OMRON

机器自动化控制器

NJ/NX系列

指令基准手册
基本篇

NX701–17□□
NX701–16□□
NX1P2–11□□□□
NX1P2–10□□□□
NX1P2–90□□□□
NJ501–□5□□
NJ501–□4□□
NJ501–□3□□
NJ301–12□□
NJ301–11□□
NJ101–10□□
NJ101–90□□

SYSMAC
always in control

SBCA–CN5–360W

前言

非常感谢您购买NJ/NX系列CPU单元。
本手册记载了使用 NJ/NX 系列 CPU 单元所必需的信息。使用前请仔细阅读本手册，充分理解其功能和性能，并用于系统的构建。
此外，阅读后请将本手册妥善保管于易取处。

阅读对象

本手册提供给下列阅读对象：
具有电工专业知识的人员(合格的电气工程师或具有同等知识的人员)；
- 引进FA设备的人员；
- 设计FA系统的人员；
- 安装或连接FA设备的人员；
- FA现场管理人员。

此外，编程语言的阅读对象为理解国际标准规格 IEC61131–3 或国内标准规格 JIS B3503 的规定内容的人员。

对象产品

本手册以下列产品为对象。
- NX系列CPU单元
  - NX701–17□□
  - NX701–16□□
  - NX1P2–11□□□□
  - NX1P2–11□□□□1
  - NX1P2–10□□□□
  - NX1P2–10□□□□1
  - NX1P2–90□□□□
  - NX1P2–90□□□□1

- NJ系列CPU单元
  - NJ501–□5□□
  - NJ501–□4□□
  - NJ501–□3□□
  - NJ301–12□□
  - NJ301–11□□
  - NJ101–10□□
  - NJ101–90□□

各产品的部分规格或限制事项可能记载在其他手册中。请确认 🕮 “分册构成 (P.2)” 及 🕮 “相关手册 (P.28)”。

分册构成

本产品手册按下表分册。请根据目的阅读，充分应用本产品。

本产品的操作主要使用自动化软件Sysmac Studio。关于Sysmac Studio，请参阅 “Sysmac Studio Version 1 操作手册(SBCA-362)”。

使用目的	手册													
	基本信息													
	NX 系列 CPU 单元 用户手册 硬件篇	NX 系列 NX1P2 CPU 单元 用户手册 硬件篇	NJ 系列 CPU 单元 用户手册 硬件篇	NJ/NX 系列 CPU 单元 用户手册 软件篇	NX 系列 NX1P2 CPU 单元 用户手册 内置 I/O、扩展板功能篇	NJ/NX 系列 指令基准手册 基本篇	NJ/NX 系列 CPU 单元 用户手册 运动控制篇	NJ/NX 系列 指令基准手册 运动篇	NJ/NX 系列 CPU 单元 内置 EtherCAT 端口 用户手册	NJ/NX 系列 CPU 单元 内置 EtherNet/IP 端口用户手册	NJ 系列 数据库连接 CPU 单元 用户手册	NJ 系列配备 SECS/GEM 的 CPU 单元 用户手册	NJ 系列 NJ Robotics CPU 单元 用户手册	NJ/NX 系列 故障诊断手册
了解NX701的概要	●													
了解NX1P2的概要		●												
了解NJ系列的概要			●											
进行安装、设置、硬件设定	●	●	●											
进行运动控制时							●							
使用EtherCAT时									●					
使用EtherNet/IP时										●				
进行软件设定				●										
进行运动控制时							●							
使用EtherCAT时									●					
使用EtherNet/IP时										●				
使用数据库连接服务时											●			
使用GEM服务时												●		
进行机器人控制时													●	
使用NX1P2的功能时					●									
编写用户程序				●		●								
进行运动控制时							●	●						
使用EtherCAT时									●					
使用EtherNet/IP时										●				
使用数据库连接服务时											●			
使用GEM服务时												●		
进行机器人控制时													●	
进行异常处理时														●
使用NX1P2的功能时					●									

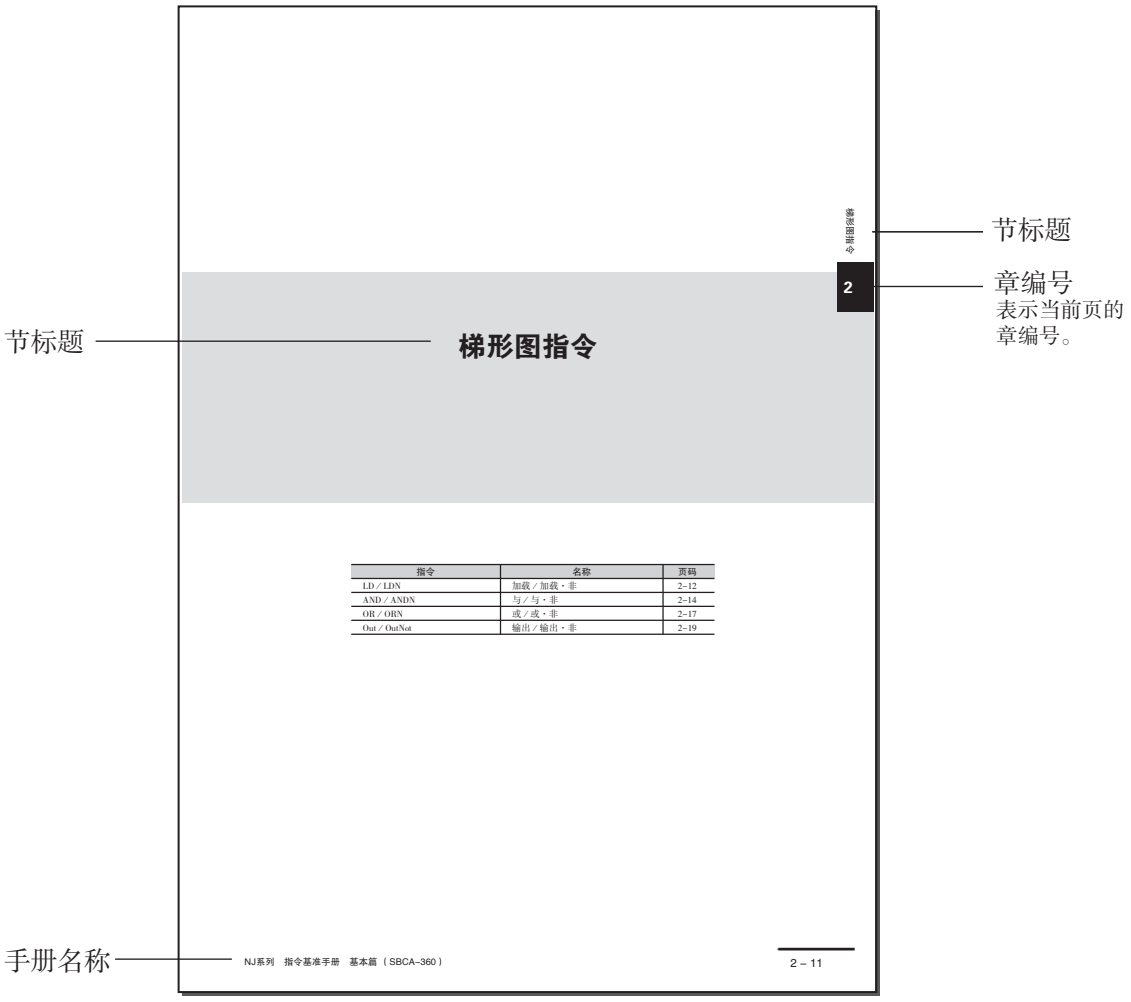
使用目的	手册													
	基本信息													
	NX 系列 用户手册 硬件篇	NX 系列 NX1P2 CPU 单元 用户手册 硬件篇	NJ 系列 CPU 单元 用户手册 硬件篇	NJ/NX 系列 CPU 单元 用户手册 软件篇	NX 系列 NX1P2 CPU 单元 用户手册 内置 I/O、扩展板功能篇	NJ/NX 系列 指令基准手册 基本篇	NJ/NX 系列 CPU 单元 用户手册 运动控制篇	NJ/NX 系列 指令基准手册 运动篇	NJ/NX 系列 CPU 单元 内置 EtherCAT 端口 用户手册	NJ/NX 系列 CPU 单元 内置 EtherNet/IP 端口用户手册	NJ 系列数据库连接 CPU 单元 用户手册	NJ 系列配备 SECS/GEM 的 CPU 单元 用户手册	NJ 系列 NJ Robotics CPU 单元 用户手册	NJ/NX 系列 故障诊断手册
进行动作确认和调试				●										
进行运动控制时							●							
使用EtherCAT时									●					
使用EtherNet/IP时										●				
使用数据库连接服务时											●			
使用GEM服务时												●		
进行机器人控制时													●	
使用NX1P2的功能时					●									
了解异常管理功能和故障发生时的处理方法 *1	△	△	△	△	△		△		△	△	△	△	△	●
了解维护作业	●	●	●											
进行运动控制时							●							
使用EtherCAT时									●					
使用EtherNet/IP时										●				

*1 关于异常管理的思路和异常项目的概要，请参阅 “NJ/NX 系列 故障诊断手册 (SBCA-361)”。关于异常详情，请根据异常内容，参阅标有△标志的手册。

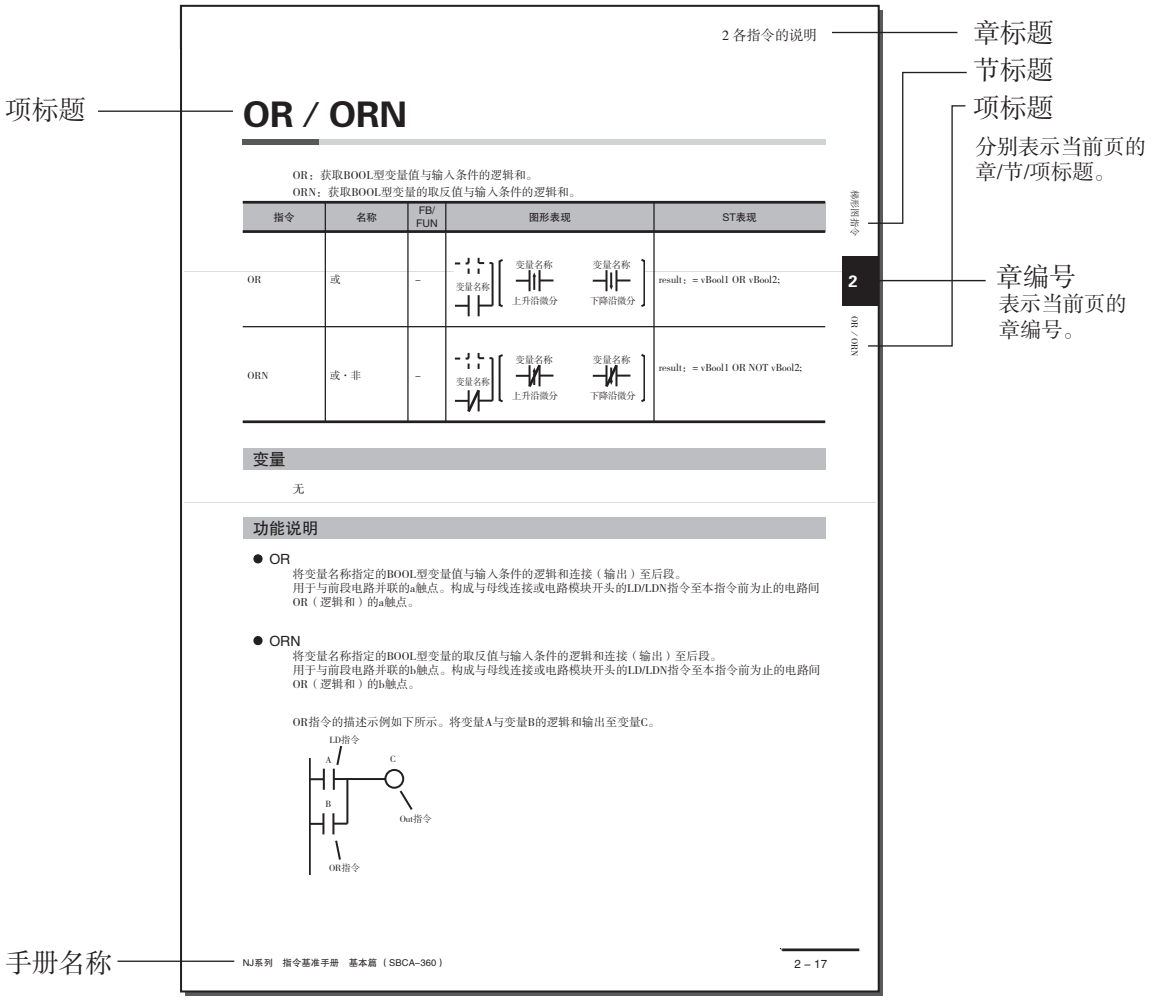
手册的阅读方法

页面构成

本手册的各页面构成如下所示。



本页为用于说明的范例页。与实际内容有所差异。



本页为用于说明的范例页。与实际内容有所差异。

图标

本资料中使用的图标，含义如下。

使用注意事项

表示为了预防产品无法动作、误动作，或者对产品性能、功能产生不良影响而应当实施或避免的事项。

参考

希望根据需要阅读的项目。
对应当了解的信息及使用时可作为参考的相关内容进行说明。

版本相关信息

对CPU单元、Sysmac Studio不同版本的不同性能和功能进行说明。

表示详细信息、相关信息的所在页。

目录构成

目录

第 1 章 指令一览

第 2 章 各指令的说明

附录

索引

承诺事项

关于 “本公司产品”，若无特殊协议，无论客户从何处购买，均适用本承诺事项中的条件。

● 定义

本承诺事项中用语的定义如下所示。

- “本公司产品”：“本公司”的FA系统设备、通用控制设备、传感设备、电子和机械零件。
- “产品样本等”：与 “本公司产品”相关的欧姆龙工控设备、电子和机械零件综合样本、其他产品样本、规格书、使用说明书、手册等，还包括通过电磁介质提供的资料。
- “使用条件等”：“产品样本等”中的 “本公司产品”的使用条件、额定值、性能、运行环境、使用方法、使用注意事项、禁止事项等。
- “用户用途”：用户使用 “本公司产品”的方法，包括直接使用或将 “本公司产品”装入用户制造的零件、印刷电路板、机械、设备或系统等。
- “适用性等”：“用户用途”中 “本公司产品”的(a)适用性、(b)动作、(c)不侵犯第三方知识产权、(d)遵守法律以及(e)遵守各种标准。

● 记载内容的注意事项

关于 “产品样本等”中的内容，请注意以下几点。

- 额定值和性能值是在各条件下进行单独试验后获取的值，并不保证在复合条件下可获取各额定值和性能值。
- 参考数据仅供参考，并不保证在该范围内始终正常运行。
- 使用实例仅供参考，“本公司”不保证 “适用性等”。
- “本公司”可能会因产品改良、本公司的原因而中止 “本公司产品”的生产或变更 “本公司产品”的规格。

● 使用注意事项

使用时，请注意以下几点。

- 使用时请符合额定值、性能以及 “使用条件等”。
- 请用户自行确认 “适用性等”，判断是否可使用 “本公司产品”。“本公司”对 “适用性等”不作任何保证。
- 用户将 “本公司产品”用于整个系统时，请务必事先自行确认配电、设置是否恰当。
- 使用 “本公司产品”时，请注意以下各事项。(i)使用 “本公司产品”时，应在额定值和性能方面留有余量，采用冗余设计等安全设计，(ii)采用安全设计，即使 “本公司产品”发生故障，也可将 “用户用途”造成的危险降至最低程度，(iii) 对整个系统采取安全措施，以便向使用者告知危险，(iv) 定期维护 “本公司产品”及 “用户用途”。
- “本公司产品”是本公司设计并制造面向一般工业产品的通用产品。但是，不可用于以下用途。如果用户将 “本公司产品”用于以下用途，则 “本公司”不对 “本公司产品”作任何保证。但经 “本公司”许可后用于以下用途或与 “本公司”签订特殊协议的情况除外。
  - (a) 需高安全性的用途(例：原子能控制设备、燃烧设备、航空航天设备、铁路设备、起重设备、娱乐设备、医疗设备、安全装置以及其他危及生命、健康的用途)
  - (b) 需高可靠性的用途(例：煤气、自来水、电力等供应系统、24小时连续运行的系统、支付系统等涉及权利、财产的用途等)
  - (c) 用于严格条件或环境下(例：需设置在室外的设备、会受化学污染的设备、会受电磁波干扰的设备、会受振动和冲击影响的设备等)
  - (d) 在 “产品样本等”中未记载的条件或环境下使用

・ 上述(a) ~ (d)以及 “本产品样本等中记载的产品”不可用于汽车(含两轮车。下同)。请勿装入汽车进行使用。关于可装入汽车的产品，请咨询本公司销售负责人。

● 保修条件

“本公司产品”的保修条件如下所述。

・ 保修期为购买本产品后的1年内。
(“产品样本等”中另有记载的情况除外。)

・ 保修内容　对发生故障的 “本公司产品”，经 “本公司”判断后提供以下任一服务。

(a) 发生故障的 “本公司产品”可在本公司维修服务网点免费维修
(不提供电子和机械零件的维修服务。)

(b) 免费提供与发生故障的 “本公司产品”数量相同的替代品

・ 非保修范围　如果因以下任一原因造成故障，则不在保修范围内。

(a) 用于 “本公司产品”原本用途以外的用途

(b) 未按 “使用条件等”进行使用

(c) 违反本承诺事项中的 “使用注意事项”进行使用

(d) 改造或维修未经 “本公司”

(e) 使用的软件程序非由 “本公司”人员编制

(f) 因以出厂时的科学技术水平无法预见的原因

(g) 除上述以外，因 “本公司”或 “本公司产品”以外的原因(包括自然灾害等不可抗力)

● 责任免除

本承诺事项中的保修即与 “本公司产品”相关的保修的所有内容。
对因 “本公司产品”造成的损害，“本公司”及 “本公司产品”的销售店概不负责。

● 出口管理

出口 “本公司产品”或技术资料或向非居民的人员提供时，应遵守日本及各国安全保障贸易管理相关的法律法规。如果用户违反上述法律法规，则可能无法向其提供 “本公司产品”或技术资料。

安全注意事项

安全注意事项请参阅下列手册。

- 🕮“NX系列CPU单元 用户手册 硬件篇(SBCA–358)”
- 🕮“NX系列NX1P2 CPU单元 用户手册 硬件篇(SBCA–448)”
- 🕮“NJ系列CPU单元 用户手册 硬件篇(SBCA–359)”

安全要点

安全要点请参阅下列手册。

- 🕮 “NX系列CPU单元 用户手册 硬件篇(SBCA-358)”
- 🕮 “NX系列NX1P2 CPU单元 用户手册 硬件篇(SBCA-448)”
- 🕮 “NJ系列CPU单元 用户手册 硬件篇(SBCA-359)”

使用注意事项

使用注意事项请参阅下列手册。

- 🕮“NX系列CPU单元 用户手册 硬件篇(SBCA–358)”
- 🕮“NX系列NX1P2 CPU单元 用户手册 硬件篇(SBCA–448)”
- 🕮“NJ系列CPU单元 用户手册 硬件篇(SBCA–359)”

法规与标准

日本国外的使用

对本产品，根据外汇和外国贸易管理法的规定，出口(或提供给非本土企业)需获得出口许可、批准的货物(或技术)时，需依照上述法规获得出口许可、批准(或劳务贸易许可)。

符合EC指令

符合指令

- EMC指令
- 低电压指令

适用途径

● EMC指令

欧姆龙的产品为装入各种机械、制造装置使用的电气设备，为使装入的机械、装置更容易符合EMC标准，产品自身需符合相关EMC标准(*)。
但客户的机械、装置多种多样，且EMC的性能因装入符合EC指令产品的机械、控制柜的构成、布线状态、配置状态等而异，因此无法确认客户使用状态下的适用性。因此，请客户自行确认机械、装置整体最终的EMC适用性。

* EMC (Electro-Magnetic Compatibility：电磁环境兼容性)相关标准中，
与EMS (Electro-Magnetic Susceptibility：电磁敏感性)相关的为EN61131-2；
与EMI (Electro-Magnetic Interference：电磁干扰)相关的为EN61131-2。
此外，EN61000-6-4 Radiated emission依照10m法。

● 低电压指令

对于以电源电压50V AC ~ 1000V AC以及75V DC ~ 1500V DC工作的设备，要求必须确保必要的安全性。适用标准为EN61010-2-201。

● 符合EU指令

NJ/NX系列符合EU指令。要使客户的机械、装置符合EU指令，需注意以下事项。

- NJ/NX系列请务必安装在控制柜内。
- 与DC电源单元及I/O单元连接的DC电源请使用强化绝缘品或双重绝缘品。
- NJ/NX系列的EU指令符合产品符合EMI相关的通用排放标准，但关于Radiated emission(10m法)，会因使用的控制柜构成、与连接的其它设备间的关系、接线等而异。
  因此，使用符合EU指令的NJ/NX系列时，也需客户自行根据机械、装置整体确认是否符合EU指令。

符合船级标准

本产品符合各种船级标准。为符合各船级标准，需设定使用条件，部分安装场所无法使用，因此使用时请务必向本公司营业部咨询。

各船级标准的使用条件(NK、LR)

- 本产品请务必安装在控制柜中。
- 控制柜的开闭口等处的间隙请使用衬垫等完全遮蔽。
- 电源线上请连接下列噪声滤波器。

噪声滤波器

厂家	型号
COSEL公司	TAH-06-683

软件许可证与著作权

本产品已安装第三方软件。该软件的相关许可证和著作权请浏览http://www.fa.omron.co.jp/nj_info_j/。

版本

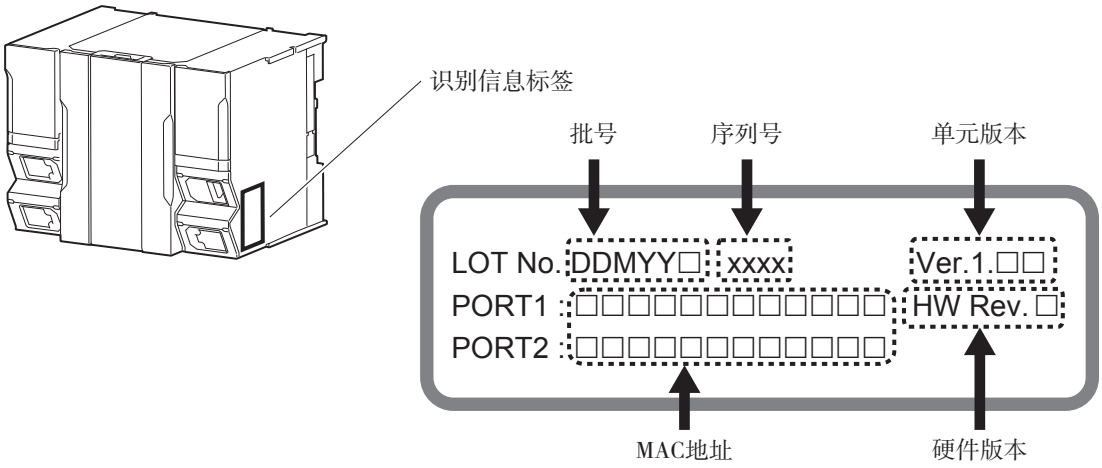
NJ/NX 系列各单元及各 EtherCAT 从站的硬件和软件是通过硬件版本、单元版本等不同编号来进行版本管理。硬件或软件每次变更规格，都将更新硬件版本或单元版本。因此，即使是同一型号的单元和EtherCAT从站，如果硬件版本或单元版本不同，配备的功能和性能就会存在差异。

版本确认方法

版本可通过识别信息标签或Sysmac Studio进行确认。

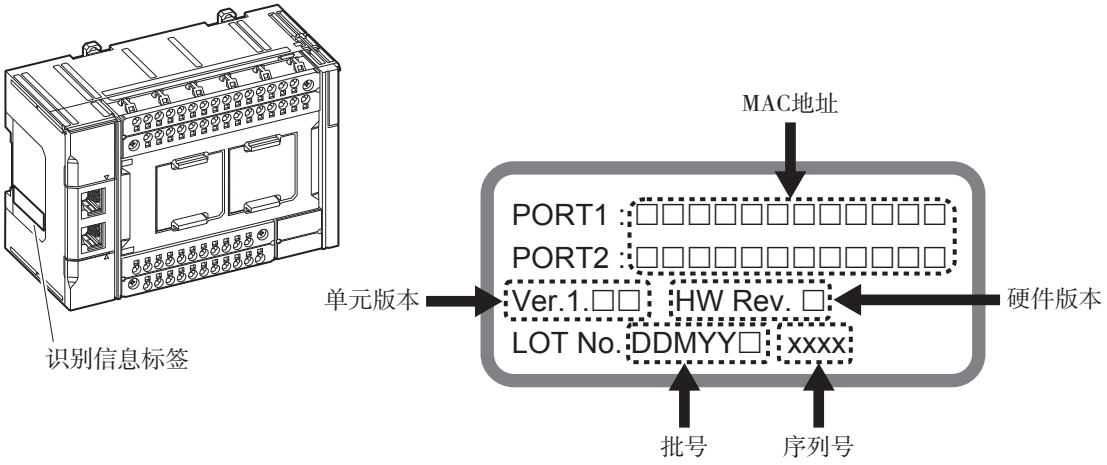
通过识别信息标签确认

版本可通过产品侧面的识别信息标签进行确认。
NX系列CPU单元NX701 – □□□□的识别信息标签如下图所示。



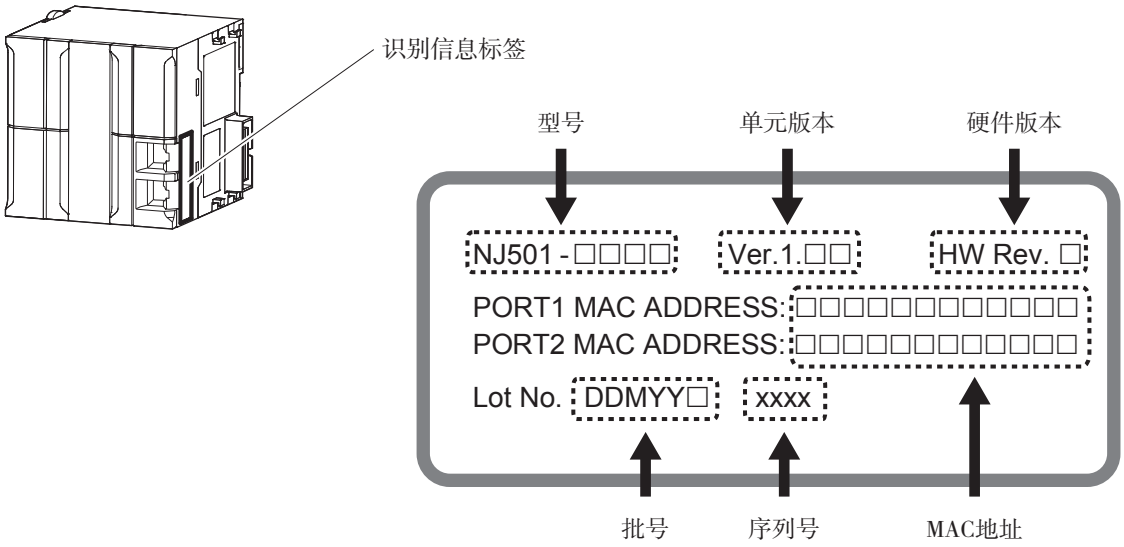
(注)　硬件版本为 “无”的单元不显示硬件版本。

NX系列CPU单元NX1P2 – □□□□□□□的识别信息标签如下图所示。



(注)　硬件版本为 “无”的单元不显示硬件版本。

NJ系列CPU单元NJ501 – □□□□的识别信息标签如下图所示。



(注)　硬件版本为 “无”的单元不显示硬件版本。

基于Sysmac Studio的确认方法

可通过Sysmac Studio确认版本。单元和EtherCAT从站的确认方法不同。

● NX系列 CPU单元的版本确认方法

单元版本可通过在线状态下的[生产信息]确认。可确认版本的单元为CPU单元。NX1P2 CPU单元也可确认CPU机架上的NX单元、扩展板的版本。

1 **在多视图浏览器中右击[构成・设定] – [CPU・扩展机架] – [CPU机架]，选择[显示生产信息]。**
显示[生产信息]对话框。

● NJ系列CPU单元的版本确认方法

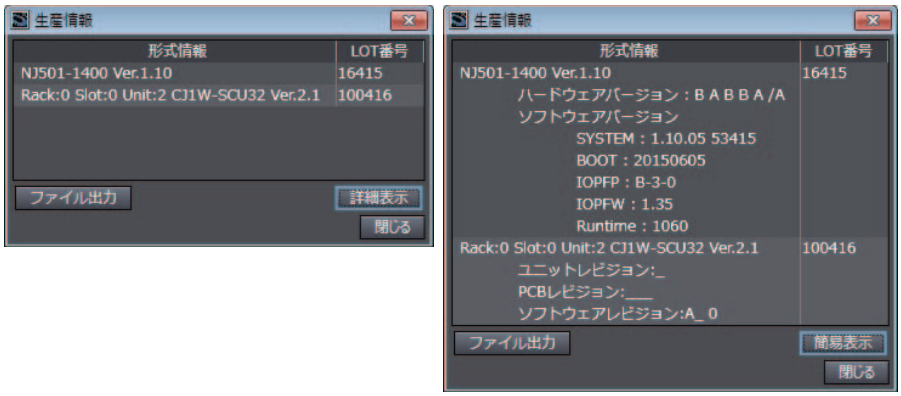
单元版本可通过在线状态下的[生产信息]确认。但可确认版本的单元只有CPU单元、CJ系列的高功能I/O单元及CJ系列的CPU高功能单元。CJ单元的基本I/O单元的版本无法通过Sysmac Studio进行确认。
版本确认方法如下所示。

1 **在多视图浏览器中双击[构成・设定] – [CPU・扩展机架]。或者右击[构成・设定] – [CPU・扩展机架]，选择[编辑]。**
显示单元编辑器。

2 **右击单元编辑器的空白处，选择[显示生产信息]。**
显示[生产信息]对话框。

● 生产信息显示内容的切换

1 **选择[生产信息]对话框右下方的[简要显示]或[详细显示]。**
[生产信息]的简要显示和详细显示将会切换。



简要显示　　详细显示

简要显示和详细显示的显示内容不同。详细显示会显示单元版本、硬件版本及软件版本。简要显示只显示单元版本。

(注)　硬件版本在硬件版本的右端以 “/” 隔开显示。硬件版本为 “无” 的单元不显示硬件版本。

● EtherCAT从站版本确认方法

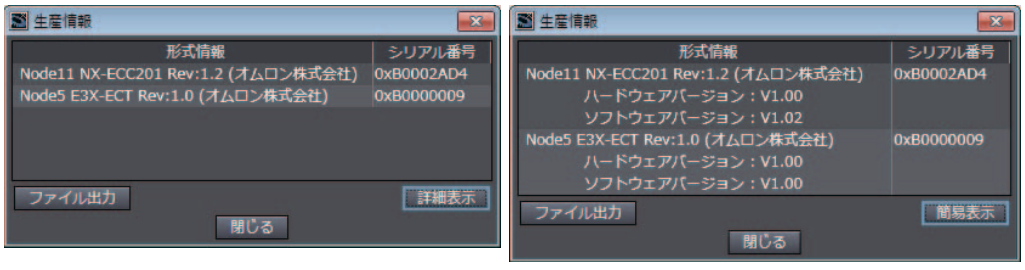
EtherCAT从站版本可通过在线状态下的[生产信息]确认。确认方法如下所示。

1 **双击多视图浏览器内的[构成・设定] – [EtherCAT]。或者右击[构成・设定] – [EtherCAT]，选择[编辑]。**
显示EtherCAT构成编辑画面。

2 **在EtherCAT构成的编辑画面中右击主机，选择[显示生产信息]。**
显示生产信息对话框。
显示的单元版本附带 “Rev” 字样。

● 生产信息显示内容的切换

1 **选择[生产信息]对话框右下方的[简要显示]或[详细显示]。**
[生产信息]的简要显示和详细显示将会切换。



简要显示　　详细显示

单元版本和Sysmac Studio版本

对应的功能因NJ/NX系列CPU单元的版本而异。使用版本升级后的新增功能时，需使用对应版本的Sysmac Studio。

CPU单元的单元版本的种类与Sysmac Studio版本之间的关系，以及单元版本支持的功能一览请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”。

相关手册

与本手册相关的手册如下表所述。请同时参阅。

手册名称	Man.No.	型号	用途	内容
NX系列 CPU单元 用户手册 硬件篇	SBCA–418	NX701–□□□□	希望了解NX701 CPU单元的概要/设计/安装/保养等基本规格时。 与硬件相关的信息为主。	对NX701的系统整体概要和CPU单元进行以下内容的说明。 • 特长和系统构成 • 概要 • 各部分的名称和功能 • 一般规格 • 安装与接线 • 维护检查
NX系列 NX1P2 CPU单元 用户手册 硬件篇	SBCA–448	NX1P2–□□□□	希望了解NX1P2 CPU单元的概要/设计/安装/保养等基本规格时。 与硬件相关的信息为主。	对NX1P2的系统整体概要和CPU单元进行以下内容的说明。 • 特长和系统构成 • 概要 • 各部分的名称和功能 • 一般规格 • 安装与接线 • 维护检查
NJ系列 CPU单元 用户手册 硬件篇	SBCA–358	NJ501–□□□□ NJ301–□□□□ NJ101–□□□□	希望了解NJ系列CPU单元的概要/设计/安装/保养等基本规格时。 与硬件相关的信息为主。	对NJ系列的系统整体概要和CPU单元进行以下内容的说明。 • 特长和系统构成 • 概要 • 各部分的名称和功能 • 一般规格 • 安装与接线 • 维护检查
NJ/NX系列 CPU单元 用户手册 软件篇	SBCA–359	NX701–□□□□ NX1P2–□□□□ NJ501–□□□□ NJ301–□□□□ NJ101–□□□□	希望了解NJ/NX系列CPU单元的编程/系统调试时。 与软件相关的信息为主。	对NJ/NX系列的CPU单元进行以下内容的说明。 • CPU 单元的动作 • CPU 单元的功能 • 初始设定 • 符合 IEC 61131–3 标准的语言规格和编程
NX系列 NX1P2 CPU单元 用户手册 内置I/O、扩展板功能篇	SBCA–449	NX1P2–□□□□	希望了解NX系列NX1P2 CPU单元独有功能的详情和NJ/NX系列功能的概要时。	对NX1P2 CPU单元功能中的以下内容进行说明。 • 内置 I/O • 串行通信扩展板 • 模拟输入输出扩展板 并对NJ/NX系列CPU单元以下功能的概要进行说明。 • 运动控制功能 • EtherNet/IP 通信功能 • EtherCAT 通信功能
NJ/NX系列 指令基准 手册基本篇	SBCA–360	NX701–□□□□ NX1P2–□□□□ NJ501–□□□□ NJ301–□□□□ NJ101–□□□□	希望了解NJ/NX系列的基本指令规格的详情时。	对各指令(IEC 61131–3标准)的详情进行说明。
NJ/NX系列 CPU单元 用户手册 运动控制篇	SBCE–363	NX701–□□□□ NX1P2–□□□□ NJ501–□□□□ NJ301–□□□□ NJ101–□□□□	希望了解运动控制的设定和编程思路时。	对用于运动控制的CPU单元的设定、动作及编程思路进行说明。
NJ/NX系列 指令基准 手册 运动篇	SBCE–364	NX701–□□□□ NX1P2–□□□□ NJ501–□□□□ NJ301–□□□□ NJ101–□□□□	希望了解运动指令规格的详情时。	对各运动指令的详情进行说明。

手册名称	Man.No.	型号	用途	内容
NJ/NX系列 CPU单元 内置EtherCAT®端口 用户手册	SBCD-358	NX701-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	使用NJ/NX系列CPU单元的内置EtherCAT端口时。	对内置EtherCAT端口进行说明。 对概要、构成、功能、安装进行描述。
NJ/NX系列 CPU单元 内置EtherNet/IP™端口 用户手册	SBCD-359	NX701-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	使用NJ/NX系列CPU单元的内置EtherNet/IP端口时。	对内置EtherNet/IP端口进行说明。 对基本设定、标签数据链接及其他功能进行描述。
NJ系列 数据库连接CPU单元 用户手册	SBCA-411	NJ501-1□20 NJ101-□□20	在NJ系列中使用数据库连接服务功能时。	对数据库连接服务功能进行说明。
NJ系列 配备SECS/GEM CPU单元用户手册	SBCA-412	NJ501-1340	在NJ系列中使用GEM服务功能时。	对GEM服务功能进行说明。
NJ系列 NJ Robotics CPU单元 用户手册	SBCA-421	NJ501-4□□□	在NJ系列中进行机器人控制时。	对机器人控制功能进行说明。
NJ/NX系列 故障诊断 手册	SBCA-361	NX701-□□□□ NX1P2-□□□□ NJ501-□□□□ NJ301-□□□□ NJ101-□□□□	希望了解通过NJ/NX系列检测异常的详情时。	对通过NJ/NX系列系统检测的异常管理的途径和各异常项目进行说明。
Sysmac Studio Version 1 操作手册	SBCA-362	SYSMAC -SE2□□□	希望了解Sysmac Studio的操作方法、功能时。	对Sysmac Studio的操作方法进行说明。
NX系列 EtherCAT®耦合器单元 用户手册	SBCD-361	NX-ECC□□□	希望了解NX系列EtherCAT耦合器单元和EtherCAT从站终端的使用方法时。	对由NX系列 EtherCAT耦合器单元和NX单元构成的EtherCAT从站终端的系统概要和构成方法，以及经由EtherCAT对NX单元进行设定、控制、监控的EtherCAT耦合器单元的硬件、设定方法及功能进行说明。
NX系列 数据基准 手册	SBCA-410	NX-□□□□□□	希望通过一览表查看NX系列各单元的系统构成所需的数据时。	汇总了NX系列各单元的 “消耗功率”、“重量”等系统构建所需的数据。
NX系列 NX单元 用户手册	SBCA-407	NX-ID□□□□ NX-IA□□□□ NX-OC□□□□ NX-OD□□□□ NX-MD□□□□	希望了解NX单元的使用方法时。	对NX单元的硬件、设定方法及功能进行说明。 备有以下单元的手册。 数字I/O单元、模拟I/O单元、系统单元、位置接口单元、通信接口单元、负载传感器输入单元、IO-Link主站单元。
	SBCA-408	NX-AD□□□□ NX-DA□□□□		
	SBCA-440	NX-TS□□□□ NX-HB□□□□		
	SBCA-409	NX-PD1□□□□ NX-PF0□□□□ NX-PC0□□□□ NX-TBX01		
	SBCE-374	NX-EC0□□□□ NX-ECS□□□□ NX-PG0□□□□		
	SBCA-422	NX-CIF□□□		
	SBCA-439	NX-RS□□□□		
	SBCD-370	NX-ILM□□□		
CJ系列 高功能单元 用户手册 (NJ系列连接篇)	SBCC-846	CJ1W-AD□□□ CJ1W-DA□□□ CJ1W-MAD42	希望了解通过NJ系列CPU单元使用CJ系列单元的方法时。	对通过NJ系列CPU单元使用CJ系列单元的方法(访问方法、用户程序I/F等)和注意事项进行说明。 备有以下单元的手册。 模拟I/O单元、绝缘型模拟单元、温度控制单元、ID传感器单元、高速计数器单元、串行通信单元、DeviceNet单元、EtherNet/IP单元、CompoNet主站单元。
	SBCC-847	CJ1W-TC□□□		
	SBCC-848	CJ1W-CT021		
	SBCC-849	CJ1W-PDC15 CJ1W-PH41U CJ1W-AD04U		
	SBCD-353	CJ1W-CRM21		
	SBCD-354	CJ1W-SCU□□		
	SBCD-355	CJ1W-EIP21		
	SBCD-357	CJ1W-DRM21		
	SDGR-703	CJ1W-V680□□□		

手册名称	Man.No.	型号	用途	内容
CX-Protocol 操作手册	SBCA-307	-	希望创建与CJ系列串行通信单元连接的通用外部设备间的数据收发步骤(协议)时。	对CX-Protocol的操作方法进行说明。

手册修订履历

手册的修订记号附加在封面和封底的Man.No.的末尾。

示例

修订记号	修订年月	修订内容
A	2011年7月	初版
B	2011年9月	修正错误
C	2012年3月	CPU单元Ver.1.01、Sysmac Studio Ver.1.02的相应修订
D	2012年5月	CPU单元Ver.1.02、Sysmac Studio Ver.1.03的相应修订
E	2012年8月	CPU单元Ver.1.03、Sysmac Studio Ver.1.04的相应修订
F	2013年2月	CPU单元Ver.1.04、Sysmac Studio Ver.1.05的相应修订
G	2013年4月	CPU单元Ver.1.05、Sysmac Studio Ver.1.06的相应修订
H	2013年6月	CPU单元Ver.1.06、Sysmac Studio Ver.1.07的相应修订
J	2013年9月	CPU单元Ver.1.07、Sysmac Studio Ver.1.08的相应修订
K	2013年12月	CPU单元Ver.1.08、Sysmac Studio Ver.1.09的相应修订
L	2014年7月	CPU单元Ver.1.09、Sysmac Studio Ver.1.10的相应修订
M	2015年1月	CPU单元Ver.1.10、Sysmac Studio Ver.1.12的相应修订
N	2015年4月	• NX系列CPU单元形NX701-□□□□的追加、Sysmac Studio Ver.1.13的相应修订 • NJ系列CPU单元形NJ101-□□□□的追加 • 勘误
P	2015年10月	• 追加硬件版本的相应修订 • 勘误
R	2016年4月	CPU单元Ver.1.11、Sysmac Studio Ver.1.15的相应修订
S	2016年7月	• CPU单元Ver.1.12、Sysmac Studio Ver.1.16的相应修订 • 勘误
T	2016年10月	• CPU单元Ver.1.13、Sysmac Studio Ver.1.17的相应修订 • 勘误
U	2016年11月	勘误
V	2017年1月	勘误
W	2017年4月	勘误

1

指令一览

本章介绍NJ/NX系列可使用的指令一览。

指令一览

种类	指令	名称	功能概要	页码
梯形图指令	LD	加载	读取BOOL型变量值。	2–16
	LDN	加载・非	读取BOOL型变量的取反值。	2–16
	AND	与	获取BOOL型变量值与输入值的逻辑积。	2–18
	ANDN	与・非	获取BOOL型变量的取反值与输入值的逻辑积。	2–18
	OR	或	获取BOOL型变量值与输入条件的逻辑和。	2–21
	ORN	或・非	获取BOOL型变量的取反值与输入条件的逻辑和。	2–21
	Out	输出	将前段的逻辑运算处理的结果输出至BOOL型变量。	2–24
	OutNot	输出非	将前段的逻辑运算处理的结果的取反值输出至BOOL型变量。	2–24
ST语法指令	IF	选择	根据指定条件表达式的评估结果，从两者中选择要执行的语句。	2–28
	CASE	多分支	根据指定的整数表达式的值，从多个语句中选择要执行的语句。	2–32
	WHILE	重复	在指定的条件表达式的评估结果为TRUE期间，重复执行某个语句。	2–36
	REPEAT	重复	执行某个语句一次，在指定的条件表达式变为TRUE之前，重复执行该语句。	2–38
	EXIT	循环中断	中断最内侧FOR指令、WHILE指令或REPEAT指令的重复处理。	2–40
	RETURN	复位	结束函数或功能块，将处理恢复为调用源。	2–43
	FOR	重复	指定重复的条件，重复执行FOR语句至END_FOR语句中的某个语句。	2–44
时序输入指令	R_TRIG,Up	上升沿微分	输入信号处于上升沿(FALSE→TRUE)时，仅1个任务周期输出TRUE。	2–46
	F_TRIG,Down	下降沿微分	输入信号处于下降沿(TRUE→FALSE)时，仅1个任务周期输出TRUE。	2–46
	TestABit	位测试	输出位串的指定位的值。	2–49
	TestABitN	非位测试	输出位串的指定位的取反值。	2–49
时序输出指令	RS	复位优先保持	保持BOOL型变量值。 置位输入与复位输入同时TRUE时，复位输入优先。	2–52
	SR	置位优先保持	保持BOOL型变量值。 置位输入与复位输入同时TRUE时，置位输入优先。	2–54
	Set	置位	设置BOOL型变量为TRUE。	2–56
	Reset	复位	设置BOOL型变量为FALSE。	2–56
	SetBits	多位置位	设置位串数据的连读多个位为TRUE。	2–59
	ResetBits	多位复位	设置位串数据的连读多个位为FALSE。	2–59
	SetABit	1位置位	设置位串数据的指定位为TRUE。	2–61
	ResetABit	1位复位	设置位串数据的指定位为FALSE。	2–61
	OutABit	1位输出	设置位串数据的指定位为TRUE/FALSE。	2–63
时序控制指令	End	结束	结束程序当前任务周期内的执行。	2–66
	RETURN	复位	结束函数或功能块，将处理恢复为调用源。	2–67
	MC	主站控制开始	表示主站控制区域的起始点，执行基于主站控制的复位。	2–68
	MCR	主站控制结束	表示主站控制区域的终止点。	2–68
	JMP	跳转	处理转移至指定的跳转位置。	2–82
	FOR	重复开始	表示重复处理起始位置的同时，指定重复的条件。	2–84
	NEXT	循环结束	表示重复处理的终止位置。	2–84
	BREAK	循环中断	中断最内侧FOR指令至NEXT指令的重复处理。	2–90

种类	指令	名称	功能概要	页码
比较指令	EQ,=	比较EQ	判定多个数据是否均相等。	2–94
	NE,<>	比较NE	判定2个数据是否不同。	2–96
	LT,<	比较LT	比较多个数值的大小(<)。	2–98
	LE,<=	比较LE	比较多个数值的大小(<=)。	2–98
	GT,>	比较GT	比较多个数值的大小(>)。	2–98
	GE,>=	比较GE	比较多个数值的大小(>=)。	2–98
	EQascii	字符串比较EQ	判定多个字符串是否均相等。	2–101
	NEascii	字符串比较NE	判定2个字符串是否不同。	2–103
	LTascii	字符串比较LT	比较多个字符串的大小(<)。	2–105
	LEascii	字符串比较LE	比较多个字符串的大小(<=)。	2–105
	GTascii	字符串比较GT	比较多个字符串的大小(>)。	2–105
	GEascii	字符串比较GE	比较多个字符串的大小(>=)。	2–105
	Cmp	比较	比较2个数值。	2–108
	ZoneCmp	区域比较	判定比较数据是否在指定的上限值与下限值之间。	2–110
	TableCmp	表比较	将比较数据与比较表指定的多个定义区间进行比较。	2–113
	AryCmpEQ	数组整体比较EQ	判定2个数组的各元素是否相等。	2–116
	AryCmpNE	数组整体比较NE	判定2个数组的各元素是否不同。	2–116
	AryCmpLT	数组整体比较LT	比较2个数组的各元素的大小(<)。	2–118
	AryCmpLE	数组整体比较LE	比较2个数组的各元素的大小(<=)。	2–118
	AryCmpGT	数组整体比较GT	比较2个数组的各元素的大小(>)。	2–118
	AryCmpGE	数组整体比较GE	比较2个数组的各元素的大小(>=)。	2–118
	AryCmpEQV	数组元素比较EQ	判定数组的各元素与数值是否相等。	2–121
	AryCmpNEV	数组元素比较NE	判定数组的各元素与数值是否不同。	2–121
	AryCmpLTV	数组元素比较LT	比较数组的各元素与数值的大小(<)。	2–123
	AryCmpLEV	数组元素比较LE	比较数组的各元素与数值的大小(<=)。	2–123
	AryCmpGTV	数组元素比较GT	比较数组的各元素与数值的大小(>)。	2–123
	AryCmpGEV	数组元素比较GE	比较数组的各元素与数值的大小(>=)。	2–123
定时器指令	TON	ON延时定时器	在启动经过设定时间后，输出TRUE的定时器。	2–128
	TOF	OFF延迟定时器	在启动经过设定时间后，输出FALSE的定时器。	2–133
	TP	脉冲输出	在启动经过设定时间后，输出TRUE的定时器。	2–136
	Accumulation Timer	累计定时器	累计定时器输入为TRUE的时间的定时器。	2–139
	Timer	100ms定时器	在启动经过设定时间后，输出TRUE的定时器。设定时间的最小单位为100ms。	2–142
计数器指令	CTD	减法计数器	每次输入计数器输入信号时进行减法运算的计数器。预设值、计数值的数据类型为INT。	2–146
	CTD_**	减法计数器组	每次输入计数器输入信号时进行减法运算的计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。	2–148
	CTU	加法计数器	每次输入计数器输入信号时进行加法运算的计数器。预设值、计数值的数据类型为INT。	2–151
	CTU_**	加法计数器组	每次输入计数器输入信号时进行加法运算的计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。	2–153
	CTUD	可逆计数器	根据加法计数器输入和减法计数器输入进行加减法运算的计数器。预设值、计数值的数据类型为INT。	2–156
	CTUD_**	可逆计数器组	根据加法计数器输入和减法计数器输入进行加减法运算的计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。	2–160

种类	指令	名称	功能概要	页码
算术指令	ADD,+	加法	对整数、实数进行加法运算。或拼接字符串。	2–166
	AddOU,+OU	加法(带溢出检查)	对整数、实数进行加法运算。并且，对整数的相加结果进行溢出检查。	2–170
	SUB,–	减法	对整数、实数进行减法运算。	2–173
	SubOU,–OU	减法(带溢出检查)	对整数、实数进行减法运算。并且，对整数的相减结果进行溢出检查。	2–176
	MUL,*	乘法	对整数、实数进行乘法运算。	2–180
	MulOU,*OU	乘法(带溢出检查)	输出对整数、实数进行乘法运算的结果。并且，对整数的相乘结果进行溢出检查。	2–184
	DIV,/	除法	对整数、实数进行除法运算。	2–187
	MOD	余数	计算对整数进行除法运算后的余数。	2–190
	ABS	绝对值	计算整数、实数的绝对值。	2–192
	RadToDeg	弧度→角度转换	将实数的单位从弧度(rad)转换为度(°)。	2–194
	DegToRad	角度→弧度转换	将实数的单位从度(°)转换为弧度(rad)。	2–194
	SIN	SIN运算	计算实数的sin(正弦值)。	2–196
	COS	COS运算	计算实数的cos(余弦值)。	2–196
	TAN	TAN运算	计算实数的tan(正切值)。	2–196
	ASIN	SIN^{-1}运算	计算实数的sin^{-1}(反正弦值)。	2–199
	ACOS	COS^{-1}运算	计算实数的cos^{-1}(反余弦值)。	2–199
	ATAN	TAN^{-1}运算	计算实数的tan^{-1}(反正切值)。	2–199
	SQRT	平方根运算	计算实数的平方根($\sqrt{\ }$)。	2–203
	LN	自然对数运算	计算实数的自然对数。	2–205
	LOG	常用对数运算	计算实数的常用对数。	2–205
	EXP	自然指数运算	计算自然对数的底e的指数函数。	2–208
	EXPT,**	乘方运算	计算2个实数的乘幂。	2–210
	Inc	增量	对整数值进行增量。	2–214
	Dec	减量	对整数值进行减量。	2–214
	Rand	生成随机数	生成伪随机数。	2–216
	AryAdd	数组整体加法	对2个数组的各元素进行加法运算。	2–218
	AryAddV	数组元素加法	数组的各元素加上相同数值。	2–220
	ArySub	数组整体减法	对2个数组的各元素进行减法运算。	2–222
	ArySubV	数组元素减法	数组的各元素减去相同数值。	2–224
	AryMean	数组元素的平均值计算	计算数组元素的平均值。	2–226
	ArySD	数组元素的标准偏差	计算数组元素的标准偏差。	2–228
	ModReal	实数余数	计算对实数进行除法运算后的余数。	2–230
	Fraction	提取实数小数部分	提取实数的小数部分。	2–232
	CheckReal	实数检查	判定实数是否无限大或非数。	2–234
BCD转换指令	**_BCD_TO_***	BCD→无符号整数转换组	将BCD的位串转换为无符号整数。	2–238
	_TO_BCD_*	无符号整数→BCD转换组	将无符号整数转换为BCD的位串。	2–241
	BCD_TO_**	BCD组→无符号整数转换组	将BCD的位串转换为无符号整数。	2–244
	BCDsToBin	带符号BCD→带符号整数转换	将带符号BCD的位串转换为带符号整数。	2–247
	BinToBCDs_**	带符号整数→BCD转换组	将带符号整数转换为带符号BCD的位串。	2–250
	AryToBCD	数组整体BCD转换	将无符号整数数组的各元素转换为BCD的位串。	2–253
	AryToBin	数组整体无符号整数转换	将BCD的位串数组的各元素转换为无符号整数。	2–255

种类	指令	名称	功能概要	页码
数据类型转换指令	**_TO_***(整数→整数转换组)	整数→整数转换组	将整数转换为不同数据类型的整数。	2–258
	_TO_*(整数→位串转换组)	整数→位串转换组	将整数转换为位串。	2–261
	_TO_*(整数→实数转换组)	整数→实数转换组	将整数转换为实数。	2–264
	_TO_*(位串→整数转换组)	位串→整数转换组	将位串转换为整数。	2–266
	_TO_*(位串→位串转换组)	位串→位串转换组	将位串转换为不同数据类型的位串。	2–268
	_TO_*(位串→实数转换组)	位串→实数转换组	将位串转换为实数。	2–270
数据类型转换指令	**_TO_***(实数→整数转换组)	实数→整数转换组	将实数转换为整数。	2–272
	_TO_*(实数→位串转换组)	实数→位串转换组	将实数转换为位串。	2–275
	_TO_*(实数→实数转换组)	实数→实数转换组	将实数转换为不同数据类型的实数。	2–277
	**_TO_STRING(整数→字符串转换组)	整数→字符串转换组	将整数转换为字符串。	2–279
	**_TO_STRING(位串→字符串转换组)	位串→字符串转换组	将位串转换为字符串。	2–281
	**_TO_STRING(实数→字符串转换组)	实数→字符串转换组	将实数转换为字符串。	2–283
	RealToFormatString	实数(REAL)→带格式字符串转换	将REAL型变量按照指定格式转换为字符串。	2–285
	LrealToFormatString	实数(LREAL)→带格式字符串转换	将LREAL型变量按照指定格式转换为字符串。	2–290
	STRING_TO_**(字符串→整数转换组)	字符串→整数转换组	将字符串转换为整数。	2–295
	STRING_TO_**(字符串→位串转换组)	字符串→位串转换组	将字符串转换为位串。	2–297
	STRING_TO_**(字符串→实数转换组)	字符串→实数转换组	将字符串转换为实数。	2–299
	TO_**(整数转换组)	整数转换组	将整数、位串、实数、字符串转换为整数。	2–302
	TO_**(位串转换组)	位串转换组	将整数、位串、实数、字符串转换为位串。	2–304
	TO_**(实数转换组)	实数转换组	将整数、位串、实数、字符串转换为实数。	2–306
	EnumToNum	枚举体→整数转换	将枚举体转换为DINT型。	2–308
	NumToEnum	整数→枚举体转换	将DINT型转换为枚举体。	2–310
	TRUNC	实数舍去	舍去实数的小数部分的第1位，取整数。	2–312
	Round	实数取整	将实数的小数部分的第1位五舍五入，取整数。	2–312
	RoundUp	实数进位	将实数的小数部分的第1位进位，取整数。	2–312

种类	指令	名称	功能概要	页码
位串运算指令	AND,&	逻辑积	计算多个布尔、位串的每1位的逻辑积。	2–316
	OR	逻辑和	计算多个布尔、位串的每1位的逻辑和。	2–316
	XOR	异或	对多个布尔、位串的每1位进行异或运算。	2–316
	XORN	同或	对多个布尔、位串的每1位进行同或运算。	2–319
	NOT	位取反	对布尔、位串的各个位进行取反。	2–321
	AryAnd	数组逻辑积	计算数组间各元素的布尔、位串的每1位的逻辑积。	2–322
	AryOr	数组逻辑和	计算数组间各元素的布尔、位串的每1位的逻辑和。	2–322
	AryXor	数组异或	对数组间各元素的布尔、位串的每1位进行异或运算。	2–322
	AryXorN	数组同或	对数组间各元素的布尔、位串的每1位进行同或运算。	2–322
选择指令	SEL	位选择	在2个选项中选择1个。	2–326
	MUX	多路复用器	在2 ~ 5个选项中选择1个。	2–328
	LIMIT	上下限限位限制	通过设定的上下限值，限制输入变量值。	2–331
	Band	死区控制	控制死区。	2–333
	Zone	死区控制	将输入值加上偏置值。	2–335
	MAX	最大值检索	检索2 ~ 5个数值中的最大值。	2–337
	MIN	最小值检索	检索2 ~ 5个数值中的最小值。	2–337
	AryMax	数组变量的最大值检索	检索1维数组元素的最大值。	2–339
	AryMin	数组变量的最小值检索	检索1维数组元素的最小值。	2–339
	ArySearch	数组检索	在1维数组中检索指定值。	2–342
数据传送指令	MOVE	传送	将变量和常数值传送至其它变量。	2–346
	MoveBit	位传送	传送位串内的1位。	2–349
	MoveDigit	数字传送	传送位串内的多个数位(1个数位为4位)。	2–351
	TransBits	多位传送	传送位串内的多位。	2–353
	MemCopy	存储器复制	传送多个数组元素。传送源和传送目标仅限相同数据类型。	2–355
	SetBlock	块设定	将变量和常数的值传送至多个数组元素。	2–357
	Exchange	数据交换	更换2个变量值。	2–359
	AryExchange	数组数据交换	更换2个数组间的元素。	2–361
	AryMove	数组的传送	传送多个数组元素。传送源和传送目标的数据类型可不同。	2–363
	Clear	初始化	初始化变量。	2–365
	Copy**ToNum(位串→带符号整数)	位模式复制(位串→带符号整数)组	将位串的内容直接复制到带符号整数。	2–367
	Copy**To***(位串→实数)	位模式复制(位串→实数)组	将位串的内容直接复制到实数。	2–369
	CopyNumTo**(带符号整数→位串)	位模式复制(带符号整数→位串)组	将带符号整数的内容直接复制到位串。	2–371
	CopyNumTo**(带符号整数→实数)	位模式复制(带符号整数→实数)组	将带符号整数的内容直接复制到实数。	2–373
	Copy**To***(实数→位串)	位模式复制(实数→位串)组	将实数的内容直接复制到位串。	2–375
	Copy**ToNum(实数→带符号整数)	位模式复制(实数→带符号整数)组	将实数的内容直接复制到带符号整数。	2–376

种类	指令	名称	功能概要	页码
移位指令	AryShiftReg	移位寄存器	将数组元素组成的整个位串左移1位，并将输入值插入最低位。	2–380
	AryShiftRegLR	左右移位寄存器	将数组元素组成的整个位串左移(或右移)1位，并将输入值插入最低位(或最高位)。	2–382
	ArySHL	数组左移N个元素	将数组元素左移(高位方向)多个元素。	2–385
	ArySHR	数组右移N个元素	将数组元素右移(低位方向)多个元素。	2–385
	SHL	左移N位	将位串左移(高位方向)多位。	2–388
	SHR	右移N位	将位串右移(低位方向)多位。	2–388
	NSHLC	N位数据带CY左移N位	通过带(CY)进位标志将位串数组左移(高位方向)多位。	2–390
	NSHRC	N位数据带CY右移N位	通过带(CY)进位标志将位串数组右移(低位方向)多位。	2–390
	ROL	左旋转N位	将位串左转(高位方向)多位。	2–392
	ROR	右旋转N位	将位串右旋转(低位方向)多位。	2–392
数据转换指令	Swap	字节交换	更换16位值的高位字节和低位字节。	2–396
	Neg	符号取反	对数值的符号进行取反。	2–397
	Decoder	位译码器	将最大256位组成的数组元素指定的1位设置为TRUE，其它位设置为FALSE。	2–399
	Encoder	位编码器	计算最大256位组成的数组元素中，值为TRUE的位位置。	2–402
	BitCnt	位计数	对位串内的值为TRUE的位的总数进行计数。	2–404
	ColmToLine_**	位串→位行转换组	提取各数组元素指定位置的位的值，并作为位串输出。	2–405
	LineToColm	位行→位串转换	分解位串，输出至数组元素指定的位位置。	2–407
	Gray	格雷码转换	将格雷码转换为角度。	2–409
	UTF8ToSJIS	字符代码转换(UTF–8→SJIS)	将字符代码UTF–8的字符串转换为字符代码SJIS的BYTE型数组。	2–414
	SJISToUTF8	字符代码转换(SJIS→UTF–8)	将文字代码SJIS的BYTE型数组转换为文字代码UTF–8的字符串。	2–416
	PWLApprox	折线近似转换(带折线数据检查)	检查折线数据是否有效，对整数、实数进行折线近似计算。	2–418
	PWLApproxNoLineChk	折线近似转换(无折线数据检查)	不检查折线数据是否有效，对整数、实数进行折线近似计算。	2–418
	PWLLineChk	折线数据检查	判定折线近似转换(无折线数据检查)指令使用的折线数据是否按X坐标升序排列。	2–423
	MovingAverage	移动平均	计算出移动平均值。	2–426
	DispartReal	实数的尾数、指数分离	将实数分离为带符号尾数部分和指数部分。	2–432
	UniteReal	尾数、指数组合成实数	将带符号的尾数部分和指数部分组合成实数。	2–435
	NumToDecString	固定长度10进制字符串转换	将整数转换为固定长度的10进制字符串格式。	2–437
	NumToHexString	固定长度16进制字符串转换	将整数转换为固定长度的16进制字符串格式。	2–437
	HexStringToNum_**	16进制字符串→数值转换组	将16进制字符串格式转换为整数。	2–440
	FixNumToString	固定小数点数→字符串转换	将带符号的固定小数点数转换为10进制字符串格式。	2–442
	StringToFixNum	字符串→固定小数点数转换	将10进制字符串转换为带符号的固定小数点数形式。	2–444
	DtToString	日期时间→字符串转换	将日期时间转换为字符串格式。	2–447
	DateToString	日期→字符串转换	将日期转换为字符串格式。	2–449
	TodToString	时刻→字符串转换	将时刻转换为字符串格式。	2–450
	GrayToBin_**	格雷码→BIN码转换组	将格雷码转换为位串。	2–452

种类	指令	名称	功能概要	页码
数据转换指令	BinToGray_**	BIN码→格雷码转换组	将位串转换为格雷码。	2–452
	StringToAry	字符串→数组转换	将字符串转换为BYTE型数组。	2–455
	AryToString	数组→字符串转换	将BYTE型数组转换为字符串。	2–457
	DispartDigit	4位分离	以4位为单位分离位串。	2–459
	UniteDigit_**	4位结合组	将以4位为单位的数据组合成位串。	2–461
	Dispart8Bit	以字节为单位的数据分离	以1字节为单位分离位串。	2–463
	Unite8Bit_**	以字节为单位的数据组合组	将以1字节为单位的数据组合成位串。	2–465
	ToAryByte	转换为字节数组	以1字节为单位分割变量后，保存至BYTE型数组。	2–467
	AryByteTo	从字节数组转换	组合BYTE型数组元素后，保存至变量。	2–472
	SizeOfAry	获取数组元素数	获取数组元素数。	2–477
	PackWord	合并2字节	将2个单字节数据合并成1个2字节数据。	2–479
	PackDword	合并4字节	将4个单字节数据合并成1个4字节数据。	2–481
堆栈/表指令	StackPush	堆栈数据保存	将值保存至堆栈中。	2–484
	StackFIFO	先入先出	提取堆栈最低位的值。	2–493
	StackLIFO	后入先出	提取堆栈最高位的值。	2–493
	StackIns	堆栈数据插入	将值插入堆栈的任意位置。	2–496
	StackDel	堆栈数据删除	删除堆栈任意位置的值。	2–498
	RecSearch	记录检索	在以结构体为元素的数组中，按指定方法检索与检索关键词一致的要素。	2–500
	RecRange Search	范围指定记录检索	在以结构体为元素的数组中，按指定方法检索与检索条件的范围一致的要素。	2–505
	RecSort	记录排序	将以结构体为元素的数组的要素进行分类。	2–510
	RecNum	记录数获取	计算以结构体为元素的数组的结尾数据为止的记录数。	2–515
	RecMax	记录最大值检索	在以结构体为元素的数组中，检索指定的结构要素的最大值。	2–517
	RecMin	记录最小值检索	在以结构体为元素的数组中，检索指定的结构要素的最小值。	2–517
FCS指令	StringSum	和值计算	计算字符串的和值。	2–522
	StringLRC	LRC值计算(字符串)	计算字符串的LRC值(水平奇偶校验)。	2–524
	String CRCCCITT	CRC–CCITT值计算(字符串)	按照XMODEM方式计算字符串的CRC–CCITT值。	2–526
	StringCRC16	CRC–16值计算(字符串)	按照MODBUS方式计算字符串的CRC–16值。	2–528
	AryLRC_**	LRC值计算(数组)组	计算数组的LRC值。	2–530
	AryCRCCCITT	CRC–CCITT值计算(数组)	按照XMODEM方式计算数组的CRC–CCITT值。	2–532
	AryCRC16	CRC–16值计算(数组)	按照MODBUS方式计算数组的CRC–16值。	2–534

种类	指令	名称	功能概要	页码
字符串指令	CONCAT	字符串・连接	连接2～5个字符串。	2–538
	LEFT	字符串・从左侧提取	从字符串的左侧(开头)提取指定数量的字符串。	2–540
	RIGHT	字符串・从右侧提取	从字符串的右侧(末尾)提取指定数量的字符串。	2–540
	MID	字符串・从任意位置提取	从字符串的任意位置提取指定数量的字符串。	2–542
	FIND	字符串・检索	检索字符串中指定字符串的位置。	2–544
	LEN	字符串・长度检测	计算字符串的字符数。	2–546
	REPLACE	字符串・置换	用其它字符串置换部分字符串。	2–547
	DELETE	字符串・删除	删除部分或全部字符串。	2–549
	INSERT	字符串・插入	在字符串中插入其它字符串。	2–551
	GetByteLen	字符串・获取字节数	对字符串的字节数进行计数。	2–553
	ClearString	字符串・清除	清除字符串。	2–554
	ToUCase	字符串・大写字母转换	将字符串中的半角字母均转换为大写字母。	2–555
	ToLCase	字符串・小写字母转换	将字符串中的半角字母均转换为小写字母。	2–555
	TrimL	字符串・左侧调整	删除字符串开头的空格。	2–557
	TrimR	字符串・右侧调整	删除字符串末尾的空格。	2–557
	AddDelimiter	带分隔符的字符串写入	将整个结构体的值转换为带分隔符的字符串。	2–559
	SubDelimiter	字符串读取分隔符删除	从字符串中读取由分隔符隔开的数据，保存为结构体结构要素的值。	2–569
时间/时刻指令	ADD_TIME	时间加法	对时间和时间进行加法运算。	2–582
	ADD_TOD_TIME	时刻和时间的加法	将时刻加上时间。	2–584
	ADD_DT_TIME	日期时刻和时间的加法	将日期时刻加上时间。	2–586
	SUB_TIME	时间减法	将时间减去时间。	2–588
	SUB_TOD_TIME	时刻和时间的减法	将时刻减去时间。	2–590
	SUB_TOD_TOD	时刻减法	将时刻减去时刻。	2–592
	SUB_DATE_DATE	日期减法	将日期减去日期。	2–593
	SUB_DT_DT	日期时刻减法	将日期时刻减去日期时刻。	2–594
	SUB_DT_TIME	日期时刻和时间的减法	将日期时刻减去时间。	2–596
	MULTIME	时间乘法	按指定乘数对时间进行乘法运算。	2–598
	DIVTIME	时间除法	按指定除数对时间进行除法运算。	2–600
	CONCAT_DATE_TOD	日期和时刻的组合	组合日期和时刻。	2–602
	DT_TO_TOD	从日期时刻中截取时刻	从日期时刻中截取时刻。	2–604
	DT_TO_DATE	从日期时刻中截取日期	从日期时刻中截取日期。	2–606
	SetTime	时钟补偿	补偿系统时刻。	2–608
	GetTime	时刻获取	读取当前时刻。	2–610
	DtToSec	日期时刻→秒转换	将日期时刻转换为1970年1月1日0时0分0秒起的秒数。	2–613
	DateToSec	日期→秒转换	将日期转换为1970年1月1日0时0分0秒起的秒数。	2–615
	TodToSec	时刻→秒转换	将时刻转换为0时0分0秒起的秒数。	2–617
	SecToDt	秒→日期时刻转换	将1970年1月1日0时0分0秒起的秒数转换为日期时刻。	2–618
	SecToDate	秒→日期转换	将1970年1月1日0时0分0秒起的秒数转换为日期。	2–620
	SecToTod	秒→时刻转换	将0时0分0秒起的秒数转换为时刻。	2–622
	TimeToNanoSec	时间→纳秒转换	将时间转换为纳秒数。	2–624
	TimeToSec	时间→秒转换	将时间转换为秒数。	2–625

种类	指令	名称	功能概要	页码
时间/时刻指令	NanoSecToTime	纳秒→时间转换	将纳秒数转换为时间。	2–626
	SecToTime	秒→时间转换	将秒数转换为时间。	2–627
	ChkLeapYear	闰年判别	判定指定的年份是否为闰年。	2–629
	GetDaysOf Month	月的天数获取	获取指定月的天数。	2–630
	DaysToMonth	天数→月转换	根据从1月1日起的天数，计算出该日为哪月。	2–632
	GetDayOfWeek	星期获取	获取指定年月日的星期。	2–634
	GetWeekOfYear	周获取	计算指定年月日为当年的第几周。	2–636
	DtToDateStruct	时刻分解	将日期时刻分解为 “年”、“月”、“日”、“ 时”、“ 分”、“秒”、“纳秒”。	2–638
	DateStructToDt	时刻组合	组合分解为 “年”、“月”、“日”、“ 时”、“ 分”、“秒”、“纳秒”的日期时刻。	2–640
	TruncTime	时间舍去	舍去TIME型变量中小于指定单位的值。	2–642
	TruncDt	日期时刻舍去	舍去DT型变量中小于指定单位的值。	2–646
	TruncTod	时刻舍去	舍去TOD型变量中小于指定单位的值。	2–650
模拟控制指令	PIDAT	带自动调谐的PID运算	执行带自动调谐的PID运算(目标值滤波器型2自由度PID控制)。	2–656
	PIDAT _HeatCool	带自动调谐的加热冷却PID运算	执行带自动调谐的加热冷却PID运算(目标值滤波器型2自由度PID控制)。	2–682
	TimeProportion alOut	分时比例输出	将操作量转换为分时比例输出。	2–719
	LimitAlarm_**	上下限警报组	输入值小于下限设定值或超过上限设定值时输出警报。	2–734
	LimitAlarmDv_**	上下限偏差警报组	输入值与基准值的偏差低于下限偏差设定值或超过上限偏差设定值时输出警报。	2–738
	LimitAlarmDvSt bySeq_**	带待机时序的上下限偏差警报组	带待机时序，输出上下限偏差警报。	2–742
	ScaleTrans	比例转换	将输入值从输入范围转换为输出范围。	2–756
	AC_Step Program	步程序	根据指定程序模式，按任务周期计算当前目标值和预测目标值。	2–759
系统控制指令	TraceSamp	数据跟踪采样	执行数据跟踪的采样。	2–784
	TraceTrig	数据跟踪触发条件成立	触发数据跟踪。	2–787
	GetTraceStatus	数据跟踪状态读取	读取数据跟踪的执行状态。	2–790
	SetAlarm	用户异常发生	发生用户异常。	2–793
	ResetAlarm	用户异常解除	解除用户异常。	2–798
	GetAlarm	用户异常状态获取	获取发生的用户异常的最重要事件的重要程度(用户异常等级1 ~ 8)和最重要事件的代码。	2–800
	ResetPLCError	PLC异常解除	解除PLC功能模块中发生的控制器异常。	2–802
	GetPLCError	PLC异常状态获取	获取PLC功能模块中发生的控制器异常的最重要状态(部分停止故障、轻度故障)和最重要的事件代码。	2–805
	ResetCJBError	I/O总线异常解除	解除I/O总线中发生的控制器异常。	2–807
	GetCJBError	I/O总线异常状态获取	获取NJ系列CPU单元I/O总线中发生的控制器异常的最重要的状态和最重要的事件代码。	2–809
	GetEIPError	EtherNet/IP异常状态获取	获取EtherNet/IP功能模块中发生的控制器异常的最重要状态(部分停止故障、轻度故障)和最重要的事件代码。	2–811
	ResetMCError	运动控制异常解除	解除运动控制功能模块中发生的控制器异常。	2–813
	GetMCError	运动控制异常状态获取	获取运动控制功能模块中发生的控制器异常的最重要状态(部分停止故障、轻度故障)和最重要的事件代码。	2–818
	ResetECError	EtherCAT异常解除	解除EtherCAT功能模块中发生的控制器异常。	2–820
	GetECError	EtherCAT异常状态获取	检测到EtherCAT主站功能模块中发生的异常。	2–822
	ResetNXBError	NX总线异常解除	解除NX总线功能模块中发生的控制器异常。	2–825
	GetNXBError	NX总线异常状态获取	获取NX系列CPU单元的NX总线功能模块中发生的控制器异常的最重要的状态。	2–827

种类	指令	名称	功能概要	页码
系统控制指令	GetNXUnitError	NX单元异常状态获取	获取NX系列CPU单元的NX总线功能模块或NX单元中发生的控制器异常的最重要的状态和最重要的事件代码。	2-829
	SetInfo	用户信息生成	生成用户信息。	2-836
	ResetUnit	单元重启	重启CPU高功能单元或高功能I/O单元。	2-838
	GetNTPStatus	NTP状态读取	读取NTP的状态。	2-842
	RestartNXUnit	NX单元重启	重启EtherCAT耦合器单元或NX单元。	2-844
	NX_Change WriteMode	NX单元写入模式变更	将EtherCAT耦合器单元或NX单元变更成可写入数据的模式。	2-849
	NX_SaveParam	NX单元参数保存	保存写入EtherCAT耦合器单元或NX单元的数据。	2-854
	NX_ReadTotal PowerOnTime	NX单元累计通电时间读取	读取通信耦合器单元及NX单元保存的累计通电时间。	2-859
	PLC_ReadTotal PowerOnTime	PLC累计通电时间读取	读取指定CPU单元保存的累计通电时间。	2-866
程序控制指令	PrgStart	程序执行指令	发出执行指定程序的指令。	2-870
	PrgStop	程序停止指令	发出指定程序的停止指令。	2-878
	PrgStatus	程序状态读取	调用指定程序的状态。	2-896
EtherCAT通信指令	EC_CoESDO Write	CoE SDO写入	将值写入EtherCAT网络上指定从站的CoE对象。	2-902
	EC_CoESDO Read	CoE SDO读取	从EtherCAT网络上指定从站的CoE对象中读取值。	2-905
	EC_StartMon	EtherCAT分组监控功能启动	开始执行EtherCAT通信的分组监控功能。	2-910
	EC_StopMon	EtherCAT分组监控功能停止	停止EtherCAT通信的分组监控功能。	2-916
	EC_SaveMon	EtherCAT分组保存	将EtherCAT通信的分组数据保存至CPU单元主存储器的内部文件中。	2-918
	EC_CopyMon	EtherCAT分组传送	将EtherCAT通信的CPU单元主存储器中内部文件内的分组数据传送至SD存储卡。	2-920
	EC_Disconnect Slave	EtherCAT从站脱离	将EtherCAT网络上的指定从站从网络中脱离。	2-922
	EC_Connect Slave	重新加入EtherCAT从站	将EtherCAT网络上的指定从站重新加入网络。	2-930
	EC_Change EnableSetting	EtherCAT从站有效/无效切换	切换EtherCAT从站的有效/无效。	2-932
	NX_WriteObj	NX对象写入	将数据写入EtherCAT耦合器单元或NX单元的NX对象。	2-951
	NX_ReadObj	NX对象读取	从EtherCAT耦合器单元或NX单元的NX对象中读取数据。	2-966
IO-Link通信指令	IOL_ReadObj	IO-Link设备对象读取	从IO-Link设备的对象中读取数据。	2-974
	IOL_WriteObj	IO-Link设备对象写入	在IO-Link设备的对象中写入数据。	2-982
EtherNet/IP通信指令	CIPOpen	CIPClass3 (Large_Forward_ Open)连接建立	按1994字节的数据长度，为指定对象节点建立CIPClass3(Large_Forward_Open)连接。	2-992
	CIPOpenWith DataSize	CIP Class3 连接建立(带大小指定)	为指定对象节点建立可收发指定数据长度以下的Class3 Explicit信息的CIP Class3连接。	2-1001
	CIPRead	变量读取(Class3 Explicit信息)	读取CIP网络上其它站控制器的变量值(Class3 Explicit信息)。	2-1004
	CIPWrite	变量写入(Class3 Explicit信息)	将值写入CIP网络上其它站控制器的变量(Class3 Explicit信息)。	2-1009
	CIPSend	任意Explicit信息发送(Class3)	将Class3的CIP信息发送至CIP网络的指定设备。	2-1015

种类	指令	名称	功能概要	页码
EtherNet/IP通信指令	CIPClose	CIP Class3连接的切断	切断指定句柄的CIP Class3的连接。	2–1020
	CIPUCMMRead	变量读取(UCMM Explicit信息)	读取指定CIP网络上其它站控制器的变量值(UCMM Explicit信息)。	2–1022
	CIPUCMMWrite	变量写入(UCMM Explicit信息)	将值写入CIP网络上其它站控制器的变量(UCMM Explicit信息)。	2–1027
	CIPUCMMSend	任意Explicit信息发送(UCMM)	将UCMM的CIP信息发送至CIP网络的指定设备。	2–1034
	SktUDPCreate	UDP Socket建立	执行内置EtherNet/IP的UDP Socket的建立请求(服务器端口打开)。	2–1045
	SktUDPRcv	UDP Socket接收	读取内置EtherNet/IP的UDP Socket接收缓存中的数据。	2–1052
	SktUDPSend	UDP Socket发送	从内置EtherNet/IP的UDP端口发送数据。	2–1055
	SktTCPAccept	TCP Socket接受	执行内置EtherNet/IP的TCP Socket的接受请求。	2–1058
	SktTCPConnect	TCP Socket连接	从内置EtherNet/IP连接目标TCP端口。	2–1061
	SktTCPRcv	TCP Socket接收	读取内置EtherNet/IP的指定TCP Socket的接收缓存中的数据。	2–1069
	SktTCPSend	TCP Socket发送	从内置EtherNet/IP的指定TCP端口发送数据。	2–1072
	SktGetTCP Status	TCP Socket的状态读取	读取TCP Socket的状态。	2–1075
	SktClose	TCP/UDP Socket闭合	闭合内置EtherNet/IP的指定TCP Socket或UDP Socket。	2–1078
	SktClearBuf	TCP/UDP Socket接收缓存清除	清除内置EtherNet/IP的指定TCP Socket或UDP Socket的接收缓存。	2–1081
	SktSetOption	TCP Socket选项设定	设定内置EtherNet/IP的指定TCP Socket的Socket选项。	2–1083
	ChangeIPAdr	IP地址变更	变更内置EtherNet/IP端口或EtherNet/IP单元的IP地址。	2–1088
	ChangeFTP Account	FTP账号变更	变更内置EtherNet/IP端口或EtherNet/IP单元的FTP登录名和密码。	2–1097
	ChangeNTP ServerAdr	NTP服务器地址变更	变更内置EtherNet/IP端口或EtherNet/IP单元的NTP服务器地址。	2–1101
	FTPGetFileList	FTP服务器的文件列表获取	获取FTP服务器内的文件列表。	2–1105
	FTPGetFile	从FTP服务器下载文件	从FTP下载文件。	2–1120
	FTPPutFile	文件上传至FTP服务器	文件上传至FTP服务器。	2–1128
	FTPRemoveFile	FTP服务器的文件删除	删除FTP服务器的文件。	2–1138
	FTPRemoveDir	FTP服务器的目录删除	删除FTP服务器的目录。	2–1148
串行通信指令	ExecPMCR	协议宏	请求执行串行通信单元中登录的收发时序(协议数据)。	2–1154
	SerialSend	串行通信单元 串行端口输出	无协议从串行通信单元的串行端口发送数据。	2–1166
	SerialRcv	串行通信单元 串行端口输入	从串行通信单元的串行端口读取无协议的接收数据。读取后，接收缓存全部清除。	2–1175
	SerialRcvNo Clear	串行通信单元 串行端口输入不清除接收缓存	从串行通信单元的串行端口读取无协议的接收数据。读取后，接收缓存不会全部清除。	2–1175
	SendCmd	指令发送	通过串行网关功能向串行通信单元发送指令。此外，向DeviceNet单元和CompoNet主站单元发送Explicit指令。	2–1190
	NX_SerialSend	无协议数据的发送	无协议从NX系列通信接口单元及扩展板的串行端口发送数据。	2–1202
	NX_SerialRcv	无协议数据的接收	无协议从NX系列通信接口单元及扩展板的串行端口读取数据。	2–1215

