会使 Out 保持其当前值。 当 InFault 由置位跳变为清零时，指令会初始化平均值算法，然后继续执行。 默认清零。
Initialize	BOOL	指令的初始化输入。此参数置位时，指令会保持 Out = In，但 InFault 置位时除外，此时，指令会使 Out 保持其当前值。当 Initialize 由置位跳变为清零时，指令会初始化平均值算法，然后继续执行。 默认清零。
SampleEnable	BOOL	In 采样使能。此参数置位时，指令会将 In 的值输入到 storage 数组并计算新的 Out 值。当 SampleEnable 清零且 Initialize 清零时，指令会使 Out 保持其当前值。 默认置位。

NumberOfSamples	DINT	计算中要使用的样本数。如果该值无效，指令会将 Status 中的相应位置位，并使 Out 保持其当前值。当 NumberOfSamples 再次变为有效时，指令将初始化平均值算法然后继续执行。 有效值 = 1 到（StorageArray 或 WeightArray（若使用）的最小大小） 默认值 = 1
UseWeights	BOOL	指令的平均值方案输入。此参数置位时，指令会使用加权方法计算 Out。此参数清零时，指令会使用统一方法计算 Out。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InFaulted (Status.1)	BOOL	In 不良（InFault 置位）。
NumberOfSampInv (Status.2)	BOOL	NumberOfSamples 无效或与数组大小不相符。

结构化文本

操作数	类型	格式	说明
MAVE tag	MOVING_AVERAGE	结构	MAVE 结构
storage	REAL	数组	存储移动平均值采样。此数组的大小至少应与 NumberOfSamples 相当。

weight	REAL	数组	（可选） 用于加权平均值。此数组的大小至少应与 NumberOfSamples 相当。 元素 [0] 用于最新的采样；元素 [n] 用于最早的采样。

MOVING_AVERAGE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
InFault	BOOL	输入不良状况指示器。如果 In 是从模拟量输入读取的，则 InFault 通常由模拟量输入的故障状态控制。置位时，InFault 指示输入信号存在错误，指令会将 Status 中的相应位置位，并且指令会使 Out 保持其当前值。 当 InFault 由置位跳变为清零时，指令会初始化平均值算法，然后继续执行。 默认清零。
Initialize	BOOL	指令的初始化输入。此参数置位时，指令会保持 Out = In，但 InFault 置位时除外，此时，指令会使 Out 保持其当前值。当 Initialize 由置位跳变为清零时，指令会初始化平均值算法，然后继续执行。 默认清零。
SampleEnable	BOOL	In 采样使能。此参数置位时，指令会将 In 的值输入到 storage 数组并计算新的 Out 值。当 SampleEnable 清零且 Initialize 清零时，指令会使 Out 保持其当前值。 默认置位。

NumberOfSamples	DINT	计算中要使用的样本数。如果该值无效，指令会将 Status 中的相应位置位，并使 Out 保持其当前值。当 NumberOfSamples 再次变为有效时，指令将初始化平均值算法然后继续执行。 有效值 = 1 到（StorageArray 或 WeightArray（若使用）的最小大小） 默认值 = 1
UseWeights	BOOL	指令的平均值方案输入。此参数置位时，指令会使用加权方法计算 Out。此参数清零时，指令会使用统一方法计算 Out。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InFaulted (Status.1)	BOOL	In 不良（InFault 置位）。
NumberOfSampInv (Status.2)	BOOL	NumberOfSamples 无效或与数组大小不相符。

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

MAVE 指令计算输入信号的加权或非加权移动平均值。
NumberOfSamples 指定移动平均值量程的长度。在每次扫描功能块时，如果 SampleEnable 置位，指令会将 In 的值移入 storage 数组并丢弃最早的值。每个 Inn 都有用户配置的 Weightn，Weightn 在 UseWeights 置位时使用。

条件	操作
加权平均法 UseWeights 置位。	$Out = \sum_{n=1}^{NumberOfSamples} Weight_n \times In_n$
统一平均法 UseWeights 清零。	$Out = \frac{\sum_{n=1}^{NumberOfSamples} In_n}{NumberOfSamples}$

指令不会将无效 In 值（NAN 或 ±INF）放入存储数组。当 In 无效时，指令会设置 Out = In，并记录轻微溢出故障（如果已启用此报告）。In 变为有效后，指令会初始化平均值算法然后继续执行。

可在运行时更改 NumberOfSamples 参数。如果增大数目，指令将对从当前样本量到新样本量的新数据递增求平均值。如果减小数目，指令会重新计算从样本数组开始到新 NumberOfSamples 值的平均值。

初始化平均值算法

特定情况下（例如指令首次扫描和指令首次运行时），需要由指令对移动平均值算法进行初始化。出现这种情况时，指令会将样本 StorageArray 视为空，并对从 1 到 NumberOfSamples 的样本递增求平均值。例如：

NumberOfSamples = 3, UseWeights is set
- Scan 1: Out = In_n*$Weight_1$
- Scan 2: Out = (In_n*$Weight_1$)+(In_{n-1}*$Weight_2$)
- Scan 3: Out = (In_n*$Weight_1$)+(In_{n-1}*$Weight_2$) +(In_{n-2}*$Weight_3$)

NumberOfSamples = 3, UseWeights is cleared
- Scan 1: Out = In_n/1
- Scan 2: Out = (In_n+In_{n-1})/2
- Scan 3: Out = (In_n+In_{n-1}+In_{n-2})/NumberOfSamples

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	请求在指令下次执行时重新初始化 storage 数组。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	将 Out 初始化为零。
后扫描	EnableIn 和 EnableOut 位设置为假。

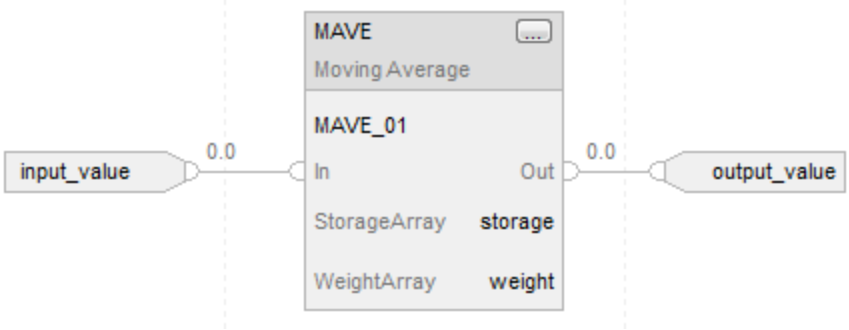
结构化文本

条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

示例

每次扫描时，指令都会将 input_value 放入数组 storage。指令会计算 storage 数组中值的平均值（可选择使用 weight 数组中的权重值），并将结果放入 Out。

功能块



结构化文本

```
MAVE_01.In := input_value;
MAVE(MAVE_01,storage,weight);
output_value := MAVE_01.Out;
```

另请参见

功能块属性 参考页数 545

通用属性 参考页数 589

结构化文本语法 参考页数 559

最大值捕捉 (MAXC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

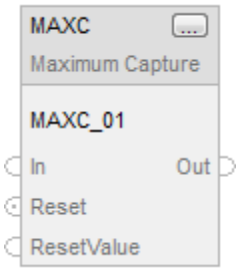
MAXC 指令用于保留某段时间内输入的最大值，并支持用户根据需要重新确定最大值。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

```
MAXC(MAXC_tag);
```

操作数

功能块

操作数	类型	格式	说明
MAXC tag	MAXIMUM_CAPTURE	结构	MAXC 结构

MAXIMUM_CAPTURE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
Reset	BOOL	复位控制算法请求。只要 Reset 置位，指令就会设置 Out = ResetValue。 默认清零。
ResetValue	REAL	指令的复位值。只要 Reset 置位，指令就会设置 Out = ResetValue。 有效值 = 任意浮点值 默认值 = 0.0

输出参数	数据类型	说明
EnableOut	BOOL	启用输出。
Out	REAL	计算所得的算法输出。

结构化文本

操作数	类型	格式	说明
MAXC tag	MAXIMUM_CAPTURE	结构	MAXC 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

说明

MAXC 指令执行此算法：

条件	操作
Reset 置位	LastMaximum = 复位值 Out = LastMaximum
Reset 清零	如果 In > LastMaximum，则更新 LastMaximum。 Out = LastMaximum。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	将请求置位以将最大值初始化为当前输入。
后扫描	EnableIn 和 EnableOut 位设置为假。

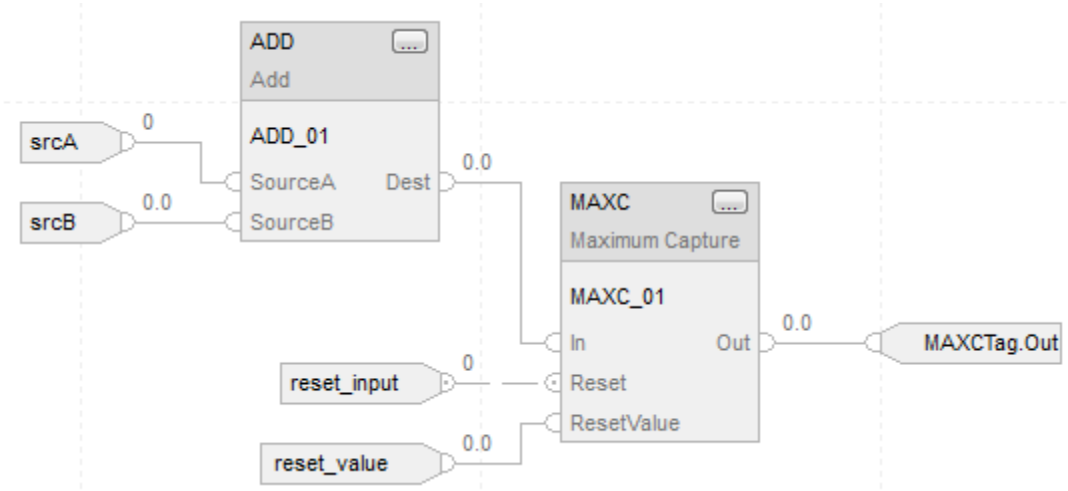
结构化文本

条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

示例

如果 Reset 置位，指令会设置 Out = ResetValue。如果 Reset 清零，当 In > LastMaximum 时，指令会设置 Out = In。否则，指令会设置 Out = LastMaximum。

功能块



结构化文本

```
MAXCTag.In := input_value;

MAXCTag.Reset := reset_input;

MAXCTag.ResetValue := reset_value;

MAXC(MAXCTag);

result := MAXCTag.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

最小值捕捉 (MINC)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

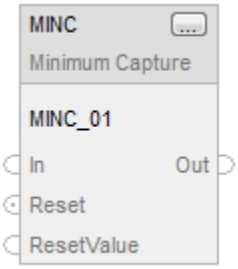
MINC 指令用于保留某段时间内输入的最小值，并支持用户根据需要重新确定最小值。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

MINC(MINC_tag);

操作数

功能块

操作数	类型	格式	说明
MINC tag	MINIMUM_CAPTURE	结构	MINC 结构

MINIMUM_CAPTURE 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
复位	BOOL	复位控制算法请求。只要 Reset 置位，指令就会设置 Out = ResetValue。 默认清零。

ResetValue	REAL	指令的复位值。只要 Reset 置位，指令就会设置 Out = ResetValue。 有效值 = 任意浮点值 默认值 = 0.0

输出参数	数据类型	说明
EnableOut	BOOL	使能输出。
Out	REAL	计算所得的算法输出。

结构化文本

操作数	类型	格式	说明
MINC tag	MINIMUM _CAPTURE	结构	MINC 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

说明

MINC 指令执行此算法：

条件	操作
Reset 置位	LastMinimum = ResetValue Out = ResetValue
Reset 清零	如果 In < LastMinimum，则更新 LastMinimum。 Out = LastMinimum。

影响数学状态标志

无

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。

Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	不适用
指令首次扫描	将请求置位以将最大值初始化为当前输入。
后扫描	EnableIn 和 EnableOut 位设置为假。

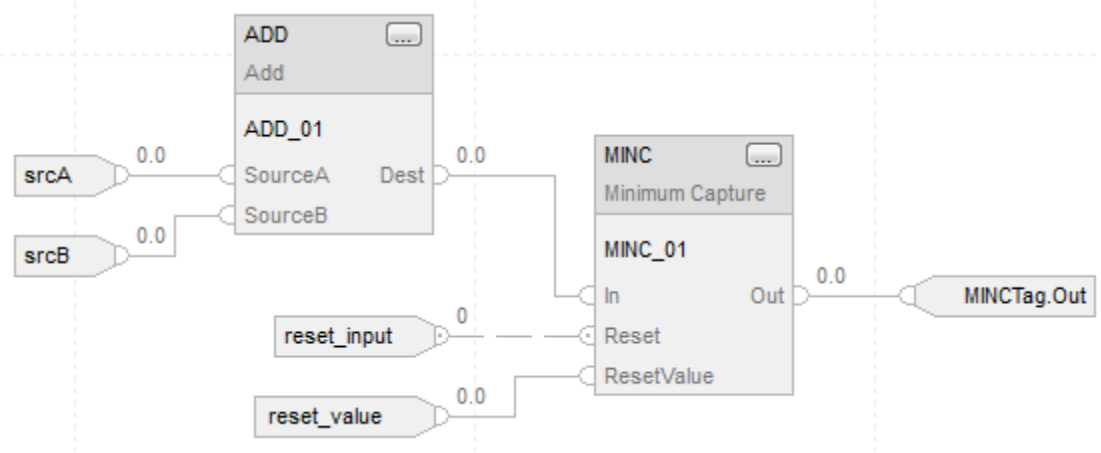
结构化文本

条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

示例

如果 Reset 置位，指令会设置 Out = ResetValue。如果 Reset 清零，当 In < LastMinimum 时，指令会设置 Out = In。否则，指令会设置 Out= LastMinimum。

功能块



结构化文本

```
MINCTag.In := input_value;

MINCTag.Reset := reset_input;

MINCTag.ResetValue := reset_value;
```

MINC(MINCTag);

result := MINCTag.Out;

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

移动标准偏差 (MSTD)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

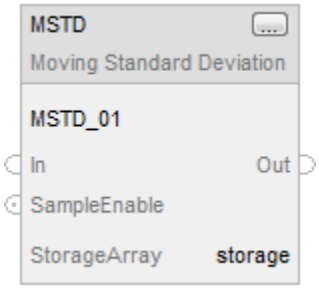
MSTD 指令可计算 In 信号的移动标准偏差和平均值。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

MSTD(MSTD_tag, StorageArray);

操作数

功能块

操作数	类型	格式	说明
block tag	MOVING_STD_DEV	结构	MSTD 结构
StorageArray	REAL	数组	用于保留 In 样本。此数组的大小至少应与 NumberOfSamples 相当。

MOVING_STD_DEV 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
In	REAL	指令的模拟信号输入。 有效值 = 任意浮点值 默认值 = 0.0
InFault	BOOL	输入不良状况指示器。如果 In 是从模拟量输入读取的，则 InFault 通常由模拟量输入的故障状态控制。此参数置位时，InFault 指示输入信号存在错误，指令会将 Status 中的相应位置位，并使 Out 和 Average 保持其当前值。当 InFault 由置位跳变为清零时，指令会初始化平均值算法，然后继续执行。 默认清零。
Initialize	BOOL	指令的初始化输入。此参数置位时，指令会设置 Out = 0.0 且 Average = In，但如果 InFault 也置位，指令会使 Out 和 Average 保持其当前值。当 Initialize 由置位跳变为清零时，指令会初始化标准偏差算法，然后继续执行。 默认清零。

SampleEnable	BOOL	In 采样使能。此参数置位时，指令会将 In 的值输入到存储数组中并计算新的 Out 和 Average 值。当 SampleEnable 清零且 Initialize 清零时，指令会使 Out 和 Average 保持其当前值。 默认清零。
NumberOfSamples	DINT	计算中要使用的样本数。如果该值无效 ,指令会将 Status 中的相应位置位，并使 Out 和 Average 保持其当前值。当 NumberOfSamples 再次变为有效时，指令将初始化标准偏差算法，然后继续执行。 有效值 = 1 到存储数组大小 默认值 = 1

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。如果 Out 溢出，则设置为假。
Out	REAL	计算所得的算法输出。
Average	REAL	计算出的算法平均值。
Status	DINT	功能块的状态。
InstructFault (Status.0)	BOOL	该指令检测到以下执行错误之一。这不是轻微或严重的控制器错误。检查其他状态位以确定发生的情况。
InFaulted (Status.1)	BOOL	In 状况不良。InFault 置位。
NumberOfSampInv (Status.2)	BOOL	NumberOfSamples 无效或与数组大小不相符。

结构化文本

操作数	类型	格式	说明
block tag	MOVING_STD_DEV	结构	MSTD 结构

StorageArray	REAL	数组	用于保留 In 样本。此数组的大小至少应与 NumberOfSamples 相当。

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

MSTD 指令支持任何输入队列长度。每次扫描时，如果 SampleEnable 置位，指令会将 In 值输入到存储数组中。存储数据存满后，每输入一个新 In 值，就将最早的条目从数组中删除。

MSTD 指令使用以下公式计算输出值：

条件	操作
Average	$Average = \frac{\sum_{n=1}^{NumberOfSamples} In_n}{NumberOfSamples}$
Out	$Out = \sqrt{\frac{\sum_{n=1}^{NumberOfSamples} (In_n - Average)^2}{NumberOfSamples}}$

指令不会将无效 In 值（NAN 或 ± INF）翻入存储数组。当 In 无效时，指令会设置 Out = In，设置 Average = In，并记录轻微溢出故障（如果已启用此报告功能）。当 In 变为有效后，指令会初始化标准偏差算法，然后继续执行。

可在运行时更改 NumberOfSamples 参数。如果增加样本数，指令会以增量方式处理从当前样本量到新样本量的新数据。如果减少样本数，指令会重新计算从样本数组起点直到新 NumberOfSamples 值的标准偏差。

初始化标准偏差算法

特定情况下（例如指令首次扫描和指令首次运行），需要由指令对标准偏差算法进行初始化。出现这种情况时，指令会将 StorageArray 视为空，并以增量方式处理从 1 到 NumberOfSamples 值对应的样本。例如：

NumberOfSamples = 3

Scan 1: $Average = In_n/1$

$Out = Square\ root\ (((In_n-Average)^2)/1)$

Scan 2: $Average = (In_n+In_{n-1})/2$

$Out = Square\ root\ (((In_n-Average)^2+(In_{n-1}-Average)^2)/2)$

Scan 3: $Average = (In_n+In_{n-1}+In_{n-2})/NumberOfSamples$

$Out = Square\ root\ (((In_n-Average)^2+(In_{n-1}-Average)^2+(In_{n-2}-Average)^2)/NumberOfSamples)$

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见“通用属性”部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	初始化先前的 Output 和 Average。
指令首次扫描	将 Out 初始化为零。 将 Average 初始化为 Input 值 初始化指令算法。
后扫描	EnableIn 和 EnableOut 位设置为假。

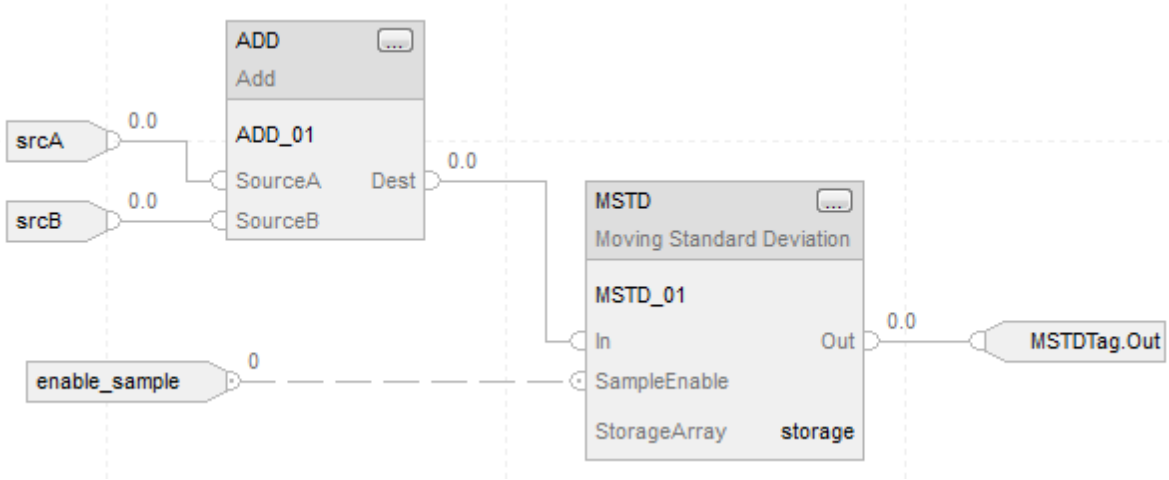
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

每次扫描时，如果 SampleEnable 置位，指令会将 In 值放入存储数组中，并计算存储数组中各数值的标准偏差，然后将结果放入 Out。Out 将成为 function_block_C 的输入参数。

功能块



结构化文本

```
MSTD_01.In := input_value;
MSTD_01.SampleEnable := enable_sample;
MSTD(MSTD_01,storage);
deviation := MSTD_01.Out;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

逻辑和移动

逻辑和移动指令

这些指令用于对输出数据进行逻辑运算和移动。

可用指令

梯形图和结构化文本

DFF	JKFF	RESD	SETD

功能块

不可用

结构化文本

DFF	JKFF	RESD	SETD

另请参见

滤波指令 参考页数 365

驱动指令 参考页数 309

过程控制指令 参考页数 21

选择/限制指令 参考页数 395

统计指令 参考页数 431

D 触发器 (DFF)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

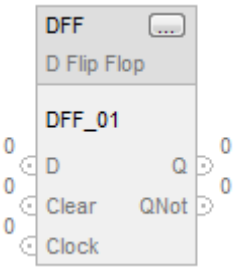
DFF 指令会在 Clock 输入由清零跳变为置位时，将 Q 输出设为 D 输入的状态。将 QNot 输出设为 Q 输出的非状态。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

DFF(DFF_tag);

操作数

功能块

操作数	类型	格式	说明
DFF tag	FLIP_FLOP_D	结构	DFF 结构

FLIP_FLOP_D 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
D	BOOL	指令的输入。 默认清零。
Clear	BOOL	指令的清零输入。如果该参数置位，指令会将 Q 清零，并将 QNot 置位。
Clock	BOOL	指令的时钟输入。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Q	BOOL	指令的输出。
QNot	BOOL	Q 输出取反。

结构化文本

操作数	类型	格式	说明
DFF tag	FLIP_FLOP_D	结构	DFF 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

说明

Clear 置位时，指令会将 Q 清零，并将 QNot 置位。否则，如果 Clock 置位且 Clockn-1 清零，指令会设置 Q=D 并设置 QNot = NOT (D)。

每次扫描时，指令会设置 Clockn-1 = Clock 状态。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。

指令首次运行	Clockn-1 置 1。 Qn-1 清零。
指令首次扫描	前一输入 Clock 状态设置为真。 前一输出 Q 状态设置为假。
后扫描	EnableIn 和 EnableOut 位设置为假。

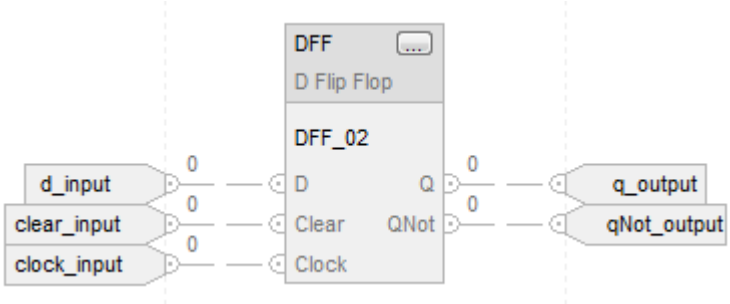
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

Clock 由清零转换为置位时，DFF 指令会设置 Q = D。Clear 置位时，Q 会清零。DFF 指令将 QNot 设置为 Q 的非状态。

功能块



结构化文本

```
DFF_03.D := d_input;
DFF_03.Clear := clear_input;
DFF_03.Clock := clock_input;
DFF(DFF_03);
q_output := DFF_03.Q;
qNot_output := DFF_03.QNot;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

JK 触发器 (JKFF)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

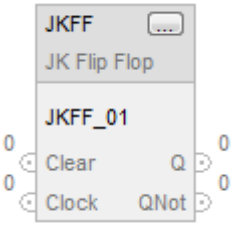
Clock 输入由清零转换为置位时，JKFF 指令会对 Q 和 QNot 输出取反。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

JKFF(JKFF_tag);

操作数

功能块

操作数	类型	格式	说明
JKFF tag	FLIP_FLOP_JK	结构	JKFF 结构

FLIP_FLOP_JK 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
Clear	BOOL	指令的清零输入。如果该参数置位，指令 会将 Q 清零，并将 QNot 置位。
Clock	BOOL	指令的时钟输入。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Q	BOOL	指令的输出。
QNot	BOOL	Q 输出取反。

结构化文本

操作数	类型	格式	说明
JKFF tag	FLIP_FLOP_JK	结构	JKFF 结构

有关结构化文本中表达式语法的详细信息，请参见“结构化文本语法”部分。

说明

Clear 置位时，指令会将 Q 清零，并将 QNot 置位。否则，如果 Clock 置位且 Clockn-1 清零，指令会切换 Q 和 QNot。

每次扫描时，指令会设置 Clockn-1 = Clock 状态。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	Clockn-1 置 1。 Qn-1 清零。
指令首次扫描	前一输入 Clock 状态设置为真。 前一输出 Q 状态为假。
后扫描	EnableIn 和 EnableOut 位设置为假。

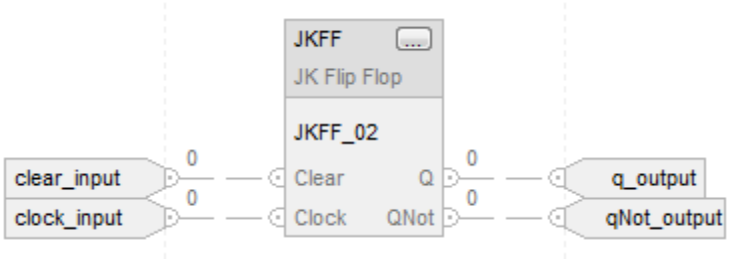
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

Clock 由清零转换为置位时，JKFF 指令会切换 Q。Clear 置位时，Q 始终清零。JKFF 指令将 QNot 设置为 Q 的非状态。

功能块



结构化文本

```
JKFF_01.Clear := clear_input;
JKFF_01.Clock := clock_input;
JKFF(JKFF_01);
q_output := JKFF_01.Q;
qNot_output := JKFF_01.QNot;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

优先复位 (RESD)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

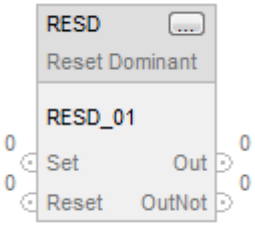
RESD 指令使用 Set 和 Reset 输入控制锁存输出。Reset 输入的优先级高于 Set 输入。

可用语言

梯形图

此指令不可用于梯形图中。

功能块



结构化文本

```
RESD(RESD_tag);
```

操作数

功能块

操作数	类型	格式	说明
RESD tag	DOMINANT_RESET	结构	RESD 结构

DOMINANT_RESET 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
Set	BOOL	指令的置位输入。 默认清零。
Clear	BOOL	指令的复位输入。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Out	BOOL	指令的输出。
OutNot	BOOL	指令的取反输出。

结构化文本

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

说明

优先复位指令使用 Set 和 Reset 输入参数控制锁存输出参数 Out 和 OutNot。Reset 输入的优先级高于 Set 输入。

Set 输入参数设置为真时，Out 将锁存为真状态。如果将 Reset 参数设置为真，将忽略 Out 的当前状态，使其设置为假。

Reset 返回假状态时，Out 将锁存为 Set 输入参数的当前状态。OutNot 将设置为 Out 的非状态。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关故障相关的属性，请参见*通用属性*部分。

执行

功能块

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	前一 Out 值设置为假。
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假。

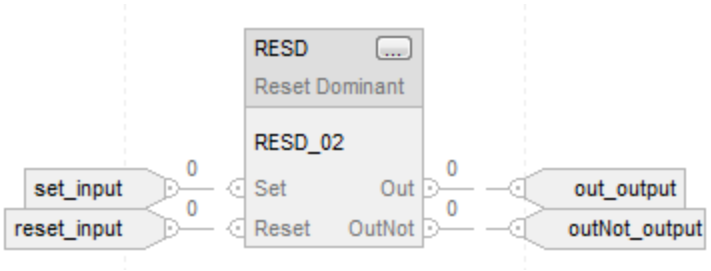
结构化文本

条件/状态	执行的操作
预扫描	请参见"功能块"表中的"预扫描"行。
正常执行	请参见"功能块"表中的"Tag.EnableIn 为真"行。
后扫描	请参见"功能块"表中的"后扫描"行。

示例

Set 为真且 Reset 为假时，Out 设置为真。Reset 为真时，Out 清零。Reset 输入的优先级高于 Set 输入。RESD 指令将 OutNot 设置为 Out 的非状态。

功能块



结构化文本

```
RESD_01.Set := set_input;
RESD_01.Reset := reset_input;
RESD(RESD_01);
out_output := RESD_01.Out;
outNot_output := RESD_01.OutNot;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

优先置位 (SETD)

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。

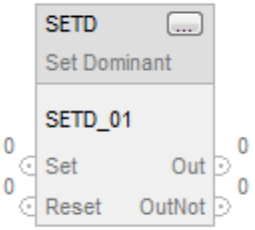
SETD 指令使用 Set 和 Reset 输入控制锁存输出。Set 输入的优先级高于 Reset 输入。

可用语言

梯形图

此指令不可用于梯形图逻辑中。

功能块



结构化文本

SETD(SETD_tag);

操作数

功能块

操作数	类型	格式	说明
SETD tag	DOMINANT_SET	结构	SETD 结构

结构化文本

操作数	类型	格式	说明
SETD tag	DOMINANT_SET	结构	SETD 结构

有关结构化文本中表达式语法的详细信息，请参见*结构化文本语法*部分。

DOMINANT_SET 结构

输入参数	数据类型	说明
EnableIn	BOOL	使能输入。如果清零，该指令将不会执行，并且不会更新输出。 默认值为置位。
Set	BOOL	指令的置位输入。 默认清零。
Clear	BOOL	指令的复位输入。 默认清零。

输出参数	数据类型	说明
EnableOut	BOOL	指示指令是否处于启用状态。
Out	BOOL	指令的输出。
OutNot	BOOL	指令的取反输出。

说明

优先置位指令使用 Set 和 Reset 输入参数控制锁存输出参数 Out 和 OutNot。Set 输入的优先级高于 Reset 输入。

Set 输入参数设置为真时，Out 将锁存为真状态。只有当 Set 输入为假时，将 Reset 参数设置为真才会使 Out 变为假。OutNot 将设置为 Out 的非状态。

影响数学状态标志

否

严重/轻微故障

无此指令特定的故障。有关操作数相关的故障，请参见*通用属性*部分。

执行

条件/状态	执行的操作
预扫描	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为假	EnableIn 和 EnableOut 位设置为假。
Tag.EnableIn 为真	EnableIn 和 EnableOut 位设置为真。 指令执行。
指令首次运行	前一 Out 值设置为真。
指令首次扫描	不适用
后扫描	EnableIn 和 EnableOut 位设置为假

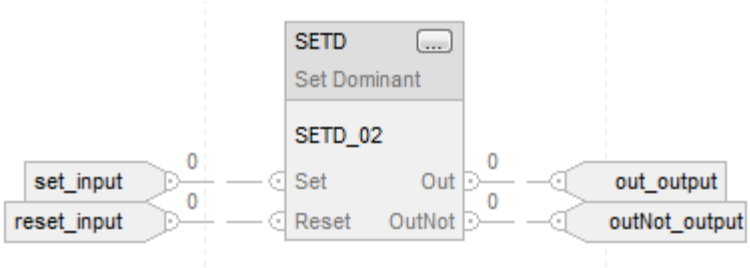
结构化文本

条件/状态	执行的操作
预扫描	请参见“功能块”表中的“预扫描”行。
正常执行	请参见“功能块”表中的“Tag.EnableIn 为真”行。
后扫描	请参见“功能块”表中的“后扫描”行。

示例

Set 为真时，Out 设置为真。Set 为假且 Reset 为真时，Out 清零。Set 输入的优先级高于 Reset 输入。SETD 指令将 OutNot 设置为 Out 的非状态。

功能块



结构化文本