种类	指令	名称	功能概要	页码
串行通信指令	NX_ModbusRtu Cmd	Modbus RTU通用指令发送	使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送通用指令。	2-1229
	NX_ModbusRtu Read	Modbus RTU Read指令发送	使用Modbus-RTU协议，从NX系列通信接口单元及扩展板向Modbus-RTU从站发送读取指令。	2-1239
	NX_ModbusRtu Write	Modbus RTU Write指令发送	使用Modbus-RTU协议，从NX系列通信接口单元及扩展板向Modbus-RTU从站发送写入指令。	2-1249
	NX_SerialSigCtl	串行控制信号的ON/OFF切换	使NX系列通信接口单元及扩展板串行端口的ER信号或RS信号ON或OFF。	2-1259
	NX_SerialSigRead	串行控制信号的读取	读取扩展板串行端口的CS信号或DR信号。	2-1267
	NX_SerialStatus Read	串行端口状态的读取	读取扩展板串行端口的状态。	2-1271
	NX_SerialBufClear	缓存清除	清除收发缓存。	2-1275
	NX_SerialStartMon	串行线路监控的开始	开始NX系列通信接口单元的串行线路监控。	2-1285
	NX_SerialStopMon	串行线路监控的停止	停止NX系列通信接口单元的串行线路监控。	2-1289
SD存储卡指令	FileWriteVar	变量文件写入	以二进制格式将1个变量值写入SD存储卡内的指定文件。	2-1294
	FileReadVar	变量文件读取	以二进制格式读取SD存储卡内指定文件的值，并写入变量。	2-1299
	FileOpen	文件打开	打开SD存储卡内的指定文件。	2-1304
	FileClose	文件关闭	关闭SD存储卡内的指定文件。	2-1308
	FileSeek	文件查找	为SD存储卡内的指定文件设定文件位置指示器。	2-1311
	FileRead	文件读取	读取SD存储卡内指定文件的数据。	2-1314
	FileWrite	文件写入	将数据写入SD存储卡内的指定文件。	2-1322
	FileGets	字符串读取	从SD存储卡内的指定文件读取1行字符串。	2-1330
	FilePuts	字符串写入	将字符串写入SD存储卡内的指定文件。	2-1338
	FileCopy	文件复制	复制SD存储卡内的指定文件。	2-1346
	FileRemove	文件删除	删除SD存储卡内的指定文件。	2-1355
	FileRename	文件名变更	变更SD存储卡内指定文件和目录的名称。	2-1360
	DirCreate	目录创建	在SD存储卡内创建指定名称的目录。	2-1366
	DirRemove	目录删除	删除SD存储卡内的指定目录。	2-1369
	BackupTo MemoryCard	SD存储卡备份	执行SD存储卡备份。	2-1373
时间戳指令	NX_DOut TimeStamp	时间戳数字输出写入	在时间戳方式对应的数字输出单元的输出触点写入值。	2-1388
	NX_AryDOut TimeStamp	时间戳数字输出数组写入	从时间戳方式对应的数字输出单元输出脉冲。	2-1393
其它指令	ReadNbit_**	N位读取组	读取位串内的多位。	2-1402
	WriteNbit_**	N位写入组	在位串内写入多位。	2-1404
	ChkRange	范围型变量检查	判定变量值是否在范围型指定的有效范围内。	2-1406
	GetMyTask Status	我的任务状态读取	读取我的任务状态。	2-1408
	GetMyTask Interval	我的任务设定周期读取	读取我的任务的任务周期。	2-1411
	Task_IsActive	任务执行中判定	判定指定任务是否正在执行。	2-1413
	Lock	任务间排他锁	开启任务间排他锁。与其它任务锁定编号相同的区间无法执行。	2-1415
	Unlock	任务间排他锁解除	解除任务间排他锁。	2-1415
	ActEventTask	事件任务启动	启动事件任务。	2-1420
	Get**Clk	时钟脉冲获取组	输出指定周期的时钟脉冲。	2-1426

种类	指令	名称	功能概要	页码
其它指令	Get**Cnt	自激加法计数器获取组	获取指定周期的自激计数器的值。	2–1428

- 运动控制指令的规格请参阅 “NJ/NX系列 指令基准手册 运动篇(SBCE–364)”。
- 模拟指令的规格请参阅 “Sysmac Studio Version1操作手册(SBCA–362)”。

2

xxxxx

各指令的说明

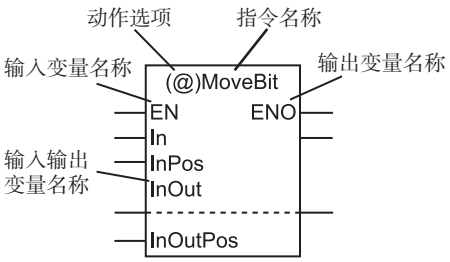
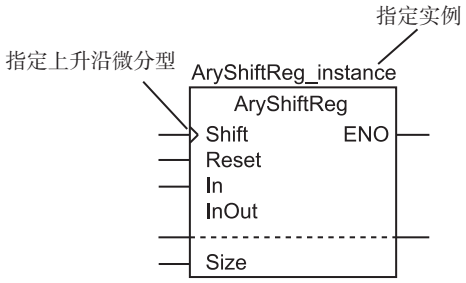
本章介绍NJ/NX系列可使用的指令规格。

本章说明

下面介绍本章(各指令的说明)的记述项目和说明。

记述项目

记述项目如下所示。

项目	内容
指令	记述指令名称。　例：MoveBit
名称	记述指令的中文名称。　例：位传送
FB/FUN	记述指令为功能块(FB)型或函数(FUN)型。 FB型的指令仅可通过程序或FB调用。 FUN型的指令通过程序、FB、FUN均可调用。
图形表现	记述该指令在梯形图程序内表达时的图。 ●FUN型示例  ●FB型示例  其中，“动作选项”、“指定上升沿微分型”、“指定实例”的含义如下所示。 动作选项　：在FUN型指令名称前标有的(@)。 带动作选项的指令可在指令名称前加上@，设为上升沿微分型。也可在指令名称前加上%，设为输入下降沿微分型。 输入上升沿微分型是指，执行条件的输入变量(EN)的值在上次任务周期中FALSE，当前任务周期中TRUE时执行的指令。 指定上升沿微分型：是指输入变量的插针上标有的箭头(–>)。带有该指定的指令以输入上升沿微分型进行动作。 指定实例　：是指FB型的指令上标有的“XX_instance”。标有指定实例的指令需附加任意的实例名称。
ST表现	记述该指令在ST程序内表达时的描述。 在ST程序内写指令时，有以下2种方法。 1. 明确表示变量(输入变量、输出变量、输入输出变量)名称和参数名称的对应的方法。 例：MoveBit(In:=abc, InPos:=def, InOut:=ghi, InOutPos:=jkl); 2. 省略变量(输入变量、输出变量、输入输出变量)名称，仅记述参数名称的方法。 例：MoveBit(In, InPos, InOut, InOutPos); 本章以2.)的方法进行记述。 标有XX_instance(变量名称)的指令需附加任意的实例名称。 例：TON_instance (In, PT, Q, ET);

项目	内容
变量	• 变量名称 记述输入变量、输出变量、输入输出变量的名称。 例：In1 多个指令通用的变量在各指令的说明页中记述。通用变量有以下8个。具体规格将于下文阐述。 (EN、ENO、Execute、Done、Busy、Error、ErrorID、ErrorIDEx) • 名称 记述变量的中文名称。 例：加法计数器 • 输入/输出 记述输入变量、输出变量、输入输出变量中的任意一个。 • 内容 记述该变量的含义和限制。 • 有效范围 记述获取该变量值的范围。 标有“遵从数据类型”时，应遵从该变量的数据类型的有效范围。各数据类型的有效范围将于下文阐述。 • 单位 记述该变量的单位。标有 “–”时，无单位。 例：字节 • 初始值 不将参数代入变量即执行指令时，将自动为该变量设定值。 标有 “–”时，含义如下所示。 输入变量 ：代入该输入变量的数据类型的默认初始值。各数据类型的默认初始值将于下文阐述。 输入变量为结构体时，通过功能说明中的结构体规格表记述初始值。 输出变量 ：未设定初始值。 输入输出变量：未设定初始值。 • 数据类型 记述该变量的数据类型。 使用枚举体、数组、结构体、联合体时，按数据类型记述。
功能	记述该指令的功能。 本文中的变量名称用 “” 标注。 例：“In1” 此外，数组名称用[]标注。 例：InOut[]
相关的系统定义变量	记述与该指令相关的系统定义变量的名称和概要。 系统定义变量的详情请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”。
相关的准用户定义变量	记述与该指令相关的准用户定义变量的名称和概要。 准用户定义变量的详情请参阅相应页标注的参阅手册。
参考	记述该指令功能的补充信息。 记述相关指令和该指令的便捷使用方法等。
使用注意事项	记述使用该指令时的注意事项。 何种场合下会发生异常也在此处记述。
示例程序	记述含有该指令的短应用程序的示例。 处理内容相同时，同时记述梯形图程序和ST程序。

通用变量

对多个指令通用的变量(EN、ENO、Execute、Done、Busy、Error、ErrorID、ErrorIDEx)的规格进行说明。这些通用变量未在各指令的变量表中记述。各指令是否含有这些通用变量请通过图形表达和ST表达栏进行确认。

EN

EN为表示FUN型指令的执行条件的输入变量。
在梯形图程序中使用FUN型指令时，请将执行条件输入(连接)至EN。

	名称	输入/输出	内容	数据类型	有效范围	初始值
EN	启用(执行条件)	输入	TRUE：执行指令(*) FALSE：不执行指令	BOOL	TRUE,FALSE	TRUE

* 动作选项中指定为输入上升沿微分型(@)时，以EN的值从FALSE变为TRUE为执行条件。指定为输入下降沿微分型(%)时，以EN的值从TRUE变为FALSE为执行条件。

- FB型的指令中无EN。
- 通过ST程序读取FUN型指令时，请省略EN的记述。指令的执行条件由ST程序的动作时序决定，因此无需EN。

ENO

ENO为梯形图程序中用于将执行条件传送至后段指令的输出变量。
通常在指令正常结束时，ENO的值为TRUE。接收后，开始执行后段指令。

	名称	输入/输出	内容	数据类型	有效范围	初始值
ENO	启用输出	输出	TRUE：正常结束(*) FALSE：异常结束、执行中、或执行条件不成立	BOOL	TRUE,FALSE	–

* 仅限于执行条件成立时。正常结束后或执行条件不成立时，ENO的值为FALSE。

- ENO在FUN型指令和FB型指令中均存在。但部分指令中无ENO。
- ST程序中请省略ENO的记述。后段指令的执行条件由ST程序的动作时序决定，因此无需ENO。

Execute、Done、Busy

Execute存在于部分FB型指令中，为表示执行条件的输入变量。
Execute的值从FALSE变为TRUE时，执行指令。指令一旦执行，即使Execute的值为FALSE或指令执行时间超出任务周期，仍将继续处理直至最后。

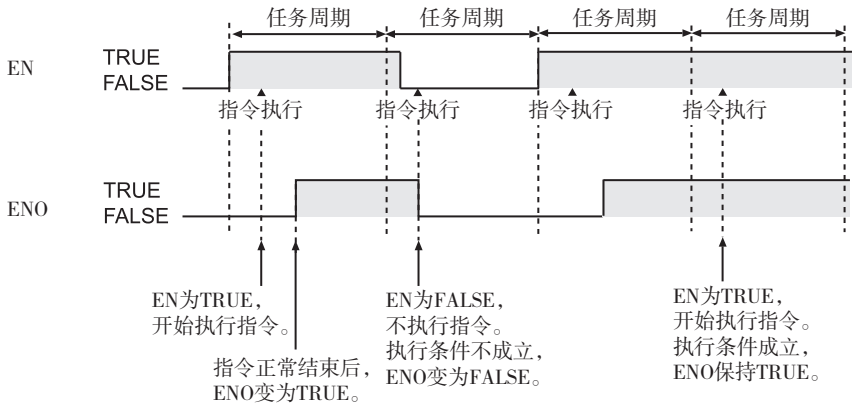
Done存在于部分FB型指令中，表示指令正常结束的输出变量。
Busy存在于部分FB型指令中，为表示指令正在执行的输出变量。

	名称	输入/输出	内容	数据类型	有效范围	初始值
Execute	启动	输入	TRUE：执行指令(*1) FALSE：不执行指令(*2)	BOOL	TRUE,FALSE	FALSE
Done	完成	输出	TRUE：正常结束(*3)(*4) FALSE：异常结束、执行中或执行条件不成立	BOOL	TRUE,FALSE	–
Busy	执行中		TRUE：执行中 FALSE：未执行			

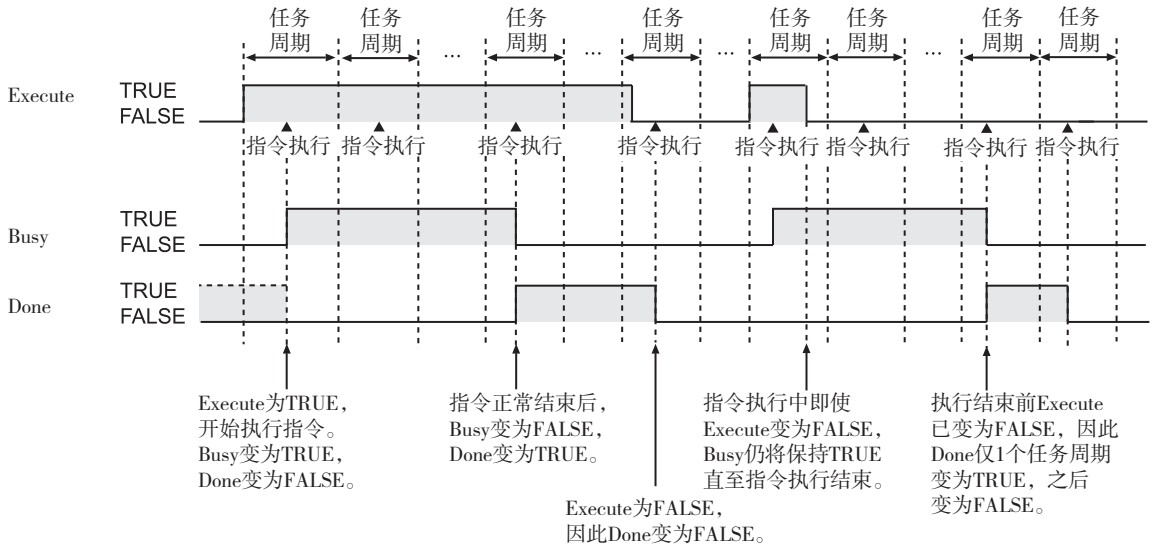
*1 开始运行时Execute的值已为TRUE时，不执行指令。执行指令时，请将Execute的值暂时设为FALSE。
*2 执行指令中Execute的值即使为FALSE，仍将继续处理直至最后。
*3 正常结束后或执行条件不成立时，Done的值为FALSE。
*4 若正常结束时执行条件未成立，则Done的值在1个任务周期中保持TRUE，之后变为FALSE。

带EN和ENO的指令即在1个任务周期中处理完成的指令以及带Execute和Busy的指令即跨越多个任务周期执行处理的指令的时序图如下所示。

● 在1个任务周期中处理完成的指令



● 跨越多个任务周期执行处理的指令



Error、ErrorID、ErrorIDEx

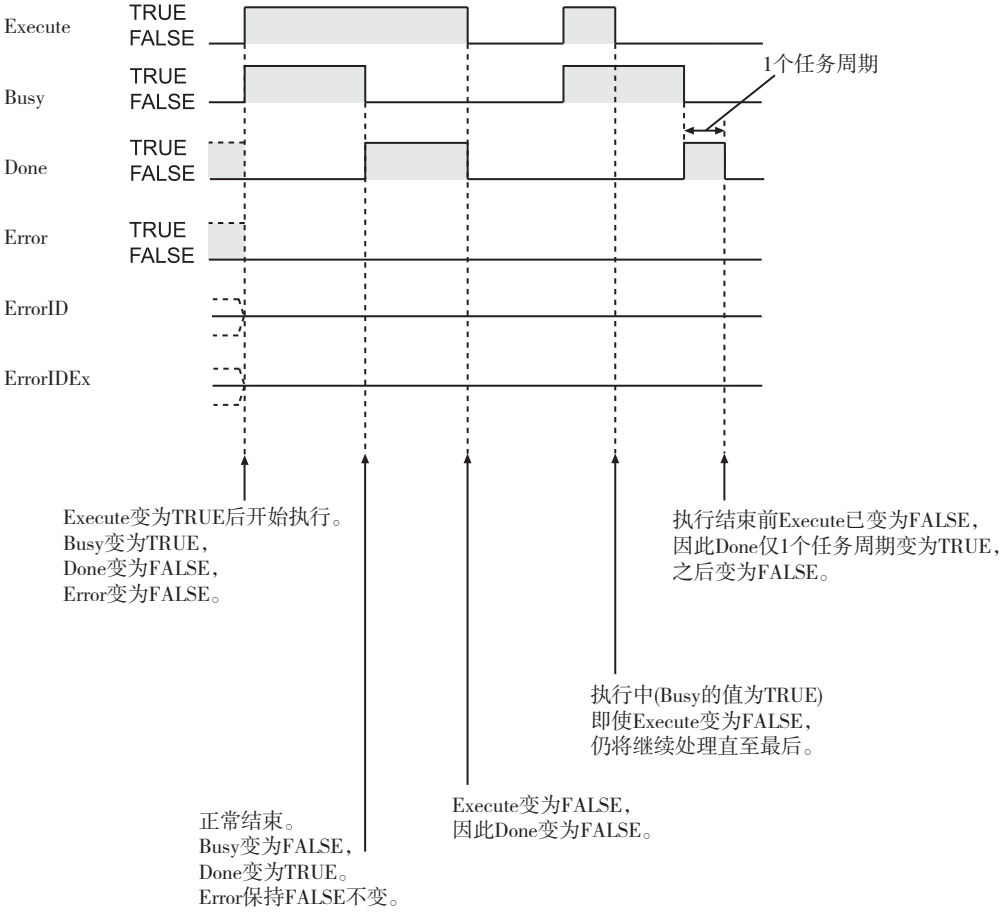
Error、ErrorID、ErrorIDEx存在于部分FB型指令中，表示指令异常结束的输出变量。

	名称	输入/输出	内容	数据类型	有效范围	初始值
Error	错误	输出	TRUE：异常结束(*1)(*2) FALSE：正常结束、执行中或执行条件不成立	BOOL	TRUE,FALSE	–
ErrorID	错误代码		异常结束时的异常ID 正常结束时为WORD#16#0	WORD	因指令而异	
ErrorIDEx	扩展错误代码		异常结束时的扩展异常ID 正常结束时为DWORD#16#0	DWORD		

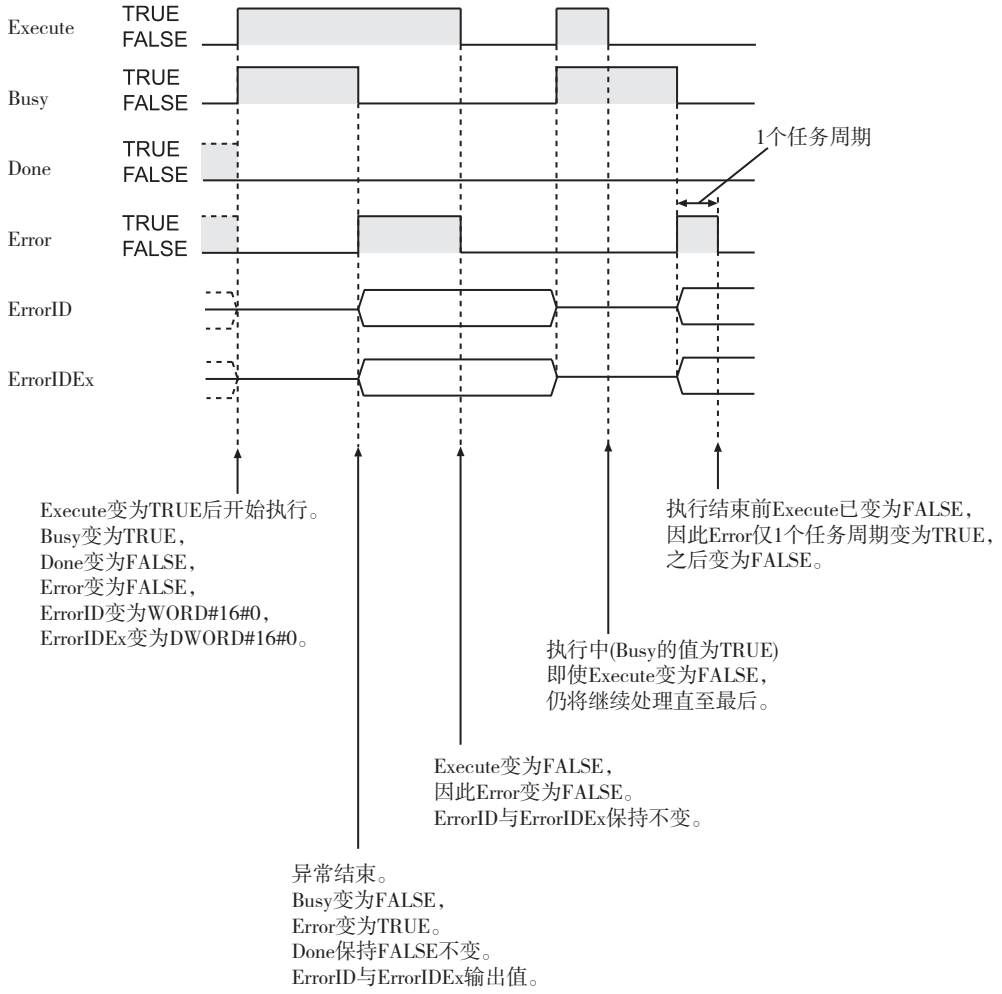
*1 异常结束后或执行条件不成立时，Error的值为FALSE。
*2 若异常结束时执行条件未成立，则Error的值在1个任务周期中保持TRUE，之后变为FALSE。

Execute、Done、Busy、Error、ErrorID、ErrorIDEx的时序图如下所示。

● 正常结束时


Execute
TRUE
FALSE
Busy
TRUE
FALSE
Done
TRUE
FALSE
Error
TRUE
FALSE
ErrorID
ErrorIDEx
1个任务周期
Execute变为TRUE后开始执行。
Busy变为TRUE，
Done变为FALSE，
Error变为FALSE。
执行结束前Execute已变为FALSE，
因此Done仅1个任务周期变为TRUE，
之后变为FALSE。
执行中(Busy的值为TRUE)
即使Execute变为FALSE，
仍将继续处理直至最后。
正常结束。
Busy变为FALSE，
Done变为TRUE。
Error保持FALSE不变。
Execute变为FALSE，
因此Done变为FALSE。

● 异常结束时



变量的有效范围和默认初始值

变量的有效范围是指获取该变量值的范围。此外，变量的默认初始值指不将参数代入输入变量即执行指令时，代入该输入变量值。

这些均由各数据类型决定。指令无特殊指定时，变量的有效范围和初始值遵从该值。

本章的变量表中，有效范围一栏中已标注 “遵从数据类型”，输入变量的初始值一栏中已标注 “–”，与其对应。

各数据类型的有效范围和初始值如下所示。

分类	数据类型	有效范围	初始值
布尔	BOOL	FALSE,TRUE	FALSE
位串	BYTE	BYTE#16#00 ~ FF	BYTE#16#00
	WORD	WORD#16#0000 ~ FFFF	WORD#16#0000
	DWORD	DWORD#16#00000000 ~ FFFFFFFF	DWORD#16#00000000
	LWORD	LWORD#16#0000000000000000 ~ FFFFFFFFFFFFFFFF	LWORD#16#0000000000000000
整数	USINT	USINT#0 ~ +255	USINT#0
	UINT	UINT#0 ~ +65535	UINT#0
	UDINT	UDINT#0 ~ +4294967295	UDINT#0
	ULINT	ULINT#0 ~ +18446744073709551615	ULINT#0
	SINT	SINT#-128 ~ +127	SINT#0
	INT	INT#-32768 ~ +32767	INT#0
	DINT	DINT#-2147483648 ~ +2147483647	DINT#0
	LINT	LINT#-9223372036854775808 ~ +9223372036854775807	LINT#0
实数	REAL	REAL#-3.402823e+38 ~ -1.175495e-38, 0, +1.175495e-38 ~ +3.402823e+38, +∞/-∞	REAL#0
	LREAL	LREAL#-1.79769313486231e+308 ~ -2.22507385850721e-308, 0, +2.22507385850721e-308 ~ +1.79769313486231e+308, +∞/-∞	LREAL#0
时刻、持续时间、日期、字符串	TIME	T#-9223372036854.775808ms (T#-106751d_23h_47m_16s_854.775808ms) ~ T#9223372036854.775807ms (T#+106751d_23h_47m_16s_854.775807ms)	T#0s
	DATE	D#1970-01-01 ~ D#2106-02-06 (1970年1月1日 ~ 2106年2月6日)	D#1970-01-01
	TOD	TOD#00:00:00.000000000 ~ TOD#23:59:59.999999999 (0时0分0.000000000秒 ~ 23时59分59.999999999秒)	TOD#00:00:00.000000000
	DT	DT#1970-01-01-00:00:00.000000000 ~ DT#2106-02-06-23:59:59.999999999 (1970年1月1日0时0分0.000000000秒 ~ 2106年2月6日23时59分59.999999999秒)	DT#1970-01-01-00:00:00.000000000
	STRING	文字代码：UTF-8 0 ~ 1986字节 (半角英数字为0 ~ 1985字符+结尾NULL字符)	''

派生数据类型(枚举体、结构体、联合体)

使用派生数据类型(枚举体、结构体、联合体)的变量在变量数据类型表中进行记述。下面对记述方法进行说明。

枚举体

对于枚举体变量，在表内记述了列举的数据类型。
示例如下所示。此时，变量“Out”的数据类型为枚举体_eDAYOFWEEK。枚举元素的详情在功能说明中进行阐述。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																	○		○	
Out	枚举体_eDAYOFWEEK　枚举元素参阅功能说明																			

结构体、联合体

对于构造体变量、联合体变量，在表内记述了结构体、联合体的数据类型。
示例如下所示。此时，变量 “In1”的数据类型为结构体 _sPORT。结构体、联合体的结构要素详情在功能说明中进行记述。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1	结构体_sPORT　详情参阅功能说明																			

可将整个结构体、结构体的1个结构要素、整个联合体、联合体的1个结构要素作为参数指定的变量在表内进行说明。
下列场合时，变量 “In1”中除基本数据类型以外，还可指定整个结构体、结构体的1个结构要素、整个联合体、联合体的1个结构要素。指定整个结构体和联合体时，请仅将结构体或联合体名称作为参数。指定结构体的1个结构要素、联合体的1个结构要素时，请将结构要素名称作为参数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定整个结构体、结构体的1个结构要素、整个联合体、联合体的1个结构要素																			

数组指定

数组指定变量中，在变量名称后加上[]，并标注“数组”。有该记述时，请将带下标的数组元素作为参数。示例如下所示。此时，“In1[]”表示BYTE型的数组。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[]数组		○																		

以下表为例，对以结构体和联合体为元素的数组进行说明。有该记述时，也请将带下标的数组元素作为参数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[]数组	指定以结构体为元素的数组、以联合体为元素的数组																			

可将整个数组、数组的1个元素作为参数指定的变量在表内进行说明。
下列场合时，变量 “In1”中除基本数据类型以外，还可指定整个数组、数组的1个元素。指定整个数组时，请仅将数组名称作为参数。指定数组的1个元素时，请将标注下标的元素编号作为参数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定整个数组、数组的1个元素																			

其它

所有指令通用的异常检测

使用注意事项中，记述各指令下可能发生的异常种类。但下列异常的检测为所有指令通用，因此不在使用注意事项中记述，敬请注意。

- 读取的超出数组指定变量的数组区域的元素时。
  (例)　对于数组指定变量a[0..3]，将a[4]设定为输入变量。
- 向作为输入变量、输出变量、输入输出变量定义数组指定变量的指令，传输非数组的参数时。
- 将长于定义字节数的字符串代入STRING型变量时。
- 将未以NULL字符结尾的字符串代入STRING型变量时。
- 将整数型变量除以0时。

所有指令通用的注意事项

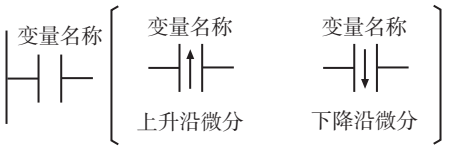
指令中，部分处理量可根据连接的参数变化。若处理量过大，则指令执行时间可能会变长，超出任务周期。届时，会超出任务周期，因此请适当调整处理量。

梯形图指令

指令	名称	页码
LD/LDN	加载/加载・非	2–16
AND/ANDN	与/与・非	2–18
OR/ORN	或/或・非	2–21
Out/OutNot	输出/输出・非	2–24

LD/LDN

LD ：读取BOOL型变量值。
LDN：读取BOOL型变量的取反值。

指令	名称	FB/FUN	图形表现	ST表现
LD	加载	–		无
LDN	加载・非	–		无

变量

无

功能

- LD

读取变量名称指定的BOOL型变量值后，连接(输出)至后段。
指定变量值为TRUE时输出TRUE，为FALSE时输出FALSE。
用于从母线开始的首个a触点或电路模块的首个a触点。

- LDN

读取变量名称指定的BOOL型变量的取反值后，连接(输出)至后段。
指定变量值为TRUE时输出FALSE，为FALSE时输出TRUE。
用于从母线开始的首个b触点或电路模块的首个b触点。

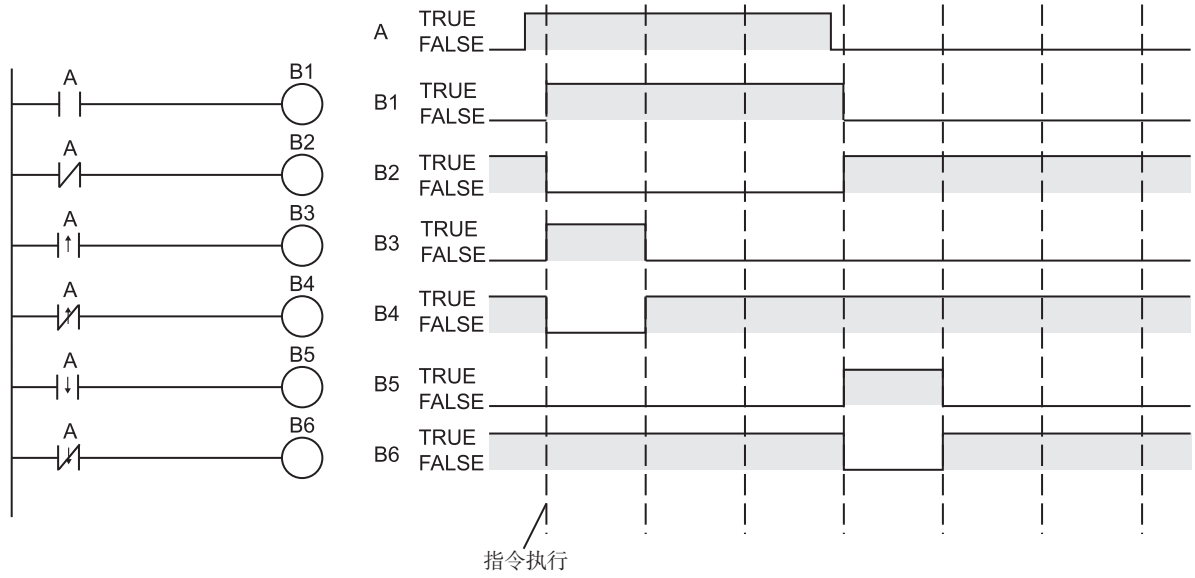
未指定上升沿微分、下降沿微分时的动作如下所示。

指令	变量值	输出
LD	TRUE	TRUE
	FALSE	FALSE
LDN	TRUE	FALSE
	FALSE	TRUE

如下所示，指定上升沿微分、下降沿微分时，根据上次执行时的变量值和当前的变量值动作。

指令	微分指定	上次执行时的变量值与当前的变量值	输出
LD	上升沿微分	上次执行时FALSE → 当前TRUE	TRUE
		上述以外型号	FALSE
	下降沿微分	上次执行时TRUE → 当前FALSE	TRUE
		上述以外型号	FALSE
LDN	上升沿微分	上次执行时FALSE → 当前TRUE	FALSE
		上述以外型号	TRUE
	下降沿微分	上次执行时TRUE → 当前FALSE	FALSE
		上述以外型号	TRUE

描述示例和时序图如下所示。

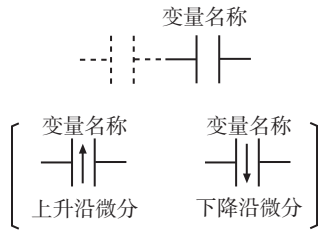
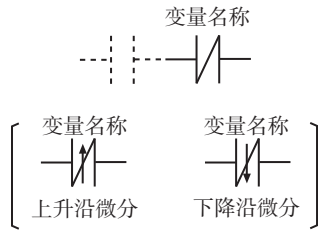


使用注意事项

- 以下情况时会发生异常，并保持上次执行时的输出值。
  - 为变量值指定数组元素，且该元素不存在时。
    (例)　定义BOOL型数组a [0..5]，指定变量a [10]后，执行本指令时
- 本指令无法用于电路的末尾段。使用时，Sysmac Studio会发生异常，无法传送至控制器。

AND/ANDN

AND ：获取BOOL型变量值与输入条件的逻辑积。
ANDN：获取BOOL型变量的取反值与输入条件的逻辑积。

指令	名称	FB/FUN	图形表现	ST表现
AND	与	–		result：= vBool1 AND vBool2; result：= vBool1 & vBool2;
ANDN	与・非	–		result：= vBool1 AND NOT vBool2;

变量

无

功能

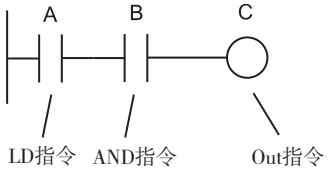
- AND

将变量名称指定的BOOL型变量值与输入条件的逻辑积连接(输出)至后段。
用于与前段电路串联的a触点。

- ANDN

将变量名称指定的BOOL型变量的取反值与输入条件的逻辑积连接(输出)至后段。
用于与前段电路串联的b触点。

AND指令的描述示例如下所示。将变量A与变量B的逻辑积输出至变量C。



将变量A与变量B的逻辑积输出至变量C。

未指定上升沿微分、下降沿微分时的动作如下所示。

指令	变量值与输入条件的组合	输出
AND	变量值：TRUE 输入条件：TRUE	TRUE
	上述以外型号	FALSE
ANDN	变量值：FALSE 输入条件：TRUE	TRUE
	上述以外型号	FALSE

如下所示，指定上升沿微分、下降沿微分时，根据上次执行时的变量值、当前的变量值及输入条件动作。

指令	微分指定	上次执行时的变量值、当前的变量值 及输入条件的组合	输出
AND	上升沿微分	变量值：上次执行时FALSE → 当前TRUE 输入条件：TRUE	TRUE
		上述以外型号	FALSE
	下降沿微分	变量值：上次执行时TRUE → 当前FALSE 输入条件：TRUE	TRUE
		上述以外型号	FALSE
ANDN	上升沿微分	变量值：上次执行时FALSE → 当前TRUE 输入条件：TRUE	FALSE
		变量值：无关 输入条件：FALSE	
		上述以外型号	TRUE
	下降沿微分	变量值：上次执行时TRUE → 当前FALSE 输入条件：TRUE	FALSE
		变量值：无关 输入条件：FALSE	
		上述以外型号	TRUE

使用注意事项

- 以下情况时会发生异常，并保持上次执行时的输出值。
  - 为变量值指定数组元素，且该元素不存在时。
    (例)　定义BOOL型数组a[0..5]，指定变量a[10]后，执行本指令时
- 本指令无法用于电路的末尾段。使用时，Sysmac Studio会发生异常，无法传送至控制器。
- 本指令无法直接连接至母线。

OR/ORN

OR ：获取BOOL型变量值与输入条件的逻辑和。
ORN：获取BOOL型变量的相反值与输入条件的逻辑和。

指令	名称	FB/FUN	图形表现	ST表现
OR	或	–		result：= vBool1 OR vBool2;
ORN	或・非	–		result：= vBool1 OR NOT vBool2;

变量

无

功能说明

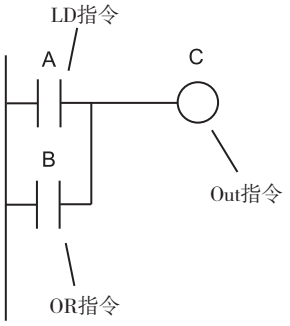
- OR

将变量名称指定的BOOL型变量值与输入条件的逻辑和连接(输出)至后段。
用于与前段电路并联的a触点。构成与母线连接或电路模块开头的LD/LDN指令至本指令前为止的电路间OR(逻辑和)的a触点。

- ORN

将变量名称指定的BOOL型变量的取反值与输入条件的逻辑和连接(输出)至后段。
用于与前段电路并联的b触点。构成与母线连接或电路模块开头的LD/LDN指令至本指令前为止的电路间OR(逻辑和)的b触点。

OR指令的描述示例如下所示。将变量A与变量B的逻辑和输出至变量C。



将变量A与变量B的逻辑和输出至变量C。

未指定上升沿微分、下降沿微分时的动作如下所示。

指令	变量值与输入条件的组合	输出
OR	变量值：FALSE 输入条件：FALSE	FALSE
	上述以外型号	TRUE
ORN	变量值：TRUE 输入条件：FALSE	FALSE
	上述以外型号	TRUE

如下所示，指定上升沿微分、下降沿微分时，根据上次执行时的变量值、当前的变量值及输入条件动作。

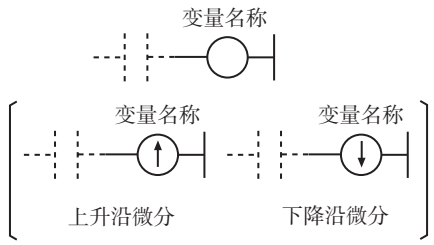
指令	微分指定	上次执行时的变量值、当前的变量值 及输入条件的组合	输出
OR	上升沿微分	变量值：上次执行时FALSE → 当前TRUE 输入条件：无关	TRUE
		变量值：无关 输入条件：TRUE	
		上述以外型号	FALSE
	下降沿微分	变量值：上次执行时TRUE → 当前FALSE 输入条件：无关	TRUE
		变量值：无关 输入条件：TRUE	
		上述以外型号	FALSE
ORN	上升沿微分	变量值：上次执行时FALSE → 当前TRUE 输入条件：FALSE	FALSE
		上述以外型号	TRUE
	下降沿微分	变量值：上次执行时TRUE → 当前FALSE 输入条件：FALSE	FALSE
		上述以外型号	TRUE

使用注意事项

- 以下情况时会发生异常，并保持上次执行本指令时的输出值。
  - 为变量值指定数组元素，且该元素不存在时。
    (例)　定义BOOL型数组a[0..5]，指定变量a[10]后，执行本指令时
- 本指令无法用于电路的末尾段。使用时，Sysmac Studio会发生异常，无法传送至控制器。

Out/OutNot

Out　　：将前段的逻辑运算处理的结果输出至BOOL型变量。
OutNot：将前段的逻辑运算处理的结果的取反值输出至BOOL型变量。

指令	名称	FB/FUN	图形表现	ST表现
Out	输出	–		变量名称：= (前段的逻辑运算公式);
OutNot	输出・非	–		变量名称：= NOT (前段的逻辑运算公式);

变量

无

功能

● Out

将前段的逻辑运算处理的结果输出为变量名称指定的BOOL型变量。

未指定上升沿微分、下降沿微分时的动作如下所示。

前段逻辑运算处理的结果	输出
TRUE	TRUE
FALSE	FALSE

Out指令中可指定上升沿微分、下降沿微分的动作。此时，前段逻辑运算处理的结果通过目前与上次执行时的变化情况确定输出值。如下所示，根据前段的逻辑运算处理的结果动作。

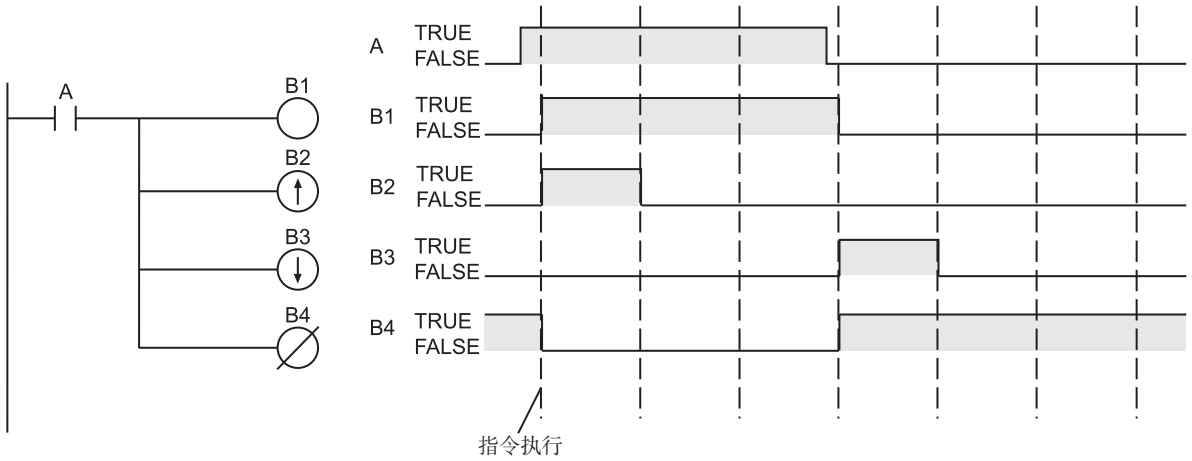
微分指定	上次执行时的前段逻辑运算处理的结果与当前的前段逻辑运算处理的结果	输出
上升沿微分	上次执行时FALSE　→　当前TRUE	TRUE
	上述以外型号	FALSE
下降沿微分	上次执行时TRUE　→　当前FALSE	TRUE
	上述以外型号	FALSE

● OutNot

将前段的逻辑运算处理的结果的取反值输出为变量名称指定的BOOL型变量。

前段逻辑运算处理的结果	输出
TRUE	FALSE
FALSE	TRUE

描述示例和时序图如下所示。



参考

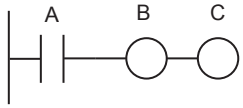
Set/Reset指令与Out/OutNot指令的区别

- Set/Reset指令在输入TRUE时动作，输入FALSE时不动作。即输入FALSE时，输出无变化。
- Out/OutNot指令无论前段的逻辑运算处理的结果是TRUE还是FALSE，都将分别根据情况进行输出。

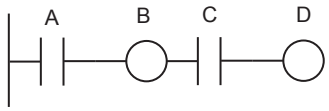
使用注意事项

- 以下情况时会发生异常，不执行输出。
  - 为变量值指定数组元素，且该元素不存在时。

  (例)　定义BOOL型数组a[0..5]，指定变量a[10]后，执行本指令时
- 可进行以下连接。
  - 在Out指令的后段连接其它Out指令。

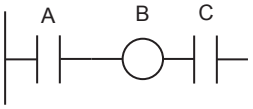


  - 在Out指令的后段连接LD指令和Out指令。

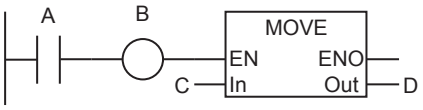


- 无法进行以下连接。

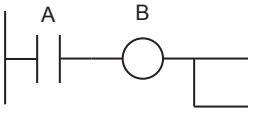
- 在Out指令的后段仅连接LD指令。

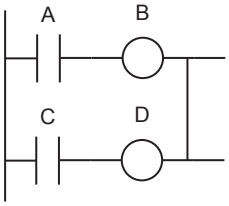


- 在Out指令的后段连接函数、功能块。



- 在Out指令的后段分支或合并。





ST语法指令

指令	名称	页码
IF	选择	2–28
CASE	多分支	2–32
WHILE	重复	2–36
REPEAT	重复	2–38
RETURN	复位	2–67
FOR	重复	2–84
EXIT	循环中断	2–90

IF

根据指定的条件表达式的评估结果，从两者中选择要执行的语句。

指令	名称	FB/FUN	图形表现	ST表现

IF	选择	–	无	IF 条件表达式 THEN
				语句;
				ELSIF 条件表达式 THEN
				语句;
				ELSE
				语句;
				END_IF;

变量

无

功能

根据指定条件表达式的评估结果，从两者中选择要执行的语句。条件表达式中记述作为评估结果的带TRUE或FALSE值的下列内容。

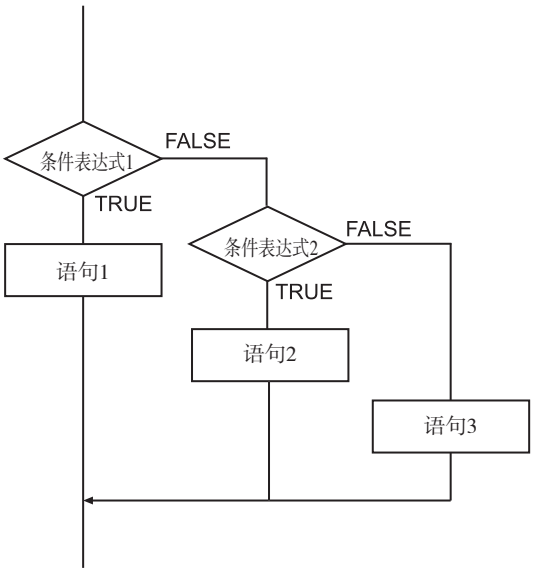
条件表达式可记述的内容	记述示例	评估结果
逻辑表达式	a>3	变量a的值大于3时为TRUE，否则为FALSE
	a=b	变量a、b的值相等时为TRUE，否则为FALSE
BOOL型变量	abc	abc的值为TRUE时TRUE，为FALSE时FALSE
BOOL型常数	TRUE	TRUE
带有BOOL型返回值的函数	FUN名称	该函数的返回值为TRUE时TRUE，为FALSE时FALSE

逻辑表达式中可使用以下运算符。

运算符	含义	记述示例	评估结果
=	一致	a=b	变量a、b的值相等时为TRUE，否则为FALSE
<>	不一致	a<>b	变量a、b的值不相等时为TRUE，否则为FALSE
<	比较	a<b	变量a的值小于b时为TRUE，否则为FALSE
< =		a<=b	变量a的值小于等于b时为TRUE，否则为FALSE
>		a>b	变量a的值大于b时为TRUE，否则为FALSE
>=		a>=b	变量a的值大于等于b时为TRUE，否则为FALSE
AND,&	逻辑积	a AND b a & b	BOOL型变量a、b的逻辑积
OR	逻辑和	a OR b	BOOL型变量a、b的逻辑和
XOR	异或	a XOR b	BOOL型变量a、b的异或
NOT	非	NOT a	非BOOL型变量a

如下图所示，以下记述时的处理流程根据条件表达式1、条件表达式2的评估结果而定。语句n中可记述多个语句。

```
IF  条件表达式1  THEN
  语句1;
ELSIF 条件表达式2  THEN
  语句2;
ELSE
  语句3;
END_IF;
```



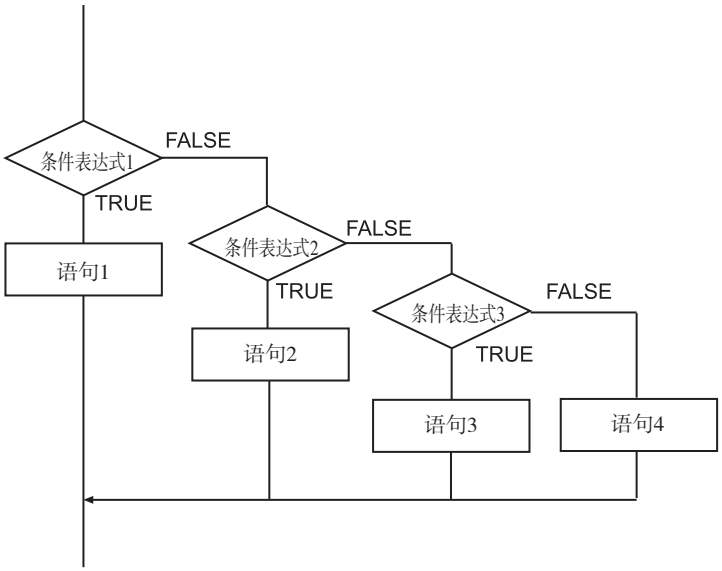
参考

- 本指令可用于分层结构。如下所示，条件表达式1的评估结果为TRUE且条件表达式11的评估结果也为TRUE时执行语句11。

```
IF  条件表达式1  THEN
    IF  条件表达式11  THEN
      语句11;
    ELSIF 条件表达式12  THEN
      语句12;
    ELSE
      语句13;
    END_IF;
ELSIF 条件表达式2  THEN
    语句2;
ELSE
    语句3;
END_IF;
```

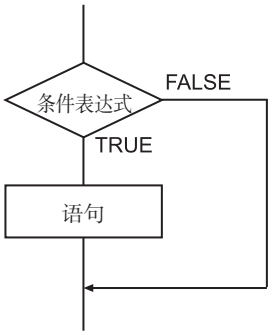
ELSIF可记述多个。参考以下记述，处理流程如下图所示。

```
IF　条件表达式1　THEN
  语句1;
ELSIF 条件表达式2　THEN
  语句2;
ELSIF 条件表达式3　THEN
  语句3;
ELSE
  语句4;
END_IF;
```



- 仅有1个条件表达式时，无需ELSIF。此外，若不符合任何条件表达式，则不进行任何处理。此时，无需ELSE。参考以下记述，处理流程如下图所示。

```
IF　条件表达式　THEN
  语句;
END_IF;
```



- 语句内容无特殊限制。除IF指令以外可记述任何内容，例如功能块的调用和FOR指令等。

使用注意事项

- IF和END_IF不可省略。此外，该两者请务必成对使用。
- 含IF指令、CASE指令、FOR指令、WHILE指令、REPEAT指令在内，最多15层。

示例程序

变量abc的值小于INT#0时，在变量def中代入INT#0。abc的值为INT#0时，在变量def中代入INT#1，在变量ghi中代入INT#2。abc的值非上述情况时，在变量def中代入INT#3。

名称	数据类型	初始值
abc	INT	0
def	INT	0
ghi	INT	0

```
IF (abc < INT#0) THEN
    def：= INT#0;
ELSIF (abc = INT#0) THEN
    def：= INT#1;
    ghi：= INT#2;
ELSE
    def：= INT#3;
END_IF;
```

CASE

根据指定的整数表达式的值，从多个语句中选择要执行的语句。

指令	名称	FB/FUN	图形表现	ST表现
CASE	多分支	–	无	CASE 整数表达式 OF 选择值： 语句； 选择值： 语句； ： ELSE 语句； END_CASE；

变量

无

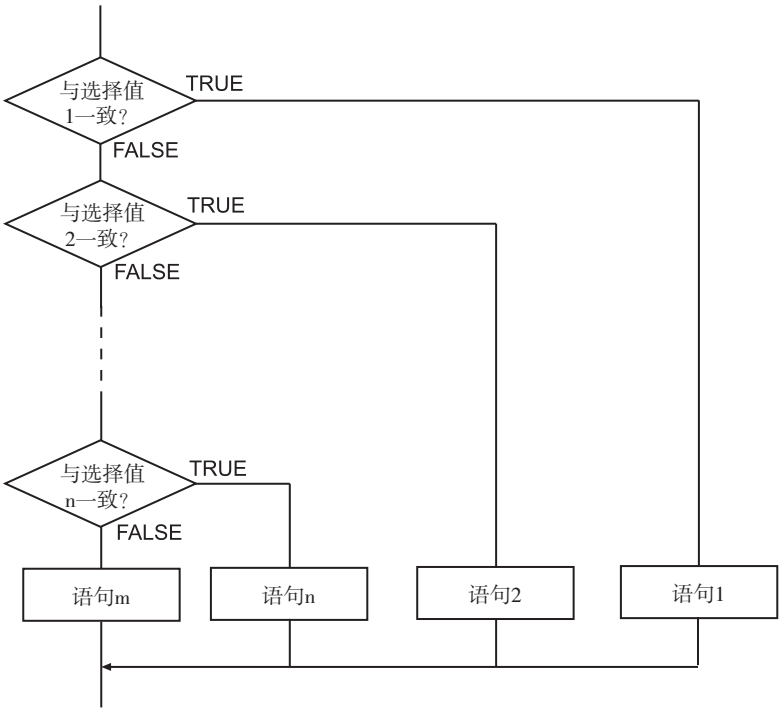
功能

根据指定的整数表达式的值，从多个语句中选择要执行的语句。
整数表达式和选择值中可记述下列内容。

	可记述的内容
整数表达式	整数型变量、整数型常数、整数型表达式、带整数型返回值的函数、枚举体变量、枚举体表达式、待枚举体返回值的函数
选择值	整数型常数

如下图所示，以下记述时的处理流程根据整数表达式的值而定。语句处可记述多个语句。

```
CASE 整数表达式 OF
    1：
      语句1；
    2：
      语句2；
   ：
    n：
      语句n；
    ELSE
      语句m；
END_CASE；
```



参考

- 本指令可用于分层结构。如下所示，整数表达式1的值为1且整数表达式11的值为2时执行语句12。

```
CASE　整数表达式1　OF
    1：
      CASE　整数表达式11　OF
          1：
            语句11;
          2：
            语句12;
          ELSE
            语句1m;
      END_CASE;
    2：
      语句2;
    3：
      语句3;
    ELSE
      语句m;
END_CASE;
```

- 选择值中可记述多个值。多个值之间用',' (逗号)隔开。如下所示，整数表达式的值为1或2时执行语句1。

```
CASE　整数表达式1　OF
    1,2：
        语句1;
    3：
        语句2;
    4：
        语句3;
    ELSE
        语句m;
END_CASE;
```

- 选择值中也可记述连续值。连续值用'..' (2个句号)隔开。如下所示，整数表达式的值大于等于10且小于等于15时执行语句1。

```
CASE　整数表达式1　OF
    10..15：
            语句1;
    16：
            语句2;
    17：
            语句3;
    ELSE
            语句m;
END_CASE;
```

- ELSE可省略。此时，整数表达式的值不等于任何选择值时，不执行任何语句。
- 语句内容无特殊限制。除CASE指令以外可记述任何内容，例如功能块的调用和FOR指令等。
- C语言的switch语句与下列规格不同。C语言的switch语句只要不使用break语句，就执行与整数表达式相等的选择值之后的语句。对此，CASE 语句仅执行符合选择值与整数表达式相等的语句。如下例所示，C语言的switch语句从语句1到语句3全部执行。对此，CASE指令则仅执行语句1。

C语言的switch语句

```
val=1;
switch val
{
    case 1:
        语句1;
    case 2:
        语句2;
    case 3:
        语句3;
}
```

CASE指令

```
val:=1;
CASE val OF
    1:
        语句1;
    2:
        语句2;
    3:
        语句3;
END_CASE;
```

使用注意事项

- CASE和END_CASE不可省略。此外，该两者请务必成对使用。
- 整数表达式与选择值的数据类型可不同。
- 选择值中无法重复记述相同的值。
- 含IF指令、CASE指令、FOR指令、WHILE指令、REPEAT指令在内，最多15层。

示例程序

变量abc的值为INT#1时，在变量def中代入INT#10；为INT#2时，代入INT#20；为INT#3时，代入INT#30。上述以外的值时，代入变量ghi的值。

名称	数据类型	初始值
abc	INT	0
def	INT	0
ghi	INT	0

```
CASE abc OF
    INT#1:
        def:=INT#10;
    INT#2:
        def:=INT#20;
    INT#3:
        def:=INT#30;
    ELSE
        def:=ghi;
END_CASE;
```

变量abc的值为INT#1时，在变量def中代入INT#10；为INT#2或INT#5时，代入INT#20；大于等于INT#6且小于等于INT#10时，代入INT#30。为上述以外的值时，不执行。

名称	数据类型	初始值
abc	INT	0
def	INT	0

```
CASE abc OF
    INT#1:
        def:=INT#10;
    INT#2, INT#5:
        def:=INT#20;
    INT#6..INT#10:
        def:=INT#30;
END_CASE;
```

WHILE

在指定的条件表达式的评估结果为TRUE期间，重复执行某个语句。

指令	名称	FB/FUN	图形表现	ST表现
WHILE	重复	–	无	WHILE 条件表达式 DO 　　语句; END_WHILE;

变量

无

功能

在指定的条件表达式的评估结果为TRUE期间，重复执行某个语句。条件表达式中记述作为评估结果的带TRUE或FALSE值的下列内容。

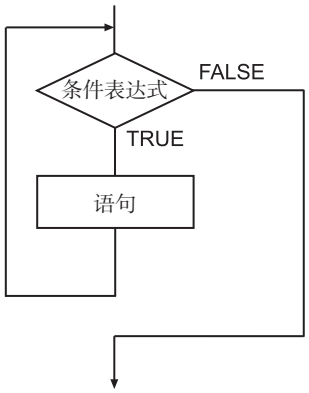
条件表达式可记述的内容	记述示例	评估结果
逻辑表达式	a>3	变量a的值大于3时为TRUE，否则为FALSE
	a=b	变量a、b的值相等时为TRUE，否则为FALSE
BOOL型变量	abc	abc的值为TRUE时TRUE，为FALSE时FALSE
BOOL型常数	TRUE	TRUE
带有BOOL型返回值的函数	FUN名称	该函数的返回值为TRUE时TRUE，为FALSE时FALSE

逻辑表达式中可使用以下运算符。

运算符	含义	记述示例	评估结果
=	一致	a=b	变量a、b的值相等时为TRUE，否则为FALSE
<>	不一致	a<>b	变量a、b的值不相等时为TRUE，否则为FALSE
<	比较	a<b	变量a的值小于b时为TRUE，否则为FALSE
< =		a<=b	变量a的值小于等于b时为TRUE，否则为FALSE
>		a>b	变量a的值大于b时为TRUE，否则为FALSE
>=		a>=b	变量a的值大于等于b时为TRUE，否则为FALSE
AND,&	逻辑积	a AND b a & b	BOOL型变量a、b的逻辑积
OR	逻辑和	a OR b	BOOL型变量a、b的逻辑和
XOR	异或	a XOR b	BOOL型变量a、b的异或
NOT	非	NOT a	非BOOL型变量a

以下记述时，处理流程如下图所示。图中语句处可记述多个语句。

```
WHILE 条件表达式 DO
    语句;
END_WHILE;
```



参考

- 条件表达式的首次评估结果为FALSE时，一次也不执行语句。
- 语句内容无特殊限制。除WHILE指令以外可记述任何内容，例如功能块的调用和FOR指令等。
- 请执行EXIT指令，以中断重复处理。此时，不执行EXIT指令至END_WHILE指令的处理。

使用注意事项

- WHILE和END_WHILE不可省略。此外，该两者请务必成对使用。
- 含IF指令、CASE指令、FOR指令、WHILE指令、REPEAT指令在内，最多15层。

示例程序

变量abc的值小于等于INT#1000期间，持续在abc上加上INT#7。

名称	数据类型	初始值
abc	INT	0

```
abc：= INT#0;
WHILE abc <= INT#1000 DO
    abc：= abc + INT#7;
END_WHILE;
```

REPEAT

执行某个语句一次，在指定的条件表达式变为TRUE之前，重复执行该语句。

指令	名称	FB/FUN	图形表现	ST表现
REPEAT	重复	–	无	REPEAT 　　语句; UNTIL　条件表达式 END_REPEAT;

变量

无

功能

执行某个语句一次，在指定的条件表达式变为TRUE之前，重复执行该语句。条件表达式中记述作为评估结果的带TRUE或FALSE值的下列内容。

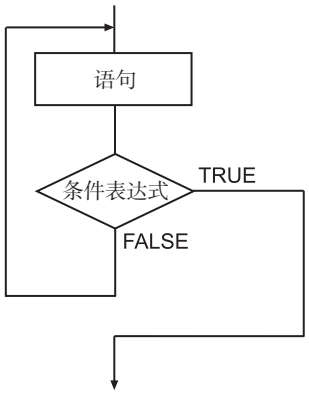
条件表达式可记述的内容	记述示例	评估结果
逻辑表达式	a>3	变量a的值大于3时为TRUE，否则为FALSE
	a=b	变量a、b的值相等时为TRUE，否则为FALSE
BOOL型变量	abc	abc的值为TRUE时TRUE，为FALSE时FALSE
BOOL型常数	TRUE	TRUE
带有BOOL型返回值的函数	FUN名称	该函数的返回值为TRUE时TRUE，为FALSE时FALSE

逻辑表达式中可使用以下运算符。

运算符	含义	记述示例	评估结果
=	一致	a=b	变量a、b的值相等时为TRUE，否则为FALSE
<>	不一致	a<>b	变量a、b的值不相等时为TRUE，否则为FALSE
<	比较	a<b	变量a的值小于b时为TRUE，否则为FALSE
< =		a<=b	变量a的值小于等于b时为TRUE，否则为FALSE
>		a>b	变量a的值大于b时为TRUE，否则为FALSE
>=		a>=b	变量a的值大于等于b时为TRUE，否则为FALSE
AND,&	逻辑积	a AND b a & b	BOOL型变量a、b的逻辑积
OR	逻辑和	a OR b	BOOL型变量a、b的逻辑和
XOR	异或	a XOR b	BOOL型变量a、b的异或
NOT	非	NOT a	非BOOL型变量a

以下记述时，处理流程如下图所示。图中语句处可记述多个语句。

```
REPEAT
    语句;
UNTIL　条件表达式
END_REPEAT;
```



参考

- 执行一次语句后，评估条件表达式。因此，务必先执行一次语句。
- 语句内容无特殊限制。除REPEAT指令以外可记述任何内容，例如功能块的调用和FOR指令等。
- 请执行EXIT指令，以中断重复处理。此时，不执行EXIT指令至END_REPEAT指令的处理。

使用注意事项

- REPEAT、UNTIL、END_REPEAT均不可省略。此外，这些指令务必成组使用。
- 含IF指令、CASE指令、FOR指令、WHILE指令、REPEAT指令在内，最多15层。

示例程序

变量abc的值小于等于INT#10期间，持续在abc上加上INT#1。

名称	数据类型	初始值
abc	INT	0

```
abc：= INT#0;
REPEAT
      abc：= abc + INT#1;
UNTIL abc > INT#10
END_REPEAT;
```

EXIT

中断最内侧FOR指令、WHILE指令或REPEAT指令的重复处理。

指令	名称	FB/FUN	图形表现	ST表现

EXIT	循环中断	–	无	FOR Index:=0 TO 9 BY 1 DO
				IF Error[Index] THEN
				EXIT;
				END_IF;
				END_FOR;

变量

无

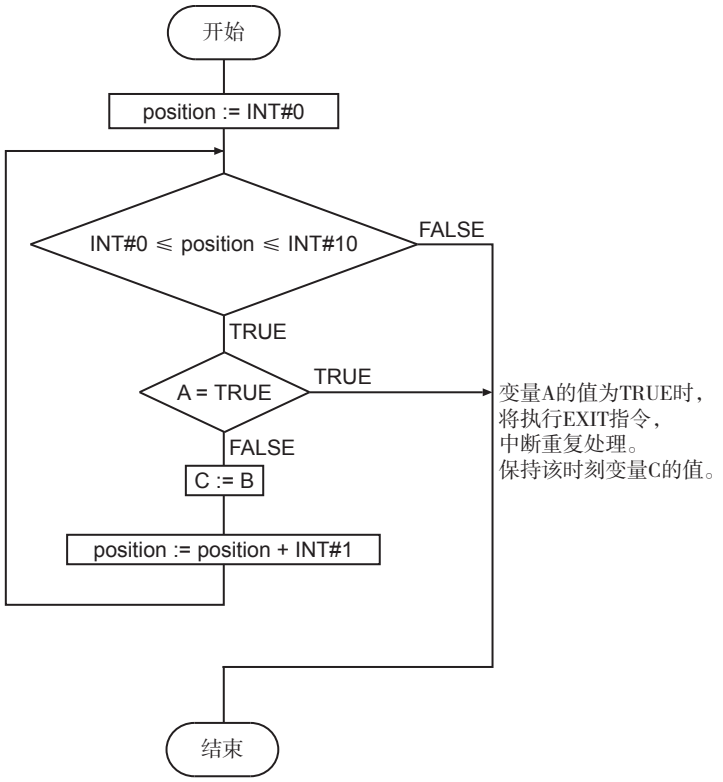
功能

中断最内侧FOR指令、WHILE指令或REPEAT指令的重复处理。处理移至重复处理的下一个处理。

以下用户程序时，在执行FOR指令的重复处理过程中，每次都会确认变量A的值。变量A的值为TRUE时，将执行EXIT指令，结束重复处理。此时，不执行END_IF的下一指令 C:=B; ，变量C保持原值。

```
FOR position:=INT#0 TO INT#10 BY INT#1 DO
    IF (A=TRUE) THEN
       EXIT;
   END_IF;
   C:=B;
END_FOR;
```

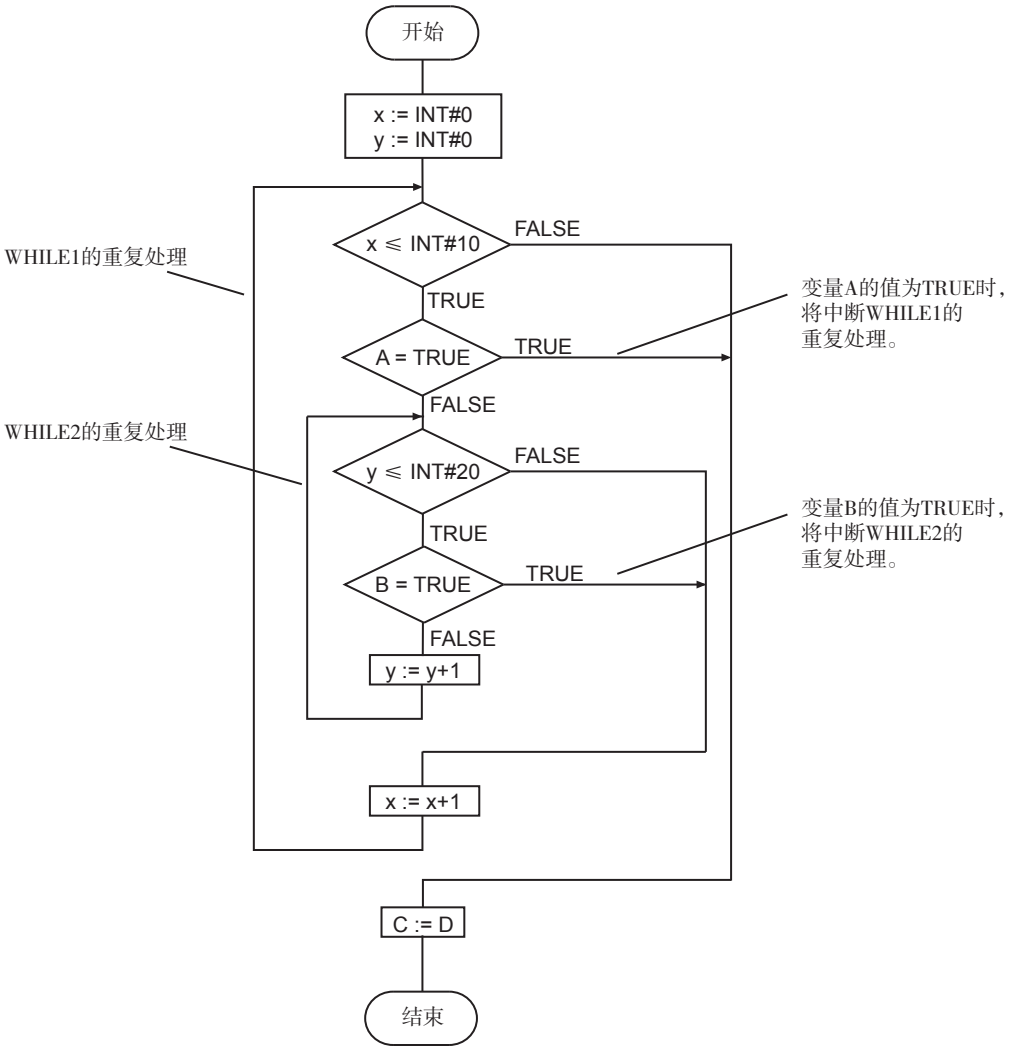
流程图如下所示。



因执行本指令而中断的只有最内侧的重复处理。因此，以下用户程序的情况，变量B的值为TRUE时则执行EXIT指令2，中断WHILE指令2的重复处理。其结果，处理移至x:=x+1;。此时，上1层WHILE指令1的重复处理将继续执行。
变量A的值为TRUE时，将执行EXIT指令1，中断WHILE指令1的重复处理。其结果，处理移至C:=D;。

```
x:=INT#0;
y:=INT#0;
WHILE x<=INT#10 DO       // WHILE指令1
  IF (A=TRUE) THEN
    EXIT;                // EXIT指令1
  END_IF;
  WHILE y<=INT#20 DO     // WHILE指令2
    IF (B=TRUE) THEN
      EXIT;              // EXIT指令2
    END_IF;
    y := y+1;
  END_WHILE;
  x = x+1;
END_WHILE
C:=D;
```

流程图如下所示。



使用注意事项

- 请务必在FOR与END_FOR之间、WHILE与END_WHILE之间或REPEAT与END_REPEAT之间使用本指令。
- 分层使用(嵌套)重复处理时，如需中断所有重复处理，仅需按层数执行本指令。

RETURN

RETURN指令的说明请参阅时序控制指令的 “RETURN指令(P.2–67)”。

FOR

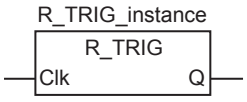
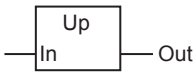
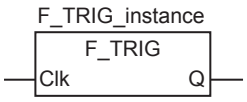
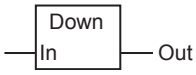
FOR指令的说明请参阅时序控制指令的 “FOR/NEXT指令(P.2-84)”。

时序输入指令

指令	名称	页码
R_TRIG,Up/ F_TRIG,Down	上升沿微分/ 下降沿微分	2–46
TestABit/TestABitN	位测试/ 非位测试	2–49

R_TRIG，Up/F_TRIG，Down

R_TRIG,Up ：输入信号处于上升沿(FALSE→TRUE)时，仅1个任务周期输出TRUE。
F_TRIG,Down：输入信号处于下降沿(TRUE→FALSE)时，仅1个任务周期输出TRUE。

指令	名称	FB/FUN	图形表现	ST表现
R_TRIG	上升沿微分	FB		R_TRIG_instance(Clk, Q);
Up		FUN		无
F_TRIG	下降沿微分	FB		F_TRIG_instance(Clk, Q);
Down		FUN		无

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Clk,In	输入信号	输入	输入信号	遵从数据类型	–	–
Q,Out	输出信号	输出	输出信号	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Clk,In	○																			
Q,Out	○																			

功能

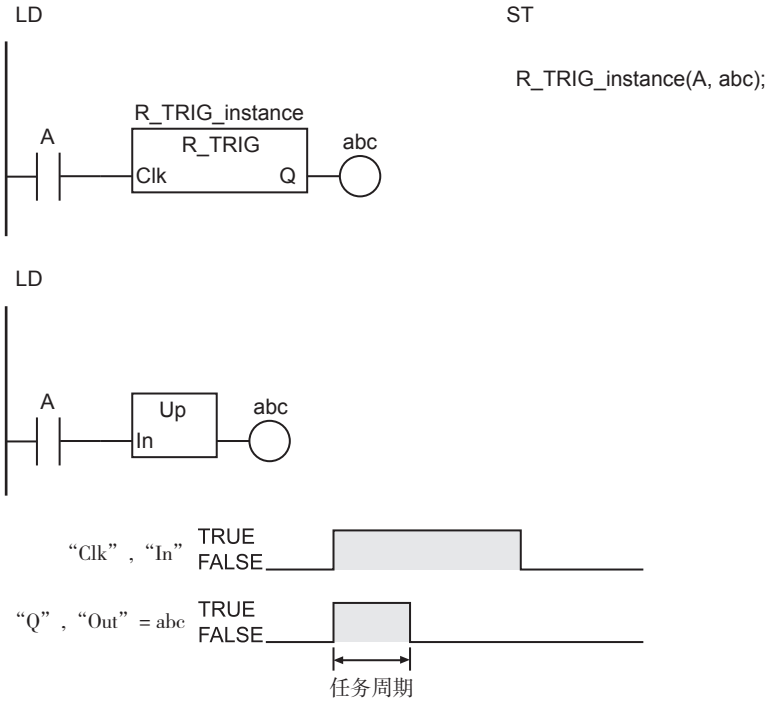
- **R_TRIG**

  输入信号 “Clk”处于上升沿(FALSE→TRUE)时，仅1个任务周期将TRUE代入输出信号 “Q”。其余情况下，“Q”的值均为FALSE。

- **Up**

  R_TRIG指令和Up指令的功能相同。请分别将F_TRIG指令功能说明中的变量名称 “Clk”改称为 “In”，“Q”改称为 “Out”。
  但首次执行本指令的任务周期的动作与R_TRIG指令的动作不同。首次执行本指令的任务周期内的动作请参阅 “使用注意事项(P.2–48)”。

描述示例和时序图如下所示。



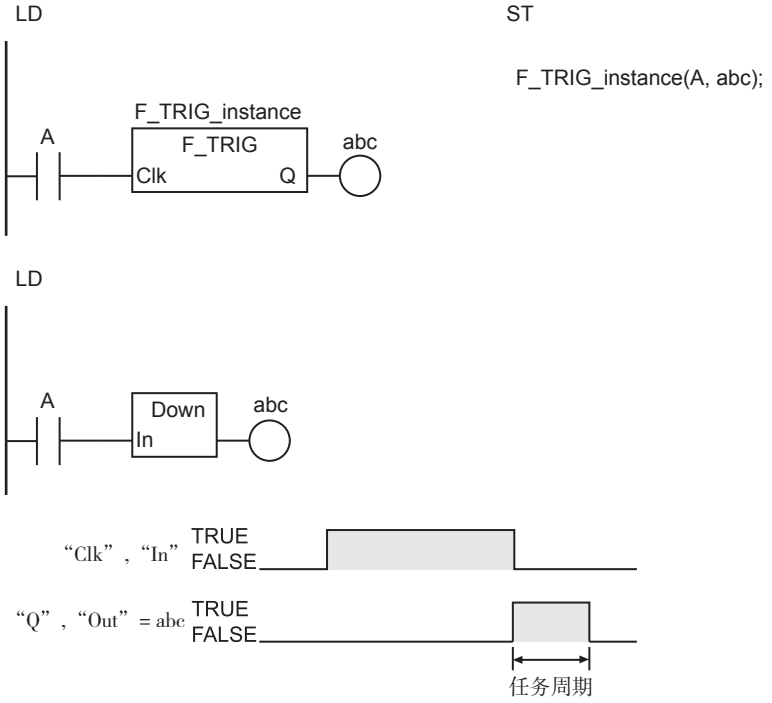
- F_TRIG

  输入信号 “Clk”处于下降沿(TRUE→FALSE)时，仅1个任务周期将TRUE代入输出信号 “Q”。其余情况下，“Q”的值均为FALSE。

- Down

  F_TRIG指令和Down指令的功能完全相同。请分别将F_TRIG指令功能说明中的变量名称 “Clk”改称为 “In”，“Q”改称为 “Out”。

  描述示例和时序图如下所示。



使用注意事项

- 根据上次和当前执行本指令时的 “Clk”或 “In”值的不同，进行上升沿/下降沿检测。因此，使用JMP指令等后未在每次任务周期内均执行本指令时，敬请注意。
- 即使电源断开，也不识别为 “Clk”或 “In”的值变为FALSE。仅 “Clk”、“In”的值在FALSE状态下，本指令评估了 “Clk”、“In”时才会识别 “Clk”、“In”的值变为FALSE。
- 首次执行Up指令的任务周期内，与 “In”的值无关，“Out”的值必定会变为FALSE。
- 电源ON时，如果Up指令的 “In”值为TRUE，则在 “In”的值变为FALSE再到变为TRUE的过程中，“Out”的值均为FALSE。
- 首次执行F_TRIG指令的任务周期内，与 “Clk”的值无关，“Q”的值必定会变为FALSE。
- 电源ON时，如果F_TRIG指令的 “Clk”值为FALSE，则在 “Clk”的值变为TRUE再到变为FALSE的过程中，“Q”的值均为FALSE。
- 首次执行Down指令的任务周期内，与 “In”的值无关，“Out”的值必定会变为FALSE。
- 电源ON时，如果Down指令的 “In”值为FALSE，则在 “In”的值变为TRUE再到变为FALSE的过程中，“Out”的值均为FALSE。

版本相关信息

如果 “Clk”的值为TRUE，并在以下时间内执行R_TRIG指令，则 “Q”的值因CPU单元的版本而异。

“Clk”的值为TRUE时执行本指令的时间	“Q”的值	
	CPU单元Ver.1.02以上	CPU单元Ver.1.01以下
首次执行R_TRIG指令的任务周期	变为TRUE。	必定变为FALSE。
电源ON时	变为TRUE。	在 “Clk”的值变为FALSE再到变为TRUE的过程中，“Q”的值均为FALSE。

TestABit/TestABitN

TestABit ：输出位串指定位的值。
TestABitN：输出位串指定位的取反值。

指令	名称	FB/FUN	图形表现	ST表现
TestABit	位测试	FUN		Out：= TestABit (In, Pos);
TestABitN	非位测试	FUN		Out：= TestABitN (In, Pos);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	位串	输入	位串	遵从数据类型	–	(*)
Pos	位位置		指定位位置	0～“In”的位数–1		0
Out	位值	输出	• TestABit 指定位的值 • TestABitN 指定位的取反值	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Pos						○														
Out	○																			

功能

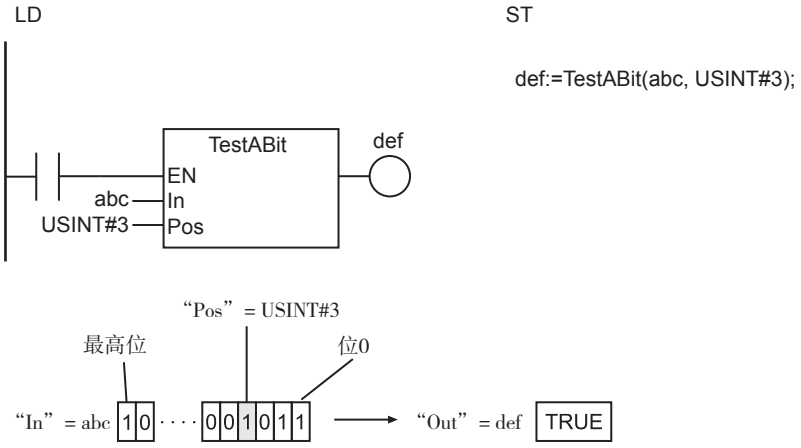
- **TestABit**

  EN为TRUE时，将位串 “In”的位位置 “Pos”的值代入位值“Out”。EN为FALSE时，“Out”的值为FALSE。

- **TestABitN**

  EN为TRUE时，将位串 “In”的位位置 “Pos”的取反值代入位值 “Out”。EN为FALSE时，“Out”的值为FALSE。

TestABit指令下，“Pos” = USINT#3时的示例如下所示。



使用注意事项

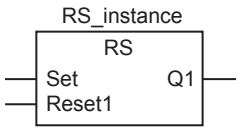
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- 以下情况时会发生异常，“Out”的值为FALSE。
  - “Pos”的值大于 “In”的位数–1时。

时序输出指令

指令	名称	页码
RS	复位优先保持	2–52
SR	置位优先保持	2–54
Set/Reset	置位/复位	2–56
SetBits/ResetBits	多位置位/ 多位复位	2–59
SetABit/ResetABit	1位置位/ 1位复位	2–61
OutABit	1位输出	2–63

RS

保持BOOL型变量的值。置位输入与复位输入同时TRUE时，复位输入优先。

指令	名称	FB/FUN	图形表现	ST表现
RS	复位优先保持	FB		RS_instance (Set, Reset1, Q1);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Set*1	置位	输入	置位输入	遵从数据类型	–	0
Reset1*1	复位		复位输入			
Q1	保持	输出	保持输出	遵从数据类型	–	–

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Set、Reset1分别缩记为S、R1。例如，可记为：RS_instance(S:=A, R1:=B, Q1=>abc);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Set	○																			
Reset1	○																			
Q1	○																			

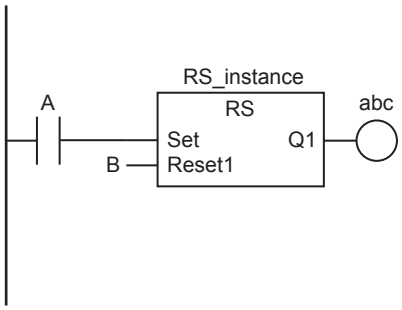
功能

复位优先的自保持电路。输入输出的关系如下所示。

“Set”的值	“Reset1”的值	“Q1”的值
TRUE	TRUE	FALSE
TRUE	FALSE	TRUE
FALSE	TRUE	FALSE
FALSE	FALSE	保持

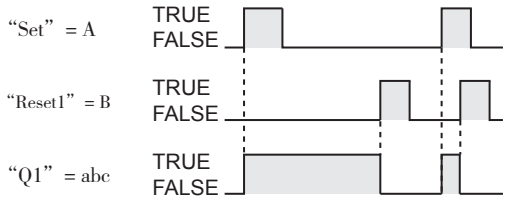
描述示例和时序图如下所示。

LD



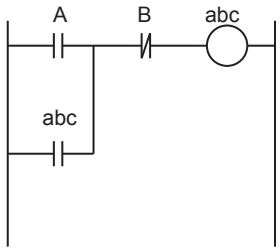
ST

```
RS_instance(A, B, abc);
```



参考

- RS指令与下图所示的自保持电路的动作相同。



- 但RS指令在主站控制区域中，通过主站控制进行复位时，与上述自保持电路的动作不同。

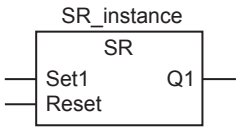
电路	B的值	abc的值
RS指令	TRUE	保持
	FALSE	FALSE
自保持电路	TRUE	FALSE
	FALSE	