```
SETD_01.Set := set_input;
SETD_01.Reset := reset_input;
SETD(SETD_01);
out_output := SETD_01.Out;
outNot_output := SETD_01.OutNot;
```

另请参见

通用属性 参考页数 589

结构化文本语法 参考页数 559

设备阶段指令

设备阶段指令

设备阶段指令包括：

可用指令

梯形图和结构化文本

PSC	PCMD	POVR	PFL	PCLF	PXRQ	PRNP	PPD	PATT	PDET

功能块

以上指令不可用于功能块中。

如需：	**使用此指令：**
向设备阶段指示状态例程已完成，进而指示设备阶段应进入下一状态。	PSC
更改设备阶段的状态或子状态	PCMD
无论所有权如何，都向设备阶段发出 Hold、Stop 或 Abort 命令。	POVR
指示设备阶段存在故障	PFL
清除设备阶段的故障代码	PCLF
发起与 FactoryTalk Batch 软件的通信。	PXRQ
将设备阶段的 NewInputParameters 位清零	PRNP
在设备阶段的逻辑中设置断点	PPD
获取设备阶段的所有权以执行以下任一操作： • 防止其他程序或 FactoryTalk Batch 软件控制设备阶段。 • 确保其他程序或 FactoryTalk Batch 软件未拥有设备阶段的所有权。	PATT
释放设备阶段所有权	PDET

另请参见

设备顺序图指令 参考页数 530

连接到设备阶段 (PATT)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PATT 指令用于获取某个设备阶段的所有权以实现以下目的：

- 防止其他程序或 FactoryTalk Batch 软件控制设备阶段。
- 确保其他程序或 FactoryTalk Batch 软件未拥有设备阶段的所有权。

PATT 指令可使程序获取某个设备阶段的所有权。

- 所有权是可选择的。只要设备阶段没有确定所有者，任何定序程序（控制器中的程序，FactoryTalk Batch 软件）都可以对设备阶段进行控制。
- FactoryTalk Batch 软件始终拥有设备阶段的所有权。
- 一旦定序程序拥有某设备阶段的所有权，则其他任何定序程序均不能对该设备阶段进行控制。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PATT(Phase_Name,Result);

操作数

梯形图

操作数	类型	格式	说明
阶段名称 (Phase Name)	PHASE	设备阶段的名称	要拥有的设备阶段。
结果 (Result)	DINT	立即数 标签	为使指令返回指示其执行成功或失败的代码，需输入用于存储结果代码的 DINT 标签。否则，需输入 0 。

结构化文本

操作数与梯形图 PATT 指令的操作数相同。

使用 PATT 指令的指导原则

指导原则	详细信息	
如果有多个定序程序使用同一个设备阶段，则需考虑所有权问题。	所有权可以确保程序对所需的所有设备阶段进行控制，并且会排除任何其他定序程序。	
	如果使用：	**则：**
	FactoryTalk Batch 软件同时在此控制器内运行序列	在执行序列（过程）之前，需获取序列使用的所有设备阶段的所有权。
	多个程序对同一设备阶段进行控制	
	其他情况	无需获取设备阶段的所有权。
请记住，Logix Designer 优先对控制器进行控制。	无论是程序还是 FactoryTalk Batch 软件拥有设备阶段的所有权 ,始终可使用 Logix Designer 优先获取所有权并控制设备阶段，使其转换为其他状态。	
	以下各项：	**优先于以下各项：**

	Logix Designer	控制器（内部定序程序），FactoryTalk Batch 软件（外部定序程序）
	控制器（内部定序程序）	无
	FactoryTalk Batch 软件（外部定序程序）	无
使用 Result 操作数来验证所有权。	使用 Result 操作数获取显示 PATT 指令执行成功或失败的代码。要解读结果代码，请参见下文的 *PATT 结果代码*部分。	
避免或规划结果代码 = 24582。	每次执行时，PATT 指令都会尝试获得设备阶段的所有权。一旦程序拥有设备阶段的所有权，再次执行 PATT 指令将生成结果代码 24582。当使用 PATT 指令时，请采取以下任一措施： • 将其执行限制为单次扫描，避免生成结果代码 24582。 • 将结果代码 = 24582 放在所有权条件中。	
序列完成后，会放弃所有权。	要放弃所有权，可使用从设备阶段断开 (PDET) 指令。	

PATT 结果代码

如果分配标签来存储 PATT 指令的结果，则该指令在执行时会返回以下代码之一：

代码（十进制）	说明
0	命令成功执行。
24579	Logix Designer 或 HMI 已获取设备阶段的所有权。调用程序已连接到设备阶段，但未获取当前控制的所有权令。 • 该程序现在也已获取设备阶段的所有权。 • 由于 Logix Designer 或 HMI 的优先级高于程序，因此程序无法控制设备阶段。 • 高优先级 HMI 所有权仅用于 CompactLogix 5370 和 ControlLogix 5570 控制器。
24582	程序已获取设备阶段的所有权。

24593	以下任一项已取得设备阶段的所有权。 • 外部定序程序（FactoryTalk Batch 软件） • 控制器中的其他程序
24594	设备阶段未规划、已被禁止或处于已被禁止的任务中。

影响数学状态标志

无

严重/轻微故障

无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

执行

在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。粗实线下方的所有状况仅能在普通扫描模式期间出现。

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	该指令按上述方式执行。

示例

梯形图

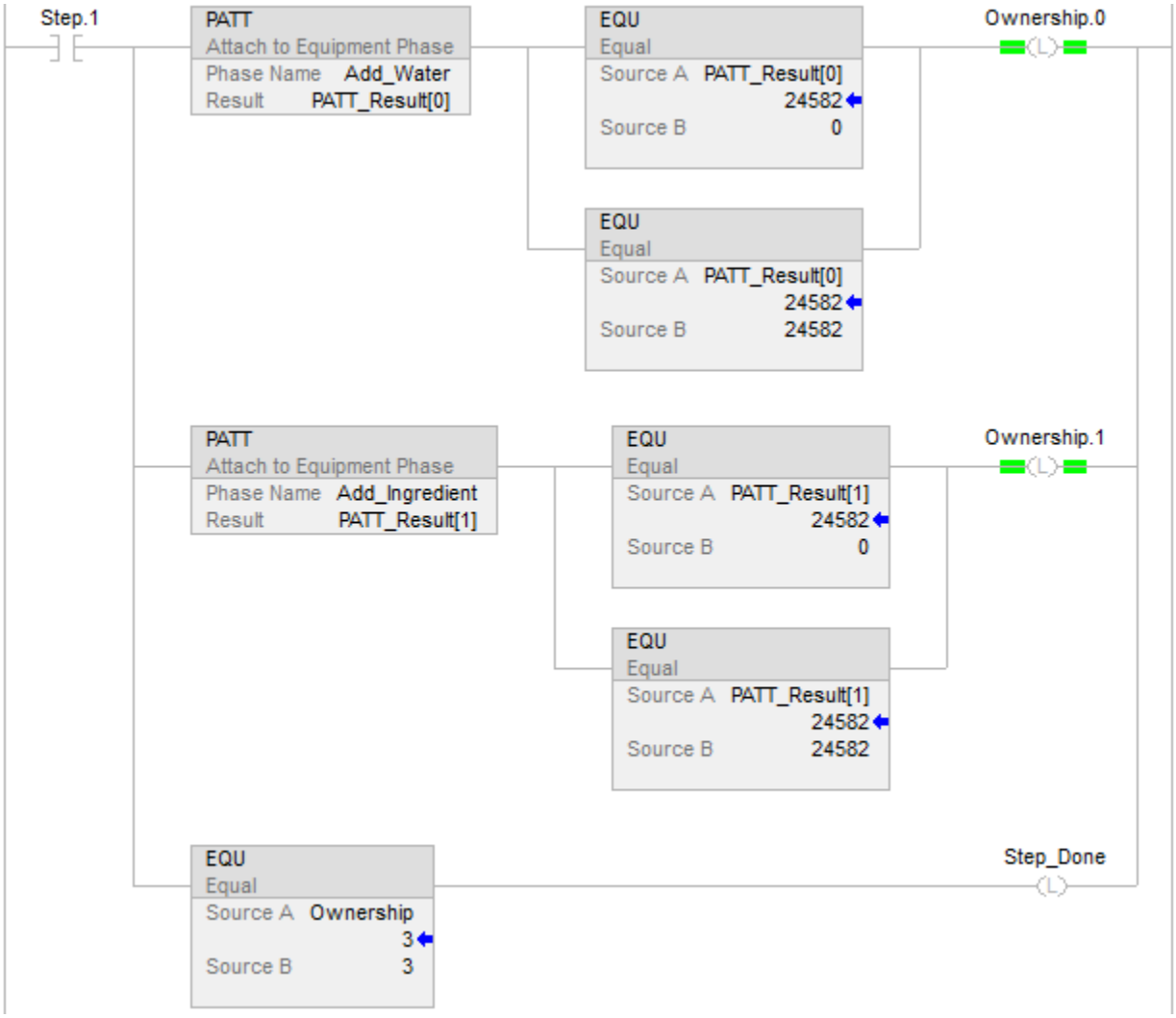
如果 *Step.1* = 1（序列的第一步），则：

每条 PATT 指令都尝试获取设备阶段的所有权。

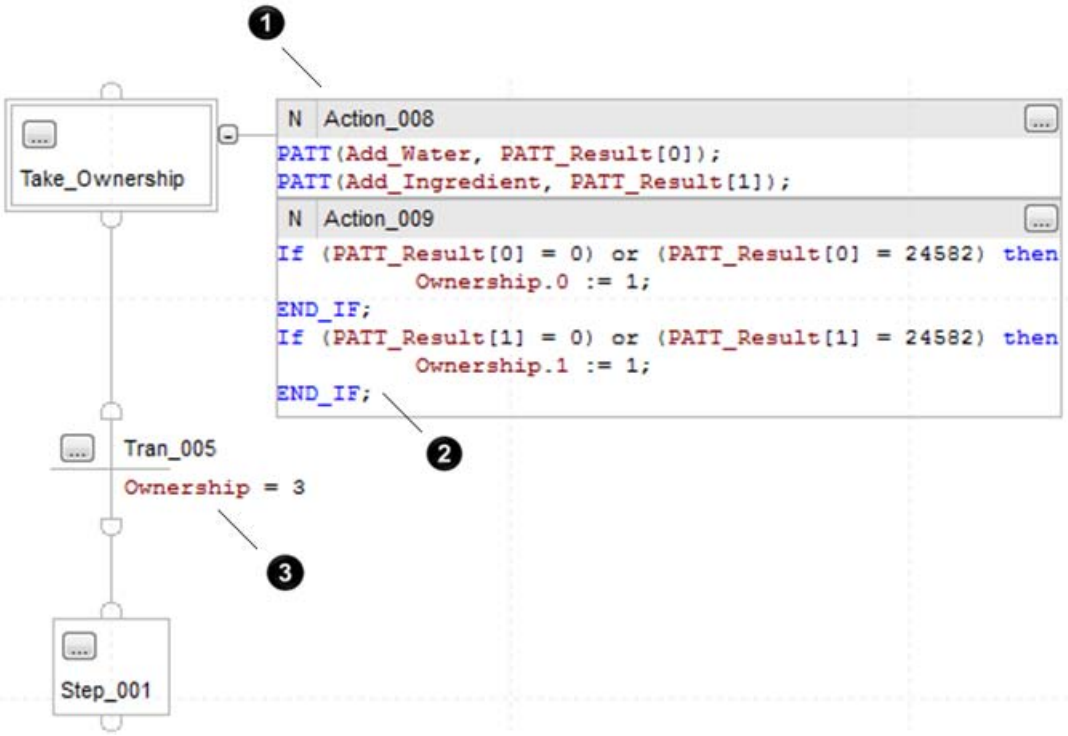
如果 PATT 指令的结果代码为 0 或 24582（程序拥有设备阶段的所有权），则
Ownership 标签的相应位 = 1。（在 *Ownership* 标签中，为每个设备阶段都分配了一个位。）

如果 *Ownership* = 3（程序拥有设备阶段的所有权，由位 0 和位 1 指示），则

Done = 1。（这表示序列进入下一步。）



结构化文本



编号	说明
❶	在该序列中的第一步，Take_Ownership 动作尝试取得该序列所用两个设备阶段的所有权。
❷	Action_009 检查程序是否拥有设备阶段的所有权。如果每条 PATT 指令的结果代码为 0 或 24282（程序拥有设备阶段的所有权），则 Ownership 标签中的位 =1。（在 Ownership 标签中，为每个设备阶段都分配了一个位。）
❸	如果 Ownership=3（程序拥有设备阶段的所有权，由位 0 和位 1 指示），则 SFC 进入下一步。

另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

从设备阶段断开 (PDET)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PDET 指令用于释放设备阶段的所有权。

程序执行 PDET 指令后，*不再*拥有设备阶段的所有权。这样可释放设备阶段的所有权，以由其他程序或 FactoryTalk Batch 软件获取。仅当程序之前曾通过连接到设备阶段 (PATT) 指令获取设备阶段的所有权时，才能使用 PDET 指令。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