使用注意事项

- 请勿从b触点的外部设备直接导入 "Reset1" 输入。AC电源断开或瞬时停电时，有时控制器的内部电源不会立即关闭，输入单元的输入会先打开。最终，"Reset1" 输入的值会变为TRUE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则保持 "Q1" 的值。
- 未通过执行跳转类指令(JMP指令等)来执行本指令时，"Q1" 保持上次执行时的值。
- 本指令在主站控制区域中，通过主站控制进行复位时，动作如下所示。
  - "Reset1" 的值为TRUE时，保持 "Q1" 的值；为FALSE时，"Q1" 的值变为FALSE。
  - 即使 "Q1" 的值为TRUE，连接至 "Q1" 后段的指令中也会输入FALSE。
- "Q1" 即使连接带保存属性的参数，断电时仍无法保持值。电源恢复后将动作模式设为运行模式并执行本指令时，"Q1" 的值将变为FALSE。但在参考栏的自保持电路的情况下，电源恢复后仍将保持值。

SR

保持BOOL型变量值。置位输入与复位输入同时TRUE时，置位输入优先。

指令	名称	FB/FUN	图形表现	ST表现
SR	置位优先保持	FB		SR_instance(Set1, Reset, Q1);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Set1*1	置位	输入	置位输入	遵从数据类型	–	0
Reset	复位		复位输入			
Q1	保持	输出	保持输出	遵从数据类型	–	–

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Set1、Reset分别缩记为S1、R。例如，可记为：SR_instance(S1:=A, R:=B, Q1=>abc);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Set1	○																			
Reset	○																			
Q1	○																			

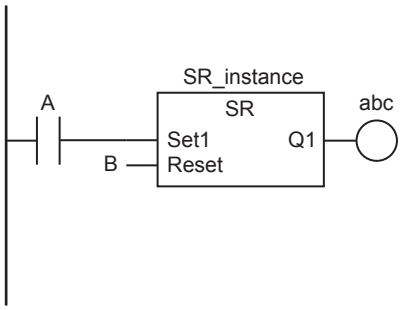
功能

置位优先的自保持电路。输入输出的关系如下所示。

“Set1”的值	“Reset”的值	“Q1”的值
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	FALSE
FALSE	FALSE	保持

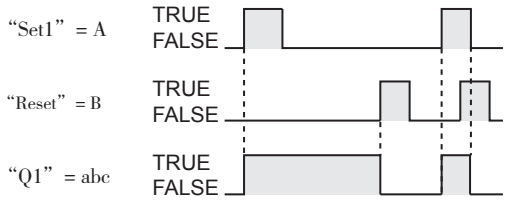
描述示例和时序图如下所示。

LD



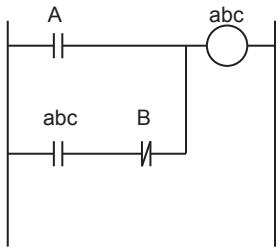
ST

```
SR_instance(A, B, abc);
```



参考

- SR指令与下图所示的自保持电路的动作相同。



- 但SR指令在主站控制区域中，通过主站控制进行复位时，与上述自保持电路的动作不同。

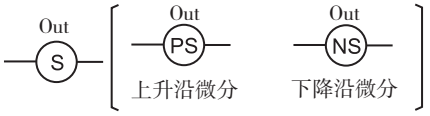
电路	B的值	abc的值
SR指令	TRUE	保持
	FALSE	FALSE
自保持电路	TRUE	FALSE
	FALSE	

使用注意事项

- 请勿从b触点的外部设备直接导入 “Reset”输入。AC电源断开或瞬时停电时，有时控制器的内部电源不会立即关闭，输入单元的输入会先打开。最终，“Reset”输入的值会变为TRUE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则保持 “Q1”的值。
- 未通过执行跳转类指令(JMP指令等)来执行本指令时，“Q1”保持上次执行时的值。
- 本指令在主站控制区域中，通过主站控制进行复位时，动作如下所示。
  - “Reset”的值为TRUE时，保持 “Q1”的值；为FALSE时，“Q1”的值变为FALSE。
  - 即使 “Q1”的值为TRUE，连接至 “Q1”后段的指令中也会输入FALSE。
- “Q1”即使连接带保存属性的参数，断电时仍无法保持值。电源恢复后将动作模式设为运行模式并执行本指令时，“Q1”的值将变为FALSE。但在参考栏的自保持电路的情况下，电源恢复后仍将保持值。

Set/Reset

Set ：设置BOOL型变量为TRUE。
Reset：设置BOOL型变量为FALSE。

指令	名称	FB/FUN	图形表现	ST表现
Set	置位	–		无
Reset	复位	–		无

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	输出	输出	输出	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			

功能

- Set

向本指令的输入为TRUE时，为 “Out”设置TRUE。
将 “Out”设置为TRUE后，即使输入变为FALSE，“Out”也不会变为FALSE。
将 “Out”恢复为FALSE时，需使用Reset指令。

- Reset

向本指令的输入为TRUE时，为 “Out”设置FALSE。
将 “Out”设置为FALSE后，即使输入变为FALSE，“Out”也不会变为TRUE。
将 “Out”恢复为TRUE时，需使用Set指令。

未指定上升沿微分、下降沿微分时的动作如下所示。

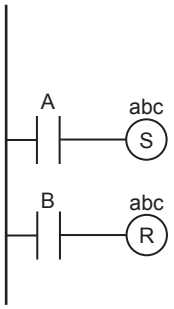
指令	输入	输出
Set	TRUE	TRUE
	FALSE	保持
Reset	TRUE	FALSE
	FALSE	保持

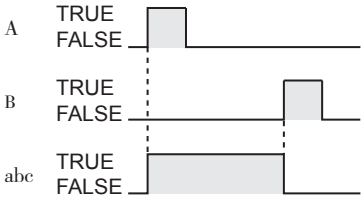
如下所示，指定上升沿微分、下降沿微分时，根据上次执行时的值和当前输入的值动作。

指令	微分指定	上次执行时的值与当前输入的值	输出
Set	上升沿微分	上次执行时FALSE → 当前TRUE	TRUE
		上述以外型号	保持
	下降沿微分	上次执行时TRUE → 当前FALSE	TRUE
		上述以外型号	保持
Reset	上升沿微分	上次执行时FALSE → 当前TRUE	FALSE
		上述以外型号	保持
	下降沿微分	上次执行时TRUE → 当前FALSE	FALSE
		上述以外型号	保持

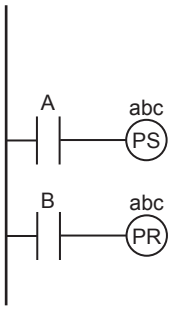
描述示例和时序图如下所示。

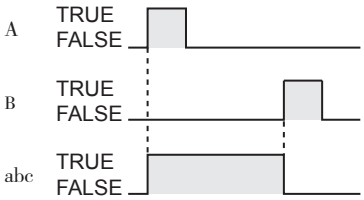
LD



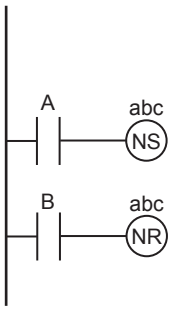


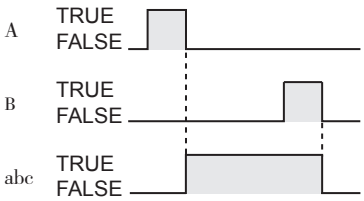
LD





LD





参考

Set/Reset指令与Out指令的区别

- Set/Reset指令在输入TRUE时动作，输入FALSE时不动作。即输入FALSE时，输出无变化。
- 到前段为止的结果为TRUE时，Out指令将指定变量设为TRUE；到前段为止的结果为FALSE时，将指定变量设为FALSE。即无论输入TRUE还是FALSE，均会动作。

Set/Reset指令与SR/RS指令的区别

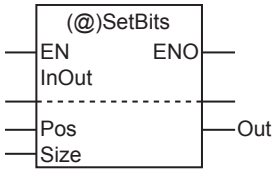
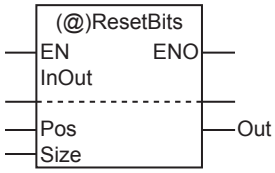
- SR/RS指令必须将 “Set”输入和 “Reset”输入记述在程序中的相同位置，而Set/Reset指令可将其记述在不同位置。

使用注意事项

- 本指令在主站控制区域中，通过主站控制进行复位时，保持 “Out”的值。
- 未通过执行跳转类指令(JMP指令等)来执行本指令时，保持 “Out”的值。
- 指定上升沿微分并在电源启动时输入为TRUE时，不会直接执行本指令。而是在输入先变为FALSE再变为TRUE时执行。
- 未指定上升沿微分并在电源启动时输入为TRUE时，即使输入未变为FALSE也将执行本指令。

SetBits/ResetBits

SetBits ：设置位串数据的连读多个位为TRUE。
ResetBits：设置位串数据的连读多个位为FALSE。

指令	名称	FB/FUN	图形表现	ST表现
SetBits	多位置位	FUN		SetBits(InOut, Pos, Size);
ResetBits	多位复位	FUN		ResetBits(InOut, Pos, Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut	位串	输入输出	位串	遵从数据类型	–	–
Pos	位位置	输入	指定位的位置	0～“InOut”的位数–1	–	0
Size	位数		位数	0～“InOut”的位数		1
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

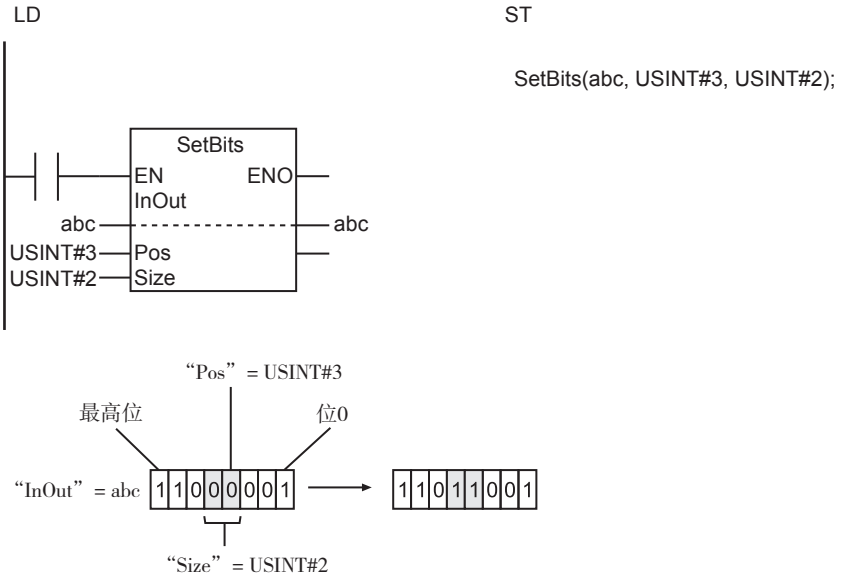
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut		○	○	○	○															
Pos						○														
Size						○														
Out	○																			

功能

- SetBits
  将位串 “InOut”的位位置 “Pos”中的 “Size”个位的值设置为TRUE。其它位不变。

- ResetBits
  将位串 “InOut”的位位置 “Pos”中的 “Size”个位的值设置为FALSE。其它位不变。

SetBits指令下，“Pos”= USINT#3、“Size”= USINT#2时的示例如下所示。



参考

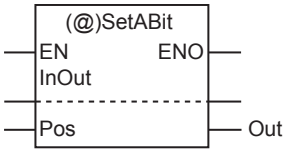
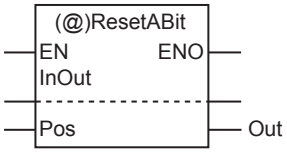
使用本指令时，对在以数据存储器等的通道(字)为单位处理数据的存储区域中AT指定的变量，可整体置位/复位为TRUE/FALSE。

使用注意事项

- 本指令在主站控制区域中，通过主站控制进行复位时，保持 “InOut”的值。
- 未通过执行跳转类指令(JMP指令等)来执行本指令时，保持 “InOut”的值。
- “Size”的值为0时，“InOut”的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、“InOut”不变。
  - “Pos”的值大于 “InOut”的位数–1时。
  - “Size”的值超过有效范围时。
  - “Pos”和 “Size”的指定超过 “InOut”的位数时。

SetABit/ResetABit

SetABit ：设置位串数据的指定位为TRUE。
ResetABit：设置位串数据的指定位为FALSE。

指令	名称	FB/FUN	图形表现	ST表现
SetABit	1位置位	FUN		SetABit (InOut, Pos);
ResetABit	1位复位	FUN		ResetABit (InOut, Pos);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut	位串	输入输出	位串	遵从数据类型	–	–
Pos	位位置	输入	指定位的位置	0～“InOut”的位数–1	–	0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut		○	○	○	○															
Pos						○														
Out	○																			

功能

- SetABit

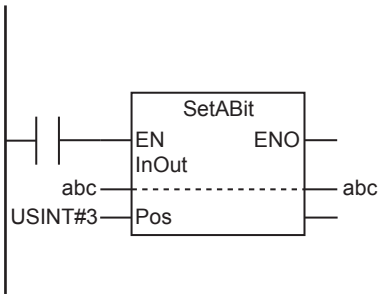
  将位串 “InOut”的位位置 “Pos”的值设置为TRUE。其它位不变。
  执行后即使EN变为FALSE，“InOut”的 “Pos”位也不变。

- ResetABit

  将位串 “InOut”的位位置 “Pos”的值设置为FALSE。其它位不变。
  执行后即使EN变为FALSE，“InOut”的 “Pos”位也不变。

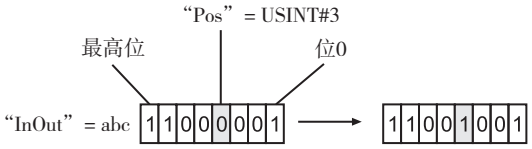
SetABit指令下，“Pos”=USINT#3时的示例如下所示。

LD



ST

```
SetABit(abc, USINT#3);
```



参考

SetABit/ResetABit指令与OutABit指令的区别

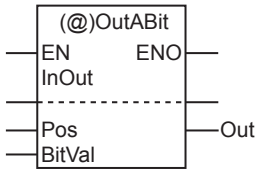
- SetABit/ResetABit指令将为指定位设置的值固定为TRUE或FALSE。
- OutABit指令为指定位设置的值可更改。

使用注意事项

- 本指令在主站控制区域中，通过主站控制进行复位时，保持 “InOut”的值。
- 未通过执行跳转类指令(JMP指令等)来执行本指令时，保持 “InOut”的值。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、“InOut”不变。
  - “Pos”的值大于 “In”的位数–1时。

OutABit

设置位串数据的指定位为TRUE/FALSE。

指令	名称	FB/FUN	图形表现	ST表现
OutABit	1位输出	FUN		OutABit (InOut, Pos, BitVal);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut	位串	输入输出	位串	遵从数据类型	–	–
Pos	位位置	输入	指定位的位置	0～“InOut”的位数–1	–	0
BitVal	置位值		进行置位的值	遵从数据类型		TRUE
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

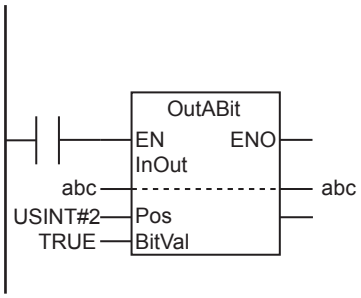
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut		○	○	○	○															
Pos						○														
BitVal	○																			
Out	○																			

功能

在位串 “InOut”的位位置 “Pos”中保存置位值 “BitVal”的值。“Pos”以外的位位置的值不变。

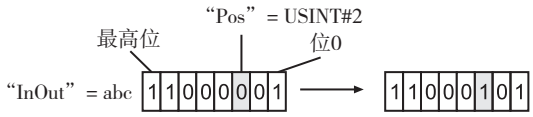
“Pos” = USINT#2、“BitVal” = TRUE时的示例如下所示。

LD



ST

```
OutABit(abc, USINT#2, TRUE);
```



参考

SetABit/ResetABit指令与OutABit指令的区别

- SetABit/ResetABit指令将为指定位设置的值固定为TRUE或FALSE。
- OutABit指令可通过更改 “BitVal”的值来更改为指定位设置的值。

使用注意事项

- 本指令在主站控制区域中，通过主站控制进行复位时，保持 “InOut”的值。
- 未通过执行跳转类指令(JMP指令等)来执行本指令时，保持 “InOut”的值。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、“InOut”不变。
  - “Pos”的值大于 “InOut”的位数–1时。

时序控制指令

End

结束程序当前任务周期内的执行。

指令	名称	FB/FUN	图形表现	ST表现
End	结束	FUN	End EN ENO	无

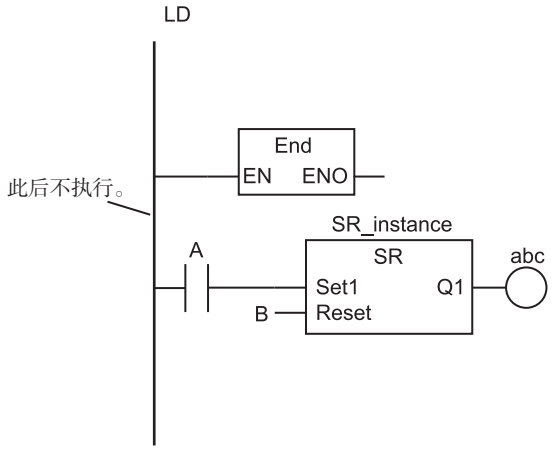
变量

无

功能

结束程序当前任务周期内的执行。

示例如下所示。本示例中，执行End指令时，不执行之后的SR指令。



使用注意事项

- 本指令仅用于程序内。
- 如果将本指令用于函数、功能块、内联ST，则编连时会发生异常。
- 本产品请务必直接连接至左母线。

RETURN

结束函数或功能块，将处理恢复为调用源。

指令	名称	FB/FUN	图形表现	ST表现
RETURN	复位	FUN		RETURN

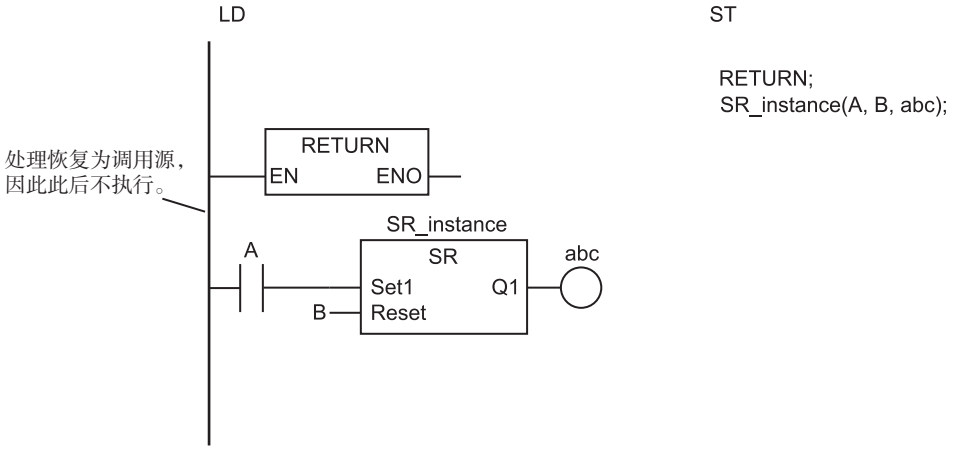
变量

无

功能

结束函数或功能块，将处理恢复为调用源。

示例如下所示。本示例中，执行RETURN指令时，不执行之后的SR指令。

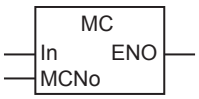
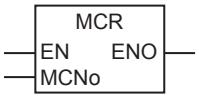


使用注意事项

- 在梯形图程序中使用本指令时，请注意以下事项。
  - 请将本指令用于函数或功能块内。如果用于程序内，则编连时会发生异常。
  - 本产品请务必直接连接至左母线。
- 执行本指令前，请对相应POU的返回值、输出变量、ENO的值进行设置。
- 多次使用本指令时，处理的流程将难以理解。请予以注意。

MC/MCR

MC ：表示主站控制区域的起始点，执行基于主站控制的复位。
MCR：表示主站控制区域的终止点。

指令	名称	FB/FUN	图形表现	ST表现
MC	主站控制开始	–		无
MCR	主站控制结束	–		无

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In (仅MC指令)	主站控制输入	输入	FALSE：通过主站控制执行复位	遵从数据类型	–	–
MCNo	主站控制No.		主站控制No.	0 ~ 14(*)		0

* 通过Sysmac Studio自动登录。用户无需设定。

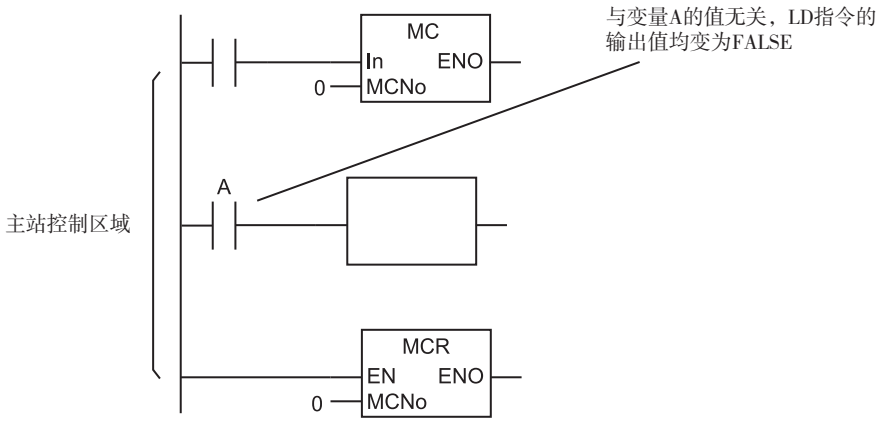
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In (仅MC指令)	○																			
MCNo							○													

功能

主站控制是指将所有存在于用户程序内的指定区域的 POU 控制为停止处理或与之相当的状态。使用主站控制功能后，可轻松控制一系列较长处理的执行条件。

通过主站控制控制的用户程序内的区域称为主站控制区域。主站控制区域的起始位置、终止位置分别配置MC指令、MCR指令。
如下图所示，MC指令的主站控制输入“In”的值变为FALSE时，主站控制区域内与左母线连接的LD指令的输出会强制变为FALSE。该状态称为基于主站控制的复位。

执行基于主站控制的复位时，LD指令后段的POU原则上会进行执行条件为FALSE时的动作。但部分POU可能会不按原则执行动作。对此将在下文阐述。



“In”的值为TRUE时，不会进行基于主站控制的复位。主站控制区域中的POU会进行常规动作。

执行基于主站控制的复位时各POU的动作

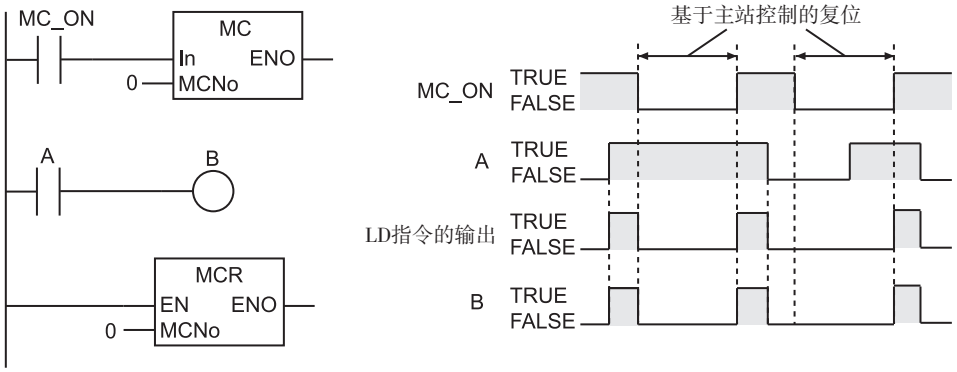
如下所示，执行基于主站控制的复位时的动作因POU而异。

POU	动作
Out/OutABit指令	向指定变量输出FALSE。
OutNot指令	向指定变量输出FALSE。
Set/Reset指令	保持执行基于主站控制的复位前的输出。
TON指令	执行定时器输入 “In” 的值为FALSE时的动作。即定时器复位。 经过时间 “ET” 的值为0，定时器输出 “Q” 的值为FALSE。
TOF指令	执行定时器输入 “In” 的值为TRUE时的动作。即定时器复位。 经过时间 “ET” 的值为0，定时器输出 “Q” 的值为TRUE。 将 “Q” 连接至后段的Out指令时，Out指令中会输入FALSE。
TP指令	执行定时器输入 “In” 的值为FALSE时的动作。即定时器复位。 计时中 ：经过时间 “ET” 的值会增加直至计时结束，结束后变为0。定时器输出 “Q” 的值保持TRUE直至计时结束，结束后变为FALSE。 非计时中：“ET” 的值变为0，“Q” 的值变为FALSE。 将 “Q” 连接至后段的Out指令时，即使在计时中，Out指令中也会输入FALSE。
AccumulationTimer指令	执行定时器输入 “In” 的值为FALSE时的动作。即定时器停止。 保持累计时间 “ET” 和定时器输出 “Q” 的值。 将 “Q” 连接至后段的Out指令相时，“Q” 的值即使为TRUE，Out指令中也会输入FALSE。 复位 “Reset” 为有效。
Timer指令	执行定时器输入 “In” 的值为FALSE时的动作。即定时器复位。 为剩余时间 “ET” 设定设定时间 “PT” 的值，定时器输出 “Q” 的值变为FALSE。
CTU/CTD/CTUD指令	不执行。执行基于主站控制的复位前，已在执行这些指令时，计数值会保持执行复位前的值。 将计数标志 “Q” 连接至后段的Out指令时，该Out指令中会输入FALSE。
JMP指令	不执行。
FOR/NEXT指令	不执行。
BREAK指令	不执行。
跨越多个任务周期执行的功能块 (输出变量中带 “Done”、“Busy”、“Error” 的指令)	连接至左母线的执行条件的值变为FALSE。 执行基于主站控制的复位前，已在执行这些指令时，将继续执行直至处理结束。此时，执行 “Done”、“Busy”、“Error” 的输出，但其下段与线圈连接时必定输出FALSE。直接将变量连接至 “Done”、“Busy”、“Error” 时，将在这些变量中代入符合指令规格的值。 也可以 “实例名称.输出变量” 的形式获取 “Done”、“Busy”、“Error” 的值。
上述以外的函数	不执行。
上述以外的功能块	连接至左母线的执行条件的值变为FALSE。

下面详细介绍典型指令的动作。

● Out

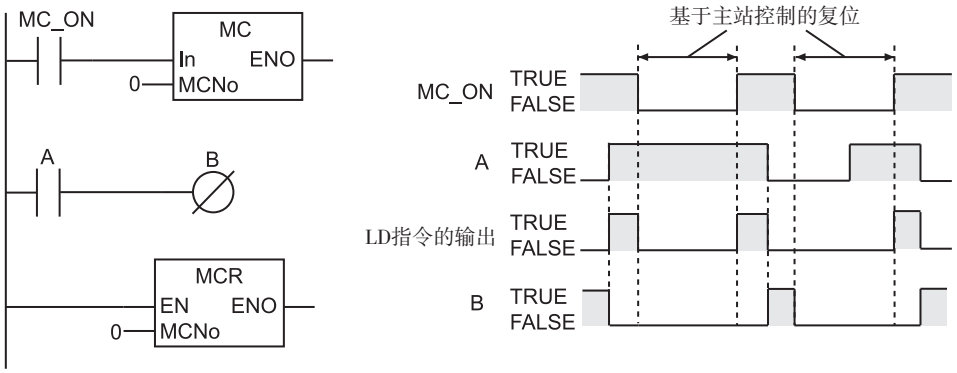
执行基于主站控制的复位时，输出FALSE。



● OutNot

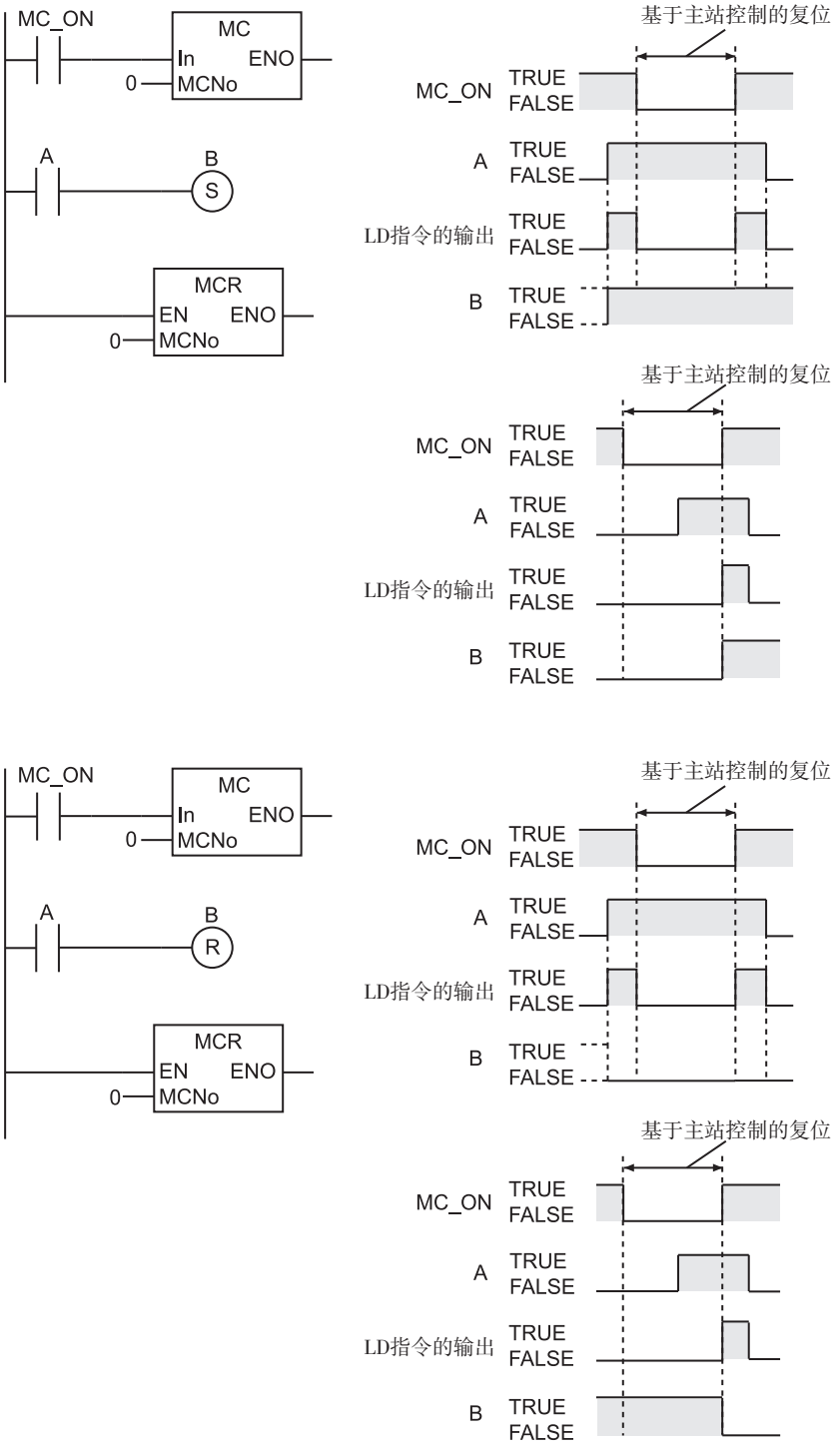
执行基于主站控制的复位时，输出FALSE。

前段LD指令的输出为FALSE时，动作与通常的OutNot指令不同，敬请注意。



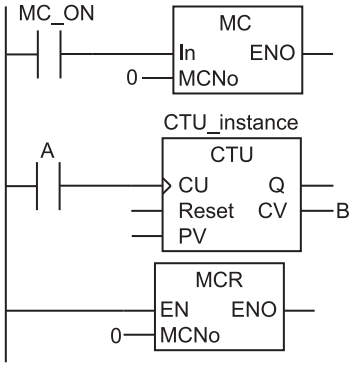
- Set/Reset

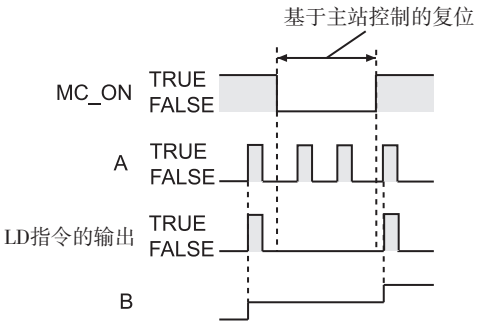
执行基于主站控制的复位时，保持此前的输出值。



● CTU/CTD/CTUD

执行基于主站控制的复位时，保持此前的计数值。解除基于主站控制的复位后，从保持的计数值开始继续计数。





带输入上升沿微分、输入下降沿微分指定的POU的动作

带输入上升沿微分指定、输入下降沿微分指定的POU中含有以下内容。

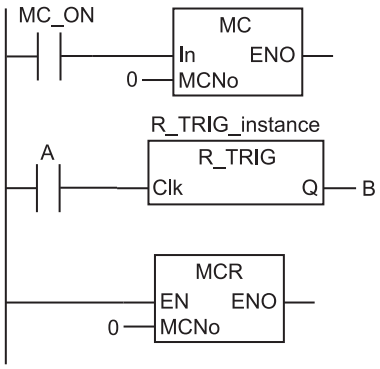
微分型	指令
输入上升沿微分型	• 带上升沿微分指定的LD、LDN、AND、ANDN、OR、ORN、Out • R_TRIG、Up • 带输入上升沿微分型的动作选项(@)的函数 • 带输入上升沿微分型指定的功能块(计数器指令等)
输入下降沿微分型	• 带下降沿微分指定的LD、LDN、AND、ANDN、OR、ORN、Out • F_TRIG、Down

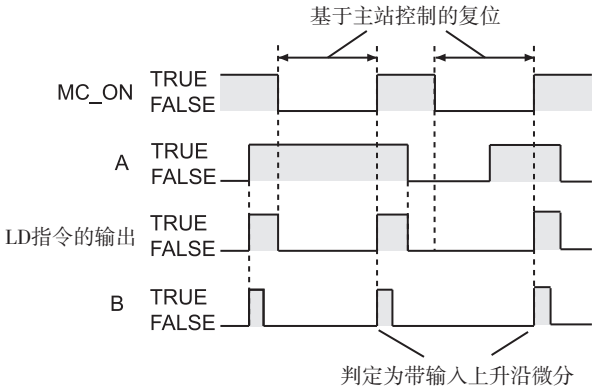
执行或解除基于主站控制的复位时，这些POU的执行条件的值会发生变化。因此，这些POU的执行条件的上升沿微分和下降沿微分的条件可能会成立。如此，发生的执行条件的上升沿微分和下降沿微分均能有效发挥作用。

下面详细介绍典型指令的动作。

● R_TRIG,Up

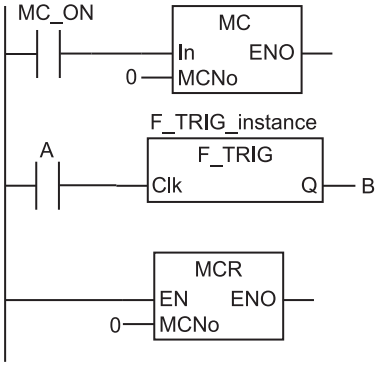
执行基于主站控制的复位时，执行条件的值会变为 FALSE。之后，解除基于主站控制的复位时，如果执行条件的值为TRUE，则判定为发生了输入上升沿微分，进行动作。

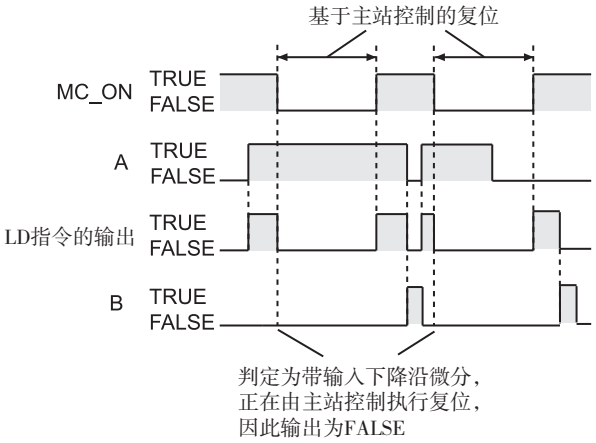




● F_TRIG,Down

执行基于主站控制的复位时，执行条件的值会变为FALSE。如果此前的执行条件的值已为TRUE，则由此判定为发生了输入下降沿微分。但执行基于主站控制的复位时的 F_TRIG、Down 的输出值强制变为FALSE，因此最终输出值为FALSE。

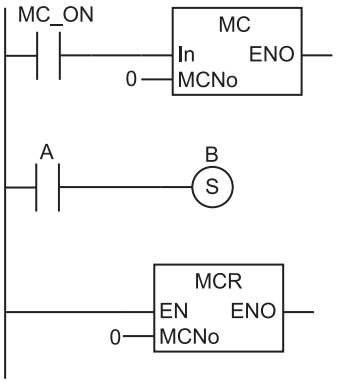


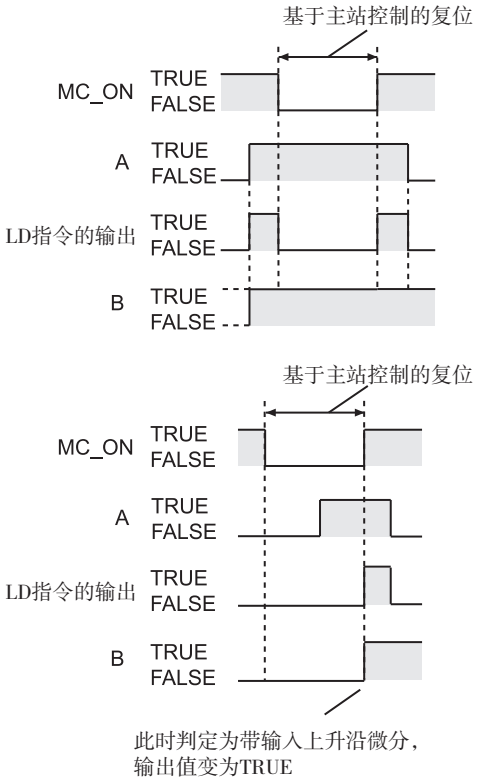


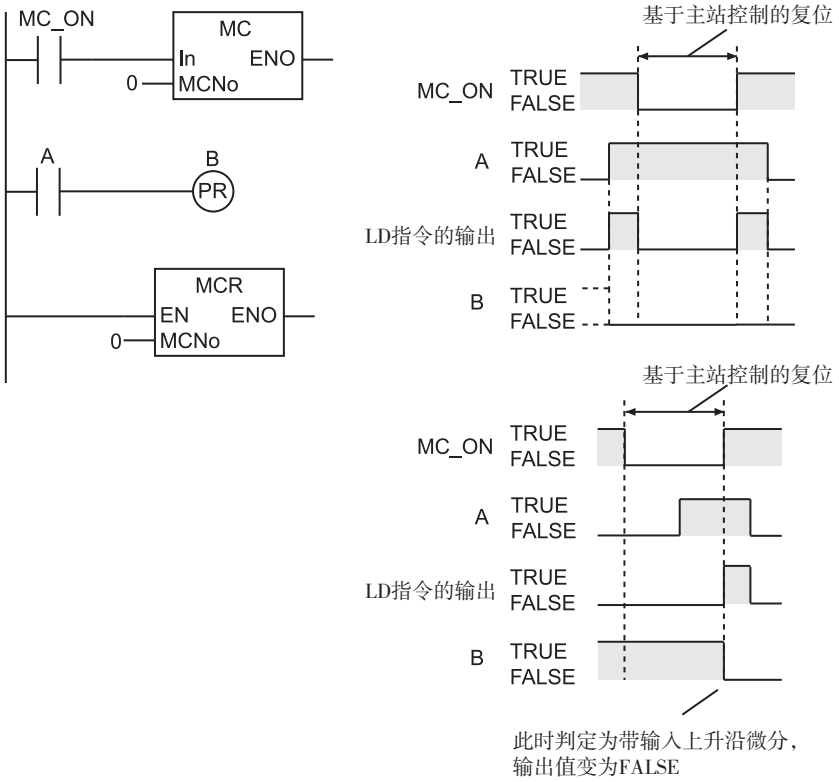
● 带输入上升沿微分指定的Set/Reset

执行基于主站控制的复位时，保持此前的输出值。

解除基于主站控制的复位时，在执行条件从FALSE变为TRUE时，进行动作。

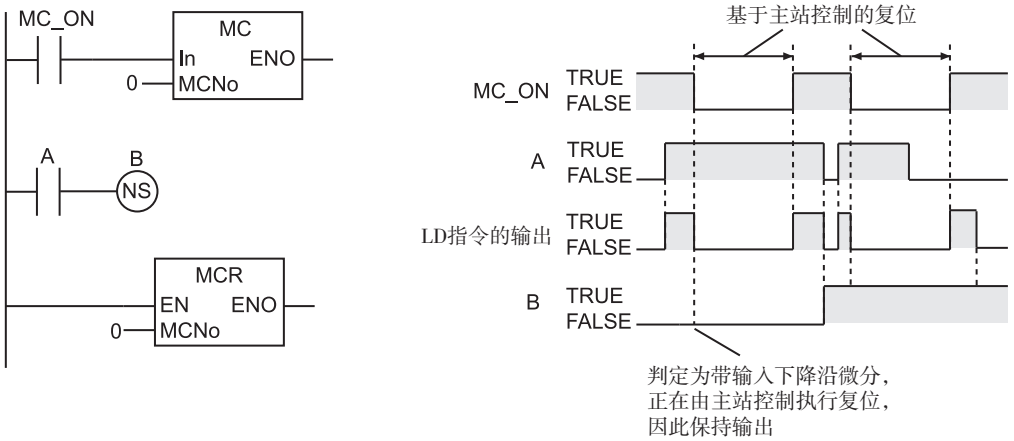


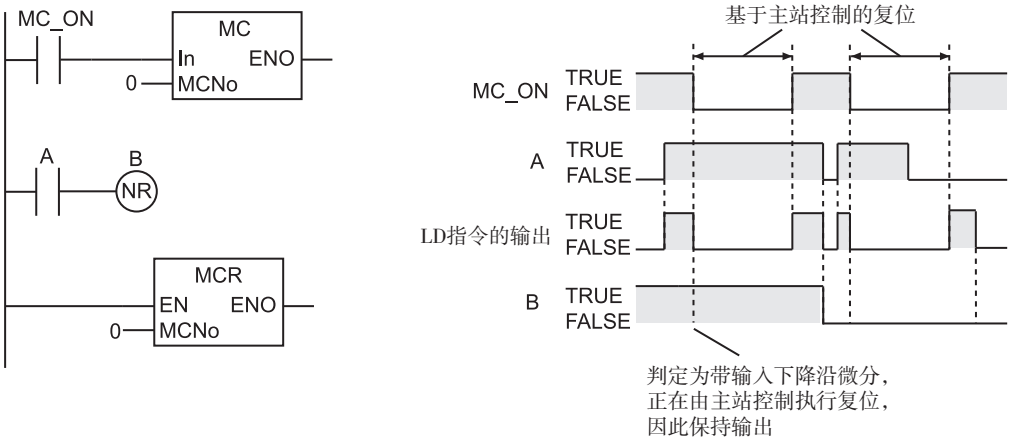




● 带输入下降沿微分指定的Set/Reset

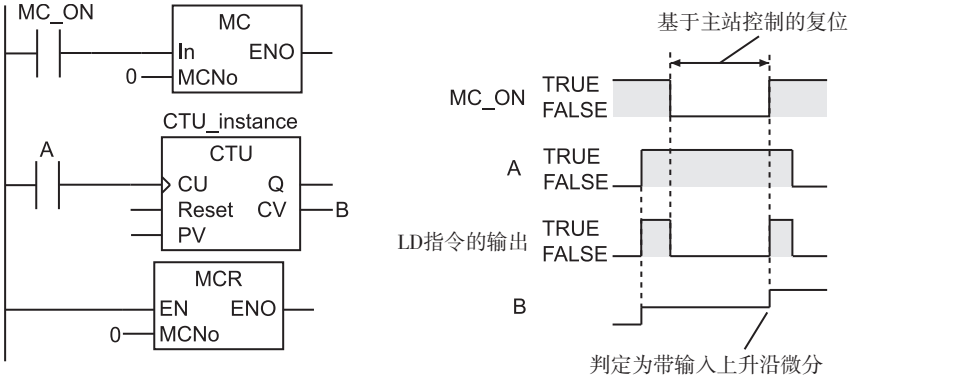
执行基于主站控制的复位时，执行条件的值会变为FALSE。如果此前的执行条件的值已为TRUE，则由此判定为发生了输入下降沿微分。但执行基于主站控制的复位时的输出值保持此前的值，因此最终保持输出值。





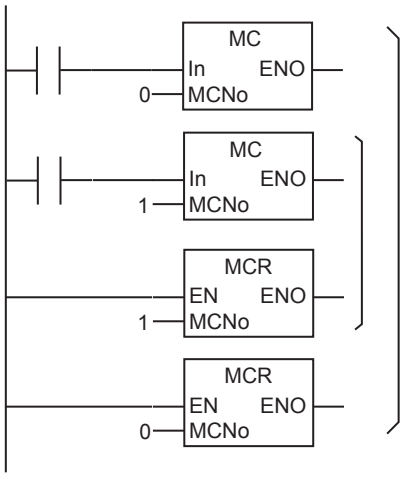
- CTU/CTD/CTUD

执行基于主站控制的复位时，计数器输入的值变为 FALSE。之后，解除基于主站控制的复位时，如果计数器输入的值为TRUE，则判定为发生了输入上升沿微分，进行计数。

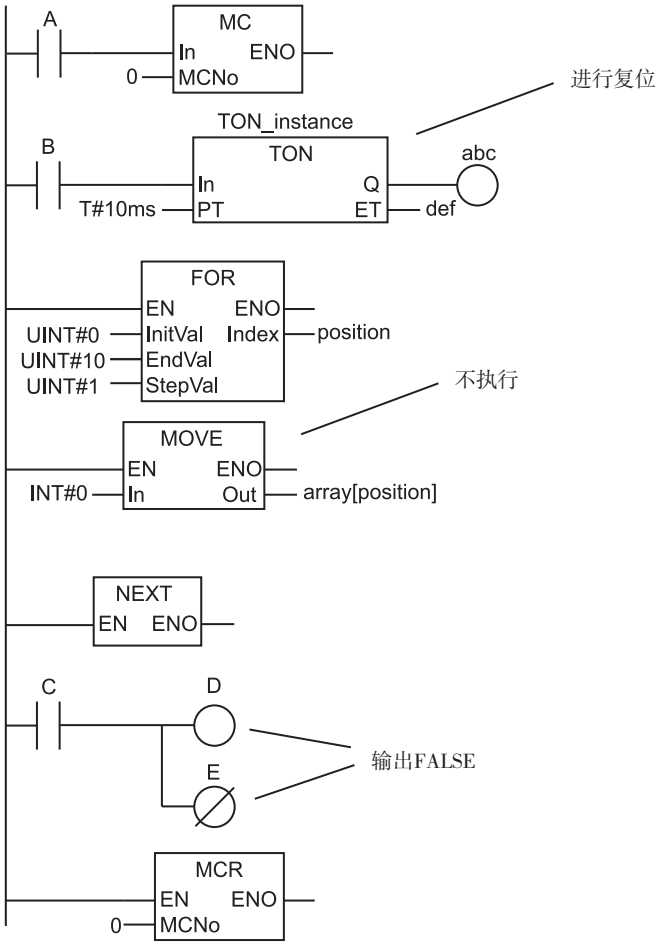


MC指令和MCR指令必须在相同POU内成对使用。为配对的MC指令和MCR指令分配的值与为主站控制No.“MCNo”分配的值相同。“MCNo”的值不是由用户任意决定，而是通过Sysmac Studio自动登录。

MC指令和MCR指令最多可嵌套15段。



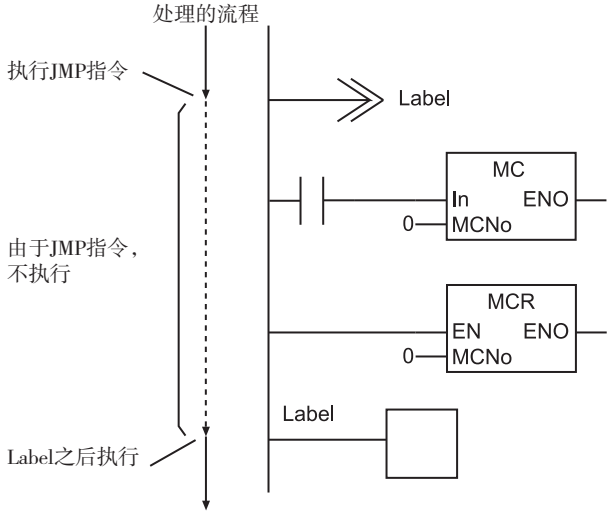
示例如下所示。
变量A的值变为FALSE时，执行基于主站控制的复位。在此期间，TON指令复位。不执行MOVE指令。Out指令和OutNot指令均向变量D和E输出FALSE。



使用注意事项

- 本指令仅可用于梯形图程序。无法用于ST程序。也无法用于梯形图程序中的内联ST。
- “In”请务必连接至左母线。“In”无法跨越变量或常数。
- MC指令和MCR指令请务必在相同POU内成对使用。
- MCR指令请务必配置在MC指令之后。
- MC指令和MCR指令无法嵌套16段以上。
- 如果主站控制区域含有内联ST，则执行基于主站控制的复位时，不执行该内联ST的描述。

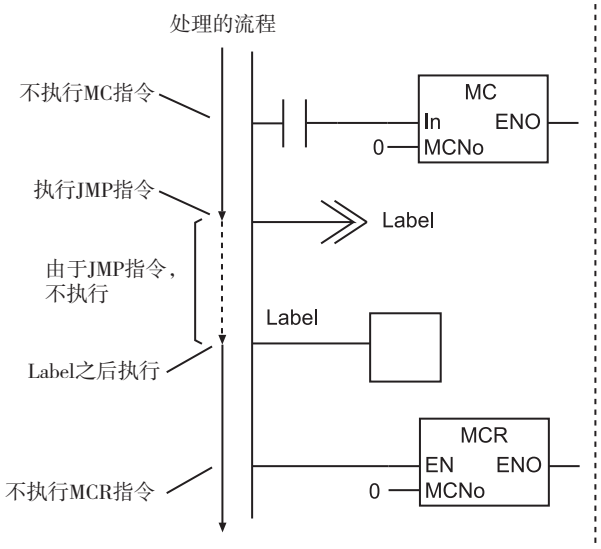
- 同时使用MC/MCR指令和JMP指令时的动作如下所示。
  - 下图中，JMP–Label的内侧有MC–MCR。因此，与 “In”的值无关，均执行跳转。



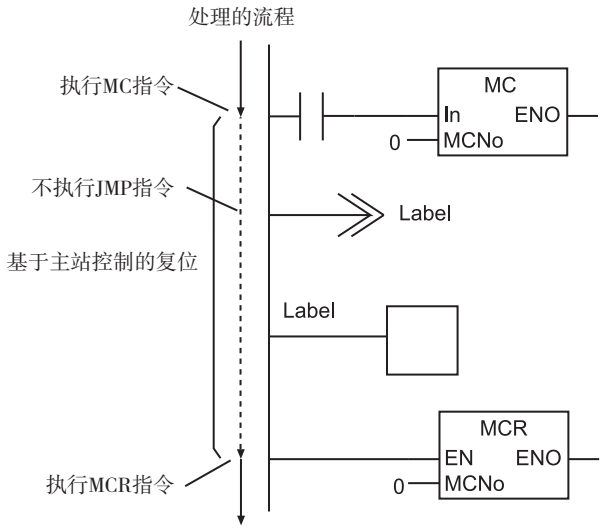
• 下图中，MC–MCR的内侧有JMP–Label。因此，动作如下表所示。

“In”的值	动作
TRUE	不执行基于主站控制的复位。 执行跳转。
FALSE	执行基于主站控制的复位。 不执行跳转。

● “In” = TRUE时

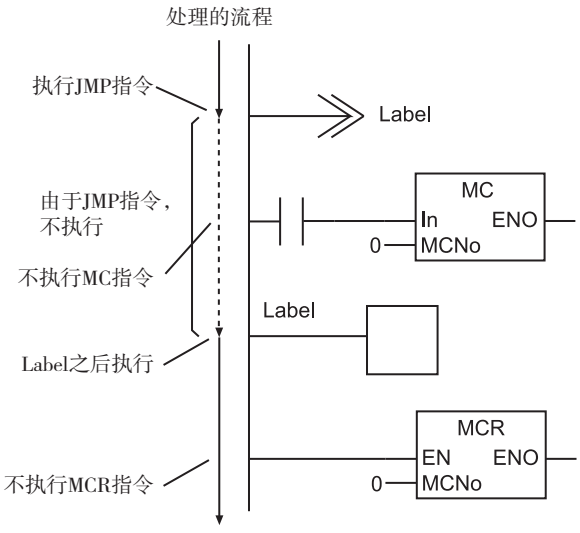


● “In” = FALSE时

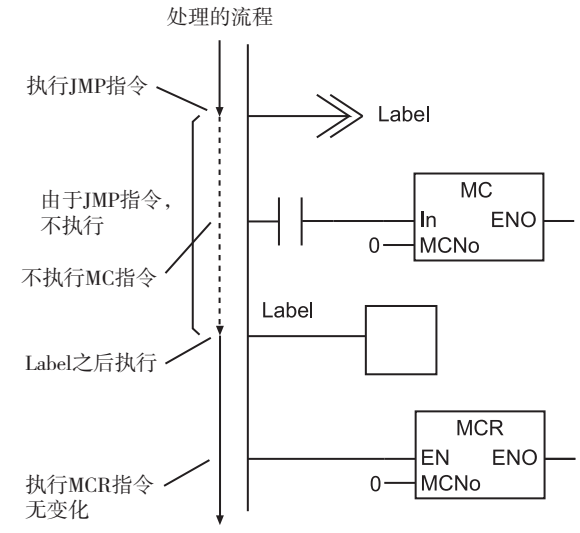


• 下图中，以JMP指令、MC指令、Label、MCR指令的顺序依次排列。因此，首先执行跳转。最终，不执行MC指令。而执行Label之后的内容。“In”的值变为FALSE时，执行MCR指令，但值不变。

● “In” = TRUE时



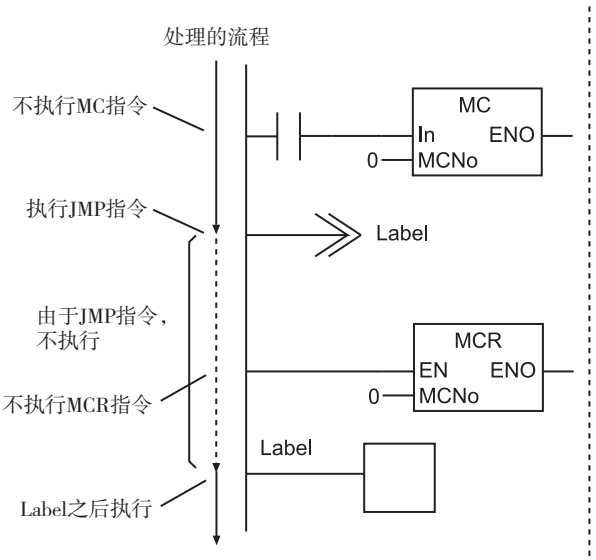
● “In” = FALSE时



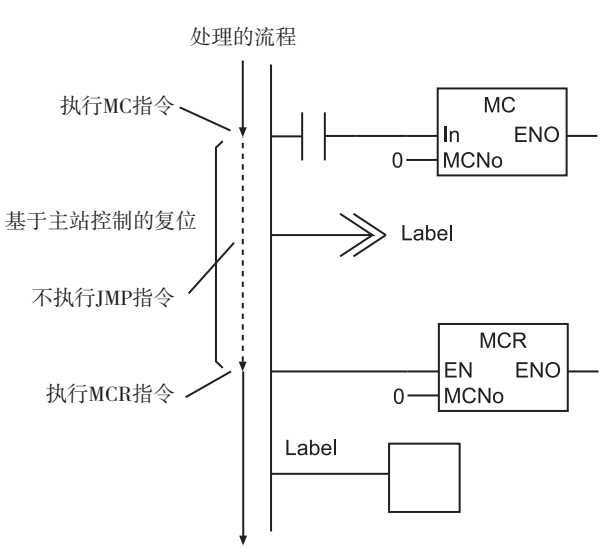
• 下图中，以MC指令、JMP指令、MCR指令、Label的顺序依次排列。因此，动作如下表所示。

“In”的值	动作
TRUE	不执行基于主站控制的复位。 执行跳转。
FALSE	执行基于主站控制的复位。 不执行跳转。

● “In” = TRUE时



● “In” = FALSE时

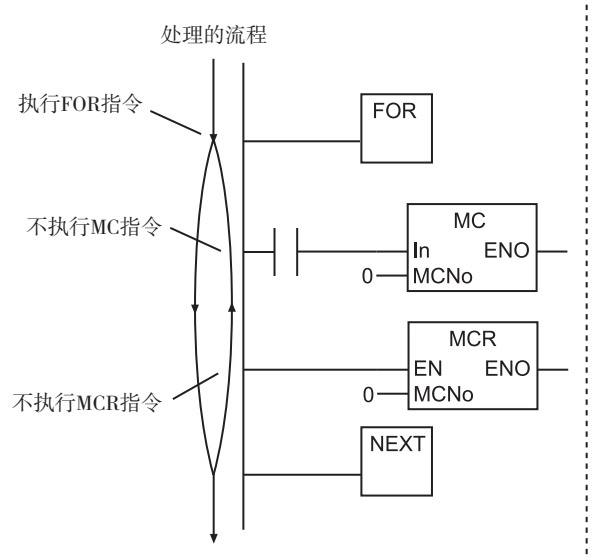


• 同时使用MC/MCR指令和FOR/NEXT指令时的动作如下所示。

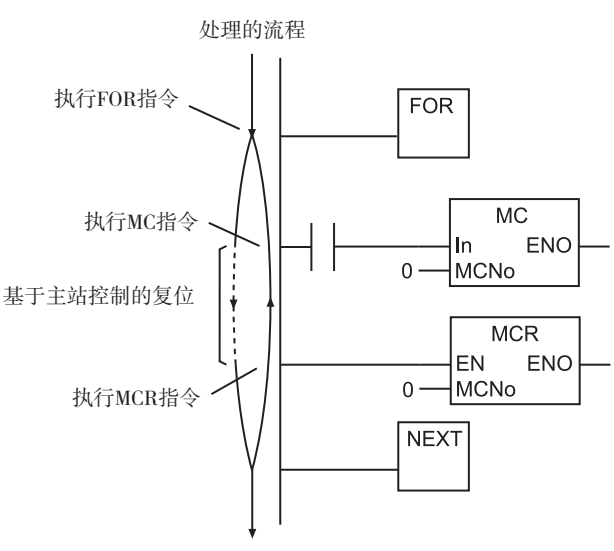
• 下图中，FOR–NEXT的内侧有MC–MCR。因此，动作如下表所示。

“In”的值	动作
TRUE	不执行基于主站控制的复位。 执行FOR循环。
FALSE	执行基于主站控制的复位。 执行FOR循环，但不执行MC–MCR内的内容。

● “In” =TRUE时



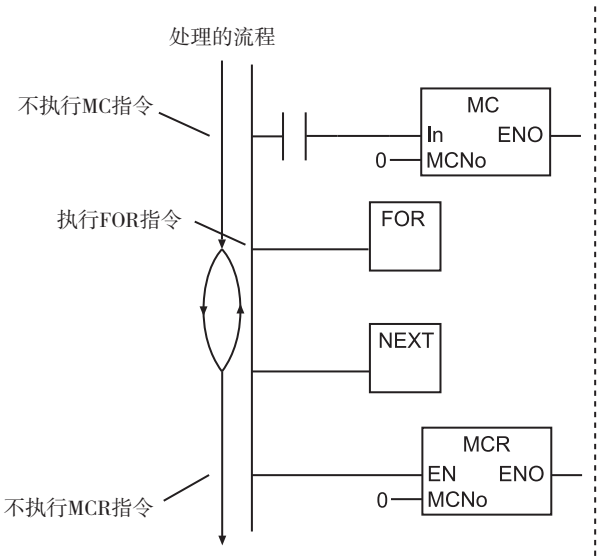
● “In” = FALSE时



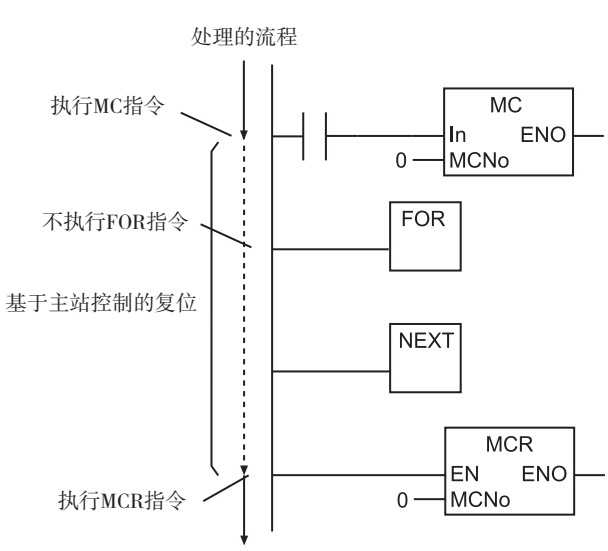
• 下图中，MC–MCR内有FOR–NEXT。因此，动作如下表所示。

“In”的值	动作
TRUE	不执行基于主站控制的复位。 执行FOR循环。
FALSE	执行基于主站控制的复位。 不执行FOR循环。

● “In” = TRUE时



● “In” = FALSE时



• 如下所示，如果FOR指令、NEXT指令、MC指令、MCR指令间的配置不同，则编连时会发生异常。
FOR、MC、NEXT、MCR的顺序或MC、FOR、MCR、NEXT的顺序

JMP

处理转移至指定的跳转位置。

指令	名称	FB/FUN	图形表现	ST表现
JMP	跳转	FUN	Label	无

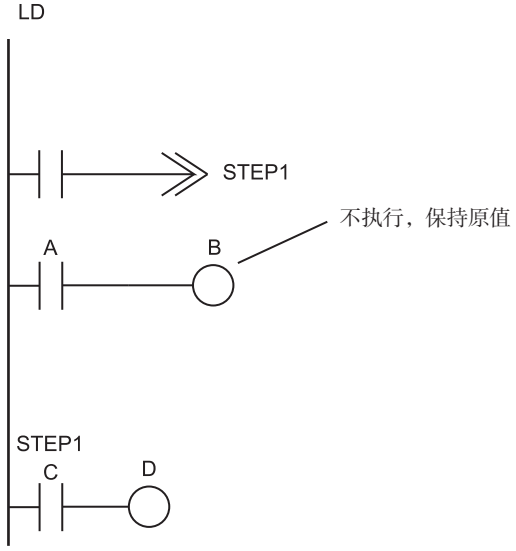
变量

无

功能

输入条件为TRUE时，处理转移至梯形图程序中由Label指定的跳转位置。Label为任意字符串。

示例如下所示。本示例中，使用字符串“STEP1”作为Label。执行JMP指令时，处理转移至记述“STEP1”的位置。此时，不执行JMP指令与Label之间的Out指令，变量B保持原值。



参考

- 也可向段的上方跳转。
- 多个JMP指令也可将相同Label指定为跳转位置。

使用注意事项

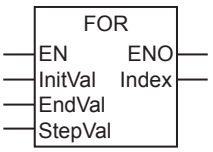
- 无法省略Label。如果省略，编连时会发生异常。
- JMP指令和Label请设定在同一段内。
- 同一段内无法多次设定相同Label。
- 无法从FOR-NEXT循环的外部跳转至内部。
- 可用作Label的字符的限制如下所示。

项目	规格
最大字节数	127字节 以ASCII码换算为127个字符 按中文换算为31个字
文字代码	UTF-8
可使用的字符	不区分大小写 英数字、各国语言文字 符号：_(下划线)、~(波浪号)
无法同时使用的字符	• 以ASCII码的数字0 ~ 9(文字代码16#30 ~ 16#39)开头的字符串。 • 仅有ASCII码的 “_”(下划线)的1个字符的字符串。 • 含有2个以上连续的ASCII码的 “_”(下划线)的字符串。 • 首字符为ASCII码的 “_”(下划线)的字符串。 • 尾字符为ASCII码的 “_”(下划线)的字符串。 • 开头2个字符为 “P_”的字符串。
禁用字符	空格、!(感叹号)、"(双引号)、#、$、&、'(单引号)、(、)、*、+、','(逗号)、-、'.'(句号)、/、:(冒号)、;(分号)、<、=、>、?、@、[、]、^、`'(反引号)、%

- 无法将变量名称指定为Label。

FOR/NEXT

FOR ：表示重复处理的起始位置，同时指定重复的条件。
NEXT ：表示重复处理的终止位置。

指令	名称	FB/FUN	图形表现	ST表现
FOR	重复开始	FUN		FOR Index：= InitVal TO EndVal BY StepVal DO 公式； END_FOR*； *ST语言表示重复处理的终止位置时，不使用 “NEXT” 而使用 “END_FOR”。
NEXT	重复结束	FUN		

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InitVal	初始值	输入	重复开始时，为 “Index” 设置的值	遵从数据类型(*1)	–	(*2)
EndVal	最终值		重复结束时 “Index” 的值			
StepVal	增减值		每次重复处理时，加上 “Index” 的值	遵从数据类型(*3)		(*4)
Index	控制变量	输出	循环索引	遵从数据类型	–	–

*1 若为梯形图程序，请设定为 “InitVal” < “EndVal”。
*2 省略输入参数时，初始值不适用。编译时会发生异常。
*3 若为梯形图程序，则不含0和负数。若为ST程序，则不含0。
*4 若为梯形图程序，即使省略输入参数，也不适用初始值。编译时会发生异常。若为ST程序，省略输入参数时，可适用初始值 “1”。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InitVal						○	○	○	○	○	○	○	○							
	也可指定枚举体、数组的1个元素、结构体的1个结构要素(*)																			
EndVal						○	○	○	○	○	○	○	○							
	也可指定数组的1个元素、结构体的1个结构要素																			
StepVal						○	○	○	○	○	○	○	○							
	也可指定数组的1个元素、结构体的1个结构要素																			
Index						○	○	○	○	○	○	○	○							
	也可指定数组的1个元素、结构体的1个结构要素																			

* 梯形图语言时，无法指定枚举体。

功能

重复进行FOR至NEXT(ST语言时为FOR至END_FOR)的处理。
FOR/NEXT的处理步骤如下所示。

1 为控制变量 “Index”设置 “InitVal”的值。

2 “StepVal”、“InitVal”、“EndVal”的值可判定下表中的条件是否满足。如果条件满足，则进至3。如果条件不满足，则不进行重复处理，转移至NEXT指令(ST语言时为END_FOR)的下一处理。

编程语言	开始重复处理的条件
梯形图	“StepVal” ≥0 且 “InitVal” < “EndVal”
ST	“StepVal” ≥0 且 “InitVal” ≤ “EndVal”
	“StepVal” <0 且 “InitVal” ≥ “EndVal”

3 “Index”、“EndVal”的值可判断下表中的条件是否满足。如果条件满足，则进至4。如果条件不满足，则结束重复处理，转移至NEXT指令(ST语言时为END_FOR)的下一处理。

编程语言	继续重复处理的条件
梯形图	“Index” ≤ “EndVal”
ST	“StepVal” ≥0时，“Index” ≤ “EndVal”
	“StepVal” <0时，“Index” ≥ “EndVal”

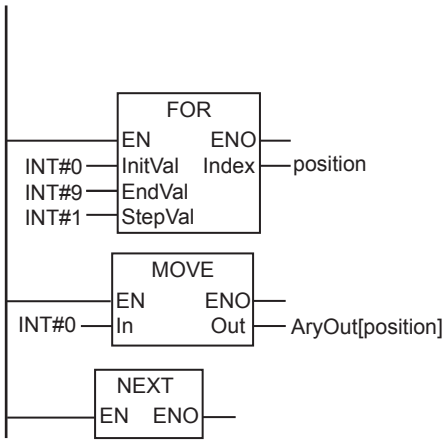
4 执行1次FOR指令与NEXT指令(ST语言时为END_FOR)之间的处理。

5 将 “StepVal”的值加上 “Index”。

6 返回3。

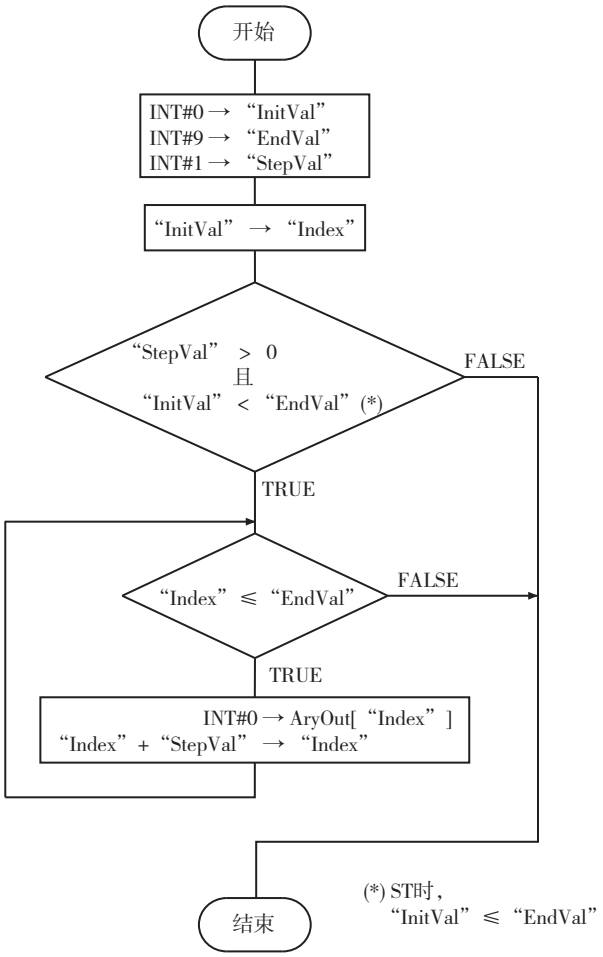
“InitVal” = INT#0、“EndVal” = INT#9、“StepVal” = INT#1时的示例如下所示。执行10次MOVE指令，在排列变量AryOut[0] ~ [9]中代入值INT#0。

LD



ST

```
FOR position:=INT#0 TO INT#9 BY INT#1 DO
    AryOut[position]:=INT#0;
END_FOR;
```



在AryOut[0] ~ AryOut[9]中
依次代入INT#0。

AryOut[0]	INT#0
AryOut[1]	INT#0
AryOut[2]	INT#0
AryOut[3]	INT#0
AryOut[4]	INT#0
AryOut[5]	INT#0
AryOut[6]	INT#0
AryOut[7]	INT#0
AryOut[8]	INT#0
AryOut[9]	INT#0

输入变量使用算术表达式及函数的ST程序的记述示例

在ST程序中使用本指令时，可在 “InitVal”、“EndVal”、“StepVal”的输入变量中作如下记述。

- 运算结果为整数值的算术表达式
- 返回整数值的函数
- 返回枚举元素的函数

用函数记述 “EndVal”，用算术表达式记述 “StepVal”的示例如下所示。

```
A := DINT#1;
B := DINT#2;
C := REAL#9.6;
FOR i := 0 TO RoundUp(C) BY A+B DO
   DINTArray[i] := i;
END_FOR;
```

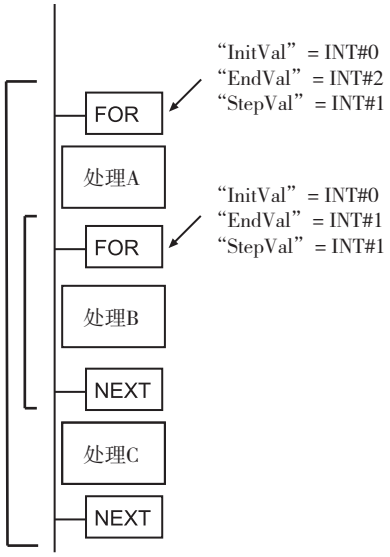
版本相关信息

Sysmac Studio Ver.1.08以上版本可在 “EndVal”、“StepVal”中记述算术表达式。
即使是Sysmac Studio Ver.1.07以下版本，也可在 “InitVal”中记述算数表达式。

参考

- 请执行BREAK指令(ST语言时为EXIT)，以中断重复处理。此时，不执行BREAK指令至NEXT指令的处理。
- 也可在FOR至NEXT(ST语言时为FOR至END_FOR)之间进一步配置(嵌套)FOR/NEXT指令。处理步骤如下图所示。

处理A→处理B→处理B→处理C→处理A→处理B→处理B→处理C→
处理A→处理B→处理B→处理C



使用注意事项

- 通过梯形图程序使用本指令时，请务必将FOR指令和NEXT指令直接连接至左母线。
- FOR指令和NEXT指令(ST语言时为FOR和END_FOR)请务必成对使用。两者的数量不同时，程序会发生异常。
- 配对的FOR指令和NEXT指令请设定在相同段内。
- 请充分注意设定重复终止条件，以避免无限进行重复处理(无限循环)。变为无限循环时，会发生任务执行超时。

  (例)　各变量的输入参数值为下表中的值时，SINT型的最大值为127，因此 “Index”的值不会大于“EndVal”的值。因此，变为无限循环。

  “EndVal”中请勿设定数据类型的最大值。

变量	输入参数值
“InitVal”	SINT#0
“EndVal”	SINT#127
“StepVal”	SINT#1
“Index”	–

- 根据 “StepVal”、“InitVal”、“EndVal”的值，动作如下所示。

编程语言	“StepVal”的值	“InitVal”、“EndVal”的值	动作
梯形图	“StepVal”>0	“InitVal” < “EndVal”	正常动作。
		“InitVal” ≥ “EndVal”	一次也不执行FOR指令与NEXT指令间的处理。 未发生异常。
	“StepVal”<0	“InitVal” < “EndVal”	不定次数执行FOR指令与NEXT指令间的处理。 请勿如此设定。 未发生异常。
		“InitVal” ≥ “EndVal”	一次也不执行FOR指令与NEXT指令间的处理。 未发生异常。
	“StepVal”=0	“InitVal” < “EndVal”	变为无限循环，会发生任务执行超时。
		“InitVal” ≥ “EndVal”	一次也不执行FOR指令与NEXT指令间的处理。 未发生异常。
ST	“StepVal”>0	“InitVal” ≤ “EndVal”	正常动作。
		“InitVal” > “EndVal”	一次也不执行FOR指令与END_FOR指令间的处理。 未发生异常。
	“StepVal”<0	“InitVal” < “EndVal”	一次也不执行FOR指令与END_FOR指令间的处理。 未发生异常。
		“InitVal” ≥ “EndVal”	正常动作。
	“StepVal”=0	“InitVal” ≤ “EndVal”	变为无限循环，会发生任务执行超时。
		“InitVal” > “EndVal”	一次也不执行FOR指令与END_FOR指令间的处理。 未发生异常。

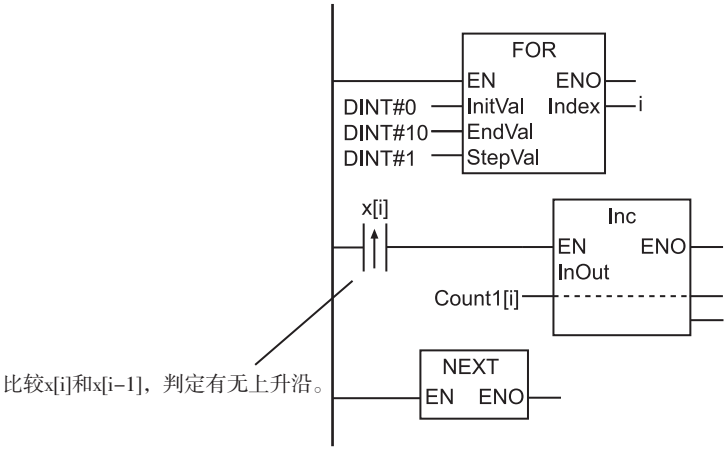
- 嵌套(FOR/NEXT指令的分层构造)的深度含IF指令、CASE指令、FOR指令、WHILE指令、REPEAT指令在内，最多15层。
- 嵌套时如需中断重复处理，仅需按层次数量执行BREAK指令(ST语言时为EXIT)。
- 请勿使用跳转类指令(JMP指令等)来中断重复处理。请务必使用BREAK指令(ST语言为EXIT)来中断重复处理。
- 重复处理中变更 “InitVal”、“EndVal”、“StepVal”的值时的动作如下所示。

变量	动作	
	梯形图语言	ST语言
“InitVal”	重复处理结束前，变更后的值不会反映。	重复处理结束前，变更后的值不会反映。
“EndVal”	即使在重复处理过程中，变更后的值也会反映。	
“StepVal”	有时会意外动作。请勿变更重复处理中的值。	

- 请将 “InitVal”、“EndVal”、“StepVal”、“Index”全部设为相同的数据类型。否则，编连时会发生异常。
- 请将 “Index”的数据类型设定为包含 “InitVal”、“EndVal”、“StepVal”的有效范围。否则，编连时会发生异常。
- 梯形图语言和 ST 语言在重复处理结束后的 “Index”的值不同。梯形图语言最后不将 “StepVal”加上 “Index”即结束。ST语言最后将 “StepVal”加上 “Index”后结束。重复次数无差异。
  (例) “InitVal”=1、“EndVal”=100、“StepVal”=1时
  梯形图语言：重复次数为100次，重复结束后的 “Index”的值为100
  ST语言 ：重复次数为100次，重复结束后的 “Index”的值为101
- 在LD程序的FOR循环中，带上升沿微分指定(或下降沿微分指定)并与数组同时使用LD指令(或AND指令、OR指令)时，需注意。

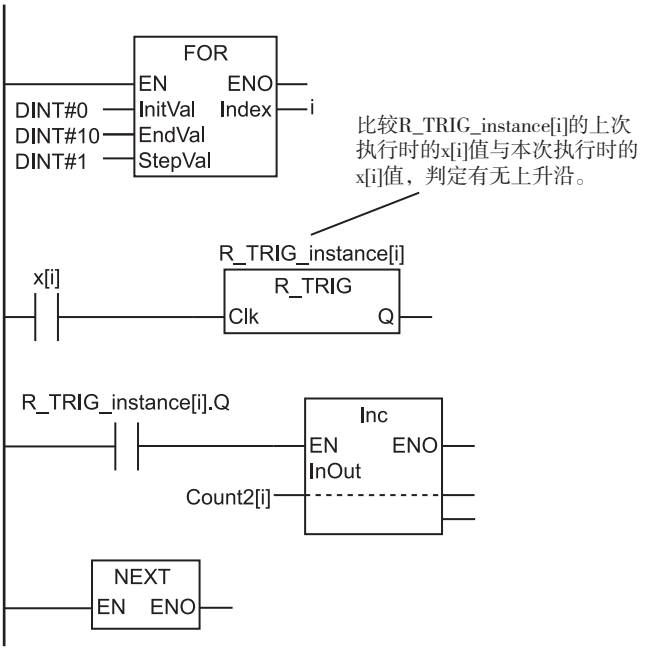
上升沿微分指定比较该指令上次执行时的指定变量值与本次执行时的指定变量值，判定有无上升沿。通常，每次指令执行时指定变量均不会变化，但在FOR循环中且指定数组时，每次执行指令时数组元素均会发生变化。因此，通过比较数组元素的值，判定有无上升沿。
以下用户程序时，x[i−1]的值、x[i]的值及对Count1 [i]的增量处理之间的关系如下表所述。

x[i−1]的值	x[i]的值	对Count1[i]的增量处理
TRUE	TRUE	不执行。
TRUE	FALSE	不执行。
FALSE	TRUE	执行。
FALSE	FALSE	不执行。



- 另一方面，使用以下程序时，通过R_TRIG指令检测x[i]有无上升沿。设有与x[i]各要素对应的R_TRIG实例，因此可根据x[i]各要素值的变化判定有无上升沿。因此，上一次执行R_TRIG_instance[i]时x[i]的值、本次执行R_TRIG_instance[i]时x[i]的值、对Count2[i]的增量处理之间的关系如下表所述。

上一次执行R_TRIG_instance[i]时x[i]的值	本次执行R_TRIG_instance[i]时x[i]的值	对Count2[i]的增量处理
TRUE	TRUE	不执行。
TRUE	FALSE	不执行。
FALSE	TRUE	执行。
FALSE	FALSE	不执行。