```
PDET(Phase_Name);
```

操作数

梯形图

操作数	类型	格式	说明
阶段名称 (Phase Name)	PHASE	设备阶段的名称	不再拥有所有权的设备阶段。

结构化文本

操作数与梯形图 PDET 指令的操作数相同。

影响数学状态标志

无

严重/轻微故障

无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

执行

在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	指令执行。

示例

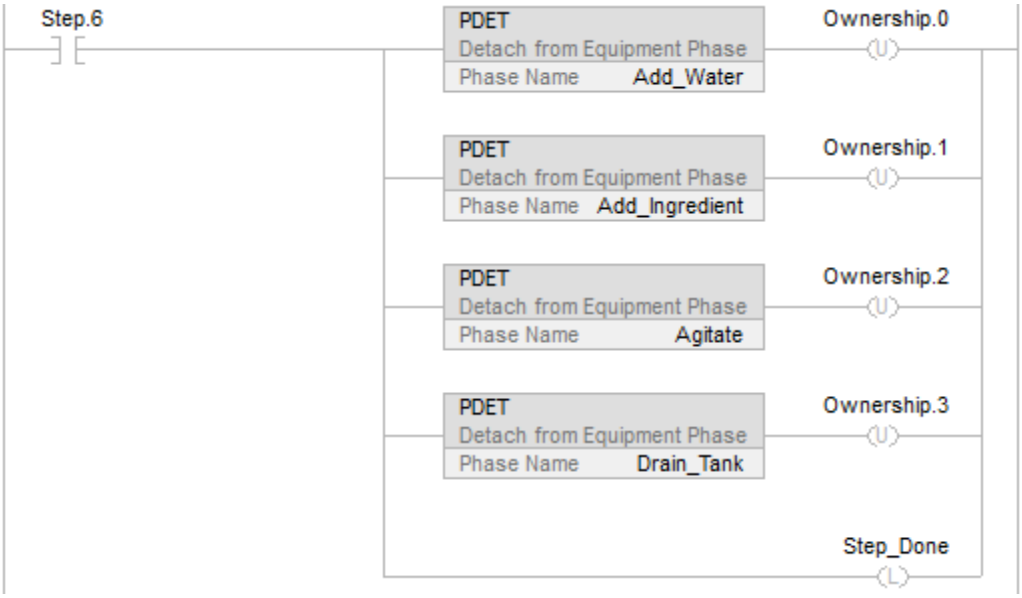
梯形图

如果 *Step.6* = 1（序列的第 6 步），则：

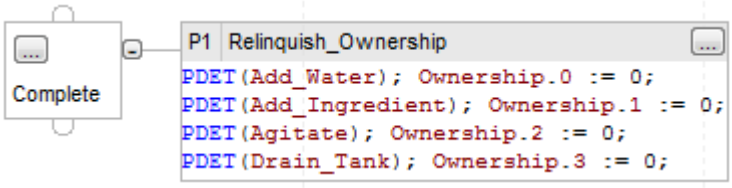
每条 PDET 指令都将交出序列对设备阶段的所有权。

各 *Ownership* 位均清 0。（在 *Ownership* 标签中，为每个设备阶段都分配了一个位。）

Done = 1。（这表示序列进入下一步。）



结构化文本



执行该序列时，Relinquish_Ownership 动作：

- 交出设备阶段所有权。
- 所有权标志（在取得设备阶段所有权后由 SFC 置位的位）清零。

如果使用 P1 类型的动作限定符，动作会仅在该步的首次扫描中执行。

另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

设备阶段清除故障 (PCLF)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PCLF 指令用于清除设备阶段的故障代码。

PCLF 指令用于清除设备阶段的故障代码。

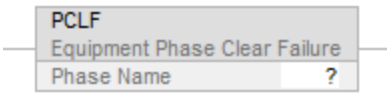
- 只有 PCLF 指令可以清除设备阶段的故障代码。
- CLR 指令、MOV 指令或赋值 (:=) 指令*不会*更改设备阶段的故障代码。

使用 PCLF 指令时，应确保设备阶段的所有权*未*由其他程序取得。如果 Logix Designer、HMI、FactoryTalk Batch 软件或其他程序拥有设备阶段的所有权，PCLF 指令将*无法*清除故障代码。

- 高优先级 HMI 所有权仅用于 CompactLogix 5370 和 ControlLogix 5570 控制器。

可用语言

梯形图



功能块

此指令不可用于功能块中。

结构化文本

PCLF(Phase_Name);

操作数

梯形图

操作数	类型	格式	说明
阶段名称 (Phase Name)	PHASE	设备阶段的名称	要清除故障代码的设备阶段。

结构化文本

操作数与梯形图 PCLF 指令的操作数相同。

影响数学状态标志

无

严重/轻微故障

无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

执行

在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	该指令按上述方式执行。

示例

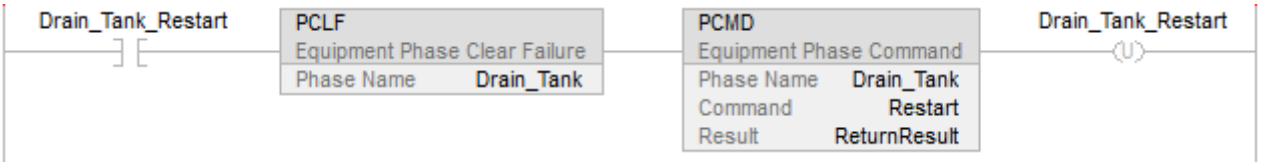
梯形图

如果 *Drain_Tank_Restart* = 1（重新启动 *Drain_Tank* 设备阶段），则

清除 *Drain_Tank* 设备阶段的故障代码

通过重新启动命令将 *Drain_Tank* 设备阶段的状态更改为重新启动。

Drain_Tank_Restart = 0;



结构化文本

（*如果 *Drain_Tank_Restart* = 1，则：

清除 *Drain_Tank* 设备阶段的故障代码。

重新启动 *Drain_Tank* 设备阶段。

将 *Drain_Tank_Restart* 清零。*）

If *Drain_Tank_Restart* Then

PCLF(*Drain_Tank*);

PCMD(*Drain_Tank*,Restart,0);

Drain_Tank_Restart := 0;

End_If;

另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

设备阶段命令 (PCMD)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

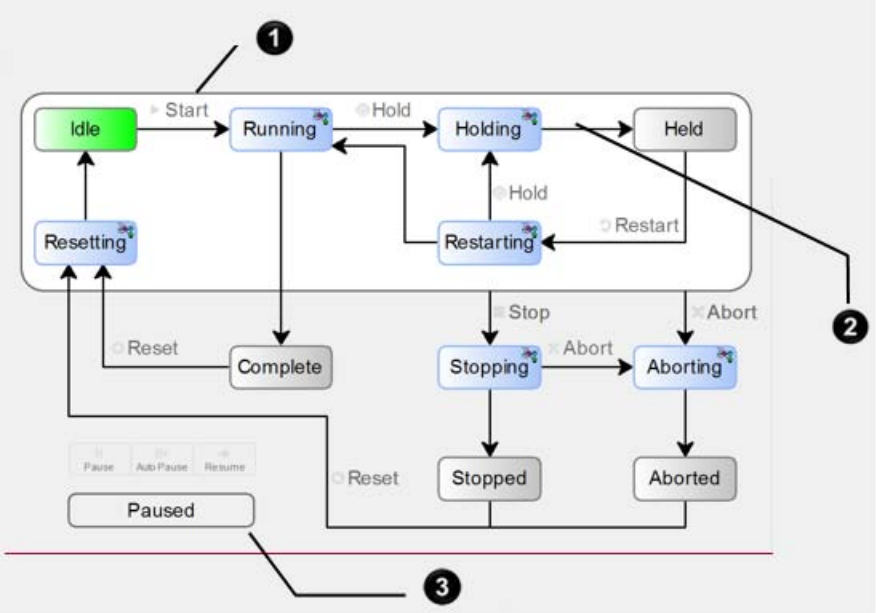
PCMD 指令将设备阶段转换为下一状态或子状态。

PCMD 指令用于更改设备阶段的状态或子状态。

在运行状态例程中，可使用 PSC 指令将设备阶段转换为完成状态。有关暂停功能的更多信息，请参见 PPD 阶段指令。

提示　要暂停设备阶段，必须使用 PPD 指令。

：



编号	说明
❶	命令 某些状态需要通过一条命令才能跳转到下一状态。如果设备阶段处于空闲状态，可通过启动命令将设备阶段转换为运行状态。一旦进入运行状态，设备阶段会执行其运行状态例程。
❷	其他状态将使用*阶段状态完成 (PSC)* 指令进入下一状态。如果设备阶段处于正在挂起状态，可使用 PSC 指令将设备阶段转换为已挂起状态。一旦进入已挂起状态，设备阶段需要执行重启命令才能进入重启状态。
❸	子状态 使用"自动暂停"、"暂停"和"恢复"来测试并调试状态例程。子状态需要通过*设备阶段暂停 (PPD)* 指令来生成逻辑程序断点。 使用"自动暂停"、"暂停"和"恢复"命令来逐步经过断点。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PCMD (PhaseName,Command,Result);

操作数

梯形图

操作数	类型	格式	说明
阶段名称 (Phase Name)	PHASE	设备阶段的名称	要更改为其他状态的设备阶段。
命令 (Command)	command	命令枚举值	要发送给设备阶段用来更改其状态的命令。要了解可用命令，请参见上图。
结果 (Result)	DINT	立即数 标签	为使指令返回指示其执行成功或失败的代码，需输入用于存储结果代码的 DINT 标签。否则，需输入 0。

结构化文本

操作数与梯形图 PCMD 指令的操作数相同。

使用 PCMD 指令的指导原则

指导原则	详细信息
将 PCMD 指令的执行限定于单次扫描。	将 PCMD 指令的执行限定于单次扫描。每条命令均适用于一种特定状态或多个状态。一旦设备阶段的状态发生变化，命令将*不再*有效。若要限制执行，可使用以下方法： • 在 P1 脉冲（上升沿）或 P0 脉冲（下降沿）动作中执行 PCMD 指令。 • 在 PCMD 指令前放置一个单脉冲触发指令。 • 执行 PCMD 指令，然后继续执行下一步。
确定是否需要取得设备阶段的所有权。	或者，可以由程序获取设备阶段的所有权。这可以防止其他程序或 FactoryTalk Batch 软件对设备阶段进行控制。

<table>
<tr><td></td><td>如果使用：</td><td>则：</td></tr>
<tr><td></td><td>FactoryTalk Batch 软件同时在此控制器内运行过程（配方）</td><td>同时在此控制器内运行过程（配方）。在使用 PCMD 指令前，使用连接到设备阶段 (PATT) 指令获取设备阶段的所有权。</td></tr>
<tr><td></td><td>多个程序对同一设备阶段进行控制</td><td>使用连接到设备阶段 (PATT) 指令获取设备阶段的所有权。</td></tr>
<tr><td></td><td>其他情况</td><td>无需获取设备阶段的所有权。</td></tr>
<tr><td>使用 POVR 指令，而非 PCMD 指令。</td><td colspan="2">1. 发出挂起、停止或中止命令？
• 否 - 使用 PCMD 指令。
• 是 - 转至步 2。
2. 即使设备阶段通过 Logix Designer 手动控制，也必须执行命令？
• 是 - 使用 POVR 指令。
• 否 - 转至步 3。
3. 即使 FactoryTalk Batch 软件或其他程序拥有设备阶段，也必须执行命令？
• 是 - 使用 POVR 指令。
• 否 - 使用 PCMD 指令。
例如，假设设备检查物料是否堵塞。如果发生堵塞，设备应中止运行。在这种情况下，使用 POVR 指令。这样，即使通过 Logix Designer 进行手动控制，设备也会中止。</td></tr>
<tr><td>确定是否需要返回代码。</td><td colspan="2">使用 Result 操作数获取表示 PCMD 指令成功或失败的代码。</td></tr>
<tr><td></td><td>若：</td><td>则在 Result 操作数中输入以下内容：</td></tr>
<tr><td></td><td>预计会出现所有权冲突或其他可能的错误</td><td>DINT 标签，用于存储指令执行结果的代码。</td></tr>
<tr><td></td><td>预计不会出现所有权冲突或其他可能的错误</td><td>0</td></tr>
<tr><td></td><td>要解读结果代码，请参见下文的结果代码表。</td><td></td></tr>
</table>

PCMD 结果代码

如果分配标签来存储 PCMD 指令的结果，则该指令在执行时会返回以下代码之一：

代码（十进制）	说明
0	成功命令。
24577	有效命令。
24578	对于设备阶段当前状态无效的命令。例如，如果设备阶段处于运行状态，则启动命令无效。
24579	无法控制设备阶段。以下任一项已取得设备阶段的所有权。 • Logix Designer • HMI • 外部定序程序（如 FactoryTalk Batch 软件） • 控制器中的其他程序
24594	计划外或禁用设备阶段，或者处于禁用任务中。

提示： 高优先级 HMI 所有权仅用于 CompactLogix 5370 和 ControlLogix 5570 控制器。

影响数学状态标志

无

严重/轻微故障

无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

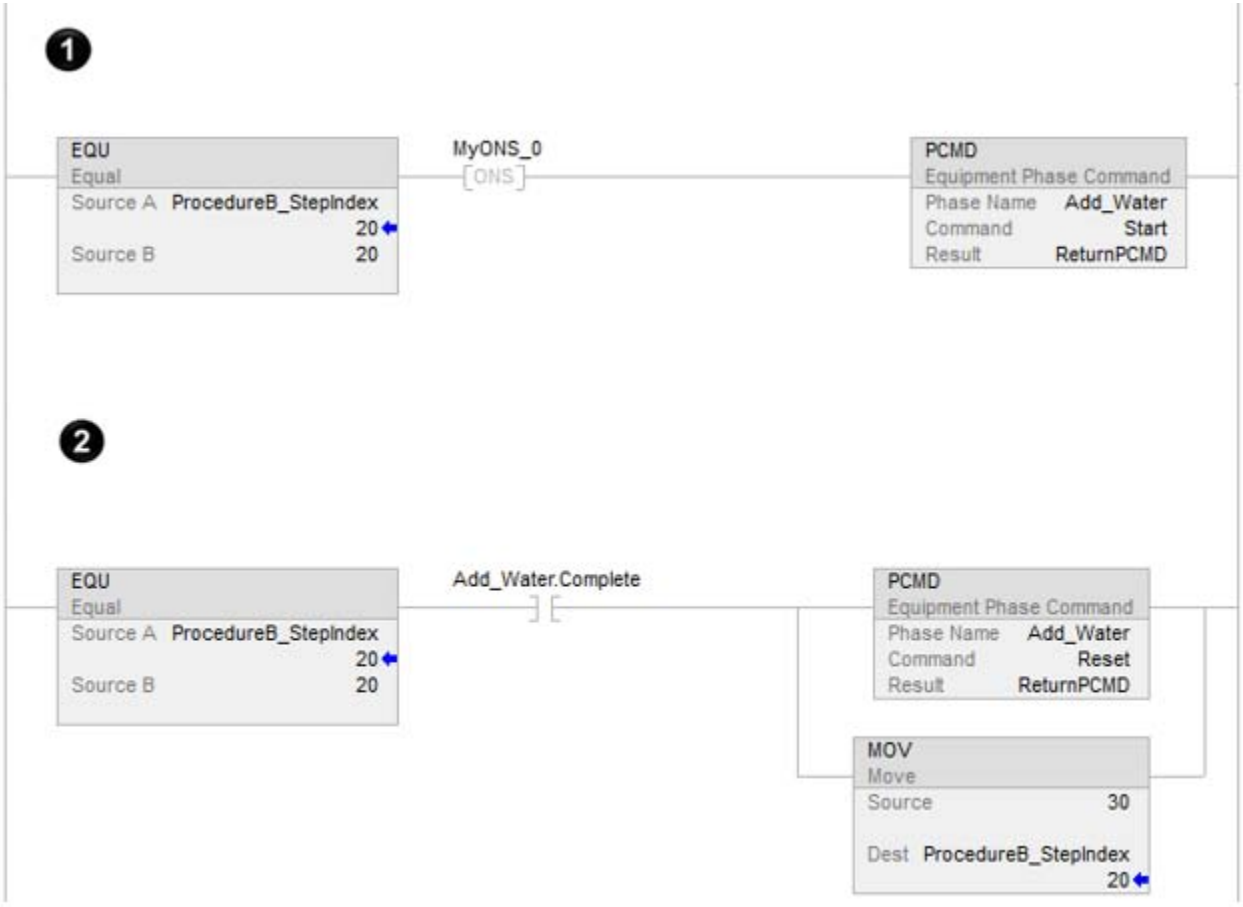
执行

在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。粗实线下方的所有条件仅能在普通扫描模式期间出现。

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	该指令按上述方式执行。

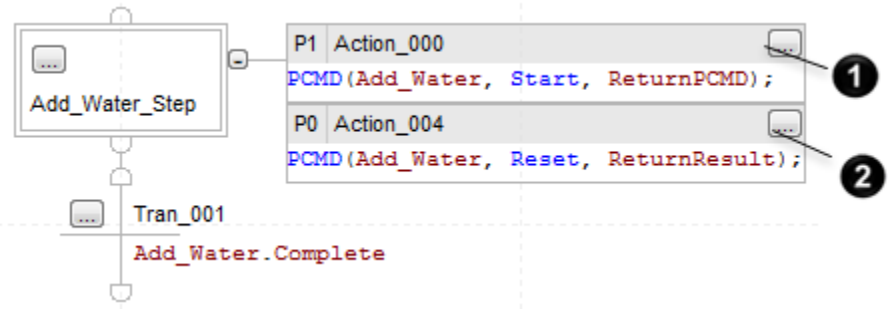
示例 1

梯形图



编号	说明
1	如果 ProcedureB_Stepindex = 20（例程处于步 20） 并且这是跳转到步 20（ONS 指令指示 EQU 指令由假跳变为真。） 则 通过启动命令将 Add_Water 设备阶段的状态更改为正在运行。
2	如果 ProcedureB_Stepindex = 20（例程处于步 20） 并且 Add_Water 设备阶段已完成 (Add_Water.Complete = 1) 则 通过复位命令将 Add_Water 设备阶段的状态更改为正在复位。转至步 30。

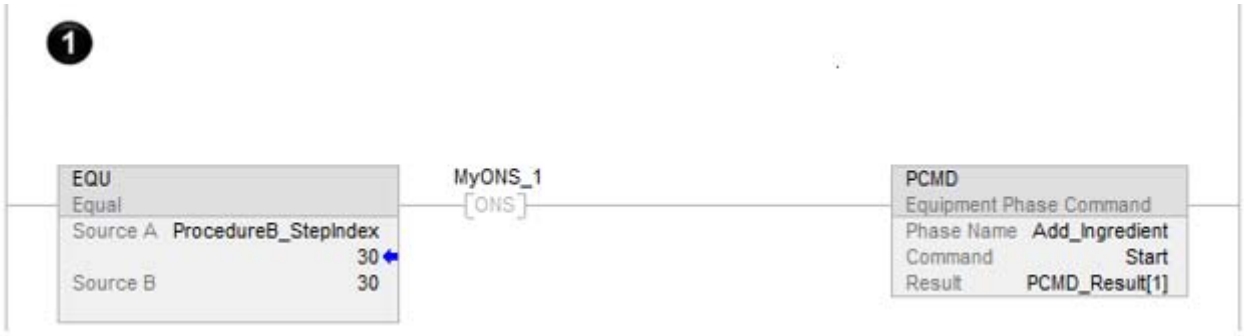
结构化文本



编号	说明
1	当 SFC 进入 *Add_Water_Step* 时，通过启动命令将 *ADD_Water* 设备阶段切换为正在运行，P1 限定符将此命令的执行限定于该步的首次扫描。
2	在 SFC 离开 *Add_Water_Step* 前 (*Add_Water_Complete* = 1)，通过复位命令将 *Add_Water* 设备阶段切换为正在复位状态。P0 限定符将此命令的执行限定于该步的末次扫描。

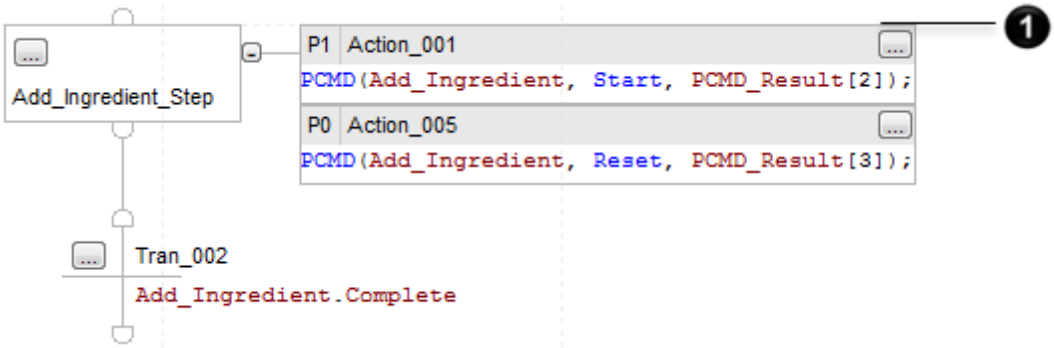
示例 2

梯形图



编号	说明
1	如果 ProcedureB_Stepindex = 30（例程处于步 30） 并且这是跳转到步 30（ONS 指令指示 EQU 指令由假跳变为真。） 则 通过启动命令将 Add_Water 设备阶段的状态更改为正在运行。 确认该命令执行成功，并将结果代码存入 PCMD_Result[1] [DINT 标签]。

结构化文本



编号	说明
1	当 SFC 进入 *Add_Ingredient_Step* 时， • 通过启动命令将 *Add_Ingredient* 设备阶段切换为正在运行状态。 • 确认该命令执行成功，并将结果代码存入 *PCDM_Result[2]*（DINT 标签）。 P1 限定符将此命令的执行限定于该步的首次扫描。

另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

设备阶段暂停（PPD） 参考页数 509

设备阶段外部请求 (PXRQ)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PXRQ 指令用于发起与 FactoryTalk® Batch 软件之间的通信。

重要事项： 将 PXRQ 指令用于 Equipment Sequence 时，仅支持全部下载 (1000) 请求以及全部上载 (2000) 请求。其他所有 PXRQ 指令请求均被忽略。

PXRQ 指令用于向 FactoryTalk Batch 软件发送请求。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PXRQ (Phase_Instruction, External_Request, Data_Value);

操作数

梯形图

操作数	类型	格式	说明
阶段指令 (Phase Instruction)	PHASE_INSTRUCTION	标签	用于控制操作的标签。
外部请求 (External Request)	请求	枚举值	请求类型。
数据值 (Data Value)	DINT	数组标签	请求的参数。

结构化文本

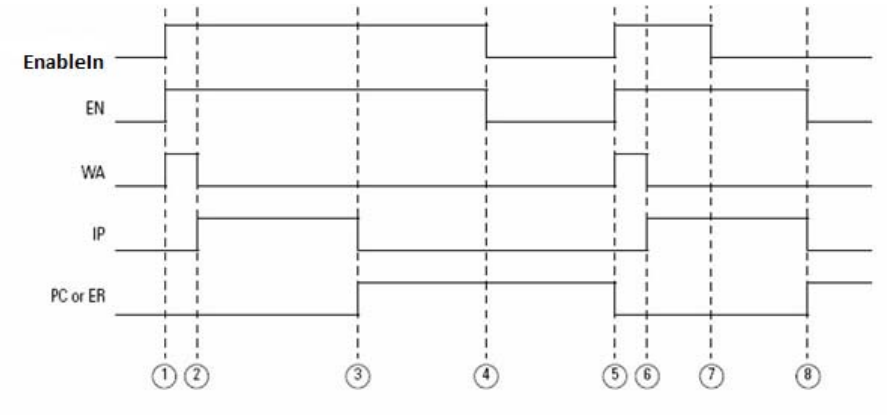
操作数与梯形图 PXRQ 指令的操作数相同。

PHASE_INSTRUCTION 数据类型

如需：	则检查或设置此成员：	数据类型	备注
确定假到真跳变是否导致指令执行	EN	BOOL	请参见下文的时序图。
确定请求是否失败	ER	BOOL	请参见下文的时序图。 若要诊断错误，请参见 ERR 和 EXERR 值。

确定 FactoryTalk Batch 软件是否已完成对请求的处理	PC	BOOL	请参见下文的时序图。	
确定 FactoryTalk Batch 软件是否正在处理请求	IP	BOOL	请参见下文的时序图。	
确定是否指令已发送请求但 FactoryTalk Batch 软件尚未应答	WA	BOOL	请参见下文的时序图。 在以下情况下 WA 也为 0： • 连接超时 • 发生网络错误 • ABORT = 1	
取消请求	ABORT	BOOL	若要中止(取消)请求 ,请将 ABORT 位设为 1。控制器中止指令时： • ER = 1 • ERR 显示中止的结果	
• 诊断错误原因 • 写入逻辑以响应特定错误	ERR	INT	如果 ER = 1，错误代码会提供诊断信息。要解读错误代码，请参见 *PXRQ 错误代码*部分。	
	EXERR	INT	如果 ER = 1，扩展错误代码会提供某些错误的额外诊断信息。要解读扩展错误代码，请参见 *PXRQ 错误代码*部分。	
对于标签的各状态位使用一个成员	STATUS	DINT	**对于以下成员：**	**使用以下位：**
			EN	31
			ER	28
			PC	27
			IP	26
			WA	25
			ABORT	24

时序图



使用 PXRQ 指令的指导原则

指导原则	详细信息
确保对"数据值"操作数使用数组。	"数据值"操作数需要使用 DINT 数组，即使该数组只包含 1 个元素（即，数据类型为 DINT[1]）也是如此。
在梯形图中，控制指令在发生跳变时执行。	此为跳变指令。每次执行指令时，都要将 EnableIn 由假跳变为真

在结构化文本中，使用结构来控制指令的执行。	以结构化文本对 PXRQ 指令进行编程时，请注意以下事项：
	• 在结构化文本中，指令会在*每次*被扫描时执行。
	• PXRQ 指令*仅*在被扫描时才会更新其状态位。
	• 为阻止指令重复执行但确保状态位更新，可将指令包含在满足以下条件的结构中：
	  • *仅*在发生跳变（条件发生变化）时发起指令的执行。
	  • 在 PC=1 或 ER=1 之前始终保持为真。

配置 PXRQ 指令

	按如下所述配置 PXRQ 指令：		
如需：	**外部请求**	**数据值数组元素**	**值**
下载所有输入参数	下载输入参数	DINT[0]	0
下载单个输入参数	下载输入参数	DINT[0]	参数 ID
下载一系列输入参数	下载输入参数	DINT[0]	第一个参数的参数 ID
		DINT[1]	待下载参数的数量

下载配置为开始或传输控制时自动下载的输入参数	下载输入参数子集	DINT[0]	启动 = 1 传输控制 = 2
下载所有输出参数	下载输出参数限制	DINT[0]	0
下载单个输出参数	下载输出参数限制	DINT[0]	参数 ID
上载所有报告	上载输出参数	DINT[0]	0
上载单份报告	上载输出参数	DINT[0]	报告 ID
上传一系列报告	上载输出参数	DINT[0]	第一份报告的报告 ID
		DINT[1]	待下载报告的数量
上载配置为在终止状态或传输控制时自动上载的输出参数	上载输出参数子集	DINT[0]	终止 = 1 传输控制 = 2
向操作员发送消息	向操作员发送消息	DINT[0]	消息 ID
清除来自操作员的消息	清除来自操作员的消息	DINT[0]	0
获取某一资源	获取资源	DINT[0]	设备 ID
获取多项资源	获取资源	DINT[0]	设备 ID
		DINT[1]	设备 ID
			...
释放某一资源	释放资源	DINT[0]	设备 ID
释放多项资源	释放资源	DINT[0]	设备 ID
		DINT[1]	设备 ID
			...
释放所有资源	释放资源	DINT[0]	0
向其他阶段发送消息（及可选数据）	向已连接阶段发送消息	DINT[0]	消息 ID
		DINT[1]	要接收消息的阶段的数量
		DINT[2]	消息值
		DINT[3]	消息值
			...

向其他阶段发送消息（及可选数据），并等待相应阶段接收消息	向已连接阶段发送消息并等待	DINT[0]	消息 ID
		DINT[1]	要接收消息的阶段的数量
		DINT[2]	消息值
		DINT[3]	消息值
			...
等待接收来自其他阶段的消息	接收来自已连接阶段的消息	DINT[0]	消息 ID
		DINT[1]	消息值
		DINT[2]	消息值
			...
取消向其他阶段发送的消息	取消向已连接阶段发送的消息	DINT[0]	消息 ID
取消向其他阶段发送的所有消息	取消向已连接阶段发送的消息	DINT[0]	0
下载客户批处理 ID	下载批处理数据	DINT[0]	1
		DINT[1]	用于存储值的参数 ID
下载唯一的批处理 ID	下载批处理数据	DINT[0]	2
		DINT[1]	用于存储值的参数 ID
下载阶段 ID	下载批处理数据	DINT[0]	3
		DINT[1]	用于存储值的参数 ID
下载配方控制与手动阶段控制	下载批处理数据	DINT[0]	4
		DINT[1]	用于存储值的参数 ID
下载阶段的当前模式	下载批处理数据	DINT[0]	5
		DINT[1]	用于存储值的参数 ID
下载输入参数的下限	下载批处理数据	DINT[0]	6 输入参数标签存储下限。

下载输入参数的上限	下载批处理数据	DINT[0]	7 输入参数标签存储上限。
下载与当前所用容器相关的数据	下载使用中的物料管理器数据容器	DINT[0]	1
		DINT[1]	属性 ID（指定单个属性）。
		DINT[2]	阶段参数 ID（指定用于存储所下载值的参数标签）
下载与当前所用容器中的当前物料相关的数据	下载使用中的物料管理器数据容器	DINT[0]	2
		DINT[1]	属性 ID（指定单个属性）。
		DINT[2]	阶段参数 ID（指定用于存储所下载值的参数标签）
下载与当前所用容器中的当前批次相关的数据	下载使用中的物料管理器数据容器	DINT[0]	3
		DINT[1]	属性 ID（指定单个属性）。
		DINT[2]	阶段参数 ID（指定用于存储所下载值的参数标签）
上载与当前所用容器相关的数据	上载使用中的物料管理器数据容器	DINT[0]	1
		DINT[1]	属性 ID（指定单个属性）。
		DINT[2]	阶段参数 ID（指定待上载值所在的参数标签）
上载与当前所用容器中的当前物料相关的数据	上载使用中的物料管理器数据容器	DINT[0]	2
		DINT[1]	属性 ID（指定单个属性）。
		DINT[2]	阶段参数 ID（指定待上载值所在的参数标签）
上载与当前所用容器中的当前批次相关的数据	上载使用中的物料管理器数据容器	DINT[0]	3
		DINT[1]	属性 ID（指定单个属性）。

		DINT[2]	阶段参数 ID（指定参数）
下载当前绑定的容器优先级	下载容器绑定优先级	DINT[0]	用于存储值的参数 ID
上载当前绑定的新容器优先级	上载容器绑定优先级	DINT[0]	包含值的参数 ID
下载有关物料是否充足的信息	下载充足物料	DINT[0]	用于存储值的参数 ID 在结果值中： • 0 = 物料不足 • 1 = 物料充足
生成签名	生成电子签名	DINT[0]	签名模板的 ID
		DINT[1]	
下载物料属性	下载物料管理器数据库数据	DINT[0]	0
		DINT[1]	阶段参数 ID（指定用于存储所下载值的参数标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
下载批次属性	下载物料管理器数据库数据	DINT[0]	1
		DINT[1]	阶段参数 ID（指定用于存储所下载值的参数标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
下载容器属性	下载物料管理器数据库数据	DINT[0]	3
		DINT[1]	阶段参数 ID（指定用于存储所下载值的参数标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
下载分配的容器优先级	下载物料管理器数	DINT[0]	4

	据库数据	DINT[1]	阶段参数 ID（指定用于存储所下载值的参数标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
		DINT[4]	
上载物料属性	上载物料管理器数据库数据	DINT[0]	5
		DINT[1]	阶段报告 ID（指定待上载值所在的阶段报告标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
上载批次属性	上载物料管理器数据库数据	DINT[0]	6
		DINT[1]	阶段报告 ID（指定待上载值所在的阶段报告标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
上载容器属性	上载物料管理器数据库数据	DINT[0]	8
		DINT[1]	阶段报告 ID（指定待上载值所在的阶段报告标签）
		DINT[2]	物料控制器 ID
		DINT[3]	属性 ID（指定单个属性）
下载分配的容器优先级	下载物料管理器数据库数据	DINT[0]	9
		DINT[1]	阶段参数 ID（指定用于存储所下载值的参数标签）
		DINT[2]	物料控制器 ID

		DINT[3]	属性 ID（指定单个属性）
		DINT[4]	

PXRQ 错误代码

ERR（十六进制）	EXERR（十六进制）	说明	建议的措施
00	0000	PXRQ 指令在将请求发送至 FactoryTalk Batch 软件之前已中止。	无
01	0000	PXRQ 指令在将请求发送至 FactoryTalk Batch 软件之后中止	无
02	0000	使用相同请求类型同时执行两条或更多 PXRQ 指令	限制一次执行一条 PXRQ 指令。
03	0110	通信错误。由于没有任何订阅者订阅该阶段，因此未发送请求	检查 FactoryTalk Batch 软件是否已连接并处于运行状态。
	0210	通信错误。由于与通知对象之间不存在任何连接，因此未发送请求。	
	0410	通信错误。发送失败。	
	1010	通信错误。由于 FactoryTalk Batch 软件未进行订阅，不能接收外部请求，因此未发送请求。	
	1020	FactoryTalk Batch 软件未与阶段相连。	
04	0002	FactoryTalk Batch 软件在处理请求时出现错误。	检查与 FactoryTalk Batch 软件的连接和通信路径。
	0003	PXRQ 指令包含无效值。	
	0004	FactoryTalk Batch 软件未处于处理请求的适当状态。	
	0005	使用不同请求类型同时执行两条或更多 PXRQ 指令	
	0006	在请求处理结束后，将结果存储至参数标签时出错。	

05	0000	FactoryTalk Batch 软件已接收到请求，但传回的 cookie 无效。	检查与 FactoryTalk Batch 软件的连接和通信路径。
06	0000	PXRQ 指令向 FactoryTalk Batch 软件发送的参数无效。	检查与 FactoryTalk Batch 软件的连接和通信路径。
07	0000	PXRQ 等待外部定序程序响应时通信断开。	检查与 FactoryTalk Batch 软件的连接和通信路径。

影响数学状态标志

无

严重/轻微故障

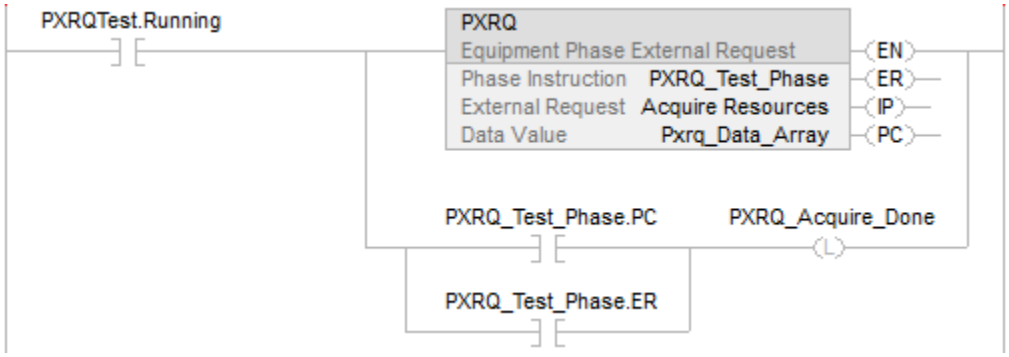
无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

执行

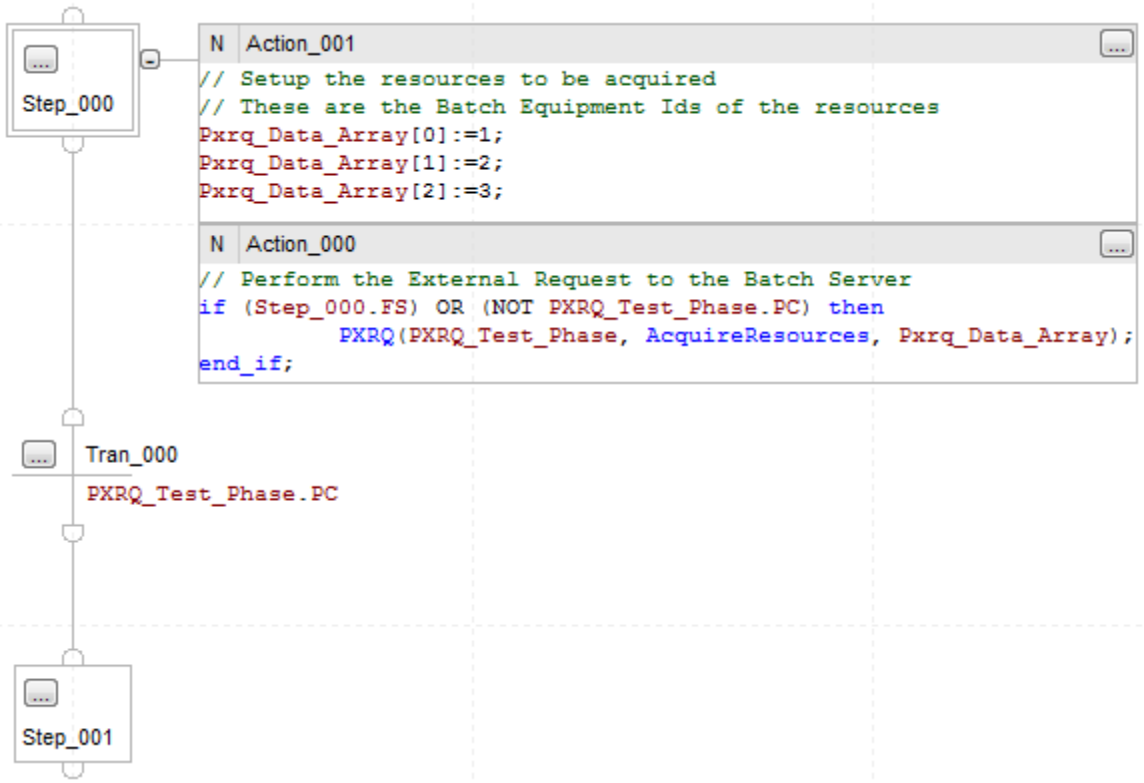
条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	指令执行。

示例

梯形图



结构化文本



另请参见

设备阶段指令 参考页数 467

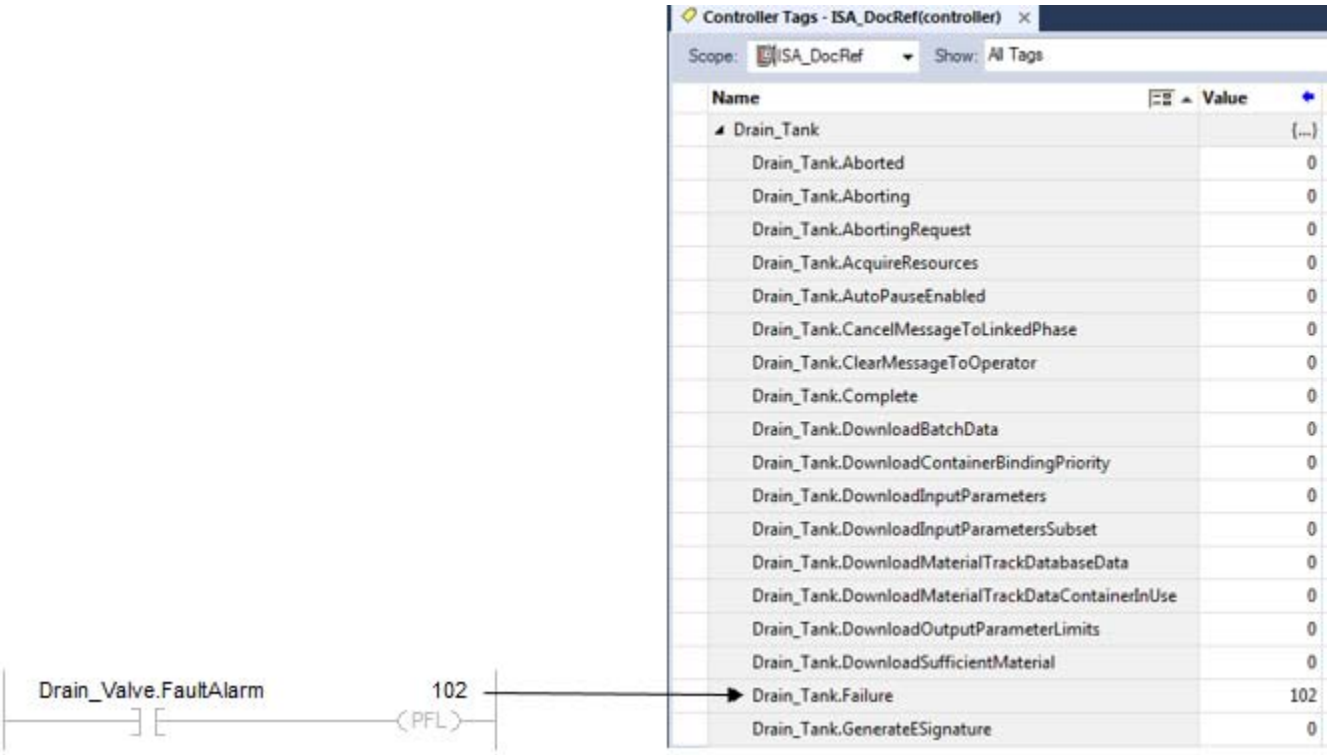
数组索引编制 参考页数 602

设备阶段故障 (PFL)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PFL 指令可用作指示设备阶段存在故障的可选方法。

PFL 指令用于设置设备阶段故障代码的值。此指令用于指示设备阶段的具体故障，如特定设备出现的故障。PFL 指令仅会将故障代码设置为大于当前值的值。



可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PFL (Failure_Code);

操作数

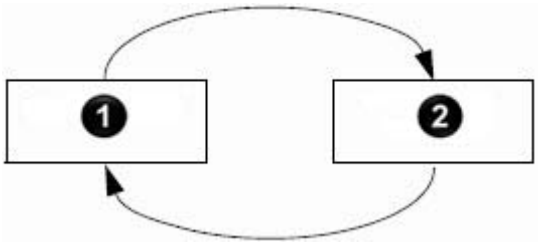
梯形图

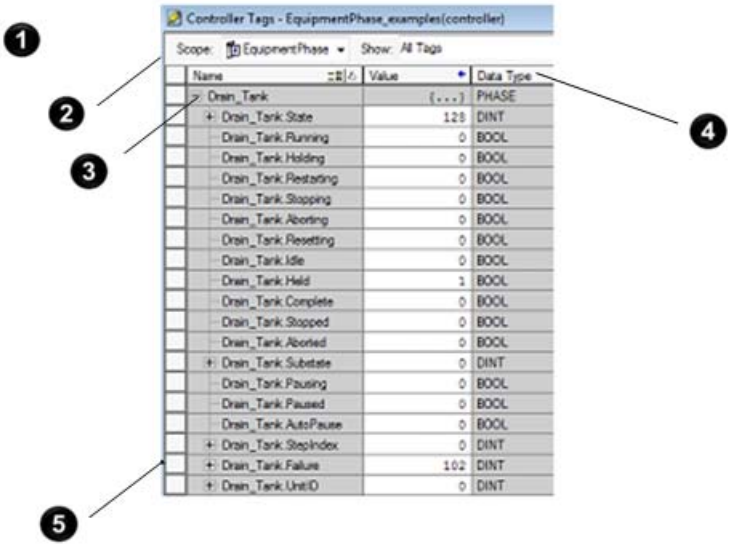
操作数	类型	格式	说明
Failure_Code	DINT	立即数 标签	希望设置的设备阶段故障代码的值。 如果给出的故障代码为负 ,则将其视为 0。

结构化文本

操作数与梯形图 PFL 指令的操作数相同。

使用 PFL 指令的指导原则

指导原则	详细信息
将 PFL 指令置于设备阶段中。	PFL 指令会为其所处的设备阶段设置故障代码。*不存*在用于识别特定设备阶段的操作数。 通常，将 PFL 指令置于设备阶段的预状态例程中。 • 控制器始终会扫描预状态例程 ,即使设备阶段处于空闲状态也是如此。 • 控制器在*每次*对状态进行扫描之前都会扫描预状态例程。  **编号** / **说明** 1 / 预状态例程 2 / 当前状态例程 使用进度例程持续监视设备阶段在经历各个状态时的健康状况。
确定故障代码的优先级。	PFL 指令仅会将故障代码设置为大于当前值的值。 • 例如 ,如果 PFL 指令设置的故障代码为 102 ,则其他 PFL 指令仅能设置 > 102 的故障代码。 • 确保为需要更高处理优先级的异常分配更大的值。否则，优先级较低的异常可能会覆盖更为严重的异常。

<table>
<tr>
<td>要在发生故障时采取相应措施，可监视 PHASE 标签的 Failure 成员。</td>
<td>PFL 指令将其值写入设备阶段 PHASE 标签的 Failure 成员。


<table>
<tr><th>编号</th><th>说明</th></tr>
<tr><td>1</td><td>创建设备阶段时，Logix Designer 应用程序会为设备阶段的状态创建一个标签。</td></tr>
<tr><td>2</td><td>控制器作用域</td></tr>
<tr><td>3</td><td>名称 = phase_name</td></tr>
<tr><td>4</td><td>PHASE 数据类型</td></tr>
<tr><td>5</td><td>PFL 指令将其值写入设备阶段的故障成员。</td></tr>
</table>
</td>
</tr>
<tr>
<td>要清除故障代码，可使用 PCLF 指令。</td>
<td>可使用 PCLF 指令清除设备阶段的故障代码。CLR 或 MOV 等指令不会更改故障代码。</td>
</tr>
</table>

影响数学状态标志

编号

严重/轻微故障

在以下情况下会发生严重故障：	故障类型	故障代码
从某个设备阶段程序的外部调用指令。	4	91

请参见下文的*数组索引编制*部分，了解关于数组索引故障的信息。

执行

在结构化文本中，EnableIn 在普通扫描期间始终为真。因此，如果指令处于由逻辑激活的控制路径中，指令将会执行。粗实线下方的所有状况仅能在普通扫描模式期间出现。

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	该指令按上述方式执行。

示例

梯形图

在设备阶段的预状态例程中...

如果 *Drain_Valve.FaultAlarm* = 1（阀未进入指定状态），则：

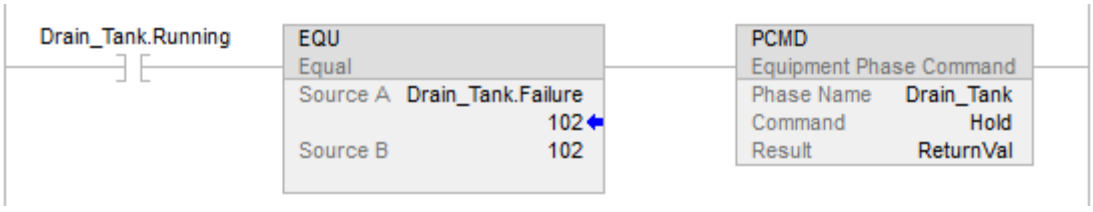
设备阶段的故障代码为 102。

如果 *Drain_Tank.Running* = 1（*Drain_Tank* 设备阶段处于运行状态。）

且 *Drain_Tank.Failure* = 102（设备阶段的故障代码）

则

通过挂起命令将 *Drain_Tank* 设备阶段的状态更改为正在挂起。



结构化文本

在设备阶段的预状态例程中...

```
(*If the drain valve does not go to the commanded state, then set the
failure code of this equipment phase = 102.*)
If Drain_Valve.FaultAlarm Then
    PFL(102);
End_If;

(*If the Drain_Tank equipment phase = running and its failure code = 102,
issue the hold command and send the equipment phase to the holding state.*)
If Drain_Tank.Running And (Drain_Tank.Failure = 102) Then
    PCMD(Drain_Tank,hold,0);
End_IF;
```

另请参见

设备阶段指令 参考页数 467

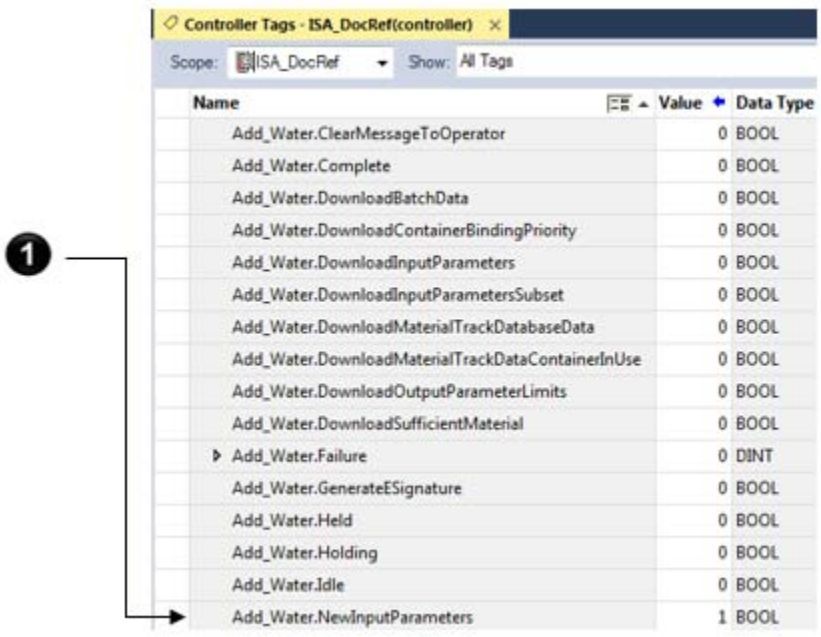
数组索引编制 参考页数 602

设备阶段新参数 (PRNP)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PRNP 指令用于将设备阶段的 NewInputParameters 位清零。

PRNP 指令会将设备阶段的 NewInputParameters 位清零。



编号	说明
❶	当 FactoryTalk Batch 软件有新参数可用于某个设备阶段时，会将该阶段的 NewInputParameters 位置位。 下载参数后，可使用 PRNP 指令将该位清零。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PRNP ( );

操作数

梯形图

无

结构化文本

无

在指令助记符后输入括号 ( )，即使没有操作数也是如此。

影响数学状态标志

无

严重/轻微故障

无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

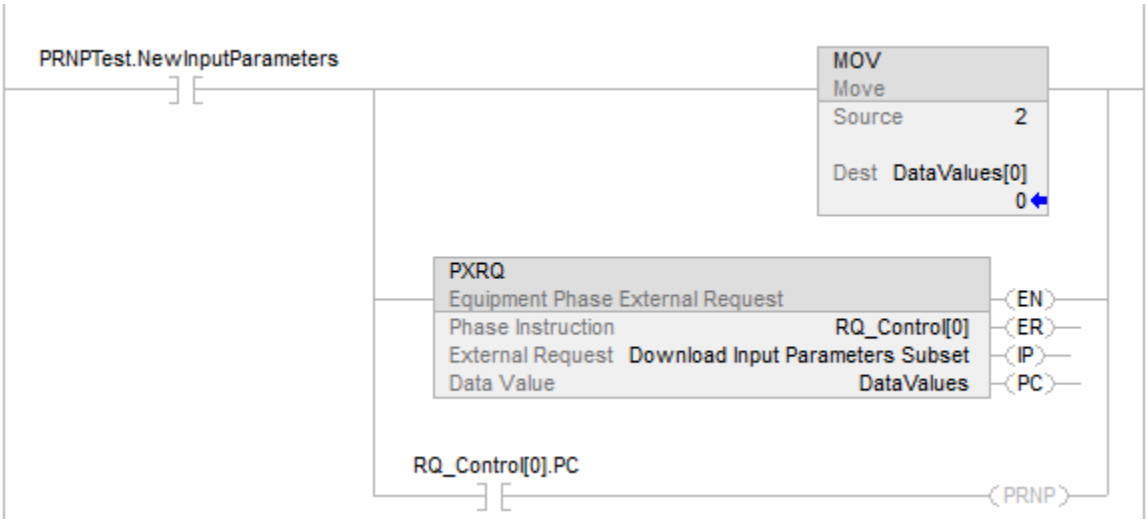
执行

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。

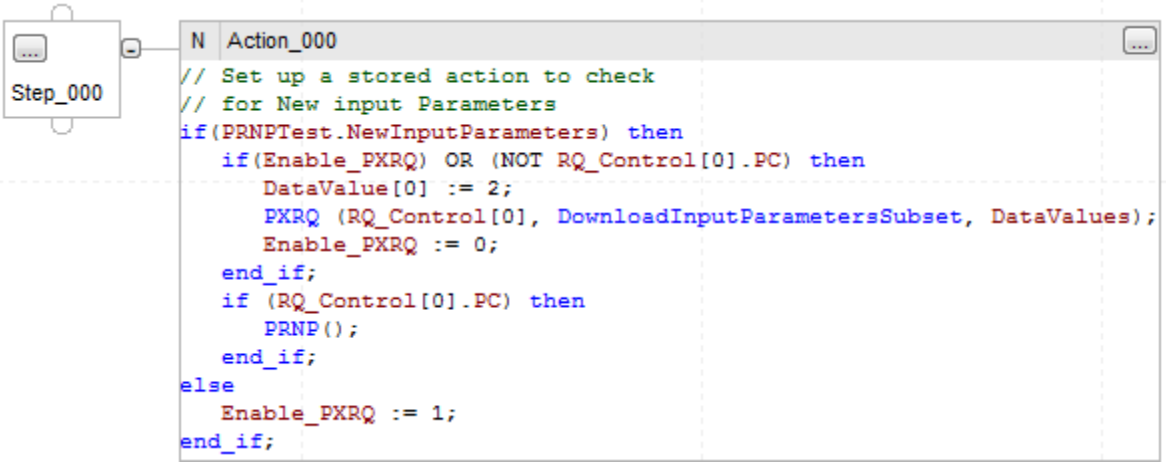
EnableIn 为假	不执行任何操作。
EnableIn 为真	指令执行。

示例

梯形图



结构化文本



如果 PRNPTest.NewInputParameters = 1（FactoryTalk Batch 软件有新输入参数可用于设备阶段），则

如果 Enable_PXRQ = 1（执行 PXRQ 指令。）

或者 RQ_Control[0].PC = 0（PXRQ 指令正在执行），则

Datavalues[0] = 设置 PXRQ 指令，以传输控制。
发送“下载输入参数子集”请求到 FactoryTalk Batch 软件。

发送 DataValues[0] = 2，该指令设置用于传输控制。
Enable_PXRQ = 0（请求完成后不重启 PXRQ 指令）

如果 RQ_Control[0].PC = 1（请求完成），则

通过 PRNP 指令设置 ThisPhase.NewInputParameters = 0。

否则

Enable_PXRQ = 1（下一次有新输入参数可用时，执行 PXRQ 指令。）

另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

设备阶段超控命令 (POVR)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

无论所有权如何，均可使用 POVR 命令向设备阶段发出 Hold、Stop 或 Abort 命令。

POVR 指令：

- 向设备阶段发出 Hold、Stop 或 Abort 命令。
- 优先于设备阶段的全部所有者。即使 Logix Designer 软件、HMI、FactoryTalk Batch 软件或其他程序已拥有设备阶段的所有权，该命令也会执行。该指令不会更改设备阶段的所有权。
- 高优先级 HMI 所有权仅用于 CompactLogix 5370 和 ControlLogix 5570 控制器。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

POVR (PhaseName, Command, Result);

操作数

梯形图

操作数	类型	格式	说明
阶段名称 (Phase Name)	阶段	设备阶段的名称	要更改为其他状态的设备阶段
命令 (Command)	command	命令名称	设备阶段的以下命令之一： • 挂起 • 停止 • 中止
结果 (Result)	DINT	立即数 标签	为使指令返回指示其执行成功或失败的代码，需输入用于存储结果代码的 DINT 标签。否则，需输入 0。

结构化文本

操作数与梯形图 POVR 指令的操作数相同。

使用 POVR 指令的指导原则

指导原则	详细信息
如果希望优先于其他所有者。	即使通过 Logix Designer 软件手动控制设备阶段，也希望设备挂起、停止或中止？ • 是 - 使用 POVR 指令 • 否 - 使用 PCMD 指令 这也适用于 HMI、FactoryTalk Batch 软件或其他程序。要在不考虑所有权归属的情况下挂起、停止或中止，必须使用 POVR。 例如，假设设备检查物料是否堵塞。如果出现堵塞，则中止设备。在这种情况下，使用 POVR 指令。这样，即使通过 Logix Designer 软件进行手动控制，设备也会中止。
将 POVR 指令的执行限制为单次扫描。	将 POVR 指令的执行限制为单次扫描。每条命令均适用于一种特定状态或多个状态。一旦设备阶段的状态发生变化，命令将*不再*有效。若要限制执行，可使用以下方法： • 在 P1 脉冲（上升沿）或 P0 脉冲（下降沿）动作中执行 POVR 指令。 • 在 POVR 指令前放置一条单脉冲触发指令。 • 执行 POVR 指令，然后继续执行下一步。

POVR 结果代码

如果分配标签来存储 POVR 指令的结果，则该指令在执行时会返回以下代码之一：

代码（十进制）	说明
0	成功命令。
24577	无效命令。
24578	对于设备阶段当前状态无效的命令。例如，如果设备阶段处于停止状态，则保持命令无效。
24594	计划外或禁用设备阶段，或者处于禁用任务中。

影响数学状态标志

无

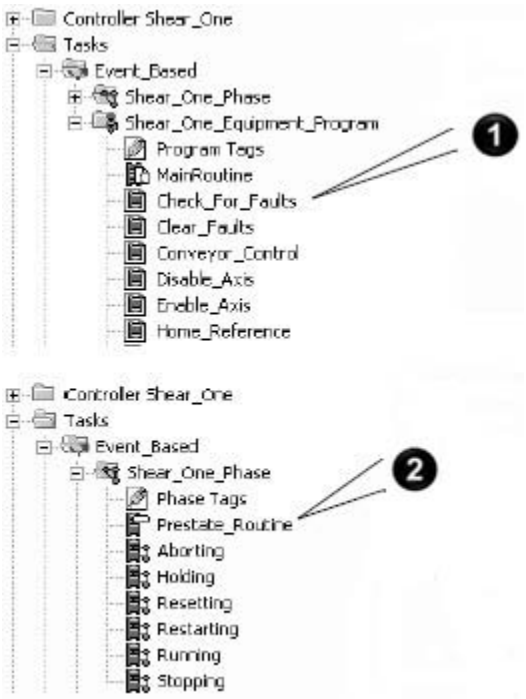
严重/轻微故障

无。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

执行

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	指令执行。

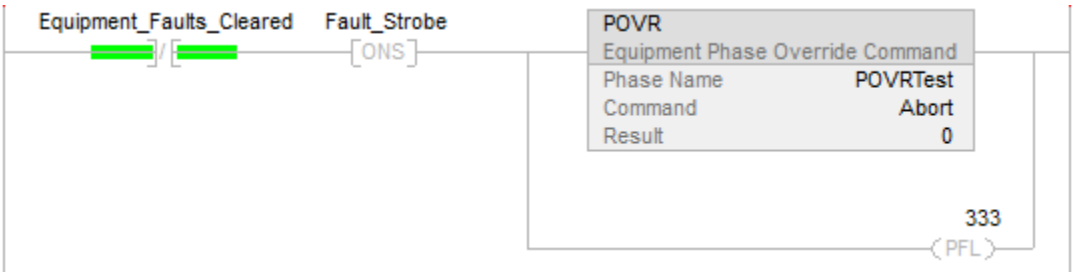
示例



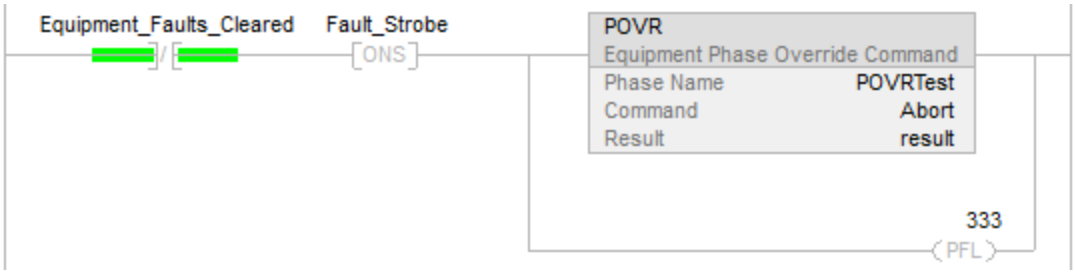
编号	说明
❶	设备程序检查以下故障： • 轴故障 • 物料堵塞 如果存在故障，则 *Local_Interface.Equipment_Faults_Cleared* = 0。此标签是控制器作用域标签 *Shear_1* 的别名。

编号	说明
❷	设备阶段的预状态例程将检查设备程序是否发出故障指示。 • 如果 *Interface_To_Equipment.Equipment_Faults_Cleared*=0，则说明存在故障。 • *Interface_To_Equipment* 和 *Local_Interface* 均作为 *Shear_1* 的别名，因此必须具有相同值。 如果存在故障，则 向 *Shear_One_Phase* 设备阶段发出中止命令。POVR 指令可确保命令即使在有人已通过 Logix Designer 软件手动控制设备阶段的情况下仍会执行。 PFL 指令设置故障代码 *Shear_One_Phase*=*333*。 *Fault_Strobe* 在单次扫描中执行这些动作。

梯形图



示例 2



结构化文本

```
If NOT Equipment_Faults_Cleared And NOT Fault_Strobe then
    POVR(POVRTest,Abort, 0);
    PFL(333);
end_if;
Fault_Strobe := NOT Equipment_Faults_Cleared;
```

另请参见

设备阶段指令 参考页数 467

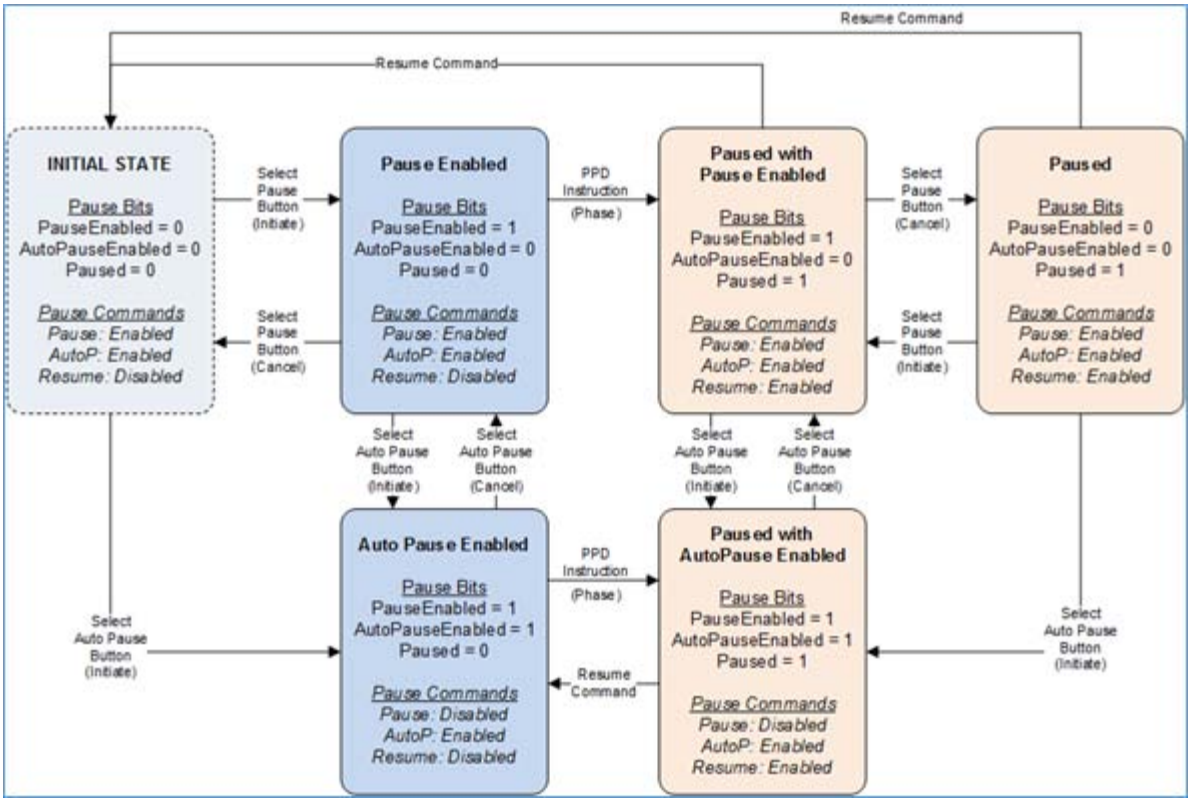
数组索引编制 参考页数 602

设备阶段暂停 (PPD)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PPD 指令用于在设备阶段的逻辑中设置断点。

若要使设备阶段暂停，可在设备阶段的状态例程逻辑中加入 PPD 指令，以此配置断点。PPD 指令执行，并且设备阶段收到下次暂停的命令时，设备阶段状态例程会暂停。



与暂停功能相关的操作员命令有三个：

- **暂停：**暂停命令用于启用或禁用在执行下一条 PPD 指令时暂停执行阶段。暂停命令会将 PauseEnabled 位切换为 ON (1) 或 OFF (0)。
- **自动暂停：**自动暂停命令用于启用或禁用在处理恢复命令后自动启用设备阶段暂停。自动暂停命令会将 AutoPauseEnabled 位切换为 ON (1) 或 OFF (0)。
- **恢复：**恢复命令用于控制固件恢复阶段状态例程逻辑的执行。恢复命令会将 PauseEnabled 和 Paused 位设置为 OFF (0)。

暂停子状态使用以下三个位：

- **PauseEnabled**：PauseEnabled 位用于存储处理暂停命令的状态。此为暂停子状态的位 0。

  当 Paused 位为 ON (1) 时，若执行 PPD 指令，将暂停执行状态例程逻辑。此位在接收到暂停命令时更新（切换此位的值）。此外，如果 AutoPauseEnabled 位为 ON (1)，则恢复命令会将 PauseEnabled 设为 OFF (0)。

- **AutoPauseEnabled**：AutoPauseEnabled 位用于存储在恢复命令后立即自动启用暂停的状态。此为暂停子状态的位 2。

  此位在接收到自动暂停命令时更新（切换此位的值）。当 AutoPauseEnabled 为 ON (1) 且阶段已暂停时，恢复命令会将 PauseEnabled 保持为 ON (1)。

- **Paused**：Paused 位用于存储阶段的暂停状态，即已暂停 (1) 或未暂停 (0)。此为暂停子状态的位 1。Paused 位还会禁用其余梯级 (RLL)，它不会终止或暂停例程的执行。

  该位仅能通过阶段的固件置位。当 PauseEnabled 位为 ON 时，若执行 PPD 指令，Paused 位会设置为已暂停 (1)，固件会暂停阶段状态例程的执行。恢复命令会将 Paused 位设为未暂停 (0)，阶段会继续执行其逻辑。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PPD( );

操作数

梯形图

无

结构化文本

无

在指令助记符后输入括号 ()，即使没有操作数也是如此。

使用 PPD 指令的指导原则

指导原则	详细信息	
将逻辑按一系列步组织起来。	如果按定义的步（如状态机或 SFC）组织逻辑，PPD 指令（断点）最易于使用。 • 断点*仅*表示特定条件已满足，而*不会*停止设备阶段的执行。 • 若要使逻辑在某个断点处真正中断（暂停），在组织逻辑时，应使逻辑保持在断点发生的步，直至收到恢复命令。	
请勿将 PPD 指令用作例程的临时终止点。	即使设备阶段暂停，它仍会继续执行其所有逻辑。 • 当 PPD 指令执行时，只会将设备阶段的 Paused 位置位。 • 如果以 RLL 对 PPD 指令进行编程，它将仅禁用其梯级上的其余逻辑。而*不会*终止或暂停例程的执行。 • 将 PPD 指令视为可根据自动暂停和暂停命令应用或忽略的条件。	
使用 PPD 指令来在多次扫描的同一断点处暂停。	当 PauseEnabled 位为真时，设备阶段将在对应条件为真的第一条 PPD 指令处暂停。如果 PPD 指令在多次扫描中执行 ,则设备阶段可能会在同一断点持续频繁暂停。	
确保一次只有 1 条 PPD 指令为真。	PPD 指令*不*包含用于记忆该指令是否已执行的控制标签。 • 只要该指令的条件为真（且设备阶段处于子阶段，PauseEnabled 为真）时，PPD 指令就会起到断点的作用（并通过禁用其余梯级逻辑来暂停该阶段）。 • 将逻辑限制为一次只有一个可行的断点，可以确保在所需断点处暂停。	
选择要使用的子状态。	PPD 指令（断点）仅在设备阶段 PauseEnabled 位为真时才有效。	
	在以下位置暂停：	**发出以下命令：**
	条件为真的每个断点	Auto Pause
	条件为真的第一个断点	Pause

影响数学状态标志

无

严重/轻微故障

以下情况下会发生严重故障：	故障类型	故障代码
从某个设备阶段程序的外部调用指令。	4	91

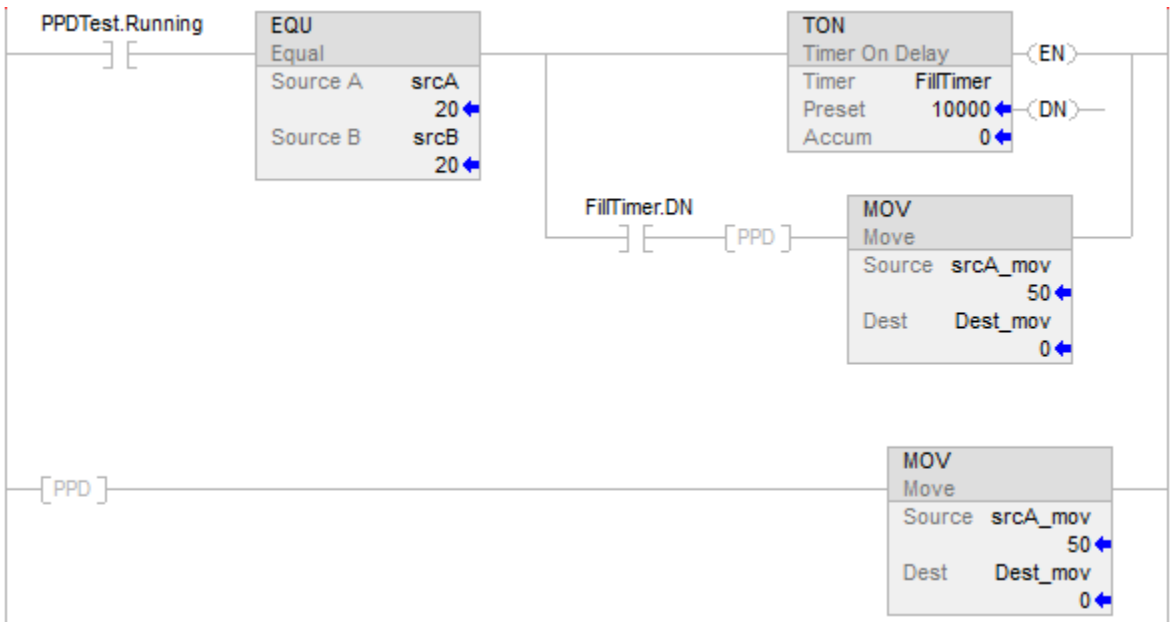
如果 Add-On 自定义指令使用 PPD 指令，并且一个非设备阶段程序调用该 Add-On 自定义指令，Logix Designer 会发出警告。检查 Add-On 自定义指令中是否有该指令，并将其禁用。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

执行

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	指令执行。

示例

梯形图



另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

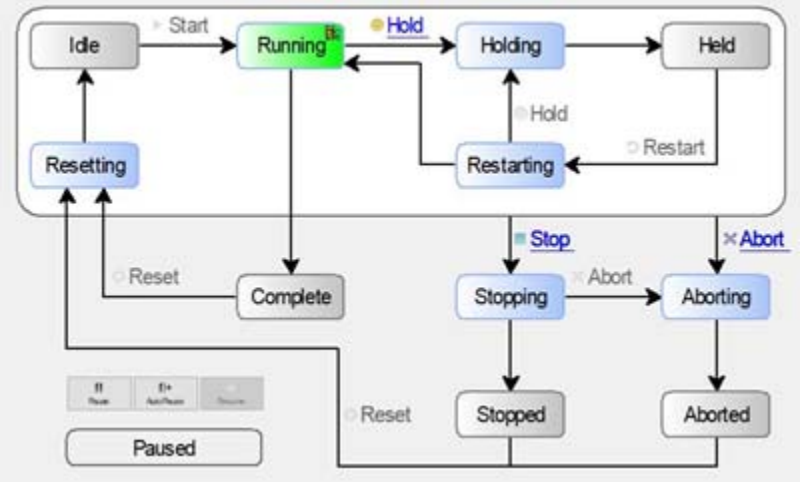
设备阶段命令 参考页数 478

阶段状态完成 (PSC)

该指令适用于 CompactLogix 5370 和 CompactLogix 5380、ControlLogix 5570 和 ControlLogix 5580、Compact GuardLogix 5370 和 Compact GuardLogix 5380 控制器。

PSC 指令用于向设备阶段指示状态例程已完成，进而指示设备阶段应进入下一状态。

PSC 指令用于指示阶段状态例程已完成。



在运行状态例程中，可使用 PSC 指令将设备阶段转换为完成状态。

可用语言

梯形图

功能块

此指令不可用于功能块中。

结构化文本

PSC( );

操作数

梯形图

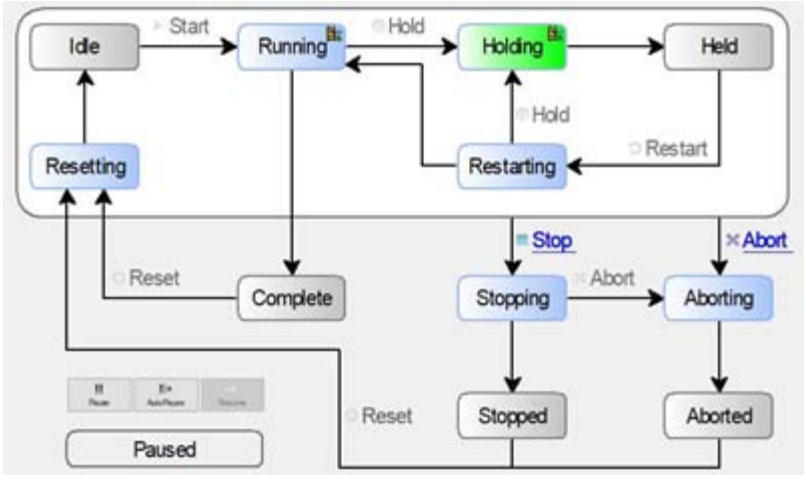
无

结构化文本

无

在指令助记符后输入括号 ()，即使没有操作数也是如此。

使用 PSC 指令的指导原则

指导原则	详细信息
将 PSC 指令用在添加至设备阶段的*每个*阶段状态例程中。	若不使用 PSC 指令，设备阶段将保持在当前状态下，*不*会进入下一状态。 • 将 PSC 指令阶段状态例程的最后一步。 • 状态完成后，执行 PSC 指令。
	在保持状态例程中，使用 PSC 指令可使设备阶段进入保持状态
请记住，PSC 指令*不*会停止例程的当前扫描。	PSC 指令执行时，控制器会扫描例程的其余部分，然后将设备阶段转换为下一状态。PSC 指令不会终止例程的执行。
*请勿*在预状态例程中使用 PSC 指令。	PSC 指令仅用于指示从一个状态至另一个状态的转换。

影响数学状态标志

无

严重/轻微故障

在以下情况下会发生严重故障：	故障类型	故障代码
从某个设备阶段程序的外部调用指令。	4	91

如果 Add-On 自定义指令使用 PSC 指令，并且一个非设备阶段程序调用该 Add-On 自定义指令，Logix Designer 会发出警告。检查 Add-On 自定义指令中是否有该指令，并将其禁用。请参见下文的*数组索引编制*，了解关于操作数故障的信息。

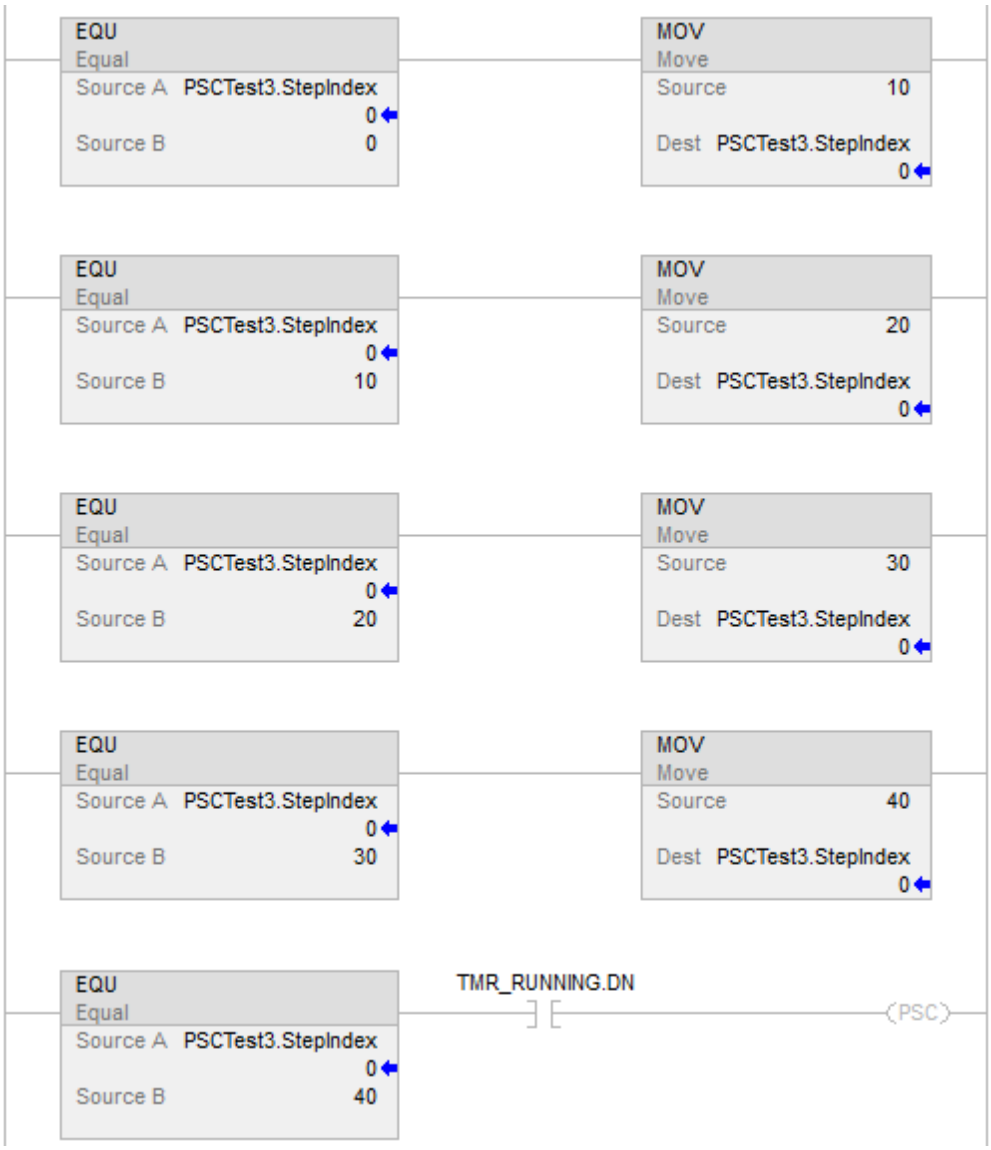
执行

在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构，或者同时使用两者。

条件/状态	执行的操作
预扫描	不执行任何操作。
后扫描	不执行任何操作。
EnableIn 为假	不执行任何操作。
EnableIn 为真	指令执行。

示例

梯形图



结构化文本

If TagEnableRunning

And PSCTest.Running Then

PSC();

End_if;

另请参见

设备阶段指令 参考页数 467

数组索引编制 参考页数 602

设备阶段超控命令 (POVR) 参考页数 504

设备顺序

设备顺序指令

下表列出了 Equipment Sequence 的命令指令。这些指令可用于使用梯形图和结构化文本编程语言的例程。它们不可用于使用功能块和顺序功能图编程语言的例程。

指令	说明	结构化文本格式
连接到设备顺序 (SATT)	请求用户例程的父程序拥有 Equipment Sequence 的所有权。	SequenceName, Result
从设备顺序断开 (SDET)	释放 Equipment Sequence 的所有权。	SequenceName
设备顺序命令 (SCMD)	向 Equipment Sequence 发送命令。	SequenceName, Command, Result
设备顺序清除故障 (SCLF)	清除 Equipment Sequence 的故障代码。	SequenceName
设备顺序超控 (SOVR)	无论所有权如何，均将 HOLD、STOP 或 ABORT 命令发送到 Equipment Sequence。	SequenceName, Command, Result
设备顺序分配顺序标识符 (SASI)	向 Equipment Sequence 分配 ID 字符串。	SequenceName, Sequence ID, Result

另请参见

设备顺序图指令 参考页数 530

SATT 参考页数 519、SDET 参考页数 522

SCMD 参考页数 528、SCLF 参考页数 525

SOVR 参考页数 531、SASI 参考页数 523

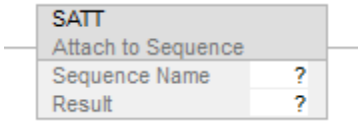
连接到设备顺序 (SATT)

连接到 Equipment Sequence (SATT) 指令用于获取 Equipment Sequence 的所有权。Equipment Sequence 的所有权必须由程序拥有，然后程序才能对其进行控制。必须分配一个标签来存储 SATT 指令的结果代码。

SATT 指令会返回以下五个结果代码之一。结果代码 **0** 表示 SATT 指令已成功执行。其他四个代码表示指令未成功执行，并提供与指令执行失败的原因相关的附加信息。

可用语言

梯形图



功能块

此语言不适用于此指令。

结构化文本：

SATT(SequenceName,Result)

操作数

SATT 指令使用以下操作数。

操作数	类型	格式	说明
Sequence Name	Sequence	Equipment Sequence 的名称	要获取所有权（进行连接）以进行控制的 Equipment Sequence（例如，Make_Product_101）。
Result	DINT	立即数 标签	为使指令返回指示其执行成功或失败的代码，需输入用于存储结果代码的 DINT 标签。否则，需输入 **0**。

影响数学状态标志

否

严重/轻微故障

SATT 指令不会触发故障，因此此指令不存在故障条件。

执行

下表介绍 SATT 指令的执行步骤。

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。

条件	执行的操作
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	• 指令执行。 • 梯级输出条件设置为真。
扫描结构化文本	不执行任何操作
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	梯级输出条件设置为假。

结构化文本

条件	执行的操作
预扫描	不执行任何操作。
梯级输入条件为假	不执行任何操作。
梯级输入条件为真	不执行任何操作。
扫描结构化文本	在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构。
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	不执行任何操作。

另请参见

SATT 指令指导原则 参考页数 533

SATT 指令的结果代码 参考页数 536

SATT 指令示例 参考页数 540

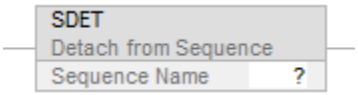
设备顺序指令 参考页数 519

从设备顺序断开 (SDET)

从 Equipment Sequence 断开 (SDET) 指令用于释放 Equipment Sequence 的所有权。程序执行 SDET 指令后，不再拥有 Equipment Sequence 的所有权。Equipment Sequence 的所有权随后可归另一程序或 FactoryTalk Batch 软件所有。仅当程序之前曾通过连接到 Equipment Sequence (SATT) 指令获取 Equipment Sequence 的所有权时，才能使用 SDET 指令。

可用语言

梯形图



功能块

此指令不可用于功能块中。

结构化文本

SDET(SequenceName)

操作数

SDET 指令使用以下操作数。

操作数	类型	格式	说明
Sequence Name	Sequence	Equipment Sequence 的名称	要释放所有权的 Equipment Sequence。

影响数学状态标志

否

严重/轻微故障

SDET 指令不会触发故障，因此此指令不存在故障条件。

执行

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。

条件	执行的操作
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	指令执行。 梯级输出条件设置为真。
扫描结构化文本	不执行任何操作
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	梯级输出条件设置为假。

结构化文本

条件	执行的操作
预扫描	不执行任何操作。
梯级输入条件为假	不执行任何操作。
梯级输入条件为真	不执行任何操作。
扫描结构化文本	在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构（包括 if、then 或 else 等条件）
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	不执行任何操作。

另请参见

SDET 指令示例 参考页数 541

设备顺序指令 参考页数 519

设备顺序分配顺序标识符 (SASI)

设备顺序分配标识符 (SASI) 指令用于向 Equipment Sequence 分配顺序 ID。只有在符合以下先决条件时才可设置顺序 ID：

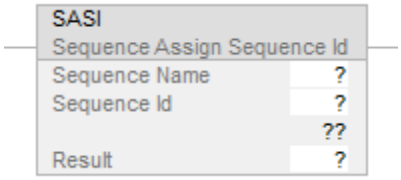
- 控制器处于联机状态。
- Equipment Sequence 处于 IDLE 状态。
- 已获取 Equipment Sequence 的所有权。

顺序 ID 最长可包含 82 个字符，可使用以下可打印 ASCII 字符：

a-z、A-Z、0-9、!"#$%&'()+,-./:;<=>?@[\]^_`{|}~ 和空格*

可用语言

梯形图



功能块

此指令不可用于功能块中。

结构化文本

SASI(SequenceName,SequenceID,SequenceIDValue,Result)

操作数

SASI 指令使用以下操作数。

操作数	类型	格式	说明
Sequence Name	Sequence	Equipment Sequence 的名称	要分配标识符的 Equipment Sequence。
Sequence ID	STRING	标签	输入用于存储标识符的 STRING 标签，或最多包含 82 个字符的引证字符串。
Result	DINT	立即数 标签	为使指令返回指示其执行成功或失败的代码，需输入用于存储结果代码的 DINT 标签。否则，需输入 **0**。

影响数学状态标志

否

严重/轻微故障

SASI 指令不会触发故障，因此此指令不存在故障条件。

执行

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	• 指令执行。 • 梯级输出条件设置为真。
扫描结构化文本	不执行任何操作
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	梯级输出条件设置为假。

结构化文本

条件	执行的操作
预扫描	不执行任何操作。
梯级输入条件为假	不执行任何操作。
梯级输入条件为真	不执行任何操作。
扫描结构化文本	在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构。
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	不执行任何操作。

另请参见

SASI 指令示例 参考页数 539

设备顺序指令 参考页数 519

设备顺序清除故障 (SCLF)

Equipment Sequence 清除故障（SCLF）指令用于清除 Equipment Sequence 的故障代码。使用 SCLF 指令时，请谨记以下事项。

- CLR 指令、MOV 指令或赋值不会更改 Equipment Sequence 的故障代码。
- 使用 SCLF 指令时，Equipment Sequence 的所有权不能归其他所有者。如果 Logix Designer 应用程序、FactoryTalk Batch 软件或其

他程序拥有 Equipment Sequence 的所有权，SCLF 指令不会清除故障代码。

- Equipment Sequence 将拒绝 RESUME 命令，直至故障清除。

可用语言

梯形图

功能块

不可用

结构化文本

SCLF(SequenceName)

操作数

SCLF 指令使用以下操作数。

操作数	类型	格式	说明
Sequence Name	Sequence	Equipment Sequence 的名称	要清除故障代码的 Equipment Sequence。
Result	DINT	立即数标签	为使指令返回成功或失败的代码，可输入用于存储结果代码的 DINT 标签。否则，需输入 **0**。

影响数学状态标志

否。

严重/轻微故障

SCLF 指令不会触发故障，因此此指令不存在故障条件。

执行

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	指令执行。 梯级输出条件设置为真。
扫描结构化文本	不执行任何操作
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	梯级输出条件设置为假。

结构化文本

条件	执行的操作
预扫描	不执行任何操作。
梯级输入条件为假	不执行任何操作。
梯级输入条件为真	不执行任何操作。
扫描结构化文本	在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构（包括 if、then 或 else 等条件）
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	不执行任何操作。

另请参见

SCLF 指令的结果代码 参考页数 536

SDET 指令示例 参考页数 541

设备顺序指令 参考页数 519

设备顺序命令 (SCMD)

Equipment Sequence 命令 (SCMD) 指令用于更改 Equipment Sequence 的状态。SCMD 指令可向 Equipment Sequence 发送以下命令：START、RESTART、HOLD、STOP、ABORT 和 RESET。程序必须先连接到 Equipment Sequence，之后 SCMD 指令才会执行。可使用 SATT 指令连接到 Equipment Sequence。

与 SCMD 指令类似，Equipment Sequence 超控指令 (SOVR) 也会更改 Equipment Sequence 的状态，但不同的是，后者无论所有权如何，都会更改状态。如果无论所有权如何都必须执行 SCMD 指令，可使用 SOVR 指令而非 SCMD 指令。

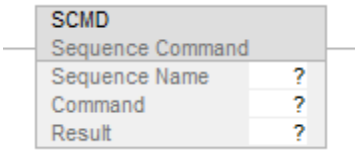
重要事项： SOVR 指令仅用于紧急情况。控制工程师应慎重考虑是否使用该指令。

当分配标签来存储 SCMD 指令的结果时，指令会在执行时返回以下五个结果代码之一。结果代码 **0** 表示 SCMD 指令已成功执行。其他四个代码表示指令未成功执行，并提供与指令执行失败的原因相关的附加信息。

可用语言

SCMD 指令支持以下语言。

梯形图



功能块

此指令不可用于功能块中。

结构化文本

SCMD(SequenceName,Command,Result)

支持的操作数

SCMD 指令使用以下操作数。

操作数	类型	格式	说明
Sequence Name	Sequence	Equipment Sequence 的名称	要更改为其他状态的 Equipment Sequence。例如，Make_Product_101。
Command	Command	命令名称	要发送至 Equipment Sequence 从而更改状态的命令。可发送以下命令之一：START、RESTART、HOLD、STOP、ABORT 或 RESET。
Result	DINT	立即数标签	为使指令返回成功或失败的代码，可输入用于存储结果代码的 DINT 标签。否则，需输入 **0**。

SCMD 指令的有效命令状态

SCMD 指令命令用于将 Equipment Sequence 转换为其他状态。SCMD 指令命令只能在某些状态下进行处理。下表列出了命令有效的状态。

命令	在以下状态下有效
START	在 IDLE 状态下有效。
RESTART	在 HELD 状态下有效。
HOLD	在 RUNNING 和 RESTARTING 状态下有效。
STOP	在 RUNNING、HOLDING、RESTARTING 和 HELD 状态下有效。
ABORT	在 RUNNING、HOLDING、RESTARTING、STOPPING 和 HELD 状态下有效。
RESET	在 ABORTED、STOPPED 和 COMPLETE 状态下有效。

算术状态标志和故障条件

算术状态标志不受 SCMD 指令的影响。SCMD 指令不会触发故障，因此此指令不存在故障条件。

执行

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	指令执行。 梯级输出条件设置为真。
扫描结构化文本	不执行任何操作

条件	执行的操作
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	梯级输出条件设置为假。

结构化文本

条件	执行的操作
预扫描	不执行任何操作。
梯级输入条件为假	不执行任何操作。
梯级输入条件为真	不执行任何操作。
扫描结构化文本	在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构。
指令执行	指令试图获取指定 Equipment Sequence 的所有权。
后扫描	不执行任何操作。

另请参见

SCMD 指令示例 参考页数 541

使用 SOVR 指令而非 SCMD 指令 参考页数 542

SCMD 指令的结果代码 参考页数 537

SCMD 指令指导原则 参考页数 534

设备顺序图指令

下表介绍 Equipment Sequence 图指令。

图标	图标名称	说明
	添加步和转换对	使用**添加步和转换对** (Add Step and Transition Pair) 可添加步和转换对。尽管是成对添加，您仍可单独选择并编辑各个元素。
	添加断开的步	使用**添加断开的步** (Add Disconnected Step) 可在不添加转换的情况下添加步。
	添加断开的转换	使用**添加断开的转换** (Add Disconnected Transition) 可在不添加步的情况下添加转换。
	添加并发扩散	使用**添加并发扩散** (Add Simultaneous Divergence) 可创建一个分支，其中所有链接的步都会同时执行。

	添加选择性扩散	使用**添加选择性扩散** (Add Selective Divergence) 可为选择性分支创建扩散。在选择性扩散中，只会执行多条路径中的一条路径 - 即第一个值为"真"的转换所在的路径。
	添加并发聚合	使用**添加并发聚合** (Add Simultaneous Convergence) 可将多个并发执行路径重新合并。
	添加选择性聚合	使用**添加选择性聚合** (Add Selective Convergence) 可在选择性分支中将多个选择性扩散路径重新合并为一个执行路径。

另请参见

设备顺序指令 参考页数 519

设备顺序超控 (SOVR)

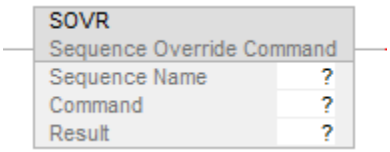
无论所有权如何，均可使用 Equipment Sequence 超控 (SOVR) 指令将 HOLD、STOP 或 ABORT 命令发送到 Equipment Sequence。

重要事项： SOVR 指令仅用于紧急情况。控制工程师应慎重考虑是否使用该指令。

当分配标签来存储 SOVR 指令的结果时，指令会在执行时返回以下五个结果代码之一。结果代码 **0** 表示 SOVR 指令已成功执行。其他四个代码表示指令未成功执行，并提供与指令执行失败的原因相关的附加信息。

可用语言

梯形图



功能块

此指令不可用于功能块中。

结构化文本

SOVR(SequenceName,Command,Result)

操作数

SOVR 指令使用以下操作数。

操作数	类型	格式	说明
Sequence Name	Sequence	Equipment Sequence 的名称	要更改为其他状态的 Equipment Sequence。例如，Make_Product_101。
Command	Command	命令名称	要发送至 Equipment Sequence 从而更改状态的命令。使用以下命令之一： • HOLD • STOP • ABORT
Result	DINT	立即数标签	为使指令返回成功或失败的代码，可输入用于存储结果代码的 DINT 标签。否则，需输入 **0**。

影响数学状态标志

否。

严重/轻微故障

SOVR 指令不会触发故障，因此此指令不存在故障条件。

执行

梯形图

条件	执行的操作
预扫描	梯级输出条件设置为假。
梯级输入条件为假	梯级输出条件设置为假。
梯级输入条件为真	指令执行。 梯级输出条件设置为真。
扫描结构化文本	不执行任何操作
指令执行	指令控制 Equipment Sequence 进入指定状态。
后扫描	梯级输出条件设置为假。

结构化文本

条件	执行的操作
预扫描	不执行任何操作。
梯级输入条件为假	不执行任何操作。
梯级输入条件为真	不执行任何操作。

条件	执行的操作
扫描结构化文本	在结构化文本中，指令会在每次被扫描时执行。要限制指令的扫描，可使用 SFC 操作的限定符或结构化文本结构（包括 if、then 或 else 等条件）。
指令执行	指令控制 Equipment Sequence 进入指定状态。SOVR 指令会发送以下命令之一：HOLD、STOP 或 ABORT。
后扫描	不执行任何操作。

另请参见

SOVR 指令指导原则 参考页数 535

SOVR 指令的结果代码 参考页数 538

SOVR 指令示例 参考页数 542

在何种情况下应使用 SOVR 指令来代替 SCMD 指令？ 参考页数 542

SATT 指令指导原则

使用连接到 Equipment Sequence (SATT) 指令时，请谨记以下指导原则。

指导原则	详细信息
请记住，Logix Designer 应用程序会优先获取 Equipment Sequence 的所有权。	所有权可确保程序控制 Equipment Sequence，并且会排除任何其他定序程序。
Equipment Sequence 的所有权必须由程序拥有，然后程序才能对其进行控制。	Equipment Sequence 和 Equipment Phase 只有在所有权都有*归属*的情况下才可执行。所有权命令为**附加** (SATT) 和 **拆离** (SDET)。 内部定序程序（程序）、外部定序程序 (FactoryTalk Batch) 和操作员均使用**连接**指令控制 Equipment Sequence。
序列完成后，会放弃所有权。	要放弃所有权，可使用从 Equipment Sequence 断开 (SDET) 指令。
如果设备顺序正在生成顺序事件，应避免提出不必要的命令请求。	不必要的命令请求可能会淹没事件处理缓冲区，从而导致用户错过重大事件。

使用**结果**代码验证所有权，并且指令中应包含因其他程序或操作员获得 Equipment Sequence 所有权而导致连接失败时应执行的步。	使用 **Result** 操作数获取显示 SATT 指令的执行成功或失败的代码。 每次执行时，SATT 指令都会尝试获得 Equipment Sequence 的所有权。当程序或操作员获得 Equipment Sequence 的所有权时，SATT 指令的再次执行会失败并生成结果代码 24582。当使用 SATT 指令时，请采取以下任一措施： • 将其执行限制为单次扫描，避免生成结果代码 24582。 • 将以下内容包括在所有权条件中：结果代码 = 24582。

另请参见

连接到设备顺序 参考页数 519

SATT 指令的结果代码 参考页数 536

设备顺序指令 参考页数 519

SCMD 指令指导原则

使用 Equipment Sequence 命令 (SCMD) 指令时，请谨记以下指导原则。SCMD 指令可发送以下命令：START、RESTART、HOLD、STOP、ABORT 和 RESET。

指导原则	详细信息
将 SCMD 指令的执行限制为单次扫描。	将 SCMD 指令的执行限制为单次扫描。每条命令均适用于一种特定状态或多个状态。一旦 Equipment Sequence 的状态发生变化，命令将不再有效。若要限制执行，可使用以下方法： • 在 P1 脉冲 (上升沿) 或 P0 脉冲 (下降沿) 动作中运行 SCMD 指令。 • 在 SCMD 指令之前放置单脉冲触发指令。 • 运行 SCMD 指令，然后继续执行下一步。
Equipment Sequence 的所有权必须由程序拥有，然后程序才能对其进行控制。	Equipment Sequence 和 Equipment Phase 只有在所有权都有*归属*的情况下才可执行。所有权命令为**附加** (SATT) 和 **拆离** (SDET)。 内部定序程序 (程序)、外部定序程序 (FactoryTalk Batch) 和操作员均使用**连接**指令控制 Equipment Sequence。

另请参见

设备顺序命令 (SCMD) 参考页数 528

设备顺序指令 参考页数 519

在何种情况下应使用 SOVR 指令来代替 SCMD 指令？ 参考页数 542

SOVR 指令指导原则

无论所有权如何，均可使用 Equipment Sequence 超控 (SOVR) 指令将 HOLD、STOP 或 ABORT 命令发送到 Equipment Sequence。

重要事项： SOVR 指令仅用于紧急情况。控制工程师应慎重考虑是否使用该指令。

使用 Equipment Sequence 超控 (SOVR) 指令时，请谨记以下指导原则。

指导原则	详细信息
确保您希望优先于其他所有者。	在大多数情况下，可使用 SCMD 指令以编程方式控制 Equipment Sequence。然而，在以下条件下，可使用 SOVR 指令控制 Equipment Sequence： • 当您发出 HOLD、STOP 或 ABORT 命令并且命令必须始终在全部所有权情况下执行时。 • 即使通过 Logix Designer 应用程序手动控制 Equipment Sequence 或者其他程序（如 FactoryTalk Batch 软件）获得 Equipment Sequence 的所有权时，也必须执行 HOLD、STOP 或 ABORT 命令的情况。
将 SOVR 指令的执行限制为单次扫描。	将 SOVR 指令的执行限制为单次扫描。每条命令均适用于一种特定状态或多个状态。一旦 Equipment Sequence 的状态发生变化，命令将不再有效。若要限制执行，可使用以下方法： • 在 P1 脉冲（上升沿）或 P0 脉冲（下降沿）动作中运行 SOVR 指令。 • 在 SOVR 指令之前放置单脉冲触发指令。 • 运行 SOVR 指令，然后继续执行下一步。
如果 Equipment Sequence 正在生成顺序事件，应避免生成不必要的命令请求。	不必要的命令请求可能会淹没事件处理缓冲区，从而导致用户错过重大事件。

另请参见

设备顺序超控命令 参考页数 531

SOVR 指令的结果代码 参考页数 538

在何种情况下应使用 SOVR 指令来代替 SCMD 指令？ 参考页数 542

SATT 指令的结果代码

当分配标签来存储连接到 Equipment Sequence (SATT) 指令的结果时，指令会在执行时返回以下代码之一。

代码（十进制）	说明
0	命令成功执行。
24579	出于以下原因，无法控制 Equipment Sequence： 程序已成功关联到 Equipment Sequence，但由于优先级更高的应用程序 Logix Designer 已优先获得所有权，因此程序不能控制此顺序。
24582	程序已获取 Equipment Sequence 的所有权。
24593	以下任一项获取设备阶段的所有权。 • 外部定序程序，如 FactoryTalk Batch 软件。 • 控制器中的其他程序。
24594	Equipment Sequence 未规划、已被禁止或处于已被禁止的任务中。

使用 **Result** 操作数获取显示 SATT 指令的执行成功或失败的代码。**Result** 操作数应包含 **0** 或一个 DINT 标签，具体取决于是否可能发生所有权冲突或其他错误。

- 如果不可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入 **0**。
- 如果可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入一个 DINT 标签。DINT 标签存储指令执行结果的代码。

另请参见

连接到设备顺序 参考页数 519

设备顺序指令 参考页数 519

SCLF 指令的结果代码

当分配标签来存储 Equipment Sequence 清除故障 (SCLF) 指令的结果时，指令会在执行时返回以下代码之一。

代码（十进制）	说明
0	命令成功执行。

48	命令未执行，因为当时无法生成用于记录命令的事件。 • 如果命令为 ABORT 命令，即使无法生成事件，也仍会执行 ABORT 命令。 • 仅当在**设备顺序属性 - 配置** (Equipment Sequence Properties - Configuration) 选项卡中启用了事件生成功能时，才会出现此代码。
24578	命令对 Equipment Sequence 的当前状态无效。例如，如果 Equipment Sequence 停止，则停止命令无效。
24594	Equipment Sequence 未规划、已被禁止或处于已被禁止的任务中。

使用 **Result** 操作数获取显示 SCLF 指令的执行成功或失败的代码。**Result** 操作数应包含 **0** 或一个 DINT 标签，具体取决于是否可能发生所有权冲突或其他错误。

- 如果不可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入 **0**。
- 如果可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入一个 DINT 标签。DINT 标签存储指令执行结果的代码。

另请参见

设备顺序清除故障 参考页数 525

设备顺序指令 参考页数 519

SCMD 指令的结果代码

当分配标签来存储 Equipment Sequence 命令 (SCMD) 指令的结果时，指令会在执行时返回以下代码之一。

代码（十进制）	说明
0	命令成功执行。
48	命令未执行，因为当时无法生成用于记录命令的事件。 • 如果命令为 ABORT 命令，即使无法生成事件，也仍会执行 ABORT 命令。 • 仅当在**设备顺序属性 - 配置** (Equipment Sequence Properties - Configuration) 选项卡中启用了事件生成功能时，才会出现此代码。
24577	命令无效。
24578	命令对 Equipment Sequence 的当前状态无效。例如，如果 Equipment Sequence 处于运行状态，则启动命令无效。

24579	出于以下原因，无法控制 Equipment Sequence： • 程序已成功关联到 Equipment Sequence，但由于优先级更高的应用程序 Logix Designer 已优先获得所有权，因此程序不能控制此顺序。
24582	未成功关联至 Equipment Sequence，因为此序列之前已关联到下列任一用户： • 外部定序器（如 FactoryTalk Batch 软件）拥有所有权。 • 控制器中的另一程序（内部定序器）拥有所用权。 • 使用 Sequence Manager ActiveX 控件的操作员拥有所有权。
24580	指令的调用程序已连接，但未获取 Equipment Sequence 的当前所有权令。较高优先级的所有者（例如 Logix Designer）正在对 Equipment Sequence 进行控制。
24594	Equipment Sequence 未规划、已被禁止或处于已被禁止的任务中。
24604	正在处理相同或更高优先级的命令。
24631	每步定义的顺序参数或步标签过多，因此无法处理事件，START 命令失败。

使用 **Result** 操作数获取显示 SCMD 指令的执行成功或失败的代码。**Result** 操作数应包含 **0** 或一个 DINT 标签，具体取决于是否可能发生所有权冲突或其他错误。

- 如果不可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入 **0**。
- 如果可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入一个 DINT 标签。DINT 标签存储指令执行结果的代码。

另请参见

设备顺序命令 参考页数 528

设备顺序指令 参考页数 519

SOVR 指令的结果代码

当分配标签来存储 Equipment Sequence 超控 (SOVR) 指令的结果时，指令会在执行时返回以下代码之一。

代码（十进制）	说明
0	命令成功执行。
48	命令未执行，因为当时无法生成用于记录命令的事件。 • 如果命令为 ABORT 命令，即使无法生成事件，也仍会执行 ABORT 命令。 • 仅当在**设备顺序属性 - 配置** (Equipment Sequence Properties - Configuration) 选项卡中启用了事件生成功能时，才会出现此代码。
24577	命令无效。

24578	命令对 Equipment Sequence 的当前状态无效。例如，如果 Equipment Sequence 停止，则停止命令无效。
24594	Equipment Sequence 未规划、已被禁止或处于已被禁止的任务中。

使用 **Result** 操作数获取显示 SOVR 指令的执行成功或失败的代码。**Result** 操作数应包含 **0** 或一个 DINT 标签，具体取决于是否可能发生所有权冲突或其他错误。

- 如果不可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入 **0**。
- 如果可能发生所有权冲突或其他错误，请在 **Result** 操作数中输入一个 DINT 标签。DINT 标签存储指令执行结果的代码。

另请参见

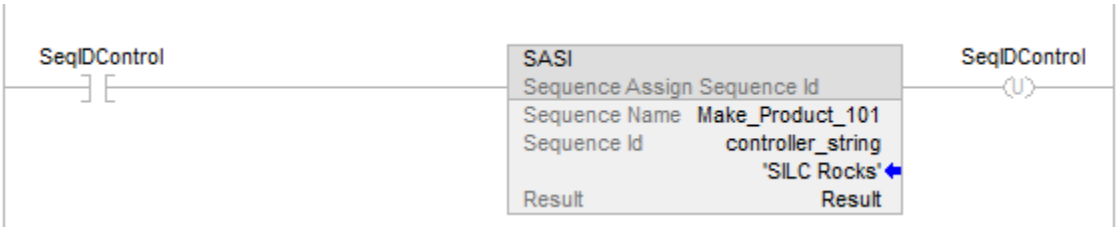
设备顺序超控命令 参考页数 531

设备顺序指令 参考页数 519

SASI 指令示例

以下示例展示梯形图和结构化文本中的 SASI 指令。

梯形图示例



提示： Sequence ID 参数可以是用于存储标识符的 STRING 标签，也可以是最多包含 82 个字符的引证字符串。

结构化文本示例