BREAK

中断最内侧FOR指令至NEXT指令的重复处理。

指令	名称	FB/FUN	图形表现	ST表现
BPEAK	循环中断	FUN	BREAK EN ENO	无

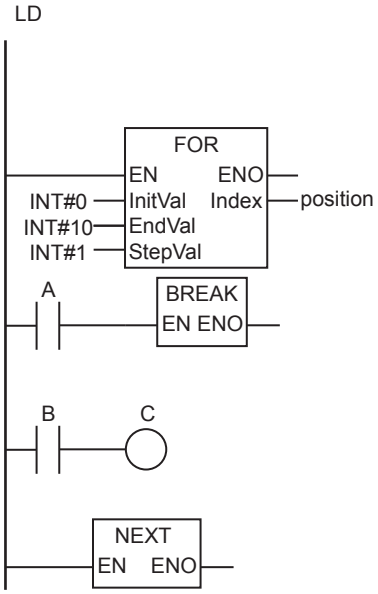
变量

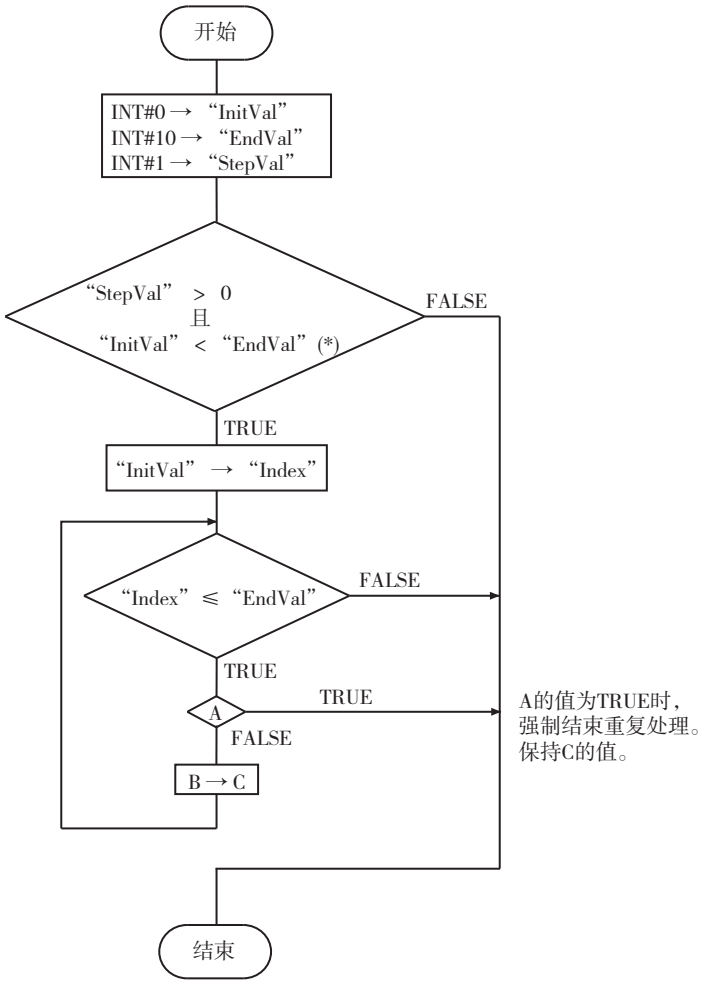
无

功能

输入条件为TRUE时，强制结束最内侧的FOR指令至NEXT指令(ST语言时为END_FOR)的重复处理，并转为NEXT指令的下一处理。
不执行BREAK指令(ST语言时为EXIT)至NEXT指令的处理。

示例如下所示。执行FOR循环的过程中，每次均确认变量A的值。变量A的值为TRUE时，强制结束重复处理。
此时，不执行BREAK指令(ST语言时为EXIT)之后的Out指令，变量C保持原值。





使用注意事项

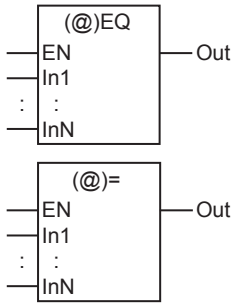
- 本指令请务必在FOR指令与NEXT指令(ST语言时为FOR与END_FOR)之间使用。
- 分层使用FOR指令和NEXT指令(ST语言时为FOR与END_FOR)时，仅需按层次数量执行BREAK指令(ST语言时为EXIT)以完全中断重复处理。
- 请勿使用跳转类指令(JMP指令等)来中断重复处理。请务必使用BREAK指令(ST语言为EXIT)来中断重复处理。

比较指令

指令	名称	页码
EQ,=	比较EQ	2–94
NE,<>	比较NE	2–96
LT,</LE,<=/GT,>/GE,>=	比较LT/比较LE/比较GT/ 比较GE	2–98
EQascii	字符串比较EQ	2–101
NEascii	字符串比较NE	2–103
LTascii/LEascii/GTascii/GEascii	字符串比较LT/ 字符串比较LE/ 字符串比较GT/ 字符串比较GE	2–105
Cmp	比较	2–108
ZoneCmp	区域比较	2–110
TableCmp	表比较	2–113
AryCmpEQ/AryCmpNE	数组整体比较EQ/ 数组整体比较NE	2–116
AryCmpLT/AryCmpLE/ AryCmpGT/AryCmpGE	数组整体比较LT/ 数组整体比较LE/ 数组整体比较GT/ 数组整体比较GE	2–118
AryCmpEQV/AryCmpNEV	数组元素比较EQ/ 数组元素比较NE	2–121
AryCmpLTV/ AryCmpLEV/ AryCmpGTV/ AryCmpGEV	数组元素比较LT/ 数组元素比较LE/ 数组元素比较GT/ 数组元素比较GE	2–123

EQ，=

判定多个数据是否全部相等。

指令	名称	FB/FUN	图形表现	ST表现
EQ,=	比较EQ	FUN		Out:= (In1=In2) & (In2=In3) & ⋯ & (InN−1=InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	比较对象	输入	要比较的数值　N为2 ~ 5	遵从数据类型	−	0*1
Out	比较结果	输出	比较结果	遵从数据类型	−	−

*1 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

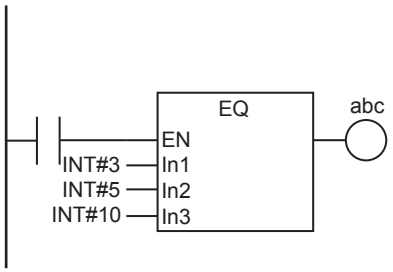
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	枚举体也可指定															*1				
Out	○																			

*1 TIME型、DATE型、TOD型、DT型、STRING型在Ver.1.02以上的Sysmac Studio中可指定。使用Ver.1.02以上的Sysmac Studio打开Ver.1.01以下的Sysmac Studio编制的程序，并使用上述的数据类型时，请更新显示。更新显示时，请右击编辑窗口的指令，选择[表示更新]。如果不更新显示，编连时会出现异常。

功能

判定2个以上5个以下的数据 “In1” ~ “InN”是否全部相等。
仅全部相等时，比较结果 “Out”的值变为TRUE。其余情况下，“Out”的值均为FALSE。
比较STRING型时，相等是指字符串长度和内容均相同。
“In1”=INT#3、“In2”=INT#5、“In3” = INT#10时的示例如下所示。变量abc的值为FALSE。

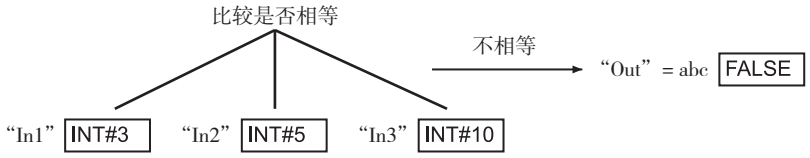
LD



ST

```
abc:=(INT#3=INT#5)&(INT#5=INT#10);
```

判定“In1” ~ “In3”是否全部相等。
由于不相等，因此abc的值为FALSE。



参考

- EQ和=是功能完全相同的指令。请采用易于使用的方法。
- 比较TIME型、DT型、TOD型时，请符合要比较的值的精度。有 “TruncTime指令(P.2–642)”、“TruncDt指令(P.2–646)”、“TruncTod指令(P.2–650)”，以符合值的精度。

使用注意事项

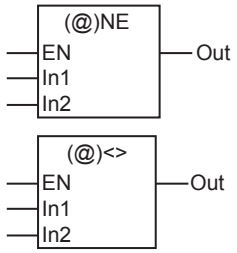
- “In1” ~ “InN”的数据类型不同时，将类型扩展为包括所有数据类型的有效范围在内的数据类型后，再进行比较。
- 位串型(BYTE，WORD，DWORD，LWORD)和整数型(SINT，INT，DINT，LINT，USINT，UINT，UDINT，ULINT)无法比较。位串型和实数型(REAL,LREAL)也无法比较。
- 带符号整数型(SINT，INT，DINT，LINT)和无符号整数型(USINT，UINT，UDINT，ULINT)无法比较。
- TIME型、DATE型、TOD型、DT型、STRING型仅可在相同的数据类型之间进行比较。如果指定了不同的数据类型，则编连时会出现异常。
- 枚举体和其他的类型无法比较。枚举体之间如果类型不一致也无法比较。
- +∞之间或–∞之间判断为相等。
- “In1” ~ “InN”中的任一值为非数时，“Out”的值为FALSE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- “In1” ~ “InN”为实数时，由于取整误差，处理结果可能会出现意外。要验证是否相等的值中有一者为实数时，请勿使用本指令，而是使用大小比较指令验证两者之差的绝对值是否小于容许范围。例如，验证REAL型变量real_a与real_b之和是否等于0.1时，请作如下记述。BOOL型变量boolv的值为TRUE时，则判断为两者相等。

```
boolv := (ABS((real_a + real_b) – 0.1) < threshold);
```

threshold：容许范围的值

NE，<>

判定2个数据是否不同。

指令	名称	FB/FUN	图形表现	ST表现
NE,<>	比较NE	FUN		Out:= (In1 <> In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1,In2	比较对象	输入	要比较的数值	遵从数据类型	–	*1(*)
Out	比较结果	输出	比较结果	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。

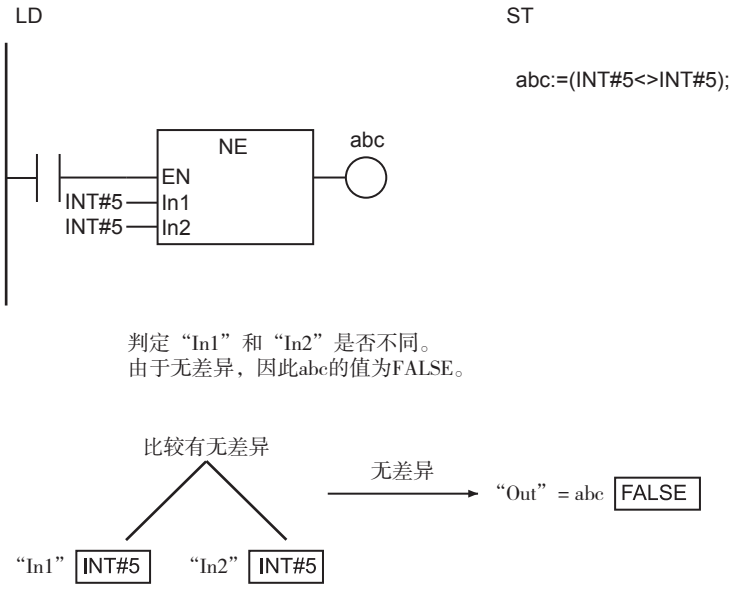
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1,In2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	枚举体也可指定															*1				
Out	○																			

*1 TIME型、DATE型、TOD型、DT型、STRING型在Ver.1.02以上的Sysmac Studio中可指定。使用Ver.1.02以上的Sysmac Studio打开Ver.1.01以下的Sysmac Studio编制的程序，并使用上述的数据类型时，请更新显示。更新显示时，请右击编辑窗口的指令，选择[表示更新]。如果不更新显示，编连时会出现异常。

功能

判定2个数据 “In1” 和 “In2” 是否不同。
如果不同，则比较结果 “Out” 为TRUE；如果相同，则为FALSE。
比较STRING型时，相等是指字符串长度和内容均相同。

“In1” = “In2” = INT#5时的示例如下所示。变量abc的值为FALSE。



参考

- 指令的功能与NE指令<>完全相同。请采用易于使用的方法。
- 比较TIME型、DT型、TOD型时，请符合要比较的值的精度。有 “TruncTime指令(P.2–642)”、“TruncDt指令(P.2–646)”、“TruncTod指令(P.2–650)”，以符合值的精度。

使用注意事项

- “In1” 和 “In2” 的数据类型不同时，将类型扩展为包括两者数据类型有效范围在内的数据类型后，再进行比较。
- 位串型(BYTE，WORD，DWORD，LWORD)和整数型(SINT，INT，DINT，LINT，USINT，UINT，UDINT，ULINT)无法比较。位串型和实数型(REAL，LREAL)也无法比较。
- 带符号整数型(SINT，INT，DINT，LINT)和无符号整数型(USINT，UINT，UDINT，ULINT)无法比较。
- TIME型、DATE型、TOD型、DT型、STRING型仅可在相同的数据类型之间进行比较。如果指定了不同的数据类型，则编连时会出现异常。
- 枚举体和其他的类型无法比较。枚举体之间如果类型不一致也无法比较。
- +∞之间或–∞之间判断为相等。
- “In1”、“In2” 中的任一值为非数时，“Out” 的值为TRUE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out” 的值为FALSE。
- “In1” ~ “InN” 为实数时，由于取整误差，处理结果可能会出现意外。要验证是否相等的值中有一者为实数时，请勿使用本指令，而是使用大小比较指令验证两者之差的绝对值是否小于容许范围。例如，验证REAL型变量real_a与real_b之和是否等于0.1时，请作如下记述。BOOL型变量boolv的值为TRUE时，则判断为两者相等。

```
boolv := (ABS((real_a + real_b) – 0.1) < threshold);
```

threshold：容许范围的值

LT，</LE，<=/GT，>/GE，>=

比较多个数值的大小。
LT,< ：比较(<)大小。
LE,<= ：比较(<=)大小。
GT,> ：比较(>)大小。
GE,>= ：比较(>=)大小。

指令	名称	FB/FUN	图形表现	ST表现
LT,<	比较LT	FUN	(@)LT EN In1 : InN Out (@)< EN In1 : InN Out	Out:= (In1<In2) & (In2<In3) & … & (InN−1<InN);
LE,<=	比较LE	FUN	(@)LE EN In1 : InN Out (@)<= EN In1 : InN Out	Out:= (In1<=In2) & (In2<=In3) & … & (InN−1<=InN);
GT,>	比较GT	FUN	(@)GT EN In1 : InN Out (@)> EN In1 : InN Out	Out:= (In1>In2) & (In2>In3) & … & (InN−1>InN);
GE,>=	比较GE	FUN	(@)GE EN In1 : InN Out (@)>= EN In1 : InN Out	Out:= (In1>=In2) & (In2>=In3) & … & (InN−1>=InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	比较对象	输入	要比较的数值 N为2 ~ 5	遵从数据类型	–	*1
Out	比较结果	输出	比较结果	遵从数据类型	–	–

*1 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
		*1														*1				
Out	○																			

*1 BYTE型、WORD型、DWORD型、LWORD型、TIME型、DATE型、TOD型、DT型、STRING型在Ver.1.02以上的Sysmac Studio中可指定。使用Ver.1.02以上的Sysmac Studio打开Ver.1.01以下的Sysmac Studio编制的程序，并使用上述的数据类型时，请更新显示。更新显示时，请右击编辑窗口的指令，选择[表示更新]。如果不更新显示，编连时会出现异常。

功能

比较2个以上5个以下的数据 “In1” ~ “InN”的大小。
各指令的输出 “Out”的值如下所示。

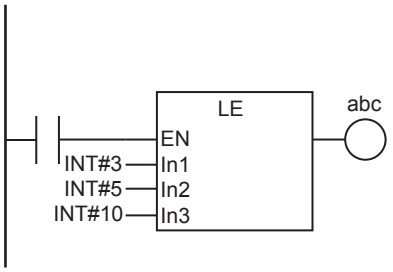
指令	“Out”的值
LT,<	“In1” < “In2” <…< “InN” 时为TRUE，其他情况时为FALSE
LE,<=	“In1” <= “In2” <=…<= “InN” 时为TRUE，其他情况时为FALSE
GT,>	“In1” > “In2” >…> “InN” 时为TRUE，其他情况时为FALSE
GE,>=	“In1” >= “In2” >=…>= “InN” 时为TRUE，其他情况时为FALSE

整数、实数以外的数据类型的值的大小关系的判断如下表所示。

数据类型	大小关系
BYTE、WORD、DWORD、LWORD	作为无符号整数判断。
TIME	值较大者判断为大。
DATE、TOD、DT	对于日期和时刻，较后者判断为大。
STRING	与 “LTascii/LEascii/GTascii/GEascii指令(P.2-105)” 的规格相同。请参阅该处。

LE指令下，“In1”= INT#3、“In2”= INT#5、“In3”= INT#10时的示例如下所示。变量abc的值为TRUE。

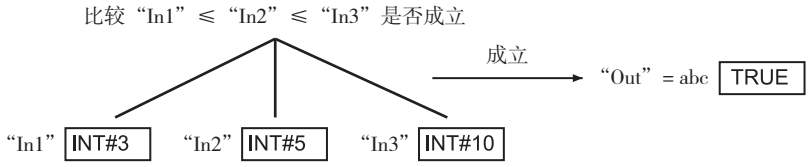
LD



ST

```
abc:=(INT#3<= INT#5)&(INT#5<=INT#10);
```

比较“In1”≤“In2”≤“In3”是否成立。
由于成立，因此abc的值为TRUE。



参考

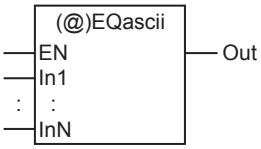
- LT与<、LE与<=、GT与>、GE与>=分别为功能完全相同的指令。请采用易于使用的方法。
- 比较TIME型、DT型、TOD型的大小时，请符合要比较的值的精度。有 “TruncTime指令(P.2-642)”、“TruncDt指令(P.2-646)”、“TruncTod指令(P.2-650)”，以符合值的精度。

使用注意事项

- “In1” ~ “InN”的数据类型不同时，将类型扩展为包括所有数据类型的有效范围在内的数据类型后，再进行比较。
- 带符号整数型(SINT，INT，DINT，LINT)和无符号整数型(USINT，UINT，UDINT，ULINT)无法比较。
- 位串型(BYTE，WORD，DWORD，LWORD)和整数型(SINT，INT，DINT，LINT，USINT，UINT，UDINT，ULINT)无法比较。位串型和实数型(REAL，LREAL)也无法比较。
- TIME型、DATE型、TOD型、DT型、STRING型仅可在相同的数据类型之间进行比较。如果指定了不同的数据类型，则编连时会出现异常。
- “In1” ~ “InN”为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- +∞之间或-∞之间判断为相等。
- “In1” ~ “InN”中的任一值为非数时，“Out”的值为FALSE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。

EQascii

判定多个字符串是否全部相等。

指令	名称	FB/FUN	图形表现	ST表现
EQascii	字符串比较EQ	FUN		Out:= EQascii(In1, …, InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	比较对象字符串	输入	要比较的字符串　N为2 ~ 5	遵从数据类型	–	''(*)
Out	比较结果	输出	比较结果	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

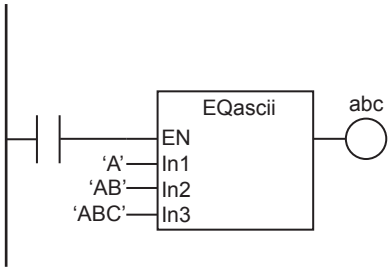
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN																				○
Out	○																			

功能

判定2个以上5个以下的字符串 “In1” ~ “InN” 是否全部相等。
仅全部相等时，比较结果 “Out” 的值变为TRUE。其余情况下，“Out” 的值均为FALSE。
相等是指长度和内容均相同。

“In1” ='A'、“In2” ='AB'、“In3” = 'ABC'时的示例如下所示。变量abc的值为FALSE。

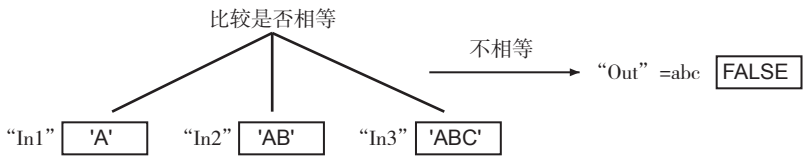
LD



ST

```
abc:=EQascii('A', 'AB', 'ABC');
```

判定“In1”～“In3”是否全部相等。
由于不相等，因此abc的值为FALSE。



参考

字符串比较的指令便于将多个字符串按照字符代码的顺序进行排列。例如，英文字母的字符代码的顺序与字母的顺序相同，因此可按照字母的顺序进行排列。

版本相关信息

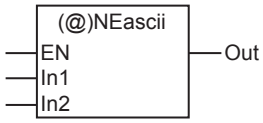
Ver.1.02以上的Sysmac Studio中，也有使用 “EQ，=指令(P.2–94)” 字符串比较的方法。
EQ，=指令的字符串比较和本指令的规格完全相同。

使用注意事项

- 本指令无法用于电路的末尾段。使用时，Sysmac Studio会发生异常，无法传送至控制器。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- “In1”～“InN”中请指定仅由ASCII码构成的字符串。

NEascii

判定2个字符串是否不同。

指令	名称	FB/FUN	图形表现	ST表现
NEascii	字符串比较NE	FUN		Out:= NEascii(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1,In2	比较对象字符串	输入	要比较的字符串	遵从数据类型	–	(*)
Out	比较结果	输出	比较结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

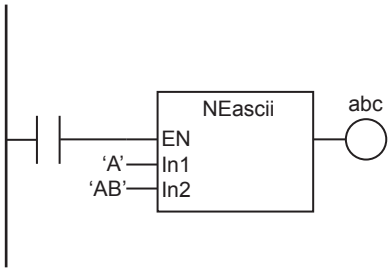
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1,In2																				○
Out	○																			

功能

判定2个字符串 “In1” 和 “In2” 是否不同。
如果不同，则比较结果 “Out” 为TRUE；如果相同，则为FALSE。
相等是指长度和内容均相同。

“In1” ='A'、“In2” ='AB'时的示例如下所示。变量abc的值为TRUE。

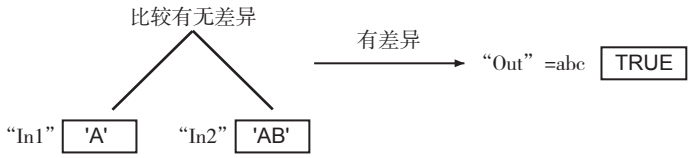
LD



ST

```
abc:=NEascii('A', 'AB');
```

判定“In1”和“In2”是否不同。
由于有差异，因此abc的值为TRUE。



参考

字符串比较的指令便于将多个字符串按照字符代码的顺序进行排列。例如，英文字母的字符代码的顺序与字母的顺序相同，因此可按照字母的顺序进行排列。

版本相关信息

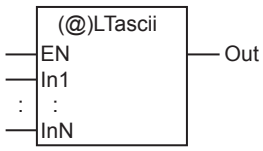
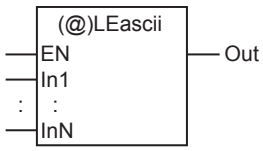
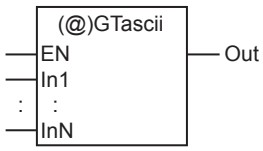
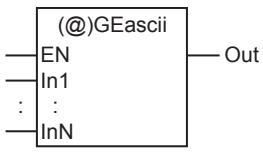
Ver.1.02以上的Sysmac Studio中，也有使用 “NE，<>指令(P.2-96)”字符串比较的方法。
NE，<>指令的字符串比较和本指令的规格完全相同。

使用注意事项

- 本指令无法用于电路的末尾段。使用时，Sysmac Studio会发生异常，无法传送至控制器。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- “In1”、“In2”中请指定仅由ASCII码构成的字符串。

LTascii/LEascii/GTascii/GEascii

比较多个字符串的大小。
LTascii ：比较(<)大小。
LEascii ：比较(<=)大小。
GTascii ：比较(>)大小。
GEascii ：比较(>=)大小。

指令	名称	FB/FUN	图形表现	ST表现
LTascii	字符串比较LT	FUN		Out:=LTascii(In1, …, InN);
LEascii	字符串比较LE	FUN		Out:=LEascii(In1, …, InN);
GTascii	字符串比较GT	FUN		Out:=GTascii(In1, …, InN);
GEascii	字符串比较GE	FUN		Out:=GEascii(In1, …, InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	比较对象字符串	输入	要比较的字符串 N为2 ~ 5	遵从数据类型	–	''(*)
Out	比较结果	输出	比较结果	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN																				○
Out	○																			

功能

比较2个以上5个以下的字符串 “In1” ~ “InN”的大小。
各指令的输出 “Out”的值如下所示。

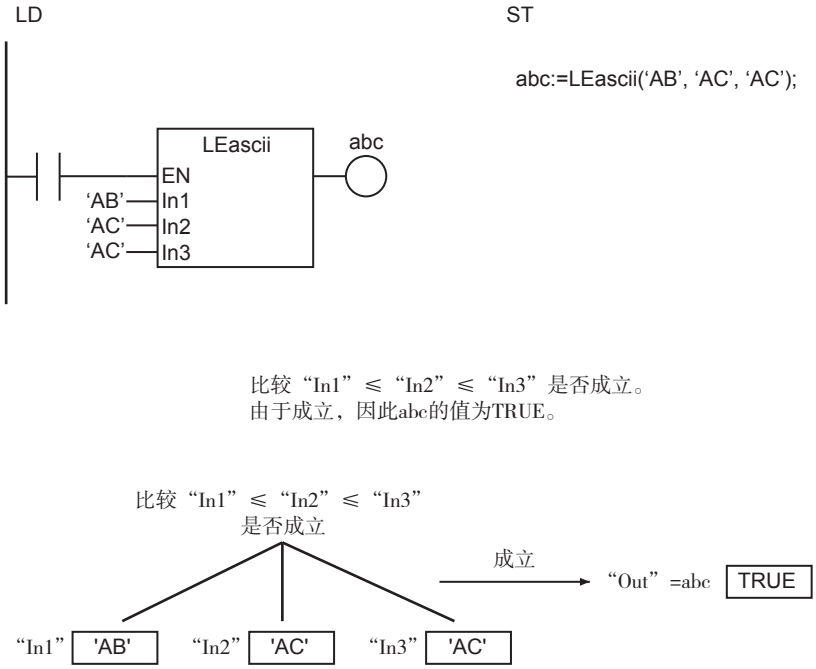
指令	“Out”的值
LTascii	“In1” < “In2” <…< “InN” 时为TRUE，其他情况时为FALSE
LEascii	“In1” <= “In2” <=…<= “InN” 时为TRUE，其他情况时为FALSE
GTascii	“In1” > “In2” >…> “InN” 时为TRUE，其他情况时为FALSE
GEascii	“In1” >= “In2” >=…>= “InN” 时为TRUE，其他情况时为FALSE

使用字符代码比较大小。比较步骤如下所示。
首先，比较各字符串的第1个字符的字符代码。如果字符代码不同，则该字符代码的大小为即为这些字符串的大小关系。
第1个字符的字符代码相同时，依次比较第2个字符、第3个字符，直至找到不同的字符代码为止。
各字符串的长度不同时，对照较长的字符串，在较短的字符串末尾添加NULL字符(16#00)后进行比较。

以下表示几个字符串的大小关系。
'AD'(16#414400) < 'BC'(16#424400)
'ADC' (16#41444300)< 'B'(16#42000000)
'ABC' (16#41424300)< 'ABD'(16#41424400)
'ABC' (16#41424300)> 'AB'(16#41420000)
'AB' (16#414200)= 'AB'(16#414200)

字符串中使用了多字节字符时，按字节分割该字符进行解释。例如，双字节字符16#C281解释为有16#C2和16#81的2个单字节字符。

LEascii指令下，“In1”='AB'、“In2”='AC'、“In3” = 'AC'时的示例如下所示。变量abc的值为TRUE。



参考

字符串比较的指令便于将多个字符串按照字符代码的顺序进行排列。例如，英文字母的字符代码的顺序与字母的顺序相同，因此可按照字母的顺序进行排列。

版本相关信息

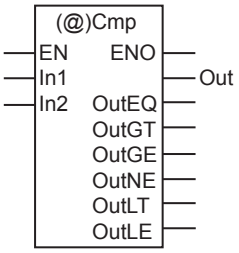
Ver.1.02以上的Sysmac Studio中，也有使用 “LT，</LE，<=/GT，>/GE，>=指令(P.2–98)”字符串比较的方法。LT,</LE,<=/GT,>/GE,>=指令的字符串比较和本指令的规格完全相同。

使用注意事项

- 本指令无法用于电路的末尾段。使用时，Sysmac Studio会发生异常，无法传送至控制器。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- “In1” ~ “InN”中请指定仅由ASCII码构成的字符串。

Cmp

比较2个数值。

指令	名称	FB/FUN	图形表现	ST表现
Cmp	比较	FUN		Out:=Cmp(In1, In2, OutEQ, OutGT, OutGE, OutNE, OutLT, OutLE); Out可省略

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1,In2	比较对象	输入	要比较的数值	遵从数据类型	–	(*)
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
OutEQ	=标志		=标志	遵从数据类型		
OutGT	>标志		>标志			
OutGE	>=标志		>=标志			
OutNE	<>标志		<>标志			
OutLT	<标志		<标志			
OutLE	<=标志		<=标志			

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1,In2						○	○	○	○	○	○	○	○	○	○					
Out	○																			
OutEQ	○																			
OutGT	○																			
OutGE	○																			
OutNE	○																			
OutLT	○																			
OutLE	○																			

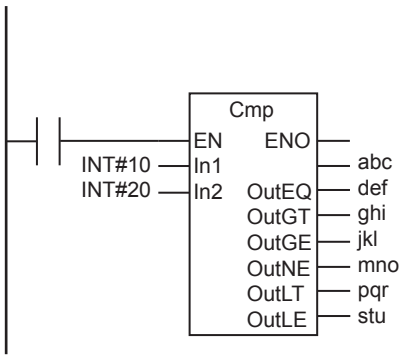
功能

比较2个数值 “In1”和 “In2”，输出各标记的值。
各标志的值如下所示。

标志	值
“OutEQ”	“In1” = “In2”　时为TRUE，其他情况时为FALSE
“OutGT”	“In1” > “In2”　时为TRUE，其他情况时为FALSE
“OutGE”	“In1” >= “In2”时为TRUE，其他情况时为FALSE
“OutNE”	“In1” <> “In2”时为TRUE，其他情况时为FALSE
“OutLT”	“In1” < “In2”　时为TRUE，其他情况时为FALSE
“OutLE”	“In1” <= “In2”时为TRUE，其他情况时为FALSE

“In1”=INT#10、“In2”=INT#20时的示例如下所示。变量def、ghi、jkl的值为FALSE，abc、mno、pqr、stu的值为TRUE。

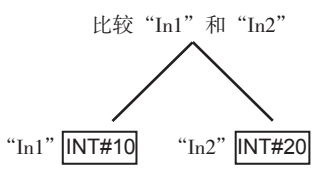
LD



ST

```
abc:=Cmp(INT#10, INT#20, def, ghi, jkl, mno, pqr, stu);
```

比较“In1”和“In2”。
根据各判定条件，结果如下所示。



“Out”	始终为TRUE
“OutEQ”	由于“In1” = “In2”不成立，因此结果为FALSE
“OutGT”	由于“In1” > “In2”不成立，因此结果为FALSE
“OutGE”	由于“In1” ≥ “In2”不成立，因此结果为FALSE
“OutNE”	由于“In1” ≠ “In2”成立，因此结果为TRUE
“OutLT”	由于“In1” < “In2”成立，因此结果为TRUE
“OutLE”	由于“In1” ≤ “In2”成立，因此结果为TRUE

→

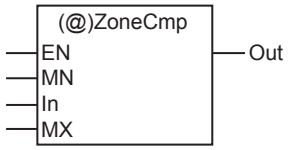
“Out” =abc	TRUE
“OutEQ” =def	FALSE
“OutGT” =ghi	FALSE
“OutGE” =jkl	FALSE
“OutNE” =mno	TRUE
“OutLT” =pqr	TRUE
“OutLE” =stu	TRUE

使用注意事项

- “In1”和 “In2”的数据类型不同时，将类型扩展为包括两者数据类型有效范围在内的数据类型后，再进行比较。
- “In1” 、“In2”为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- 带符号整数型(SINT，INT，DINT，LINT)和无符号整数型(USINT，UINT，UDINT，ULINT)无法比较。
- +∞之间或–∞之间判断为相等。
- “In1”、“In2”中任一值为非数时，“OutEQ”、“OutGT”、“OutGE”、“OutNE”、“OutLT”、“OutLE”的值为FALSE。

ZoneCmp

判定比较数据是否在指定的上限值与下限值之间。

指令	名称	FB/FUN	图形表现	ST表现
ZoneCmp	区域比较	FUN		Out:=ZoneCmp(MN, In, MX);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
MN	下限值	输入	下限值	遵从数据类型	–	0
In	比较数据		要比较的数值			(*)
MX	上限值		上限值			0
Out	比较结果	输出	比较结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
MN						○	○	○	○	○	○	○	○	○	○	○	○	○	○	
																(*)				
In						○	○	○	○	○	○	○	○	○	○	○	○	○	○	
																(*)				
MX						○	○	○	○	○	○	○	○	○	○	○	○	○	○	
																(*)				
Out	○																			

* TIME型、DATE型、TOD型、DT型在Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio中可指定。

功能

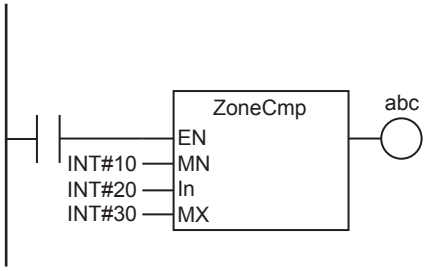
判定比较数据 “In”是否在上限值 “MX”和下限值 “MN”之间。
“MX” ≥ “In” ≥ “MN”时，“Out”的值为TRUE，否则为FALSE。

整数、实数以外的数据类型的值的大小关系的判断如下表所示。

数据类型	大小关系
TIME	值较大者判断为大。
DATE、TOD、DT	对于日期和时刻，较后者判断为大。

“MN”=INT#10、“In”=INT#20、“MX”=INT#30时的示例如下所示。变量abc的值为TRUE。

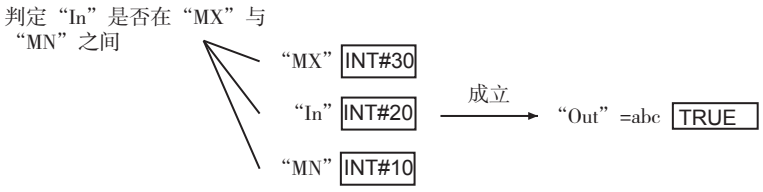
LD



ST

```
abc:=ZoneCmp(INT#10, INT#20, INT#30);
```

判定“MX” ≥ “In” ≥ “MN”是否成立。
由于成立，因此abc的值为TRUE。



参考

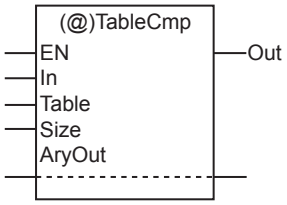
比较TIME型、DT型、TOD型的大小时，请符合要比较的值的精度。有 “TruncTime指令(P.2-642)”、“TruncDt指令(P.2-646)”、“TruncTod指令(P.2-650)”，以符合值的精度。

使用注意事项

- “In”、“MX”、“MN”的数据类型不同时，将类型扩展为包括所有数据类型的有效范围在内的数据类型后，再进行比较。
- “In”、“MX”、“MN”为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- 带符号整数型(SINT，INT，DINT，LINT)和无符号整数型(USINT，UINT，UDINT，ULINT)无法比较。
- TIME型、DATE型、TOD型、DT型仅可在相同的数据类型之间进行比较。如果指定了不同的数据类型，则编连时会出现异常。
- +∞之间或–∞之间判断为相等。
- “In”的值为非数时，“Out”的值为FALSE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- 以下情况时会发生异常，“Out”的值为FELSE。
  - “MN”的值大于 “MX”的值时。
  - “MX”、“MN”中任意一个为非数时。

TableCmp

将比较数据与比较表指定的多个定义区间进行比较。

指令	名称	FB/FUN	图形表现	ST表现
TableCmp	表比较	FUN		Out:=TableCmp(In, Table, Size, AryOut);

变量

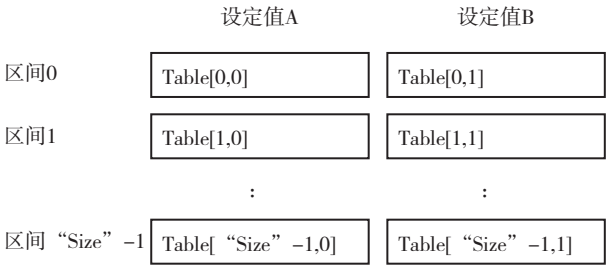
	名称	输入/输出	内容	有效范围	单位	初始值
In	比较数据	输入	要比较的数值	遵从数据类型	–	(*)
Table[ ] 2维数组	比较表		以各定义区间为元素的2维数组			
Size	比较规格		与 “In” 比较的Table[ ]的元素数			1
AryOut[ ] 数组	单独比较结果数组	输入输出	Table[ ]的各元素比较结果 TRUE：一致 FALSE：不一致	遵从数据类型	–	–
Out	比较结果	输出	TRUE：Table[ ]的所有元素与“In”一致 FALSE：不一致的元素至少有1个	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

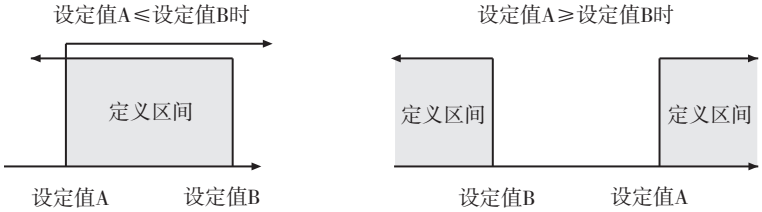
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○	○	○					
Table[ ] 2维数组	带有与 “In” 相同数据类型的元素的2维数组																			
Size							○													
AryOut[ ] 数组	○																			
Out	○																			

功能

与比较数据 “In”和比较表Table[ ]指定的 “Size”组的定义区间进行比较。
Table[ ]为2维数组，第1维为定义区间的编号，第2维的0号元素表示定义区间的设定值A，1号元素表示定义区间的设定值B。

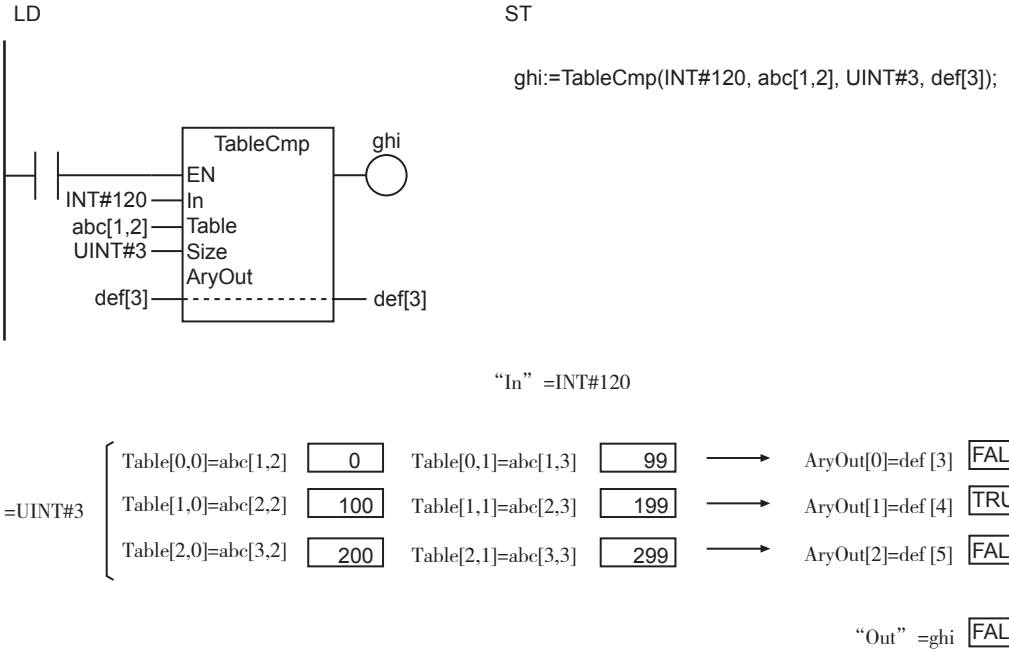


设定值A和设定值B的定义区间指定方法如下图所示。设定值A和设定值B的值包含在定义区间内。



“In”和表[ ]的比较结果保存在单独比较结果数组AryOut[ ]中。如果 “In”在编入i号的定义区间内，则AryOut[i]的值为TRUE；如果超出，则AryOut[i]的值为FALSE。如果AryOut[ ]的 “Size”个元素均为TRUE，则比较结果 “Out”的值为TRUE，否则为FALSE。

“In”=INT#120、“Size”=UINT#3时的示例如下所示。

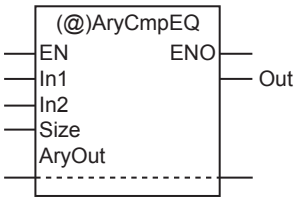
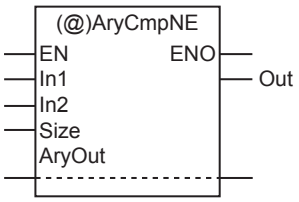


使用注意事项

- 请将 “In”和Table[ ]的元素的数据类型设为相同。否则，编连时会发生异常。
- 请务必使Table[ ]为2维数组。
- Table[ ]的第2维的数组的大小大于等于3时，将忽略第2维的2号之后的元素。
- AryOut[ ]的数组的大小大于等于 “Size”时，将比较结果保存在AryOut[0] ~ AryOut[“Size” –1]中。其他的数组元素不变。
- 带符号整数型(SINT，INT，DINT，LINT)和无符号整数型(USINT，UINT，UDINT，ULINT)无法比较。
- 比较对象为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- “Size”的值为0时，“Out”的值为FALSE，AryOut[ ]不变。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- 以下情况时会发生异常。“Out”为FALSE。
  - “Size”的值超出AryOut[ ]的数组大小时。
  - “Size”的值超出Table[ ]的第1维的数组大小时。
  - Table[ ]的第2维的数组大小为1时。

AryCmpEQ/AryCmpNE

比较2个数组的各元素。
AryCmpEQ：判定是否相等。
AryCmpNE：判定是否不同。

指令	名称	FB/FUN	图形表现	ST表现
AryCmpEQ	数组整体比较EQ	FUN		AryCmpEQ(In1, In2, Size, AryOut);
AryCmpNE	数组整体比较NE	FUN		AryCmpNE(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ],In2[ ]数组	比较数组	输入	以要比较的数值为元素的数组	遵从数据类型	–	(*)
Size	比较元素数		要比较的元素数			1
AryOut[ ]数组	比较结果数组	输入输出	比较结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

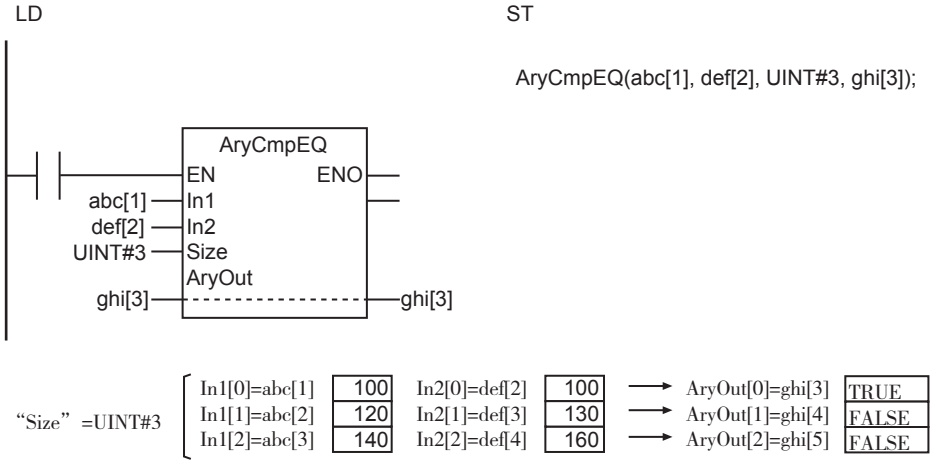
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○					
In2[ ]数组	与In1[ ]相同数据类型的数组																			
Size							○													
AryOut[ ]数组	○																			
Out	○																			

功能

2个数组In1[0]～In1[“Size”−1]、In2[0]～In2[“Size”−1]的相同元素编号之间进行比较。将比较结果保存在比较结果数组AryOut[0]～AryOut[“Size”−1]对应的元素编号中。
各指令的AryOut[i]的值如下所示。

指令	AryOut[i]的值
AryCmpEQ	In1[i]=In2[i] 时为TRUE，其他情况时为FALSE
AryCmpNE	In1[i]≠In2[i] 时为TRUE，其他情况时为FALSE

AryCmpEQ指令下 “Size”=UINT#3时的示例如下所示。

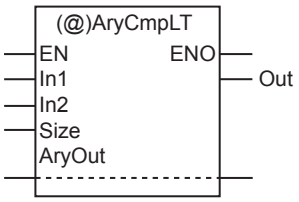
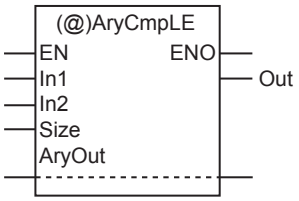
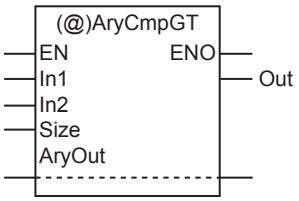
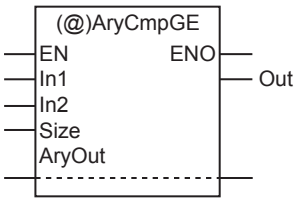


使用注意事项

- 请将In1[ ]和In2[ ]的数据类型设为相同。
- 请使AryOut[ ]数组的大小大于等于 “Size”。
- In1[ ]、In2[ ]为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、In2[ ]、AryOut[ ]中任一数组大小小于 “Size”时。

AryCmpLT/AryCmpLE/ AryCmpGT/AryCmpGE

比较2个数组的各元素的大小。
AryCmpLT： 比较(<)大小。
AryCmpLE： 比较(<=)大小。
AryCmpGT： 比较(>)大小。
AryCmpGE： 比较(>=)大小。

指令	名称	FB/ FUN	图形表现	ST表现
AryCmpLT	数组整体比较 LT	FUN		AryCmpLT(In1, In2, Size, AryOut);
AryCmpLE	数组整体比较 LE	FUN		AryCmpLE(In1, In2, Size, AryOut);
AryCmpGT	数组整体比较 GT	FUN		AryCmpGT(In1, In2, Size, AryOut);
AryCmpGE	数组整体比较 GE	FUN		AryCmpGE(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ],In2[ ] 数组	比较数组	输入	以要比较的数值为元素的数组	遵从数据类型	–	(*)
Size	比较元素数		要比较的元素数			1
AryOut[ ] 数组	比较结果数组	输入输出	比较结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

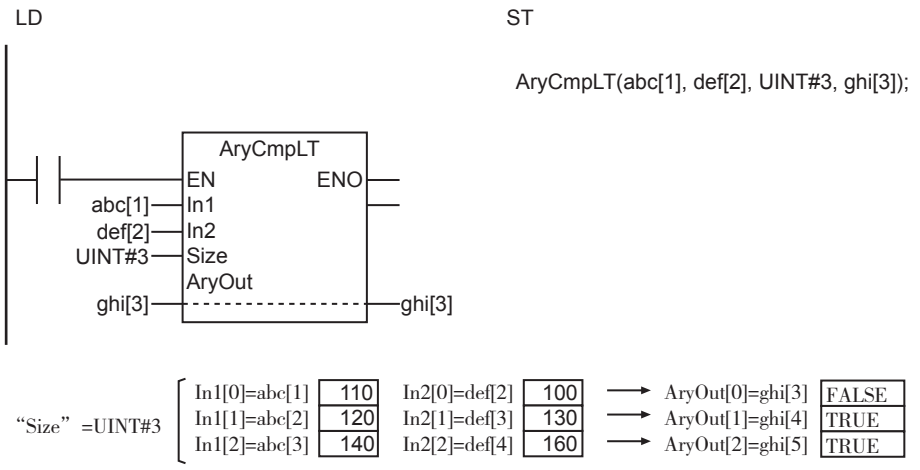
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组						○	○	○	○	○	○	○	○	○	○					
In2[ ] 数组	与In1[ ]相同数据类型的数组																			
Size							○													
AryOut[ ] 数组	○																			
Out	○																			

功能

2个数组In1[0] ~ In1[“Size” −1]、In2[0] ~ In2[“Size” −1]的相同元素编号之间比较大小。将比较结果保存在比较结果数组AryOut[0] ~ AryOut[“Size” −1]对应的元素编号中。
各指令的AryOut[i]的值如下所示。

指令	AryOut[i]的值
AryCmpLT	In1[i]<In2[i] 时为TRUE，其他情况时为FALSE
AryCmpLE	In1[i]<=In2[i]时为TRUE，其他情况时为FALSE
AryCmpGT	In1[i]>In2[i] 时为TRUE，其他情况时为FALSE
AryCmpGE	In1[i]>=In2[i]时为TRUE，其他情况时为FALSE

AryCmpLT指令下 “Size” =UINT#3时的示例如下所示。

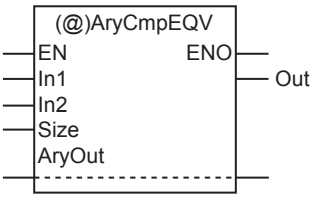
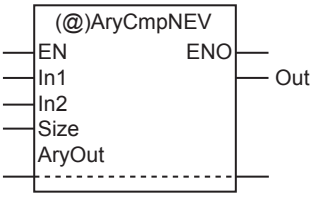


使用注意事项

- 请将In1[ ]和In2[ ]的数据类型设为相同。
- 请使AryOut[ ]数组的大小大于等于 “Size”。
- In1[ ]、In2[ ]为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、In2[ ]、AryOut[ ]中任一数组大小小于 “Size”时。

AryCmpEQV/AryCmpNEV

比较数组的各元素与数值。
AryCmpEQV：判定是否相等。
AryCmpNEV：判定是否不同。

指令	名称	FB/FUN	图形表现	ST表现
AryCmpEQV	数组元素比较EQ	FUN		AryCmpEQV(In1, In2, Size, AryOut);
AryCmpNEV	数组元素比较NE	FUN		AryCmpNEV(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ]数组	比较数组	输入	以要比较的数值为元素的数组	遵从数据类型	–	(*)
In2	比较数值		要比较的数值			
Size	比较元素数		要比较的元素数			1
AryOut[ ]数组	比较结果数组	输入输出	比较结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

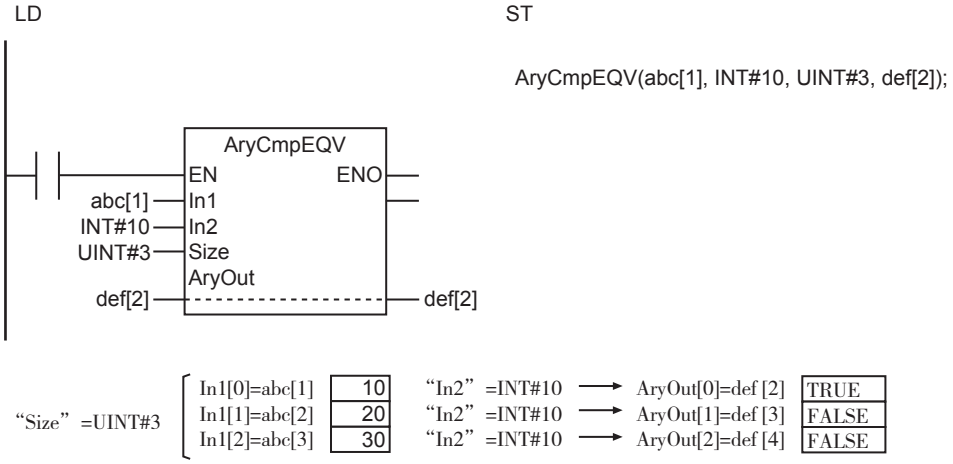
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○					
In2	与In1[ ]的元素相同的数据类型																			
Size							○													
AryOut[ ]数组	○																			
Out	○																			

功能

对比较数组In1[0] ~ In1[“Size” −1]和比较数值 “In2”进行比较。
将比较结果保存在比较结果数组AryOut[0] ~ AryOut[“Size” −1]对应的元素编号中。
各指令的AryOut[i]的值如下所示。

指令	AryOut[i]的值
AryCmpEQV	In1[i]= “In2” 时为TRUE，其他情况时为FALSE
AryCmpNEV	In1[i]≠ “In2”时为TRUE，其他情况时为FALSE

AryCmpEQV指令下 “In2”=INT#10、“Size”=UINT#3时的示例如下所示。

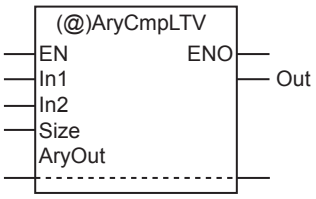
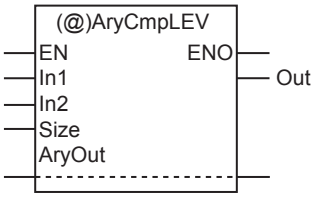
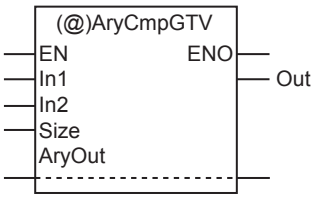
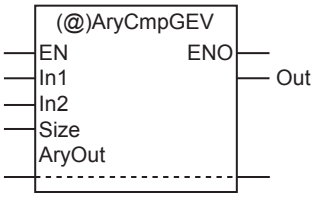


使用注意事项

- 请将In1[ ]和 “In2”的数据类型设为相同。
- 请使AryOut[ ]数组的大小大于等于 “Size”。
- In1[ ]、“In2”为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、AryOut[ ]中任一数组大小小于 “Size”时。

AryCmpLTV/AryCmpLEV/ AryCmpGTV/AryCmpGEV

比较数组的各元素与数值的大小。
AryCmpLTV ： 比较(<)大小。
AryCmpLEV ： 比较(<=)大小。
AryCmpGTV ： 比较(>)大小。
AryCmpGEV ： 比较(>=)大小。

指令	名称	FB/ FUN	图形表现	ST表现
AryCmpLTV	数组元素比较 LT	FUN		AryCmpLTV(In1, In2, Size, AryOut);
AryCmpLEV	数组元素比较 LE	FUN		AryCmpLEV(In1, In2, Size, AryOut);
AryCmpGTV	数组元素比较 GT	FUN		AryCmpGTV(In1, In2, Size, AryOut);
AryCmpGEV	数组元素比较 GE	FUN		AryCmpGEV(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ] 数组	比较数组	输入	以要比较的数值为元素的数组	遵从数据类型	–	(*)
In2	比较数值		要比较的数值			
Size	比较元素数		要比较的元素数			1
AryOut[ ] 数组	比较结果数组	输入输出	比较结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

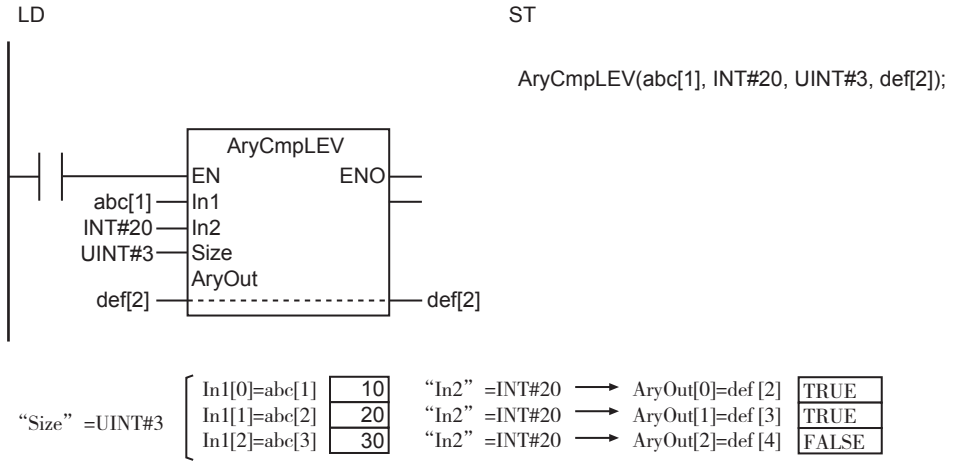
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组						○	○	○	○	○	○	○	○	○	○					
In2	与In1[ ]的元素相同的数据类型																			
Size							○													
AryOut[ ] 数组	○																			
Out	○																			

功能

对比较数组In1[0] ~ In1[“Size” –1]和比较数值 “In2”的大小进行比较。
将比较结果保存在比较结果数组AryOut[0] ~ AryOut[“Size” –1]对应的元素编号中。
各指令的AryOut[i]的值如下所示。

指令	AryOut[i]的值
AryCmpLTV	In1[i]< “In2” 时为TRUE，其他情况时为FALSE
AryCmpLEV	In1[i]<= “In2”时为TRUE，其他情况时为FALSE
AryCmpGTV	In1[i]> “In2” 时为TRUE，其他情况时为FALSE
AryCmpGEV	In1[i]>= “In2”时为TRUE，其他情况时为FALSE

AryCmpLEV指令下 “In2” =INT#20、“Size” =UINT#3时的示例如下所示。



使用注意事项

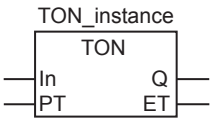
- 请将In1[ ]和 “In2”的数据类型设为相同。
- 请使AryOut[ ]数组的大小大于等于 “Size”。
- In1[ ]、“In2”为实数时，如果含有除不尽的除法结果等，由于误差的原因，处理结果可能会出现意外。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、AryOut[ ]中任一数组大小小于 “Size”时。

定时器指令

指令	名称	页码
TON	ON延时定时器	2–128
TOF	OFF延迟定时器	2–133
TP	脉冲输出	2–136
AccumulationTimer	累计定时器	2–139
Timer	100ms定时器	2–142

TON

在启动经过设定时间后，输出TRUE的定时器。

指令	名称	FB/FUN	图形表现	ST表现
TON	ON延时定时器	FB		TON_instance (In, PT, Q, ET);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	定时器输入	输入	TRUE ：定时器启动指示 FALSE：定时器复位指示	遵从数据类型	–	FALSE
PT	设定时间		从定时器启动到 “Q”变为TRUE的时间	(*)	ms	0
Q	定时器输出	输出	TRUE ：定时器输出ON FALSE：定时器输出OFF	遵从数据类型	–	–
ET	经过时间		定时器启动后经过的时间	(*)	ms	

* T#0ms ~ T#106751d_23h_47m_16s_854.775807ms

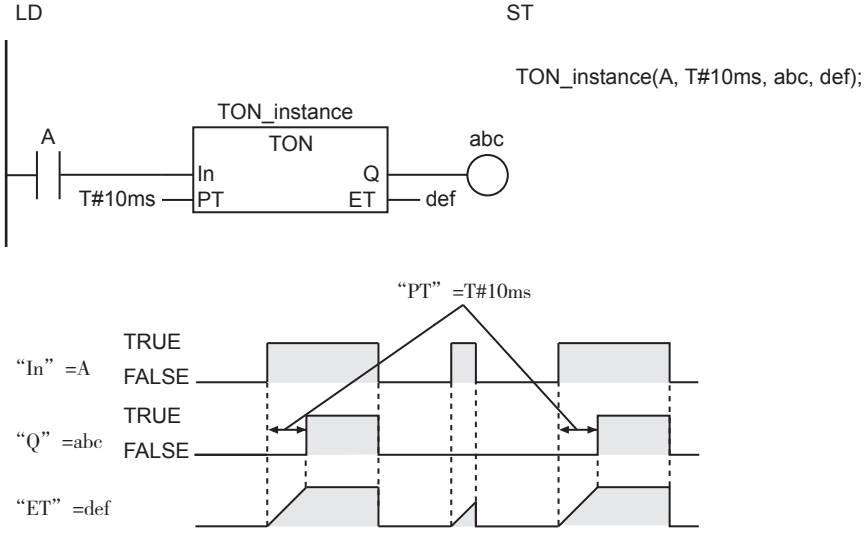
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○																			
PT																○				
Q	○																			
ET																○				

功能

在启动经过设定时间后，输出TRUE的定时器。设定时间的最小单位为ns。
定时器输入 “In”从FALSE变为TRUE时，定时器启动，经过时间 “ET”与时间同时增加。
“ET”到达设定时间 “PT”时，定时器输出 “Q”变为TRUE。此时，“ET”停止增加。
“In”为FALSE时，定时器复位。“ET”为0，“Q”为FALSE。

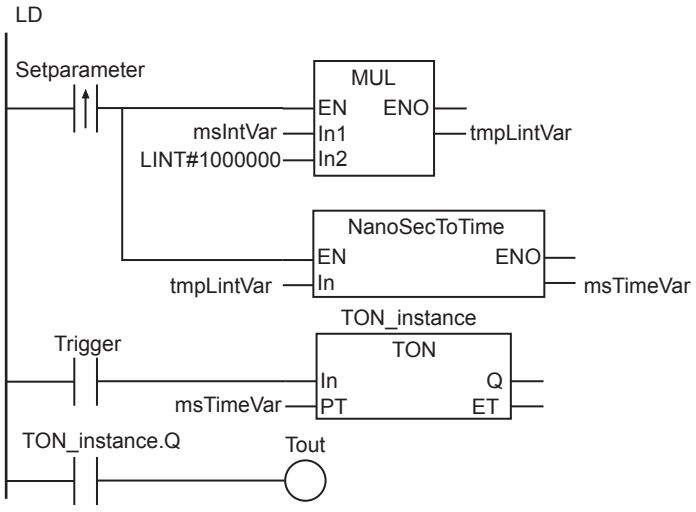
定时器启动后，即使在 “ET”到达 “PT”之前，“In”为FALSE时，定时器也将复位。

“PT”=T#10ms时的示例和时序图如下所示。变量A变为TRUE的10ms后，变量abc的值变为TRUE。



参考

- 如果需要定时器启动后输出立即变为TRUE，经过设定时间后变为FALSE，请使用 “TP指令(P.2–136)”。
- 如果需要 “In”为FALSE时定时器启动，经过时间到达设定时间后，定时器输出变为FALSE，请使用 “TOF指令(P.2–133)”。
- 如果需要缩短指令执行时间，请使用以100ms为单位计时的 “Timer指令(P.2–142)”。
- 连接不支持TIME型的触摸屏等时，需先将以整数型表示的设定时间转换为TIME型后，再输入至本指令。从整数型转换为TIME型时，请使用 “NanoSecToTime指令(P.2–626)”。从TIME型转换为整数型时，请使用 “TimeToNanoSec指令(P.2–624)”。上述指令的时间单位均为纳秒。INT型变量msIntVar的设定时间以ms为单位时的用户程序如下图所示。




```
ST
tmpLintVar:=msIntVar*LINT#1000000;
msTimeVar:=NanoSecToTime(tmpLintVar);
TON_instance(In:=Trigger, PT:=msTimeVar, Q=>Tout);
```

使用注意事项

- 相对于 “PT”，“Q”变为TRUE的计时误差为–100ns ~ (100ns+1个任务周期)。
  上述范围的详情如下。
  - ± 100ns为 “ET”的计时误差。
  - 在各任务周期判断 “ET”计时是否达到 “PT”，如果是，则立即延迟1个任务周期。
- Sysmac Studio以0.001ms为单位表示时间值，但计时精度为1ns。
- 如果开始运行时 “In”的值已为TRUE，则从该时刻开始计时。
- 为 “PT”设定了T#0ms或负数时，“In”的值变为TRUE后，“Q”变为TRUE。
- “In”的值为TRUE的期间，可变更 “PT”的值。此时的动作如下所示。

定时器的状态	“Q”的值	变更后的 “PT”值	动作
计时结束后	TRUE	–	“Q”的值保持为TRUE。 “ET”的值也不变(保持变更前的 “PT”的值)。
计时中	FALSE	“PT” ≥ “ET”	继续计时。“ET”的值达到变更后的“PT”的值时，“Q”的值变为TRUE，“ET”停止增加。
		“PT” < “ET”	“Q”的值立即变为TRUE。 “ET”也立即停止增加。

- 本指令存在于主站控制区域，通过主站控制执行复位时，复位定时器。“ET”的值为 0，“Q”的值为FALSE。
- 因执行JMP系统指令(JMP指令等)而未执行本指令时，不更新 “ET”的值。但该期间仍将继续计时。因此，之后执行本指令时，“ET”将更新为正确的值。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值为FALSE。

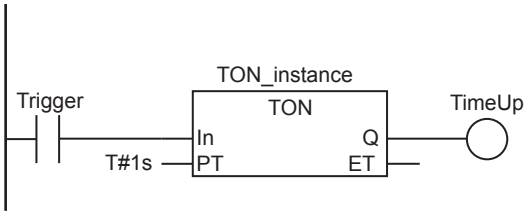
示例程序

● 用1个ON延时定时器测量时间的示例

Trigger的值变为TRUE的1s后，TimeUp的值变为TRUE。

LD

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
TimeUp	BOOL	FALSE	定时器输出
TON_instance	TON		



ST

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
TimeUp	BOOL	FALSE	定时器输出
TON_instance	TON		

```
IF (Trigger=TRUE) THEN
    TON_instance(In:=TRUE, PT:=T#1s, Q=>TimeUp);
ELSE
    TON_instance(In:=FALSE, Q=>TimeUp);
END_IF;
```

ST

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
TimeUp	BOOL	FALSE	定时器输出
TON_instance	TON		

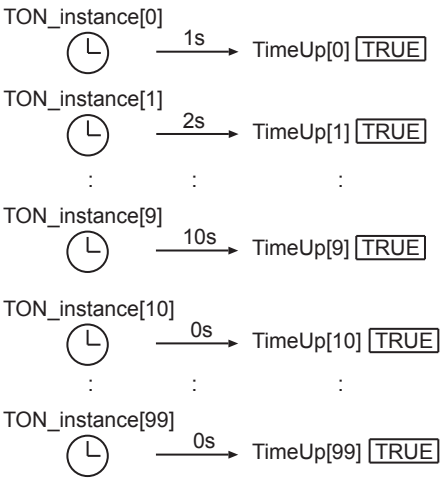
```
TON_instance(In:=Trigger, PT:=T#1s, Q=>TimeUp);
```

● 用多个ON延时定时器测量时间的示例

准备TON_instance[0] ~ TON_instance[99]的100个ON延时定时器。通过将定时器输入Input[0] ~ Input[99]的值设为TRUE，来启动每个定时器。

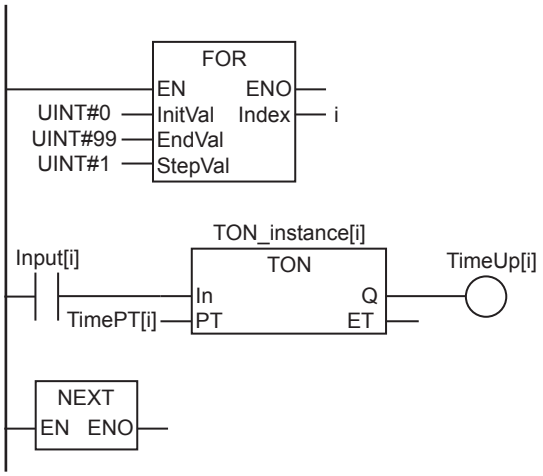
最初10个定时器TON_instance[0] ~ TON_instance[9]分别启动后，i+1秒后(i=0 ~ 9)，将TimeUp[i](i=0 ~ 9)的值设为TRUE。

对于剩余 90 个定时器 TON_instance[10] ~ TON_instance[99]，启动后立即将 TimeUp[i](i=10 ~ 99) 的值设为 TRUE。



LD

名称	数据类型	初始值	注释
Input	ARRAY[0..99] OF BOOL	[100(FALSE)]	定时器输入
TimeUp	ARRAY[0..99] OF BOOL	[100(FALSE)]	定时器输出
TimePT	ARRAY[0..99] OF TIME	[T#1s, T#2s, T#3s, T#4s, T#5s, T#6s, T#7s, T#8s, T#9s, T#10s, 90(T#0s)]	设定时间
TON_instance	ARRAY[0..99] OF TON		
i	UINT	0	索引



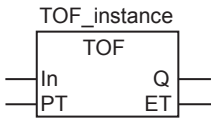
ST

名称	数据类型	初始值	注释
Input	ARRAY[0..99] OF BOOL	[100(FALSE)]	定时器输入
TimeUp	ARRAY[0..99] OF BOOL	[100(FALSE)]	定时器输出
TimePT	ARRAY[0..99] OF TIME	[T#1s, T#2s, T#3s, T#4s, T#5s, T#6s, T#7s, T#8s, T#9s, T#10s, 90(T#0s)]	设定时间
TON_instance	ARRAY[0..99] OF TON		
i	UINT	0	索引

```
FOR i:=UINT#0 TO UINT#99 DO
  TON_instance[i](
    In := Input[i],
    PT := TimePT[i],
    Q  =>TimeUp[i]);
END_FOR;
```

TOF

在启动经过设定时间后，输出FALSE的定时器。

指令	名称	FB/FUN	图形表现	ST表现
TOF	OFF延迟定时器	FB		TOF_instance (In, PT, Q, ET);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	定时器输入	输入	TRUE ：定时器复位指示 FALSE：定时器启动指示	遵从数据类型	–	FALSE
PT	设定时间		从定时器启动到 “Q”变为FALSE的时间	(*)	ms	0
Q	定时器输出	输出	TRUE ：定时器输出ON FALSE：定时器输出OFF	遵从数据类型	–	–
ET	经过时间		定时器启动后经过的时间	(*)	ms	

* T#0ms ~ T#106751d_23h_47m_16s_854.775807ms

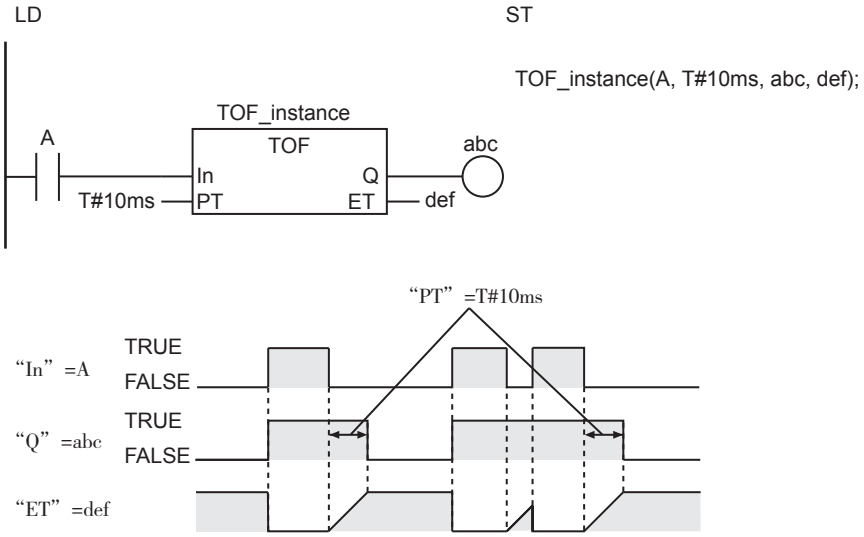
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○																			
PT																○				
Q	○																			
ET																○				

功能

在启动经过设定时间后，输出FALSE的定时器。设定时间的最小单位为ns。
定时器输入 “In”从TRUE变为FALSE时，定时器启动，经过时间 “ET”与时间同时增加。
“ET”到达设定时间 “PT”时，定时器输出 “Q”为FALSE。此时，“ET”停止增加。
“In”为TRUE时，定时器复位。“ET”为0，“Q”为TRUE。

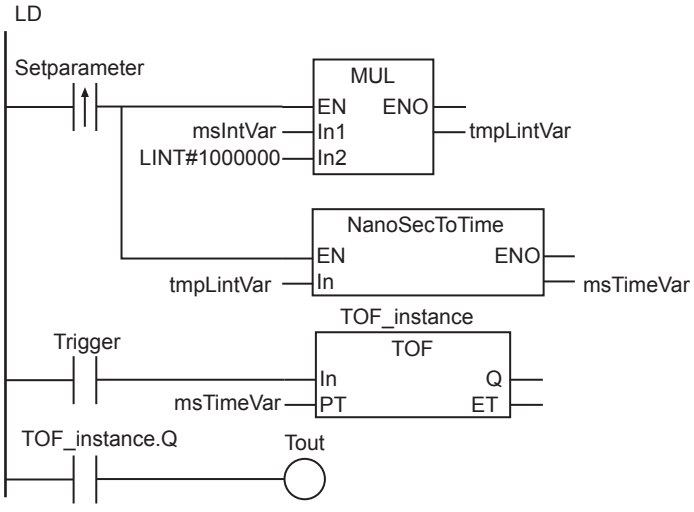
定时器启动后，即使在 “ET”到达 “PT”之前，“In”为FALSE时，定时器也将复位。

“PT”=T#10ms时的示例和时序图如下所示。变量A变为FALSE的10ms后，变量abc的值变为FALSE。



参考

- 如果需要定时器启动后输出立即为TRUE，经过设定时间后变为FALSE，请使用 “TP指令(P.2-136)”。
- 如果需要 “In”为 TRUE 时定时器启动，经过时间到达设定时间时，定时器输出变为 TRUE，请使用 “TON指令(P.2-128)”。
- 连接不支持 TIME 型的触摸屏等时，需先将以整数型表示的设定时间转换为 TIME 型后，再输入至本指令。从整数型转换为TIME型时，请使用 “NanoSecToTime指令(P.2-626)”。从TIME型转换为整数型时，请使用 “TimeToNanoSec指令(P.2-624)”。上述指令的时间单位均为纳秒。INT型变量msIntVar的设定时间以ms为单位时的用户程序如下所示。




```
ST
tmpLintVar:=msIntVar*LINT#1000000;
msTimeVar:=NanoSecToTime(tmpLintVar);
TOF_instance(In:=Trigger, PT:=msTimeVar, Q=>Tout);
```

使用注意事项

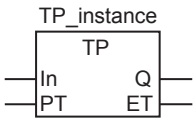
- 相对于 “PT”，“Q”变为TRUE的计时误差为–100ns ~ (100ns+1个任务周期)。
  上述范围的详情如下。
  - ±100ns为 “ET”的计时误差。
  - 在各任务周期判断 “ET”计时是否达到 “PT”，如果是，则立即延迟1个任务周期。
- Sysmac Studio以0.001ms为单位表示时间值，但计时精度为1ns。
- 为 “PT”设定了T#0ms或负数时，“In”的值刚变为FALSE后，“Q”变为FALSE。
- 执行本指令后，如果 “In”的值变为TRUE，则 “Q”的值变为TRUE。定时器启动后仅经过 “PT”时间时，“Q”的值变为FALSE。
- “In”的值为FALSE的期间，可变更 “PT”的值。此时的动作如下所示。

定时器的状态	“Q”的值	变更后的 “PT”值	动作
计时结束后	FALSE	–	“Q”的值保持为FALSE。 “ET”的值也不变(保持变更前的 “PT”的值)。
计时中	TRUE	“PT” ≥ “ET”	继续计时。“ET”的值达到变更后的“PT”的值时，“Q”的值变为FALSE，“ET”停止增加。
		“PT” < “ET”	“Q”的值立即变为FALSE。 “ET”也立即停止增加。

- 本指令存在于主站控制区域，通过主站控制执行复位时，动作如下所示。
  - “ET”的值为0，“Q”的值为TRUE。
  - 将 “Q”连接至后段的Out指令时，该Out指令中会输入FALSE。
  - 复位解除后开始计时。
- 因执行JMP系统指令(JMP指令等)而未执行本指令时，不更新 “ET”的值。但该期间仍将继续计时。因此，之后执行本指令时，“ET”将更新为正确的值。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值为FALSE。

TP

在启动经过设定时间后，输出TRUE的定时器。

指令	名称	FB/FUN	图形表现	ST表现
TP	脉冲输出	FB		TP_instance (In, PT, Q, ET);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	定时器输入	输入	TRUE ：定时器启动指示 FALSE：定时器复位指示	遵从数据类型	–	FALSE
PT	设定时间		“Q”保持TRUE的时间	(*)	ms	0
Q	定时器输出	输出	TRUE ：定时器输出ON FALSE：定时器输出OFF	遵从数据类型	–	–
ET	经过时间		定时器启动后经过的时间	(*)	ms	

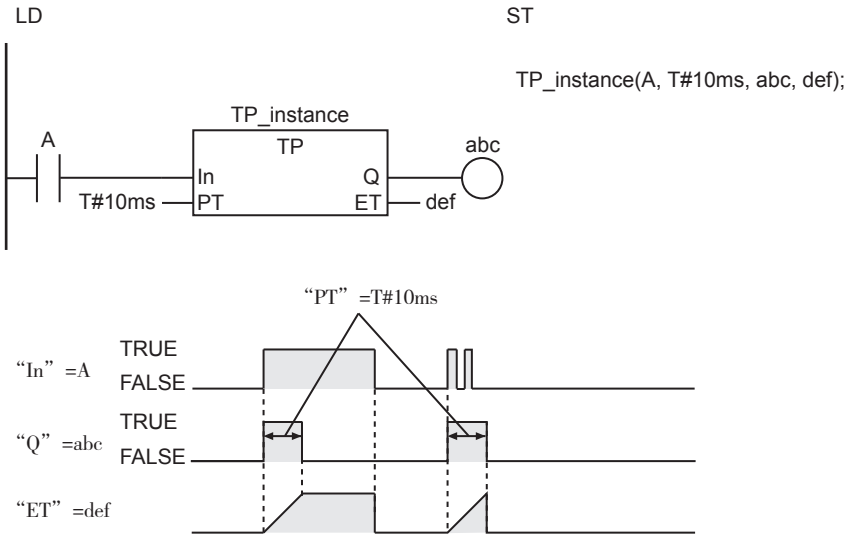
* T#0ms ~ T#106751d_23h_47m_16s_854.775807ms

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○																			
PT																○				
Q	○																			
ET																○				

功能

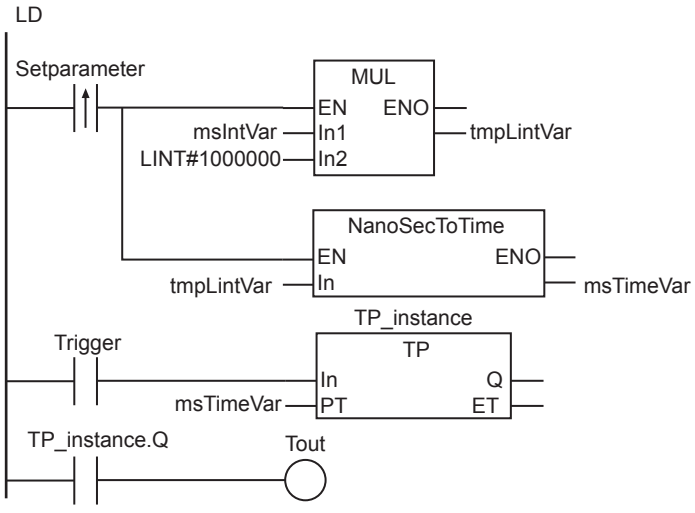
在启动经过设定时间后，输出TRUE的定时器。设定时间的最小单位为ns。
定时器输入 “In”从FALSE变为TRUE时，定时器启动，定时器输出 “Q”变为TRUE。经过时间 “ET”与时间同时增加。
“ET”到达设定时间 “PT”时，定时器输出 “Q”为FALSE。此时，“ET”停止增加。
“In”变为FALSE时，定时器复位。“ET”变为0。
但启动后，“ET”到达 “PT”之前，“In”变为FALSE时，定时器不会复位。

“PT”=T#10ms时的示例和时序图如下所示。变量A变为TRUE时，变量abc的值变为TRUE，10ms后变为FALSE。



参考

- 如果需要 “In”变为TRUE时定时器启动，经过时间到达设定时间时，定时器输出变为TRUE，请使用 “TON指令(P.2-128)”。
- 如果需要 “In”变为FALSE时定时器启动，经过时间到达设定时间时，定时器输出变为FALSE，请使用 “TOF指令(P.2-133)”。
- 连接不支持 TIME 型的触摸屏等时，需先将以整数型表示的设定时间转换为 TIME 型后，再输入至本指令。从整数型转换为TIME型时，请使用 “NanoSecToTime指令(P.2-626)”。从TIME型转换为整数型时，请使用 “TimeToNanoSec指令(P.2-624)”。上述指令的时间单位均为纳秒。INT型变量msIntVar的设定时间以ms为单位时的用户程序如下所示。




```
ST
tmpLintVar:=msIntVar*LINT#1000000;
msTimeVar:=NanoSecToTime(tmpLintVar);
TP_instance(In:=Trigger, PT:=msTimeVar, Q=>Tout);
```

使用注意事项

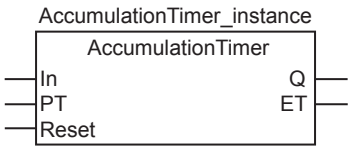
- 相对于 “PT”，“Q”变为TRUE的计时误差为–100ns ~ (100ns+1个任务周期)。
  上述范围的详情如下。
  - ±100ns为 “ET”的计时误差。
  - 在各任务周期判断 “ET”计时是否达到 “PT”，如果是，则立即延迟1个任务周期。
- Sysmac Studio以0.001ms为单位表示时间值，但计时精度为1ns。
- 如果开始运行时 “In”的值已为TRUE，则从该时刻开始计时。
- 为 “PT”设定了T#0ms或负数时，即使 “In”的值变为TRUE，“Q”也不会变为TRUE。
- “In”的值为TRUE的期间，可变更 “PT”的值。此时的动作如下所示。

定时器的状态	“Q”的值	变更后的 “PT”值	动作
计时结束后	FALSE	–	“Q”的值保持为FALSE。 “ET”的值也不变(保持变更前的 “PT”的值)。
计时中	TRUE	“PT” ≥ “ET”	继续计时。“ET”的值达到变更后的“PT”的值时，“Q”的值变为FALSE，“ET”停止增加。
		“PT” < “ET”	“Q”的值立即变为FALSE。 “ET”也立即停止增加。