```
if (IdControl) then
  SASI (Make_Product_101, NewId, Results);
end_if;
```

另请参见

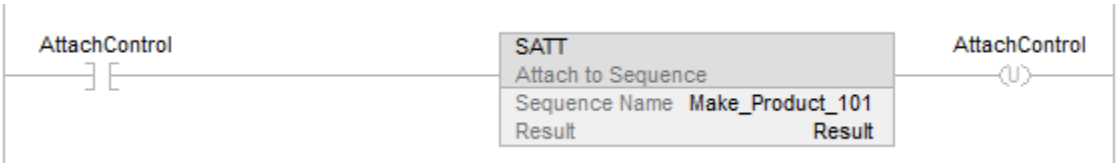
设备顺序分配顺序标识符 参考页数 523

设备顺序指令 参考页数 519

SATT 指令示例

以下示例展示梯形图和结构化文本中的 SATT 指令。

梯形图



结构化文本

```
if (AttachControl) then
  SATT (Make_Product_101, Result);
end_if;
```

另请参见

连接到设备顺序 参考页数 519

SATT 指令指导原则 参考页数 533

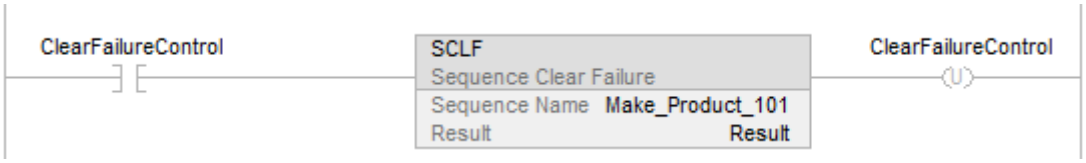
SATT 指令的结果代码 参考页数 536

设备顺序指令 参考页数 519

SCLF 指令示例

以下示例展示梯形图和结构化文本中的 SCLF 指令。

梯形图示例



结构化文本示例

```
if (ClearFailureControl) then
  SCLF (Make_Product_101);
```

```
end_if;
```

另请参见

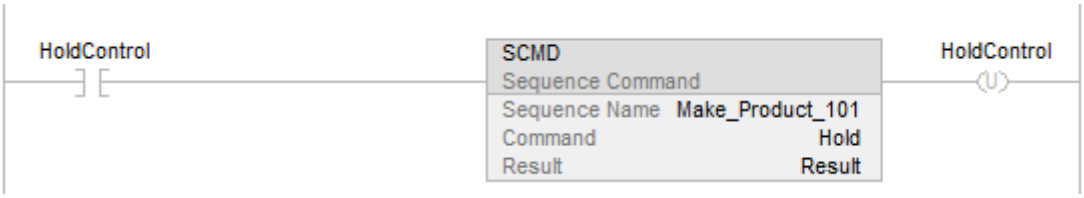
设备顺序清除故障 参考页数 525

设备顺序指令 参考页数 519

SCMD 指令示例

以下示例展示梯形图和结构化文本中的设备序列命令（SCMD）指令。

梯形图



结构化文本

```
if (HoldControl) then
  SCMD (Make_Product_101), Hold, Result);
end_if;
```

另请参见

设备顺序命令 参考页数 528

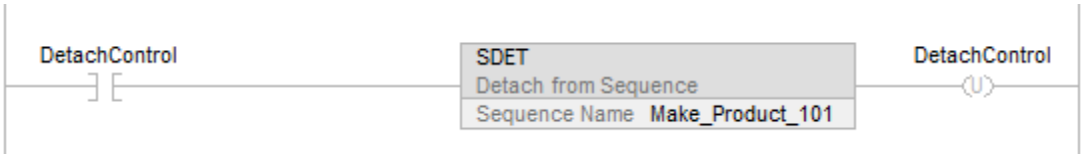
SCMD 指令指导原则 参考页数 534

设备顺序指令 参考页数 519

SDET 指令示例

以下示例展示梯形图和结构化文本中的 SDET 指令。

梯形图



结构化文本

```
if (DetachControl) then
```

SDET (Make_Product_101);

end_if;

另请参见

从设备顺序断开 参考页数 522

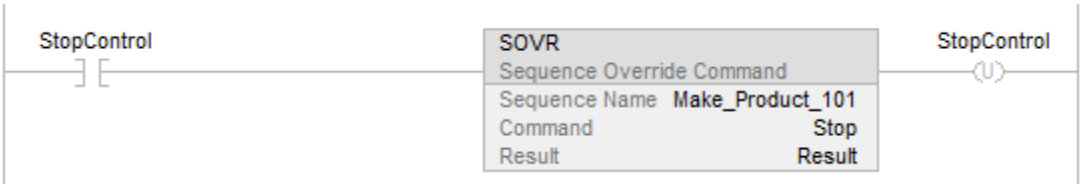
连接到设备顺序 参考页数 519

设备顺序指令 参考页数 519

SOVR 指令示例

以下示例展示梯形图和结构化文本中的 SOVR 指令。

梯形图



结构化文本

if (StopControl) then

SOVR (Make_Product_101, Stop, Results);

end_if;

另请参见

设备顺序命令 参考页数 528

SCMD 指令指导原则 参考页数 534

设备顺序指令 参考页数 519

在何种情况下应使用 SOVR 指令来代替 SCMD 指令？

在大多数情况下，可使用 SCMD 指令以编程方式控制 Equipment Sequence。然而，在以下条件下，可使用 SOVR 指令控制 Equipment Sequence：

- 当您发出 HOLD、STOP 或 ABORT 命令并且命令必须始终在全部所有权情况下执行时。
- 即使通过 Logix Designer 应用程序手动控制 Equipment Sequence 或者其他程序（如 FactoryTalk Batch 软件）获得 Equipment

Sequence 的所有权时，也必须执行 HOLD、STOP 或 ABORT 命令的情况。

例如，假设设备检查物料是否堵塞。如果存在堵塞，则始终希望设备中止。在这种情况下，使用 SOVR 指令。这样，即使通过 Logix Designer 应用程序进行手动控制，设备也会中止。

另请参见

设备顺序命令 参考页数 528

设备顺序超控命令 参考页数 531

SCMD 指令指导原则 参考页数 534

设备顺序指令 参考页数 519

功能块属性

要了解功能块编程特有问题的更多信息，请单击下方的相应主题。请认真阅读这些信息，以确保您了解功能块例程的工作方式。

另请参见

选择功能块元素 参考页数 545

锁存数据 参考页数 546

执行顺序 参考页数 548

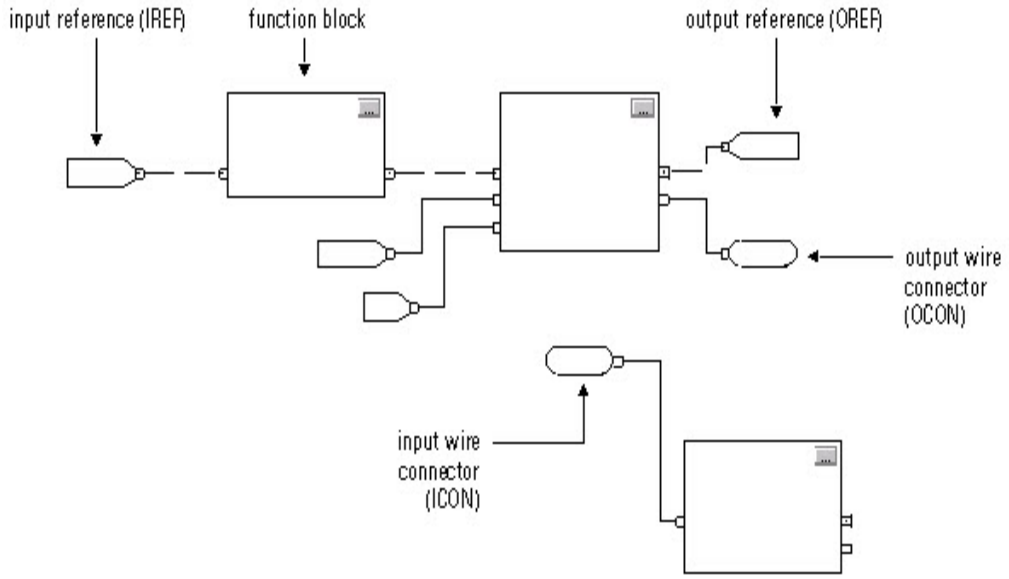
功能块对溢出条件的响应 参考页数 547

时序模式 参考页数 552

程序/操作员控制 参考页数 555

选择功能块元素

若要控制设备，可使用以下元素：



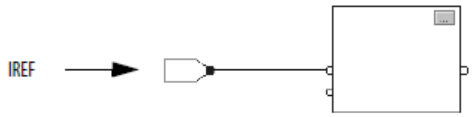
下表可帮助您选择功能块元素：

若希望由输入设备或标签提供值	则使用输入参考 (IREF)
将值发送至输出设备或标签	输出参考 (OREF)
对一个或多个输入值执行运算，并生成一个或多个输出值	功能块
当功能块符合以下条件时，在功能块之间传送数据： • 在同一表单内相隔较远 • 在同一例程内的不同表单上	输出接线器 (OCON) 和输入接线器 (ICON)
将数据分散到例程中的多个点上	单个输出接线器 (OCON) 和多个输入接线器 (ICON)

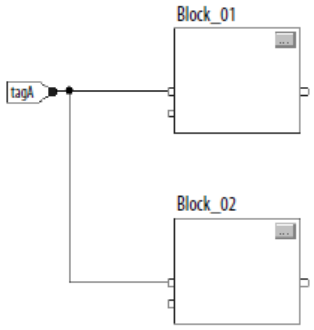
功能块将输入参考移入块结构中。如有必要，功能块会将这些输入参考转换为 REAL 值。功能块执行并将结果移入输出参考。同样，如有必要，功能块会将结果值由 REAL 值转换为输出参考的数据类型。

锁存数据

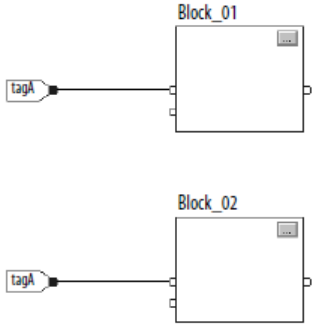
如果使用 IREF 指定功能块指令的输入数据，会锁存此 IREF 中的数据，以便扫描功能块例程。IREF 会锁存程序作用域标签和控制器作用域标签中的数据。控制器会在每次扫描开始时更新所有 IREF 数据。



在本例中，tagA 的值是在例程开始执行时存储的。Block_01 执行时，会使用存储值。Block_02 执行时，也会使用相同的存储值。如果 tagA 的值在例程执行过程中发生变化，则在例程下次开始执行之后，IREF 中 tagA 的存储值才会发生变化。

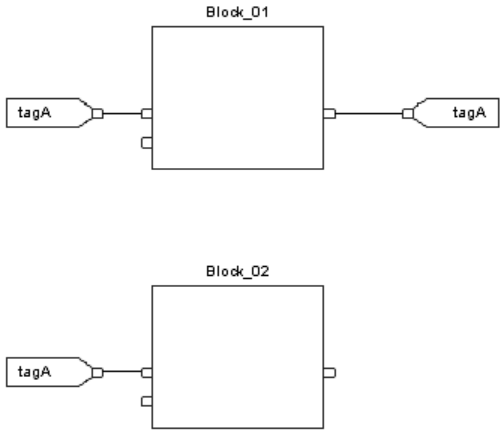


此例与上一示例相同。仅会在例程开始执行时存储一次 tagA 的值。在整个例程中都使用这一存储值。



在同一例程的多个 IREF 和一个 OREF 中可以使用同一个标签。由于每次扫描例程时 IREF 中的标签值都被锁存，因此所有的 IREF 都将使用相同的值，即使在例程的执行过程中 OREF 获得不同的标签值也是如此。

在本示例中，如果当例程开始执行此次扫描时 tagA 的值是 25.4，并且 Block_01 将 tagA 的值改为 50.9，则当 Block_02 执行本次扫描时，接入 Block_02 的第二个 IREF 仍将使用值 25.4。在下次扫描开始后，此例程中的 IREF 才会使用 tagA 的新值 50.9。



功能块对溢出条件的响应

通常，存有历史记录的功能块指令在发生溢出时不会使用 ±NAN 或 ±INF 值来更新历史记录。每条指令都会对溢出条件作出以下任一响应。

响应	指令

响应 1 功能块执行其算法并检查结果是否为 ±NAN 或 ±INF。如果为 ±NAN 或 ±INF，功能块会输出 ±NAN 或 ±INF。	ALM NTCH DEDT PMUL DERV POSP ESEL RLIM FGEN RMPS HPF SCRV LDL2 SEL LDLG SNEG LPF SRTP MAVE SSUM MAXC TOT MINC UPDN MSTD MUX
响应 2 具有输出限制的功能块执行其算法，并检查结果是否为 ±NAN 或 ±INF。输出限值由输入参数 HighLimit 和 LowLimit 定义。如果结果为 ±INF，功能块会输出受限制的结果。如果结果为 ±NAN，则不会使用输出限值，功能块会输出 ±NAN。	HLL、 INTG、 PI、 PIDE、 SCL、 SOC
响应 3 溢出条件不适用。这些指令通常均有具有布尔型输出。	BAND、 BNOT、 BOR、 BXOR、 CUTD、 D2SD、 D3SD、 DFF、 JKFF、 OSFI、 OSRI、 RESD、 RTOR、 SETD、 TOFR、 TONR

执行顺序

以下情况下，Logix Designer 编程应用程序会自动确定例程中各功能块的执行顺序：

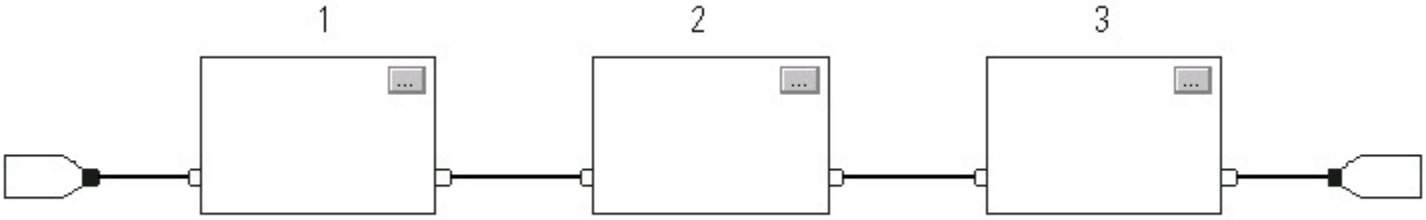
- 验证功能块例程
- 验证包含功能块例程的项目
- 下载包含功能块例程的项目

要定义执行顺序，可将功能块连接在一起，必要时可指示任何反馈线的数据流。

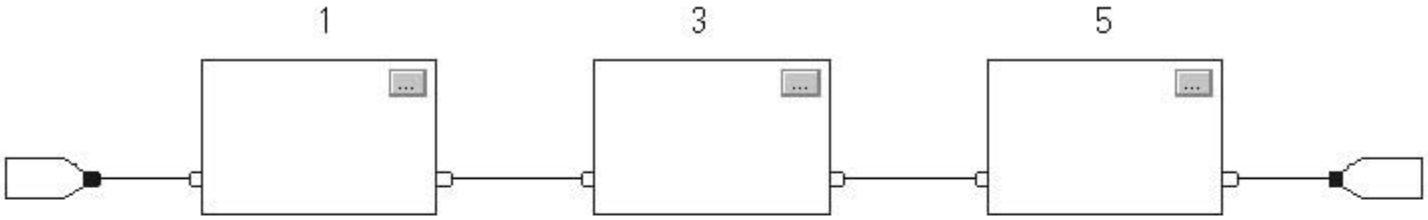
如果功能块未连接在一起，则功能块的执行顺序并不重要。功能块之间不存在数据流

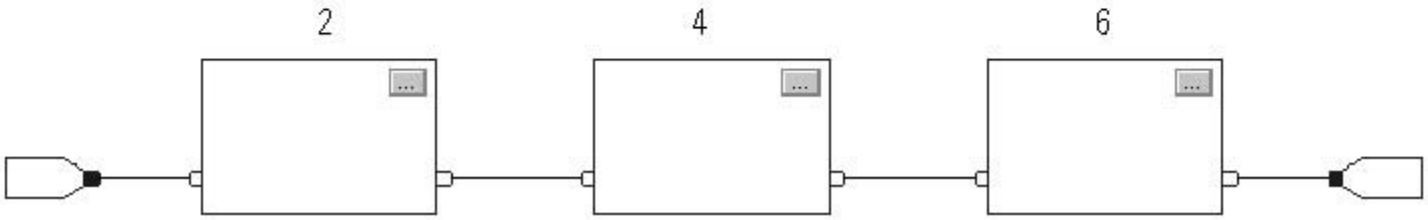
如果将功能块按顺序连接在一起，将按由输入到输出的顺序执行。要使控制器执行某个功能块，必须首先获得该功能块的输入数据。例如，由

于功能块 2 的输出将送入功能块 3 的输入，因此功能块 2 必须在功能块 3 之前执行。



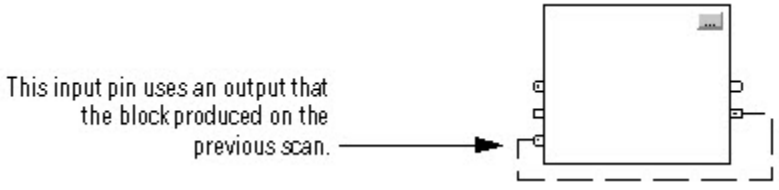
执行顺序仅与连接在一起的功能块相关。在下面的示例中，由于两组功能块未连接在一起，因此不存在执行顺序问题。在特定分组中的功能块将按照相对于该分组中各功能块的顺序执行。



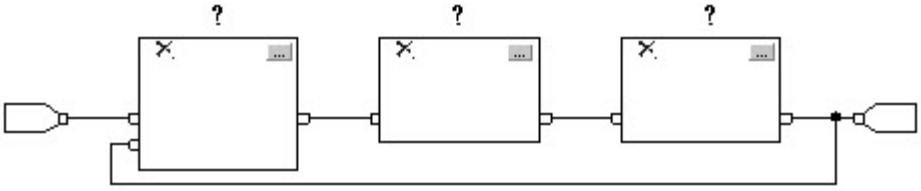


解析回路

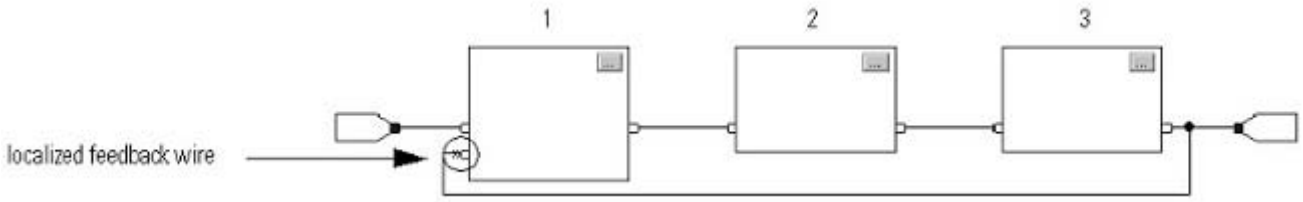
要围绕某个功能块创建反馈回路，应将功能块的输出引脚连接到同一功能块的输入引脚。下面的示例不存在问题。该回路只包含一个功能块，因此执行顺序并不重要。



如果回路中有一组功能块，控制器将无法确定先执行哪一个功能块。也就是说，控制器无法解析回路。

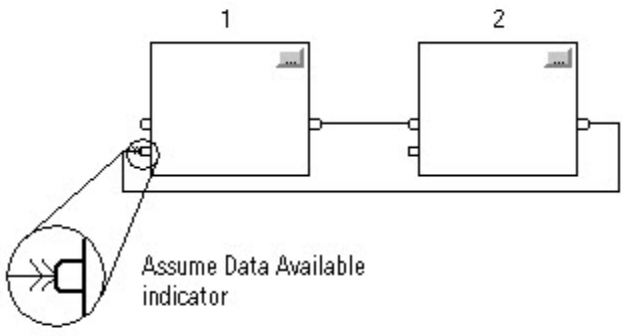
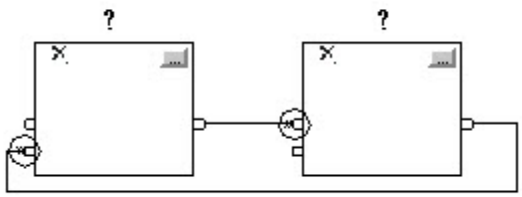


要确定首先执行哪个功能块，应使用 *Assume Data Available*（假设数据可用）指示符标记用于创建回路的输入线（反馈线）。在下例中，功能块 1 使用前次执行例程时功能块 3 的输出。



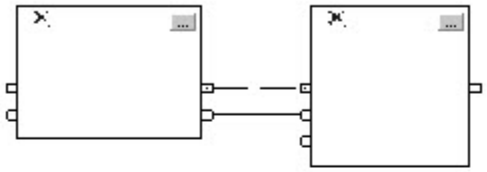
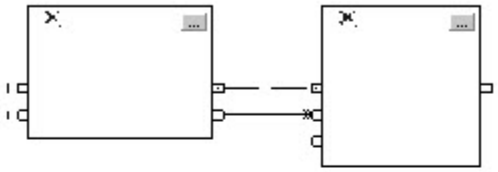
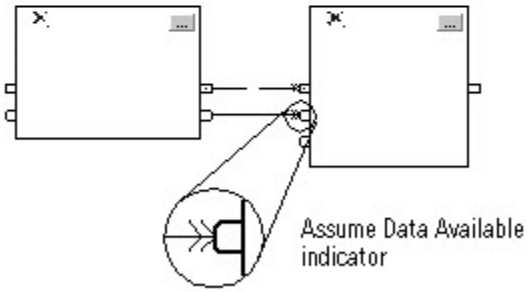
Assume Data Available（假设数据可用）指示符用于定义回路中的数据流。箭头指示数据将作为回路中第一个功能块的输入。

请*勿*使用 *Assume Data Available*（假设数据可用）指示符标记回路中的所有线路。

This is OK	This is NOT OK
 The *Assume Data Available* indicator defines the data flow within the loop.	 The controller cannot resolve the loop because all the wires use the *Assume Data Available* indicator.

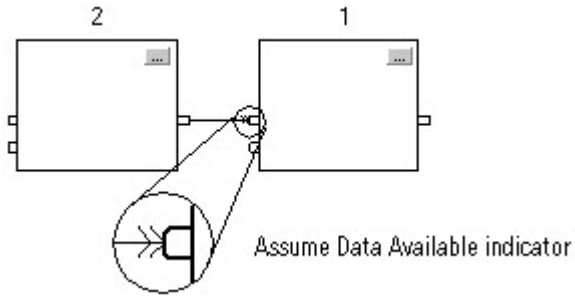
解析两个功能块之间的数据流

如果使用两条或多条导线连接两个功能块，应对两个功能块之间的所有导线使用相同的数据流指示符。

This is OK	This is NOT OK
 Neither wire uses the *Assume Data Available* indicator.	 One wire uses the *Assume Data Available* indicator while the other wire does not.
 Both wires use the *Assume Data Available* indicator.	

创建一个扫描周期的延时

要在两个功能块之间产生一个扫描周期的延时，应使用“Assume Data Available”（假设数据可用）指示符。在下面的示例中，功能块 1 首先执行。它使用上次扫描例程时功能块 2 的输出。



总结

综上所述，功能块例程按以下顺序执行：

1. 控制器将所有数据值锁存在 IREF 中。
2. 控制器按照由接线方式确定的顺序执行其他功能块。
3. 控制器将输出写入 OREF。

时序模式

这些过程控制和驱动指令支持不同的时序模式。

- DEDT
- DERV
- HPF
- INTG
- LDL2
- LDLG
- LPF
- NTCH
- PI
- PIDE
- RLIM
- SCRV
- SOC
- TOT

时序模式有三种。

时序模式	说明	
周期性	周期性模式为默认模式，适用于大多数控制应用。我们建议将使用此模式的指令置于在周期性任务中执行的例程中。指令的增量时间 (DeltaT) 的确定方法如下：	
	如果指令在以下任务中执行	**DeltaT 将等于**
	周期性任务	任务的周期
	事件任务或连续任务	自上次执行后所经过的时间 控制器将经过的时间截断为整毫秒 (ms) 的形式。例如，如果经过的时间 = 10.5 ms，则控制器会设置 DeltaT = 10 ms。
	过程输入的更新需要与任务的执行同步，或者其采样速度需比任务的执行速度快 5-10 倍，以最小化输入和指令之间的采样误差。	
过采样	在过采样模式下，指令使用的增量时间 (DeltaT) 是写入指令的 OversampleDT 参数中的值。如果过程输入具有时戳值，则改为使用实时采样模式。 向程序添加逻辑以控制指令何时执行。例如，可以使用设置为 OversampleDeltaT 值的计时器通过使用指令的 EnableIn 输入来控制执行。 过程输入的采样速度需要比指令的执行速度快 5-10 倍，以最小化输入和指令之间的采样误差。	
实时采样	在实时采样模式下，指令使用的增量时间 (DeltaT) 是与过程输入更新相对应的两个时戳值之差。当过程输入具有与其更新相关的时戳，并且需要精确协调时，使用此模式。 从为指令的 RTSTimeStamp 参数输入的标签名称中读取时戳值。通常，此标签名称是与过程输入相关的输入模块上的参数。 指令将配置的 RTSTime 值（预期更新周期）与计算得到的 DeltaT 进行比较，以确定过程输入的每次更新是否都被指令读取。如果 DeltaT 不在 1 毫秒的配置时间内，则指令会将 RTSMissed 状态位置位，以表明读取模块上输入的更新时出现问题。	

基于时间的指令需要 DeltaT 为常数值，以便控制算法正确计算过程输出。如果 DeltaT 发生变化，则在过程输出中将出现不连续现象。不连续的严重程度取决于指令和 DeltaT 的变化范围。

如果发生以下情况，则会出现不连续现象：

- 在扫描期间未执行指令。
- 在任务运行期间多次执行指令。
- 任务正在运行时，任务扫描速率或过程输入的采样时间发生变化。
- 用户在任务运行时更改时基模式。
- 任务运行时，Order 参数在过滤块上发生更改。
- 更改 Order 参数在指令中选择其他控制算法。

时序模式的通用指令参数

支持时基模式的指令具有以下输入和输出参数。

输入参数

输入参数	数据类型	说明
TimingMode	DINT	选择时序执行模式。 值：说明： 0 周期模式 1 过采样模式 2 实时采样模式 有效值 = 0 到 2 默认值 = 0 当 TimingMode = 0 且任务是周期性任务时，启用周期性时序，DeltaT 将设置为任务扫描速率。当 TimingMode = 0 且任务是事件任务或连续任务时，启用周期性时序，DeltaT 将设置为等于自上次执行指令后所经过的时间长度。 当 TimingMode = 1 时，启用过采样时序，DeltaT 将设置为 OversampleDT 参数的值。当 TimingMode = 2 时，启用实时采样时序，DeltaT 将是当前时戳值和先前时戳值（从输入相关的模块中读取）之差。 如果 TimingMode 无效，指令会将 Status 中的相应位置位。
OversampleDT	REAL	过采样时序的执行时间。用于 DeltaT 的值以秒为单位。如果 TimingMode = 1，则 OversampleDT = 0.0 将禁止控制算法的执行。如果该值无效，指令将设置 DeltaT = 0.0，并将 Status 中的相应位置位。 有效值 = 0 到 4194.303 秒 默认值 = 0.0

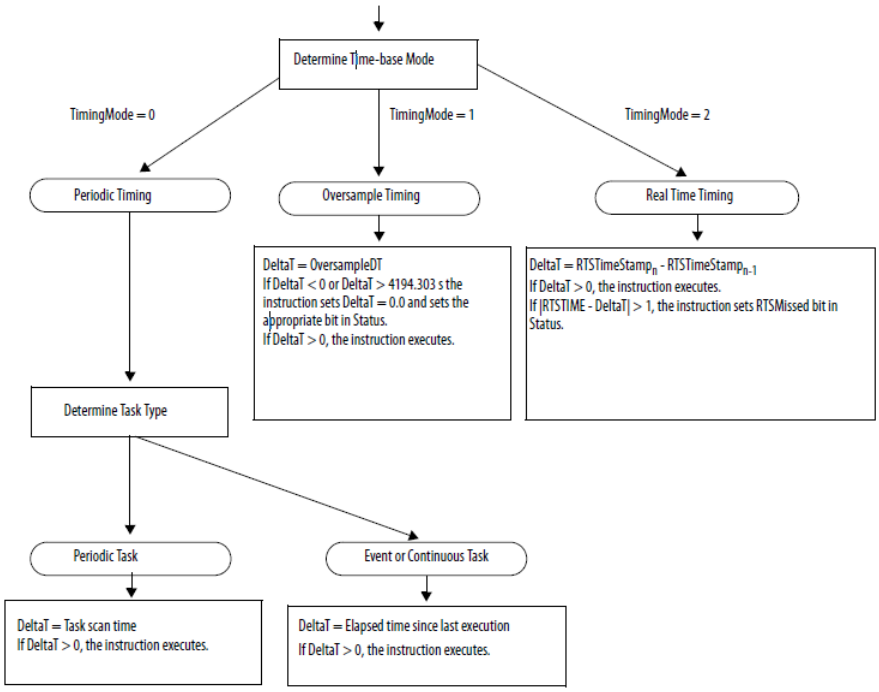
RTSTime	DINT	实时采样时序的模块更新周期。预期 DeltaT 更新周期以毫秒为单位。更新周期通常是用于配置模块更新时间的值。如果该值无效，指令会将 Status 中的相应位置位，并禁用 RTSMissed 检查。 有效值 = 1…32,767 ms 默认值 = 1
RTSTimeStamp	DINT	实时采样时序的模块时戳值。对应于输入信号的上次更新的时戳值。此值用于计算 DeltaT。如果该值无效，指令会将 Status 中的相应位置位，并禁用控制算法的执行和 RTSMissed 检查。 有效值 = 0…32,767 ms（从 32767 跳回到 0） 1 次计数 = 1 毫秒 默认值 = 0

输出参数

输出参数	数据类型	说明
DeltaT	REAL	两次更新间隔的时间。控制算法计算过程输出所用的时间（秒）。 周期性：如果任务是周期性任务，则 DeltaT = 任务扫描速率；如果任务是事件任务或连续任务，则 DeltaT = 自上次执行指令后所经过的时间 过采样：DeltaT = OversampleDT 实时采样：DeltaT = (RTSTimeStampn - RTSTimeStampn-1)
状态 (Status)	DINT	功能块的状态。
TimingModeInv (Status.27)	BOOL	TimingMode 值无效。
RTSMissed (Status.28)	BOOL	仅用于实时采样模式。当 ABS|DeltaT - RTSTime| > 1（0.001 秒）时置位。
RTSTimeInv (Status.29)	BOOL	RTSTime 值无效。
RTSTimeStampInv (Status.30)	BOOL	RTSTimeStamp 值无效。
DeltaTInv (Status.31)	BOOL	DeltaT 值无效。

时序模式概览

下图显示了指令如何确定适当的时序模式。



程序/操作员控制

以下指令支持程序/操作员控制的概念。

- 增强型选择（ESEL）
- 累加器（TOT）
- 增强型 PID (PIDE)
- 升温/持温（RMPS）
- 离散 2 态设备（D2SD）
- 离散 3 态设备（D3SD）

程序/操作员控制允许用户同时从用户程序和操作员界面设备执行对这些指令的控制。当处于程序控制时，指令由指令的程序输入来控制；当处于操作员控制时，指令由指令的操作员输入来控制。

程序或操作员控制通过使用以下输入来确定。

输入	说明
.ProgProgReq	进入程序控制的程序请求。

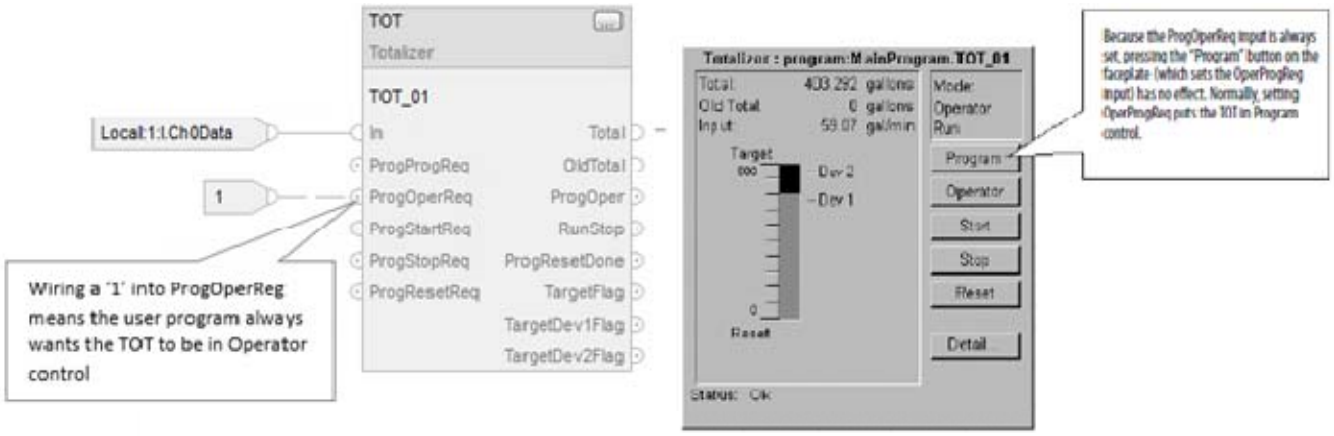
.ProgOperReq	进入操作员控制的程序请求。
.OperProgReq	进入程序控制的操作员请求。
.OperOperReq	进入操作员控制的操作员请求。

要确定指令是处于程序控制还是操作员控制，请检查 ProgOper 输出。如果 ProgOper 置位，则指令处于程序控制；如果 ProgOper 清零，则指令处于操作员控制。

如果两个输入请求位均置位，则操作员控制的优先级高于程序控制。例如，如果 ProgProgReq 和 ProgOperReq 均置位，则指令进入操作员控制。

程序请求输入的优先级高于操作员请求输入。因而可使用 ProgProgReq 和 ProgOperReq 输入将指令“锁定”在所需控制下。

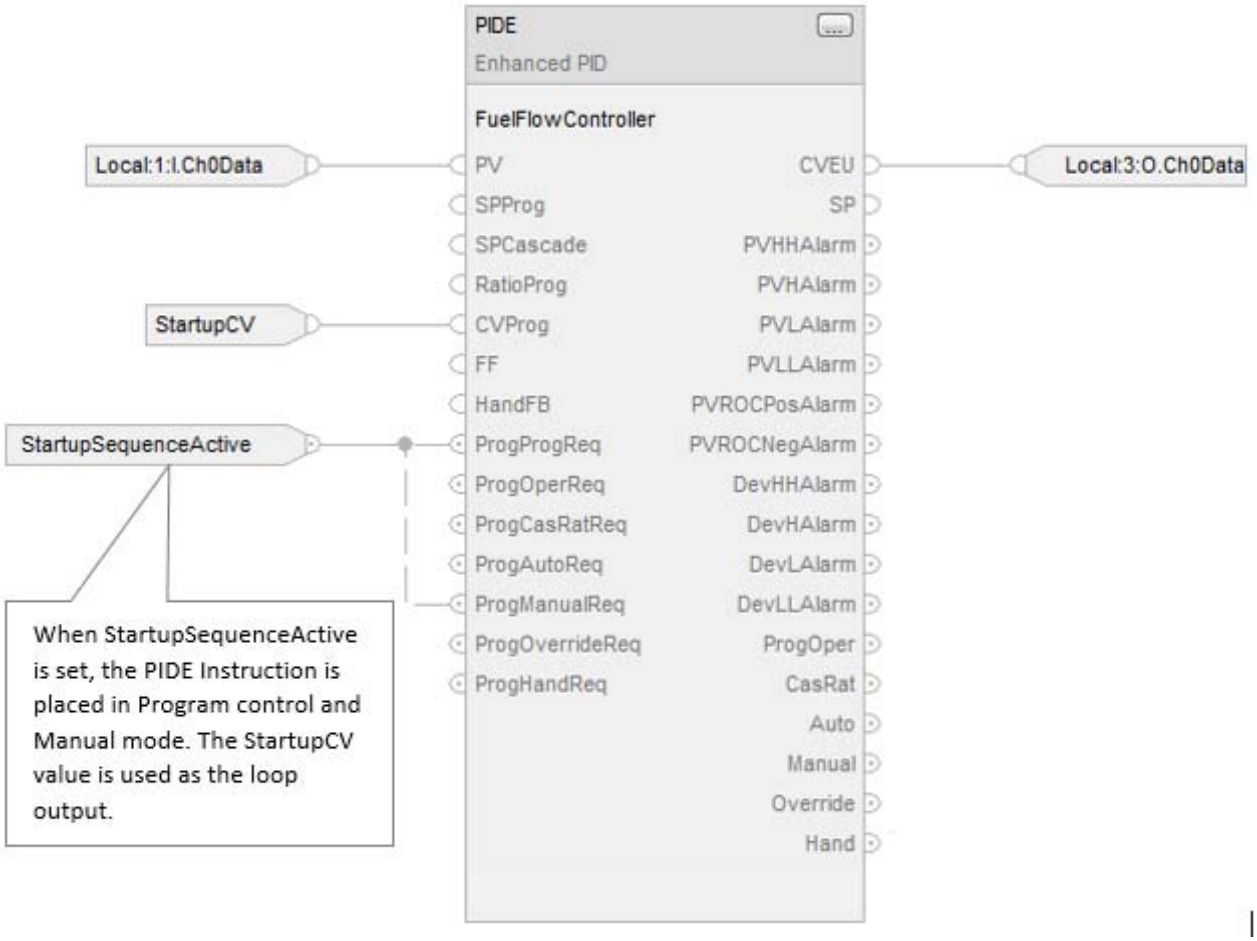
例如，假定一条累加器指令始终处于操作员控制下，并且用户程序始终不控制该累加器的运行或停止。在这种情况下，可将字面值 1 连接到 ProgOperReq。这将防止操作员通过从操作员界面设备将 OperProgReq 置位来将累加器置于程序控制下。



同样，始终将 ProgProgReq 置位可以将指令“锁定”在程序控制下。当用户想让程序控制指令的动作而无需担心操作员意外控制指令时，这对于自动启动序列十分有用。

在此示例中，用户在启动时使用程序来置位 ProgProgReq 输入，然后在启动完成后清零 ProgProgReq 输入。ProgProgReq 输入清零后，指令将保持在程序控制下，直到它接收到变更请求。例如，操作员可以通过面板置位 OperOperReq 输入以控制该指令。

以下示例显示了如何将指令锁定在程序控制下。

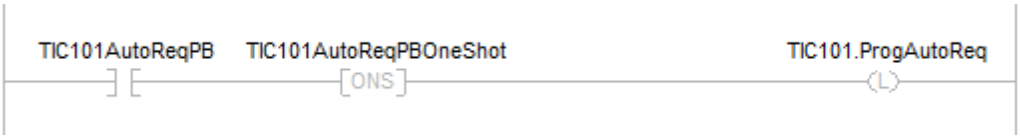


指令的操作员请求输入始终在指令执行时被指令清零。这使得操作员界面可以只通过置位所需模式请求位来使用这些指令。用户无需编程操作员界面来复位请求位。例如，如果操作员界面置位 PIDE 指令的 OperAutoReq 输入，当 PIDE 指令执行时，它将确定适当的响应并清零 OperAutoReq。

程序请求输入通常不由指令清零，因为它们通常作为输入连接到指令。如果指令清零这些输入，则输入将重新由所连接的输入置位。有可能出现这样的情况，用户想由指令来清零程序请求，从而使用其他逻辑来置位程序请求。在这种情况下，用户可以置位 ProgValueReset 输入，指令在执行时始终会清零程序模式请求输入。

在本示例中，另一例程中的梯形图逻辑梯级用于在按下按钮时以单脉冲触发形式闭锁 PIDE 指令的 ProgAutoReq。

按下 TIC101AutoReq 按钮后，将以单脉冲触发形式闭锁 PIDE 指令 TIC101 的 ProgAutoReq。TIC101 已经在 ProgValueReset 输入被置位的情况下配置。ProgAutoReq 会复位，因为 ProgValueReset 始终置位。



功能块状态

基于 Logix 的控制器根据不同条件的状态来评估功能块指令：

条件	说明
预扫描	功能块例程的预扫描与梯形图例程的相同。唯一的区别在于，每个功能块指令的 EnableIn 参数在预扫描期间清零。
指令首次扫描	指令首次扫描是指在预扫描后首次执行指令。控制器使用指令首次扫描来读取当前输入并确定要处于的适当状态。
指令首次运行	指令首次运行是指首次使用数据结构的新实例执行指令。控制器使用指令首次运行生成系数，以及在初次下载后对于功能块来说不发生变化的其他数据存储。

每个功能块指令还包括 EnableIn 和 EnableOut 参数：

- 功能块指令在 EnableIn 置位时正常执行。
- 当 EnableIn 清零时，功能块指令会执行预扫描逻辑、后扫描逻辑或仅跳过正常的算法执行。
- EnableOut 反映 EnableIn。但是，如果功能块检测到溢出条件，则 EnableOut 也会清零。
- 当 EnableIn 从清零切换为置位时，功能块将从停止的位置恢复。但是，当 EnableIn 从清零切换为置位时，还存在一些指定特殊功能（例如重新初始化）的功能块指令。对于具有时基参数的功能块指令，当 EnableIn 从清零切换为置位时，只要时序模式为过采样，指令便始终从停止的位置恢复。

如果 EnableIn 参数未连接，则指令始终正常执行，EnableIn 保持置位状态。如果将 EnableIn 清零，则它将在下次执行指令时切换为置位状态。

结构化文本编程

本章介绍结构化文本编程所特有的问题。查看以下主题以确保您理解结构化文本编程的执行方式。

结构化文本语法

结构化文本是一种文本编程语言，使用语句来定义要执行的内容。

- 结构化文本不区分大小写。
- 使用制表符和回车（单独的行）可以使结构化文本更容易阅读。它们对结构化文本的执行没有影响。

结构化文本不区分大小写。结构化文本可包含以下组成部分。

术语	定义	示例
赋值	使用赋值语句为标签赋值。运算符 := 为赋值运算符。 赋值语句以分号";"结束。	标签 := 表达式;

Expression	表达式是一个完整的赋值或结构语句的一部分。表达式可以求出一个数值（数值表达式）、一个字符串（字符串表达式）或者一个或真或假的状态（BOOL 表达式）	
标签表达式	已命名的存储器区域，用来存储数据（BOOL、SINT、INT、DINT、REAL 和字符串）。	value1
立即表达式	一个常数值	4
运算符表达式	一种用来指定表达式中运算的符号或助记符。	tag1 + tag2 tag1 >= value1
函数表达式	执行时，函数将产生一个值。使用括号将函数的操作数括起来。 即使函数和指令的语法类似，但它们却是不同的，函数只能用于表达式中。指令不能用于表达式中。	function(tag1)

指令	指令是一个独立语句。
	指令使用括号将其操作数括起来。
	根据指令的不同，可以有零个、一个或多个操作数。
	执行后，指令将生成一个或多个值，这些值是数据结构的一部分。
	指令以分号 (;) 结束。
	即使指令与函数的语法相似，但它们却是不同的，指令不能用于表达式中。函数只能用于表达式中。
		instruction();
		
		instruction(operand);
		
		instruction(operand1, operand2,operand3);

结构	用来触发结构化文本代码（即其他语句）的条件语句。结构以分号 (;) 结束。	IF...THEN CASE FOR...DO WHILE...DO REPEAT...UNTIL EXIT
注释	解释或阐明某段结构化文本的功能的文字。 使用注释可以更轻松地解读结构化文本。 注释对结构化文本的执行没有影响。 注释可以出现在结构化文本中的任何位置。	//注释 (*注释开始...注释结束*) /*注释开始...注释结束*/

另请参见

结构化文本组成部分：赋值 参考页数 561

结构化文本组成部分：表达式 参考页数 564

结构化文本组成部分：指令 参考页数 570

结构化文本组成部分：结构 参考页数 571

结构化文本组成部分：注释 参考页数 560

结构化文本组成部分：注释

为了使结构化文本更容易解读，请添加注释。

- 通过注释，可以使用普通语言来描述结构化文本的工作方式。

- 注释对结构化文本的执行没有影响。

给结构化文本添加注释：

要添加注释	使用以下格式之一
在单独一行上	//注释
在结构化文本某一行的末尾	(*注释*) /*注释*/
在结构化文本某一行当中	(*注释*) /*注释*/
跨越多行	(*注释开始...注释结束*) /*注释开始...注释结束*/

例如：

格式	示例

//注释	**在行的开头**
	//Check conveyor belt direction
	IF conveyor_direction THEN...
	在行的末尾
	ELSE //If conveyor isn't moving, set alarm light
	light := 1;
	END_IF;

(*注释*)	Sugar.Inlet[:=]1;(*open the inlet*) IF Sugar.Low (*low level LS*)& Sugar.High (*high level LS*)THEN... (*Controls the speed of the recirculation pump.The speed depends on the temperature in the tank.*) IF tank.temp > 200 THEN...
/*注释*/	Sugar.Inlet:=0;/*close the inlet*/ IF bar_code=65 /*A*/ THEN... /*Gets the number of elements in the Inventory array and stores the value in the Inventory_Items tag*/ SIZE(Inventory,0,Inventory_Items);

结构化文本组成部分：赋值

赋值语句可用于更改标签内存储的值。赋值采用以下语法：

标签 := 表达式;

其中：

组成部分	说明	
标签	表示获取新值的标签；标签必须为 BOOL、SINT、INT、DINT、STRING 或 REAL。 **提示：**STRING 标签仅适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器。	
:=	为赋值符号	
Expression	表示要赋值给标签的新值	
	如果标签为以下数据类型	**请使用以下类型的表达式**
	BOOL	BOOL
	SINT INT DINT REAL	数字
	STRING （只适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器）。	字符串类型，包括字符串标签和字符串字面值 （只适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器）。
;	结束赋值	

在其他赋值语句更改所赋的值之前，标签将一直保持该值。

表达式可以很简单（如立即数或其他标签名），也可以很复杂，包括多个运算符和/或函数。请参阅“表达式”以了解更多信息。

提示：I/O 模块数据与逻辑执行异步更新。如果在逻辑中多次引用某个输入，该输入可能会在不同的引用间更改状态。如果您需要在每次引用时，该输入的状态保持不变，则缓存输入值并引用该缓冲区标签。有关更多信息，请参见 LOGIX 5000 Controllers Common Procedures，出版号 1756-PM001。

还可以使用 Input 和 Output 程序参数，在逻辑执行期间自动缓存数据。请参见 LOGIX 5000 Controllers Program Parameters Programming Manual，出版号 1756-PM021。

另请参见

将 ASCII 字符赋值给字符串数据成员 参考页数 564

指定非保持型赋值 参考页数 563

结构化文本组成部分：表达式 参考页数 564

字符串字面值 参考页数 572

指定非保持型赋值

非保持型赋值不同于上述的常规赋值，控制器每次执行以下动作时，非保持型赋值语句中的标签都将复位为零：

- 进入运行模式
- 在配置 SFC 为自动复位的情况下，离开 SFC 的程序步。仅适用于在步动作中嵌入该赋值语句，或通过该动作使用 JSR 指令调用结构化文本例程时。

非保持型赋值采用以下语法：

标签 [:=] 表达式 ;

其中：

组成部分	说明	
标签	表示获取新值的标签；标签必须为 BOOL、SINT、INT、DINT、STRING 或 REAL。 **提示：**STRING 标签仅适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器。	
[:=]	为非保持型赋值符号。	
expression	表示要赋值给标签的新值。	
	如果标签为以下数据类型	**请使用以下类型的表达式**
	BOOL	BOOL
	SINT	数字
	INT	
	DINT	
	REAL	
	STRING（只适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器）。	字符串类型，包括字符串标签和字符串字面值 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器（只适用）

另请参见

将 ASCII 字符赋值给字符串数据成员 参考页数 564

结构化文本组成部分：赋值 参考页数 561

将 ASCII 字符赋值给字符串数据成员

将 ASCII 字符赋值给字符串数据成员

使用赋值运算符将 ASCII 字符赋给字符串标签的 DATA 成员的元素。要将字符赋给字符串，需要指定字符的值或指定标签名称、DATA 成员和字符的元素。例如：

可行情况	不可行情况
string1.DATA[0] := 65;	string1.DATA[0] := A;
string1.DATA[0]:= string2.DATA[0];	string1 := string2; **提示：**这会将 string2 的全部字符赋值给 string1，而不只是一个字符。

要将一串字符添加或插入到字符串标签中，请使用以下 ASCII 字符串指令中的一种：

目的	使用此指令
将字符添加到字符串末尾	CONCAT
将字符插入字符串内	INSERT

另请参见

结构化文本组成部分：表达式 参考页数 564

字符串字面值 参考页数 572

结构化文本组成部分：表达式

表达式是一个标签名称、等式或比较。要编写一个表达式，可使用以下内容：

- 用来存储值的标签名称（变量）
- 直接输入到表达式中的数字（立即 值）
- 直接输入到表达式中的字符串字面值（只适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器）
- 函数，如：ABS、TRUNC
- 运算符，如：+、-、<、>、And、Or

编写表达式时，需遵循以下指导原则：

- 表达式中可混合使用大小写字母。例如，“AND”可以表示为：AND、And、and。

- 对于较复杂的要求，可在表达式内使用圆括号组合多个表达式。这样可以提高整个表达式的可读性，并确保表达式按照期望的顺序执行。

将这些表达式用于结构化文本：

BOOL 表达式：生成 BOOL 值 1（真）或 0（假）的表达式。

- BOOL 表达式使用布尔标签、关系运算符和逻辑运算符来比较值或检查条件是真还是假。例如，tag1 > 65。
- 简单的布尔表达式可以是一个单一的 BOOL 标签。
- 通常，可以使用布尔表达式作为其他逻辑的执行条件。

数值表达式：计算整型值或浮点值的表达式。

- 数值表达式使用算术运算符、算术函数和按位运算符。例如，tag1+5。
- 在 BOOL 表达式内嵌套数值表达式。例如，(tag1+5)>65。

字符串表达式：表示字符串的表达式

- 简单的表达式可以是字符串字面值或字符串标签

使用下表选择表达式的运算符。

如果	使用
计算算术值	算术运算符和函数
比较两个值或字符串	关系运算符
确定条件的真或假	逻辑运算符
比较值内各位	按位运算符

另请参见

使用算术运算符和函数 参考页数 565

使用关系运算符 参考页数 568

使用逻辑运算符 参考页数 567

使用按位运算符 参考页数 567

使用算术运算符和函数

可以在算术表达式中组合多个运算符和函数。

运算符计算新值。

目的	使用以下运算符	最佳数据类型
加	+	DINT 和 REAL
减/求反	-	DINT 和 REAL
乘	*	DINT 和 REAL
指数（x 的 y 次幂）	**	DINT 和 REAL
除	/	DINT 和 REAL
取模除法	MOD	DINT 和 REAL

函数执行数学运算。为函数指定一个常量、非布尔型标签或表达式。

运算	使用以下函数	最佳数据类型
绝对值	ABS (numeric_expression)	DINT 和 REAL
反余弦	ACOS (numeric_expression)	REAL
反正弦	ASIN (numeric_expression)	REAL
反正切	ATAN (numeric_expression)	REAL
余弦	COS (numeric_expression)	REAL
弧度转角度	DEG (numeric_expression)	DINT 和 REAL
自然对数	LN (numeric_expression)	REAL
以 10 为底的对数	LOG (numeric_expression)	REAL
角度转弧度	RAD (numeric_expression)	DINT 和 REAL
正弦	SIN (numeric_expression)	REAL
平方根	SQRT (numeric_expression)	DINT 和 REAL
正切	TAN (numeric_expression)	REAL
截断	TRUNC (numeric_expression)	DINT 和 REAL

下表提供了使用算术运算符和函数的示例。

使用以下格式	示例	
	对于以下情况	写入
value1 operator value2	如果 gain_4 和 gain_4_adj 为 DINT 标签且要求： “将 15 与 gain_4 相加，并将结果存储到 gain_4_adj”	gain_4_adj := gain_4+15;
operator value1	如果 alarm 和 high_alarm 为 DINT 标签且要求： “将 high_alarm 求反并将结果存储在 alarm 中。”	alarm:= -high_alarm;

function(numeric_expression)	如果 overtravel 和 overtravel_POS 为 DINT 标签且要求：“计算 overtravel 的绝对值并将结果存储在 overtravel_POS 中。”	overtravel_POS := ABS(overtravel);
value1 operator (function((value2+value3)/2)	如果 adjustment 和 position 为 DINT 标签，sensor1 和 sensor2 为 REAL 标签且要求：“计算 sensor1 和 sensor2 的平均值，再取这个平均值的绝对值，然后将此绝对值加上 adjustment，并将结果存储在 position 中。”	position := adjustment + ABS((sensor1 + sensor2)/2);

另请参见

结构化文本组成部分：表达式 参考页数 564

使用按位运算符

按位运算符根据两个值操作一个值内的位。

下表概括介绍了按位运算符。

运算	使用以下运算符	最佳数据类型
按位与	&，AND	DINT
按位或	或者	DINT
按位异或	XOR	DINT
按位求补	NOT	DINT

以下是一个示例。

使用以下格式	示例	
	对于以下情况	使用
value1 operator value2	如果 input1、input2 和 result1 为 DINT 标签且要求：“计算 input1 和 input2 的按位结果。将结果存储在 result1 中。”	result1 := input1 AND input2;

另请参见

结构化文本组成部分：表达式 参考页数 564

使用逻辑运算符

逻辑运算符用于验证多个条件的真或假。逻辑运算的结果是一个 BOOL 值。

如果比较结果为	则结果为
真	1
假	0

使用下列逻辑运算符。

要执行以下比较	使用以下运算符	最佳数据类型
逻辑与	&，AND	BOOL
逻辑或	或者	BOOL
逻辑异或	XOR	BOOL
逻辑取反	NOT	BOOL

下表提供了使用逻辑运算符的示例。

使用以下格式	示例	
	对于以下情况	使用
BOOLtag	如果 photoeye 是 BOOL 标签且要求："如果 photoeye_1 开启，则..."	IF photoeye THEN...
NOT BOOLtag	如果 photoeye 是 BOOL 标签且要求："如果 photoeye 关闭，则..."	IF NOT photoeye THEN...
表达式 1 & 表达式 2	如果 photoeye 是 BOOL 标签，temp 是 DINT 标签，且要求："如果 photoeye 开启，且 temp 小于 100，则..."	IF photoeye & (temp<100) THEN...
表达式 1 OR 表达式 2	如果 photoeye 是 BOOL 标签，temp 是 DINT 标签，且要求："如果 photoeye 开启，或 temp 小于 100，则..."	IF photoeye OR (temp<100) THEN...
表达式 1 XOR 表达式 2	如果 photoeye1 和 photoeye2 均为 BOOL 标签且要求："如果： photoeye1 开启且 photoeye2 关闭，或 photoeye1 关闭且 photoeye2 开启 则..."	IF photoeye1 XOR photoeye2 THEN...
BOOLtag := 表达式 1 & 表达式 2	如果 photoeye1 和 photoeye2 均为 BOOL 标签，open 是 BOOL 标签，且要求："如果 photoeye1 和 photoeye2 均开启，则将 open 设置为真"	open := photoeye1 & photoeye2;

另请参见

结构化文本组成部分：表达式 参考页数 564

使用关系运算符

关系运算符可以对两个值或字符串进行比较并得出一个或真或假的结果。关系运算的结果是一个 BOOL 值。

如果比较结果为	则结果为
真	1
假	0

使用下列关系运算符。

要执行以下比较	使用以下运算符	最佳数据类型
等于	=	DINT、REAL、字符串类型
小于	<	DINT、REAL、字符串类型
小于等于	<=	DINT、REAL、字符串类型
大于	>	DINT、REAL、字符串类型
大于等于	>=	DINT、REAL、字符串类型
不等于	<>	DINT、REAL、字符串类型

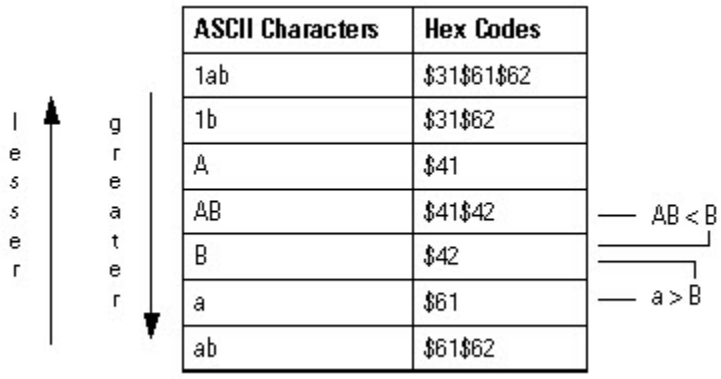
下表提供了使用关系运算符的示例

使用以下格式	示例	
	对于以下情况	写入
value1 operator value2	如果 temp 为 DINT 标签且要求：“如果 temp 小于 100，则…...”	IF temp<100 THEN...
stringtag1 operator stringtag2	如果 bar_code 和 dest 为字符串标签且要求：“如果 bar_code 等于 dest，则…”	IF bar_code=dest THEN...
stringtag1 operator 'character string literal'	如果 bar_code 为字符串标签且要求：“如果 bar_code 等于'Test PASSED'，则...”	IF bar_code='Test PASSED' THEN...
char1 operator char2 要直接在表达式中输入 ASCII 字符，请输入该字符的十进制值。	如果 bar_code 为字符串标签且要求：“如果 bar_code.DATA[0] 等于'A'，则…”	IF bar_code.DATA[0]=65 THEN...
bool_tag := bool_expressions	如果 count 和 length 为 DINT 标签，done 为 BOOL 标签且要求：“如果 count 大于等于 length，则表示完成计数。”	Done := (count >= length);

字符串的计算方式

ASCII 字符的十六进制值确定一个字符串是小于还是大于另一个字符串。

- 两个字符串按照电话号码簿方式排序时，它们的大小由字符串的顺序决定。



- 如果其字符匹配，则字符串相等。
- 字符区分大小写。大写“A” ($41) 不等于小写“a” ($61)。

另请参见

结构化文本组成部分：表达式 参考页数 564

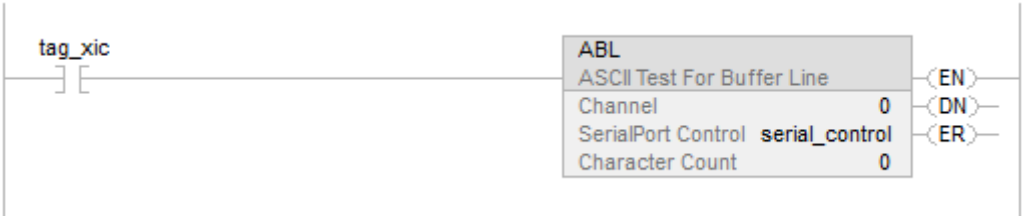
结构化文本组成部分：指令

结构化文本语句也可以是指令。结构化文本指令在每次被扫描的时候执行。每次结构的条件为真的时，执行结构中的结构化文本指令。如果结构的条件为假，不会对该结构内的语句进行扫描。没有触发执行的梯级条件或者状态转换。

这与使用 EnableIn 触发执行的功能块指令不同。结构化文本指令的执行就像 EnableIn 总是置位一样。

这也不同于使用梯级输入条件触发执行的梯形图指令。一些梯形图指令只在梯级输入条件由假切换为真时执行。这些是梯形图跳变指令。在结构化文本中，指令将在每次被扫描到的时候执行，除非用户预先设定该结构化文本指令的执行条件。

例如，ABL 指令是梯形图中的一个跳变指令。在本示例中，只在扫描到 tag_xic 从清零状态转变为置位状态时执行 ABL 指令。当 tag_xic 保持置位状态或清零时，不执行 ABL 指令。



在结构化文本中，如果将本示例编写为：