- 本指令存在于主站控制区域，通过主站控制执行复位时，如果正在计时，则将继续计时直至计时完成。之后，“ET”的值变为0，“Q”的值变为FALSE。将“Q”连接至后段的Out指令相时，“Q”的值即使为TRUE，Out指令中也会输入FALSE。
- 因执行JMP系统指令(JMP指令等)而未执行本指令时，不更新 “ET”的值，也不计时。之后执行本指令时，重新开始计时。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值为FALSE。

AccumulationTimer

累计定时器输入为TRUE的时间的计时器。

指令	名称	FB/FUN	图形表现	ST表现
Accumulation Timer	累计定时器	FB		AccumulationTimer_instance(In, PT, Reset, Q, ET);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	定时器输入	输入	TRUE ：定时器工作 FALSE：定时器停止	遵从数据类型	–	FALSE
PT	设定时间		计时的最大值	(*)	ms	0
Reset	复位		TRUE ：定时器复位 FALSE ：定时器不复位	遵从数据类型	–	FALSE
Q	定时器输出	输出	TRUE ：“ET”到达 “PT” FALSE：“ET”未到达 “PT”	遵从数据类型	–	–
ET	累计时间		累计时间	(*)	ms	

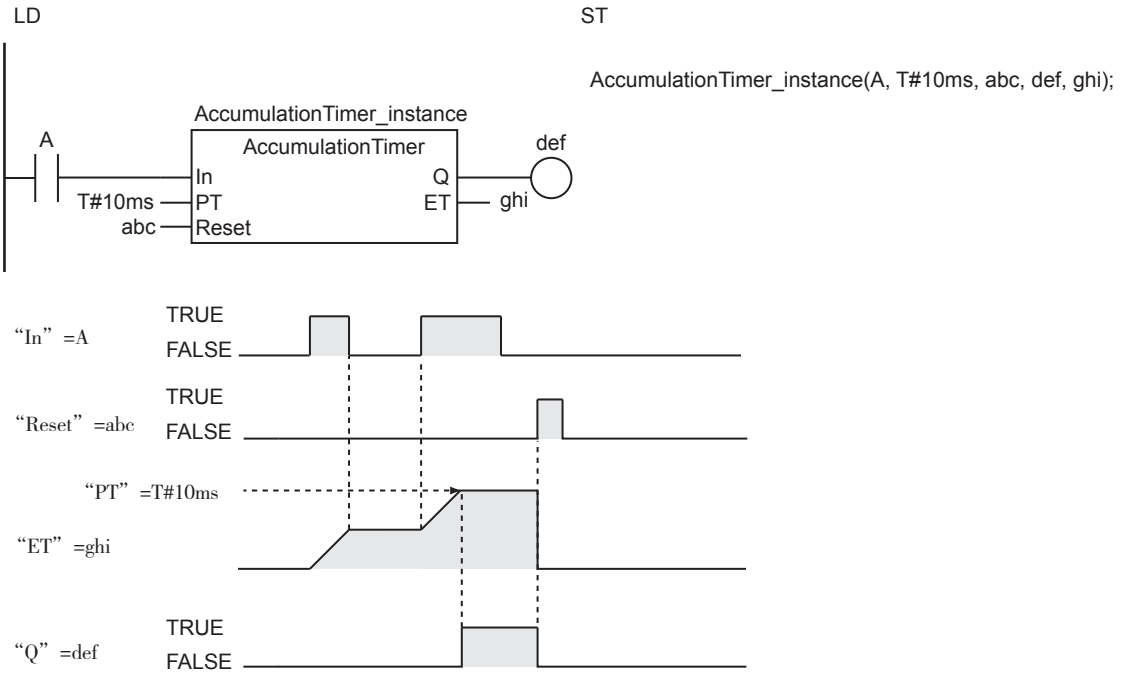
* T#0ms ~ T#106751d_23h_47m_16s_854.775807ms

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○																			
PT																○				
Reset	○																			
Q	○																			
ET																○				

功能

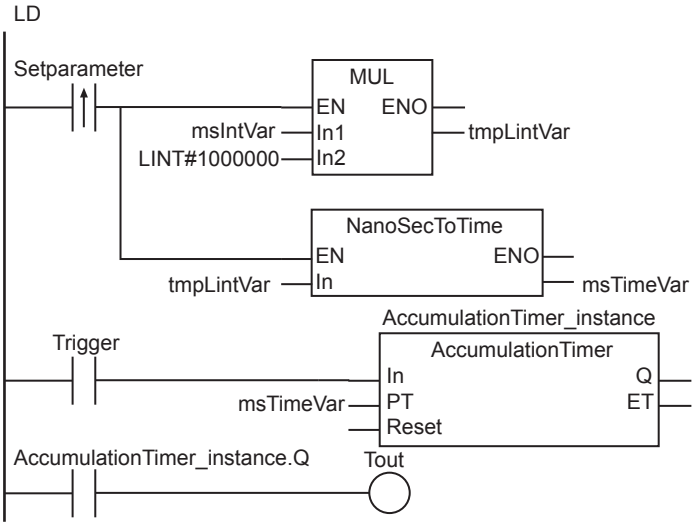
累计定时器输入 “In”为TRUE的时间的定时器。设定时间的最小单位为ns。
复位“Reset”为FALSE时，“In”从FALSE变为TRUE后，定时器启动，累计时间 “ET”与时间同时增加。
“In”变为FALSE时，定时器停止。但保持该时刻的 “ET”值。
“In”再次变为TRUE时，定时器再次启动。“ET”从保持值开始增加。
“ET”到达设定时间 “PT”时，定时器输出 “Q”变为TRUE。此时，“ET”停止增加。
“Reset”变为TRUE时，定时器复位。“ET”的值变为0，“Q”变为FALSE。

“PT”=T#10ms时的示例和时序图如下所示。变量A为TRUE的累计时间到达10ms时，变量abc的值变为TRUE。



参考

- “In”变为FALSE时，如果要将定时器输出和经过时间复位，请使用 “TON指令(P.2-128)”。
- 连接不支持TIME型的触摸屏等时，需先将以整数型表示的设定时间转换为TIME型后，再输入至本指令。从整数型转换为TIME型时，请使用 “NanoSecToTime指令(P.2-626)”。从TIME型转换为整数型时，请使用 “TimeToNanoSec指令(P.2-624)”。上述指令的时间单位均为纳秒。INT型变量msIntVar的设定时间以ms为单位时的用户程序如下所示。




```
ST
tmpLintVar:=msIntVar*LINT#1000000;
msTimeVar:=NanoSecToTime(tmpLintVar);
AccumulationTimer_instance(In:=Trigger, PT:=msTimeVar, Q=>Tout);
```

使用注意事项

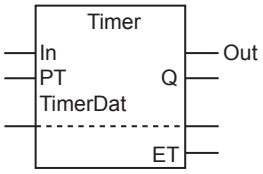
- 相对于 “PT”，“Q”变为TRUE的计时误差为–100ns ~ (100ns+1个任务周期)。
  上述范围的详情如下。
  - ±100ns为 “ET”的计时误差。
  - 在各任务周期判断 “ET”计时是否达到 “PT”，如果是，则立即延迟1个任务周期。
- Sysmac Studio以0.001ms为单位表示时间值，但计时精度为1ns。
- 如果开始运行时 “In”的值已为TRUE，则从该时刻开始计时。
- 为 “PT”设定了T#0ms或负数时，“In”的值变为TRUE后，“Q”变为TRUE。
- “ET”的值到达 “PT”的值之前，可变更 “PT”的值。此时的动作如下所示。

定时器的状态	“Q”的值	变更后的 “PT”值	动作
计时结束后	TRUE	–	“Q”的值保持为TRUE。 “ET”的值也不变(保持变更前的 “PT”的值)。
计时中	FALSE	“PT” ≥ “ET”	“In”的值变为TRUE时，继续计时。“ET”的值达到变更后的“PT”的值时，“Q”的值变为TRUE，“ET”停止增加。
		“PT” < “ET”	“In”的值变为TRUE时，“Q”的值立即变为TRUE。 “ET”也立即停止增加。

- 本指令存在于主站控制区域，通过主站控制执行复位时，动作如下所示。
  - 定时器停止。“ET”和 “Q”保持该时刻的值。
  - 通过主站控制解除复位时，“ET”从保持值重新开始增加。
  - 将 “Q”连接至后段的Out指令时，“Q”的值即使为TRUE，Out指令中也会输入FALSE。
  - “Reset”为有效。
- 因执行JMP系统指令(JMP指令等)而未执行本指令时，不更新 “ET”的值。但该期间仍将继续计时。因此，之后执行本指令时，“ET”将更新为正确的值。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值为FALSE。

Timer

在启动经过设定时间后，输出TRUE的定时器。设定时间单位和计时精度均为100ms。

指令	名称	FB/FUN	图形表现	ST表现
Timer	100ms定时器	FUN		Out:=Timer (In, PT, TimerDat, Q, ET);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	定时器输入	输入	TRUE ：定时器启动指示 FALSE：定时器复位指示	遵从数据类型	–	FALSE
PT	设定时间		从定时器启动到 “Q”变为TRUE的时间		ms	(*)
TimerDat	定时器状态	输入输出	定时器当前的状态	–	–	–
Out	返回值	输出	TRUE ：定时器输出ON FALSE：定时器输出OFF	遵从数据类型	–	–
Q	定时器输出		与 “Out”意义相同			
ET	剩余时间		剩余时间		ms	

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○																			
PT							○													
TimerDat	结构体_sTimer																			
Out	○																			
Q	○																			
ET							○													

功能

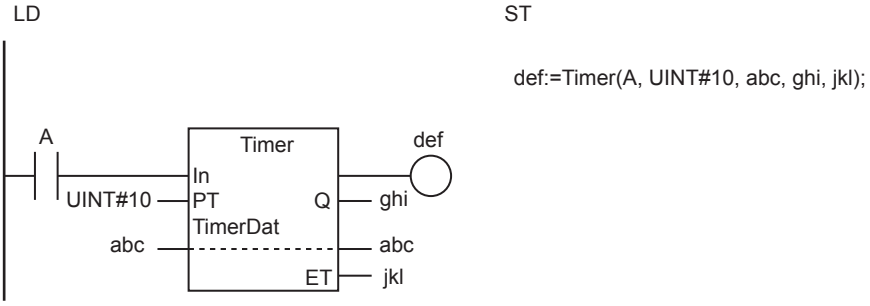
在启动经过设定时间后，输出TRUE的定时器。设定时间单位和计时单位均为100ms。

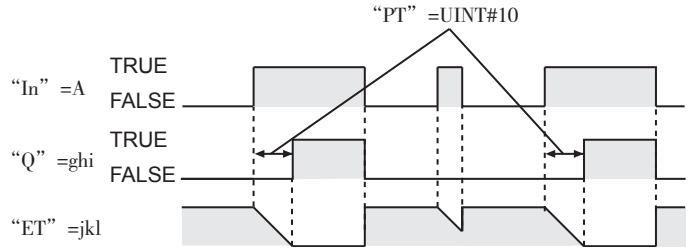
定时器输入 “In”的值变为FALSE时，定时器复位。为剩余时间 “ET”设定设定时间 “PT”，定时器输出 “Q”的值变为FALSE。
“In”从TRUE变为FALSE时，定时器启动。“ET”的值与时间同时减少。
“ET”的值到达0时，定时器输出 “Q”变为TRUE。此时，停止减少 “ET”的值。

定时器启动后，即使在 “ET”到达0之前，“In”变为FALSE时，定时器也将复位。

定时器状态 “TimerDat”的数据类型为结构体_sTimer。

“PT”=UINT#10时的示例和时序图如下所示。变量A变为TRUE的1000ms(1s)后，变量ghi的值变为TRUE。





参考

要进行更准确的计时，请使用 “TON指令(P.2-128)”，以100ns为单位进行计时。TON指令在执行指令时以100ns为单位进行计时，因此可比Timer指令更准确地进行计时。但Timer指令的指令执行时间较短。

使用注意事项

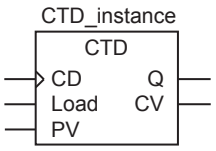
- 在记述本指令的POU的开头进行计时。因此，在相同POU内的无论何处执行本指令，“ET”的值均相同。
- 相对于 “PT”，“Q”变为TRUE的计时误差为+1个任务周期。
  上述范围的详情如下。
  - 在各任务周期判断 “ET”计时是否达到 “PT”，如果是，则立即延迟1个任务周期。
- “TimerDat”为输入输出变量，无需传输值。请确保结构体_sTimer所需的存储区域，传输至本指令。
- 请勿变更 “TimerDat”的内容。
- 如果开始运行时 “In”的值已为TRUE，则从该时刻开始计时。
- 变更了 “PT”的值后，将在之后使定时器复位时反映。计时过程中不会反映。
- 本指令存在于主站控制区域，通过主站控制执行复位时，复位定时器。“ET”中设定“PT”的值，“Q”的值变为FALSE。
- 因执行JMP系统指令(JMP指令等)而未执行本指令时，不更新 “ET”的值。但该期间仍将继续计时。因此，之后执行本指令时，“ET”将更新为正确的值。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则“Q”和“Out”的值变为FALSE。

计数器指令

指令	名称	页码
CTD	减法计数器	2–146
CTD_**	减法计数器组	2–148
CTU	加法计数器	2–151
CTU_**	加法计数器组	2–153
CTUD	可逆计数器	2–156
CTUD_**	可逆计数器组	2–160

CTD

每次输入计数器输入信号时进行减法运算的计数器。预设值、计数值的数据类型为INT。

指令	名称	FB/FUN	图形表现	ST表现
CTD	减法计数器	FB		CTD_instance (CD, Load, PV, Q, CV);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
CD	计数器输入	输入	计数器输入	遵从数据类型	–	FALSE
Load*1	加载信号		TRUE：向 “CV” 加载 “PV”			
PV	预设值		计数器的预设值	0 ~ 32767		0
Q	计数器输出	输出	TRUE ：计数器输出ON FALSE：计数器输出OFF	遵从数据类型	–	–
CV	计数值		计数器的当前值	0 ~ 32767		

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Load缩记为LD。例如，可记为：CTD_instance (CD:=A, LD:=abc, PV:=INT#5, Q=>def, CV=>ghi);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
CD	○																			
Load	○																			
PV											○									
Q	○																			
CV											○									

功能

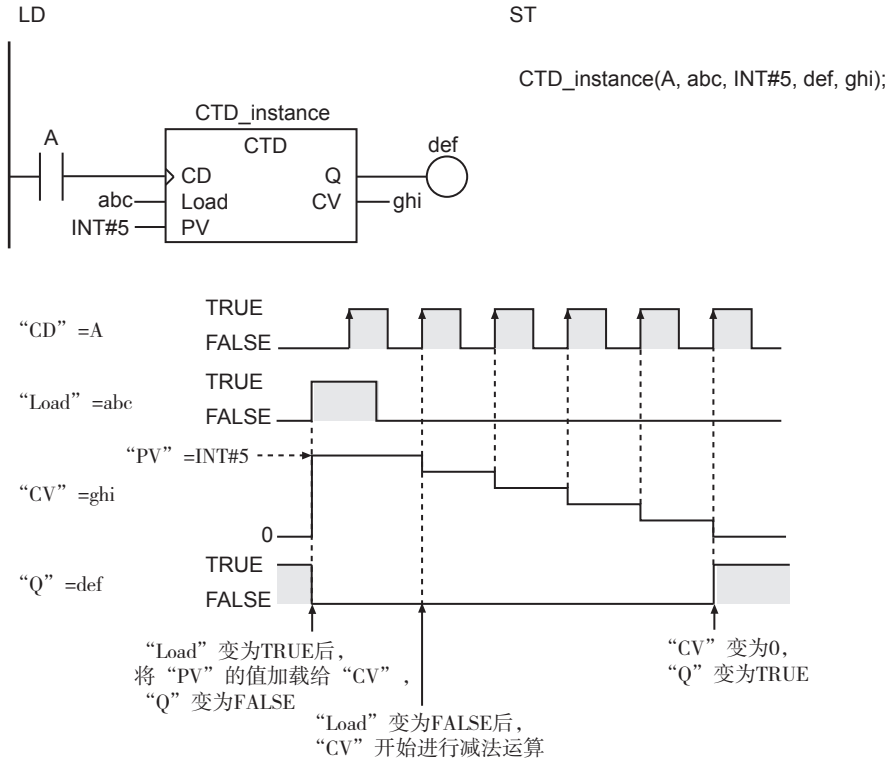
减法计数器。预设值、计数值的数据类型为INT。

加载信号 “Load” 设置为TRUE时，将预设值 “PV” 的值加载到计数值 “CV” 中，计数器输出 “Q” 变为FALSE。
计数器输入信号 “CD” 处于上升沿时，使 “CV” 进行减法运算。“CV” 的值小于0时，“Q” 的值变为TRUE。

“CV” 的值小于0时，即使 “CD” 处于上升沿，“CV” 也不会变化。

“Load” 为TRUE期间，忽略 “CD”。“CV” 不进行减法运算。

“PV”=INT#5时的示例和时序图如下所示。



参考

- 如果需要计数器每次输入计数器输入信号时进行加法运算，
  请使用 “CTU指令(P.2–151)”。
- 如果需要计数器同时进行加法运算和减法运算，请使用 “CTUD指令(P.2–156)”。

使用注意事项

- 倒计数结束后要使计数器再次启动时，请先将 “Load”的值设为TRUE后再设为FALSE。
- 为 “PV”设定了负数时，在 “Load”的值变为TRUE时，将 “PV”的值加载到 “CV”中。“CV”的值小于0，因此 “Q”的值会立即变为TRUE。之后，即使 “CD”变化，“CV”也不进行减法运算。
- “CD”的值为 FALSE 的状态下发生电源断开或由程序控制动作模式后，重新开始执行本指令时，如果“CD”的值变为TRUE，则 “CV”进行1次减法运算。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值变为FALSE。

CTD_**

每次输入计数器输入信号时进行减法运算的计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。

指令	名称	FB/FUN	图形表现	ST表现
CTD_**	减法计数器组	FB	CTD_**_instance CTD_** CD Q Load CV PV **为DINT、LINT、UDINT、ULINT中的任意一个	CTD_**_instance (CD, Load, PV, Q, CV); **为DINT、LINT、UDINT、ULINT中的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
CD	计数器输入	输入	计数器输入	遵从数据类型	–	FALSE
Load*1	加载信号		TRUE：向 “CV” 加载 “PV”			
PV	预设值		计数器的预设值	遵从数据类型*2		0
Q	计数器输出	输出	TRUE ：计数器输出ON FALSE：计数器输出OFF	遵从数据类型	–	–
CV	计数值		计数器的当前值	遵从数据类型		

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Load缩记为LD。例如，可记为：CTD_LINT_instance(CD:=A, LD:=abc, PV:=LINT#5, Q=>def, CV=>ghi);。

*2 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
CD	○																			
Load	○																			
PV								○	○			○	○							
Q	○																			
CV	与 “PV” 相同的数据类型																			

功能

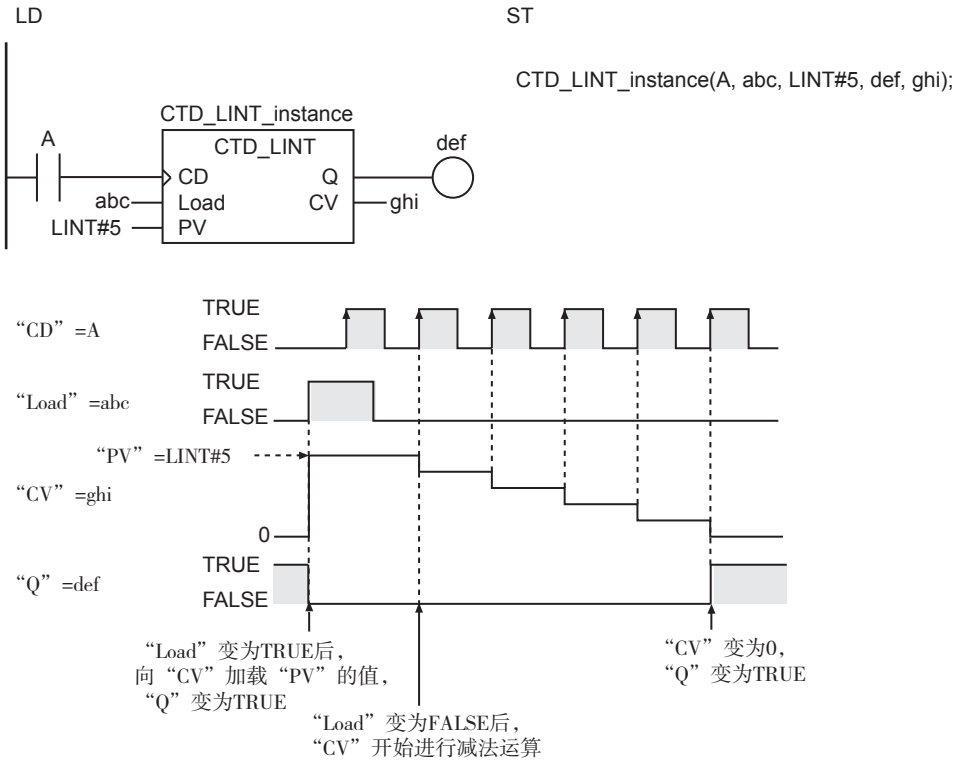
减法计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。
指令名称因 “PV”和 “CV”的数据类型而异。例如，两者均为LINT型时，指令名称为CTD_LINT。

加载信号 “Load”设置为TRUE时，将预设值 “PV”的值加载到计数值 “CV”中，计数器输出 “Q”变为FALSE。
计数器输入信号 “CD”处于上升沿时，使 “CV”进行减法运算。“CV”的值小于0时，“Q”的值变为TRUE。

“CV”的值小于0时，即使 “CD”处于上升沿，“CV”也不会变化。

“Load”为TRUE期间，忽略 “CD”。“CV”不进行减法运算。

CTD_LINT指令下，“PV”=LINT#5时的示例和时序图如下所示。



参考

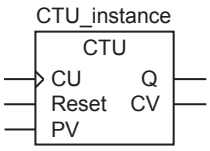
- 如果需要计数器每次输入计数器输入信号时进行加法运算，
  请使用 “CTU指令(P.2–151)”。
- 如果需要计数器同时进行加法运算和减法运算，请使用 “CTUD指令(P.2–156)”。

使用注意事项

- 倒计数结束后要使计数器再次启动时，请先将 “Load” 的值设为TRUE后再设为FALSE。
- 请将 “PV” 和 “CV” 的数据类型设为相同。
- 为 “PV” 设定了负数时，在 “Load” 的值变为TRUE时，将 “PV” 的值加载到 “CV” 中。“CV” 的值小于0，因此 “Q” 的值会立即变为TRUE。之后，即使 “CD” 变化，“CV” 也不进行减法运算。
- “CD” 的值为 FALSE 的状态下发生电源断开或由程序控制动作模式后，重新开始执行本指令时，如果 “CD” 的值变为TRUE，则 “CV” 进行1次减法运算。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q” 的值变为FALSE。

CTU

每次输入计数器输入信号时进行加法运算的计数器。预设值、计数值的数据类型为INT。

指令	名称	FB/FUN	图形表现	ST表现
CTU	加法计数器	FB		CTU_instance (CU, Reset, PV, Q, CV);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
CU	计数器输入	输入	计数器输入	遵从数据类型	–	FALSE
Reset*1	复位信号		TRUE：将 “CV” 复位为0			
PV	预设值		计数器的预设值	0 ~ 32767		0
Q	计数器输出	输出	TRUE：计数器输出ON FALSE：计数器输出OFF	遵从数据类型	–	–
CV	计数值		计数器的当前值	0 ~ 32767		

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Reset缩记为R。例如，可记为：CTU_instance (CU:=A, R:=abc, PV:=INT#5, Q=>def, CV=>ghi);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
CU	○																			
Reset	○																			
PV											○									
Q	○																			
CV											○									

功能

加法计数器。预设值、计数值的数据类型为INT。

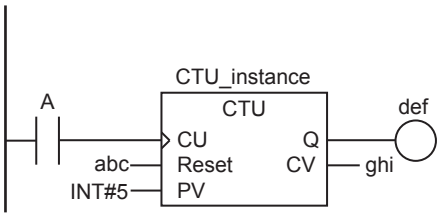
复位信号 “Reset” 设置为TRUE时，计数值 “CV” 的值变为0，计数器输出 “Q” 变为FALSE。
计数器输入信号 “CU” 处于上升沿时，使 “CV” 进行加法运算。“CV” 的值大于预设值 “PV” 的值时，“Q” 的值变为TRUE。

“CV” 的值大于 “PV” 的值时，即使输入更大的 “CU” 的值，“CV” 也不会变化。

“Rreset” 为TRUE期间，忽略 “CU”。“CV” 不进行加法运算。

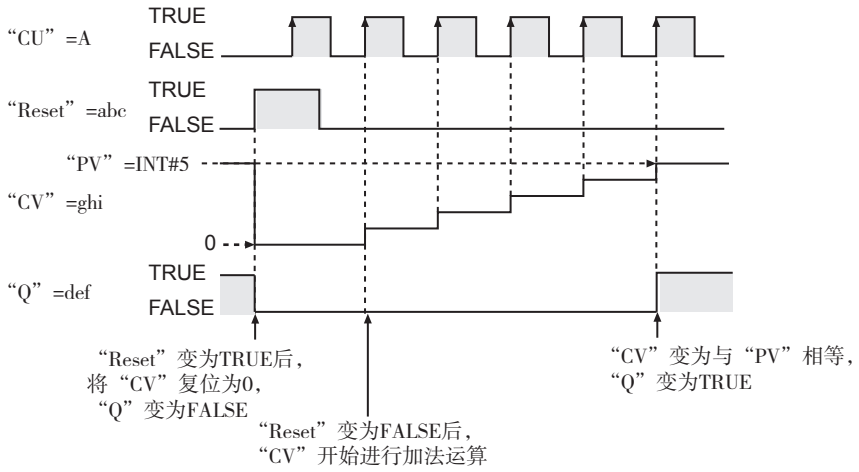
“PV”=INT#5时的示例和时序图如下所示。

LD



ST

```
CTU_instance(A, abc, INT#5, def, ghi);
```



参考

- 如果需要计数器每次输入计数器输入信号时进行减法运算，
  请使用 “CTD指令(P.2-146)”。
- 如果需要计数器同时进行加法运算和减法运算，请使用 “CTUD指令(P.2-156)”。

使用注意事项

- 计数结束后要使计数器再次启动，请先将 “Reset”的值设为TRUE后再设为FALSE。
- 为 “PV”设定了负数时，在 “Reset”的值变为TRUE时，“CV”的值变为0。“CV”的值大于 “PV”的值，因此 “Q”的值会立即变为TRUE。之后，即使 “CU”变化，“CV”也不进行加法运算。
- “Reset”的值为FALSE的状态下，“PV”的值如有变更，则其动作如下所示。

“PV”的值	含义
大于该时刻的 “CV”	继续计数。
小于该时刻的 “CV”	计数结束。“Q”的值变为TRUE。“CV”的值保持该时刻的值不变。

- “CU”的值为FALSE状态下发生电源断开或由程序控制动作模式后，重新开始执行本指令时，如果“CU”的值变为TRUE，则 “CV”进行1次加法运算。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值变为FALSE。

CTU_**

每次输入计数器输入信号时进行加法运算的计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。

指令	名称	FB/FUN	图形表现	ST表现
CTU_**	加法计数器组	FB	CTU_**_instance CTU_** CU Q Reset CV PV **为DINT、LINT、UDINT、ULINT中的任意一个	CTU_**_instance (CU, Reset, PV, Q, CV); **为DINT、LINT、UDINT、ULINT中的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
CU	计数器输入	输入	计数器输入	遵从数据类型	–	FALSE
Reset*1	复位信号		TRUE：将 “CV”复位为0			
PV	预设值		计数器的预设值	遵从数据类型*2		0
Q	计数器输出	输出	TRUE：计数器输出ON FALSE：计数器输出OFF	遵从数据类型	–	–
CV	计数值		计数器的当前值	遵从数据类型*2		

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Reset缩记为R。例如，可记为：CTU_LINT_instance(CU:=A, R:=abc, PV:=LINT#5, Q=>def, CV=>ghi);。

*2 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
CU	○																			
Reset	○																			
PV								○	○			○	○							
Q	○																			
CV	与 “PV”相同的数据类型																			

功能

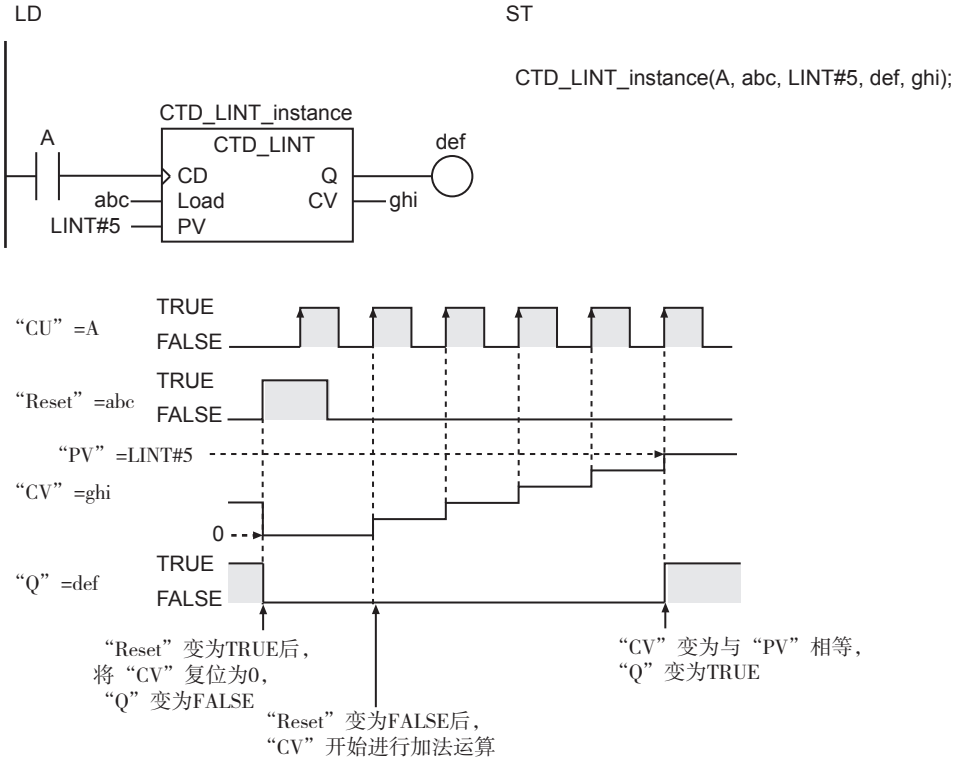
加法计数器。预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。
指令名称因 “PV” 和 “CV” 的数据类型而异。例如，两者均为LINT型时，指令名称为CTU_LINT。

复位信号 “Reset” 设置为TRUE时，计数值 “CV” 的值变为0，计数器输出 “Q” 变为FALSE。
计数器输入信号 “CU” 处于上升沿时，使 “CV” 进行加法运算。“CV” 的值大于预设值 “PV” 的值时，“Q” 的值变为TRUE。

“CV” 的值大于 “PV” 的值时，即使输入更大的 “CU” 的值，“CV” 也不会变化。

“Rreset” 为TRUE期间，忽略 “CU”。“CV” 不进行加法运算。

CTU_LINT指令下，“PV” =LINT#5时的示例和时序图如下所示。



参考

- 如果需要计数器每次输入计数器输入信号时进行减法运算，
  请使用 “CTD指令(P.2-146)”。
- 如果需要计数器同时进行加法运算和减法运算，请使用 “CTUD指令(P.2-156)”。

使用注意事项

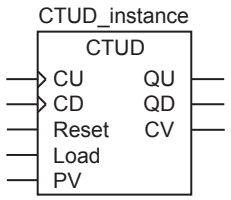
- 计数结束后要使计数器再次启动，请先将 “Reset”的值设为TRUE后再设为FALSE。
- 为 “PV”设定了负数时，在 “Reset”的值变为TRUE时，“CV”的值变为0。“CV”的值大于 “PV”的值，因此 “Q”的值会立即变为TRUE。之后，即使 “CU”变化，“CV”也不进行加法运算。
- 请将 “PV”和 “CV”的数据类型设为相同。
- “Reset”的值为FALSE的状态下，“PV”的值如有变更，则其动作如下所示。

“PV”的值	含义
大于该时刻的 “CV”	继续计数。
小于该时刻的 “CV”	计数结束。“Q”的值变为TRUE。“CV”的值保持该时刻的值不变。

- “CU”的值为FALSE状态下发生电源断开或由程序控制动作模式后，重新开始执行本指令时，如果“CU”的值变为TRUE，则 “CV”进行1次加法运算。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Q”的值变为FALSE。

CTUD

根据加法计数器输入和减法计数器输入进行加减法运算的计数器。预设值、计数值的数据类型为INT。

指令	名称	FB/FUN	图形表现	ST表现
CTUD	可逆计数器	FB		CTUD_instance (CU, CD, Reset, Load, PV, QU, QD, CV);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
CU	加法计数器输入	输入	加法计数器输入	遵从数据类型	–	FALSE
CD	减法计数器输入		减法计数器输入			
Reset*1	复位信号		TRUE：将 “CV”复位为0			
Load*1	加载信号		TRUE：向 “CV”加载 “PV”。			
PV	预设值		加法计数器的计数结束值 减法计数器的初始值	0 ~ 32767		0
QU	加法计数器输出	输出	TRUE： 　加法计数器输出ON FALSE： 　加法计数器输出OFF	遵从数据类型	–	–
QD	减法计数器输出		TRUE： 　减法计数器输出ON FALSE： 　减法计数器输出OFF			
CV	计数值		计数器的当前值	0 ~ 32767		

*1　在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Reset缩记为R，将Load缩记为LD。例如，可记为：CTUD_instance(CU:=A, CD:=B, R:=abc, LD:=def, PV:=INT#3, QU=>ghi, QD=>jkl, CV=>mno);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
CU	○																			
CD	○																			
Reset	○																			
Load	○																			
PV											○									
QU	○																			
QD	○																			
CV											○									

功能

根据加法计数器输入信号和减法计数器输入信号进行加减法运算的计数器。
兼具加法计数器和减法计数器两者的功能。
预设值、计数值的数据类型为INT。

加法计数器的功能

复位信号 “Reset”设置为TRUE时，计数值 “CV”的值变为0，加法计数器输出 “QU”变为FALSE。
加法计数器输入信号 “CU”处于上升沿时，使 “CV”进行加法运算。“CV”的值大于预设值 “PV”的值时，“QU”的值变为TRUE。
“CV”的值大于 “PV”的值时，即使输入更大的 “CU”的值，“CV”也不会变化。

减法计数器的功能

加载信号“Load”设置为TRUE时，将预设值“PV”的值加载到计数值“CV”中，减法计数器输出“QD”变为FALSE。
减法计数器输入信号 “CD”处于上升沿时，使 “CV”进行减法运算。“CV”的值小于0时，“QD”的值变为TRUE。
“CV”的值小于0时，即使输入更大的 “CD”的值，“CV”也不会变化。

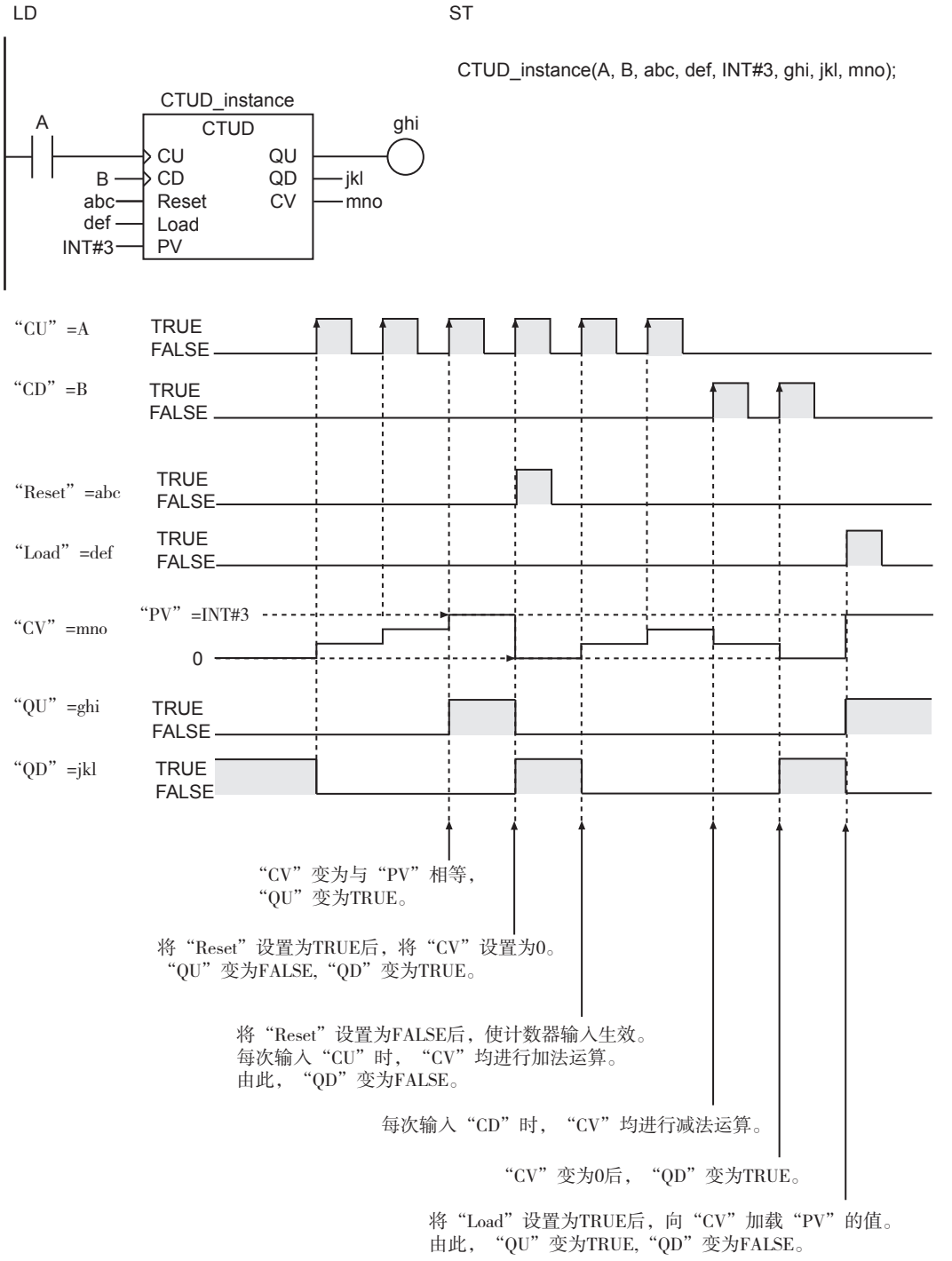
加法计数器、减法计数器的共同功能

“Load”或 “Reset”为TRUE期间，忽略 “CU”和 “CD”。“CV”不进行加减法运算。
“CU”和 “CD”同时处于上升沿时，“CV”不会变化。
“Reset”和 “Load”均为TRUE时，“Reset”优先，“CV”的值变为0。
“Reset”设置为TRUE时，“CV”的值变为0，因此 “QD”的值变为TRUE。
“Load”设置为TRUE时，“CV”的值与 “PV”相等，因此 “QU”的值将变为TRUE。

“Reset”、“Load”、“CV”、“QU”、“QD”之间的关系如下所示。“PV”的值设置为大于0。

“Reset”	“Load”	“CV”	“QU”	“QD”	动作
FALSE	FALSE	0以下	FALSE	TRUE	仅作为加法计数器动作。 • “CU”处于上升沿时，“CV”进行加法运算。 “CD”处于上升沿时，不进行减法运算。
		0与 “PV”之间	FALSE	FALSE	作为加法计数器、减法计数器动作。 • “CU”处于上升沿时，“CV”进行加法运算； “CD”处于上升沿时，“CV”进行减法运算。
		大于 “PV”	TRUE	FALSE	仅作为减法计数器动作。 • “CD”处于上升沿时，“CV”进行减法运算。 “CU”处于上升沿时，不进行加法运算。
TRUE	FALSE	0	FALSE	TRUE	加法计数器的复位状态。 • “CV”的值设置为0。
FALSE	TRUE	“PV”	TRUE	FALSE	减法计数器的复位状态。 • “CV”的值设置为 “PV”。
TRUE	TRUE	0	FALSE	TRUE	加法计数器的复位状态。“Reset”优先于 “Load”。 • “CV”的值设置为0。

“PV”=INT#3时的示例和时序图如下所示。



参考

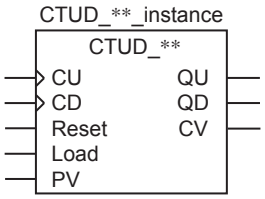
仅减法计数器或加法计数器就足够时，也有 “CTD指令(P.2–146)”、“CTU指令(P.2–151)”。

使用注意事项

- 请注意，为使加法计数器复位，将 “Reset”的值设为TRUE时，“QU”的值变为FALSE，同时 “QD”的值变为TRUE。
- 请注意，为使减法计数器复位，将 “Load”的值设为TRUE时，“QD”的值变为FALSE，同时 “QU”的值变为TRUE。
- 为 “PV”设定了负数时，在 “Load”的值变为TRUE时，将 “PV”的值加载到 “CV”中。“CV”的值小于0，因此 “QD”的值会立即变为TRUE。之后，即使 “CD”变化，“CV”也不进行减法运算。“Reset”的值变为TRUE时，“CV”的值变为0。“CV”的值大于 “PV”的值，因此 “QU”的值会立即变为TRUE。之后，即使 “CU”变化，“CV”也不进行加法运算。
- 在执行本指令的过程中，可变更 “PV”的值。此时，如果变更后的 “PV”值小于该时刻的 “CV”，则 “QU”的值立即变为TRUE。
- “CU”或“CD”的值为FALSE状态下发生电源断开或由程序控制动作模式后，重新开始执行本指令时，如果 “CU”或 “CD”的值变为TRUE，则 “CV”进行1次加法或减法运算。

CTUD_**

根据加法计数器输入和减法计数器输入进行加减法运算的计数器。预设值、计数值的数据类型为 DINT、LINT、UDINT、ULINT中的任意一种。

指令	名称	FB/FUN	图形表现	ST表现
CTUD_**	可逆计数器组	FB	 **为DINT、LINT、UDINT、ULINT中的任意一个	CTUD_**_instance (CU, CD, Reset, Load, PV, QU, QD, CV); *为DINT、LINT、UDINT、ULINT中的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
CU	加法计数器输入	输入	加法计数器输入	遵从数据类型	–	FALSE
CD	减法计数器输入		减法计数器输入			
Reset*1	复位信号		TRUE：将 “CV”复位为0			
Load*1	加载信号		TRUE：向 “CV”加载 “PV”。			
PV	预设值		加法计数器的计数结束值 减法计数器的初始值	遵从数据类型*2		0
QU	加法计数器输出	输出	TRUE： 加法计数器输出ON FALSE： 加法计数器输出OFF	遵从数据类型	–	–
QD	减法计数器输出		TRUE： 减法计数器输出ON FALSE： 减法计数器输出OFF			
CV	计数值		计数器的当前值	遵从数据类型*2		

*1 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Reset缩记为R，将Load 缩记为 LD。例如，可记为：CTUD_LINT_instance(CU:=A, CD:=B, R:=abc, LD:=def, PV:=LINT#3, QU=>ghi, QD=>jkl, CV=>mno);。

*2 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
CU	○																			
CD	○																			
Reset	○																			

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Load	○																			
PV								○	○			○	○							
QU	○																			
QD	○																			
CV	与 “PV” 相同的数据类型																			

功能

根据加法计数器输入信号和减法计数器输入信号进行加减法运算的计数器。
兼具加法计数器和减法计数器两者的功能。
预设值、计数值的数据类型为DINT、LINT、UDINT、ULINT中的任意一种。
指令名称因 “PV” 和 “CV” 的数据类型而异。例如，两者均为LINT型时，指令名称为CTUD_LINT。

加法计数器的功能

复位信号 “Reset” 设置为TRUE时，计数值 “CV” 的值变为0，加法计数器输出 “QU” 变为FALSE。
加法计数器输入信号 “CU” 处于上升沿时，使 “CV” 进行加法运算。“CV” 的值大于预设值 “PV” 的值时，“QU” 的值变为TRUE。
“CV” 的值大于 “PV” 的值时，即使输入更大的 “CU” 的值，“CV” 也不会变化。

减法计数器的功能

加载信号 “Load” 设置为TRUE时，将预设值 “PV” 的值加载到计数值 “CV” 中，减法计数器输出 “QD” 变为FALSE。
减法计数器输入信号 “CD” 处于上升沿时，使 “CV” 进行减法运算。“CV” 的值小于0时，“QD” 的值变为TRUE。
“CV” 的值小于0时，即使输入更大的 “CD” 的值，“CV” 也不会变化。

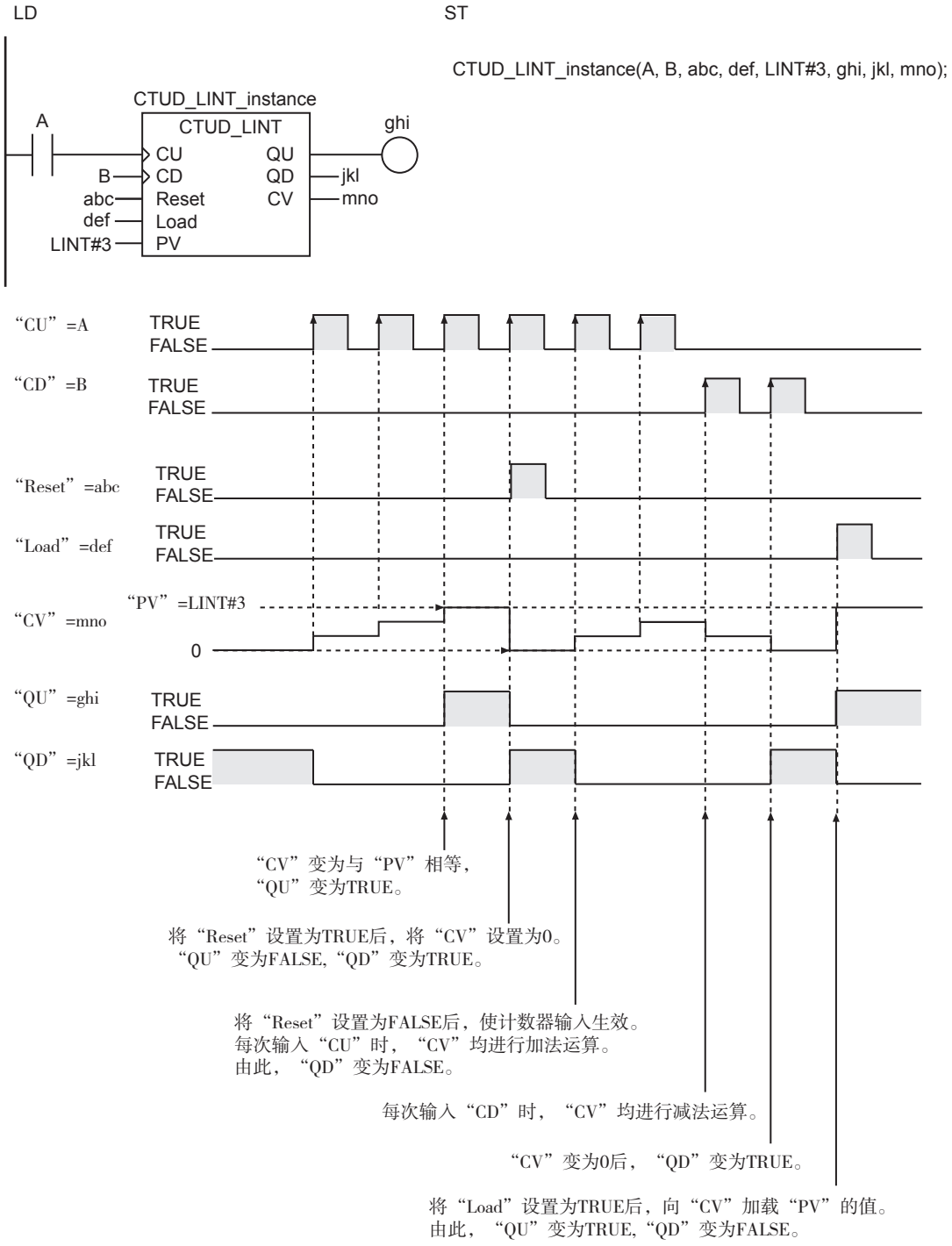
加法计数器、减法计数器的共同功能

“Load” 或 “Reset” 为TRUE期间，忽略 “CU” 和 “CD”。“CV” 不进行加减法运算。
“CU” 和 “CD” 同时处于上升沿时，“CV” 不会变化。
“Reset” 和 “Load” 均为TRUE时，“Reset” 优先，“CV” 的值变为0。
“Reset” 设置为TRUE时，“CV” 的值变为0，因此 “QD” 的值变为TRUE。
“Load” 设置为TRUE时，“CV” 的值与 “PV” 相等，因此 “QU” 的值将变为TRUE。

“Reset”、“Load”、“CV”、“QU”、“QD”之间的关系如下所示。“PV”的值设置为大于0。

“Reset”	“Load”	“CV”	“QU”	“QD”	动作
FALSE	FALSE	0以下	FALSE	TRUE	仅作为加法计数器动作。 • “CU”处于上升沿时，“CV”进行加法运算。 “CD”处于上升沿时，不进行减法运算。
		0与“PV”之间	FALSE	FALSE	作为加法计数器、减法计数器动作。 • “CU”处于上升沿时，“CV”进行加法运算； “CD”处于上升沿时，“CV”进行减法运算。
		大于 “PV”	TRUE	FALSE	仅作为减法计数器动作。 • “CD”处于上升沿时，“CV”进行减法运算。 “CU”处于上升沿时，不进行加法运算。
TRUE	FALSE	0	FALSE	TRUE	加法计数器的复位状态。 • “CV”的值设置为0。
FALSE	TRUE	“PV”	TRUE	FALSE	减法计数器的复位状态。 • “CV”的值设置为 “PV”。
TRUE	TRUE	0	FALSE	TRUE	加法计数器的复位状态。“Reset”优先于 “Load”。 • “CV”的值设置为0。

CTUD_LINT指令下，“PV”=LINT#3时的示例和时序图如下所示。



参考

仅减法计数器或加法计数器就足够时，也有 “CTD_**指令(P.2-148)”、“CTU_**指令(P.2-153)”。

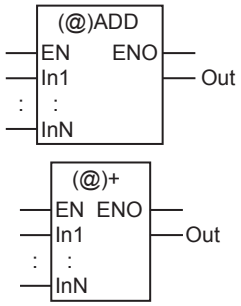
使用注意事项

- 请注意，为使加法计数器复位，将 “Reset”的值设为TRUE时，“QU”的值变为FALSE，同时 “QD”的值变为TRUE。
- 请注意，为使减法计数器复位，将 “Load”的值设为TRUE时，“QD”的值变为FALSE，同时 “QU”的值变为TRUE。
- 为 “PV”设定了负数时，在 “Load”的值变为TRUE时，将 “PV”的值加载到 “CV”中。“CV”的值小于0，因此 “QD”的值会立即变为TRUE。之后，即使 “CD”变化，“CV”也不进行减法运算。“Reset”的值变为TRUE时，“CV”的值变为0。“CV”的值大于 “PV”的值，因此 “QU”的值会立即变为TRUE。之后，即使 “CU”变化，“CV”也不进行加法运算。
- 在执行本指令的过程中，可变更 “PV”的值。此时，如果变更后的 “PV”值小于该时刻的 “CV”，则 “QU”的值立即变为TRUE。
- 请将 “PV”和 “CV”的数据类型设为相同。
- “CU”或“CD”的值为FALSE状态下发生电源断开或由程序控制动作模式后，重新开始执行本指令时，如果 “CU”或 “CD”的值变为TRUE，则 “CV”进行1次加法或减法运算。

算术指令

ADD，+

对整数、实数进行加法运算。或拼接字符串。

指令	名称	FB/FUN	图形表现	ST表现
ADD，+	加法	FUN		Out:=In1 + In2;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	加数	输入	相加数 梯形图：N为2 ~ 5 ST ：N为2*1	遵从数据类型	–	0*2
Out	输出值	输出	输出值	遵从数据类型	–	–

*1 如result := val1 + val2 + val3；所示，可将本指令作为运算符在1个表达式中多次记述。最多可在1个表达式中记述64个运算符。

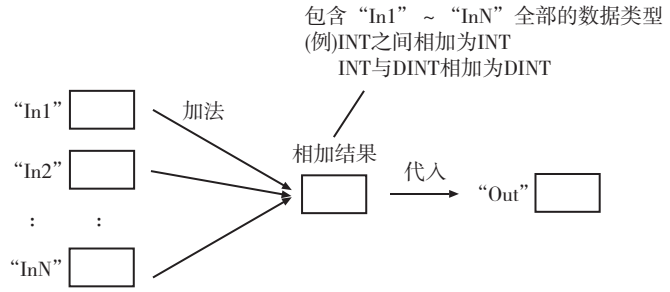
*2 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN						○	○	○	○	○	○	○	○	○	○					○
Out						○	○	○	○	○	○	○	○	○	○					○

功能

将2个以上5个以下的整数或实数的相加结果输出至输出值 “Out”。
ST时，将2个整数或实数的相加结果输出至输出值 “Out”。

加数 “In1” ~ “InN”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相加结果为DINT型。
关于数据类型的包含关系，请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的 “数据类型的等级表”及 “转换规则”。



发生溢出时的处理

“In1”～“InN”之和超过相加结果的数据类型的有效范围称为溢出。发生溢出时的“In1”～“InN”的数据类型、相加结果的数据类型、相加结果的值之间的关系如下所示。

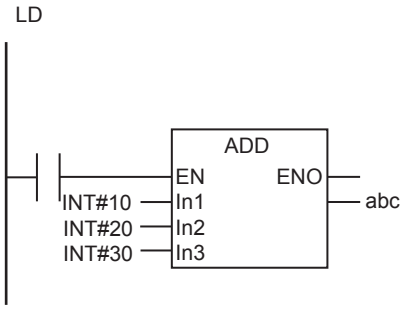
“In1”～“InN”的数据类型	相加结果的数据类型	相加结果的值
所有整数型	整数型	“In1”～“InN”之和中，可通过相加结果的数据类型的位数表现的值(*1)
任意一个为实数型	实数型	±∞(*2)

*1 例如，“In1”的值为INT#32767，“In2”的值为INT#3时，相加结果的数据类型为INT型。相加结果的值在两者之和32770的低16位为INT#–32766。

*2 “In1”～“InN”之和为正数时为+∞；为负数时为–∞。

记述示例

“In1”=INT#10、“In2”=INT#20、“In3”=INT#30时的示例如下所示。INT型变量abc的值为INT#60。



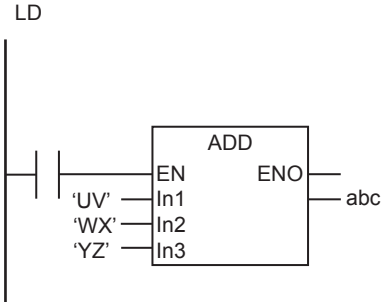
ST

```
abc:=INT#10+INT#20+INT#30;
```

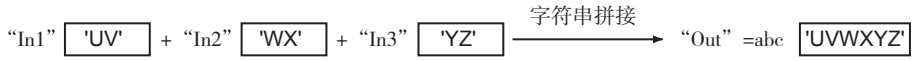
对“In1”～“InN”进行加法运算。
10+20+30=60，因此abc的值为INT#60。

字符串的拼接

“In1” ~ “InN”为STRING型时，拼接这些字符串。
“In1” ~ “InN”仅在使用梯形图程序时可指定STRING型。
“In1”='UV'、“In2”='WX'、“In3”='YZ'时的示例如下所示。STRING型变量abc的值为'UVWXYZ'。



拼接“In1” ~ “InN”的字符串。
'UV' + 'WX' + 'YZ' = 'UVWXYZ'，因此abc的值为'UVWXYZ'。



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时存在规格区别。规格区别如下所述。此外，梯形图程序的ADD指令与梯形图程序的+指令的规格完全相同。

规格项目	梯形图	ST
加数的最多个数	5	2*1
可否省略加数的输入参数	除InN连接的输入参数外可以省略	所有输入参数均不可省略
变量EN、ENO的存在与否	有	无
加数的数据类型均为整数时运算处理的位数	8,16,32,64*2	32,64*3
字符串可否拼接	可	不可
有无异常	拼接字符串时，拼接结果的大小超过1986字节时将发生异常	无

*1 如result := val1 + val2 + val3；所示，可将本指令作为运算符在1个表达式中多次记述。最多可在1个表达式中记述64个运算符。
*2 运算位数以加数中最大的数据类型为准。例如，对SINT型、INT型、DINT型进行加法运算时，根据DINT型的大小按照32位进行运算。
*3 加数中无LINT型及ULINT型时，则按照32位进行运算。例如，对多个SINT型进行加法运算时，也按照32位进行运算。加数中有LINT型或ULINT型时，则按照64位进行运算。

参考

- 运算实数时，请使用 “CheckReal指令(P.2-234)”判定 “Out”是否为+∞、−∞、非数。
- 通过ST程序拼接字符串时，请使用 “CONCAT指令(P.2-538)”。

使用注意事项

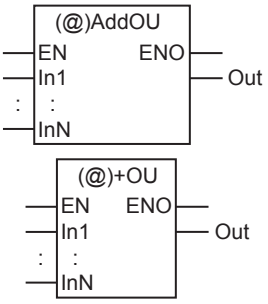
- “Out”的数据类型可与相加结果不同。请设为包含相加结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”的“数据类型的等级表”及 “转换规则”。
- 拼接字符串时，请将 “In1” ~ “InN”和 “Out”全部设为STRING型。
- 即使因加法而发生溢出或下溢，也不会发生异常。
- 因加法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。
- 与实数的+∞/–∞相对应的相加结果的值如下所示。

加法内容	相加结果的值
+∞和数值相加	+∞
–∞和数值相加	–∞
+∞和+∞相加	+∞
–∞和–∞相加	–∞
+∞和–∞相加	非数

- “In1” ~ “InN”中任意一个的值为非数时，相加结果的值为非数。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - 拼接字符串时，“In1” ~ “InN”中任意一个未以NULL字符结尾时。
  - 拼接字符串时，拼接的字符串大小超过1986字节时。

AddOU，+OU

对整数、实数进行加法运算。并且，对整数的相加结果进行溢出检查。

指令	名称	FB/FUN	图形表现	ST表现
AddOU，+OU	加法 (带溢出检查)	FUN		Out:=AddOU(In1, …, InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	加数	输入	相加数　N为2 ~ 5	遵从数据类型	–	0(*)
Out	输出值	输出	输出值	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN						○	○	○	○	○	○	○	○	○ (*)	○ (*)					
Out						○	○	○	○	○	○	○	○	○	○					

* “In1” ~ “InN”中任意一个的数据类型为实数型时，不进行溢出检查。

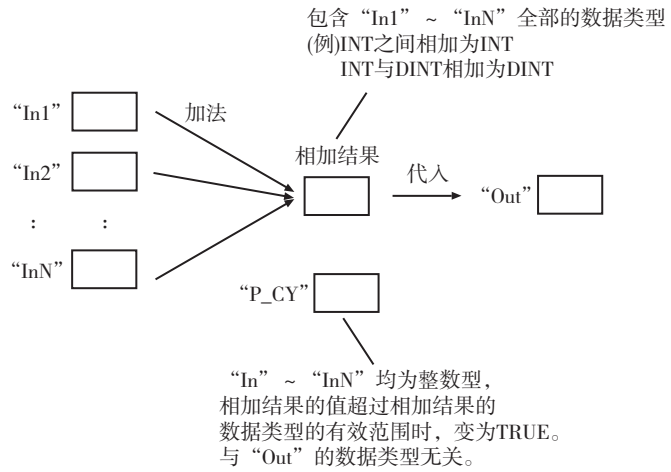
功能

将2个以上5个以下的整数或实数的相加结果输出至输出值 “Out”。

加数 “In1” ~ “InN”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相加结果为DINT型。
关于数据类型的包含关系，请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的 “数据类型的等级表”及 “转换规则”。

发生溢出时的处理

“In1” ~ “InN”之和超过相加结果的数据类型的有效范围称为溢出。
“In1” ~ “InN”均为整数型时，若发生溢出，则系统定义变量 “P_CY”(进位标志)的值变为TRUE。
“In1” ~ “InN”中任意一个为实数型时，不进行溢出检查。因此，“P_CY”的值不变。



发生溢出时的 “In1” ~ “InN”的数据类型、相加结果的数据类型、相加结果的值、“P_CY”的值之间的关系如下所示。

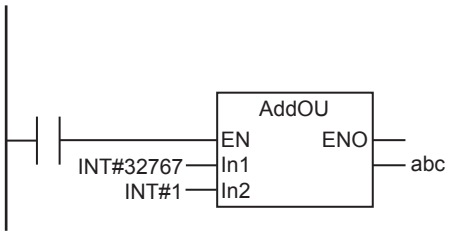
“In1” ~ “InN”的数据类型	相加结果的数据类型	相加结果的值	“P_CY”的值
所有整数型	整数型	“In1” ~ “InN”之和中，可通过相加结果的数据类型的位数表现的值(*1)	TRUE
任意一个为实数型	实数型	± ∞(*2)	不变化

*1 例如，“In1”的值为INT#32767，“In2”的值为INT#3时，相加结果的数据类型为INT型。相加结果的值在两者之和32770的低16位为INT#–32766。
*2 “In1” ~ “InN”之和为正数时为+∞；为负数时为–∞。

记述示例

“In1”=INT#32767、“In2”=INT#1，变量abc为INT型时的示例如下所示。
“In1”和 “In2”均为INT型，因此相加结果的数据类型为INT型。两者之和32768超过INT型的有效范围，因此 “P_CY”的值变为TRUE。INT型变量abc的值变为INT#–32768(32768的低16位的值)。

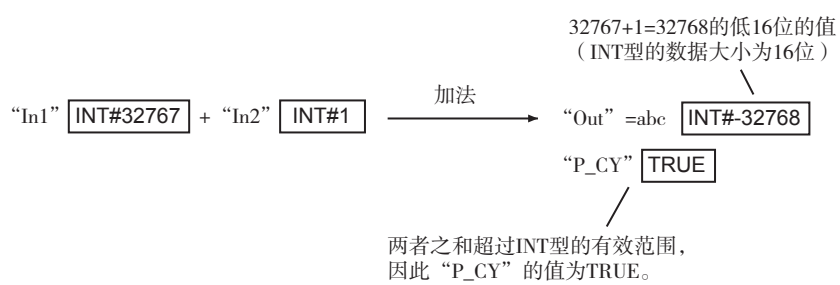
LD



ST

```
abc:=AddOU(INT#32767, INT#1);
```

对“In1” ~ “InN”进行加法运算。
两者之和32768超过INT型的有效范围，因此“P_CY”的值为TRUE。



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时无规格区别。此外，梯形图程序的AddOU指令与梯形图程序的+OU指令也无规格区别。

相关的系统定义变量

变量名称	名称	数据类型	内容
P_CY	进位(CY)标志	BOOL	TRUE ：整数运算有溢出 FALSE：整数运算无溢出

参考

- “Out”为实数型时，请使用 “CheckReal指令(P.2-234)”判定值是否为+∞、−∞、非数。
- 无需溢出检查时，请使用 “ADD，+指令(P.2-166)”。这样可缩短处理时间。

使用注意事项

- “Out”的数据类型可与相加结果不同。请设为包含相加结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的“数据类型的等级表”及 “转换规则”。
- 因加法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。
- 与实数的+∞/−∞相对应的相加结果的值如下所示。

加法内容	相加结果的值
+∞和数值相加	+∞
−∞和数值相加	−∞
+∞和+∞相加	+∞
−∞和−∞相加	−∞
+∞和−∞相加	非数

- “In1” ~ “InN”中任意一个的值为非数时，相加结果的值为非数。

SUB，–

对整数、实数进行减法运算。

指令	名称	FB/FUN	图形表现	ST表现
SUB，–	减法	FUN	(@)SUB EN ENO In1 Out In2 (@)- EN ENO In1 Out In2	Out:=In1 – In2;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减数	输入	被减数	遵从数据类型	–	0(*)
In2	减数		减数			
Out	输出值	输出	输出值	遵从数据类型	–	–

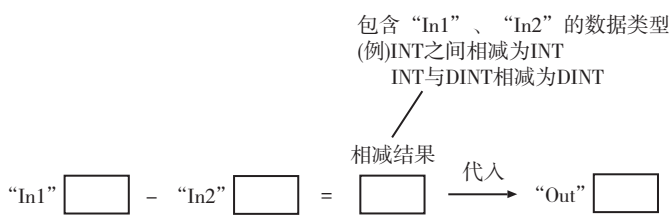
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1						○	○	○	○	○	○	○	○	○	○					
In2						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

功能

将被减数 “In1”减去减数 “In2”后的结果输出至输出值 “Out”。

“In1”和 “In2”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相减结果为DINT型。



关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的 “数据类型的等级表”及 “转换规则”。

发生溢出时的处理

“In1”和 “In2”之差超过相减结果的数据类型的有效范围称为溢出。发生溢出时的 “In1”、“In2”的数据类型、相减结果的数据类型、相减结果的值之间的关系如下所示。

“In1”、“In2”的数据类型	相减结果的数据类型	相减结果的值
所有整数型	整数型	“In1”和 “In2”之差中，可通过相减结果的数据类型的位数表现的值(*1)
任意一个为实数型	实数型	± ∞(*2)

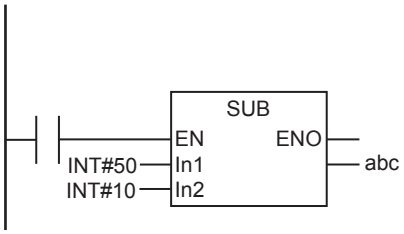
*1 例如，“In1”的值为INT#32767，“In2”的值为INT#-3时，相减结果的数据类型为INT型。相减结果的值在两者之差32770的低16位为INT#-32766。

*2 “In1”和 “In2”之差为正数时变为+∞、负数时变为-∞。

记述示例

“In1” =INT#50、“In2” =INT#10时的示例如下所示。INT型变量abc的值为INT#40。

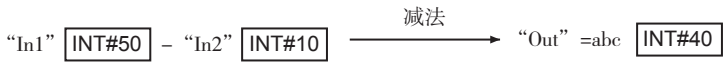
LD



ST

```
abc:=INT#50-INT#10;
```

将“In1”减去“In2”。
50-10=40，因此abc的值为INT#40。



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时存在规格区别。规格区别如下所述。此外，梯形图程序的SUB指令与梯形图程序的 - 指令的规格完全相同。

规格项目	梯形图	ST
变量EN、ENO的存在与否	有	无
被减数和减数的数据类型为整数时运算处理的位数	8,16,32,64*1	32,64*2

*1 运算位数以被减数和减数中最大的数据类型为准。例如，对SINT型和DINT型进行减法运算时，根据DINT型的大小按照32位进行运算。

*2 被减数和减数中无LINT型及ULINT型时，则按照32位进行运算。例如，对多个SINT型进行减法运算时，也按照32位进行运算。被减数和减数中有LINT型或ULINT型时，则按照64位进行运算。

参考

运算实数时，请使用 “CheckReal指令(P.2–234)” 判定 “Out”是否为+∞、–∞、非数。

使用注意事项

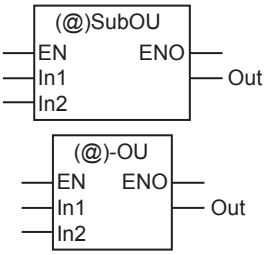
- “Out”的数据类型可与相减结果不同。请设为包含相减结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”的“数据类型的等级表”及 “转换规则”。
- 即使因减法而发生溢出或下溢，也不会发生异常。
- 因减法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。
- 与实数的+∞/–∞相对应的相减结果如下所示。

减法内容	相减结果
(+∞) – (数值)	+∞
(数值) – (+∞)	–∞
(–∞) – (数值)	–∞
(数值) – (–∞)	+∞
(+∞) – (+∞)	非数
(+∞) – (–∞)	+∞
(–∞) – (+∞)	–∞
(–∞) – (–∞)	非数

- “In1”、“In2”中任意一个的值为非数时，相减结果的值为非数。

SubOU，–OU

对整数、实数进行减法运算。并且，对整数的相减结果进行溢出检查。

指令	名称	FB/FUN	图形表现	ST表现
SubOU，–OU	减法 (带溢出检查)	FUN		Out:=SubOU(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减数	输入	被减数	遵从数据类型	–	0(*)
In2	减数		减数			
Out	输出值	输出	输出值	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1						○	○	○	○	○	○	○	○	○ (*)	○ (*)					
In2						○	○	○	○	○	○	○	○	○ (*)	○ (*)					
Out						○	○	○	○	○	○	○	○	○	○					

* “In1”、“In2”中任意一个的数据类型为实数型时，不进行溢出检查。

功能

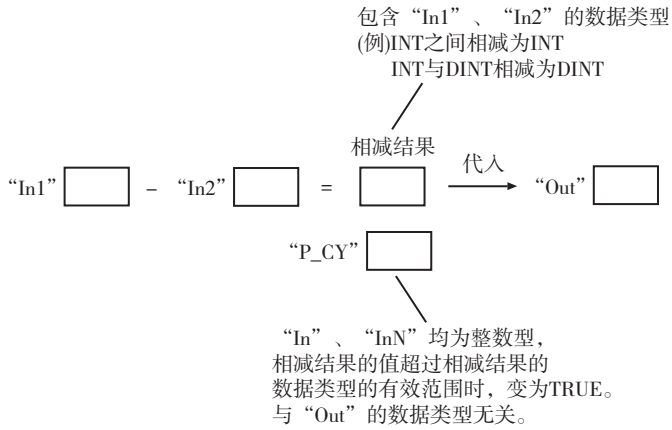
将被减数 “In1”减去减数 “In2”后的结果输出至输出值 “Out”。

“In1”和 “In2”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相减结果为DINT型。

关于数据类型的包含关系，请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)” 或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)” 的 “数据类型的等级表”及 “转换规则”。

发生溢出时的处理

“In1”和 “In2”之差超过相减结果的数据类型的有效范围称为溢出。“In1”和 “In2”均为整数型时，若发生溢出，则系统定义变量 “P_CY”(进位标志)的值变为TRUE。
“In1”、“In2”中任意一个为实数型时，不进行溢出检查。因此，“P_CY”的值不变。



发生溢出时的 “In1”、“In2”的数据类型、相减结果的数据类型、相减结果的值、“P_CY”的值之间的关系如下所示。

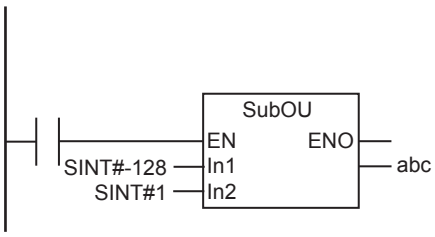
“In1”、“In2”的数据类型	相减结果的数据类型	相减结果的值	“P_CY”的值
所有整数型	整数型	“In1”和“In2”之差中，可通过相减结果的数据类型的位数表现的值(*1)	TRUE
任意一个为实数型	实数型	± ∞ (*2)	不变化

*1 例如，“In1”的值为INT#32767，“In2”的值为INT#–3时，相减结果的数据类型为INT型。相减结果的值在两者之差32770的低16位为INT#–32766。
*2 “In1”和 “In2”之差为正数时变为+ ∞ 、负数时变为– ∞ 。

记述示例

“In1”=SINT#–128，“In2”=SINT#1，变量abc为SINT型时的示例如下所示。
“In1”和 “In2”均为SINT型，因此相减结果的数据类型为SINT型。两者之差–129超过SINT型的有效范围，因此 “P_CY”的值变为TRUE。SINT型变量abc的值变为SINT#127(–129的低8位的值)。

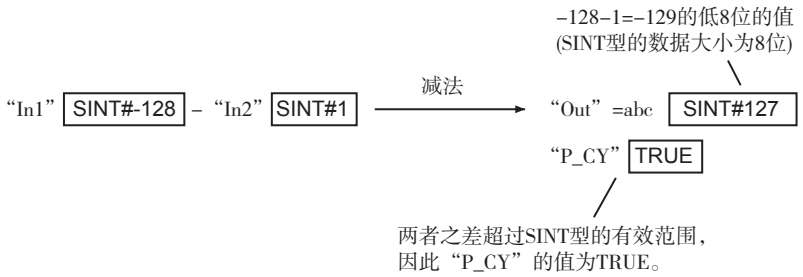
LD



ST

```
abc:=SubOU(SINT#-128, SINT#1);
```

将“In1”减去“In2”。
两者之差–129超过SINT型的有效范围，因此“P_CY”的值为TRUE。



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时无规格区别。此外，梯形图程序的SubOU指令与梯形图程序的–OU指令也无规格区别。

相关的系统定义变量

变量名称	名称	数据类型	内容
P_CY	进位(CY)标志	BOOL	TRUE ：整数运算有溢出 FALSE：整数运算无溢出

参考

- 运算实数时，请使用 “CheckReal指令(P.2–234)”判定 “Out”是否为+∞、–∞、非数。
- 无需溢出检查时，请使用 “SUB，–指令(P.2–173)”。这样可缩短处理时间。

使用注意事项

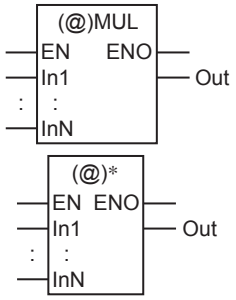
- “Out”的数据类型可与相减结果不同。请设为包含相减结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”的“数据类型的等级表”及 “转换规则”。
- 因减法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。
- 与实数的+∞/–∞相对应的相减结果如下所示。

减法内容	相减结果
(+∞) – (数值)	+∞
(数值) – (+∞)	–∞
(–∞) – (数值)	–∞
(数值) – (–∞)	+∞
(+∞) – (+∞)	非数
(+∞) – (–∞)	+∞
(–∞) – (+∞)	–∞
(–∞) – (–∞)	非数

- “In1”、“In2”中任意一个的值为非数时，相减结果的值为非数。

MUL，*

对整数、实数进行乘法运算。

指令	名称	FB/FUN	图形表现	ST表现
MUL，*	乘法	FUN		Out:=In1 * In2;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	乘数	输入	待乘数 梯形图：N为2 ~ 5 ST　　：N为2*1	遵从数据类型	–	1*2
Out	输出值	输出	输出值	遵从数据类型	–	–

*1 如result := val1 * val2 * val3；所示，可将本指令作为运算符在1个表达式中多次记述。最多可在1个表达式中记述64个运算符。

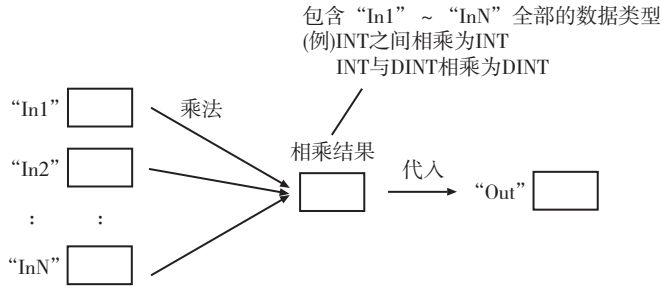
*2 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

功能

将2个以上5个以下的整数或实数的相乘结果输出至输出值 “Out”。
ST时，将2个整数或实数的相乘结果输出至输出值 “Out”。

乘数 “In1” ~ “InN”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相乘结果为DINT型。
关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)” 或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)” 的 “数据类型的等级表”及 “转换规则”。



发生溢出时的处理

“In1” ~ “InN”之积超过相乘结果的数据类型的有效范围称为溢出。发生溢出时的 “In1” ~ “InN”的数据类型、相乘结果的数据类型、相乘结果的值之间的关系如下所示。

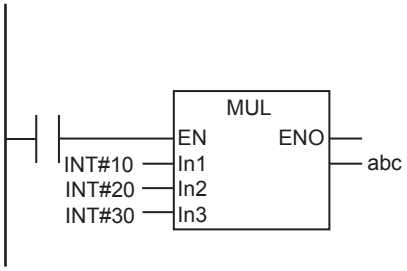
“In1” ~ “InN”的数据类型	相乘结果的数据类型	相乘结果的值
所有整数型	整数型	“In1” ~ “InN”之积中，可通过相乘结果的数据类型的位数表现的值(*1)
任意一个为实数型	实数型	± ∞(*2)

*1 例如，“In1”的值为INT#16384，“In2”的值为INT#2时，相乘结果的数据类型为INT型。相乘结果的值在两者之积32768的低16位为INT#-32768。
*2 “In1” ~ “InN”之积为正数时变为+∞、负数时变为-∞。

记述示例

“In1”=INT#10，“In2”=INT#20，“In3”=INT#30时的示例如下所示。INT型变量abc的值为INT#6000。

LD



ST

```
abc:=INT#10*INT#20*INT#30;
```

对“In1”～“InN”进行乘法运算。
10*20*30 = 6000，因此abc的值为INT#6000。

“In1” INT#10 * “In2” INT#20 * “In3” INT#30 —乘法→ “Out” =abc INT#6000

梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时存在规格区别。规格区别如下所述。此外，梯形图程序的MUL指令与梯形图程序的*指令的规格完全相同。

规格项目	梯形图	ST
乘数的最多个数	5	2*1
可否省略乘数的输入参数	除InN连接的输入参数外可以省略	所有输入参数均不可省略
变量EN、ENO的存在与否	有	无
乘数的数据类型均为整数时运算处理的位数	8,16,32,64*2	32,64*3

*1 如result := val1 * val2 * val3；所示，可将本指令作为运算符在1个表达式中多次记述。最多可在1个表达式中记述64个运算符。

*2 运算位数以乘数中最大的数据类型为准。例如，对SINT型、INT型、DINT型进行乘法运算时，根据DINT型的大小按照32位进行运算。

*3 乘数中无LINT型及ULINT型时，则按照32位进行运算。例如，对多个SINT型进行乘法运算时，也按照32位进行运算。乘数中有LINT型或ULINT型时，则按照64位进行运算。

参考

运算实数时，请使用 “CheckReal指令(P.2-234)”判定 “Out”是否为+∞、−∞、非数。

使用注意事项

- “Out”的数据类型可与相乘结果不同。请设为包含相乘结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的“数据类型的等级表”及 “转换规则”。
- 即使因乘法而发生溢出或下溢，也不会发生异常。
- 因乘法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。

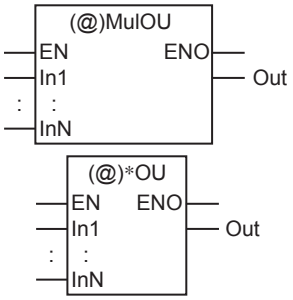
- 与实数的+∞/–∞相对应的相乘结果如下所示。

乘法内容	相乘结果
+∞和正数相乘	+∞
+∞和负数相乘	–∞
–∞和正数相乘	–∞
–∞和负数相乘	+∞
+∞和+∞相乘	+∞
–∞和–∞相乘	+∞
+∞和–∞相乘	–∞
+∞和0相乘	非数
–∞和0相乘	非数

- “In1” ~ “InN”中任意一个的值为非数时，相乘结果的值为非数。

MulOU，*OU

输出对整数、实数进行乘法运算的结果。并且，对整数的相乘结果进行溢出检查。

指令	名称	FB/FUN	图形表现	ST表现
MulOU，*OU	乘法 (带溢出检查)	FUN		Out:=MulOU(In1, ⋯, InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	乘数	输入	待乘数　N为2 ~ 5	遵从数据类型	–	1(*)
Out	输出值	输出	输出值	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN						○	○	○	○	○	○	○	○	○ (*)	○ (*)					
Out						○	○	○	○	○	○	○	○	○	○					

* “In1” ~ “InN”中任意一个的数据类型为实数型时，不进行溢出检查。

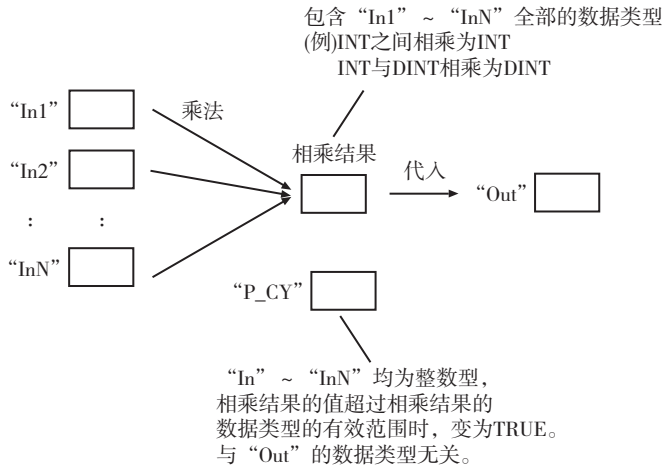
功能

将2个以上5个以下的整数或实数的相乘结果输出至输出值 “Out”。

乘数 “In1” ~ “InN”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相乘结果为DINT型。
关于数据类型的包含关系，请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”的 “数据类型的等级表”及 “转换规则”。

发生溢出时的处理

“In1” ~ “InN”之积超过相乘结果的数据类型的有效范围称为溢出。
“In1” ~ “InN”均为整数型时，若发生溢出，则系统定义变量 “P_CY”(进位标志)的值变为TRUE。
“In1” ~ “InN”中任意一个为实数型时，不进行溢出检查。因此，“P_CY”的值不变。