```
IF tag_xic THEN ABL(0,serial_control);

END_IF;
```

则将在每次扫描到 tag_xic 置位时执行 ABL 指令，而不只在 tag_xic 从清空状态转变为置位时执行。

如果要让 ABL 指令只在 tag_xic 从清零转变为置位状态时执行，则必须限制结构化文本指令的执行条件。使用单脉冲触发来触发执行。

```
osri_1.InputBit := tag_xic;

OSRI(osri_1);


IF (osri_1.OutputBit) THEN

ABL(0,serial_control);

END_IF;
```

结构化文本组成部分：结构

结构可单独编程，也可嵌套在其他结构中。

如果	使用以下结构
在满足特定条件时，执行某些操作	IF.。.THEN
根据某个数值选择要执行的操作	CASE...OF
在执行其他操作前，将某项操作执行指定次数	FOR...DO
只要某些条件为真就一直执行某项操作	WHILE...DO
一直执行某项操作，直到条件为真	REPEAT...UNTIL

一些关键字保留

下面的结构不可用：

- GOTO
- REPEAT

Logix Designer 不允许使用它们作为标签名称或结构。

另请参见

IF_THEN 参考页数 580

CASE_OF 参考页数 574

FOR_DO 参考页数 576

WHILE_DO 参考页数 585

REPEAT_UNTIL 参考页数 583

字符串字面值

字符串字面值包括单字节或双字节编码的字符。单字节字符串字面值是用单引号 (') 括上的零个或多个字符组成的序列。在单字节字符串中，美元符号 ($) 后跟两个十六进制数的三字节组合，被解释为八位字符代码的十六进制数表示，如下表所示。

提示：

- 字符串字面值仅适用于 CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器。
- Studio 5000 只支持单字节字符。

字符串字面值

编号	说明	示例
1a	空字符串（长度为零）	''
1b	长度为一的字符串或只包含一个字符的 CHAR 字符	'A'
1c	长度为一的字符串或只包含一个“空格”字符的 CHAR 字符	' '
1d	长度为一的字符串或只包含一个“单引号”字符的 CHAR 字符	'$''
1e	长度为一的字符串或只包含一个“双引号”字符的 CHAR 字符	'"'
1f	支持两个字符的组合	'RL'
1g	支持由 '$' 和两个十六进制字符构成的字符表示	'$0A'

字符串中的两个字符组合

编号	说明	示例
1	美元符号	$$
2	单引号	$'

3	换行	$L 或 $l
4	新行	$N 或 $n
5	换页（页）	$P 或 $p
6	回车	$R 或 $r
7	制表符	$T 或 $t

提示：
- 新行字符为物理和文件 I/O 提供了独立实现定义数据行结尾的方法；打印时的效果是，结束一个数据行并在下一行开头恢复打印。
- $' 组合仅在单引号括起的字符串字面值中有效。

另请参见

结构化文本组成部分：赋值 参考页数 561

字符串类型 参考页数 573

字符串类型

可在使用字符串类型数据类型的标签中存储 ASCII 字符，以：

- 使用最多可存储 82 个字符的默认 STRING 数据类型
- 创建可存储较少或较多字符的全新字符串类型

要创建新的字符串类型，请参见 LOGIX 5000 Controllers ASCII Strings Programming Manual（出版号 1756-PM013）。

每个字符串类型包含以下成员：

名称	数据类型	说明	备注
LEN	DINT	字符串中的字符数	在以下情况下，LEN 会自动更新为新字符数： • 使用字符串浏览器输入字符 • 使用读取、转换或操作字符串的指令 LEN 显示当前字符串的长度。DATA 成员可能包含不包括在 LEN 计数中的其他旧字符。
DATA	SINT array	字符串的 ASCII 字符	要访问字符串的字符，需按标签名称寻址。例如，要访问 string_1 标签的字符，请输入 string_1。 DATA 数组的每个元素都包含一个字符。 创建可存储较少或较多字符的全新字符串类型。

另请参见

字符串字面值 参考页数 572

CASE_OF

使用 CASE_OF 可根据数值选择要执行的操作。

操作数

CASE numeric_expression OF

选择器 1: 语句;

选择器 N: 语句; ELSE

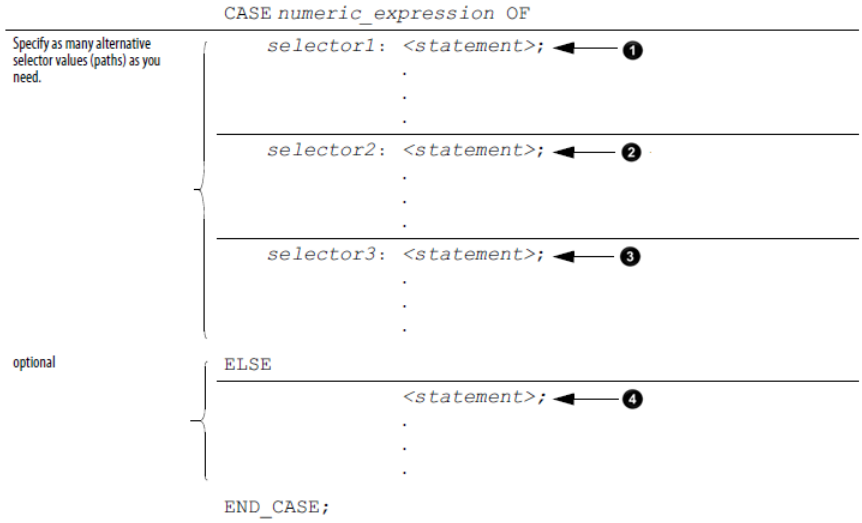
结构化文本

操作数	类型	格式	输入
Numeric_ expression	SINT INT DINT REAL	标签表达式	赋值为数值的标签或表达式（数值表达式）
Selector	SINT INT DINT REAL	立即数	与 numeric_expression 的类型相同

重要事项：	REAL 值极有可能位于一定的值范围内，而不是与某个特定值精确匹配，因此如果使用 REAL 值，就必须为选择器设定一个值范围。

说明

语法在下表中说明。



下面是输入选择器值时所用的语法。

当选择器为	输入
一个值	值: 语句
多个不同的值	value1, value2, valueN : <语句> 使用逗号 (,) 分隔每个值。
值范围	value1..valueN : <语句> 使用两个句点 (..) 来标识范围。
多个不同的值加上值范围	valuea, valueb, value1..valueN : <语句>

CASE 结构类似于 C 或 C++ 编程语言中的 switch 语句。对于 CASE 结构，控制器只执行与第一个匹配的选择器值关联的语句。总是在该选择器的语句后中断执行，并转到 END_CASE 语句。

影响数学状态标志

无

严重/轻微故障

无

示例

如果要实现以下要求	输入以下结构化文本
如果配方号 = 1，则成分 A 出口 1 = 打开 (1) 成分 B 出口 4 = 打开 (1)	CASE recipe_number OF
	1: Ingredient_A.Outlet_1 :=1; Ingredient_B.Outlet_4 :=1;
如果配方号 = 2 或 3，则 成分 A 出口 4 = 打开 (1) 成分 B 出口 2 = 打开 (1)	2,3: Ingredient_A.Outlet_4 :=1; Ingredient_B.Outlet_2 :=1;
如果配方号 = 4、5、6 或 7，则成分 A 出口 4 = 打开 (1) 成分 B 出口 2 = 打开 (1)	4 到 7：Ingredient_A.Outlet_4 :=1; Ingredient_B.Outlet_2 :=1;
如果配方号 = 8、11、12 或 13，则成分 A 出口 1 = 打开 (1) 成分 B 出口 4 = 打开 (1)	8,11...13 Ingredient_A.Outlet_1 :=1; Ingredient_B.Outlet_4 :=1;
否则所有出口 = 关闭 (0)	ELSE
	Ingredient_A.Outlet_1 [:=]0; Ingredient_A.Outlet_4 [:=]0; Ingredient_B.Outlet_2 [:=]0; Ingredient_B.Outlet_4 [:=]0;
	END_CASE;

每当控制器发生以下情况时，[:=] 都会指示控制器清空出口标签：

进入运行模式。

在配置 SFC 为自动复位的情况下，离开 SFC 的程序步。这仅适用于在步动作中嵌入该赋值语句，或通过该动作使用 JSR 指令调用结构化文本例程。

FOR_DO

FOR_DO 循环用于在执行其他操作前将某项操作执行指定次数。

使能后，FOR 指令重复执行例程，直到 Index 值超出 Terminal value 。步长值可以是正值，也可以是负值。如果是负数，则当索引值小于终止值时，循环结束..如果是正数，则当索引值大于终止值时，循环结束。

每次 FOR 指令执行例程时，都会向 Index 值加上 Step size 值。

在一个扫描周期内，循环次数不应过多。重复次数过多会导致控制器的看门狗超时，进而引发严重故障。

操作数

FOR count:= initial_value TO

final_value BY increment DO

<语句>;

END_FOR;

操作数	类型	格式	说明
count	SINT INT DINT	标签	FOR_DO 执行时，存储计数位置的标签
initial_ value	SINT INT DINT	标签表达式 立即数	必须赋值为一个数 指定计数的初始值
final_ value	SINT INT DINT	标签表达式 立即数	指定计数的最终值，该值确定何时退出循环
increment	SINT INT DINT	标签表达式 立即数	可选）循环一次时计数值的增量 如果不指定增量，则计数递增 1。

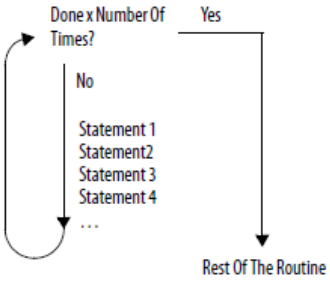
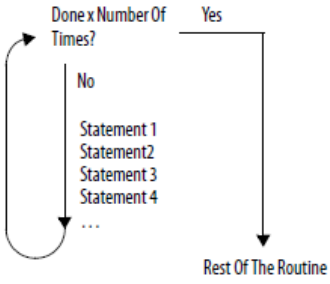
重要事项：	在一个扫描周期内，循环的迭代次数不应过多。 控制器在完成循环前不会执行例程中的其他语句。 如果完成循环所用的时间超过了任务的看门狗计时器时间，将出现严重故障。 请考虑使用其他结构，例如 IF_THEN。

说明

语法在下表中说明。