发生溢出时的 “In1” ~ “InN”的数据类型、相乘结果的数据类型、相乘结果的值、“P_CY”的值之间的关系如下所示。

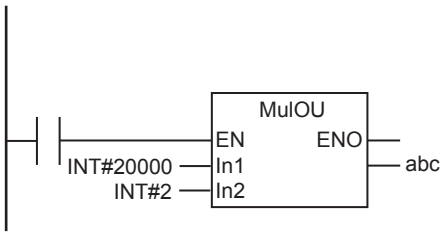
“In1” ~ “InN”的数据类型	相乘结果的数据类型	相乘结果的值	“P_CY”的值
所有整数型	整数型	“In1” ~ “InN”之积中，可通过相乘结果的数据类型的位数表现的值(*1)	TRUE
任意一个为实数型	实数型	± ∞ (*2)	不变化

*1 例如，“In1”的值为INT#16384，“In2”的值为INT#2时，相乘结果的数据类型为INT型。相乘结果的值在两者之积32768的低16位为INT#–32768。
*2 “In1” ~ “InN”之积为正数时变为+ ∞ 、负数时变为– ∞ 。

记述示例

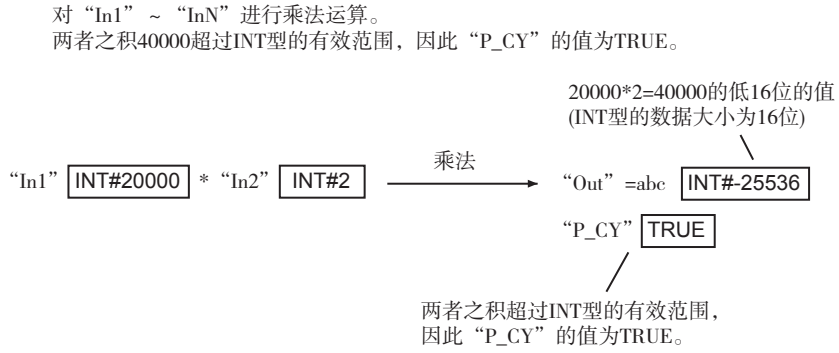
“In1”=INT#20000、“In2”=INT#2，变量abc为INT型时的示例如下所示。
“In1”和 “In2”均为INT型，因此相加结果的数据类型为INT型。两者之积40000超过INT型的有效范围，因此 “P_CY”的值为TRUE。INT型变量abc的值变为INT#–25536(40000的低16位的值)。

LD



ST

```
abc:=MulOU(INT#20000, INT#2);
```



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时无规格区别。此外，梯形图程序的MulOU指令与梯形图程序的*OU指令也无规格区别。

相关的系统定义变量

变量名称	名称	数据类型	内容
P_CY	进位(CY)标志	BOOL	TRUE ：整数运算有溢出 FALSE：整数运算无溢出

参考

- 运算实数时，请使用 “CheckReal指令(P.2–234)”判定 “Out”是否为+∞、–∞、非数。
- 无需溢出检查时，请使用 “MUL， *指令(P.2–180)”。这样可缩短处理时间。

使用注意事项

- “Out”的数据类型可与相乘结果不同。请设为包含相乘结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”的“数据类型的等级表”及 “转换规则”。
- 因乘法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。
- 与实数的+∞/–∞相对应的相乘结果如下所示。

乘法内容	相乘结果
+∞和正数相乘	+∞
+∞和负数相乘	–∞
–∞和正数相乘	–∞
–∞和负数相乘	+∞
+∞和+∞相乘	+∞
–∞和–∞相乘	+∞
+∞和–∞相乘	–∞
+∞和0相乘	非数
–∞和0相乘	非数

- “In1” ~ “InN”中任意一个的值为非数时，相乘结果的值为非数。

DIV，/

对整数、实数进行除法运算。

指令	名称	FB/FUN	图形表现	ST表现
DIV，/	除法	FUN	(@)DIV EN ENO In1 Out In2 (@)/ EN ENO In1 Out In2	Out:=In1/ In2;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被除数	输入	被除数	遵从数据类型	–	(*)
In2	除数		除数			
Out	输出值	输出	输出值	遵从数据类型	–	–

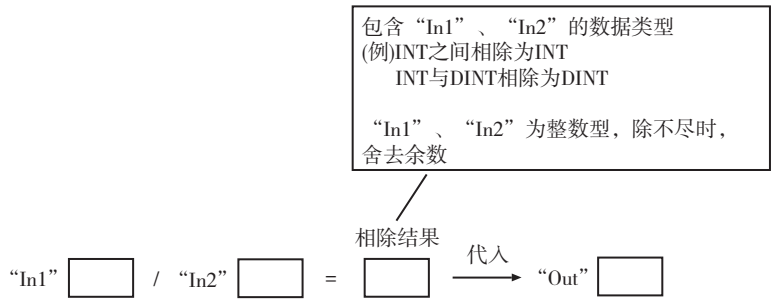
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1						○	○	○	○	○	○	○	○	○	○					
In2						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

功能

将被除数 “In1”除以除数 “In2”后的结果输出至输出值 “Out”。

“In1”和 “In2”可为不同的数据类型。此时，以包含上述所有加数的数据类型进行运算处理。例如，“In1”为INT型、“In2”为DINT型时，以DINT型进行运算处理。因此，相除结果为DINT型。
“In1”、“In2”中任意一个为整数且除不尽时，舍去余数。
关于数据类型的包含关系，请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”的 “数据类型的等级表”及 “转换规则”。



发生溢出时的处理

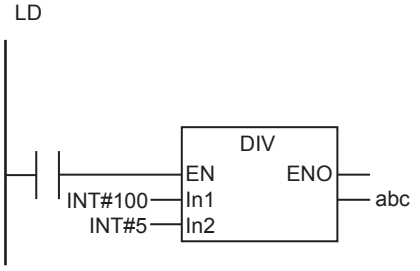
“In1”和“InN”之商超过相除结果的数据类型的有效范围称为溢出。发生溢出时的“In1”、“In2”的数据类型、相除结果的数据类型、相除结果的值之间的关系如下所示。

“In1”、“In2”的数据类型	相除结果的数据类型	相除结果的值
任意一个为实数型	实数型	± ∞(*)

*“In1”、“In2”之商为正数时变为+∞，为负数时变为−∞。

记述示例

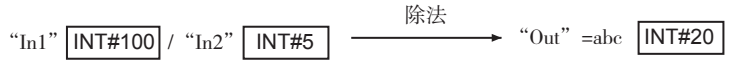
“In1”=INT#100、“In2”=INT#5时的示例如下所示。INT型变量abc的值为INT#20。



ST

```
abc:=INT#100/INT#5;
```

“In1”除以“In2”。
100/5=20，因此abc的值为INT#20。



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用本指令时存在规格区别。规格区别如下所述。此外，梯形图程序的DIV指令与梯形图程序的/指令的规格完全相同。

规格项目	梯形图	ST
变量EN、ENO的存在与否	有	无
被除数和除数的数据类型为整数时运算处理的位数	8,16,32,64*1	32,64*2

*1 运算位数以被除数和除数中最大的数据类型为准。例如，对SINT型和DINT型进行除法运算时，根据DINT型的大小按照32位进行运算。

*2 被除数和除数中无LINT型及ULINT型时，则按照32位进行运算。例如，对多个SINT型进行除法运算时，也按照32位进行运算。被除数和除数中有LINT型或ULINT型时，则按照64位进行运算。

参考

运算实数时，请使用 “CheckReal指令(P.2-234)”判定 “Out”是否为+∞、−∞、非数。

使用注意事项

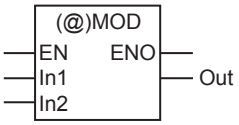
- “Out”的数据类型可与相除结果不同。请设为包含相除结果数据类型的数据类型。否则，编连时会发生异常。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的“数据类型的等级表”及 “转换规则”。
- 即使因除法而发生溢出或下溢，也不会发生异常。
- 因除法而发生溢出或下溢时，运算结果可能与预期不同。为了避免输入参数、输出参数的数据类型的大小发生溢出或下溢，请留有充分余量。
- 与实数的+∞/−∞和0相对应的相除结果如下所示。

		“In1”				
		+∞	正数	0	负数	−∞
“In2”	+∞	非数	0	0	0	非数
	正数	+∞	正数	0	负数	−∞
	0	+∞	+∞	非数	−∞	−∞
	负数	−∞	负数	0	正数	+∞
	−∞	非数	0	0	0	非数

- “In1”、“In2”中任意一个的值为非数时，相除结果的值为非数。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In1”、“In2”为整数型，“In2”的值为0时。

MOD

计算对整数进行除法运算后的余数。

指令	名称	FB/FUN	图形表现	ST表现
MOD	余数	FUN		Out:=In1 MOD In2;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被除数	输入	被除数	遵从数据类型	–	(*)
In2	除数		除数			
Out	余数	输出	余数	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1						○	○	○	○	○	○	○	○							
In2						○	○	○	○	○	○	○	○							
Out						○	○	○	○	○	○	○	○							

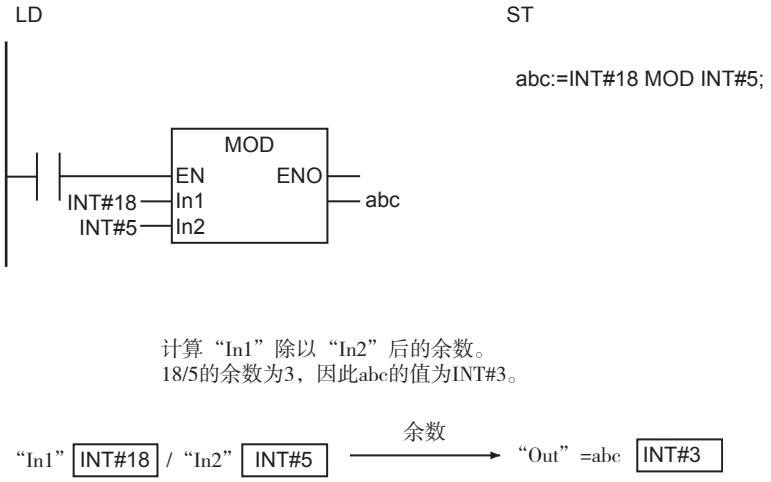
功能

计算被除数 “In1” 除以除数 “In2” 后的余数。
“In1”、“In2” 及余数 “Out” 允许为不同的数据类型，但请设为存在相容性的组合。
关于数据类型的包含关系，请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)” 或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)” 的 “数据类型的等级表” 及 “转换规则”。

本指令的运算通过下式进行。
“Out” = “In1” –(“In1” / “In2”)* “In2”　　括弧内的除法舍去小数点后的数字。
因此，“In1”、“In2” 及 “Out” 的值的示例如下所示。

“In1” 的值	“In2” 的值	“Out” 的值
5	3	2
5	–3	2
–5	3	–2
–5	–3	–2

“In1”=INT#18，“In2”=INT#5时的示例如下所示。变量abc的值为INT#3。



计算“In1”除以“In2”后的余数。
18/5的余数为3，因此abc的值为INT#3。

使用注意事项

- 请将 “Out”的数据类型设定为包含 “In1”、“In2”的有效范围。关于数据类型的包含关系，请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”的 “数据类型的等级表”及 “转换规则”。

ABS

计算整数、实数的绝对值。

指令	名称	FB/FUN	图形表现	ST表现
ABS	绝对值	FUN	(@)ABS EN ENO In Out	Out:=ABS(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	运算对象	输入	运算对象	遵从数据类型	–	(*1)
Out	绝对值	输出	绝对值	遵从数据类型 (*2)	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 不含负数。

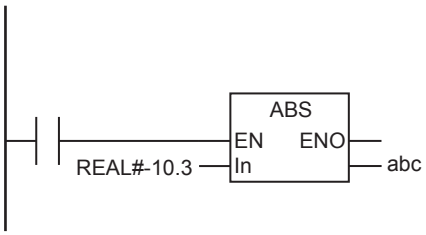
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

功能

输出运算对象 “In”的绝对值。
“In”和绝对值 “Out”可为不同的数据类型。

“In”=REAL#–10.3时的示例如下所示。变量abc的值为REAL#10.3。

LD



ST

```
abc:=ABS(REAL#-10.3);
```

输出“In”的绝对值。
为REAL#–10.3的绝对值，因此abc的值为REAL#10.3。

“In” REAL#-10.3 —绝对值|–10.3|→ “Out”=abc REAL#10.3

参考

运算实数时，请使用 “CheckReal指令(P.2–234)” 判定 “Out”是否为+∞、–∞、非数。

使用注意事项

- 请将 “Out”的数据类型设定为涵盖 “In”的绝对值。
- “In”的值为+∞、–∞、非数时，“Out”的值如下所示。

“In”的值	“Out”的值
+∞	+∞
–∞	+∞
非数	非数

RadToDeg/DegToRad

RadToDeg：将实数的单位从弧度(rad)转换为度(°)。
DegToRad：将实数的单位从度(°)转换为弧度(rad)。

指令	名称	FB/FUN	图形表现	ST表现
RadToDeg	弧度→角度转换	FUN	(@)RadToDeg EN ENO In Out	Out:=RadToDeg(In);
DegToRad	角度→弧度转换	FUN	(@)DegToRad EN ENO In Out	Out:=DegToRad(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	待转换数据	遵从数据类型	• RadToDeg：rad • DegToRad：°	(*)
Out	转换结果	输出	转换结果	遵从数据类型	• RadToDeg：° • DegToRad：rad	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

功能

- **RadToDeg**

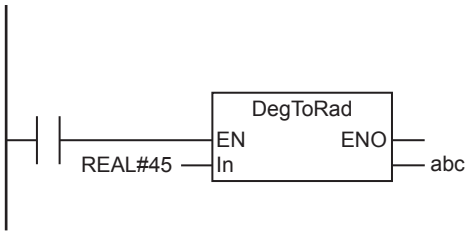
  将转换对象 “In” 的单位从弧度(rad)转换为度(°)。
  转换公式如下所示。
  “Out” = “In” *180/π

- **DegToRad**

  将转换对象 “In” 的单位从度(°)转换为弧度(rad)。
  转换公式如下所示。
  “Out” = “In” *π/180

DegToRad指令下，“In”=REAL#45时的示例如下所示。REAL型变量abc的值为REAL#0.785398。

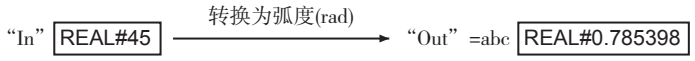
LD



ST

```
abc:=DegToRad(REAL#45);
```

将“In”的值从度(°)转换为弧度(rad)。
45° 转换为弧度(rad)为0.785398，因此abc的值为REAL#0.785398。



参考

请使用 “CheckReal指令(P.2–234)” 判定 “Out”是否为+∞、–∞、非数。

使用注意事项

- 转换结果的绝对值大于 “Out”的数据类型可表现的最大值时，“Out”的值为+∞/–∞。
- 转换结果的绝对值小于 “Out”的数据类型可表现的最小值时，“Out”的值为0。
- 请将 “Out”的数据类型大小设为 “In”的数据类型以上。
- “In”的值为+∞、–∞、非数时，“Out”的值如下所示。

“In”的值	“Out”的值
+∞	+∞
–∞	–∞
非数	非数

- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

SIN/COS/TAN

进行实数的三角函数运算。
SIN ： sin(正弦值)
COS ： cos(余弦值)
TAN ： tan(正切值)

指令	名称	FB/FUN	图形表现	ST表现
SIN	SIN运算	FUN	(@)SIN EN ENO In Out	Out:=SIN(In);
COS	COS运算	FUN	(@)COS EN ENO In Out	Out:=COS(In);
TAN	TAN运算	FUN	(@)TAN EN ENO In Out	Out:=TAN(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	运算对象	输入	运算对象	遵从数据类型	rad	(*1)
Out	运算结果	输出	运算结果	• SIN：(*2) • COS：(*2) • TAN：遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 REAL时为–1.000000e+0 ~ 1.000000e+0。LREAL时为–1.00000000000000e+0 ~ 1.00000000000000e+0。

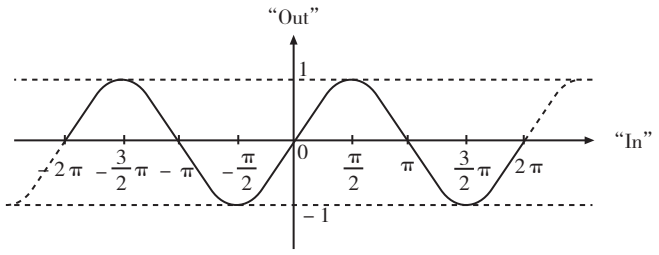
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

功能

进行实数的三角函数运算。
运算对象 “In”为弧度(rad)单位的角度。

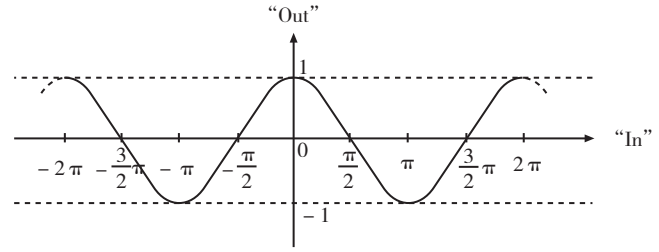
● SIN

计算 “In”的sin(正弦值)。



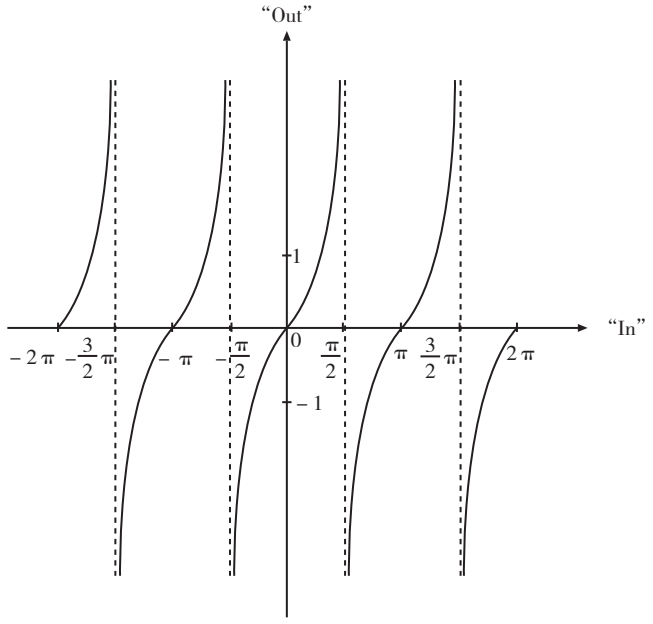
● COS

计算 “In”的cos(余弦值)。



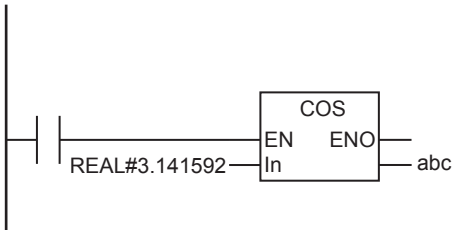
● TAN

计算 “In”的tan(正切值)。



COS指令下，“In”=REAL#3.141592时的示例如下所示。变量abc的值为REAL#−1.0。

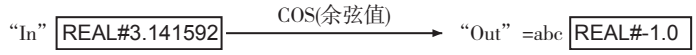
LD



ST

```
abc:=COS(REAL#3.141592);
```

计算“In”的COS(余弦值)。
3.141592的COS(余弦值)为–1.0，因此abc的值为REAL#–1.0。



参考

- 请使用 “RadToDeg/DegToRad指令(P.2–194)”进行度(°)和弧度(rad)之间的转换。
- TAN指令下，“In”为nπ/2(n为整数)时，“Out”的值为+∞/–∞。请使用 “CheckReal指令(P.2–234)”判定 “Out”的值是否为+∞/–∞。

使用注意事项

- “In”的值为+∞、–∞、非数时，“Out”的值为非数。
- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

ASIN/ACOS/ATAN

进行实数的反三角函数运算。

ASIN ：$\sin^{-1}$(反正弦值)

ACOS ：$\cos^{-1}$(反余弦值)

ATAN ：$\tan^{-1}$(反正切值)

指令	名称	FB/FUN	图形表现	ST表现
ASIN	SIN^{-1}运算	FUN	(@)ASIN EN ENO In Out	Out:=ASIN(In);
ACOS	COS^{-1}运算	FUN	(@)ACOS EN ENO In Out	Out:=ACOS(In);
ATAN	TAN^{-1}运算	FUN	(@)ATAN EN ENO In Out	Out:=ATAN(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	运算对象	输入	运算对象	遵从数据类型	–	(*)
Out	运算结果	输出	运算结果	• ASIN：$-\pi/2 \sim \pi/2$ • ACOS：$0 \sim \pi$ • ATAN：$-\pi/2 \sim \pi/2$	rad	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

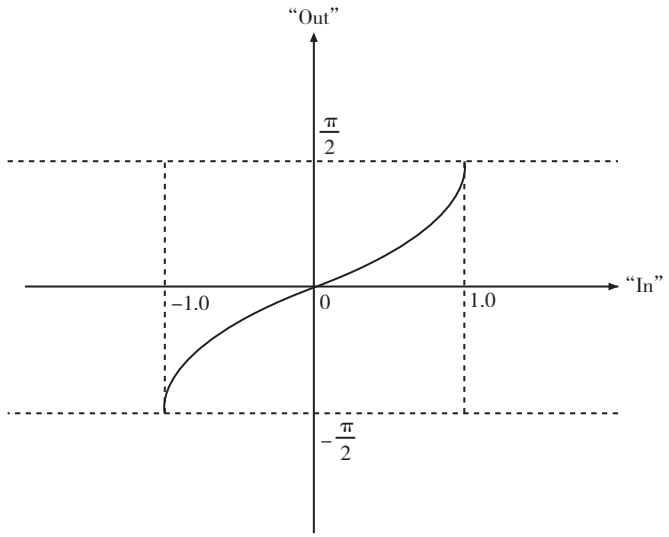
功能

进行实数的反三角函数运算。

运算结果 “Out”为弧度单位的角度。

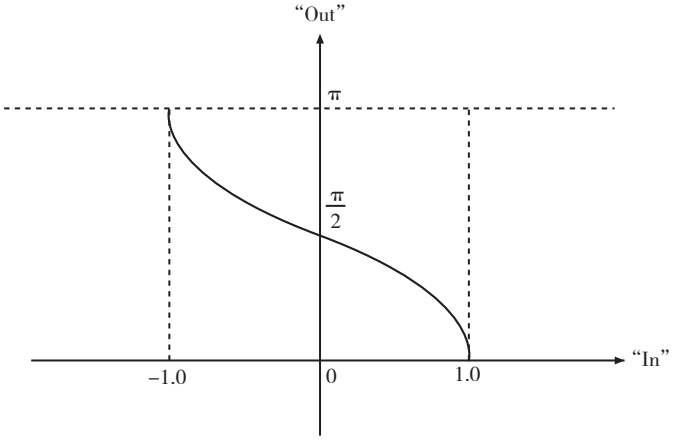
● ASIN

输出 “In” 的$\sin^{-1}$(反正弦值)。“Out” 的范围为$-\pi/2 \sim \pi/2$。



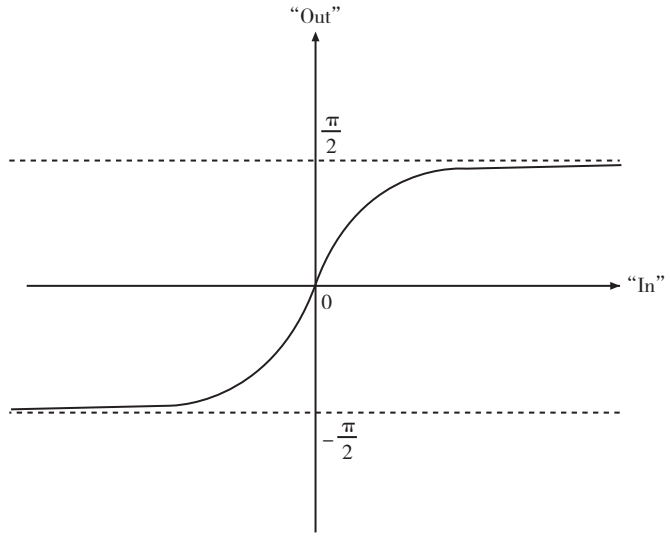
● ACOS

输出 “In” 的$\cos^{-1}$(反余弦值)。“Out” 的范围为$0 \sim \pi$。



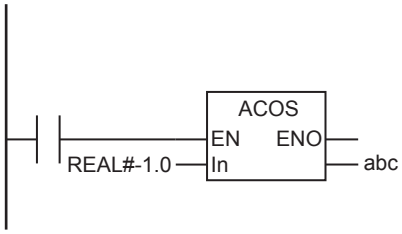
● ATAN

输出 “In”的$\tan^{-1}$(反正切值)。“Out”的范围为$-\pi/2 \sim \pi/2$。
“In”的值为$+\infty$时，“Out”的值为$\pi/2$。“In”的值为$-\infty$时，“Out”的值为$-\pi/2$。



ACOS指令下，“In”=REAL#−1.0时的示例如下所示。变量abc的值为REAL#3.141592。

LD



ST

```
abc:=ACOS(REAL#-1.0);
```

计算“In”的COS^{-1}(反余弦值)。
−1.0的COS^{-1}(反余弦值)为3.141592，因此abc的值为REAL#3.141592。

“In” REAL#-1.0 —— COS^{-1}(反余弦值) ——→ “Out” =abc REAL#3.141592

参考

请使用 “RadToDeg/DegToRad指令(P.2−194)”进行度(°)和弧度(rad)之间的转换。

使用注意事项

- ASIN、ACOS指令下，“In”的值为–1.0 ~ 1.0以外时，“Out”的值为非数。“In”的值为+∞、–∞，非数时也一样。
- ATAN指令下，“In”的值为非数时，“Out”的值为非数。
- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

SQRT

计算实数的平方根(√‾)。

指令	名称	FB/FUN	图形表现	ST表现
SQRT	平方根运算	FUN	(@)SQRT EN ENO In Out	Out:=SQRT(In);

变量

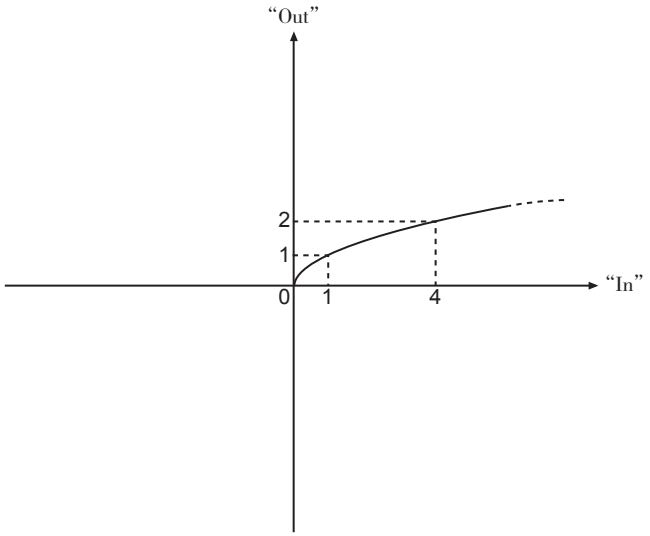
	名称	输入/输出	内容	有效范围	单位	初始值
In	运算对象	输入	运算对象	遵从数据类型(*1)	–	(*2)
Out	平方根	输出	平方根	(*3)	–	–

*1 不含负数。
*2 省略输入参数时，初始值不适用。编连时会发生异常。
*3 REAL时为0.000000e+00 ~ 1.844674e+19或+∞。LREAL时为0.00000000000000e+000 ~ 1.34078079299425e+154或+∞。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

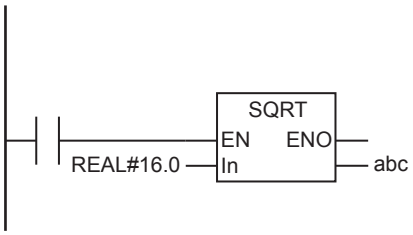
功能

计算实数的运算对象 “In” 的平方根(√‾)。
“In” 和平方根 “Out” 可为不同的数据类型。



“In”=REAL#16.0时的示例如下所示。变量abc的值为REAL#4.0。

LD



ST

```
abc:=SQRT(REAL#16.0);
```

计算“In”的平方根(√)。
16.0的平方根(√)为4.0，因此abc的值为REAL#4.0。

“In” REAL#16.0 —平方根(√)→ “Out”=abc REAL#4.0

参考

请使用 “CheckReal指令(P.2–234)” 判定 “Out”的值是否为+∞。

使用注意事项

- “In”的值非正数值时，“Out”的值如下所示。

“In”的值	“Out”的值
负数值	非数
0	0
+∞	+∞
–∞	非数
非数	非数

- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

LN/LOG

计算实数的对数。
LN ：自然对数
LOG：常用对数

指令	名称	FB/FUN	图形表现	ST表现
LN	自然对数运算	FUN	(@)LN EN ENO In Out	Out:=LN(In);
LOG	常用对数运算	FUN	(@)LOG EN ENO In Out	Out:=LOG(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	运算对象	输入	运算对象	遵从数据类型 (*1)	–	(*2)
Out	对数	输出	对数	(*3)	–	–

*1 不含负数。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

- LN
  “In”、“Out”为REAL时 –8.73365448e+1 ~ 8.87228390e+1 或–∞/+∞
  “In”为REAL，“Out”为LREAL时 –8.7336544750000000e+1 ~ 8.8722839050000000e+1 或–∞/+∞
  “In”为LREAL，“Out”为REAL时 –7.08384950e+2 ~ 7.09782712e+2 或–∞/+∞
  “In”、“Out”为LREAL时 –7.0838495021978327e+1 ~ 7.0978271289338399e+2 或–∞/+∞
- LOG
  “In”、“Out”为REAL时 –3.79297795e+1 ~ 3.85318394e+1 或–∞/+∞
  “In”为REAL，“Out”为LREAL时 –3.7929779453965430e+1 ~ 3.8531839419564961e+1 或–∞/+∞
  “In”为LREAL、“Out”为REAL时 –3.07652656e+2 ~ 3.08254716e+2 或–∞/+∞
  “In”、“Out”为LREAL时 –3.0765265556858878e+2 ~ 3.0825471555991674e+2 或–∞/+∞

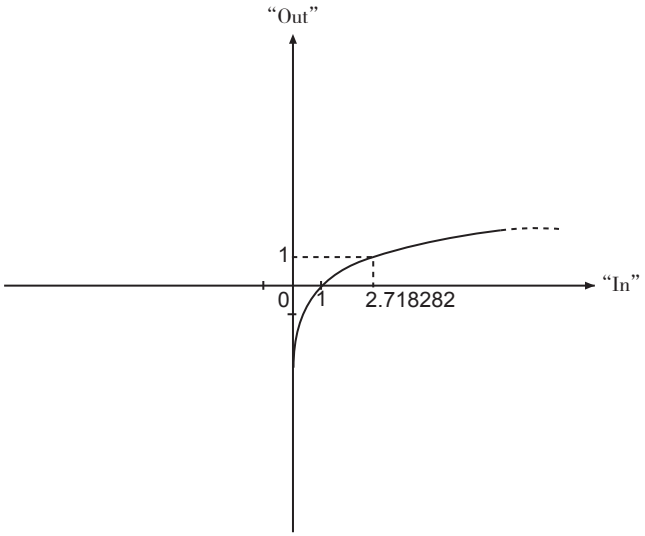
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

功能

计算实数的对数。

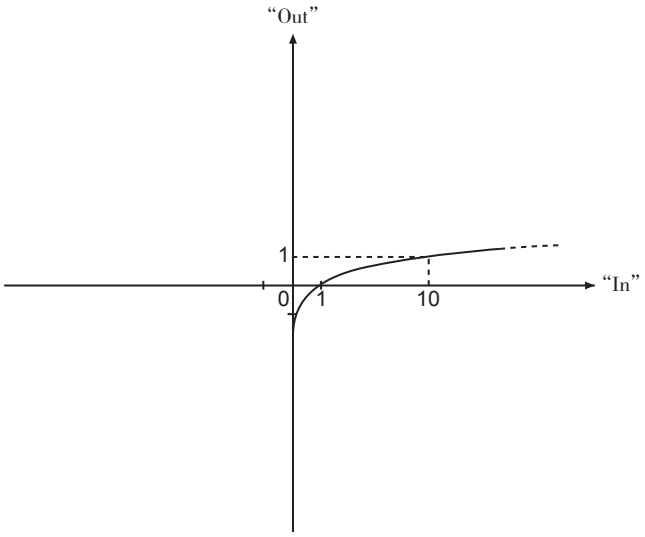
- LN

计算自然对数(底为e 2.718282的对数)。



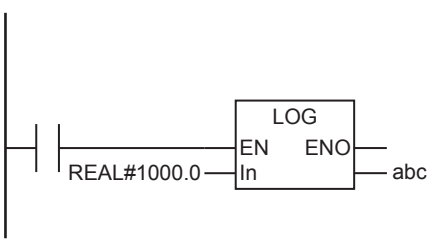
- LOG

计算常用对数(底为10的对数)。



LOG指令下，“In”=REAL#1000.0时的示例如下所示。变量abc的值为REAL#3.0。

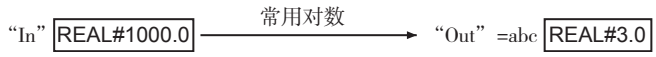
LD



ST

```
abc:=LOG(REAL#1000.0);
```

计算“In”的常用对数。
1000.0的常用对数为3.0，因此abc的值为REAL#3.0。



参考

请使用 “CheckReal指令(P.2-234)”判定 “Out”的值是否为+∞、−∞、非数。

使用注意事项

- “In”的值非正数值时，“Out”的值如下所示。

“In”的值	“Out”的值
负数值	非数
0	−∞
+∞	+∞
−∞	非数
非数	非数

- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

EXP

计算自然对数的底e的指数函数。

指令	名称	FB/FUN	图形表现	ST表现
EXP	自然指数运算	FUN	(@)EXP EN ENO In Out	Out:=EXP(In);

变量

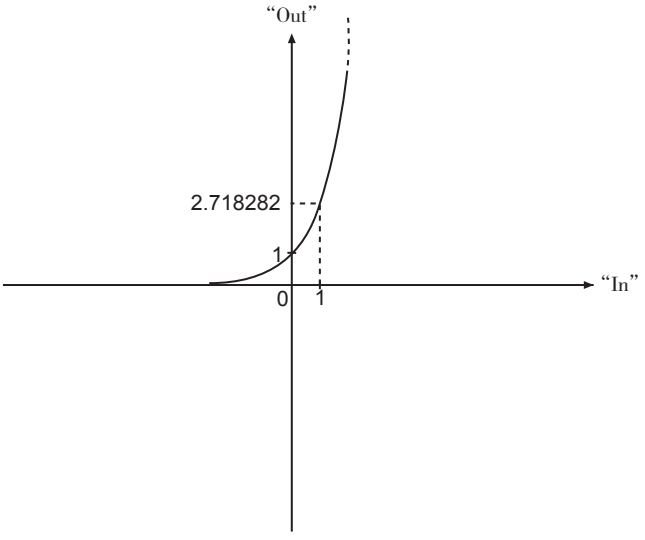
	名称	输入/输出	内容	有效范围	单位	初始值
In	指数	输入	指数	遵从数据类型	–	(*1)
Out	运算结果	输出	运算结果	遵从数据类型 (*2)	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 不含负数。

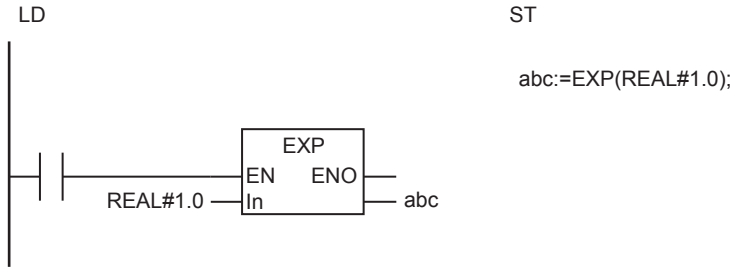
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

功能

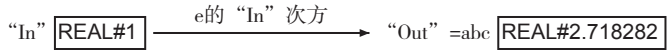
计算自然对数的底e的 “In”。



“In”=REAL#1.0时的示例如下所示。变量abc的值为REAL#2.718282。



计算自然对数的底e的“In”。
e1为2.718282，因此abc的值为REAL#2.718282。



参考

- 计算以e以外为底的指数时，请使用 “EXPT，**指令(P.2–210)”。
- 请使用 “CheckReal指令(P.2–234)”判定 “Out”的值是否为+∞、–∞、非数。

使用注意事项

- “In”的值为0.0、+∞、–∞、非数时，“Out”的值如下所示。

“In”的值	“Out”的值	
	右述以外	NX1P2
0.0	1.0	1.0
+∞	非数	+∞
–∞	非数	0.0
非数	非数	非数

- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

EXPT，**

计算2个实数的乘幂。

指令	名称	FB/FUN	图形表现	ST表现
EXPT，**	乘方运算	FUN	(@)EXPT EN ENO In Out Pwr (@)** EN ENO In Out Pwr	Out:=EXPT(In, Pwr); Out:=In ** Pwr;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	底数	输入	底数　5^2时的5	遵从数据类型	–	(*)
Pwr	指数		指数　5^2时的2			
Out	运算结果	输出	运算结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

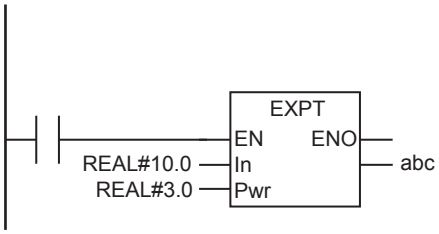
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Pwr														○	○					
Out														○	○					

功能

对底数 “In”和指数 “Pwr”，计算In^{Pwr}的值。

“In”=REAL#10.0，“Pwr”=REAL#3.0时的示例如下所示。变量abc的值为REAL#1000.0。

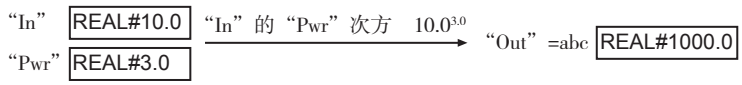
LD



ST

```
abc:=EXPT(REAL#10.0, REAL#3.0);
```

计算“In”的“Pwr”次方。
$10.0^{3.0}$为1000.0，因此abc的值为REAL#1000.0。



梯形图与ST的规格区别

通过梯形图程序使用本指令时与通过ST程序使用**运算符时存在规格区别。规格区别如下所述。此外，梯形图程序的EXPT指令与梯形图程序的**指令、ST程序的EXPT函数的规格完全相同。

规格项目	梯形图及ST程序的EXPT函数	ST程序的**运算符
变量EN、ENO的存在与否	有	无

参考

- 计算以e为底的指数时，请使用 “EXP指令(P.2-208)”。
- 请使用 “CheckReal指令(P.2-234)”判定 “Out”的值是否为+∞、−∞、非数。

使用注意事项

- 运算结果的绝对值小于实数可表现的最小值时，“Out”的值为0。

  (例) $(1.175494e-38)^2 \rightarrow 0$
- 梯形图的EXPT指令、**指令及ST的EXPT函数时，如将整数型的参数传输至"In"，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

“In”与“Pwr”的值的组合

根据“In”和“Pwr”的值的组合，“Out”的值如下所示。

● NX1P2 CPU单元以外使用“EXPT”FUN时

		In									
		+∞	1 ~ +∞	1	0 ~ 1	0	−1 ~ 0	−1	−1 ~ −∞	−∞	非数
Pwr	+∞	+∞	+∞	1	0	1	0	1	+∞	1	非数
	正偶数	+∞	数值*1*2	1	数值*1*2	0	数值*1*2	1	数值*1*2	+∞	非数
	正奇数						数值*2*3	−1	数值*2*3		
	正小数						非数				
	0	1	1			1	1			1	1
	负偶数	0	数值*1*2	1	数值*1*2	+∞	数值*1*2	1	数值*1*2	0	非数
	负奇数						数值*2*3	−1	数值*2*3		
	负小数						非数				
	−∞	0	0	1	+∞	+∞	+∞	1	0	0	非数
	非数	1	非数	1	非数	1	非数			1	非数 1

*1 运算结果超过“Out”的数据类型的有效范围时，“Out”的值为+∞。
*2 运算结果接近0导致“Out”的数据类型无法表现或运算结果为非规格化数时，“Out”的值为0。
*3 运算结果超过“Out”的数据类型的有效范围时，“Out”的值为−∞。

● NX1P2 CPU单元使用“EXPT”FUN时

		In									
		+∞	1 ~ +∞	1	0 ~ 1	0	−1 ~ 0	−1	−1 ~ −∞	−∞	非数
Pwr	+∞	+∞	+∞	1	0	1	0	1	+∞	1	非数
	正偶数	+∞	数值*1*2	1	数值*1*2	0	数值*1*2	1	数值*1*2	+∞	非数
	正奇数						数值*2*3	−1	数值*2*3	−∞	
	正小数						非数			+∞	
	0	1	1			1	1			1	1
	负偶数	0	数值*1*2	1	数值*1*2	+∞	数值*1*2	1	数值*1*2	0	非数
	负奇数						数值*2*3	−1	数值*2*3	−0	
	负小数						非数			0	
	−∞	0	0	1	+∞	+∞	+∞	1	0	0	非数
	非数	1	非数	1	非数	1	非数			1	非数 1

*1 运算结果超过“Out”的数据类型的有效范围时，“Out”的值为+∞。
*2 运算结果接近0导致“Out”的数据类型无法表现或运算结果为非规格化数时，“Out”的值为0。
*3 运算结果超过“Out”的数据类型的有效范围时，“Out”的值为−∞。

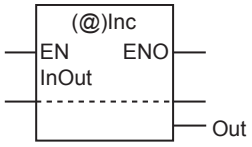
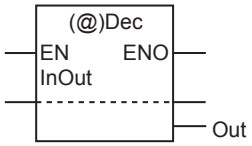
● 使用 “**”运算符时

		In									
		+∞	1 ~ +∞	1	0 ~ 1	0	−1 ~ 0	−1	−1 ~ −∞	−∞	非数
Pwr	+∞	+∞	+∞	1	0	1	0	1	+∞	1	非数
	正偶数	+∞	数值[*1*2]	1	数值[*1*2]	0	数值[*1*2]	1	数值[*1*2]	+∞	非数
	正奇数						数值[*2*3]	−1	数值[*2*3]	−∞	
	正小数						非数			+∞	
	0	1	1			1	1			1	1
	负偶数	0	数值[*1*2]	1	数值[*1*2]	+∞	数值[*1*2]	1	数值[*1*2]	0	非数
	负奇数						数值[*2*3]	−1	数值[*2*3]	−0	
	负小数						非数			0	
	−∞	0	0	1	+∞	+∞	+∞	1	0	0	非数
	非数	1	非数			1	非数			1	非数 1

*1 运算结果超过 “Out”的数据类型的有效范围时，“Out”的值为+∞。
*2 运算结果接近0导致 “Out”的数据类型无法表现或运算结果为非规格化数时，“Out”的值为0。
*3 运算结果超过 “Out”的数据类型的有效范围时，“Out”的值为−∞。

Inc/Dec

Inc ：对整数值进行增量。
Dec：对整数值进行减量。

指令	名称	FB/FUN	图形表现	ST表现
Inc	增量	FUN		Inc(InOut);
Dec	减量	FUN		Dec(InOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut	对象数据	输入输出	对象数据	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut						○	○	○	○	○	○	○	○							
Out	○																			

功能

- Inc

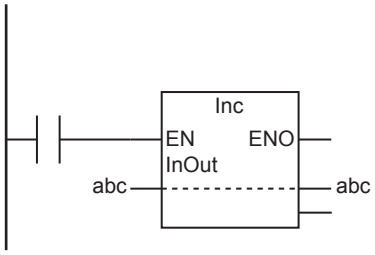
  对对象数据 “InOut”进行增量。最终，超过 “InOut”的最大值时，恢复到最小值。

- Dec

  对对象数据 “InOut”进行减量。最终，超过 “InOut”的最小值时，恢复到最大值。

Inc指令下，将变量abc传输至 “InOut”时的示例如下所示。abc的值为INT#4时，执行指令后abc的值为INT#5。

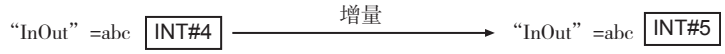
LD



ST

```
Inc(abc);
```

对“InOut”进行增量。
abc的值为INT#4时，执行指令后abc的值为INT#5。

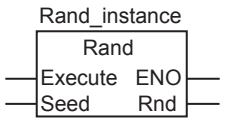


使用注意事项

在ST程序中使用本指令时，不使用返回值 “Out”。

Rand

生成伪随机数。

指令	名称	FB/FUN	图形表现	ST表现
Rand	生成随机数	FB		Rand_instance(Execute, Seed, Rnd);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Seed	随机数模式	输入	随机数序列的指定 0：无指定	遵从数据类型	–	(*1)
Rnd	随机数	输出	随机数	(*2)	–	–

*1 省略输入参数时，值为0。不是初始值属性中指定的值。

*2 0.00000000000000e+0 ~ 1.00000000000000e+0

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Seed							○													
Rnd															○					

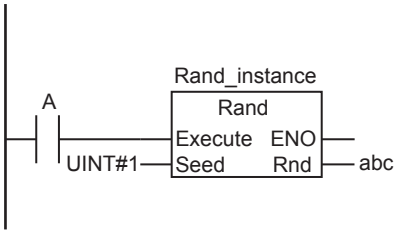
功能

生成随机数 “Rnd”。每次执行指令时 “Rnd”的值均不相同。
随机数模式 “Seed”指定随机数序列。如果 “Seed”的值相同，则每次接通电源时都会生成相同的随机数序列。因此，可生成具有再现性的随机数。

“Seed”的值为 0 时，生成无再现性的随机数。若要每次接通电源时不生成相同的随机数序列，请将“Seed”的值设为0。

“Seed”=UINT#1时的示例如下所示。“Seed”的值不为0，因此生成具有再现性的随机数。

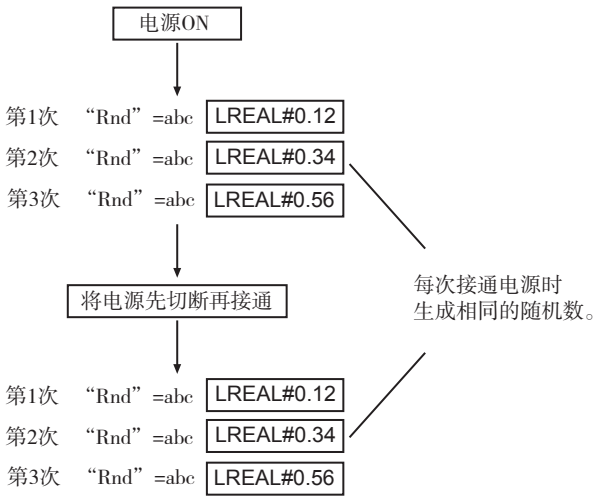
LD



ST

```
Rand_instance(A, UINT#1, abc);
```

生成具有再现性的随机数。



(*)上述随机数的值为示例。与实际值有所差异。

参考

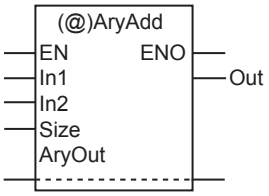
“Rnd”的值为0～1范围内的实数。因此，可通过以下处理生成特定范围内的随机数。

(例)　生成100～200范围内的随机数时

```
Rand_instance(A, UINT#1, abc);
随机数:=LREAL_TO_INT((200.0-100.0)*abc)+100;
```

AryAdd

对2个数组的各元素进行加法运算。

指令	名称	FB/FUN	图形表现	ST表现
AryAdd	数组整体加法	FUN		AryAdd(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ] 数组 In2[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	–	(*)
Size	运算对象的元素数		运算对象的元素数			1
AryOut[ ] 数组	运算结果数组	输入输出	运算结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

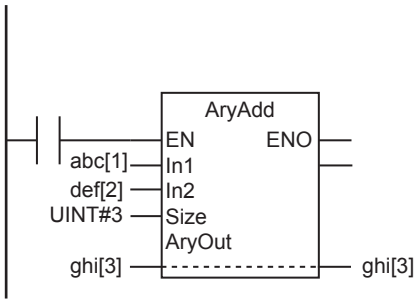
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组						○	○	○	○	○	○	○	○	○	○					
In2[ ] 数组	与In1[ ]相同数据类型的数组																			
Size							○													
AryOut[ ] 数组	与In1[ ]相同数据类型的数组																			
Out	○																			

功能

将运算对象数组In1[ ]、In2[ ]的In1[0]、In2[0]之后的“Size”个元素各自的加数代入运算结果数组AryOut[ ]。

“Size”=UINT#3时的示例如下所示。

LD



ST

```
AryAdd(abc[1], def[2], UINT#3, ghi[3]);
```

“Size”=UINT#3

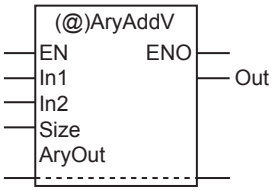
In1[0]=abc[1]	1234	+	In2[0]=def[2]	2345	→	AryOut[0]=ghi[3]	3579
In1[1]=abc[2]	2345	+	In2[1]=def[3]	3456	→	AryOut[1]=ghi[4]	5801
In1[2]=abc[3]	3456	+	In2[2]=def[4]	4567	→	AryOut[2]=ghi[5]	8023

使用注意事项

- 请将In1[ ]、In2[ ]、AryOut[ ]的数据类型设为相同。
- 运算结果在AryOut[ ]的有效范围外时，结果为错误值。此时，不会发生异常。也不会破坏其元素相邻的存储区域。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Size”的值超过In1[ ]、In2[ ]、AryOut[ ]中任一数组区域时。

AryAddV

数组的各元素加上相同数值。

指令	名称	FB/FUN	图形表现	ST表现
AryAddV	数组元素加法	FUN		AryAddV(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ] 数组	加法数组	输入	加法数组	遵从数据类型	–	(*)
In2	加数		加数			
Size	元素数		待加In1[ ]的元素数			1
AryOut[ ] 数组	相加结果数组	输入输出	相加结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

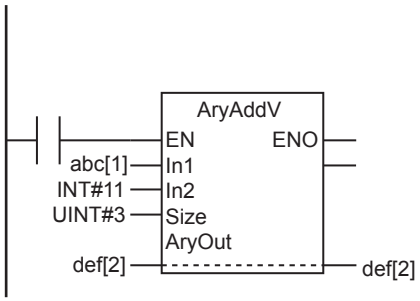
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组						○	○	○	○	○	○	○	○	○	○					
In2	与In1[ ]的元素相同的数据类型																			
Size							○													
AryOut[ ] 数组	与In1[ ]相同的数据类型																			
Out	○																			

功能

加法数组In1[ ]的从In1[0]起的 “Size”个各元素加上加数 “In2”的值，并输出至相加结果数组AryOut[ ]。

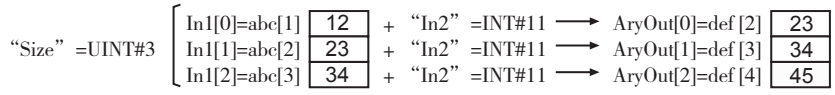
“In2”=INT#11、“Size”=UINT#3时的示例如下所示。

LD



ST

```
AryAddV(abc[1], INT#11, UINT#3, def[2]);
```

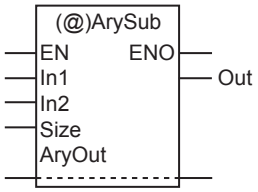


使用注意事项

- 请将In1[ ]、“In2”、AryOut[ ]的数据类型设为相同。
- 相加结果在AryOut[ ]的有效范围外时，AryOut[ ]的元素为错误值。此时，不会发生异常。也不会破坏其元素相邻的存储区域。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、“In2”、AryOut[ ]的数据类型不同时。
  - “Size”的值超过In1[ ]或AryOut[ ]的数组区域时。

ArySub

对2个数组的各元素进行减法运算。

指令	名称	FB/FUN	图形表现	ST表现
ArySub	数组整体减法	FUN		ArySub(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ] 数组	被减数组	输入	被减数组	遵从数据类型	–	(*)
In2[ ] 数组	减法数组		减法数组			
Size	元素数		待减元素数			1
AryOut[ ] 数组	相减结果数组	输入输出	相减结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

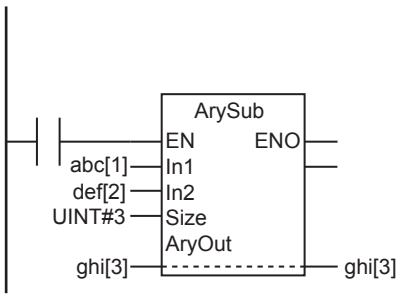
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组						○	○	○	○	○	○	○	○	○	○					
In2[ ] 数组	与In1[ ]相同的数据类型																			
Size							○													
AryOut[ ] 数组	与In1[ ]相同的数据类型																			
Out	○																			

功能

对被减数组In1[ ]和相减数组In2[ ]进行从开头的 “Size”个各元素的减法。相减结果输出至相减结果数组AryOut[ ]。

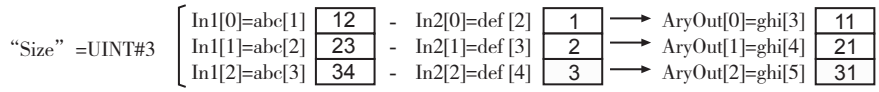
“Size”=UINT#3时的示例如下所示。

LD



ST

```
ArySub(abc[1], def[2], UINT#3, ghi[3]);
```

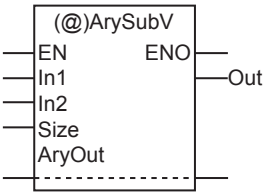


使用注意事项

- 请将In1[ ]、In2[ ]、AryOut[ ]的数据类型设为相同。
- 相减结果在AryOut[ ]的有效范围外时，AryOut[ ]的元素为错误值。此时，不会发生异常。也不会破坏其元素相邻的存储区域。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、In2[ ]、AryOut[ ]的数据类型不同时。
  - “Size”的值超过In1[ ]、In2[ ]、AryOut[ ]中任一数组区域时。

ArySubV

数组的各元素减去相同数值。

指令	名称	FB/FUN	图形表现	ST表现
ArySubV	数组元素减法	FUN		ArySubV(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ] 数组	被减数组	输入	被减数组	遵从数据类型	–	(*)
In2	减数		减数			
Size	元素数		待减In1[ ]的元素数			1
AryOut[ ] 数组	相减结果数组	输入输出	相减结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

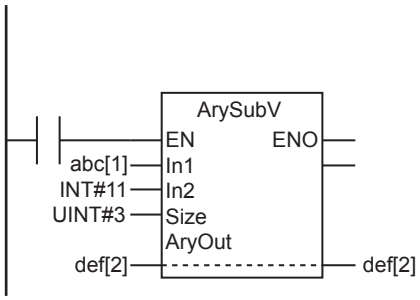
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组						○	○	○	○	○	○	○	○	○	○					
In2	与In1[ ]的元素相同的数据类型																			
Size							○													
AryOut[ ] 数组	与In1[ ]相同的数据类型																			
Out	○																			

功能

被减数组In1[ ]的从In1[0]起的 “Size”个各元素减去减数 “In2”的值，并输出至相减结果数组AryOut[ ]。

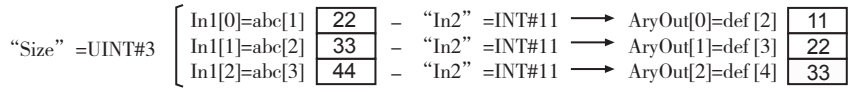
“In2”=INT#11、“Size”=UINT#3时的示例如下所示。

LD



ST

```
ArySubV(abc[1], INT#11, UINT#3, def[2]);
```



使用注意事项

- 请将In1[ ]、“In2”、AryOut[ ]的数据类型设为相同。
- 相减结果在AryOut[ ]的有效范围外时，AryOut[ ]的元素为错误值。此时，不会发生异常。也不会破坏其元素相邻的存储区域。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In1[ ]、“In2”、AryOut[ ]的数据类型不同时。
  - “Size”的值超过In1[ ]或AryOut[ ]的数组区域时。

AryMean

计算数组元素的平均值。

指令	名称	FB/FUN	图形表现	ST表现
AryMean	数组元素的平均值计算	FUN	(@)AryMean EN ENO In Out Size	Out:= AryMean(In, Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	–	(*)
Size	运算对象的元素数		In[ ]的元素数			1
Out	运算结果	输出	运算结果	遵从数据类型	–	–

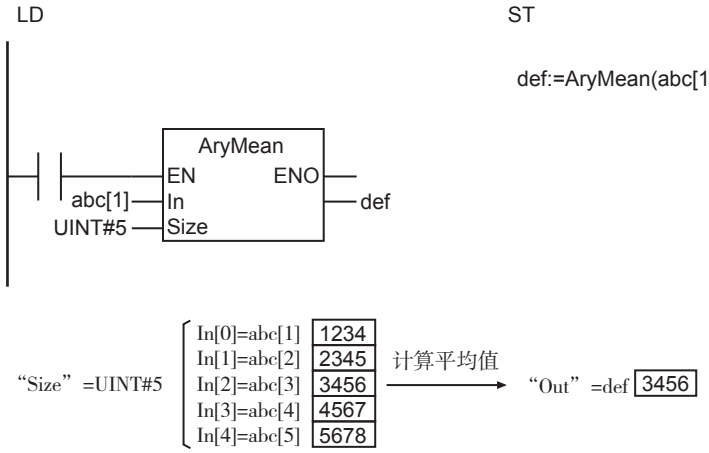
* 省略输入参数时，初始值不适用。编译时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组						○	○	○	○	○	○	○	○	○	○	○				
Size							○													
Out						○	○	○	○	○	○	○	○	○	○	○				

功能

计算运算对象数组In[ ]的In[0]之后的 “Size”个元素的平均值。

“Size”=UINT#5时的示例如下所示。

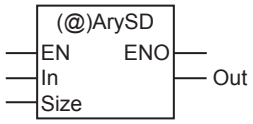


使用注意事项

- 关于In[ ]的值为+∞/–∞和非数时的四则运算的结果，请参阅 “ADD，+指令(P.2–166)”、“SUB，–指令(P.2–173)”、“MUL，*指令(P.2–180)”、“DIV，/指令(P.2–187)”的功能说明。
- In[ ]、“Out”为整数型时，舍去平均值的小数点后的数字。
- In[ ]和 “Out”的数据类型不同时，请将 “Out”的有效范围设为包含In[ ]的有效范围。
- 运算结果超过 “Out”的有效范围时，“Out”的值为错误值。此时，不会发生异常。
- 运算过程的中间值超过In[ ]的有效范围时，“Out”的值为错误值。此时，不会发生异常。
- “Size”的值为0时，“Out”的值为0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过In[ ]的数组区域时。

ArySD

计算数组元素的标准偏差。

指令	名称	FB/FUN	图形表现	ST表现
ArySD	数组元素的标准偏差	FUN		Out:=ArySD(In, Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	–	(*)
Size	元素数		待转换的In[ ]的元素数			2
Out	标准偏差	输出	标准偏差	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组														○	○					
Size							○													
Out														○	○					

功能

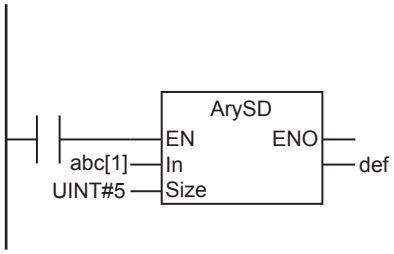
计算运算对象数组In[ ]的In[0]起的 “Size”个元素的标准偏差。

$$\text{标准偏差} = \sqrt{\frac{\sum_{i} (In[i]-InM)^2}{\text{“Size”}-1}}$$

i ：In[ ]的下标 0～“Size”−1
InM：In[0]～In[“Size”−1]的平均值

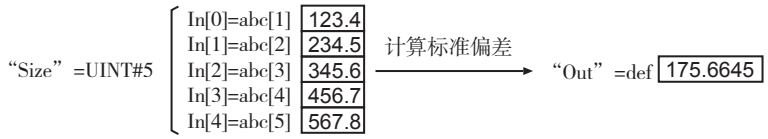
“Size”=UINT#5时的示例如下所示。

LD



ST

```
def:=ArySD(abc[1], UINT#5);
```

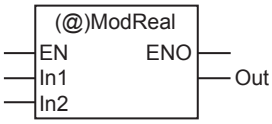


使用注意事项

- “Size”的值为0或1时，“Out”的值为0。
- 运算过程的中间值超过In[ ]的有效范围时，“Out”的值为错误值。此时，不会发生异常。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过In[ ]的数组区域时。

ModReal

计算对实数进行除法运算后的余数。

指令	名称	FB/FUN	图形表现	ST表现
ModReal	实数余数	FUN		Out:=ModReal(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被除数	输入	被除数	遵从数据类型	–	(*)
In2	除数		除数			
Out	余数	输出	余数	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1														○	○					
In2														○	○					
Out														○	○					

功能

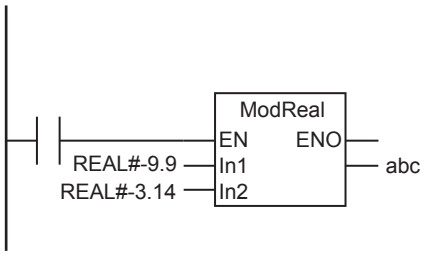
计算被除数 “In1” 除以除数 “In2” 后的余数。

本指令的运算通过下式进行。
“Out” = “In1” –(“In1” / “In2”)* “In2” 括弧内的除法舍去小数点后的数字。
因此，“In1”、“In2” 及 “Out” 的值的示例如下所示。

“In1” 的值	“In2” 的值	“Out” 的值
9.9	3.14	0.48
9.9	–3.14	0.48
–9.9	3.14	–0.48
–9.9	–3.14	–0.48

“In1”=REAL#–9.9、“In2”=REAL#–3.14时的示例如下所示。变量abc的值为REAL#–0.48。

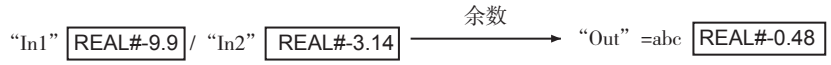
LD



ST

```
abc:=ModReal(REAL#-9.9, REAL#-3.14);
```

计算“In1”除以“In2”后的余数。
–9.9/(–3.14)的余数为–0.48，因此abc的值为REAL#–0.48。



参考

请使用 “CheckReal指令(P.2–234)” 检测 “Out”的值是否为+∞、–∞、非数。

使用注意事项

- 根据 “In1”和 “In2”的值的组合，“Out”的值如下所示。

		“In1”				
		0	数值	+∞	–∞	非数
“In2”	0	非数	非数	非数	非数	非数
	数值	0	“In1”/“In2”的余数	非数	非数	非数
	+∞	0	“In1”的值	非数	非数	非数
	–∞	0	“In1”的值	非数	非数	非数
	非数	非数	非数	非数	非数	非数

- 如下所示，将整数型的参数传输至 “In1”或 “In2”时，转换数据类型后输入。

传输至 “In1”或 “In2”的参数的数据类型	“In1”或 “In2”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

Fraction

提取实数的小数部分。

指令	名称	FB/FUN	图形表现	ST表现
Fraction	提取实数小数部分	FUN	(@)Fraction EN ENO In Out	Out:=Fraction(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	实数	输入	实数	遵从数据类型	–	(*)
Out	小数部分	输出	小数部分	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

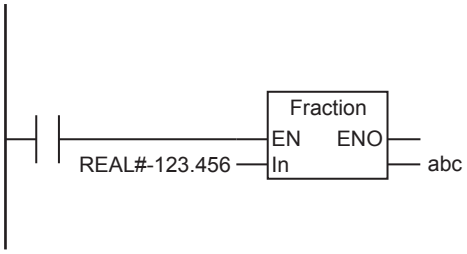
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

功能

提取实数 “In”的小数部分。

“in”=REAL#–123.456时的示例如下所示。变量abc的值为REAL#–0.456。

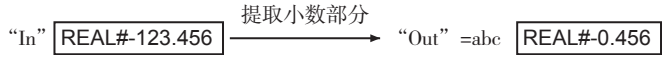
LD



ST

```
abc:=Fraction(REAL#-123.456);
```

提取“In”的小数部分。
–123.456的小数部分为–0.456，因此abc的值为REAL#–0.456。



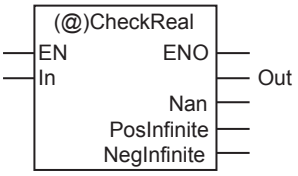
参考

- 请使用 “CheckReal指令(P.2–234)” 检测 “Out”的值是否为+∞、–∞、非数。
- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

CheckReal

判定实数是否无限大或非数。

指令	名称	FB/FUN	图形表现	ST表现
CheckReal	实数检查	FUN		CheckReal(In, Nan, PosInfinite, NegInfinite);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	实数	输入	实数	遵从数据类型	–	(*)
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
Nan	非数判定结果		TRUE：非数 FALSE：不是非数	遵从数据类型		
PosInfinite	正无穷大判定结果		TRUE：正无穷大 FALSE：不是正无穷大			
NegInfinite	负无穷大判定结果		TRUE：负无穷大 FALSE：不是负无穷大			

* 省略输入参数时，初始值不适用。编连时会发生异常。

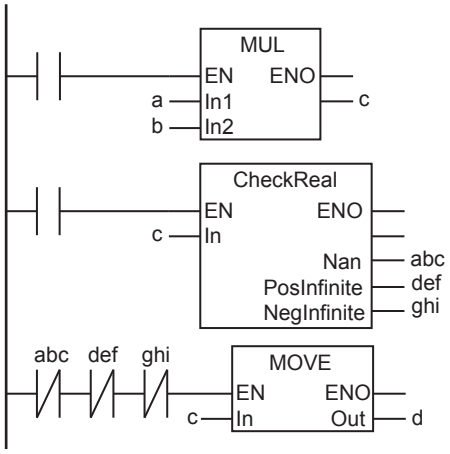
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out	○																			
Nan	○																			
PosInfinite	○																			
NegInfinite	○																			

功能

判定实数 “In”是否为非数、正无穷大、负无穷大，将结果分别输出至 “Nan”、“PosInfinite”、“NegInfinite”。

示例如下所示。检查REAL型的变量a、b的相乘结果是否为有效实数。如果为有效实数，则将相乘结果代入变量d。

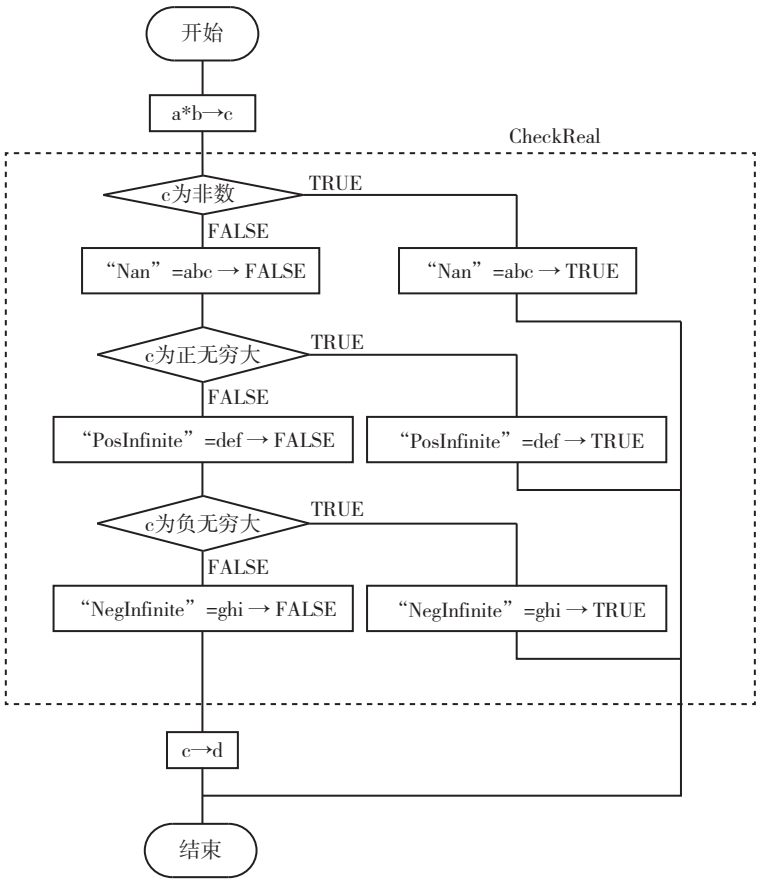
LD



ST

```
c:=a*b;
CheckReal(c, abc, def, ghi);
IF ( (abc=FALSE) AND (def=FALSE) AND (ghi=FALSE) ) THEN
    d:=c;
END_IF;
```

如果a与b之积c为非数、正无穷大、负无穷大中任意一个，则将c代入d。



参考

检测使用实数的运算指令的结果是否为非数、正无穷大、负无穷大时，请使用本指令。

使用注意事项

- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 如下所示，将整数型的参数传输至 “In”时，转换数据类型后输入。

传输至 “In”的参数的数据类型	“In”的数据类型
USINT，UINT，SINT，INT	REAL
UDINT，DINT	LREAL
ULINT，LINT	编连时异常

BCD转换指令

指令	名称	页码
_BCD_TO_*	BCD→无符号整数转换组	2–238
_TO_BCD_*	无符号整数→BCD转换组	2–241
BCD_TO_**	BCD组→无符号整数转换组	2–244
BCDsToBin	带符号BCD→带符号整数转换	2–247
BinToBCDs_**	带符号整数→BCD转换组	2–250
AryToBCD	数组整体BCD转换	2–253
AryToBin	数组整体无符号整数转换	2–255

_BCD_TO_*

将BCD的位串转换为无符号整数。

指令	名称	FB/FUN	图形表现	ST表现
_BCD_TO_*	BCD→无符号整数转换组	FUN	(@)**_BCD_TO_*** EN ENO In Out **为位串的数据类型名称，***为整数的数据类型名称	Out:=**_BCD_TO_*** (In); **为位串的数据类型名称，***为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

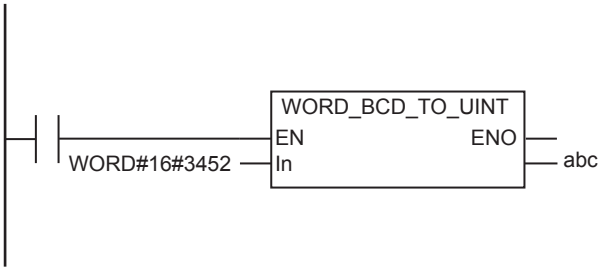
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out						○	○	○	○	○	○	○	○							

功能

将BCD的位串的转换对象 “In”转换为无符号整数。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为WORD型，“Out”为UINT型时，指令名称为WORD_BCD_TO_UINT。

WORD_BCD_TO_UINT指令下，“In”=WORD#16#3452时的示例如下所示。

LD



ST

```
abc:=WORD_BCD_TO_UINT(WORD#16#3452);
```

如下所示，根据数据类型的组合，“In”和“Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
BYTE	USINT	16#00 ~ 16#99(BCD)	0 ~ 99
	UINT		
	UDINT		
	ULINT		
	SINT		
	INT		
	DINT		
	LINT		
WORD	USINT	16#0000 ~ 16#0255(BCD)	0 ~ 255
	UINT	16#0000 ~ 16#9999(BCD)	0 ~ 9999
	UDINT		
	ULINT		
	SINT	16#0000 ~ 16#0127(BCD)	0 ~ 127
	INT	16#0000 ~ 16#9999(BCD)	0 ~ 9999
	DINT		
	LINT		
DWORD	USINT	16#0000_0000 ~ 16#0000_0255(BCD)	0 ~ 255
	UINT	16#0000_0000 ~ 16#0006_5535(BCD)	0 ~ 65535
	UDINT	16#0000_0000 ~ 16#9999_9999(BCD)	0 ~ 99999999
	ULINT		
	SINT	16#0000_0000 ~ 16#0000_0127(BCD)	0 ~ 127
	INT	16#0000_0000 ~ 16#0003_2767(BCD)	0 ~ 32767
	DINT	16#0000_0000 ~ 16#9999_9999(BCD)	0 ~ 99999999
	LINT		
LWORD	USINT	16#0000_0000_0000_0000 ~ 16#0000_0000_0000_0255(BCD)	0 ~ 255
	UINT	16#0000_0000_0000_0000 ~ 16#0000_0000_0006_5535(BCD)	0 ~ 65535
	UDINT	16#0000_0000_0000_0000 ~ 16#0000_0042_9496_7295(BCD)	0 ~ 4294967295
	ULINT	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)	0 ~ 9999999999999999
	SINT	16#0000_0000_0000_0000 ~ 16#0000_0000_0000_0127(BCD)	0 ~ 127
	INT	16#0000_0000_0000_0000 ~ 16#0000_0000_0003_2767(BCD)	0 ~ 32767
	DINT	16#0000_0000_0000_0000 ~ 16#0000_0021_4748_3647(BCD)	0 ~ 2147483647
	LINT	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)	0 ~ 9999999999999999

参考

- 从任意的BCD位串转换为整数时，请使用 “BCD_TO_**指令(P.2-244)”。
- 从整数转换为BCD位串时，请使用 “**_TO_BCD_***指令(P.2-241)”。

使用注意事项

- 指令名称请务必符合 “In” 和 “Out” 的数据类型。
- “Out” 的数据大小大于 “In” 的数据大小时，在 “Out” 的高位添加0。
- 以下情况时会发生异常。ENO变为FALSE，“Out” 不变。
  - “In” 的值超过有效范围时。
  - “In” 的值不是BCD的位串(16进制标记包含A，B，C，D，E，F中任意一个)时。

_TO_BCD_*

将无符号整数转换为BCD的位串。

指令	名称	FB/FUN	图形表现	ST表现
_TO_BCD_*	无符号整数→BCD转换组	FUN	(@)**_TO_BCD_*** EN ENO In Out **为整数的数据类型名称，***为位串的数据类型名称	Out:=**_TO_BCD_*** (In); **为整数的数据类型名称，***为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

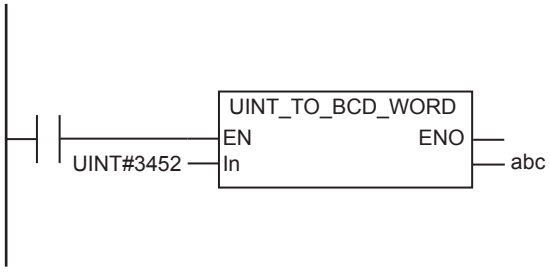
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○							
Out		○	○	○	○															

功能

将无符号整数的转换对象 “In”转换为BCD的位串。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为UINT型，“Out”为WORD型时，指令名称为UINT_TO_BCD_WORD。

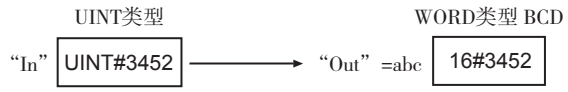
UINT_TO_BCD_WORD指令下，“In”=UINT#3452时的示例如下所示。

LD



ST

```
abc:=UINT_TO_BCD_WORD(UINT#3452);
```



如下所示，根据数据类型的组合，“In”和 “Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
USINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 255	16#0000 ~ 16#0255(BCD)
	DWORD		16#0000_0000 ~ 16#0000_0255(BCD)
	LWORD		16#0000_0000_0000_0000 ~ 16#0000_0000_0000_0255(BCD)
UINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
	DWORD	0 ~ 65535	16#0000_0000 ~ 16#0006_5535(BCD)
	LWORD		16#0000_0000_0000_0000 ~ 16#0000_0000_0006_5535(BCD)
UDINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
	DWORD	0 ~ 99999999	16#0000_0000 ~ 16#9999_9999(BCD)
	LWORD	0 ~ 4294967295	16#0000_0000_0000_0000 ~ 16#0000_0042_9496_7295(BCD)
ULINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
	DWORD	0 ~ 99999999	16#0000_0000 ~ 16#9999_9999(BCD)
	LWORD	0 ~ 9999999999999999	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)
SINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 127	16#0000 ~ 16#0127(BCD)
	DWORD		16#0000_0000 ~ 16#0000_0127(BCD)
	LWORD		16#0000_0000_0000_0000 ~ 16#0000_0000_0000_0127(BCD)
INT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
	DWORD	0 ~ 32767	16#0000_0000 ~ 16#0003_2767(BCD)
	LWORD		16#0000_0000_0000_0000 ~ 16#0000_0000_0003_2767(BCD)
DINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
	DWORD	0 ~ 99999999	16#0000_0000 ~ 16#9999_9999(BCD)
	LWORD	0 ~ 2147483647	16#0000_0000_0000_0000 ~ 16#0000_0021_4748_3647(BCD)
LINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
	DWORD	0 ~ 99999999	16#0000_0000 ~ 16#9999_9999(BCD)
	LWORD	0 ~ 9999999999999999	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)

参考

- 从特定的BCD位串转换为整数时，请使用 “**_BCD_TO_***指令(P.2–238)”。
- 从任意的BCD位串转换为整数时，请使用 “BCD_TO_**指令(P.2–244)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- “Out”的数据大小大于 “In”的数据大小时，在 “Out”的高位添加0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值超过有效范围时。

BCD_TO_**

将BCD的位串转换为无符号整数。

指令	名称	FB/FUN	图形表现	ST表现
BCD_TO_**	BCD组→无符号整数转换组	FUN	(@)BCD_TO_** EN ENO In Out **为整数的数据类型名称	Out:=BCD_TO_** (In); **为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*1)	–	(*2)
Out	转换结果	输出	转换结果	(*1)	–	–

*1 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

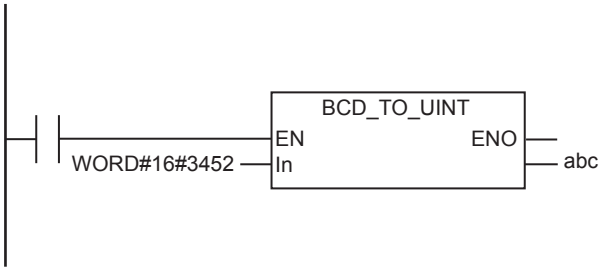
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out						○	○	○	○	○	○	○	○							

功能

将BCD的位串的转换对象 “In”转换为无符号整数。
指令名称因转换结果 “Out”的数据类型而异。例如，“Out”为UINT型时，指令名称为BCD_TO_UINT。

BCD_TO_UINT指令下，“In”=WORD#16#3452时的示例如下所示。

LD



ST

```
abc:=BCD_TO_UINT(WORD#16#3452);
```

如下所示，根据数据类型的组合，“In”和“Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
BYTE	USINT	16#00 ~ 16#99(BCD)	0 ~ 99
	UINT		
	UDINT		
	ULINT		
	SINT		
	INT		
	DINT		
	LINT		
WORD	USINT	16#0000 ~ 16#0255(BCD)	0 ~ 255
	UINT	16#0000 ~ 16#9999(BCD)	0 ~ 9999
	UDINT		
	ULINT		
	SINT	16#0000 ~ 16#0127(BCD)	0 ~ 127
	INT	16#0000 ~ 16#9999(BCD)	0 ~ 9999
	DINT		
	LINT		
DWORD	USINT	16#0000_0000 ~ 16#0000_0255(BCD)	0 ~ 255
	UINT	16#0000_0000 ~ 16#0006_5535(BCD)	0 ~ 65535
	UDINT	16#0000_0000 ~ 16#9999_9999(BCD)	0 ~ 99999999
	ULINT		
	SINT	16#0000_0000 ~ 16#0000_0127(BCD)	0 ~ 127
	INT	16#0000_0000 ~ 16#0003_2767(BCD)	0 ~ 32767
	DINT	16#0000_0000 ~ 16#9999_9999(BCD)	0 ~ 99999999
	LINT		
LWORD	USINT	16#0000_0000_0000_0000 ~ 16#0000_0000_0000_0255(BCD)	0 ~ 255
	UINT	16#0000_0000_0000_0000 ~ 16#0000_0000_0006_5535(BCD)	0 ~ 65535
	UDINT	16#0000_0000_0000_0000 ~ 16#0000_0042_9496_7295(BCD)	0 ~ 4294967295
	ULINT	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)	0 ~ 9999999999999999
	SINT	16#0000_0000_0000_0000 ~ 16#0000_0000_0000_0127(BCD)	0 ~ 127
	INT	16#0000_0000_0000_0000 ~ 16#0000_0000_0003_2767(BCD)	0 ~ 32767
	DINT	16#0000_0000_0000_0000 ~ 16#0000_0021_4748_3647(BCD)	0 ~ 2147483647
	LINT	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)	0 ~ 9999999999999999

参考

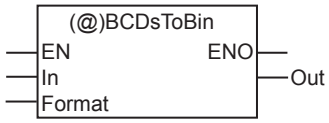
- 从特定的BCD位串转换为整数时，请使用 “**_BCD_TO_***指令(P.2-238)”。
- 从整数转换为BCD位串时，请使用 “**_TO_BCD_***指令(P.2-241)”。

使用注意事项

- 指令名称请务必符合 “Out” 的数据类型。
- “Out” 的数据大小大于 “In” 的数据大小时，在 “Out” 的高位添加0。
- 以下情况时会发生异常。ENO变为FALSE，“Out” 不变。
  - “In” 的值超过有效范围时。
  - “In” 的值不是BCD的位串(16进制标记包含A，B，C，D，E，F中任意一个)时。

BCDsToBin

将带符号BCD的位串转换为带符号整数。

指令	名称	FB/FUN	图形表现	ST表现
BCDsToBin	带符号BCD→带符号整数转换	FUN		Out:=BCDsToBin(In, Format);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*1)	–	(*2)
Format	数据格式指定编号		BCD的位串格式	_BCD0 ~ _BCD3		_BCD0
Out	转换结果	输出	转换结果	(*1)	–	–

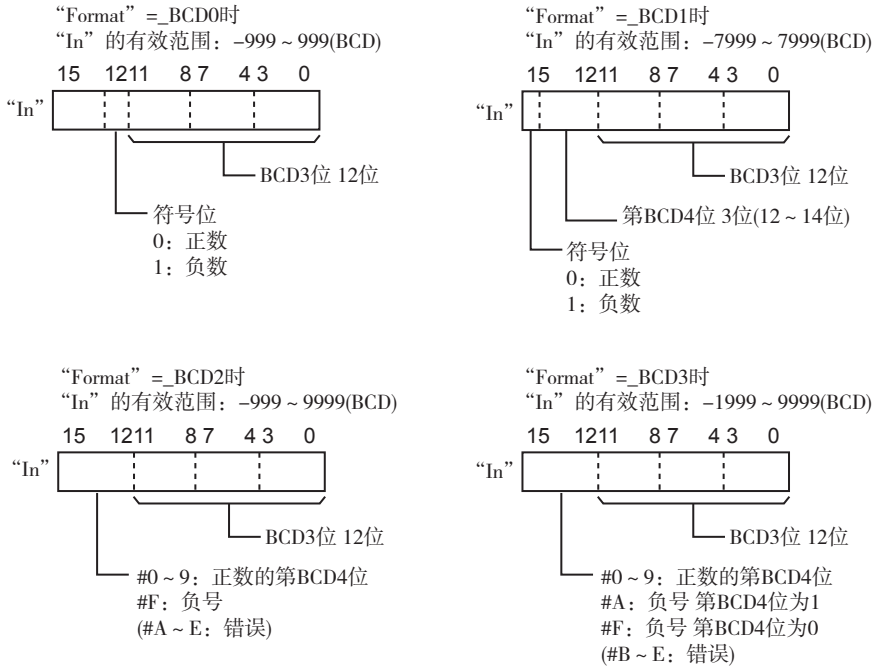
*1 有效范围因 “Format”的值而异。详情请参阅功能说明。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Format	枚举体_eBCD_FORMAT 枚举元素参阅功能说明																			
Out	与 “In”大小相同的带符号整数的数据类型																			

功能

将带符号BCD的位串 “In”转换为带符号整数。

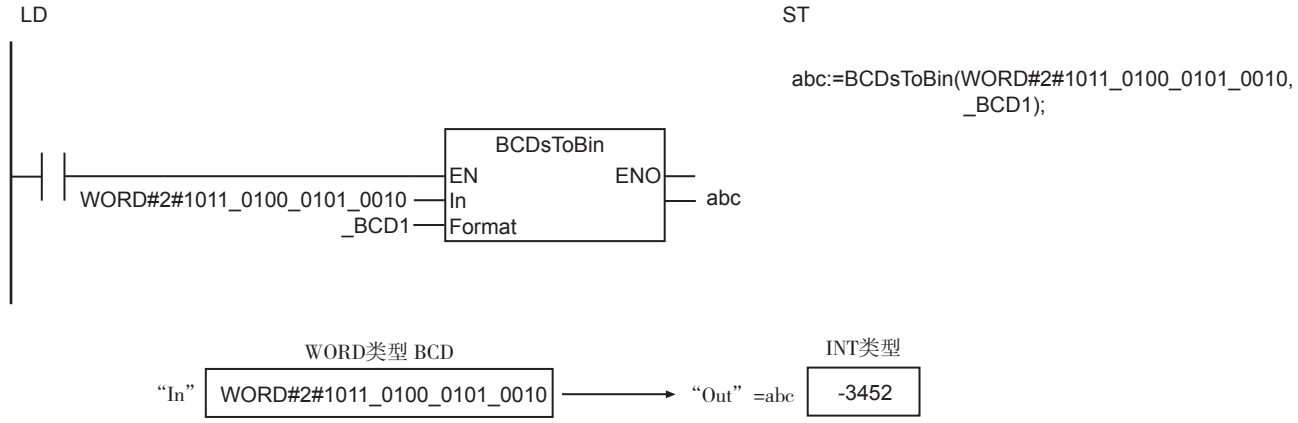
数据格式指定编号“Format”的数据类型为枚举体_eBCD_FORMAT。枚举元素为_BCD0、_BCD1、_BCD2、_BCD3四种。“In”的最高4位的符号表现方式因该值而异。以 “In”为WORD型时的情形为例，数据格式如下所示。



“In”和 “Out”数据类型的大小相同。如下所示，各自的有效范围因 “Format”的值而异。

		“Format”的值			
		_BCD0	_BCD1	_BCD2	_BCD3
“In”的数据类型↓“Out”的数据类型	BYTE↓SINT	–9～9	–79～79	–9～99	–19～99
	WORD↓INT	–999～999	–7999～7999	–999～9999	–1999～9999
	DWORD↓DINT	–9999999～9999999	–79999999～79999999	–9999999～99999999	–19999999～99999999
	LWORD↓LINT	–999999999999999～999999999999999	–7999999999999999～7999999999999999	–999999999999999～9999999999999999	–1999999999999999～9999999999999999

“In”=WORD#2#1011_0100_0101_0010，“Format”=_BCD1时的示例如下所示。



使用注意事项

- 请将 “In”和 “Out”的数据类型的大小设为相同。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Format”的值为_BCD0，“In”的最高位为2～F时。
  - “Format”的值为_BCD2，“In”的最高位为A～E时。
  - “Format”的值为_BCD3，“In”的最高位为B～E时。
  - 除上述条件以外，“In”的任意数位为A～F时。
  - “Format”的值超过有效范围时。

BinToBCDs_**

将带符号整数转换为带符号BCD的位串。

指令	名称	FB/FUN	图形表现	ST表现
BinToBCDs_**	带符号整数→BCD转换组	FUN	(@)BinToBCDs_** EN ENO In Out Format **为位串的数据类型名称	Out:=BinToBCDs_**(In, Format); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Format	数据格式指定编号		BCD的位串格式	_BCD0 ~ _BCD3		_BCD0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “Format”的值而异。详情请参阅功能说明。

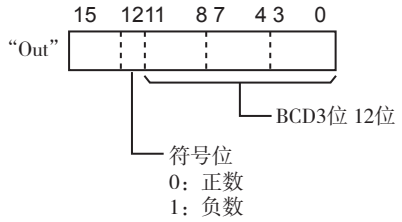
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In										○	○	○	○							
Format	枚举体_eBCD_FORMAT 枚举元素参阅功能说明																			
Out	与 “In”大小相同的位串的数据类型																			

功能

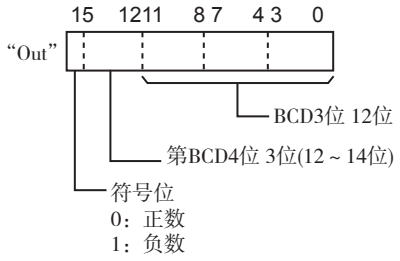
将带符号整数 “In”转换为带符号BCD的位串。
指令名称因 “Out”的数据类型而异。例如，“Out”为WORD型时，指令名称为BinToBCDs_WORD。

数据格式指定编号“Format”的数据类型为枚举体_eBCD_FORMAT。枚举元素为_BCD0、_BCD1、_BCD2、_BCD3四种。“Out”的最高4位的符号表现方式因该值而异。以 “Out”为WORD型时的情形为例，数据格式表示如下。

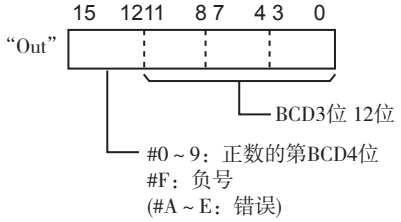
“Format”=_BCD0时
“Out”的有效范围：–999 ~ 999(BCD)



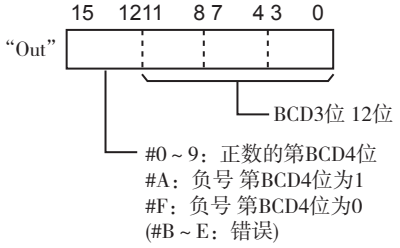
“Format”=_BCD1时
“Out”的有效范围：–7999 ~ 7999(BCD)



“Format”=_BCD2时
“Out”的有效范围：–999 ~ 9999(BCD)



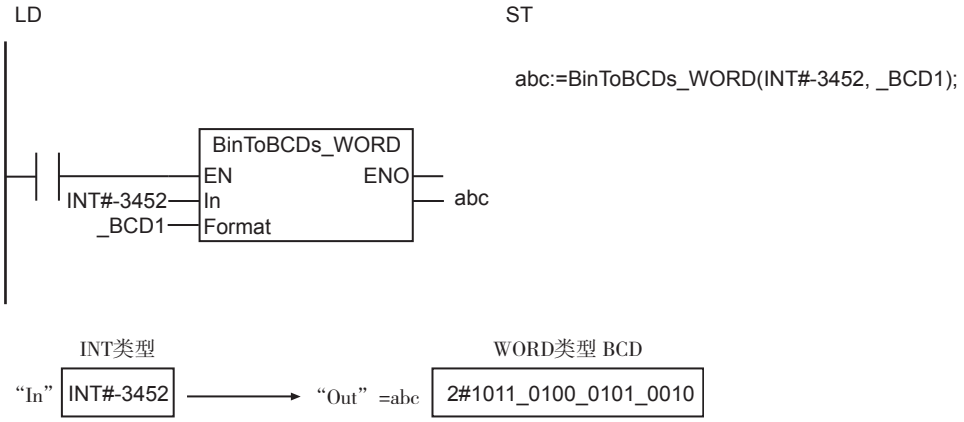
“Format”=_BCD3时
“Out”的有效范围：–1999 ~ 9999(BCD)



“In”和 “Out”的数据类型的大小相同。如下所示，各自的有效范围因 “Format”的值而异。

		“Format”的值			
		_BCD0	_BCD1	_BCD2	_BCD3
“In”的数据类型 ↓ “Out”的数据类型	SINT ↓ BYTE	–9 ~ 9	–79 ~ 79	–9 ~ 99	–19 ~ 99
	INT ↓ WORD	–999 ~ 999	–7999 ~ 7999	–999 ~ 9999	–1999 ~ 9999
	DINT ↓ DWORD	–9999999 ~ 9999999	–79999999 ~ 79999999	–9999999 ~ 99999999	–19999999 ~ 99999999
	LINT ↓ LWORD	–999999999999999 ~ 999999999999999	–7999999999999999 ~ 7999999999999999	–9999999999999999 ~ 9999999999999999	–19999999999999999 ~ 19999999999999999

BinToBCDs_WORD指令下，“In”=INT#-3452、“Format”=_BCD1时的示例如下所示。

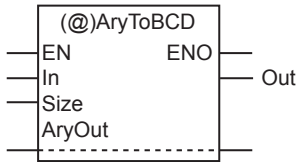


使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值超过有效范围时。
  - “Format”的值超过有效范围时。

AryToBCD

将无符号整数数组的各元素转换为BCD的位串。

指令	名称	FB/FUN	图形表现	ST表现
AryToBCD	数组整体BCD转换	FUN		AryToBCD(In, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	无符号整数数组	输入	无符号整数数组	(*1)	–	(*2)
Size	元素数		待转换的In[ ]的元素数	遵从数据类型		1
AryOut[ ] 数组	BCD数组	输入输出	BCD数组	(*1)	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

*1 有效范围因In[ ]、AryOut[ ]的元素的数据类型的组合而异。详情请参阅功能说明。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

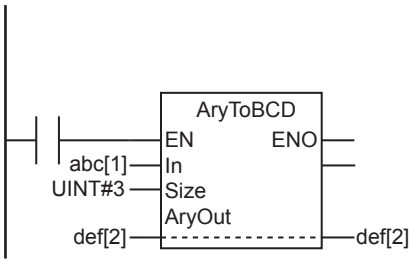
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组						○	○	○	○											
Size							○													
AryOut[ ] 数组	位串的数组。与In[ ]的元素大小相同的数据类型。																			
Out	○																			

功能

将无符号整数数组In[ ]的从In[0]起的 “Size”个的各元素转换为BCD的位串，输出至BCD数组AryOut[ ]。

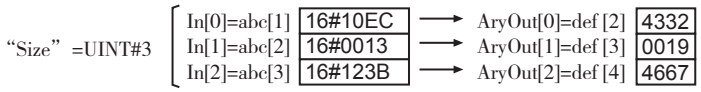
“Size” =UINT#3时的示例如下所示。

LD



ST

```
AryToBCD(abc[1], UINT#3, def[2]);
```



如下所示，根据元素的数据类型的组合，In[ ]和AryOut[ ]的有效范围有所差异。

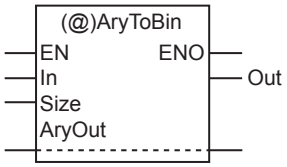
In[ ]元素的数据类型	AryOut[ ]元素的数据类型	In[ ]的有效范围	AryOut[ ]的有效范围
USINT	BYTE	0 ~ 99	16#00 ~ 16#99(BCD)
UINT	WORD	0 ~ 9999	16#0000 ~ 16#9999(BCD)
UDINT	DWORD	0 ~ 99999999	16#0000_0000 ~ 16#9999_9999(BCD)
ULINT	LWORD	0 ~ 9999999999999999	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)

使用注意事项

- 请将 In[ ]的元素和 AryOut[ ]的元素的数据类型的大小设为相同。例如，In[ ]的元素为 UINT 型时，请将AryOut[ ]的元素设为WORD型。
- 无法进行带符号BCD转换。因此，请将In[ ]的数据类型设为无符号整数(USINT、UINT、UDINT、ULINT)中任意一个。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - In[ ]的值超过有效范围时。
  - “Size”的值超过In[ ]或AryOut[ ]的数组区域时。

AryToBin

将BCD的位串数组的各元素转换为无符号整数。

指令	名称	FB/FUN	图形表现	ST表现
AryToBin	数组整体无符号整数转换	FUN		AryToBin(In, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	BCD的位串数组	输入	BCD的位串数组	(*1)	–	(*2)
Size	元素数		待转换的In[ ]的元素数	遵从数据类型		1
AryOut[ ] 数组	无符号整数数组	输入输出	无符号整数数组	(*1)	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

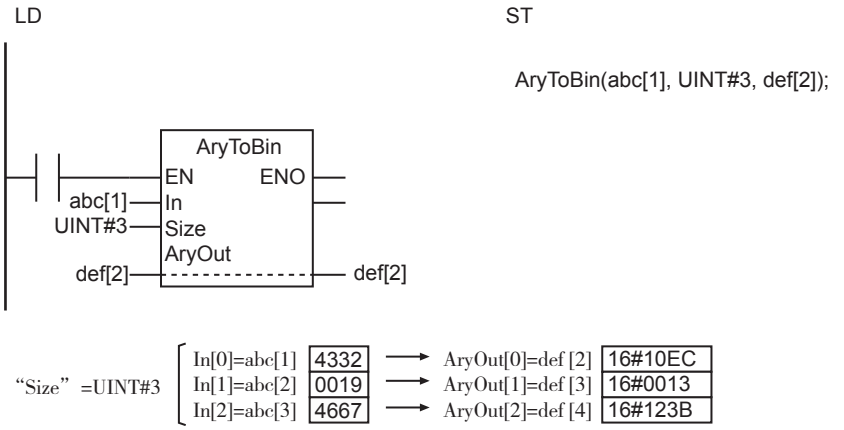
*1 有效范围因In[ ]、AryOut[ ]的元素的数据类型的组合而异。详情请参阅功能说明。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○	○	○	○															
Size							○													
AryOut[ ] 数组	无符号整数的数组。与In[ ]的元素大小相同的数据类型。																			
Out	○																			

功能

将BCD的位串数组In[ ]的从In[0]起的 “Size”个各元素转换为无符号整数，输出至无符号整数数组AryOut[ ]。

“Size”=UINT#3时的示例如下所示。



如下所示，根据元素的数据类型的组合，In[ ]和AryOut[ ]的有效范围有所差异。

In[ ]元素的数据类型	AryOut[ ]元素的数据类型	In[ ]的有效范围	AryOut[ ]的有效范围
BYTE	USINT	16#00 ~ 16#99(BCD)	0 ~ 99
WORD	UINT	16#0000 ~ 16#9999(BCD)	0 ~ 9999
DWORD	UDINT	16#0000_0000 ~ 16#9999_9999(BCD)	0 ~ 99999999
LWORD	ULINT	16#0000_0000_0000_0000 ~ 16#9999_9999_9999_9999(BCD)	0 ~ 9999999999999999

使用注意事项

- 请将In[ ]和AryOut[ ]的数据类型的大小设为相同。例如，In[ ]的元素为WORD型时，请将AryOut[ ]的元素设为USINT型。
- 无法进行带符号 BCD 转换。因此，请将 AryOut[ ] 的数据类型设为无符号整数 (USINT、UINT、UDINT、ULINT)中任意一个。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Size”的值超过In[ ]或AryOut[ ]的数组区域时。
  - In[ ]的值不是BCD的位串(16进制标记包含A，B，C，D，E，F中任意一个)时。

数据类型转换指令

_TO_*(整数→整数转换组)

将整数转换为不同数据类型的整数。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	整数→整数转换组	FUN	(@)**_TO_*** EN ENO In Out **、***为不同整数的数据类型名称	Out:=**_TO_*** (In); **、***为不同整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

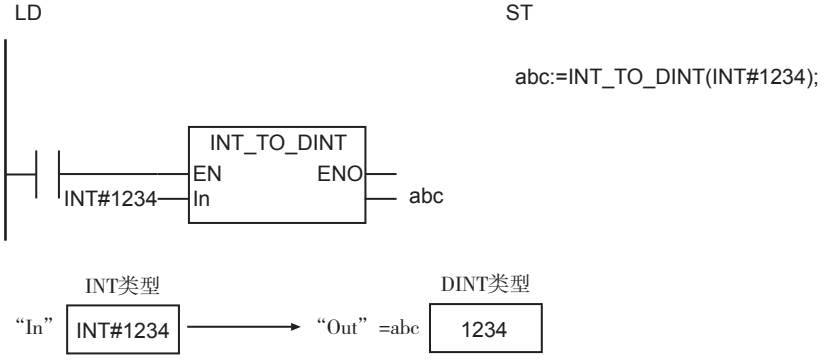
* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○							
Out						○	○	○	○	○	○	○	○							

功能

将整数 “In”转换为不同数据类型的整数。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为INT型，“Out”为DINT型时，指令名称为INT_TO_DINT。

INT_TO_DINT指令下，“In”=INT#1234的示例如下所示。



如下所示，根据数据类型的组合，“In”和 “Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”“Out”的有效范围
USINT	UINT	0 ~ 255
	UDINT	
	ULINT	
	SINT	0 ~ 127
	INT	0 ~ 255
	DINT	
	LINT	
UINT	USINT	0 ~ 255
	UDINT	0 ~ 65535
	ULINT	
	SINT	0 ~ 127
	INT	0 ~ 32767
	DINT	0 ~ 65535
	LINT	
UDINT	USINT	0 ~ 255
	UINT	0 ~ 65535
	ULINT	0 ~ 4294967295
	SINT	0 ~ 127
	INT	0 ~ 32767
	DINT	0 ~ 2147483647
	LINT	0 ~ 4294967295
ULINT	USINT	0 ~ 255
	UINT	0 ~ 65535
	UDINT	0 ~ 4294967295
	SINT	0 ~ 127
	INT	0 ~ 32767
	DINT	0 ~ 2147483647
	LINT	0 ~ 9223372036854775807
SINT	USINT	0 ~ 127
	UINT	
	UDINT	
	ULINT	
	INT	–128 ~ 127
	DINT	
	LINT	
INT	USINT	0 ~ 255
	UINT	0 ~ 32767
	UDINT	
	ULINT	
	SINT	–128 ~ 127
	DINT	–32768 ~ 32767
	LINT	
DINT	USINT	0 ~ 255
	UINT	0 ~ 65535
	UDINT	0 ~ 2147483647
	ULINT	
	SINT	–128 ~ 127
	INT	–32768 ~ 32767
	LINT	–2147483648 ~ 2147483647

“In”的数据类型	“Out”的数据类型	“In”“Out”的有效范围
LINT	USINT	0 ~ 255
	UINT	0 ~ 65535
	UDINT	0 ~ 4294967295
	ULINT	0 ~ 9223372036854775807
	SINT	−128 ~ 127
	INT	−32768 ~ 32767
	DINT	−2147483648 ~ 2147483647

参考

从任意的数据类型转换为整数型时，请使用 “TO_**(整数转换组)指令(P.2−302)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- “In”为带符号的整数，“Out”的数据大小大于 “In”的数据大小时，进行符号扩展。
- “In”为无符号的整数，“Out”的数据大小大于 “In”的数据大小时，在 “Out”的高位添加0。
- “Out”的数据大小小于 “In”的数据大小时，舍去高位。

_TO_*(整数→位串转换组)

将整数转换为位串。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	整数→位串转换组	FUN	(@)**_TO_*** EN ENO In Out **为整数的数据类型名称， ***为位串的数据类型名称	Out:=**_TO_*** (In); **为整数的数据类型名称，***为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out” 的数据类型的组合而异。详情请参阅功能说明。

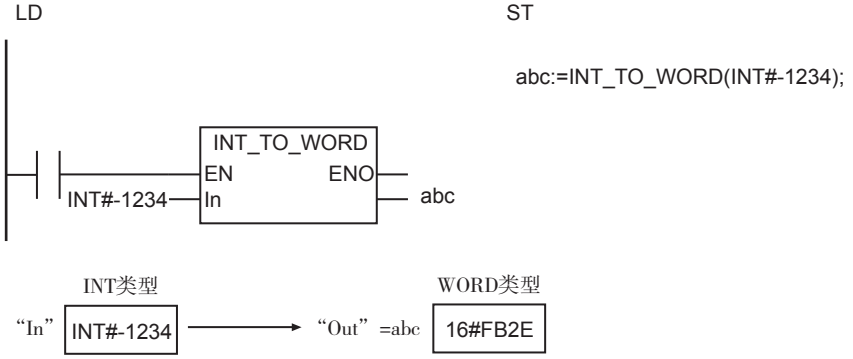
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○*1							
Out		○	○	○	○															

*1 LINT_TO_WORD指令无法在NX1P2 CPU单元Ver.1.13中使用。

功能

将整数 “In” 转换为位串。
指令名称因 “In” 和转换结果 “Out” 的数据类型的组合而异。例如，“In” 为INT型，“Out” 为WORD型时，指令名称为INT_TO_WORD。

INT_TO_WORD指令下，“In” =INT#–1234时的示例如下所示。



如下所示，根据数据类型的组合，“In”和 “Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
USINT	BYTE	0 ~ 255	16#00 ~ 16#FF
	WORD		
	DWORD		
	LWORD		
UINT	BYTE	0 ~ 255	16#00 ~ 16#FF
	WORD	0 ~ 65535	16#0000 ~ 16#FFFF
	DWORD		
	LWORD		
UDINT	BYTE	0 ~ 255	16#00 ~ 16#FF
	WORD	0 ~ 65535	16#0000 ~ 16#FFFF
	DWORD	0 ~ 4294967295	16#0000_0000 ~ 16#FFFF_FFFF
	LWORD		
ULINT	BYTE	0 ~ 255	16#00 ~ 16#FF
	WORD	0 ~ 65535	16#0000 ~ 16#FFFF
	DWORD	0 ~ 4294967295	16#0000_0000 ~ 16#FFFF_FFFF
	LWORD	0 ~ 18446744073709551645	16#0000_0000_0000_0000 ~ 16#FFFF_FFFF_FFFF_FFFF
SINT	BYTE	−128 ~ 127	16#00 ~ 16#FF
	WORD		
	DWORD		
	LWORD		
INT	BYTE	−128 ~ 127	16#00 ~ 16#FF
	WORD	−32768 ~ 32767	16#0000 ~ 16#FFFF
	DWORD		
	LWORD		
DINT	BYTE	−128 ~ 127	16#00 ~ 16#FF
	WORD	−32768 ~ 32767	16#0000 ~ 16#FFFF
	DWORD	−2147483648 ~ 2147483647	16#0000_0000 ~ 16#FFFF_FFFF
	LWORD		
LINT	BYTE	−128 ~ 127	16#00 ~ 16#FF
	WORD	−32768 ~ 32767	16#0000 ~ 16#FFFF
	DWORD	−2147483648 ~ 2147483647	16#0000_0000 ~ 16#FFFF_FFFF
	LWORD	−9223372036854775808 ~ 9223372036854775807	16#0000_0000_0000_0000 ~ 16#FFFF_FFFF_FFFF_FFFF

参考

- 从位串转换为整数时，请使用 “**_TO_***(位串→整数转换组)指令(P.2−266)”。
- 从任意的数据类型转换为位串时，请使用 “TO_**(位串转换组)指令(P.2−304)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- “In”为带符号的整数，“Out”的数据大小大于 “In”的数据大小时，进行符号扩展。
- “In”为无符号的整数，“Out”的数据大小大于 “In”的数据大小时，在 “Out”的高位添加0。
- “Out”的数据大小小于 “In”的数据大小时，舍去高位。

_TO_*(整数→实数转换组)

将整数转换为实数。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	整数→实数转换组	FUN	(@)**_TO_*** EN ENO In Out **为整数的数据类型名称， ***为实数的数据类型名称	Out:=**_TO_*** (In); **为整数的数据类型名称，***为实数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

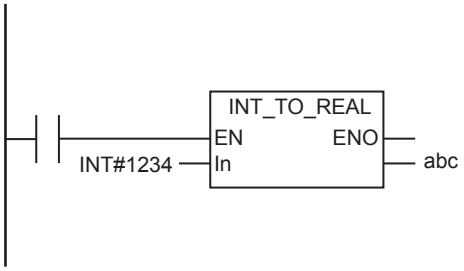
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○							
Out														○	○					

功能

将整数 “In”转换为实数。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为INT型，“Out”为REAL型时，指令名称为INT_TO_REAL。

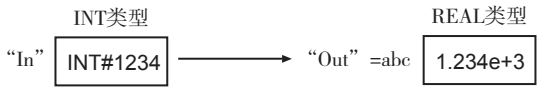
INT_TO_REAL指令下，“In”=INT#1234时的示例如下所示。

LD



ST

```
abc:=INT_TO_REAL(INT#1234);
```



如下所示，根据数据类型的组合，“In”和“Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
USINT	REAL	0 ~ 255	0 ~ 2.55e+2
	LREAL		
UINT	REAL	0 ~ 65535	0 ~ 6.5535e+4
	LREAL		
UDINT	REAL	0 ~ 4294967295	0 ~ 4.294967e+9
	LREAL		0 ~ 4.294967295e+9
ULINT	REAL	0 ~ 18446744073709551615	0 ~ 1.844674e+19
	LREAL		0 ~ 1.84467440737095e+19
SINT	REAL	−128 ~ 127	−1.28e+2 ~ 1.27e+2
	LREAL		
INT	REAL	−32768 ~ 32767	−3.2768e+4 ~ 3.2767e+4
	LREAL		
DINT	REAL	−2147483648 ~ 2147483647	−2.147483e+9 ~ 2.147483e+9
	LREAL		−2.147483648e+9 ~ 2.147483647e+9
LINT	REAL	−9223372036854775808 ~ 9223372036854775807	−9.223372e+18 ~ 9.223372e+18
	LREAL		−9.22337203685477e+18 ~ 9.22337203685477e+18

参考

- 从实数转换为整数时，请使用 “**_TO_***(实数→整数转换组)指令(P.2−272)”。
- 从任意的数据类型转换为实数时，请使用 “TO_**(实数转换组)指令(P.2−306)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- 根据 “In”和 “Out”的数据类型的组合，有时会发生与实数的有效数位相应的取整，导致转换前后的值产生差异。会发生差异的数据类型的组合如下所示。

“In”的数据类型	“Out”的数据类型	发生差异的值
DINT	REAL	-16777216以下或16777216以上
LINT		
UDINT	REAL	16777216以上
ULINT		
LINT	LREAL	-9007199254740992以下或9007199254740992以上
ULINT	LREAL	9007199254740992以上

_TO_*(位串→整数转换组)

将位串转换为整数。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	位串→整数转换组	FUN	(@)**_TO_*** EN ENO In Out **为位串的数据类型名称， ***为整数的数据类型名称	Out:=**_TO_*** (In); **为位串的数据类型名称，***为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

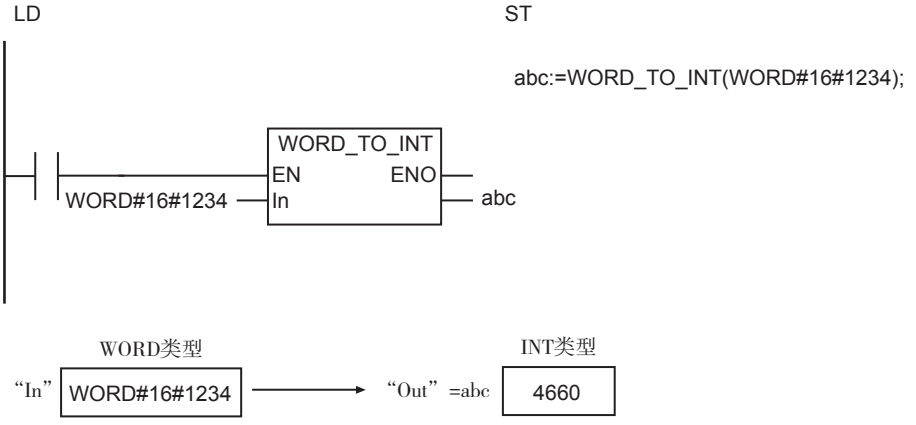
* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out						○	○	○	○	○	○	○	○							

功能

将位串 “In”转换为整数。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为WORD型，“Out”为INT型时，指令名称为WORD_TO_INT。

WORD_TO_INT指令下，“In”=WORD#16#1234时的示例如下所示。



如下所示，根据数据类型的组合，“In”和 “Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
BYTE	USINT	16#00 ~ 16#FF	0 ~ 255
	UINT		
	UDINT		
	ULINT		
	SINT		–128 ~ 127
	INT		
	DINT		
	LINT		
WORD	USINT	16#00 ~ 16#FF	0 ~ 255
	UINT	16#0000 ~ 16#FFFF	0 ~ 65535
	UDINT		
	ULINT		
	SINT	16#00 ~ 16#FF	–128 ~ 127
	INT	16#0000 ~ 16#FFFF	–32768 ~ 32767
	DINT		
	LINT		
DWORD	USINT	16#00 ~ 16#FF	0 ~ 255
	UINT	16#0000 ~ 16#FFFF	0 ~ 65535
	UDINT	16#0000_0000 ~ 16#FFFF_FFFF	0 ~ 4294967295
	ULINT		
	SINT	16#00 ~ 16#FF	–128 ~ 127
	INT	16#0000 ~ 16#FFFF	–32768 ~ 32767
	DINT	16#0000_0000 ~ 16#FFFF_FFFF	–2147483648 ~ 2147483647
	LINT		
LWORD	USINT	16#00 ~ 16#FF	0 ~ 255
	UINT	16#0000 ~ 16#FFFF	0 ~ 65535
	UDINT	16#0000_0000 ~ 16#FFFF_FFFF	0 ~ 4294967295
	ULINT	16#0000_0000_0000_0000 ~ 16#FFFF_FFFF_FFFF_FFFF	0 ~ 18446744073709551645
	SINT	16#00 ~ 16#FF	–128 ~ 127
	INT	16#0000 ~ 16#FFFF	–32768 ~ 32767
	DINT	16#0000_0000 ~ 16#FFFF_FFFF	–2147483648 ~ 2147483647
	LINT	16#0000_0000_0000_0000 ~ 16#FFFF_FFFF_FFFF_FFFF	–9223372036854775808 ~ 9223372036854775807

参考

- 从整数转换为位串时，请使用 “**_TO_***(整数→位串转换组)指令(P.2–261)”。
- 从任意的数据类型转换为位串时，请使用 “TO_**(位串转换组)指令(P.2–304)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- “Out”的数据大小大于 “In”的数据大小时，在 “Out”的高位添加0。
- “Out”的数据大小小于 “In”的数据大小时，舍去高位。

_TO_*(位串→位串转换组)

将位串转换为不同数据类型的位串。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	位串→位串转换组	FUN	(@)**_TO_*** EN ENO In Out **、***为不同位串的数据类型名称	Out:=**_TO_*** (In); **、***为不同位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

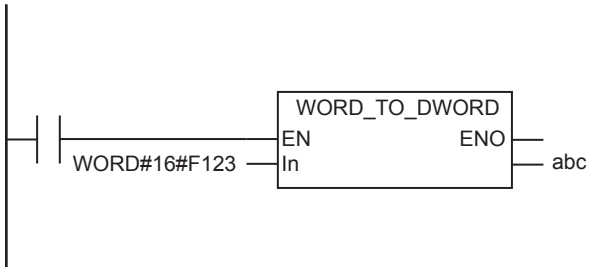
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out		○	○	○	○															

功能

将位串 “In”转换为不同数据类型的位串。
指令名称因“In”和转换结果“Out”的数据类型的组合而异。例如，“In”为WORD型，“Out”为DWORD型时，指令名称为WORD_TO_DWORD。

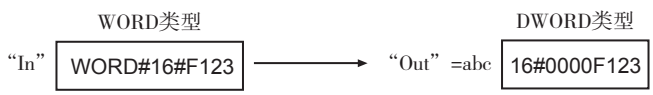
WORD_TO_DWORD指令下，“In”=WORD#16#F123时的示例如下所示。

LD



ST

```
abc:=WORD_TO_DWORD(WORD#16#F123);
```



如下所示，根据数据类型的组合，“In”和 “Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”“Out”的有效范围
BYTE	WORD	16#00 ~ 16#FF
	DWORD	
	LWORD	
WORD	BYTE	16#00 ~ 16#FF
	DWORD	16#0000 ~ 16#FFFF
	LWORD	
DWORD	BYTE	16#00 ~ 16#FF
	WORD	16#0000 ~ 16#FFFF
	LWORD	16#0000_0000 ~ 16#FFFF_FFFF
LWORD	BYTE	16#00 ~ 16#FF
	WORD	16#0000 ~ 16#FFFF
	DWORD	16#0000_0000 ~ 16#FFFF_FFFF

参考

从任意的数据类型转换为位串时，请使用 “TO_**(位串转换组)指令(P.2–304)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- “Out”的数据大小大于 “In”的数据大小时，在 “Out”的高位添加0。
- “Out”的数据大小小于 “In”的数据大小时，舍去高位。

_TO_*(位串→实数转换组)

将位串转换为实数。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	位串→实数转换组	FUN	(@)**_TO_*** EN ENO In Out **为位串的数据类型名称， ***为实数的数据类型名称	Out:=**_TO_*** (In); **为位串的数据类型名称，***为实数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

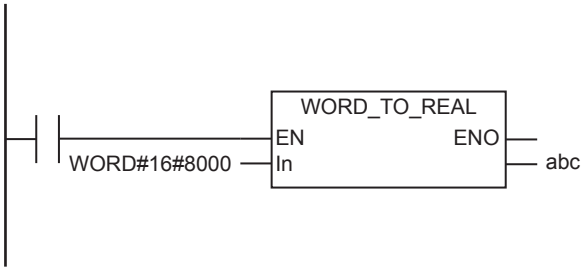
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out														○	○					

功能

将位串 “In”视为相同大小的无符号整数，将其转换为实数型。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为WORD型，“Out”为REAL型时，指令名称为WORD_TO_REAL。

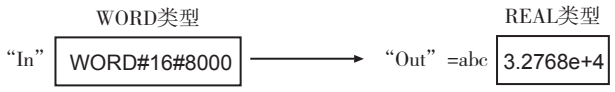
WORD_TO_REAL指令下，“In”=WORD#16#8000时的示例如下所示。

LD



ST

```
abc:=WORD_TO_REAL(WORD#16#8000);
```



如下所示，根据数据类型的组合，“In”和“Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
BYTE	REAL	16#00 ~ 16#FF	0 ~ 2.55e+2
	LREAL		
WORD	REAL	16#0000 ~ 16#FFFF	0 ~ 6.5535e+4
	LREAL		
DWORD	REAL	16#0000_0000 ~ 16#FFFF_FFFF	0 ~ 4.294967e+9
	LREAL		0 ~ 4.294967295e+9
LWORD	REAL	16#0000_0000_0000_0000 ~ 16#FFFF_FFFF_FFFF_FFFF	0 ~ 1.844674e+19
	LREAL		0 ~ 1.84467440737095e+19

参考

- 从实数转换为位串时，请使用 “**_TO_***(实数→位串转换组)指令(P.2–275)”。
- 从任意的数据类型转换为实数时，请使用 “TO_**(实数转换组)指令(P.2–306)”。

使用注意事项

- 指令名称请务必符合“In”和“Out”的数据类型。
- 根据“In”和“Out”的数据类型的组合，有时会发生与实数的有效数位相应的取整，导致转换前后的值产生差异。会发生差异的数据类型的组合如下所示。

“In”的数据类型	“Out”的数据类型	发生差异的值
DWORD	REAL	16#0100_0000以上
LWORD	LREAL	16#0002_0000_0000_0000以上

_TO_*(实数→整数转换组)

将实数转换为整数。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	实数→整数转换组	FUN	(@)**_TO_*** EN ENO In Out **为实数的数据类型名称， ***为整数的数据类型名称	Out:=**_TO_*** (In); **为实数的数据类型名称，***为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

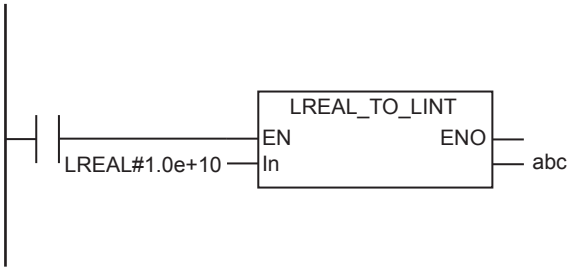
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out						○	○	○	○	○	○	○	○							

功能

将实数 “In”转换为整数。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为LREAL型，“Out”为LINT型时，指令名称为LREAL_TO_LINT。

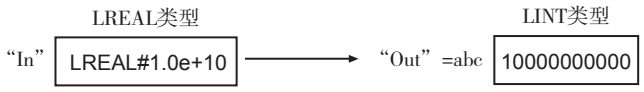
LREAL_TO_LINT指令下，“In” = LREAL#1.0e+10时的示例如下所示。

LD



ST

```
abc:=LREAL_TO_LINT(LREAL#1.0e+10);
```



“In”的值的小数点后以五舍五入取整。五舍五入是指如下表所示的处理。

小数部分的值	处理	示例
小于0.5	舍去	1.49→1 −1.49→−1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2 −1.50→−2 −2.50→−2
大于0.5	进位	1.51→2 −1.51→−2

如下所示，根据数据类型的组合，“In”和 “Out”的有效范围有所差异。

“In”的数据类型	“Out”的数据类型	“In”的有效范围	“Out”的有效范围
REAL	USINT	0 ~ 2.55e+2	0 ~ 255
	UINT	0 ~ 6.5535e+4	0 ~ 65535
	UDINT	0 ~ 4.294967e+9	0 ~ 4294967295
	ULINT	0 ~ 1.844674e+19	0 ~ 18446744073709551615
	SINT	−1.28e+2 ~ 1.27e+2	−128 ~ 127
	INT	−3.2768e+4 ~ 3.2767e+4	−32768 ~ 32767
	DINT	−2.147483e+9 ~ 2.147483e+9	−2147483648 ~ 2147483647
	LINT	−9.223372e+18 ~ 9.223372e+18	−9223372036854775808 ~ −9223372036854775807
LREAL	USINT	0 ~ 2.55e+2	0 ~ 255
	UINT	0 ~ 6.5535e+4	0 ~ 65535
	UDINT	0 ~ 4.294967295e+9	0 ~ 4294967295
	ULINT	0 ~ 1.84467440737095e+19	0 ~ 18446744073709551615
	SINT	−1.28e+2 ~ 1.27e+2	−128 ~ 127
	INT	−3.2768e+4 ~ 3.2767e+4	−32768 ~ 32767
	DINT	−2.147483648e+9 ~ 2.147483647e+9	−2147483648 ~ 2147483647
	LINT	−9.22337203685477e+18 ~ 9.22337203685477e+18	−9223372036854775808 ~ −9223372036854775807

参考

- 从整数转换为实数时，请使用 “**_TO_***(整数→实数转换组)指令(P.2−264)”。
- 从任意的数据类型转换为整数时，请使用 “TO_**(整数转换组)指令(P.2−302)”。
- 除本指令以外，实数→整数转换指令还有TRUNC(小数点后舍去)、Round(小数点后五舍五入)、RoundUp(小数点后进位)。上述指令下，输入为REAL型时输出为DINT型，输入为LREAL型时输出为LINT型。各指令在处理上的差异如下所示。

输入值	输出值			
	REAL_TO_INT	TRUNC	Round	RoundUp
REAL#1.6	INT#2	DINT#1	DINT#2	DINT#2
REAL#1.5	INT#2	DINT#1	DINT#2	DINT#2
REAL#1.4	INT#1	DINT#1	DINT#1	DINT#2
REAL#2.5	INT#2	DINT#2	DINT#2	DINT#3
REAL#−1.6	INT#−2	DINT#−1	DINT#−2	DINT#−2
REAL#−1.5	INT#−2	DINT#−1	DINT#−2	DINT#−2
REAL#−1.4	INT#−1	DINT#−1	DINT#−1	DINT#−2
REAL#−2.5	INT#−2	DINT#−2	DINT#−2	DINT#−3

使用注意事项

- 指令名称请务必符合 “In” 和 “Out” 的数据类型。
- 转换结果超过 “Out” 的有效范围时，“Out” 的值为错误值。请务必检查 “In” 的值是否在有效范围内，以免转换结果超过 “Out” 的有效范围。

_TO_*(实数→位串转换组)

将实数转换为位串。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	实数→位串转换组	FUN	(@)**_TO_*** EN ENO In Out **为实数的数据类型名称， ***为位串的数据类型名称	Out:=**_TO_*** (In); **为实数的数据类型名称，***为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0
Out	转换结果	输出	转换结果	遵从数据类型	–	–

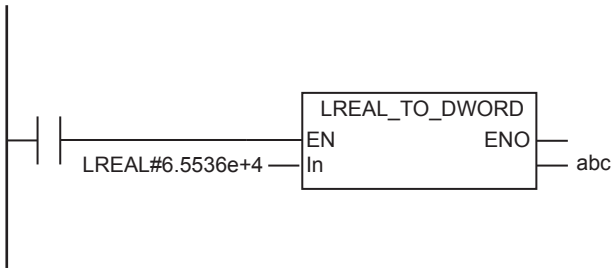
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out		○	○	○	○															

功能

将实数 “In” 转换为位串。
指令名称因 “In” 和转换结果 “Out” 的数据类型的组合而异。例如，“In” 为LREAL型，“Out” 为DWORD型时，指令名称为LREAL_TO_DWORD。

LREAL_TO_DWORD指令下，“In” = LREAL#6.5536e+4时的示例如下所示。

LD



ST

```
abc:=LREAL_TO_DWORD(LREAL#6.5536e+4);
```

转换的步骤如下所示。

1 **“In”的值的小数点后以五舍五入(下文阐述)取整。**

2 **整数部分作为无符号整数以位串表示。**

五舍五入是指如下表所示的处理。

小数部分的值	处理	示例
小于0.5	舍去	1.49→1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2
大于0.5	进位	1.51→2

转换示例如下。

“In”的值	取整后的值	“Out”的值
1.6	2	16#0002
3.5	4	16#0004

参考

从位串转换为实数时，请使用 “**_TO_***(位串→实数转换组)指令(P.2-270)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- 转换结果超过 “Out”的有效范围时，“Out”的值为错误值。请务必检查 “In”的值是否在有效范围内，以免转换结果超过 “Out”的有效范围。
- 输入负值时，转换结果因CPU单元的型号而异。输入负值时，请充分调试后再使用。

_TO_*(实数→实数转换组)

将实数转换为不同数据类型的实数。

指令	名称	FB/FUN	图形表现	ST表现
_TO_*	实数→实数转换组	FUN	(@)**_TO_*** EN ENO In Out **、***为不同实数的数据类型名称	Out:=**_TO_*** (In); **、***为不同实数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In” “Out”的数据类型的组合而异。详情请参阅功能说明。

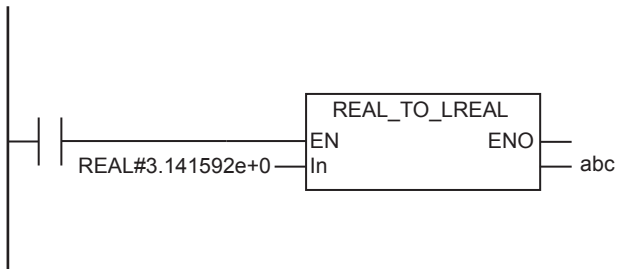
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out														○	○					

功能

将实数 “In”转换为不同数据类型的实数。
指令名称因 “In”和转换结果 “Out”的数据类型的组合而异。例如，“In”为REAL型，“Out”为LREAL型时，指令名称为REAL_TO_LREAL。

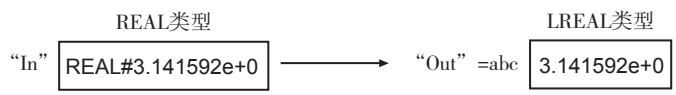
REAL_TO_LREAL指令下，“In” = REAL#3.141592e+0时的示例如下所示。

LD



ST

```
abc:=REAL_TO_LREAL(REAL#3.141592e+0);
```



“In”和 “Out”的有效范围如下所示。

“In”的数据类型	“Out”的数据类型	“In”“Out”的有效范围
REAL	LREAL	–3.402823e+38 ~ 3.402823e+38
LREAL	REAL	或+∞/–∞

参考

从任意的数据类型转换为实数时，请使用 “TO_**(实数转换组)指令(P.2–306)”。

使用注意事项

- 指令名称请务必符合 “In”和 “Out”的数据类型。
- “In”的值为+∞/–∞时，“Out”的值为+∞/–∞。
- “In”的值为非数时，“Out”的值为非数。
- 转换结果超过 “Out”的有效范围时，“Out”的值为与 “In”的值同符号的无穷大。
- LREAL_TO_REAL指令下，“In”的值比 ± 1.175494e–38更接近0时，“Out”的值为0。

**_TO_STRING(整数→字符串转换组)

将整数转换为字符串。

指令	名称	FB/FUN	图形表现	ST表现
**_TO_STRING	整数→字符串转换组	FUN	(@)**_TO_STRING EN ENO In Out **为整数的数据类型名称	Out:=**_TO_STRING(In); **为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In”的数据类型而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○							
Out																				○

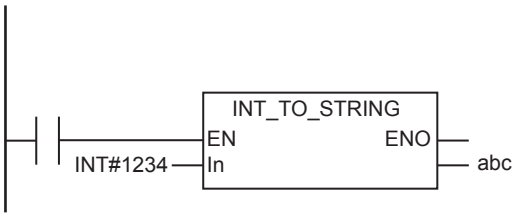
功能

将整数 “In”转换为字符串。“In”以数字格式表现，转换后为 “Out”，以字符串格式表现。在 “Out”的结尾加上NULL字符(16#00)。

转换时左对齐。“In”的值不到 “In”的数据类型的最大位数时，高位的0不输出为字符串。即进行消零。“In”为负数时，在字符串的开头加上减号'-'。
指令名称因 “In”的数据类型而异。例如，“In”为INT型时，指令名称为INT_TO_STRING。

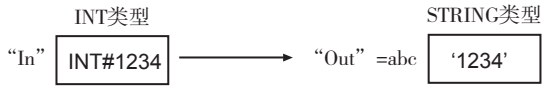
INT_TO_STRING指令下，“In”=INT#1234时的示例如下所示。

LD



ST

```
abc:=INT_TO_STRING(INT#1234);
```



如下所示，“Out”的有效范围因 “In”的数据类型而异。

“In”的数据类型	“Out”的有效范围(最大字节数)
USINT	4个字节(3个半角英数字字符+结尾NULL字符)
UINT	6个字节(5个半角英数字字符+结尾NULL字符)
UDINT	11个字节(10个半角英数字字符+结尾NULL字符)
ULINT	21个字节(20个半角英数字字符+结尾NULL字符)
SINT	5个字节(4个半角英数字字符+结尾NULL字符)
INT	7个字节(6个半角英数字字符+结尾NULL字符)
DINT	12个字节(11个半角英数字字符+结尾NULL字符)
LINT	21个字节(20个半角英数字字符+结尾NULL字符)

参考

从实数转换为整数时，请使用 “STRING_TO_**(字符串→整数转换组)指令(P.2–295)”。

使用注意事项

- 指令名称请务必符合 “In”的数据类型。

**_TO_STRING(位串→字符串转换组)

将位串转换为字符串。

指令	名称	FB/FUN	图形表现	ST表现
**_TO_STRING	位串→字符串转换组	FUN	(@)**_TO_STRING EN ENO In Out **为位串的数据类型名称	Out:=**_TO_STRING(In); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0
Out	转换结果	输出	转换结果	(*)	–	–

* 有效范围因 “In”的数据类型而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out																				○

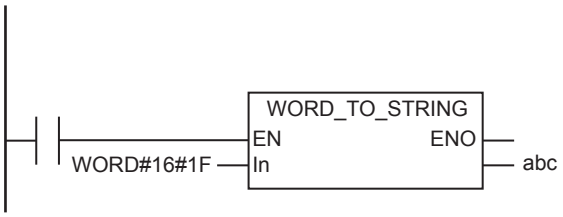
功能

将位串 “In”转换为字符串。“In”以16进制英数字格式表现，转换后为 “Out”，以字符串格式表现。“Out”中不包含代表16进制的前缀'#16'。在 “Out”的结尾加上NULL字符(16#00)。

转换时左对齐。“In”的值不到 “In”的数据类型的最大位数时，高位以'0'填充。即进行补零。因此，包含结尾的NULL字符在内的 “Out”的字节数与(“In”的字节数 × 2+1)保持一致。
指令名称因 “In”的数据类型而异。例如，“In”为WORD型时，指令名称为WORD_TO_STRING。

WORD_TO_STRING指令下，“In”=WORD#16#1F时的示例如下所示。

LD



ST

```
abc:=WORD_TO_STRING(WORD#16#1F);
```

如下所示，“Out”的有效范围因 “In”的数据类型而异。

“In”的数据类型	“Out”的有效范围(最大字节数)
BYTE	3个字节(2个半角英数字字符+结尾NULL字符)
WORD	5个字节(4个半角英数字字符+结尾NULL字符)
DWORD	9个字节(8个半角英数字字符+结尾NULL字符)
LWORD	17个字节(16个半角英数字字符+结尾NULL字符)

参考

要将 “In”转换为带符号字符串时，先通过 “**_TO_***(位串→整数转换组)指令(P.2-266)”转换为带符号整数，再使用 “**_TO_STRING(整数→字符串转换组)指令(P.2-279)”。

使用注意事项

- 指令名称请务必符合 “In”的数据类型。

**_TO_STRING(实数→字符串转换组)

将实数转换为字符串。

指令	名称	FB/FUN	图形表现	ST表现
**_TO_STRING	实数→字符串转换组	FUN	(@)**_TO_STRING EN ENO In Out **为实数的数据类型名称	Out:=**_TO_STRING(In); **为实数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0.0
Out	转换结果	输出	转换结果	(*)	–	–

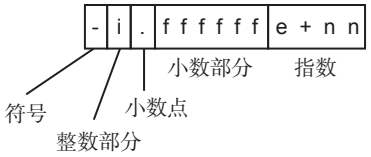
* 有效范围因 “In”的数据类型而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out																				○

功能

将实数 “In”转换为字符串。“In”以英数字的字符串表现，转换后为 “Out”。

“Out”的构成如下所示。



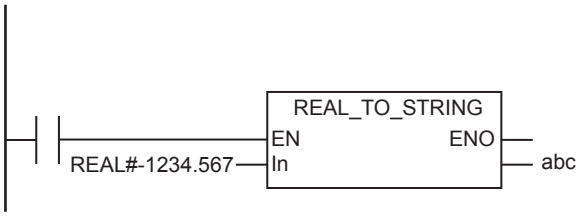
项目	内容
符号	仅 “In”为负数时，加上'–'(减号)。 “In”为正数时，不加'+'(加号)。
整数部分	始终以1个数位表示。
小数点	即使 “In”不是小数，也始终表示。
小数部分	“In”为REAL型时，以6个数位表示； “In”为LREAL型时，以14个数位表示。
指数	始终表示。nn的位数为2或3。 “In”的绝对值为1.0以上时，nn的符号为'+'(加号)；为1.0以下时，nn的符号为'–'(减号)。

在 “Out”的结尾加上NULL字符(16#00)。

指令名称因 “In”的数据类型而异。例如，“In”为REAL型时，指令名称为REAL_TO_STRING。

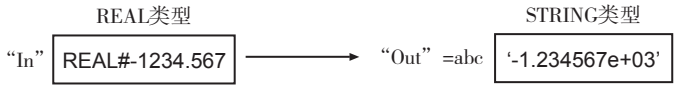
REAL_TO_STRING指令下，“In”=REAL#–1234.567时的示例如下所示。

LD



ST

```
abc:=REAL_TO_STRING(REAL#-1234.567);
```



“In”的值为0、无穷大、非数时，“Out”的值如下所示。

“In”的值	“Out”的值
0	'0'
+∞	'inf'
–∞	'–inf'
非数	'nan'或'–nan'

参考

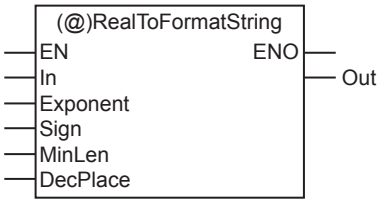
- 从字符串转换为实数时，请使用 “STRING_TO_**(字符串→实数转换组)指令(P.2–299)”。
- 指定格式，再从实数转换为字符串时，请使用 “RealToFormatString指令(P.2–285)”、“LrealToFormatString指令(P.2–290)”。

使用注意事项

- 指令名称请务必符合 “In”的数据类型。

RealToFormatString

以指定格式将REAL型变量转换为字符串。

指令	名称	FB/FUN	图形表现	ST表现
RealTo FormatString	实数(REAL)→带格式字符串转换	FUN		Out:=RealToFormatString(In, Exponent, Sign, MinLen, DecPlace);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0.0
Exponent	指数		TRUE ：有指数 FALSE：无指数			FALSE
Sign	符号		TRUE ：有符号 FALSE：无符号			
MinLen	最小位数		“Out”的最小位数			6
DecPlace	精度		“Out”的小数点后的位数	0 ~ 15		
Out	转换结果	输出	转换结果	最大327个字节(326个半角英数字字符+结尾NULL字符)	–	–

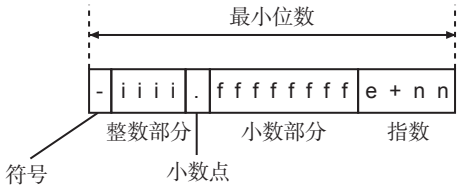
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○						
Exponent	○																			
Sign	○																			
MinLen						○														
DecPlace						○														
Out																				○

功能

将REAL型变量 “In”转换为字符串。“In”以英数字的字符串表现，转换后为 “Out”。在 “Out”的结尾加上NULL字符(16#00)。

“In”的值为负数时，开头加上'-'(减号)； “In”的值为正数时，开头不加'+'(加号)。

“Out”的格式因指数 “Exponent”、符号 “Sign”、最小位数 “MinLen”、精度 “DecPlace”而异。



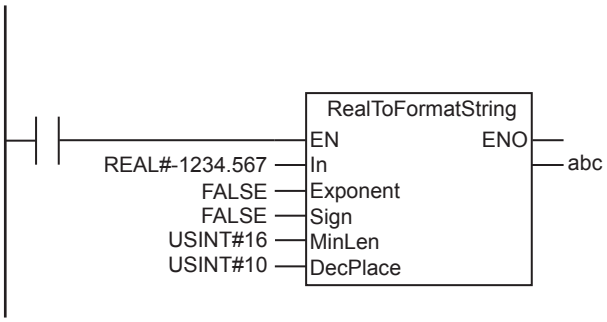
输入变量	含义
“Exponent”	指示有无指数 TRUE ：有指数 FALSE：无指数
“Sign”	指示有无符号 TRUE ：有符号 FALSE：无符号 符号仅填入'-'(减号)。正数且有符号时，符号填入' '(空格)。 负数且无符号时，整数部分的开头加上'-'(减号)。 转换结果的位数超过 “MinLen”的值且为正数时，符号填入最高位的数字。
“MinLen”	符号、整数部分、小数点、小数部分、指数合计的整体位数的最小值。 转换结果的位数不到 “MinLen”的值时，符号以外的字符串右对齐，空位添加' '(空格)。 转换结果的位数超过 “MinLen”的值时，字符串左对齐，超过 “MinLen”的值的位数的字符串代入 “Out”。
“DecPlace”	小数部分的位数。 超过 “DecPlace”的值的数位以五舍五入(下文阐述)取整。 “DecPlace”的值为0时，无小数点和小数部分。

“In”的值为REAL#-1234.567时，各输入变量的值与 “Out”的值之间的关系如下所示。

(例1) “Exponent”：FALSE
“Sign” ：FALSE
“MinLen” ：USINT#16
“DecPlace”：USINT#10

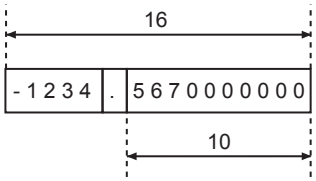
为无符号的负数，因此整数部分的开头为'–'(减号)。

LD



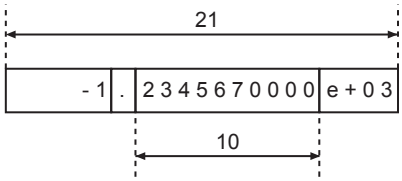
ST

```
abc:=RealToFormatString(REAL#-1234.567, FALSE,
                        FALSE, USINT#16,
                        USINT#10);
```



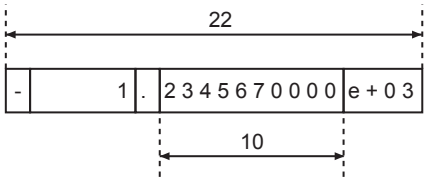
(例2) “Exponent”：TRUE
“Sign”：FALSE
“MinLen”：USINT#21
“DecPlace”：USINT#10

“MinLen”大于字符串的位数，因此字符串右对齐，开头以' '(空格)填充。



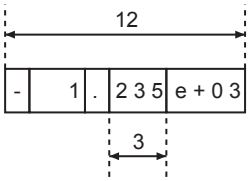
(例3) “Exponent”：TRUE
“Sign”：TRUE
“MinLen”：USINT#22
“DecPlace”：USINT#10

符号必定在左端。整数部分的开头以' '(空格)填充。



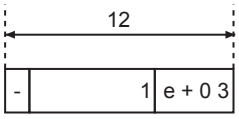
(例4) “Exponent”：TRUE
“Sign”：TRUE
“MinLen”：USINT#12
“DecPlace”：USINT#3

“DecPlace”=USINT#3，因此小数点后第4位五舍五入。



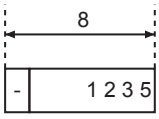
(例5) “Exponent”：TRUE
“Sign” ：TRUE
“MinLen” ：USINT#12
“DecPlace”：USINT#0

“DecPlace”=USINT#0，因此小数点后第1位五舍五入。无小数点。



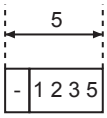
(例6) “Exponent”：FALSE
“Sign” ：TRUE
“MinLen” ：USINT#8
“DecPlace”：USINT#0

无指数，整数部分变为4个数位。同时，小数点后第1位五舍五入。



(例7) “Exponent”：FALSE
“Sign” ：TRUE
“MinLen” ：USINT#2
“DecPlace”：USINT#0

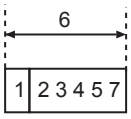
“In”的整数部分位数4大于 “MinLen”=USINT#2，因此整数部分以4个数位表示。



“In”的值为REAL#123456.7时，输入变量的值与 “Out”的值之间的关系如下所示。

(例8) “Exponent”：FALSE
“Sign” ：TRUE
“MinLen” ：USINT#4
“DecPlace”：USINT#0

“In”的整数部分位数6大于 “MinLen”=USINT#4，因此整数部分以6位表示。“In”的值为正数，因此符号填入最高位的数字。



“In”的值为无穷大、非数时，“Out”的值如下所示。

“In”的值	“Out”的值
+∞	'inf'
−∞	'−inf'
非数	'nan'或'−nan'

五舍五入是指如下表所示的处理。

小数部分的值	处理	示例
小于0.5	舍去	1.49→1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2
大于0.5	进位	1.51→2

参考

- “Exponent”、“Sign”、“MinLen”、“DecPlace”可分别省略。省略时使用各自的初始值。
- 从LREAL型变量转换为字符串时，请使用 “LrealToFormatString指令(P.2−290)”。
- 从字符串转换为实数时，请使用 “STRING_TO_**(字符串→实数转换组)指令(P.2−299)”。

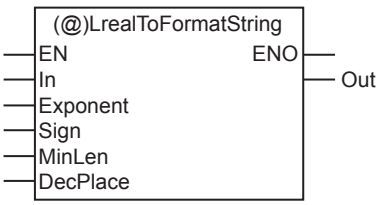
使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “DecPlace”的值超过有效范围时。
- “DecPlace”的值大于 “MinLen”的值时。

LrealToFormatString

以指定格式将LREAL型变量转换为字符串。

指令	名称	FB/FUN	图形表现	ST表现
LrealTo FormatString	实数(LREAL)→带格式字符串转换	FUN		Out:=LrealToFormatString(In, Exponent, Sign, MinLen, DecPlace);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0.0
Exponent	指数		TRUE：有指数 FALSE：无指数			FALSE
Sign	符号		TRUE：有符号 FALSE：无符号			
MinLen	最小位数		“Out”的最小位数			6
DecPlace	精度		“Out”的小数点后的位数	0 ~ 15		
Out	转换结果	输出	转换结果	最大327个字节(326个半角英数字字符+结尾NULL字符)	–	–

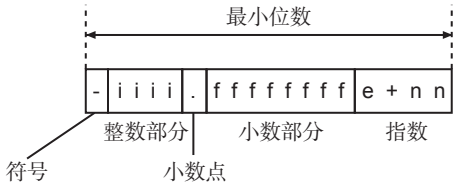
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In															○					
Exponent	○																			
Sign	○																			
MinLen						○														
DecPlace						○														
Out																				○

功能

将LREAL型变量 “In”转换为字符串。“In”以英数字的字符串表现，转换后为 “Out”。在 “Out”的结尾加上NULL字符(16#00)。

“In”的值为负数时，开头加上'–'(减号)； “In”的值为正数时，开头不加'+'(加号)。

“Out”的格式因指数 “Exponent”、符号 “Sign”、最小位数 “MinLen”、精度 “DecPlace”而异。



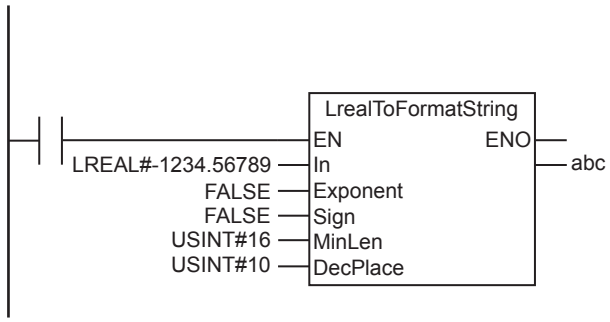
输入变量	含义
“Exponent”	指示有无指数 TRUE ：有指数 FALSE：无指数
“Sign”	指示有无符号 TRUE ：有符号 FALSE：无符号 符号仅填入'–'(减号)。正数且有符号时，符号填入' '(空格)。 负数且无符号时，整数部分的开头加上'–'(减号)。 转换结果的位数超过 “MinLen”的值且为正数时，符号填入最高位的数字。
“MinLen”	符号、整数部分、小数点、小数部分、指数合计的整体位数的最小值。 转换结果的位数不到 “MinLen”的值时，符号以外的字符串右对齐，空位添加' '(空格)。 转换结果的位数超过 “MinLen”的值时，字符串左对齐，超过 “MinLen”的值的位数的字符串代入 “Out”。
“DecPlace”	小数部分的位数。 超过 “DecPlace”的值的数位以五舍五入(下文阐述)取整。 “DecPlace”的值为0时，无小数点和小数部分。

“In”的值为LREAL#–1234.56789时，各输入变量的值与 “Out”的值之间的关系如下所示。

(例1) “Exponent” ：FALSE
“Sign” ：FALSE
“MinLen” ：USINT#16
“DecPlace” ：USINT#10

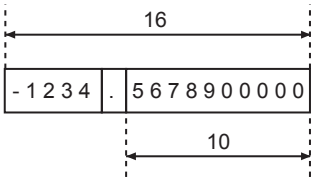
为无符号的负数，因此整数部分的开头为'-'(减号)。

LD



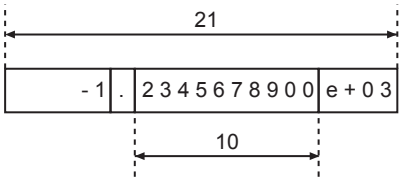
ST

```
abc:=LrealToFormatString(LREAL#-1234.56789, FALSE,
                         FALSE, USINT#16,
                         USINT#10);
```



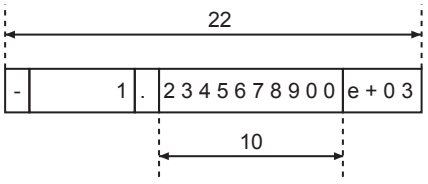
(例2) “Exponent” ：TRUE
“Sign” ：FALSE
“MinLen” ：USINT#21
“DecPlace” ：USINT#10

“MinLen”大于字符串的位数，因此字符串右对齐，开头以' '(空格)填充。



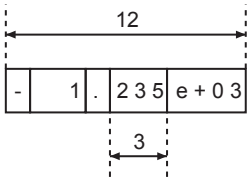
(例3) “Exponent” ：TRUE
“Sign” ：TRUE
“MinLen” ：USINT#22
“DecPlace” ：USINT#10

符号必定在左端。整数部分的开头以' '(空格)填充。



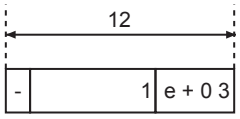
(例4) “Exponent” ：TRUE
“Sign” ：TRUE
“MinLen” ：USINT#12
“DecPlace” ：USINT#3

“DecPlace”=USINT#3，因此小数点后第4位五舍五入。



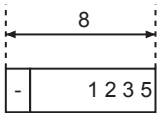
(例5) “Exponent” ：TRUE
“Sign” ：TRUE
“MinLen” ：USINT#12
“DecPlace” ：USINT#0

“DecPlace”=USINT#0，因此小数点后第1位五舍五入。无小数点。



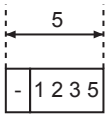
(例6) “Exponent” ：FALSE
“Sign” ：TRUE
“MinLen” ：USINT#8
“DecPlace” ：USINT#0

无指数，整数部分变为4个数位。同时，小数点后第1位五舍五入。



(例7) “Exponent” ：FALSE
“Sign” ：TRUE
“MinLen” ：USINT#2
“DecPlace” ：USINT#0

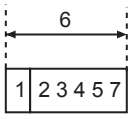
“In”的整数部分位数4大于 “MinLen”=USINT#2，因此整数部分以4个数位表示。



“In”的值为LREAL#123456.789时，输入变量的值与 “Out”的值之间的关系如下所示。

(例8) “Exponent” ：FALSE
“Sign” ：TRUE
“MinLen” ：USINT#4
“DecPlace” ：USINT#0

“In”的整数部分位数6大于 “MinLen”=USINT#4，因此整数部分以6个数位表示。“In”的值为正数，因此符号填入最高位的数字。



“In”的值为无穷大、非数时，“Out”的值如下所示。

“In”的值	“Out”的值
+∞	'inf'
−∞	'−inf'
非数	'nan'或'−nan'

五舍五入是指如下表所示的处理。

小数部分的值	处理	示例
小于0.5	舍去	1.49→1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2
大于0.5	进位	1.51→2

参考

- “Exponent”、“Sign”、“MinLen”、“DecPlace”可分别省略。省略时使用各自的初始值。
- 从REAL型变量转换为字符串时，请使用 “RealToFormatString指令(P.2−285)”。
- 从字符串转换为实数时，请使用 “STRING_TO_**(字符串→实数转换组)指令(P.2−299)”。

使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “DecPlace”的值超过有效范围时。
- “DecPlace”的值大于 “MinLen”的值时。

STRING_TO_**(字符串→整数转换组)

将字符串转换为整数。

指令	名称	FB/FUN	图形表现	ST表现
STRING_TO_**	字符串→整数转换组	FUN	(@)STRING_TO_** EN ENO In Out **为整数的数据类型名称	Out:=STRING_TO_** (In); **为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	''
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 有效范围因 “Out”的数据类型而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out						○	○	○	○	○	○	○	○							

功能

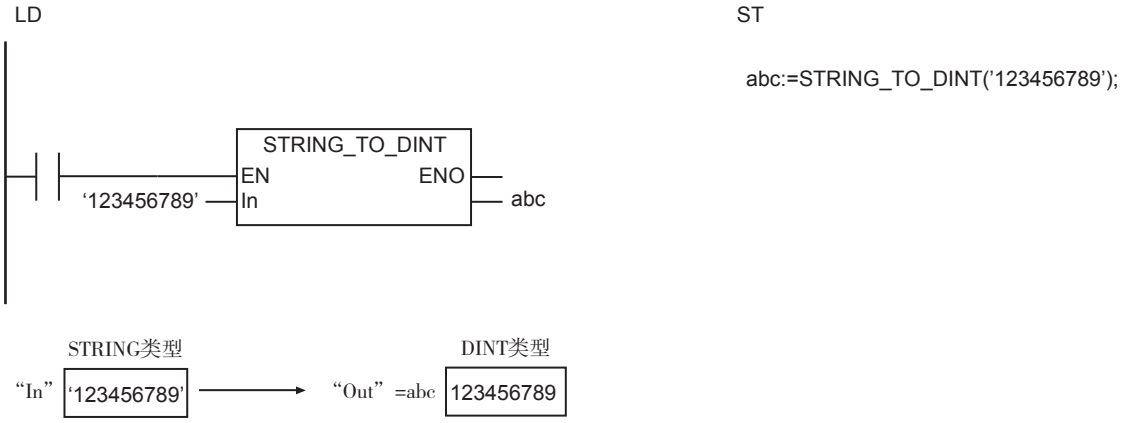
将字符串 “In”转换为整数型。

“In”的字符串仅可由'0' ~ '9'的数字构成。以下情况例外。

- 开头仅带一个'–'(减号)或'+'(加号)时，将其作为符号处理。
- 开头带几个' '(空格)时，将其忽略。
- 开头的'–'(减号)或'+'(加号)与数字之间带几个' '(空格)时，忽略' '(空格)。
- 任意位置含有单独的'_'(下划线)时，将其忽略。
- 任意位置上有2个以上连续'_'(下划线)时，会出现异常。
- 开头和末尾有'_'(下划线)时，会出现异常。
- 开头的'–'(减号)或'+'(加号)与数字之间有'_'(下划线)时，会出现异常。

指令名称因转换结果“Out”的数据类型而异。例如，“Out”为DINT型时，指令名称为STRING_TO_DINT。

STRING_TO_DINT指令下，“In”='123456789'时的示例如下所示。



如下所示，“In”的有效范围因 “Out”的数据类型而异。

“Out”的数据类型	“In”的有效范围(最大字节数)(*)
USINT	4个字节(3个半角英数字字符+结尾NULL字符)
UINT	6个字节(5个半角英数字字符+结尾NULL字符)
UDINT	11个字节(10个半角英数字字符+结尾NULL字符)
ULINT	21个字节(20个半角英数字字符+结尾NULL字符)
SINT	5个字节(4个半角英数字字符+结尾NULL字符)
INT	7个字节(6个半角英数字字符+结尾NULL字符)
DINT	12个字节(11个半角英数字字符+结尾NULL字符)
LINT	21个字节(20个半角英数字字符+结尾NULL字符)

* 字节数中不包含字符串开头的多个' '(空格)、字符串开头的多个'0'、字符串中单独的'_'(下划线)。

参考

- 要将字符串转换为16进制时，请使用 “STRING_TO_**(字符串→位串转换组)指令(P.2–297)”。
- 从整数型转换为字符串型时，请使用 “**_TO_STRING(整数→字符串转换组)指令(P.2–279)”。

使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- “In”的值为'–0'时，“Out”的值为0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值不是表示数值的字符串时。
  - 转换结果超过 “Out”的数据类型的有效范围时。

STRING_TO_**(字符串→位串转换组)

将字符串转换为位串。

指令	名称	FB/FUN	图形表现	ST表现
STRING_TO_**	字符串→位串转换组	FUN	(@)STRING_TO_** EN ENO In Out **为位串的数据类型名称	Out:=STRING_TO_** (In); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*)	–	''
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 有效范围因 “Out”的数据类型而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out		○	○	○	○															

功能

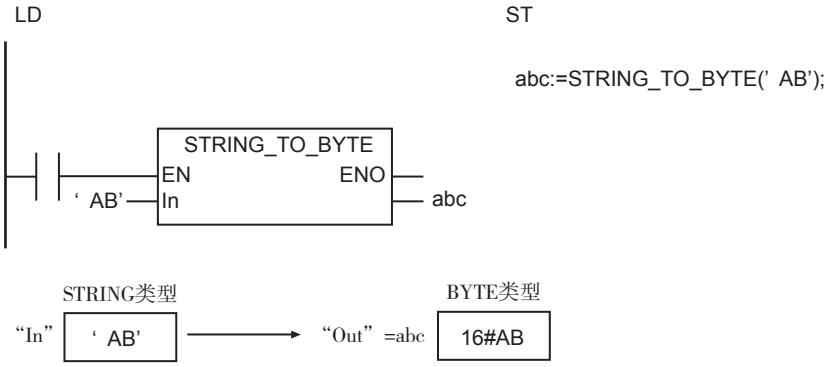
将字符串 “In”视为16进制，将其转换为位串。

“In”的字符串仅可由'0' ~ '9'的数字、'a' ~ 'f'或'A' ~ 'F'构成。以下情况例外。

- 开头带连续' '(空格)或'0'时，将其忽略。
- 任意位置含有单独的'_'(下划线)时，将其忽略。
- 任意位置上有2个以上连续'_'(下划线)时，会出现异常。
- 开头和末尾有'_'(下划线)时，会出现异常。
- 开头的'-'(减号)或'+'(加号)与数字之间有'_'(下划线)时，会出现异常。

指令名称因转换结果“Out”的数据类型而异。例如，“Out”为BYTE型时，指令名称为STRING_TO_BYTE。

STRING_TO_BYTE指令下，“In”=' AB'时的示例如下所示。忽略开头的' '(空格)。



如下所示，“In”的有效范围因 “Out”的数据类型而异。

“Out”的数据类型	“In”的有效范围(最大字节数)(*)
BYTE	3个字节(2个半角英数字字符+结尾NULL字符)
WORD	5个字节(4个半角英数字字符+结尾NULL字符)
DWORD	9个字节(8个半角英数字字符+结尾NULL字符)
LWORD	17个字节(16个半角英数字字符+结尾NULL字符)

* 字节数中不包含字符串开头的多个' '(空格)、字符串开头的多个'0'、字符串中单独的'_'(下划线)。

参考

- 处理表示带符号数值的字符串时，请使用 “STRING_TO_**(字符串→整数转换组)指令(P.2–295)”。
- 从位串转换为字符串时，请使用 “**_TO_STRING(位串→字符串转换组)指令(P.2–281)”。

使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值不是表示16进制数值的字符串时。
  - 转换结果超过 “Out”的数据类型的有效范围时。

STRING_TO_**(字符串→实数转换组)

将字符串转换为实数。

指令	名称	FB/FUN	图形表现	ST表现
STRING_TO_**	字符串→实数转换组	FUN	(@)STRING_TO_** EN ENO In Out **为实数的数据类型名称	Out:=STRING_TO_** (In); **为实数的数据类型名称

变量

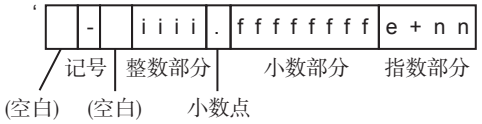
	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	最大311个字节(310个半角英数字字符+结尾NULL字符)	–	''
Out	转换结果	输出	转换结果	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out														○	○					

功能

将字符串 “In”转换为实数。
指令名称因转换结果 “Out”的数据类型而异。例如，“Out”为LREAL型时，指令名称为STRING_TO_LREAL。

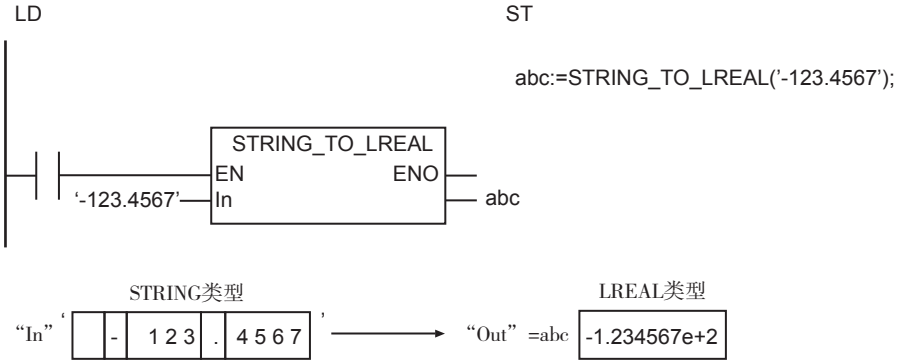
“In”的字符串格式如下所示。



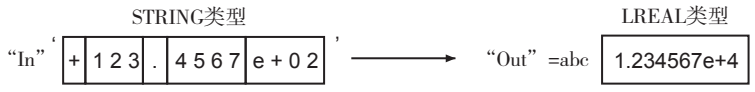
名称	格式
记号	• 以忽略字符串开头出现的连续' '(空格)后出现的单个 '+' 或 '–'为 符号。 • 可省略'+'符号。 • 同时忽略符号后的连续空格。

名称	格式
整数部分	• 符号后到小数点前为整数部分。整数部分不包含符号后连续的空格。有时也省略符号。 • 无小数点和小数部分时，指数部分前为整数部分。 • 无小数点、小数部分和指数部分时，到字符串结尾为止为整数部分。 • 整数部分由'0' ~ '9'的数字构成。 • 不可省略整数部分。 • 整数部分的最大位数为最大字符串长度1985减去符号、小数点、小数部分、指数部分及符号前后的空格的总字节数后的位数。
小数点	• 整数部分后的单个'.'(句号)为小数点。 • 无小数部分时，请省略小数点。
小数部分	• 小数点后到指数部分前为小数部分。 • 无指数部分时，到字符串结尾为止为小数部分。 • 小数部分由'0' ~ '9'的数字构成。 • 可省略小数部分。 • 小数部分的最大位数为15位。 • 无小数点时，也无小数部分。
指数部分	• 小数部分后的单个'e'或'E'、单个'+'或'−'到字符串结尾为指数部分。 • 无小数部分时，小数点后的字符串为指数部分。 • 无小数点和小数部分时，整数后的上述字符串为指数部分。 • 指数部分的数字部分由'0' ~ '9'的数字构成。 • 可省略指数部分。 • 指数部分的数字部分的最大位数为3。

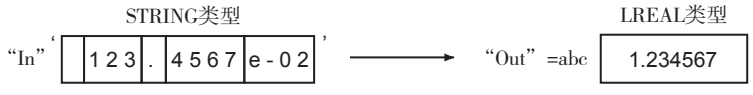
(例1) 有符号、小数点、小数部分，无指数部分



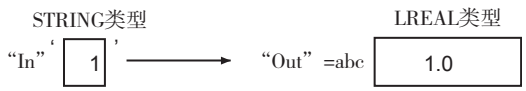
(例2) 有符号、小数点、小数部分、指数部分



(例3) 无符号，有小数点、小数部分、指数部分



(例4) 无符号、小数点、小数部分、指数部分



“In”的值为'+inf'时，“Out”的值为+∞。“In”的值为'–inf'时，“Out”的值为–∞。无论何种情况均无大小写之分。

参考

从实数转换为字符串时，请使用 “**_TO_STRING(实数→字符串转换组)指令(P.2–283)”。

使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- “In”的任意位置含有单独的'_'(下划线)时，将其忽略。
- “In”的开头和结尾有'_'(下划线)时，会出现异常。
- “In”的任意位置上有2个以上连续的'_'(下划线)时，会出现异常。
- “In”开头的'–'(减号)或'+'(加号)与数字之间有'_'(下划线)时，会出现异常。
- “In”的内容超过 “Out”的数据类型可表现的精度时，进行取整。
- “In”的内容比 “Out”的数据类型可表现的值更接近0时，“Out”的值为0。
- “In”的内容超过 “Out”的有效范围时，若为正数，则 “Out”的值为+∞；若为负数，“Out”的值为–∞。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值不是表示数值的字符串时。
  - “In”的字符串有小数点，但无小数部分时。

TO_**(整数转换组)

将整数、位串、实数、字符串转换为整数。

指令	名称	FB/FUN	图形表现	ST表现
TO_**	整数转换组	FUN	(@)TO_** EN ENO In Out **为整数的数据类型名称	Out:=TO_** (In); **为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*1)	–	(*2)
Out	转换结果	输出	转换结果	(*1)	–	–