```
FOR count := initial_value
   TO final_value
optional [ BY increment                    ] If you don't specify an increment, the loop increments by 1.
   DO
   <statement>;
optional [ IF bool_expression THEN         ]
            EXIT;                          ] <-- If there are conditions when you want to exit the loop early, use other statements, such as an IF...THEN construct, to condition an EXIT statement.
   END_IF;                                 ]
END_FOR;
```

下图解释了 FOR_DO 循环的执行过程以及如果通过 EXIT 语句提前退出循环。

	
FOR_DO 循环执行特定次数。	要在计数达到最终值前终止循环，请使用 EXIT 语句。

影响数学状态标志

无

严重/轻微故障

在以下情况下会发生严重故障	故障类型	故障代码
结构循环过长。	6	1

示例 1

如果执行以下操作，	输入以下结构化文本

将 BOOL 数组的位 0…31 清零：
将 subscript 标签初始化为 0。
清除 i。例如，当 subscript = 5 时，清除 array[5]。
将 subscript 加 1。
如果 subscript ≤ 31，重复步骤 2 和 3。
否则将停止。
	For subscript:=0 to 31 by 1 do
	array[subscript] := 0;
	End_for;

示例 2

如果执行以下操作，	输入以下结构化文本

通过用户自定义数据类型（结构）存储以下库存货物信息：
• 货物的条形码 ID（字符串数据类型）
• 货物的库存数量（DINT 数据类型）
如上结构的数组包含了库存中各个不同货物的元素。用户想要搜索特定产品的数组（使用条形码），并确定库存数量。
1. 获取 Inventory 数组的大小（货物数），并将结果存储到
2. Inventory_Items（DINT 标签）中。
将 position 标签初始化为 0。
3. 如果 Barcode 与数组中某条目的 ID 匹配，则：
设置 Quantity 标签 = Inventory[position].Qty。这将生成该货物的库存量。
停止。
Barcode 是字符串标签，用于存储所搜索货物的条形码。例如，当
position = 5 时，将 Barcode 与 Inventory[5].ID 进行比较。
4. 将 position 加 1。
5. 如果 position ≤ (Inventory_Items -1)，重复步骤 3 和 4。由于元素编号起始于 0，所以最后一个元素的编号比数组的元素数小 1。
否则将停止。
	SIZE(Inventory,0,Inventory_Items);
	For position:=0 to Inventory_Items - 1 do
	If Barcode = Inventory[position].ID then
	Quantity := Inventory[position].Qty;
	Exit;
	End_if;
	End_for;

IF_THEN

IF_THEN 用于在满足特定条件时执行某项操作。

操作数

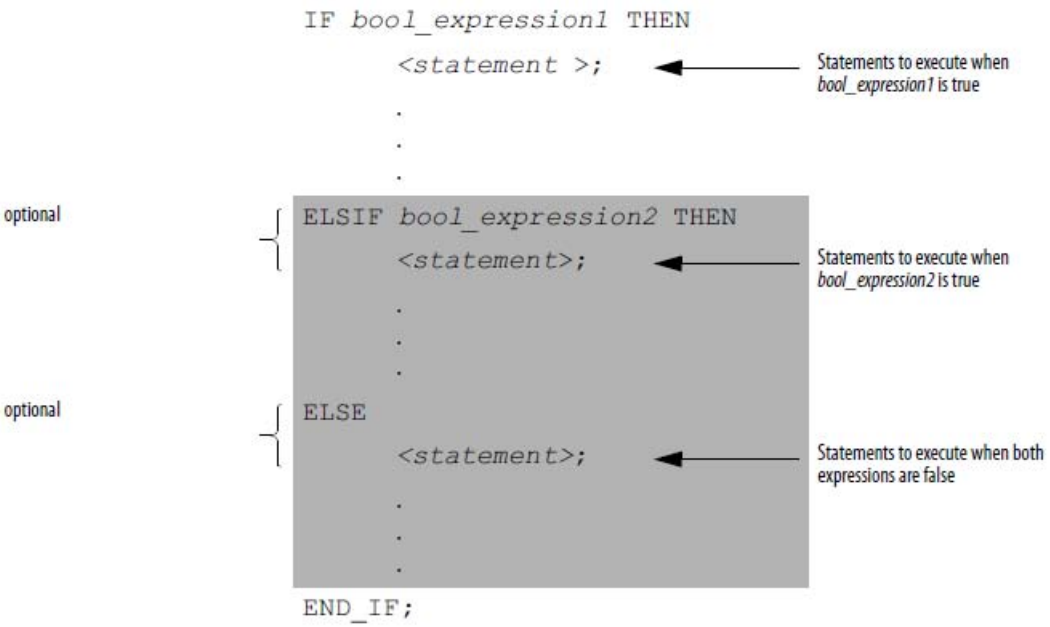
IF bool_expression THEN

<语句>;

操作数	类型	格式	输入
Bool_ expression	BOOL	标签表达式	赋值为 BOOL 值的 BOOL 标签或表达式（BOOL 表达式）

说明

语法在下表中说明。



根据以下原则使用 ELSIF 或 ELSE。

要从多个可能的语句组中选择，添加一个或多个 ELSIF 语句。

每个 ELSIF 表示一个候选路径。

根据需要指定多个 ELSIF 路径。

控制器执行首个为真的 IF 或 ELSIF，并跳过其余的 ELSIF 和 ELSE。

要在所有 IF 或 ELSIF 条件为假时执行某项操作，则添加一条 ELSE 语句。

下表汇总了 IF、THEN、ELSIF 和 ELSE 语句的不同组合。

如果	且	使用以下结构
当条件为真时，执行某项操作	如果条件为假，则不执行操作	IF_THEN
	如果条件为假，则执行其他操作	IF_THEN_ELSE
根据输入条件，从多个候选语句或语句组中进行选择	如果条件为假，则不执行操作	IF_THEN_ELSIF
	如果所有条件均为假 ,则分配默认语句	IF_THEN_ELSIF_ELSE

影响数学状态标志

无

严重/轻微故障

无。

示例

示例 1

IF...THEN

如果执行此操作	输入以下结构化文本
如果废品数 > 3 则	IF rejects > 3 THEN
传送带 = 关闭 (0)	conveyor := 0;
报警 = 开启 (1)	alarm := 1;
	END_IF;

示例 2

IF_THEN_ELSE

如果执行此操作	输入以下结构化文本
如果传送带方向触头 = 正向 (1)，则	IF conveyor_direction THEN
指示灯 = 灭	light := 0;
否则指示灯 = 亮	ELSE
	light [:=] 1;
	END_IF;

每当控制器发生以下情况时，[:=] 都会指示控制器将 light 清空：

进入运行模式。

在配置 SFC 为自动复位的情况下，离开 SFC 的程序步。（仅适用于在步动作中嵌入该赋值语句，或通过该动作使用 JSR 指令调用结构化文本例程时。）

示例 3

IF...THEN...ELSIF

如果执行此操作	输入以下结构化文本
如果糖料低限位开关 = 低（接通）且糖料高限位开关 = 不高 （接通），则	IF Sugar.Low & Sugar.High THEN
进给阀 = 打开（导通）	Sugar.Inlet [:=] 1;
直至糖料高限位开关 = 高（断开）	ELSIF NOT(Sugar.High) THEN
	Sugar.Inlet := 0;
	END_IF;

每当控制器发生以下情况时，[:=] 都会指示控制器将 Sugar.Inlet 清空：

进入运行模式。

在配置 SFC 为自动复位的情况下，离开 SFC 的程序步。（仅适用于在步动作中嵌入该赋值语句，或通过该动作使用 JSR 指令调用结构化文本例程时。）

示例 4

IF...THEN...ELSIF...ELSE

如果执行此操作	输入以下结构化文本
如果罐温度 > 100	IF tank.temp > 200 THEN
则泵 = 缓慢运转	pump.fast :=1; pump.slow :=0; pump.off :=0;
如果罐温度 > 200	ELSIF tank.temp > 100 THEN
则泵 = 快速运转	pump.fast :=0; pump.slow :=1; pump.off :=0;
否则泵 = 关闭	ELSE
	pump.fast :=0; pump.slow :=0; pump.off :=1;
	END_IF;

REPEAT_UNTIL

REPEAT_UNTIL 循环用于一直执行某项操作，直到条件为真。

操作数

REPEAT

<语句>;

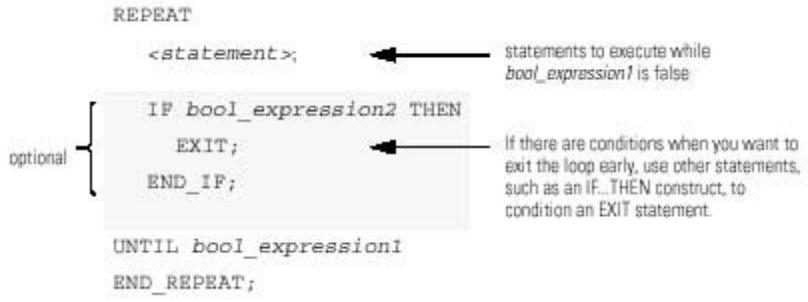
结构化文本

操作数	类型	格式	输入
bool_ expression	BOOL	标签表达式	赋值为 BOOL 值的 BOOL 标签或表达式（BOOL 表达式）

重要事项： 在一个扫描周期内，循环的迭代次数不应过多。
控制器在完成循环前不会执行例程中的其他语句。
如果完成循环所用的时间超过了任务的看门狗计时器时间，将出现严重故障。
请考虑使用其他结构，例如 IF_THEN。

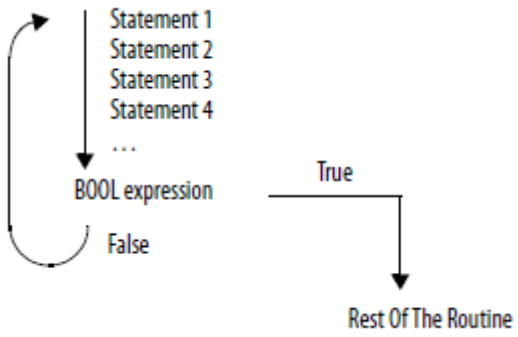
说明

语法如下：

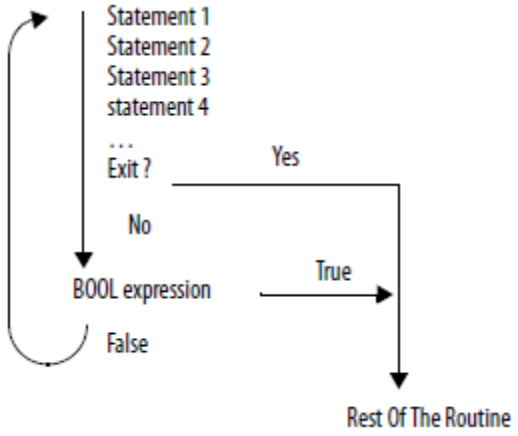


下图显示了 REPEAT_UNTIL 循环如何执行以及 EXIT 语句如何提前退出循环。

当 bool_expression 为假时，控制器仅执行 REPEAT_UNTIL 循环内的语句。



要在条件为假前终止循环，请使用 EXIT 语句。



影响数学状态标志

无

故障条件

在以下情况下会发生严重故障	故障类型	故障代码
结构循环过长	6	1

示例 1

如果执行以下操作，	输入以下结构化文本
REPEAT_UNTIL 循环首先执行结构内的语句，然后在再次执行	pos := -1;
	REPEAT

语句前确定条件是否为真。该循环不同于 WHILE_DO 循环，因为 WHILE_DO 循环将首先对其条件进行求值。 如果条件为真 ,则控制器执行循环内的语句。REPEAT_UNTIL 循环中的语句始终至少执行一次。而 WHILE_DO 循环中的语句可能从不执行。	pos := pos + 2; UNTIL ((pos = 101) OR (structarray[pos].value = targetvalue)) end_repeat;

示例 2

如果执行以下操作，	输入以下结构化文本

将 SINT 数组中的 ASCII 字符移入字符串标签。（在 SINT 数组中，每个元素保存一个字符。）到达回车时停止。
将 Element_number 初始化为 0。
计算 SINT_array（含有 ASCII 字符的数组）中的元素数，并将结果存入 SINT_array_size（DINT 标签）中。
设置 String_tag[element_number] = SINT_array[element_number] 中的字符。
将 element_number 加 1。这会使控制器检查 SINT_array 中的下一个字符。
设置 String_tag 的长度成员 = element_number。（其将记录当前 String_tag 中的字符数。）
如果 element_number = SINT_array_size，则停止。（已执行到数组末尾但无回车。）
如果 SINT_array[element_number] 中的字符 = 13（回车的十进制值），则停止。
	element_number := 0;
	SIZE(SINT_array, 0, SINT_array_size);
	Repeat
	String_tag.DATA[element_number] := SINT_array[element_number];
	element_number := element_number + 1;
	String_tag.LEN := element_number;
	If element_number = SINT_array_size then
	exit;
	end_if;
	Until SINT_array[element_number] = 13
	end_repeat;

WHILE_DO

WHILE_DO 循环用于在某些条件为真时一直执行某项操作。

操作数

WHILE bool_expression DO

<语句>;

结构化文本

操作数	类型	格式	说明
bool_expression	BOOL	标签 expression	赋值为 BOOL 值的 BOOL 标签或表达式

重要事项： 在一个扫描周期内，循环的迭代次数不应过多。

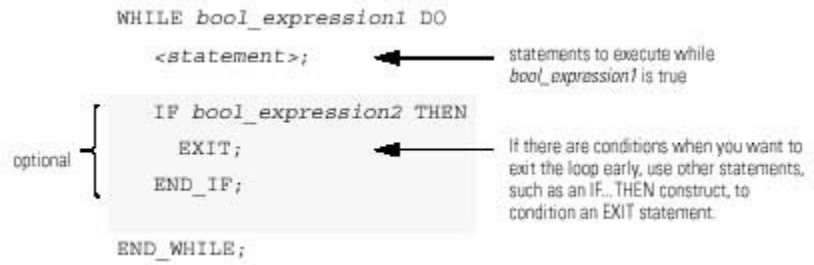
控制器在完成循环前不会执行例程中的其他任何语句。

如果完成循环所用的时间超过了任务的看门狗计时器时间，将出现严重故障。

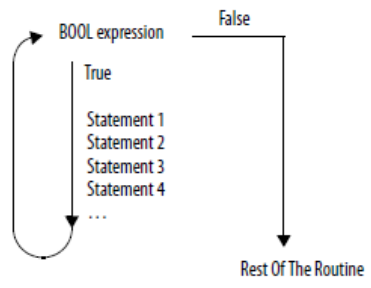
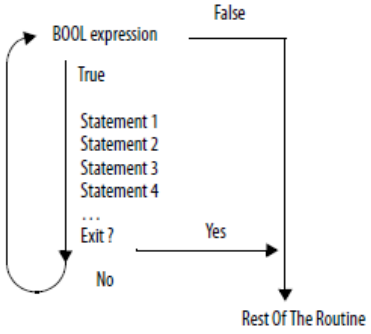
请考虑使用其他结构，例如 IF_THEN。

说明

语法如下：



下图解释了 WHILE_DO 循环的执行过程以及如何通过 EXIT 语句提前退出循环。

	
当 bool_expression 为真时，控制器只执行 WHILE_DO 循环内的语句。	要在条件为真前终止循环，请使用 EXIT 语句。

影响数学状态标志

无

故障条件

在以下情况下会发生严重故障	故障类型	故障代码
结构循环过长	6	1

示例 1

如果执行以下操作，	输入以下结构化文本

WHILE_DO 循环首先评估其条件。如果条件为真，则控制器执行循环内的语句。
该循环不同于 REPEAT_UNTIL 循环，因为 REPEAT_UNTIL 循环首先执行结构内的语句，然后在再次执行语句前确定条件是否为真。REPEAT_UNTIL 循环中的语句始终至少执行一次。而 WHILE_DO 循环中的语句可能从不执行。
	pos := 0;
	While ((pos <= 100) & structarray[pos].value <> targetvalue)) do
	  pos := pos + 2;
	  String_tag.DATA[pos] := SINT_array[pos];
	end_while;

示例 2

如果执行以下操作，	输入以下结构化文本

将 SINT 数组中的 ASCII 字符移入字符串标签。（在 SINT 数组中，每个元素保存一个字符。）到达回车时停止。
将 Element_number 初始化为 0。
计算 SINT_array（含有 ASCII 字符的数组）中的元素数，并将结果存入 SINT_array_size（DINT 标签）中。
如果 SINT_array[element_number] 中的字符 = 13（回车的十进制值），则停止。
设置 String_tag[element_number] = SINT_array[element_number] 中的字符。
将 element_number 加 1。这会使控制器检查 SINT_array 中的下一个字符。
设置 String_tag 的长度成员 = element_number。（其将记录当前 String_tag 中的字符数。）
如果 element_number = SINT_array_size，则停止。（已执行到数组末尾但无回车。）
	element_number := 0;
	SIZE(SINT_array, 0, SINT_array_size);
	While SINT_array[element_number] <> 13 do
	String_tag.DATA[element_number] := SINT_array[element_number];
	element_number := element_number + 1;
	String_tag.LEN := element_number;
	If element_number = SINT_array_size then
	exit;
	end_if;
	end_while;

结构化文本属性

有关结构化文本编程特有的问题的更多信息，请单击下方的相应主题。请认真阅读此信息，以确保您了解结构化文本编程的执行方式。

另请参见

结构化文本组成部分：赋值 参考页数 561

结构化文本组成部分：表达式 参考页数 564

结构化文本指令 参考页数 570

结构化文本组成部分：结构 参考页数 571

结构化文本组成部分：注释 参考页数 560

高级过程控制和驱动指令的通用属性

请遵从本章有关高级过程控制和驱动指令通用属性的原则。

通用属性

要了解 LOGIX 5000™ 指令通用属性的更多信息，请单击以下任一主题。

数学状态标志

数学状态标志的确定应遵循本主题所述的原则。

说明

控制器	说明
CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器	一组可通过指令直接访问的数学状态标志。这些标志仅能在梯形图例程中更新，它们并非标签，因此不适用别名。
CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370 和 GuardLogix 5570 控制器	一组可通过指令直接访问的数学状态标志。这些标志可在所有类型的图例程中更新，但它们并非标签，因此不适用别名。

状态标志

状态标志	说明 （CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器）	说明 （CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370 和 GuardLogix 5570 控制器）
S:FS 首次扫描标志	以下情况下，控制器将首次扫描标志置位： • 控制器进入运行模式后首次对程序进行扫描 • 在程序解除禁用后首次进行扫描 • 从 SFC 操作调用某个例程后，首次扫描该操作所属的程序步。 可以借助首次扫描标志来初始化数据，以便在后续扫描中使用。这也称为首个传递位。	以下情况下，控制器将首次扫描标志置位： • 控制器进入运行模式后首次对程序进行扫描 • 在程序解除禁用后首次进行扫描 • 从 SFC 操作调用某个例程后，首次扫描该操作所属的程序步。 可以借助此标志来初始化数据，以便在后续扫描中使用。这也称为首个传递位。
S:N 负值标志	当算术或逻辑运算的结果为负值时，控制器将负值标志置位。可借助此标志来快速测试负值。	当算术或逻辑运算的结果为负值时，控制器将负值标志置位。可借助此标志来快速测试负值。 相比 CMP 指令，使用 S:N 的效率更高。
S:Z 零标志	当算术或逻辑运算的结果为零时，控制器将零标志置位。可借助此标志来快速测试零值。 启动可设置该标志的指令时，即可清除零标志。	当算术或逻辑运算的结果为零值时，控制器将零值标志置位。可借助此标志来快速测试零值。
S:V 溢出标志	控制器在以下情况下将溢出标志置位： • 算术运算的结果导致溢出。 例如，当值范围为 127 到 -128 时，SINT 加 1 即可导致溢出。 • 目标标签过小，无法容纳数值。 例如，尝试将值 123456 存储到 SINT 或 INT 标签中。 可以使用溢出标志来检查运算结果是否超出范围。 如果存储的数据为字符串型，若字符串过长而无法填入目标标签，S:V 将置位。 **提示：**如果适用，可以使用 OTE 或 OTL 指令来设置 S:V。 单击**控制器属性 > "高级"选项卡 > 报告溢出故障**（Controller Properties > Advanced tab > Report Overflow Faults）来启用或禁用报告溢出故障。 如果在计算数组下标时发生溢出，将产生一个轻微故障和一个严重故障（指示索引超出范围）。	控制器在以下情况下将溢出标志置位： • 算术运算的结果导致溢出。 例如，当值范围为 127…-128 时，SINT 加 1 即可导致溢出。 • 目标标签过小，无法容纳数值。 例如，尝试将值 123456 存储到 SINT 或 INT 标签中。 可以使用溢出标志来检查运算结果是否超出范围。 每次溢出标志置位时都会产生一个轻微故障。 **提示：**如果适用，可以使用 OTE 或 OTL 指令来设置 S:V。

S:C 进位标志	当算术运算结果导致最高有效位进位时，控制器将进位标志置位。 只有针对整型值使用 ADD 和 SUB 指令（非 +/- 运算符）时，才会影响此标志。	当算术运算结果导致最高有效位进位时，控制器将进位标志置位。
S:MINOR 轻微故障标志	当发生至少一个轻微程序故障时，控制器将轻微故障标志置位。 可以使用轻微故障标签来测试是否发生了轻微故障。该位仅能由编程故障触发（例如溢出），而不会由电池故障触发。每次扫描开始时都将清除该位。 **提示：**如果适用，可以使用 OTE 或 OTL 指令来显式设置 S:MINOR。	当发生至少一个轻微程序故障时，控制器将轻微故障标志置位。 可使用轻微故障标签来测试是否发生了轻微故障并采取适当措施。该位仅能由编程故障触发（例如溢出），而不会由电池故障触发。每次扫描开始时都将清除该位。 **提示：**如果适用，可以使用 OTE 或 OTL 指令来显式设置 S:MINOR。
重要事项：	数学状态标志基于存储的值进行设置。对于通常不影响数学状态标志的指令，如果由于指令参数混合使用不同数据类型而进行类型转换，可能会影响数学状态标志。类型转换过程会设置数学状态标志。	

数组下标表达式

控制器	**说明**
CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、CompactGuardLogix 5380 和 GuardLogix 5580 控制器	表达式不会根据算术运算的结果设置状态标志。如果表达式溢出： • 将产生一个轻微故障（前提是将控制器配置为产生轻微故障）。 • 将产生一个严重故障（类型 4，代码 20），指示结果值超出范围。
CompactLogix 5370、ControlLogix 5570、CompactGuardLogix 5370 和 GuardLogix 5570 控制器	表达式根据算术运算的结果设置状态标志。如果数组下标以表达式表示，则表达式和指令都会产生轻微故障。

提示： 如果数组下标过大（超出范围），则会产生一个严重故障（类型 4，代码 20）。

立即数

输入十进制立即数（常数，例如 -2、3）后，控制器会将值存储为 32 位形式。如果输入基数形式而非十进制形式的值（例如二进制或十六进

制），但并未指定全部 32 位，则控制器会在未指定的位补零（填零）。

重要事项：	不足 32 位的二进制、八进制或十六进制立即数需填零。

输入值	存储值
-1	16#ffff ffff (-1)
16#ffff (-1)	16#0000 ffff (65535)
8#1234 (668)	16#0000 029c (668)
2#1010 (10)	16#0000 000a (10)

整型立即数

输入值	存储值
无后缀	DINT
"U"	UDINT
"L"	LINT
"UL"	ULINT

浮点型立即数

输入值	存储值
无后缀	REAL
"L"	LREAL

数据转换

如果在编程中混用数据类型，将会发生数据转换。

进行以下编程时：	在以下情况下会发生数据转换：
梯形图 结构化文本	一条指令或表达式中 混用不同数据类型的参数。
功能块	将数据类型不同的两个参数连接在一起

以下条件下，可以提高指令执行速度、节省内存空间：

- 使用相同数据类型。
- 立即数数据类型：

- 所有功能块指令仅支持一种数据类型的操作数。
- 如果混用不同数据类型或使用非最优数据类型的标签，控制器会根据以下规则进行数据转换：
  - 按照数据类型等级由低到高的顺序对操作数执行数据转换，SINT、USINT、INT、UINT、DINT、UDINT、LINT、ULINT、REAL 和 LREAL 的等级从 1（最低）到 10（最高）依次排列。

提示： 要避免因数据转换而占用时间和内存，应在一条指令中全部使用相同数据类型的操作数。

将 SINT 或 INT 型转换为 DINT 型，或者将 DINT 转换为 LINT 型

对于 SINT 或 INT 型输入源标签，可通过符号扩展升级为 DINT 型。将 SINT 或 INT 值转换为 DINT 值的指令会使用以下转换方法之一。

转换方法	具体处理方式
符号扩展	将最高有效位的值(值的符号位)放在现有位左侧的每一位 ,直至达到 32 或 64 位。
填零	在现有位左侧填零，直至达到 32 或 64 位。

逻辑指令使用填零方式。其他所有指令均使用符号扩展方式。

以下示例显示使用符号扩展和填零方式转换某个值后所得的结果。

该值	2#1111_1111_1111_1111	(-1)
通过符号扩展转换为该值	2#1111_1111_1111_1111_1111_1111_1111_1111	(-1)
通过填零转换为该值	2#0000_0000_0000_0000_1111_1111_1111_1111	(65535)

如果在指令中使用 SINT 或 INT 标签和立即数，而该指令通过符号扩展方式进行数据转换，则应使用以下一种方法处理立即数。

以十进制基数形式指定立即数。

如果输入基数形式而非十进制形式的值，应指定该立即数的全部 32 位。为此，应将最高有效位的值输入其左侧的每一位，直至达到 32 位。

为每个操作数创建一个标签，并在指令中使用相同的数据类型。要分配常数值，可采用两种方法：

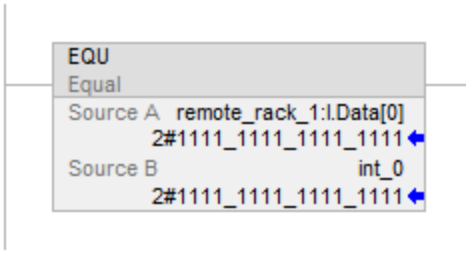
将值输入一个标签。

增加一个 MOV 指令，将值移动到一个标签中。

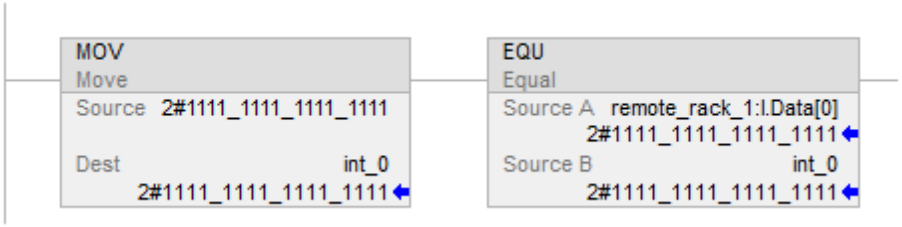
使用 MEQ 指令仅检查所需位。

以下示例介绍两种混用立即数与 INT 标签的方法。在两个示例中，均通过检查 1771 I/O 模块的位，来确认是否使用了所有位。1771 I/O 模块的输入数据字为 INT 标签，因此最简单的方法是使用 16 位常数值。

重要事项：	混用 INT 标签与立即数 由于 remote_rack_1:I.Data[0] 为 INT 标签，用于对照检查该标签的值也输入为 INT 标签。



重要事项：	混用 INT 标签与立即数 由于 remote_rack_1:I.Data[0] 为 INT 标签，用于对照检查该标签的值首先移至 int_0（同样为 INT 标签）中。随后，EQU 指令会对两个标签进行比较。



整型转换为 REAL 型

控制器以 IEEE 单精度浮点数格式存储 REAL 值。它使用一位存储值的符号，使用 23 位存储底数，使用八位存储指数（共 32 位）。如果同一条指令的输入混合使用了整型标签（SINT、INT 或 DINT）以及 REAL 型标签，控制器首先会将整型值转换为 REAL 值，然后再执行此指令。

- SINT 或 INT 值始终会转换为相等的 REAL 值。
- DINT 值可能不会转换为相等的 REAL 值：
- REAL 值最多使用 24 位来存储底数（23 个存储位加一个“隐藏”位）。
- DINT 值最多使用 32 位存储值（一位存储符号，31 位存储值）。

如果 DINT 值需要 24 个以上的有效位，则无法转换为相等的 REAL 值。控制器会将值四舍五入为最近的偶数值，并存储最高的 24 位。

DINT 型转换为 SINT 或 INT 型

将 DINT 值转换为 SINT 或 INT 值时，控制器会截断 DINT 的高位部分，并根据数据类型的要求存储相应的低位。如果该值过大，转换过程中会发生溢出。

	DINT 型转换为 INT 和 SINT 型	
DINT 值	转换为更小值	
16#0001_0081 (65,665)	INT：	16#0081 (129)
	SINT：	16#81 (-127)

REAL 型转换为 SINT、INT 或 DINT 型

将 REAL 值转换为整型值时，控制器会对小数部分四舍五入，并根据数据类型的要求存储相应的位。如果该值过大，转换过程中会发生溢出。

数值的四舍五入参见以下示例。

< 0.5 的小数向下舍入到最接近的整数。

> 0.5 的小数向上舍入到最接近的整数。

= 0.5 的小数向上或向下舍入到最接近的整数。

重要事项： REAL 值转换为 DINT 值	
REAL 值	**转换为 DINT 值**
-2.5 -3.5	-2 -4
-1.6	-2
-1.5	-2
-1.4	-1
1.4	1

1.5	2
1.6	2
2.5 3.5	2 4

基本数据类型

控制器支持的元素数据类型遵循 IEC 1131-3 定义的数据类型。基本数据类型包括：

数据类型	说明	范围
BOOL	1 位布尔型	0 = 清零 1 = 置位
SINT	1 字节整型	-128 至 127
INT	2 字节整型	-32,768 至 32,767
DINT	4 字节整型	-2,147,483,648 至 2,147,483,647
REAL	4 字节浮点型	$-3.402823E^{38}$ 至 $-1.1754944E^{-38}$ （负值） 和 0 和 $1.1754944E^{-38}$ 至 $3.402823E^{38}$ （正值）
LINT	8 字节整型	0 至 32,535,129,599,999,999
USINT	1 字节无符号整型	0 到 255
UINT	2 字节无符号整型	0 至 65535
UDINT	4 字节无符号整型	0 至 4,294,967,295
ULINT	8 字节无符号整型	0 至 18,446,744,073,709,551,615
REAL	4 字节浮点型	-3.4028235E38 至 -1.1754944E-38 （负值） 和 0.0 和 1.1754944E-38 至 3.4028235E38 （正值）

LREAL	8 字节浮点型	-1.7976931348623157E308 至 -2.2250738585072014E-308（负值） 和 0.0 和 2.2250738585072014E-308 至 1.7976931348623157E308（正值）

这些控制器支持以下基本数据类型：

控制器	数据类型
CompactLogix 5380、CompactLogix 5480、ControlLogix 5580、Compact GuardLogix 5380 和 GuardLogix 5580 控制器	SINT、INT、DINT、LINT、REAL USINT、UINT、UDINT、ULINT、LREAL
CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370 和 GuardLogix 5570 控制器	SINT、INT、DINT、LINT、REAL

控制器将所有立即数按 DINT 数据类型进行处理。

REAL 数据类型也存储±无穷大和± NAN，但软件显示因显示格式而有所不同。

数据类型转换

如果同一指令中的操作数混用多种不同的数据类型，有些指令会将数据自动转换为最适合该指令的数据类型。有些情况下，控制器将对数据进行转换以适应新的数据类型；有些情况下，控制器仅仅是尽可能适应数据。

转换	结果		
较大的整数转换为较小的整数	控制器会截断较大整数的高位并生成溢出故障。 例如：		
	十进制		**二进制**
	DINT	65,665	0000_0000_0000_0001_0000_0000_1000_0001
	INT	129	0000_0000_1000_0001

	SINT	-127	1000_0001
SINT 或 INT 转换为 REAL	数据精度无损失		
DINT 转换为 REAL	可能会损失数据精度。两种数据类型都是以 32 位存储数据，但 REAL 类型 32 位中的一些位用于存储指数值。如果精度出现损失，控制器从 DINT 的最低有效位获取数据。		
LREAL 到 LREAL	数据精度无损失。		
LREAL 到 REAL	可能会损失数据精度。		
LREAL/REAL 到无符号整型	可能会损失数据精度。如果源数据过大，无法存储在控制器的目标存储位置，则可能会溢出。		
有符号整型/无符号整型到 LREAL/REAL	如果整型值的有效位过多，无法存储在目标位置，则低有效位将被截断。		
有符号整型到无符号整型	如果源数据过大，无法存储在控制器的目标存储位置，则可能会溢出。		
无符号整型到有符号整型	如果源数据过大，无法存储在控制器的目标存储位置，则可能会溢出。		
REAL 转换为整型	控制器对小数部分进行四舍五入并截断非小数部分的高位。如果数据丢失，控制器会将溢出状态标志置位。 四舍五入至最接近的整数： 小于 0.5，舍去；等于 0.5，取为最接近的偶数；大于 0.5，则进位 例如：		
	REAL（源数据）	**DINT（结果）**	
	1.6	2	
	-1.6	-2	
	1.5	2	
	-1.5	-2	
	1.4	1	
	-1.4	-1	
	2.5	2	
	-2.5	-2	

BOOL 数据类型无法与其他类型相互转换。

重要事项：	数学状态标志根据存储的值进行置位。对于通常不影响数学状态关键字的指令，如果由于指令参数混合使用不同数据类型而进行类型转换，可能会影响数学状态关键字。类型转换过程会将数学状态关键字置位。

安全数据类型

若用户自定义类型或 Add-On 自定义类型由安全标签直接或间接引用，Logix Designer 应用程序会阻止修改会导致无效数据类型的用户自定义类型或 Add-On 自定义类型。（其中包括嵌套结构。）

安全标签可使用以下数据类型：

- 所有基本数据类型
- 用于安全应用指令的预定义类型。
- 用户定义数据类型或由前两个类型组成的数组。

安全标签中 UDT 成员名称的联机编辑

在 CompactLogix 5380、Compact GuardLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器上，可以对用户自定义数据类型的成员名称进行联机编辑。但如果用户自定义数据类型用于安全标签，且控制器处于安全保护状态，联机编辑将被禁用。

另请参见

数学状态标志 参考页数 589

LINT 数据类型

LINT 数据类型为 64 位整型。

LINT 数据类型可用于 Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 或 GuardLogix 5580 控制器 的多条指令，但 LINT 数据类型不适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570 控制器的大多数指令。

在 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570 控制器 中使用 LINT 数据类型时，需考虑以下条件。

提示： LINT 只能用于复制（COP，CPS）指令。这些指令适用于 CST/WallClock 时间属性、时间同步和 Add-On 自定义指令。您无法对此标签类型执行加、减、乘或除运算。

使用 LINT 数据类型时，若出现以下问题，请考虑以下说明。

执行的操作	说明
将两个双整型 DINT 值移动/复制至一个 LINT 标签	创建一个包含两个元素的双整型数组 ,总计 64 位(即 DINT[2]) ，随后将其复制到一个长整型标签中。
纠正日期/时间显示错误	当标签的值为负值时 ,不能以日期/时间形式显示。在标签编辑器中，将标签样式由日期/时间更改为二进制，以此检查值是否为负值。当最高有效位 (最左侧位) 为 1 时，值为负值，因此不能以日期或时间形式显示。

浮点值

此信息适用于 CompactLogix 5370、ControlLogix 5570、Compact GuardLogix 5370、GuardLogix 5570、Compact GuardLogix 5380、CompactLogix 5380、CompactLogix 5480、ControlLogix 5580 和 GuardLogix 5580 控制器。在适用的情况下会注明控制器的差异。

Logix 控制器根据 IEEE 754 浮点数算术标准处理浮点值。该标准规定了浮点数的存储和计算方法。IEEE 754 浮点数运算标准的设计宗旨是，能够在合理的存储空间内迅速处理超大量的数字。

REAL 标签存储单精度型归一化浮点数。

LEAL 标签存储双精度型归一化浮点数。

控制器支持以下基本数据类型：

控制器	数据类型
CompactLogix 5380、 CompactLogix 5480、 ControlLogix 5580、 Compact GuardLogix 5380 和 GuardLogix 5580 控制器	REAL、 LREAL

控制器	数据类型
CompactLogix 5370、 ControlLogix 5570、 Compact GuardLogix 5370 和 GuardLogix 5570 控制器	REAL

非归一化数值和 -0.0 按 0.0 处理

如果计算结果为 NAN 值，则符号位可以为正或负。这种情况下，软件会显示无符号的 1#.NAN。

并非所有十进制数值都能以这种标准格式准确表示，可能会出现精度损失。例如，10.1 减去 10，结果应为 0.1。但在 Logix 控制器中，结果可能是 0.10000038。在本例中，0.1 与 0.10000038 相差 0.000038%，近似为零。对大多数运算而言，这种细微的不精确性并不重要。从更深层次看，将一个浮点值发送给一个模拟量输出模块，与发送一个相差 0.000038% 的值相比，不会引起输出电压的差异。

浮点数算术运算法则

运算法则如下：

执行某些浮点算术运算时，可能会因四舍五入误差而造成精度损失。浮点数处理器的内在精度会影响结果值。

不应对金额或累加器函数使用浮点运算。应使用 INT 或 DINT 值，放大数值并跟踪小数位（或者对美元使用一个 INT 或 DINT 值，对美分使用另一个 INT 或 DINT 值）。

不应比较浮点数。但可以检查数值是否在某个范围内。LIM 指令专用于此用途。

累加器示例

REAL 数据类型的精度会影响累加运算，在将超小数与超大数相加时，将出现误差。

例如，在一段时间内，将某个值与 1 累加。到了某个点，由于中间总和值远大于 1，加法运算不再对结果产生影响，而且没有足够的位来存储整个结果。加法运算会存储尽可能多的高位，而丢弃其余的低位。

为解决这个问题，可先累加小数值，直到获得较大结果。然后将结果传输到其他位置，进行进一步的大数值运算。例如：

- x 为小增量变量。
- y 为大增量变量。

- Z 为可在任何位置使用的当前总计数。
- x = x+1;
- if x = 100,000;
- {
- y = y + 100,000;
- x = 0;
- }
- z = y + x;

另一个示例：

- x = x + some_tiny_number;
- if (x >= 100)
- {
- z = z + 100;
- x = x - 100; // there might be a tiny remainder
- }

数组索引编制

为动态更改逻辑所引用的数组元素，可使用标签或表达式作为指向元素的下标。这类似于 PLC-5 逻辑中的间接寻址。可以在表达式中使用以下运算符来指定数组下标：

提示：
- Logix Designer 仅支持以扩展数据类型标签作为下标，不支持具有扩展数据类型的下标表达式。
- 所有可用基本整型数据类型均可用作下标索引。仅能将 SINT、INT 和 DINT 标签与运算符结合使用，来创建下标表达式。

运算符	说明
+	加
-	减/求反
*	乘
/	除
与	与

运算符	说明
FRD	BCD 转换为整型
NOT	求补
或者	或者
TOD	整型转换为 BCD
SQR	平方根
XOR	异或

例如：

定义	示例	说明
my_list 定义为 DINT[10]	my_list[5]	本示例引用数组中的元素 5。该引用为静态引用，因为下标值保持常量。
my_list 定义为 DINT[10] 位置 定义为 DINT	MOV the value 5 into position my_list[position]	本示例引用数组中的元素 5。该引用为动态引用，因为逻辑可以通过更改 position 值来更改 下标。
my_list 定义为 DINT[10] 位置 定义为 DINT 偏移 定义为 DINT	MOV the value 2 into position MOV the value 5 into offset my_list[position+offset]	本例引用数组中的元素 7 (2+5)。该引用为动态引用，因为逻辑可以通过更改 position 或 offset 的值来更改下标。

提示： 输入数组下标时，应确保其在指定数组的边界范围内。如果下标超出相应的维度，将数组视为元素集合的指令会生成严重故障（类型 4，代码 20）。

位寻址

位寻址用于访问较大容器中的特定位。较大容器包括任何整型、结构或 BOOL 数组。例如：

定义	示例	说明
Variable0 定义为 LINT 类型 有 64 位	variable0.42	该示例用于引用 variable0 的位 42。

定义	示例	说明
variable1 定义为 DINT 类型 具有 32 位	variable1.2	该示例用于引用 variable1 的位 2。
variable2 定义为 INT 类型 具有 16 位	variable2.15	该示例用于引用 variable2 的位 15。
variable3 定义为 SINT 类型 具有 8 位	variable3.[4]	该示例用于引用 variable3 的位 4。
variable4 定义为 COUNTER 结构 具有 5 个状态位	variable4.DN	该示例用于引用 variable4 的 DN 位。
MyVariable 定义为 BOOL[100] MyIndex 定义为 SINT	MyVariable[(MyIndex AND NOT 7) / 8].[MyIndex AND 7]	该示例用于引用 BOOL 型数组中的一位。
MyArray 定义为 BOOL[20]	MyArray[3]	该示例用于引用 MyArray 的位 3。
variable5 定义为 ULINT 有 64 位	variable5.53	该示例用于引用 variable5 的位 53。

在任何支持 BOOL 类型标签的位置均可使用位寻址。

另请参见

数组索引编制 参考页数 602

功能块面板控件

Logix Designer 编程应用程序中包含某些功能块指令的面板控件。这些面板是 Active-X 控件，可以用在作为 Active-X 容器的应用程序中。面板通过 <RSLC> OPC 服务器或 FactoryTalk Linx 网关与控制器通信。

重要事项： Logix Designer 编程应用程序不是有效的 Active-X 容器。要使用面板，必须有 Active-X 容器。

以下指令具有面板：

- 过程离散输入（PDI）
- 过程离散输出（PDO）
- 过程模拟量输入（PAI）
- 过程模拟量输出（PAO）
- 报警（ALM）

- 增强型选择 (ESEL)
- 累加器 (TOT)
- 升温/持温 (RMPS)
- 离散 2 态设备 (D2SD)
- 离散 3 态设备 (D3SD)
- 增强型 PID (PIDE)

要配置面板，可通过由容器应用程序打开属性页面。

所有面板通常都有以下属性页面：

- 常规
- 显示 (Display)
- 字体 (Font)
- 地区 (Locale)

另请参见

面板控制属性 (Faceplate Control Properties) 对话框 - 常规 (General) 选项卡 参考页数 605

面板控制属性 (Faceplate Control Properties) 对话框 - 显示 (Display) 选项卡 参考页数 606

面板控制属性 (Faceplate Control Properties) 对话框 - 字体 (Font) 选项卡 参考页数 607

面板控制属性 (Faceplate Control Properties) 对话框 - 地区 (Locale) 选项卡 参考页数 608

面板控制属性 (Faceplate Control Properties) 对话框 - 常规 (General) 选项卡

使用此选项卡定义/修改控件的工作方式。

参数

通信 (Communication)

选择 RSLinx Classic OPC 服务器或 FactoryTalk Linx。如果选择 RSLinx Classic OPC 服务器，还必须指定：

- 是否远程启动
- 远程机器的访问路径

如果选择 FactoryTalk Linx FactoryTalk，则还必须指定 FactoryTalk 区域

标签（Tag）

输入与此控件连接的特定功能块指令的名称。

更新速率（Update Rate）

输入控件的更新速率（单位为秒）。可以单击箭头以 0.25 秒的增量递增该值。默认值为 1.00 秒。

另请参见

面板控制属性（Faceplate Control Properties）对话框 - 显示（Display）选项卡 参考页数 606

面板控制属性（Faceplate Control Properties）对话框 - 字体（Font）选项卡 参考页数 607

面板控制属性（Faceplate Control Properties）对话框 - 地区（Locale）选项卡 参考页数 608

面板控制属性 (Faceplate Control Properties) 对话框 - 显示 (Display) 选项卡

使用此选项卡来定义面板控件在屏幕上的显示方式。

参数

背景颜色（Background Color）

此按钮指示面板背景的颜色。单击此按钮可更改其颜色。默认颜色为浅灰色。

显示边框（Show Frame）

根据您是否希望在此控件周围显示三维边框，来决定是选中还是取消选中此框。使用此选项可以将控件与显示屏上可能出现的其他项分开。默认情况下选中此选项。

确定（OK）

单击此按钮可接受编辑内容并关闭“面板控制属性”(Faceplate Control Properties) 对话框。

取消（Cancel）

单击此按钮可取消编辑内容并关闭“面板控制属性”(Faceplate Control Properties）对话框。

应用（Apply）

单击此按钮可应用编辑内容并在“面板控制属性”(Faceplate Control Properties）对话框中继续编辑。

另请参见

面板控制属性（Faceplate Control Properties）对话框 - 常规（General）选项卡 参考页数 605

面板控制属性（Faceplate Control Properties）对话框 - 字体（Font）选项卡 参考页数 607

面板控制属性（Faceplate Control Properties）对话框 - 地区（Locale）选项卡 参考页数 608

面板控制属性 (Faceplate Control Properties) 对话框 - 字体（Font）选项卡

使用此选项卡定义面板上显示的字体。在此处配置要在面板的主要部分中使用的 ControlFont 以及要在面板的标尺和其他次要部分中使用的 MinorFont 字体。

参数

属性名称（Property Name）

从下拉菜单中选择要配置的字体。选择 ControlFont 或 MinorFont；默认为 ControlFont。

字体（Font）

从可用字体列表中选择要用于控件的字体。默认字体是 Arial。

样式（Style）

从下拉菜单中选择要用于控件的样式。默认样式是“常规”(Regular)。

字号（Size）

输入要用于此字体的字号。ControlFont 的默认字号为 10.5；MinorFont 的默认字号为 8.25。

删除线（Strikeout）

如果要添加删除线效果（在字体上绘制一条线），则选中此框。默认情况下取消选中此选项。

下划线（Underline）

如果要添加下划线效果（在字体下绘制一条线），则选中此框。默认情况下取消选中此选项。

确定（OK）

单击此按钮可接受编辑内容并关闭“面板控制属性”(Faceplate Control Properties）对话框。

取消（Cancel）

单击此按钮可取消编辑内容并关闭“面板控制属性”(Faceplate Control Properties）对话框。

应用（Apply）

单击此按钮可应用编辑内容并在“面板控制属性”(Faceplate Control Properties）对话框中继续编辑。

另请参见

面板控制属性 (Faceplate Control Properties) 对话框 - 常规 (General) 选项卡 参考页数 605

面板控制属性 (Faceplate Control Properties) 对话框 - 显示 (Display) 选项卡 参考页数 606

面板控制属性 (Faceplate Control Properties) 对话框 - 地区 (Locale) 选项卡 参考页数 608

面板控制属性 (Faceplate Control Properties) 对话框 - 地区 (Locale) 选项卡

使用此选项卡定义面板的语言要求。

参数

地区（Locale）

从下拉菜单中选择要使用的语言。从以下选项中进行选择：

- 英语（English）
- 葡萄牙语（Portuguese）
- 法语（French）
- 意大利语（Italian）
- 德语（German）
- 西班牙语（Spanish）

确定（OK）

单击此按钮可接受编辑内容并关闭“面板控制属性”(Faceplate Control Properties）对话框。

取消（Cancel）

单击此按钮可取消编辑内容并关闭“面板控制属性”(Faceplate Control Properties）对话框。

应用（Apply）

单击此按钮可应用编辑内容并在“面板控制属性”(Faceplate Control Properties）对话框中继续编辑。

另请参见

面板控制属性（Faceplate Control Properties）对话框 - 常规（General）选项卡 参考页数 605

面板控制属性（Faceplate Control Properties）对话框 - 显示（Display）选项卡 参考页数 606

面板控制属性（Faceplate Control Properties）对话框 - 字体（Font）选项卡 参考页数 607

ASCII 字符代码

字符	十进制	十六进制	字符	十进制	十六进制	字符	十进制	十六进制	字符	十进制	十六进制
[ctrl-@] NUL	0	$0	空格	32	$20	@	64	$40	‘	96	$60
[ctrl-A] SOH	1	$1	!	33	$21	A	65	$41	a	97	$61
[ctrl-B] STX	2	$2	“	34	$22	B	66	$42	b	98	$62
[ctrl-C] ETX	3	$3	#	35	$23	C	67	$43	c	99	$63
[ctrl-D] EOT	4	$4	$	36	$24	D	68	$44	d	100	$64

[ctrl-E] ENQ	5	$5
[ctrl-F] ACK	6	$6
[ctrl-G] BEL	7	$7
[ctrl-H] BS	8	$8
[ctrl-I] HT	9	$9
[ctrl-J] LF	10	$l ($0A)
[ctrl-K] VT	11	$0B
[ctrl-L] FF	12	$0C
[ctrl-M] CR	13	$r ($0D)
[ctrl-N] SO	14	$0E
[ctrl-O] SI	15	$0F
[ctrl-P] DLE	16	$10
[ctrl-Q] DC1	17	$11
[ctrl-R] DC2	18	$12
[ctrl-S] DC3	19	$13
[ctrl-T] DC4	20	$14
[ctrl-U] NAK	21	$15
[ctrl-V] SYN	22	$16
[ctrl-W] ETB	23	$17
[ctrl-X] CAN	24	$18
[ctrl-Y] EM	25	$19
[ctrl-Z] SUB	26	$1A
ctrl-[ESC	27	$1B
[ctrl-\] FS	28	$1C
ctrl-] GS	29	$1D
[ctrl-^] RS	30	$1E
[ctrl-_] US	31	$1F

%	37	$25
&	38	$26
‘	39	$27
(	40	$28
)	41	$29
*	42	$2A
+	43	$2B
,	44	$2C
-	45	$2D
.	46	$2E
/	47	$2F
0	48	$30
1	49	$31
2	50	$32
3	51	$33
4	52	$34
5	53	$35
6	54	$36
7	55	$37
8	56	$38
9	57	$39
:	58	$3A
;	59	$3B
<	60	$3C
=	61	$3D
>	62	$3E
?	63	$3F

E	69	$45
F	70	$46
G	71	$47
H	72	$48
I	73	$49
J	74	$4A
K	75	$4B
L	76	$4C
M	77	$4D
N	78	$4E
O	79	$4F
P	80	$50
Q	81	$51
R	82	$52
S	83	$53
T	84	$54
U	85	$55
V	86	$56
W	87	$57
X	88	$58
Y	89	$59
Z	90	$5A
[	91	$5B
\	92	$5C
]	93	$5D
^	94	$5E
_	95	$5F

e	101	$65
f	102	$66
g	103	$67
h	104	$68
i	105	$69
j	106	$6A
k	107	$6B
l	108	$6C
m	109	$6D
n	110	$6E
o	111	$6F
p	112	$70
q	113	$71
r	114	$72
s	115	$73
t	116	$74
u	117	$75
v	118	$76
w	119	$77
x	120	$78
y	121	$79
z	122	$7A
{	123	$7B
|	124	$7C
}	125	$7D
~	126	$7E
DEL	127	$7F

索引

功

升

增

微

标

滤

离

累

设

选

面

Rockwell Automation 支持

网站上提供了罗克韦尔自动化在技术信息，以帮助您使用其产品。在 http://www.rockwellautomation.com/support 您可以找到技术和应用手册，实例代码及软件服务包的链接。您还可以在 https://rockwellautomation.custhelp.com 访问"支持中心"，进行软件的更新，支持聊天和论坛，技术信息，FAQs，注册产品更新通知。

此外，这里为安装、配置及排除故障提供了多样化的支持程序。更多信息，请联系您当地的经销商或 Rockwell Automation 代表，也可以访问 http://www.rockwellautomation.com/support/。

安装帮助

如果您在安装后的 24 小时内遇到问题，请查阅本手册中的相关信息。您也可以联系客户支持获取有关使产品恢复正常运行的初步帮助。

美国或加拿大	1.440.646.3434
美国或加拿大以外地区	使用 http://www.rockwellautomation.com/support/americas/phone_en.html 上的"全球地点"或联系您当地的 "Rockwell Automation" 代理。

新产品退货

所有产品出厂前，Rockwell Automation 都会进行相关测试，以确保产品能够全面运转。但是，如果您的产品不能正常工作，需要退货，请遵循下列步骤。

美国	请联系您的经销商。您必须向经销商提供客户支持案例号码（可拨打以上电话号码获取）以完成退货流程。
美国以外地区	请联系您当地的 Rockwell Automation 代表以了解退货流程。

文档反馈

您的意见将有助于我们更好地满足您对文档的需求。如果您有更好的建议，请完成此反馈表，出版号RA-DU002。

Rockwell Otomasyon Ticaret A.Ş., Kar Plaza İş Merkezi E Blok Kat:6 34752 İçerenköy, İstanbul, Tel: +90 (216) 5698400

中文网址 **www.rockwellautomation.com.cn**
新浪微博 **www.weibo.com/rockwellchina**

动力、控制与信息解决方案总部
美洲地区：罗克韦尔自动化，南二大街1201号，密尔沃基市，WI 53204-2496 美国，电话：(1) 414.382.2000，传真：(1) 414.382.4444
欧洲/中东/非洲：罗克韦尔自动化，NV, Pegasus Park, De Kleetlaan 12a, 1831布鲁塞尔，比利时，电话：(32) 2 663 0600, 传真：(32) 2 663 0640
亚太地区：罗克韦尔自动化，香港数码港道100号数码港3座F区14楼1401-1403 电话：(852)2887 4788 传真：(852)2508 1486
中国总部：上海市徐汇区虹梅路1801号宏业大厦 邮编：200233 电话：(86 21)6128 8888 传真：(86 21)6128 8899
客户服务电话：400 620 6620 (中国地区)　　+852 2887 4666 (香港地区)

Rockwell Automation 出版号 1756-RM006K-ZH-P - 2018 年 11 月