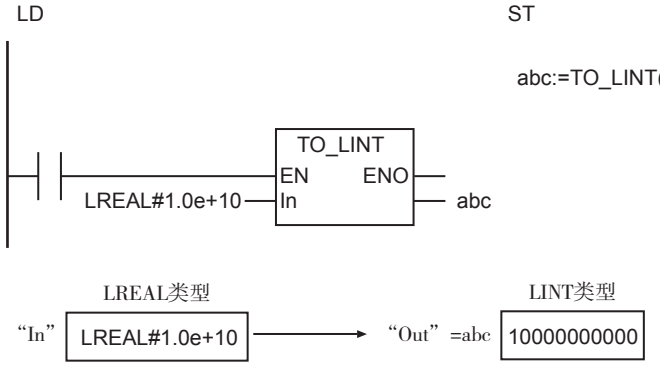
*1 有效范围因 “In”“Out”的数据类型的组合而异。
*2 省略输入参数时，初始值不适用。编译时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○	○	○	○	○	○	○	○	○	○	○					○
Out						○	○	○	○	○	○	○	○							

功能

将整数、位串、实数、字符串中任一 “In”转换为整数。
指令名称因转换结果 “Out”的数据类型而异。例如，“Out”为LINT型时，指令名称为TO_LINT。

TO_LINT指令下，“In”=LREAL#1.0e+10时的示例如下所示。



- 在 “In”的数据类型的有效位数范围内执行转换。“In”为浮点型时，小数点后以五舍五入取整。五舍五入是指如下表所示的处理。

小数部分的值	处理	示例
小于0.5	舍去	1.49→1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2
大于0.5	进位	1.51→2

“In”和“Out”的有效范围因数据类型的组合而异。关于各自的有效范围，请参阅 “**_TO_***(整数→整数转换组)指令(P.2–258)”、“**_TO_***(位串→整数转换组)指令(P.2–266)”、“**_TO_***(实数→整数转换组)指令(P.2–272)”的功能说明。

关于“In”为STRING型时的详细规格，请参阅 “STRING_TO_**(字符串→整数转换组)指令(P.2–295)”的功能说明。

使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- “In”的数据类型为位串，“In”和 “Out”的数据类型的大小不同时，进行如下处理。
  - “Out”的数据大小大于 “In”的数据大小时，在 “Out”的高位添加0。
  - “Out”的数据大小小于 “In”的数据大小时，舍去高位。
- “In”为STRING型时，请注意以下内容。
  - “In”的开头仅存在一个'–'(减号)或'+'(加号)时，作为符号处理。
  - 除 “In”开头的'–'(减号)或'+'(加号)以外，仅可由连续的'0' ~ '9'的数字构成。但字符串中可存在'_'(下划线)，'–'和'+'前后可存在' '(空格)。
- 转换结果超过 “Out”的有效范围时，“Out”的值为错误值。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”为STRING型，该值不是表示数值的字符串时。

TO_**(位串转换组)

将整数、位串、实数、字符串转换为位串。

指令	名称	FB/FUN	图形表现	ST表现
TO_**	位串转换组	FUN	(@)TO_** EN ENO In Out **为位串的数据类型名称	Out:=TO_** (In); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*1)	–	(*2)
Out	转换结果	输出	转换结果	(*1)	–	–

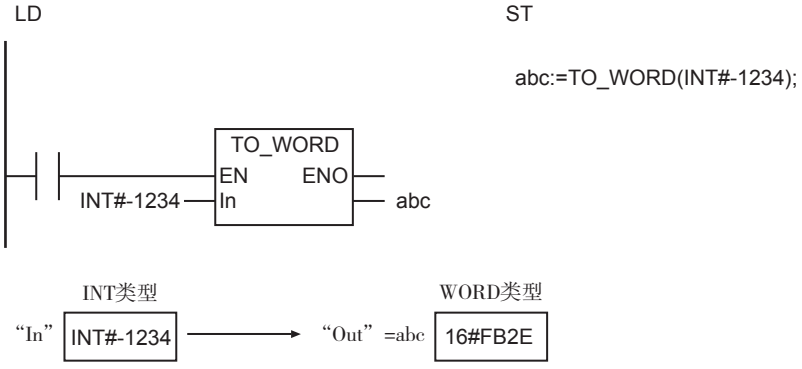
*1 有效范围因 “In” “Out”的数据类型的组合而异。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○	○	○	○	○	○	○	○	○	○	○					○
Out		○	○	○	○															

功能

将整数、实数、位串、字符串中任一 “In”转换为位串。
指令名称因转换结果 “Out”的数据类型而异。例如，“Out”为WORD型时，指令名称为TO_WORD。

TO_WORD指令下，“In”=INT#–1234时的示例如下所示。



“In”和“Out”的有效范围因数据类型的组合而异。关于各自的有效范围，请参阅 “**_TO_***(整数→位串转换组)指令(P.2–261)”、“**_TO_***(位串→位串转换组)指令(P.2–268)”、“**_TO_***(实数→位串转换组)指令(P.2–275)”的功能说明。

关于“In”为STRING型时的详细规格，请参阅 “STRING_TO_**(字符串→位串转换组)指令(P.2–297)”的功能说明。

使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- 转换结果超过 “Out”的有效范围时，“Out”的值为错误值。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”为STRING型，该值不是表示数值的字符串时。

TO_**(实数转换组)

将整数、位串、实数、字符串转换为实数。

指令	名称	FB/FUN	图形表现	ST表现
TO_**	实数转换组	FUN	(@)TO_** EN ENO In Out **为实数的数据类型名称	Out:=TO_** (In); **为实数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	(*1)(*2)	–	(*3)
Out	转换结果	输出	转换结果	(*1)	–	–

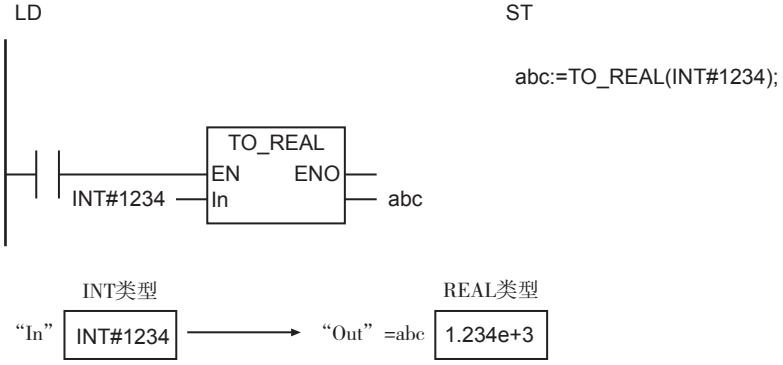
*1 有效范围因 “In” “Out”的数据类型的组合而异。
*2 STRING时，最大311个字节(310个半角英数字字符+结尾NULL文字)。
*3 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○	○	○	○	○	○	○	○	○	○	○					○
Out														○	○					

功能

将整数、位串、实数、字符串中任一 “In”转换为实数型。
指令名称因转换结果 “Out”的数据类型而异。例如，“Out”为REAL型时，指令名称为TO_REAL。
“In”的值为+∞/–∞时，“Out”的值为+∞/–∞。

TO_REAL指令下，“In”=INT#1234时的示例如下所示。



“In”和“Out”的有效范围因数据类型的组合而异。关于各自的有效范围，请参阅 “**_TO_***(整数→实数转换组)指令(P.2–264)”、“**_TO_***(位串→实数转换组)指令(P.2–270)”、“**_TO_***(实数→实数转换组)指令(P.2–277)”的功能说明。

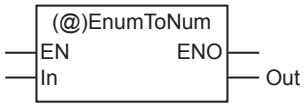
关于“In”为STRING型时的详细规格，请参阅 “STRING_TO_**(字符串→实数转换组)指令(P.2–299)”的功能说明。

使用注意事项

- 指令名称请务必符合 “Out”的数据类型。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”为STRING型，该值不是表示数值的字符串时。

EnumToNum

将枚举体转换为DINT型。

指令	名称	FB/FUN	图形表现	ST表现
EnumToNum	枚举体→整数转换	FUN		Out:=EnumToNum(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	–	–	0
Out	转换结果	输出	转换结果	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	枚举体																			
Out												○								

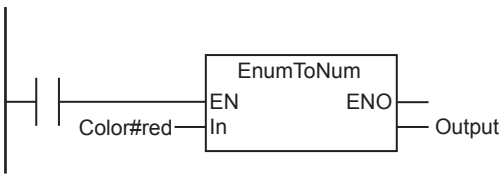
功能

将枚举体的值的转换对象 “In”转换为DINT型的值，转换后输出为 “Out”。

要通过显示器等无法处理枚举体变量的设备来监控枚举体变量的值等场合下，使用本指令。

将枚举体Color的枚举元素red转换为DINT型变量Output时的描述示例如下所示。red的枚举值为0时，Output为DINT#0。

LD



ST

```
Output:=EnumToNum(Color#red);
```

使用注意事项

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

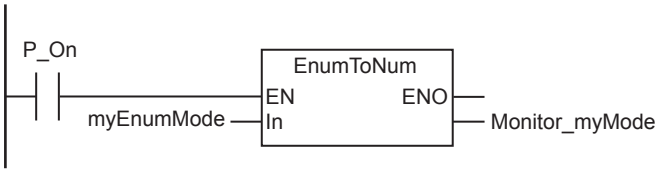
用户程序的动作模式通过枚举体 EnumMode 定义。将枚举体 EnumMode 的变量 myEnumMode 的值转换为 DINT 型的变量 Monitor_myMode，以通过显示器监控动作模式。例如，myEnumMode 的值为 mode2 时，Monitor_myMode的值为2。

● 数据类型的定义

名称	枚举值	注释
EnumMode	–	枚举体
mode0	0	结构要素
mode1	1	结构要素
mode2	2	结构要素

LD

名称	数据类型	初始值	注释
myEnumMode	EnumMode	mode0	枚举体的模式的值
Monitor_myMode	DINT	0	模式的监控值



ST

名称	数据类型	初始值	注释
myEnumMode	EnumMode	mode0	枚举体的模式的值
Monitor_myMode	DINT	0	模式的监控值

```
Monitor_myMode:=EnumToNum(myEnumMode);
```

NumToEnum

将DINT型转换为枚举体。

指令	名称	FB/FUN	图形表现	ST表现
NumToEnum	整数→枚举体转换	FUN	(@)NumToEnum EN In InOut Out	NumToEnum(In,InOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0
InOut	转换结果	输入输出	转换结果	–	–	–
Out	返回值	输出	TRUE：执行指令且无异常 FALSE：不执行指令或有异常	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In												○								
InOut	枚举体																			
Out	○																			

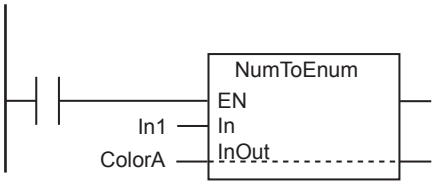
功能

将DINT型值的转换对象 “In”转换为枚举体的值，转换后输出为 “InOut”。

要通过显示器等无法处理枚举体变量的设备来变更枚举体变量的值等场合下，使用本指令。

将DINT型的变量In1转换为枚举体Color的变量ColorA时的描述示例如下所示。In1的值为1，与Color的枚举值对应的枚举元素为green时，ColorA的值为green。

LD



ST

```
NumToEnum(In1,ColorA);
```

参考

通过梯形图程序使用本指令时，可通过 “Out”确认 “In”的值是否在 “InOut”的值的范围内。

使用注意事项

“In”的值超过 “InOut”的值的范围时，会出现异常。“Out”为FALSE，“InOut”的值不变。

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

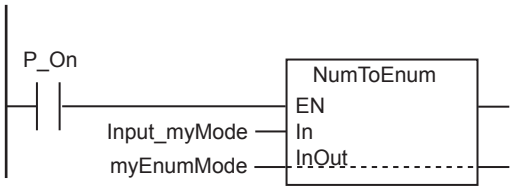
用户程序的动作模式通过枚举体EnumMode定义。写入DINT型的值Input_myMode，以通过显示器变更动作模式。通过用户程序将Input_myMode转换为枚举体EnumMode的变量myEnumMode。例如，Input_myMode的值为1时，myEnumMode的值为mode1。

● 数据类型的定义

名称	枚举值	注释
EnumMode	–	枚举体
mode0	0	结构要素
mode1	1	结构要素
mode2	2	结构要素

LD

名称	数据类型	初始值	注释
myEnumMode	EnumMode	mode0	枚举体的模式的值
Input_myMode	DINT	0	要变更的模式的值



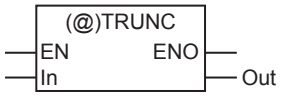
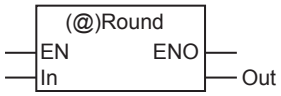
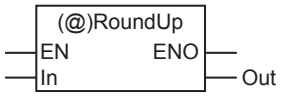
ST

名称	数据类型	初始值	注释
myEnumMode	EnumMode	mode0	枚举体的模式的值
Input_myMode	DINT	0	要变更的模式的值

```
NumToEnum(Input_myMode,myEnumMode);
```

TRUNC/Round/RoundUp

将实数处理为整数。
TRUNC ：小数点后第1位舍去。
Round ：小数点后第1位五舍五入。
RoundUp：小数点后第1位进位。

指令	名称	FB/FUN	图形表现	ST表现
TRUNC	实数舍去	FUN		Out:=TRUNC(In);
Round	实数取整	FUN		Out:=Round(In);
RoundUp	实数进位	FUN		Out:=RoundUp(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	(*)
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out												○	○							

功能

对实数 “In”的小数点后第1位进行处理，使其成为整数。

- TRUNC

  小数点后第1位舍去。

● Round

小数点后第1位五舍五入，使其成为整数。五舍五入是指如下表所示的处理。

小数部分的值	处理	示例
小于0.5	舍去	1.49→1 –1.49→–1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2 –1.50→–2 –2.50→–2
大于0.5	进位	1.51→2 –1.51→–2

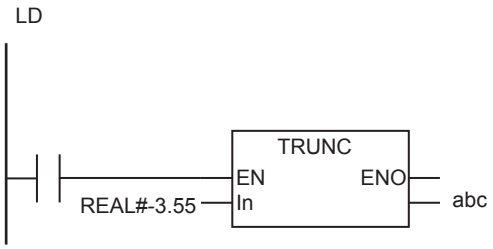
● RoundUp

小数点后第1位进位。

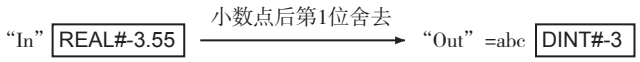
3种指令的动作区别如下所示。

输入值	输出值		
	TRUNC	Round	RoundUp
REAL#1.6	DINT#1	DINT#2	DINT#2
REAL#1.5	DINT#1	DINT#2	DINT#2
REAL#1.4	DINT#1	DINT#1	DINT#2
REAL#2.5	DINT#2	DINT#2	DINT#3
REAL#–1.6	DINT#–1	DINT#–2	DINT#–2
REAL#–1.5	DINT#–1	DINT#–2	DINT#–2
REAL#–1.4	DINT#–1	DINT#–1	DINT#–2
REAL#–2.5	DINT#–2	DINT#–2	DINT#–3

TRUNC指令下，“In”= REAL#–3.55时的示例如下所示。变量abc的值为DINT#–3。



“In”的小数点后第1位舍去。
“In”的值为REAL#–3.55，因此abc的值为DINT#–3。



参考

“In”的数据类型为REAL时，“Out”的数据类型为DINT。“In”的数据类型为LREAL时，“Out”的数据类型为LINT。

使用注意事项

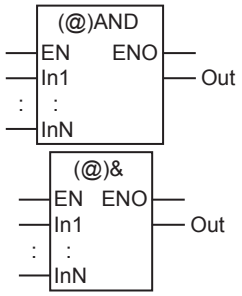
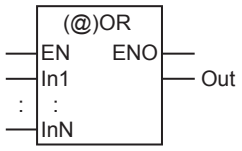
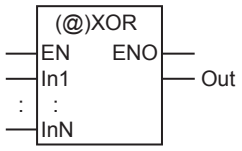
转换结果超过 “Out”的有效范围时，“Out”的值为错误值。请务必检查 “In”的值是否在有效范围内，以免转换结果超过 “Out”的有效范围。

位串运算指令

指令	名称	页码
AND，&/OR/XOR	逻辑积/逻辑和/异或	2–316
XORN	同或	2–319
NOT	位取反	2–321
AryAnd/AryOr/ AryXor/AryXorN	数组逻辑积/数组逻辑和/ 数组异或/数组同或	2–322

AND，&/OR/XOR

进行多个布尔、位串的每1位的运算。
AND，& ：逻辑积
OR ：逻辑和
XOR ：异或

指令	名称	FB/FUN	图形表现	ST表现
AND，&	逻辑积	FUN		Out:=In1 AND … AND InN; Out:=In1 & … & InN;
OR	逻辑和	FUN		Out:=In1 OR … OR InN;
XOR	异或	FUN		Out:=In1 XOR … XOR InN;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	运算对象	输入	运算对象 N为2 ~ 5	遵从数据类型	–	0(*)
Out	运算结果	输出	运算结果	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN	○	○	○	○	○															
Out	与 “In1” ~ “In2” 相同的数据类型																			

功能

对多个布尔或位串的运算对象 “In1” ~ “InN”，进行对应的每一位的运算。
因此，“In1” ~ “InN”和 “Out”的数据类型必须相同。

运算对象超过3个时，处理的步骤如下所示。

1 通过 “In1”和 “In2”进行运算

2 通过1的结果和 “In3”进行运算

3 通过2的结果和 “In4”进行运算

：　　：

运算的输入输出关系如下所示。

- AND，&

两者均为TRUE时，运算结果为TRUE。除此以外，运算结果为FALSE。

“In1”的位	“In2”的位	“Out”的位
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

- OR

两者均为FALSE时，运算结果为FALSE。除此以外，运算结果为TRUE。

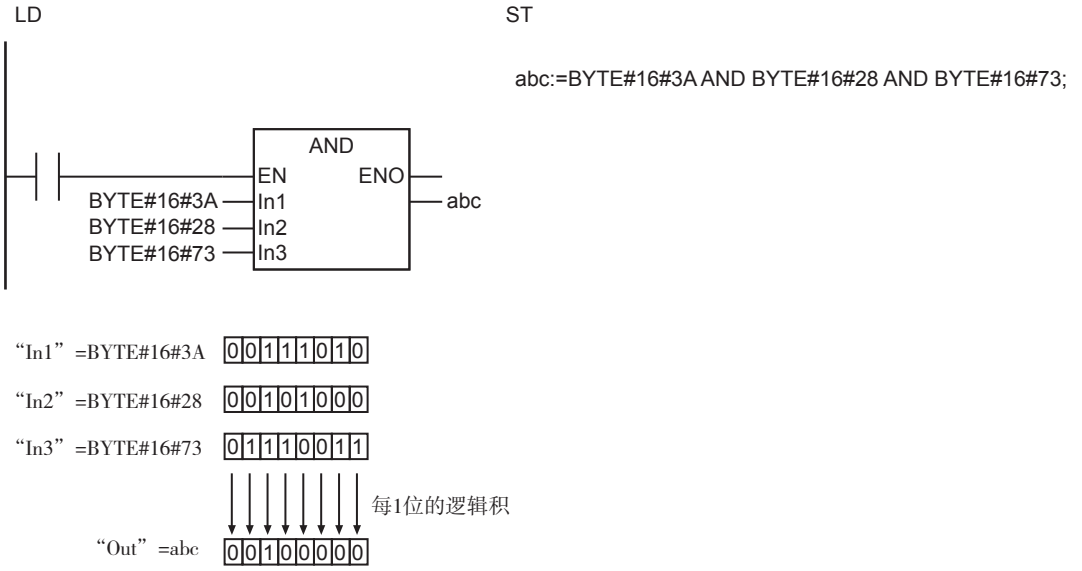
“In1”的位	“In2”的位	“Out”的位
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

- XOR

两者的值相同时，运算结果为FALSE。不同时，运算结果为TRUE。

“In1”的位	“In2”的位	“Out”的位
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

AND指令下，“In1”=BYTE#16#3A、“In2”=BYTE#16#28、“In3”=BYTE#16#73时的示例如下所示。



AND指令和&指令的功能完全相同。请采用易于使用的方法。

参考

使用ST语言进行如下描述时，输入变量数无限制。

Out:=In1 AND In2 AND In3 AND In4 AND In5 AND In6 …

Out:=In1 & In2 & In3 & In4 & In5 & In6 …

Out:=In1 OR In2 OR In3 OR In4 OR In5 OR In6 …

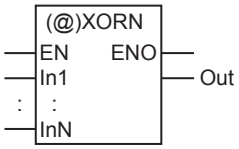
Out:=In1 XOR In2 XOR In3 XOR In4 XOR In5 XOR In6 ··

使用注意事项

请将 “In1” ~ “InN” 和 “Out” 的数据类型全部设为相同。否则，编连时会发生异常。

XORN

对多个布尔、位串的每1位进行同或运算。

指令	名称	FB/FUN	图形表现	ST表现
XORN	同或	FUN		Out:=In1 XOR NOT … XOR NOT InN;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	运算对象	输入	运算对象　N为2 ~ 5	遵从数据类型	–	0(*)
Out	运算结果	输出	运算结果	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN	○	○	○	○	○															
Out	与 “In1” ~ “In2” 相同的数据类型																			

功能

对多个布尔或位串的运算对象 “In1” ~ “InN”，进行对应的每一位的运算。
因此，“In1” ~ “InN”和 “Out”的数据类型必须相同。

运算对象超过3个时，处理的步骤如下所示。

1 通过 “In1”和 “In2”进行运算

2 通过1的结果和 “In3”进行运算

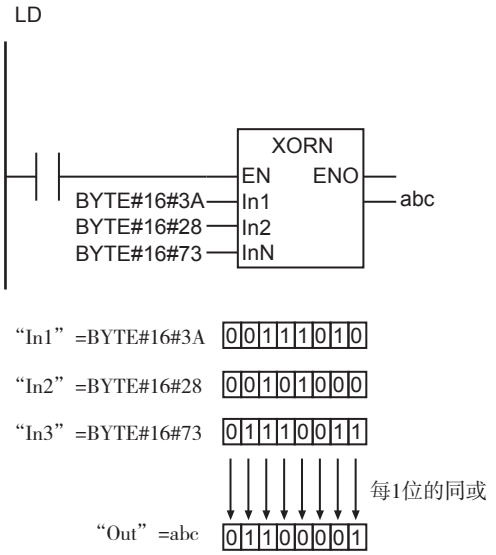
3 通过2的结果和 “In4”进行运算

: :

运算的输入输出关系如下所示。两者的值相同时，运算结果为TRUE，除此以外为FALSE。

“In1”的位	“In2”的位	“Out”的位
FALSE	FALSE	TRUE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

“In1”=BYTE#16#3A、“In2”=BYTE#16#28、“In3”=BYTE#16#73时的示例如下所示。

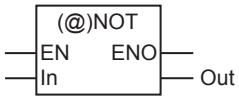


使用注意事项

请将 “In1” ~ “InN” 和 “Out” 的数据类型全部设为相同。否则，编连时会发生异常。

NOT

对布尔、位串的各个位进行取反。

指令	名称	FB/FUN	图形表现	ST表现
NOT	位取反	FUN		Out:=NOT(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	运算对象	输入	运算对象	遵从数据类型	–	(*)
Out	运算结果	输出	运算结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

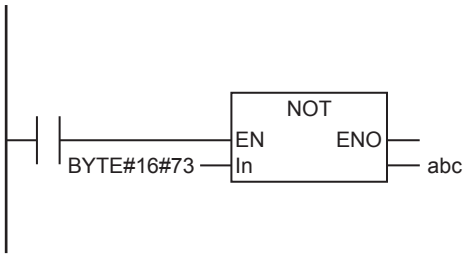
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○	○	○	○	○															
Out	与 “In” 相同的数据类型																			

功能

对布尔或位串的运算对象 “In” 的各个位进行取反。
因此，“In” 与运算结果 “Out” 位数相同，即数据类型必须相同。

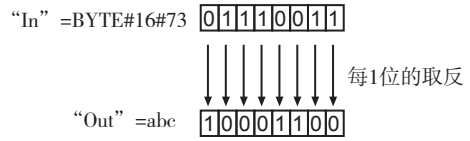
“In” =BYTE#16#73时的示例如下所示。

LD



ST

```
abc:=NOT(BYTE#16#73);
```



使用注意事项

请将 “In” 和 “Out” 的数据类型设为相同。否则，编连时会发生异常。

AryAnd/AryOr/AryXor/AryXorN

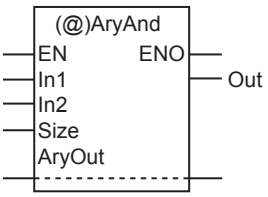
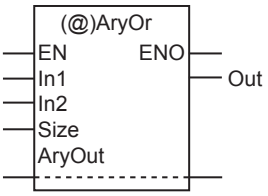
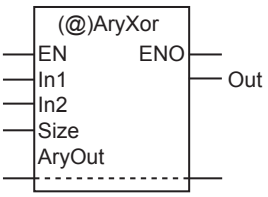
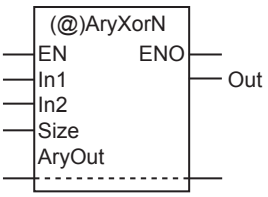
对数组间各元素的布尔、位串的每1位进行运算。

AryAnd ：逻辑积

AryOr ：逻辑和

AryXor ：异或

AryXorN ：同或

指令	名称	FB/FUN	图形表现	ST表现
AryAnd	数组逻辑积	FUN		AryAnd(In1, In2, Size, AryOut);
AryOr	数组逻辑和	FUN		AryOr(In1, In2, Size, AryOut);
AryXor	数组异或	FUN		AryXor(In1, In2, Size, AryOut);
AryXorN	数组同或	FUN		AryXorN(In1, In2, Size, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1[ ] ,In2[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	–	(*)
Size	元素数		运算对象元素数			1
AryOut[ ] 数组	运算结果数组	输入输出	运算结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1[ ] 数组	○	○	○	○	○															
In2[ ] 数组	与In1[ ]相同的数据类型																			
Size							○													
AryOut[ ] 数组	与In1[ ]相同的数据类型																			
Out	○																			

功能

对运算对象数组In1[ ]、In2[ ]开头的 “Size”个元素，对每个对应元素的对应位进行逻辑运算。运算结果保存在AryOut[ ]的对应元素中。
因此，In1[ ]、In2[ ]及AryOut[ ]的数据类型必须相同。

运算的输入输出关系如下所示。

- AryAnd

两者均为TRUE时，运算结果为TRUE。除此以外，运算结果为FALSE。

In1[ ]的元素的位	In2[ ]的元素的位	AryOut[ ]的位
FALSE	FALSE	FALSE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

- AryOr

两者均为FALSE时，运算结果为FALSE。除此以外，运算结果为TRUE。

In1[ ]的元素的位	In2[ ]的元素的位	AryOut[ ]的位
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	TRUE

- AryXor

两者的值相同时，运算结果为FALSE。不同时，运算结果为TRUE。

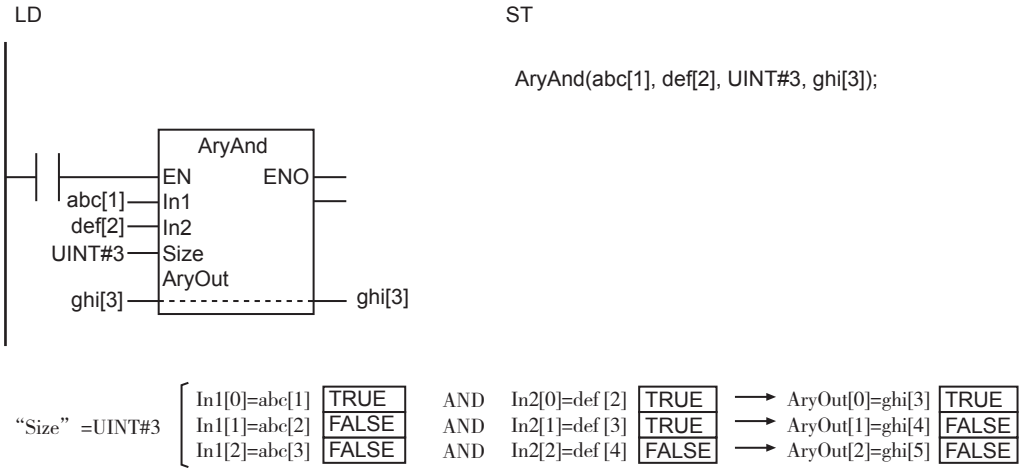
In1[ ]的元素的位	In2[ ]的元素的位	AryOut[ ]的位
FALSE	FALSE	FALSE
FALSE	TRUE	TRUE
TRUE	FALSE	TRUE
TRUE	TRUE	FALSE

- AryXorN

两者的值相同时，运算结果为TRUE。不同时，运算结果为FALSE。

In1[ ]的元素的位	In2[ ]的元素的位	AryOut[ ]的位
FALSE	FALSE	TRUE
FALSE	TRUE	FALSE
TRUE	FALSE	FALSE
TRUE	TRUE	TRUE

AryAnd指令下，“Size”=UINT#3时的示例如下所示。



使用注意事项

- 请将In1[ ]、In2[ ]、AryOut[ ]的数据类型全部设为相同。
- 请将AryOut[ ]的数组元素数设为 “Size”个以上。
- “Size”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Size”的值超过In1[ ]、In2[ ]、AryOut[ ]中任一元素数时。

选择指令

SEL

在2个选项中选择1个。

指令	名称	FB/FUN	图形表现	ST表现
SEL	位选择	FUN	(@)SEL EN ENO G Out In0 In1	Out:=SEL(G、In0、In1);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
G	门	输入	FALSE：选择 “In0” TRUE ：选择 “In1”	遵从数据类型	–	FALSE
In0、In1	选项		选项			(*)
Out	选择结果	输出	选择结果	遵从数据类型	–	–

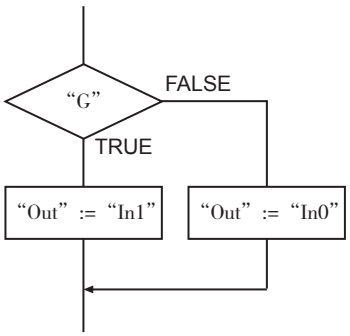
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
G	○																			
In0、In1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	枚举体也可指定*1																			
Out	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	枚举体也可指定*1																			

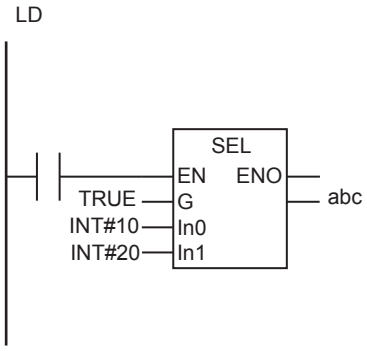
*1 CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时可指定。

功能

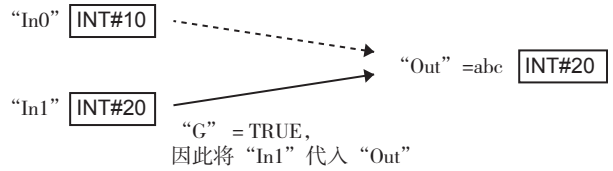
在2个选项 “In0”、“In1”中选择1个。
选择 “In0”与 “In1”中的一个时，通过门 “G”指定。
“G”的值为FALSE时，将 “In0”代入 “Out”；为TRUE时，将 “In1”代入 “Out”。



“In0”=INT#10、“In1”=INT#20、“G”=TRUE时的示例如下所示。变量abc的值为INT#20。



选择“In0”、“In1”中的一个。
“G” = TRUE，因此选择“In1” = INT#20，代入abc。



参考

版本相关信息

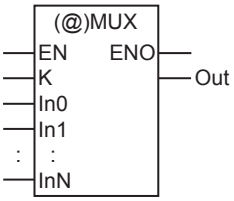
CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时，也有使用 “MUX指令(P.2–328)” 的方法。MUX指令为从2～5个选项中选择1个的指令。

使用注意事项

- “In0”、“In1”、“Out”允许为不同数据类型，但请注意下述事项。
  - “Out”的有效范围须包含 “In0”、“In1”的有效范围。
  - “In0”、“In1”、“Out”不得为非同类数据(位串与整数、整数与字符串等)。

MUX

从2～5个选项中选择1个。

指令	名称	FB/FUN	图形表现	ST表现
MUX	多路复用器	FUN		Out:=MUX(K、In0、In1、…、InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
K	选择器	输入	0：选择 “In0” 1：选择 “In1” 2：选择 “In2” 3：选择 “In3” 4：选择 “In4”	0～N	–	*1
In0～InN	选项		选项 N为1～4*2	遵从数据类型		0*3
Out	选择结果	输出	选择结果	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。

*2 CPU单元Ver.1.01以下或Sysmac Studio Ver.1.02以下时，N的范围为2～4。

*3 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如，N=2时，如果省略连接至In0与In1的输入参数，则初始值适用；如果省略连接至In2的输入参数，则编连时异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
K						○			○											
In0～InN	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	枚举体也可指定*2																			
Out	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	枚举体也可指定*2																			

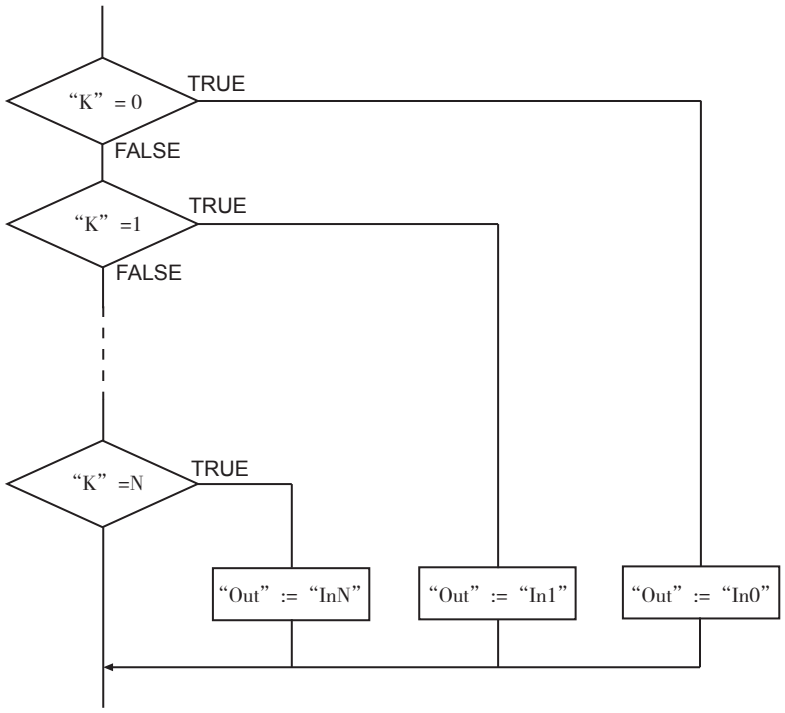
*1 CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时，为ULINT型。CPU单元Ver.1.01以下或Sysmac Studio Ver.1.02以下时，为USINT型。

*2 CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时可指定。

功能

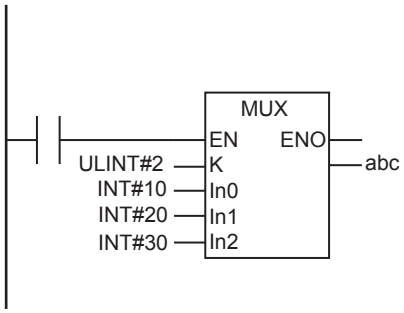
在2～5个选项 “In0” ～ “InN”中选择1个。

选择 “In0” ～ “InN”中的一个时，通过选择器 “K”指定。
“K”的值为0时，将 “In0”代入 “OUT”；为1时，将 “In1”代入 “OUT”；为N时，将符合下述“K”值的选项值代入 “OUT”。



“In0” =INT#10、“In1” =INT#20、“In2” =INT#30、“K” =ULINT#2时的示例如下所示。变量abc的值为INT#30。

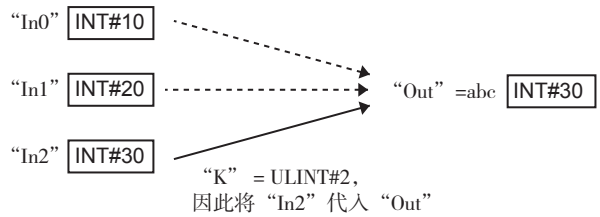
LD



ST

```
abc:=MUX(ULINT#2, INT#10, INT#20, INT#30);
```

选择“In0” ～ “InN”中的一个。
“K” = ULINT#2，因此选择“In2” = INT#30，代入abc。

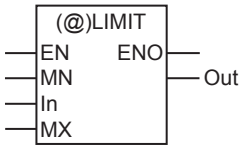


使用注意事项

- “In0” ~ “InN”及 “Out”允许为不同数据类型，但请注意下述事项。
  - “Out”的有效范围须包含 “In0” ~ “InN”的所有有效范围。
  - “In0” ~ “InN”、“Out”不得为非同类数据(位串与整数、整数与字符串等)。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “K”的值不在有效范围内(0以下或大于N)时。

LIMIT

通过设定的上下限值，限制输入变量值。

指令	名称	FB/FUN	图形表现	ST表现
LIMIT	上下限限位限制	FUN		Out:=LIMIT(MN、In、MX);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
MN	下限值	输入	限制的下限值	遵从数据类型	–	(*)
In	限制对象		限制对象			
MX	上限值		限制的上限值			
Out	处理结果	输出	处理结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

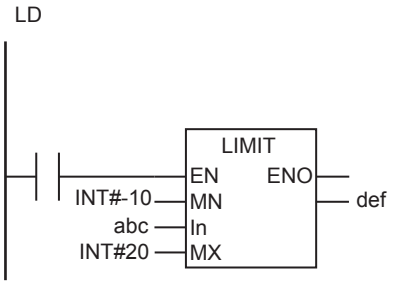
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
MN						○	○	○	○	○	○	○	○	○	○					
In						○	○	○	○	○	○	○	○	○	○					
MX						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

功能

根据上限值 “MX”、下限值 “MN”，对限制对象 “In”的值进行限制。
处理结果 “Out”的值如下表所示。

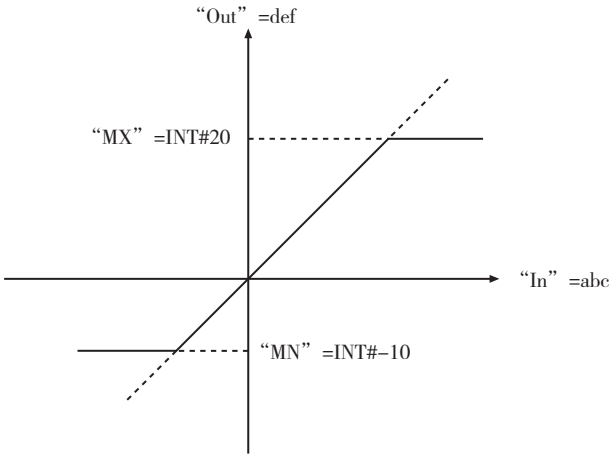
“In”的值	“Out”的值
“In” < “MN”	“MN”
“MN” ≤ “In” ≤ “MX”	“In”
“MX” < “In”	“MX”

“MN”=INT#–10、“MX”=INT#20时的示例如下图所示。



ST

```
def:=LIMIT(INT#-10, abc, INT#20);
```

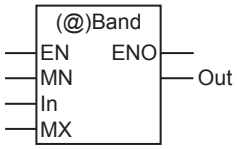


使用注意事项

- “In”、“MN”、“MX”及 “Out”允许为不同数据类型，但请注意下述事项。
  - “Out”的有效范围须包含 “In”、“MN”、“MX”的所有有效范围。
  - “In”、“MN”、“MX”的无符号整数型(USINT、UINT、UDINT、ULINT)不得与带符号整数型(SINT、INT、DINT、LINT)混合。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “MX”的值小于 “MN”的值时。

Band

控制死区(不感带)。

指令	名称	FB/FUN	图形表现	ST表现
Band	死区控制	FUN		Out:=Band(MN、In、MX);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
MN	下限值	输入	死区的下限值	遵从数据类型	–	(*)
In	控制对象		控制对象			
MX	上限值		死区的上限值			
Out	处理结果	输出	处理结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
MN										○	○	○	○	○	○					
In										○	○	○	○	○	○					
MX										○	○	○	○	○	○					
Out										○	○	○	○	○	○					

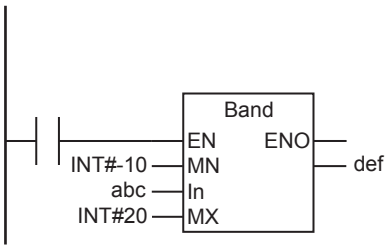
功能

根据上限值 “MX”、下限值 “MN”，对控制对象 “In”的值进行控制。
处理结果 “Out”的值如下表所示。

“In”的值	“Out”的值
“In” < “MN”	“In” – “MN”
“MN” ≤ “In” ≤ “MX”	0
“MX” < “In”	“In” – “MX”

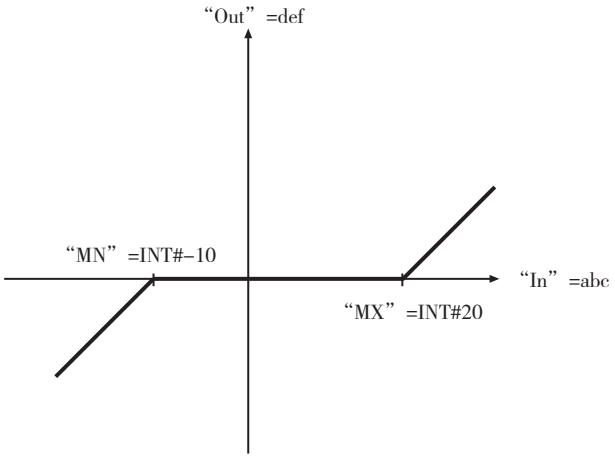
“MN”=INT#−10、“MX”=INT#20时的示例如下图所示。

LD



ST

```
def:=Band(INT#-10, abc, INT#20);
```



使用注意事项

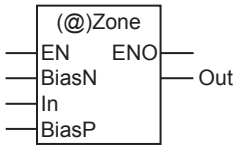
- “In”、“MN”、“MX”及“Out”允许为不同数据类型，但请注意下述事项。
  - “Out”的有效范围须包含“In”、“MN”、“MX”的所有有效范围。
- “In”的值为非数时，“Out”的值为非数。
- “In”、“MN”、“MX”的值为+∞、−∞时，“Out”的值如下表所示。

“In”的值	“MN”的值	“MX”的值	“Out”的值
+∞	+∞	+∞	0
		−∞	异常
	−∞	+∞	0
		−∞	+∞
−∞	+∞	+∞	−∞
		−∞	异常
	−∞	+∞	0
		−∞	0

- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “MX”的值小于“MN”的值时。
  - “MX”、“MN”中的一个为非数时。
  - 处理结果超过“Out”的有效范围时。

Zone

将输入值加上偏置值。

指令	名称	FB/FUN	图形表现	ST表现
Zone	死区控制	FUN		Out;=Zone(BiasN、In、BiasP);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
BiasN	负偏置值	输入	负偏置值	遵从数据类型	–	(*)
In	控制对象		控制对象			
BiasP	正偏置值		正偏置值			
Out	处理结果	输出	处理结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

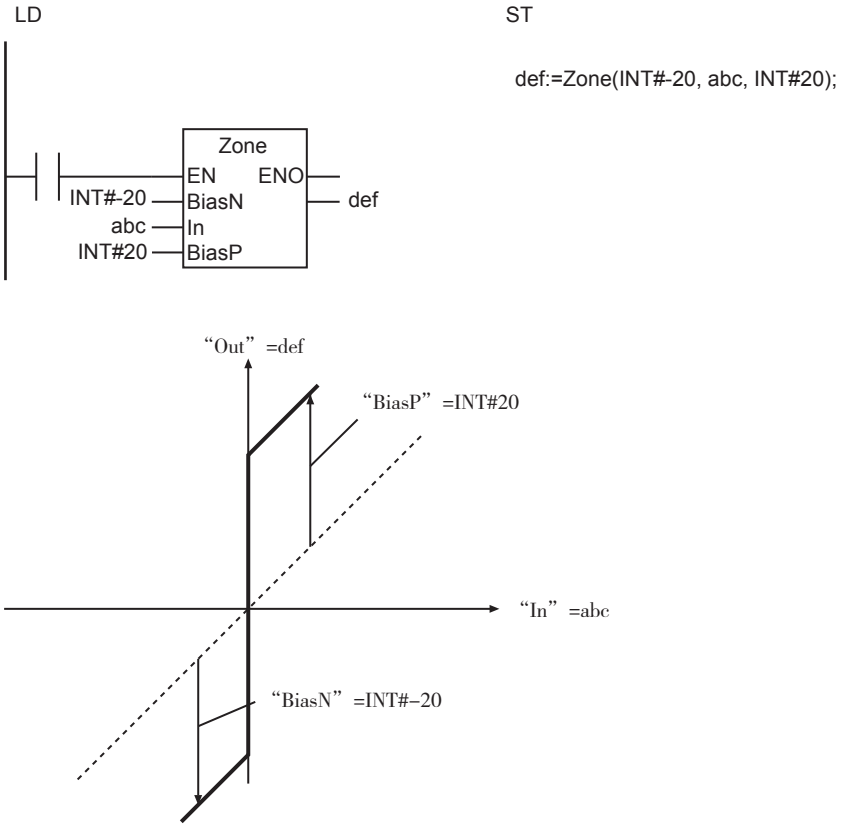
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
BiasN										○	○	○	○	○	○					
In										○	○	○	○	○	○					
BiasP										○	○	○	○	○	○					
Out										○	○	○	○	○	○					

功能

根据正偏置值 “BiasP”、负偏置值 “BiasN”，对控制对象 “In”的值进行控制。
处理结果 “Out”的值如下表所示。

“In”的值	“Out”的值
“In” <0	“In” + “BiasN”
“In” =0	0
0< “In”	“In” + “BiasP”

“BiasP”=INT#20、“BiasN”=INT#−20时的示例如下图所示。



使用注意事项

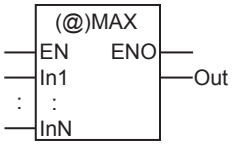
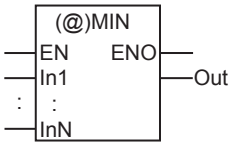
- “In”、“BiasP”、“BiasN”及 “Out”允许为不同数据类型，但请注意下述事项。
  - “Out”的有效范围须包含 “In”、“BiasP”、“BiasN”的所有有效范围。
- “In”的值为非数时，“Out”的值为非数。
- “In”、“BiasP”、“BiasN”的值为+∞、−∞时，“Out”的值如下表所示。

“In”的值	“BiasP”的值	“BiasN”的值	“Out”的值
+∞	+∞	+∞	+∞
		−∞	+∞
	−∞	+∞	异常
		−∞	0
−∞	+∞	+∞	0
		−∞	−∞
	−∞	+∞	异常
		−∞	−∞

- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “BiasP”< “BiasN”时。
  - “BiasP”、“BiasN”中的一个为非数时。
  - 处理结果超过 “Out”的有效范围时。

MAX/MIN

MAX：检索2～5个数值中的最大值。
MIN ：检索2～5个数值中的最小值。

指令	名称	FB/FUN	图形表现	ST表现
MAX	最大值检索	FUN		Out:=MAX(In1、In2、…、InN);
MIN	最小值检索	FUN		Out;=MIN(In1、In2、…、InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1～InN	运算对象	输入	运算对象 N为2～5	遵从数据类型	–	0(*)
Out	检索结果	输出	检索结果	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

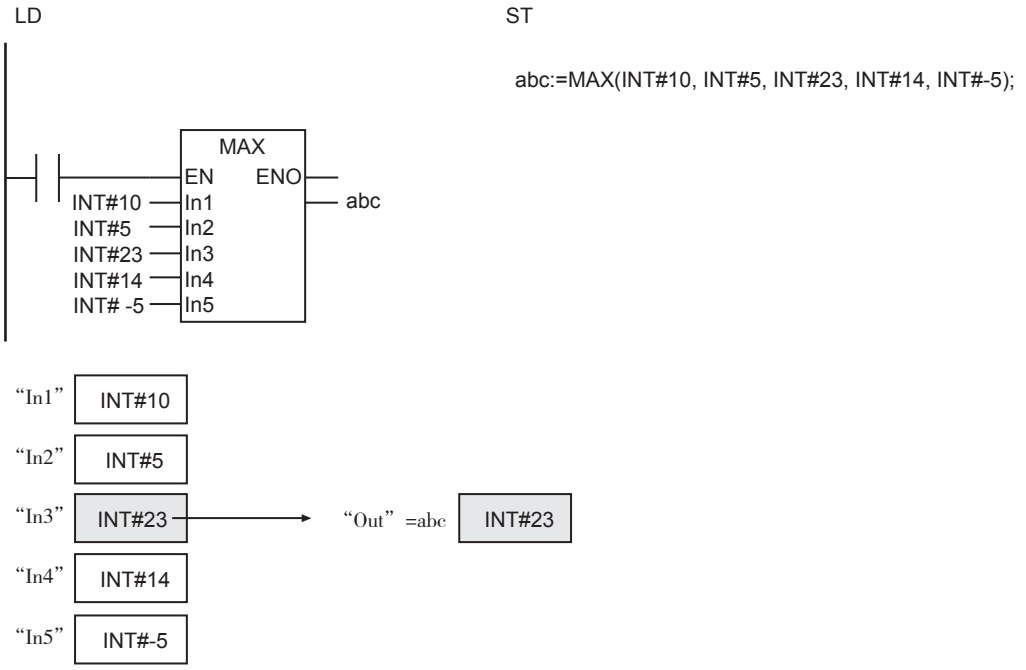
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1～InN						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

功能

- MAX
  检索2～5个运算对象 “In1” ～ “InN”中的最大值。

- MIN
  检索2～5个运算对象 “In1” ～ “InN”中的最小值。

通过MAX指令，使 “In1”=INT#10、“In2”=INT#5、“In3”=INT#23、“In4”=INT#14、“In5”=INT#−5的示例如下图所示。



参考

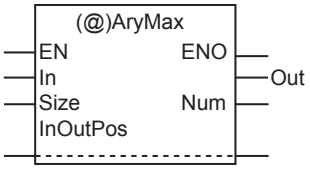
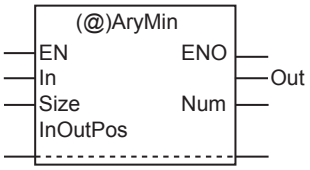
希望检索6个以上运算对象的最大值/最小值时，请使用 “AryMax/AryMin指令(P.2−339)”。

使用注意事项

- “In1” ~ “InN”及 “Out”允许为不同数据类型，但请注意下述事项。
  - “Out”的有效范围须包含 “In1” ~ “InN”的所有有效范围。
  - “In1” ~ “InN”的无符号整数型(USINT、UINT、UDINT、ULINT)不得与带符号整数型(SINT、INT、DINT、LINT)混合。
- “In1” ~ “InN”为实数时，因误差的影响，可能无法获取期望的结果。

AryMax/AryMin

AryMax：检索1维数组元素的最大值。
AryMin：检索1维数组元素的最小值。

指令	名称	FB/FUN	图形表现	ST表现
AryMax	数组变量的最大值检索	FUN		Out:=AryMax(In、Size、InOutPos、Num);
AryMin	数组变量的最小值检索	FUN		Out:=AryMin(In、Size、InOutPos、Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	检索对象数组	输入	检索对象数组	遵从数据类型	–	(*)
Size	检索对象的元素数		在In[ ]的元素中，作为检索对象的元素数量			1
InOutPos	检索元素编号	输入输出	检索值的数组元素编号	遵从数据类型	–	–
Out	检索结果	输出	检索结果	遵从数据类型	–	–
Num	检索值的数量		检索值的数量			

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
																(*)				
Size							○													
InOutPos							○													
Out						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
																(*)				
Num							○													

* TIME型、DATE型、TOD型、DT型、STRING型在Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio中可指定。

功能

针对从检索对象数组In[ ]的In[0]开始的 “Size”个的数组元素进行检索。
分别将检索值代入 “Out”、检索元素编号代入 “InOutPos”、检索值的数量代入 “Num”。“Num”大于1时，“InOutPos”的值变为检索值中最低位的元素编号。

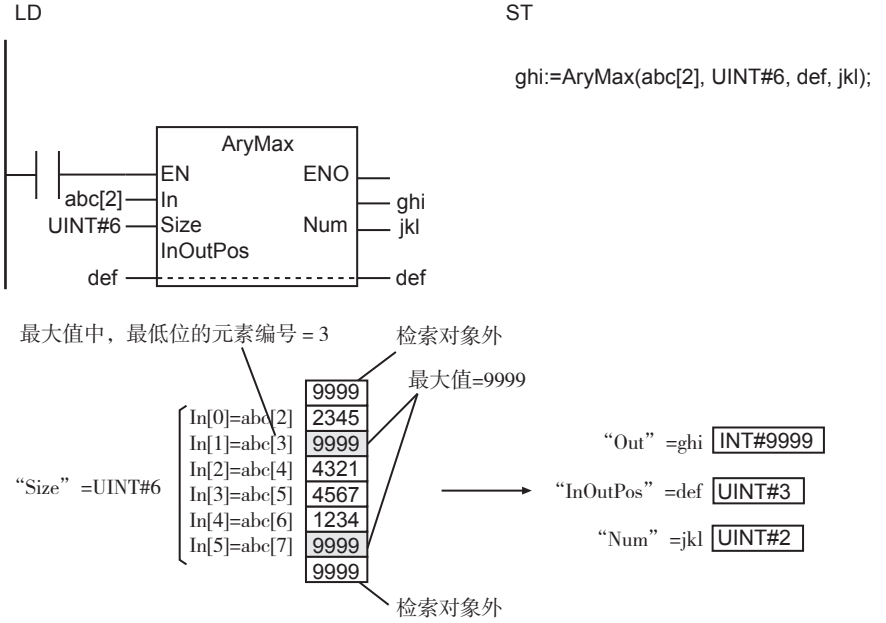
整数、实数以外的数据类型的值的大小关系的判断如下表所示。

数据类型	大小关系
TIME	值较大者判断为大。
DATE、TOD、DT	对于日期和时刻，较后者判断为大。
STRING	与 “LTascii/LEascii/GTascii/GEascii指令(P.2-105)” 的规格相同。请参阅该处。

- **AryMax**
  检索最大值。

- **AryMin**
  检索最小值。

AryMax指令下，“Size”=UINT#6时的示例如下所示。
发送至In[ ]的输入参数为abc[2]，因此abc[2]以后变为检索对象。



参考

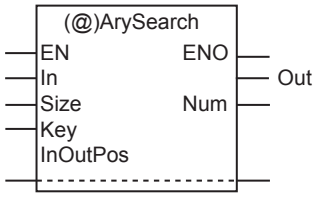
比较TIME型、DT型、TOD型的大小时，请符合待比较的值的精度。有 “TruncTime指令(P.2-642)”、“TruncDt指令(P.2-646)”、“TruncTod指令(P.2-650)”，以符合值的精度。

使用注意事项

- In[ ]和 “Out”的数据类型不同时，请将 “Out”的有效范围设为包含In[ ]的有效范围。
- In[ ]为实数时，因误差的影响，可能无法获取期望的结果。
- 请务必使In[ ]为1维数组。
- “Size”的值为0时，“Out”、“Num”的值变为0。此外，“InOutPos”的值不变。
- In[ ]为STRING型、“Size”的值为0时，“Out”仅为NULL字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过有效范围时。
  - “Size”超出In[ ]的数组区域时。
  - In[ ]非1维数组时。
  - In[ ]为STRING型，且未以NULL字符结尾时。

ArySearch

在1维数组中检索指定值。

指令	名称	FB/FUN	图形表现	ST表现
ArySearch	数组检索	FUN		Out:=ArySearch(In、Size、Key、InOutPos、Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	检索对象数组	输入	检索对象数组	遵从数据类型	–	(*)
Size	检索对象的元素数		在In[ ]的元素中，作为检索对象的元素数量			1
Key	检索关键词		检索值	遵从数据类型		(*)
InOutPos	检索元素编号	输入输出	检索值的数组元素编号	遵从数据类型	–	–
Out	检索结果	输出	检索结果	遵从数据类型	–	–
Num	检索值的数量		检索值的数量			

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体的数组															*1				
Size							○													
Key	与In[ ]的元素相同的数据类型																			
InOutPos							○													
Out	○																			
Num							○													

*1 TIME型、DATE型、TOD型、DT型在Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio中可指定。

功能

针对从1维检索对象数组In[ ]的In[0]开始的“Size”个的数组元素，检索与检索关键词“Key”同值的元素。

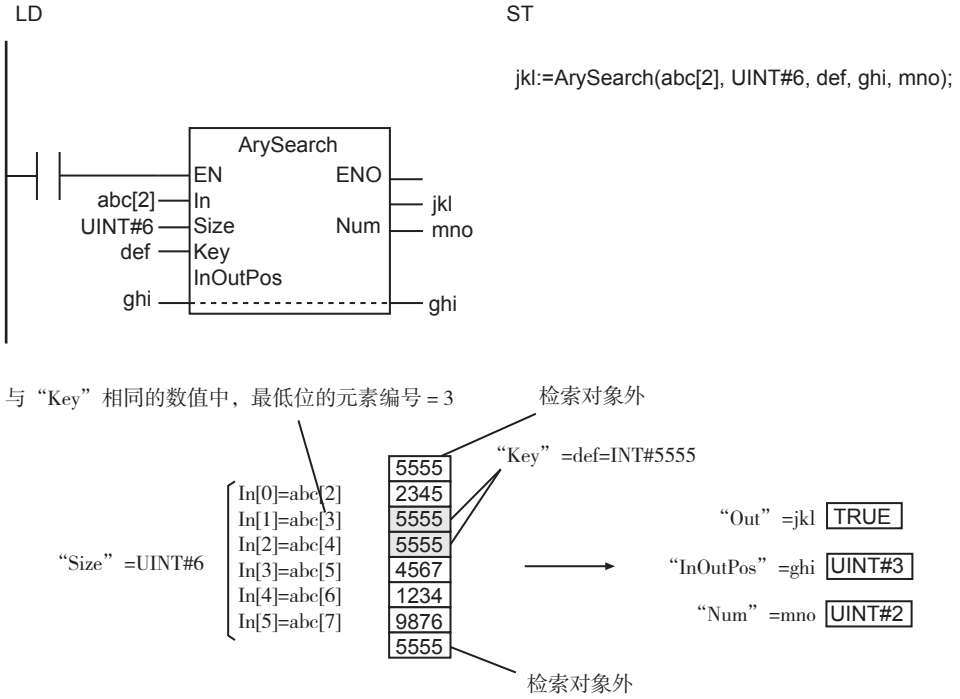
检索结果 “Out”、检索元素编号 “InOutPos”、检索值的数量 “Num”的值如下表所示。

与 “Key”同值的元素的存在	“Out”	“InOutPos”	“Num”
存在	TRUE	与 “Key”同值的元素中，最低位元素的元素编号	与 “Key”同值的元素数量
不存在	FALSE	不变	0

整数、实数以外的数据类型的值的大小关系的判断如下表所示。

数据类型	大小关系
TIME	值较大者判断为大。
DATE、TOD、DT	对于日期和时刻，较后者判断为大。

“Size”=UINT#6时的示例如下所示。
发送至In[ ]的输入参数为abc[2]，因此abc[2]以后变为检索对象。



参考

比较TIME型、DT型、TOD型的大小时，请符合待比较的值的精度。有 “TruncTime指令(P.2-642)”、“TruncDt指令(P.2-646)”、“TruncTod指令(P.2-650)”，以符合值的精度。

使用注意事项

- 请务必使In[ ]为1维数组。
- 请确保 “Key”与In[ ]的元素为相同数据类型。
- “Size”的值为0时，“Out”、“Num”的值变为0。此外，“InOutPos”的值不变。
- 请务必将传输至 “Key”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “Key”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、“Num”、“InOutPos”不变。
  - “Size”超出In[ ]的数组区域时。
  - In[ ]或 “Key”为STRING型，且未以NULL字符结尾时。
  - In[ ]非1维数组时。

数据传送指令

指令	名称	页码
MOVE	传送	2–346
MoveBit	位传送	2–349
MoveDigit	数字传送	2–351
TransBits	多位传送	2–353
MemCopy	存储器复制	2–355
SetBlock	块设定	2–357
Exchange	数据交换	2–359
AryExchange	数组数据交换	2–361
AryMove	数组的传送	2–363
Clear	初始化	2–365
Copy**ToNum (位串→带符号整数)	位模式复制(位串→带符号整数)组	2–367
Copy**To*** (位串→实数)	位模式复制(位串→实数)组	2–369
CopyNumTo** (带符号整数→位串)	位模式复制(带符号整数→位串)组	2–371
CopyNumTo** (带符号整数→实数)	位模式复制(带符号整数→实数)组	2–373
Copy**To*** (实数→位串)	位模式复制(实数→位串)组	2–375
Copy**ToNum (实数→带符号整数)	位模式复制(实数→带符号整数)组	2–376

MOVE

将变量和常数值传送至其它变量。

指令	名称	FB/FUN	图形表现	ST表现
MOVE	传送	FUN	(@)MOVE EN ENO In Out	Out:=In;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	传送源	输入	传送源	遵从数据类型	–	(*)
Out	传送目标	输出	传送目标	遵从数据类型	–	(*)

* 省略输入参数时，初始值不适用。编连时会发生异常。

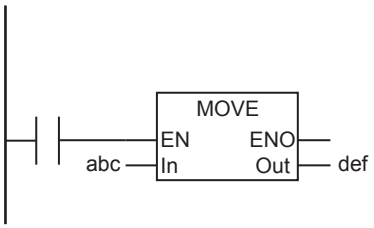
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素																			
Out	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	“In”为枚举体、数组的1个元素、整个结构体、结构体的1个结构要素时，指定与 “In”相同的数据类型 “In”为整个数组时，指定与 “In”相同的数据类型、大小、下标的数组																			

功能

将传送源 “In”的值传送至传送目标 “Out”。
传输至 “In”的输入参数可为常数，也可为变量。此外，也可给 “In”指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素。

示例如下所示。将变量abc的内容传送至变量def。

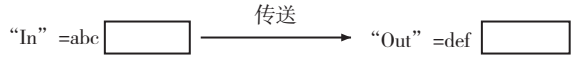
LD



ST

```
def:=abc;
```

将“In”的值传送至“Out”。

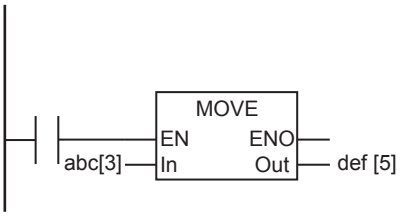


参考

- 传送数组时，可仅传送1个元素，也可传送整个数组。仅传送1个元素时，给数组变量名添加下标。传送整个数组时，不给数组变量名添加下标。

  仅传送1个数组元素时

  LD

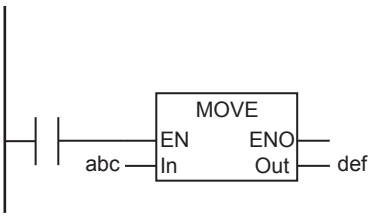
  

  ST

  ```
  def [5]:=abc[3];
  ```

  传送整个数组时

  LD

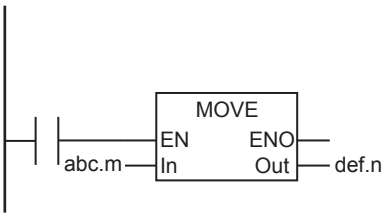
  ST

  ```
  def:=abc;
  ```

- 传送结构体时，可仅传送1个结构要素，也可传送整个结构体。仅传送1个结构要素时，明示结构要素。传送整个结构体时，仅记述结构体名。

  仅传送1个结构体的结构要素时

  LD

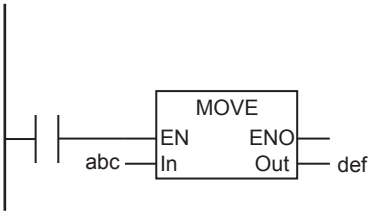
  

  ST

  ```
  def.n:=abc.m;
  ```

  传送整个结构体时

  LD

  ST

  ```
  def:=abc;
  ```

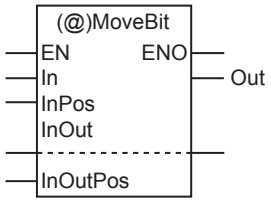
- 传送整个数组时，可通过本指令高速处理MemCopy指令。

使用注意事项

- “In”和 “Out”的数据类型不同时，请设为无论下述哪一种数据类型群，“Out”的有效范围均包含“In”的有效范围的组合。
  - BYTE、WORD、DWORD、LWORD
  - USINT、UINT、UDINT、ULINT、SINT、INT、DINT、LINT、REAL、LREAL
- “In”为枚举体、数组的1个元素、整个结构体、结构体的1个结构要素时，请使 “Out”与 “In”为相同的数据类型。
- “In”为整个数组时，请使 “Out”与 “In”为相同的数据类型、大小、下标的数组。

MoveBit

传送位串内的1位。

指令	名称	FB/FUN	图形表现	ST表现
MoveBit	位传送	FUN		MoveBit(In、InPos、InOut、InOutPos);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	传送源	输入	传送源	遵从数据类型	–	(*)
InPos	传送源的位位置		“In”中的传送位位置	0～“In”的位数–1		0
InOutPos	传送目标的位位置		“Out”中的传送目标位位置	0～“InOut”的位数–1		
InOut	传送目标	输入输出	传送目标	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

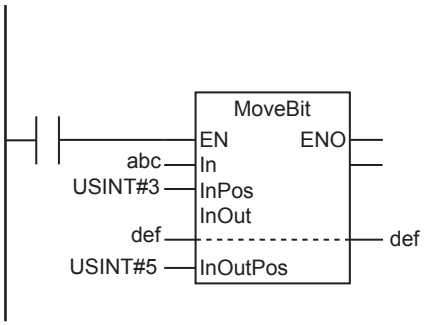
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
InPos						○														
InOutPos						○														
InOut		○	○	○	○															
Out	○																			

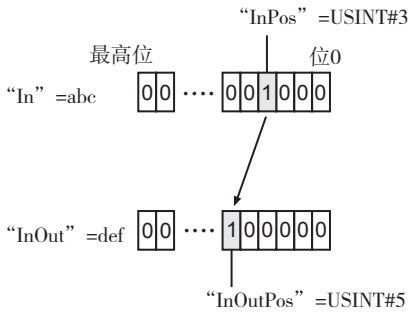
功能

将传送源 “In”的位位置 “InPos”的1位数据传送至传送目标 “InOut”的位位置 “InOutPos”。

“InPos”=USINT#3、“InOutPos”=USINT#5时的示例如下所示。

LD





ST

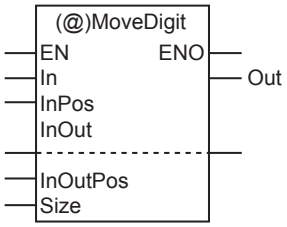
```
MoveBit(abc, USINT#3, def, USINT#5);
```

使用注意事项

- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“InOut”不变。
  - “InPos”的值超过有效范围时。
  - “InOutPos”的值超过有效范围时。

MoveDigit

传送位串内的多个数位(1个数位为4位)。

指令	名称	FB/FUN	图形表现	ST表现
MoveDigit	数字传送	FUN		MoveDigit(In、InPos、InOut、InOutPos、Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	传送源	输入	传送源	遵从数据类型	–	(*1)
InPos	传送源的数位		“In”中的传送数位	(*2)		0
InOutPos	传送目标的数位		“Out”中的传送目标数位	(*3)		
Size	传送位数		传送位数	(*4)		1
InOut	传送目标	输入输出	传送目标	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 0～“In”的位数/4 – 1
*3 0～“InOut”的位数/4 – 1
*4 0～“In”的位数/4

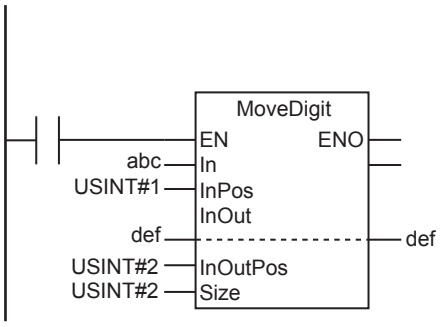
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
InPos						○														
InOutPos						○														
Size						○														
InOut		○	○	○	○															
Out	○																			

功能

从传送源“In”的数位“InPos”，将“Size”位传送至传送目标“InOut”的数位“InOutPos”。1个数位为4位。

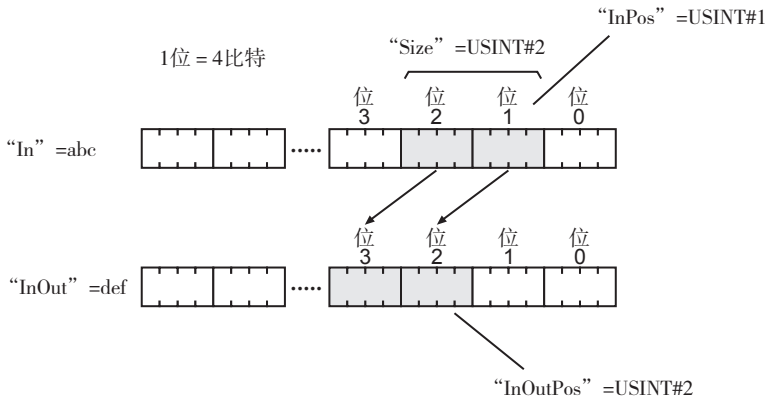
“InPos”=USINT#1、“InOutPos”=USINT#2、“Size”=USINT#2时的示例如下所示。

LD



ST

```
MoveDigit(abc, USINT#1, def, USINT#2, USINT#2);
```

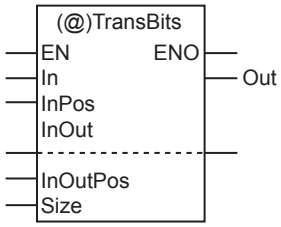


使用注意事项

- 指定传送目标的数位超出 “InOut”的最高位时，使超出的数据绕至 “InOut”的最低位后保存。
- 指定传送源的数位超出 “In”的最高位时，使超出的数据绕至 “In”的最低位后传送。
- “Size”的值为0时，“Out”的值变为TRUE，“InOut”不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“InOut”不变。
  - “InPos”的值超过有效范围时。
  - “InOutPos”的值超过有效范围时。
  - “Size”的值超过有效范围时。

TransBits

传送位串内的多位。

指令	名称	FB/FUN	图形表现	ST表现
TransBits	多位传送	FUN		TransBits(In、InPos、InOut、InOutPos、Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	传送源	输入	传送源	遵从数据类型	–	(*1)
InPos	传送源的位位置		“In”中的传送位位置	(*2)		0
InOutPos	传送目标的位位置		“Out”中的传送目标位位置	(*3)		
Size	传送位数		传送位数	(*4)		1
InOut	传送目标	输入输出	传送目标	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 0～“In”的位数－1
*3 0～“InOut”的位数－1
*4 0～“In”的位数

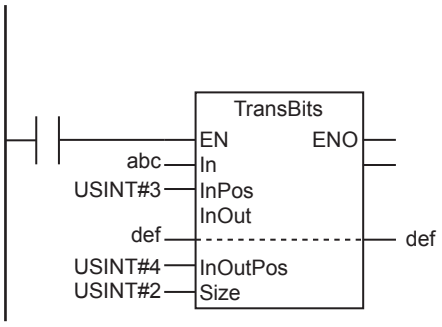
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
InPos						○														
InOutPos						○														
Size						○														
InOut		○	○	○	○															
Out	○																			

功能

从传送源 “In”的位位置 “InPos”，将 “Size”位传送至传送目标 “InOut”的位位置 “InOutPos”。

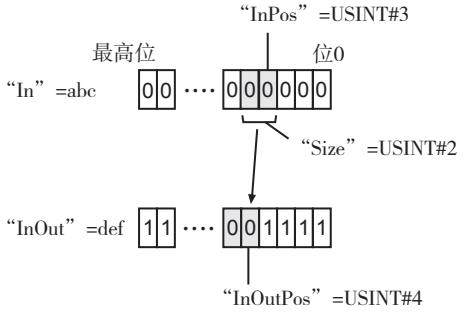
“InPos”=USINT#3、“InOutPos”=USINT#4、“Size”=USINT#2时的示例如下所示。

LD



ST

```
TransBits(abc, USINT#3, def, USINT#4, USINT#2);
```



参考

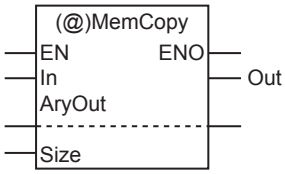
传送源与传送目标的数据区域允许重叠。

使用注意事项

- 请勿指定传送源及传送目标的位位置超出 “In”及 “InOut”的最高位。此时变为异常，不进行动作。
- “Size”的值为0时，不进行传送。
- “InOut”内与传送无关的位不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“InOut”不变。
  - “InPos”的值超过有效范围时。
  - “InOutPos”的值超过有效范围时。
  - “Size”的值超过有效范围时。
  - “InPos”和 “Size”的指定超过 “In”的位数时。
  - “InOutPos”和 “Size”的指定超过 “InOut”的位数时。

MemCopy

传送多个数组元素。传送源和传送目标仅限相同数据类型。

指令	名称	FB/FUN	图形表现	ST表现
MemCopy	存储器复制	FUN		MemCopy(In、AryOut、Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	传送源数组	输入	传送源数组	遵从数据类型	–	(*)
Size	传送元素数量		传送数组元素数量			1
AryOut[ ] 数组	传送目标数组	输入输出	传送目标数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

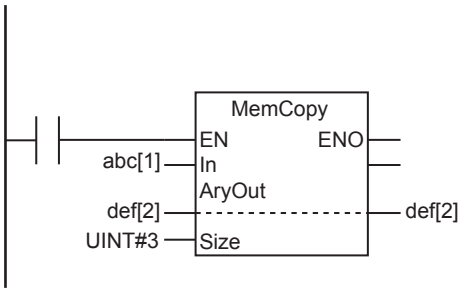
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
Size							○													
AryOut[ ] 数组	与In[ ]相同数据类型的数组																			
Out	○																			

功能

将从传送源数组In[ ]的In[0]开始的 “Size”个的元素传送至传送目标数组AryOut[ ]的AryOut[0]之后。

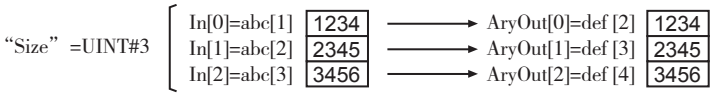
“Size”=UINT#3的示例如下所示。

LD



ST

```
MemCopy(abc[1], def[2], UINT#3);
```



参考

- 也可给In[ ]与AryOut[ ]指定相同数组的不同位置。此外，两者的位置允许重复。

  In=A[2]、AryOut=A[4]、“Size”=UINT#3时的示例如下所示。

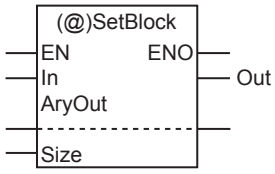
- 传送源和传送目标的数据类型不同时，请使用 “AryMove指令(P.2-363)”。
- 传送源和传送目标的数据类型相同时，可通过AryMove指令高速处理本指令。
- 传送非数组变量时，请使用 “MOVE指令(P.2-346)”。

使用注意事项

- 请将In[ ]和AryOut[ ]的数据类型设为相同。如果不同，则编连时会发生异常。
- In[ ]和AryOut[ ]为STRING型的数组时，请使各区域大小保持一致。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Size”超出In[ ]的数组区域时。
  - “Size”超出AryOut[ ]的数组区域时。

SetBlock

将变量和常数的值传送至多个数组元素。

指令	名称	FB/FUN	图形表现	ST表现
SetBlock	块设定	FUN		SetBlock(In、AryOut、Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	传送源	输入	传送源	遵从数据类型	–	(*)
Size	传送元素数量		传送数组元素数量			1
AryOut[ ]数组	传送目标数组	输入输出	传送目标数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

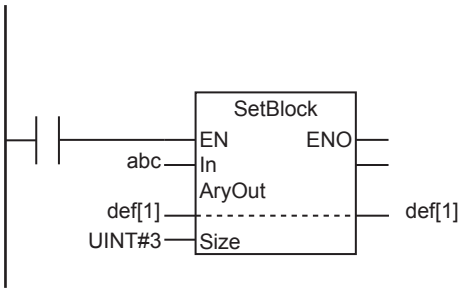
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个结构体、结构体的1个结构要素																			
Size							○													
AryOut[ ]数组	将与 “In”相同的数据类型作为元素的数组																			
Out	○																			

功能

将传送源 “In”的值传送至从传送目标数组AryOut[ ]的AryOut[0]开始的 “Size”个的位置。

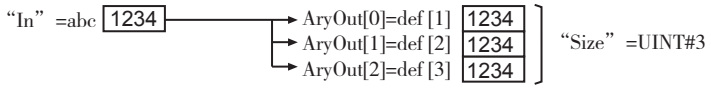
“Size”=UINT#3时的示例如下所示。

LD



ST

```
SetBlock(abc, def[1], UINT#3);
```

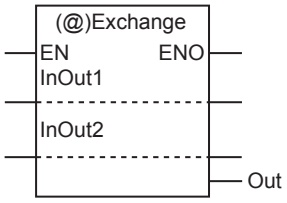


使用注意事项

- 请将 “In”和AryOut[ ]的数据类型设为相同。如果不同，则编连时会发生异常。
- “In”和AryOut[ ]为STRING型时，请使各区域大小保持一致。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Size”的值超过AryOut[ ]的数组区域时。

Exchange

更换2个变量值。

指令	名称	FB/FUN	图形表现	ST表现
Exchange	数据交换	FUN		Exchange(InOut1、InOut2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut1、InOut2	交换对象	输入输出	交换对象	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

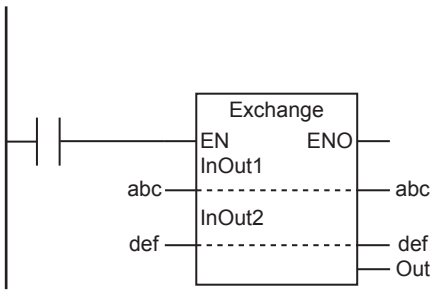
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个结构体、结构体的1个结构要素																			
InOut2	与 “InOut1” 相同的数据类型																			
Out	○																			

功能

交换作为交换对象的 “InOut1” 与 “InOut2” 的值。
也可给 “InOut1”、“InOut2” 指定枚举体、整个结构体、结构体的1个结构要素。

示例如下所示。交换变量abc的值与变量def的值。

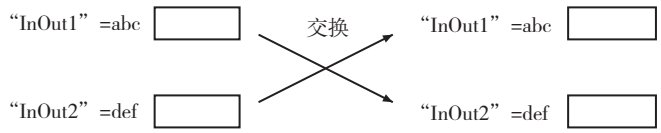
LD



ST

```
Exchange(abc, def);
```

交换“InOut1”与“InOut2”的值。

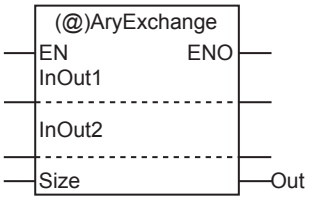


使用注意事项

- 请将 “InOut1”和 “InOut2”的数据类型设为相同。如果不同，则编连时会发生异常。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“InOut1”、“InOut2”不变。
  - “InOut1”、“InOut2”为STRING型，任一字符串长度不符合另一方的大小时。

AryExchange

更换2个数组间的元素。

指令	名称	FB/FUN	图形表现	ST表现
AryExchange	数组数据交换	FUN		AryExchange(InOut1、InOut2、Size);

变量

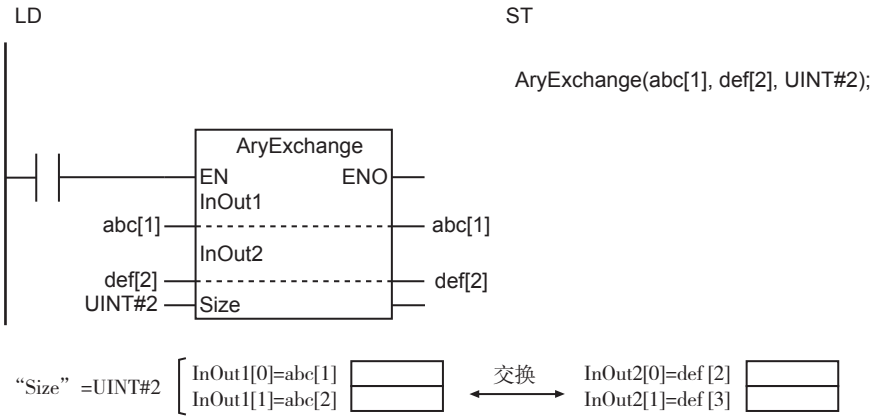
	名称	输入/输出	内容	有效范围	单位	初始值
Size	交换元素数量	输入	交换元素数量	遵从数据类型	–	1
InOut1[ ]、InOut2[ ]数组	交换对象数组	输入输出	交换对象数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Size							○													
InOut1[ ]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
InOut2[ ]数组	与InOut1[ ]相同的数据类型的数组																			
Out	○																			

功能

交换从交换对象数组InOut1[ ]的InOut1[0]开始的 “Size”个的元素、从InOut2[ ]的InOut2[0]开始的 “Size”个的元素。

“Size”=UINT#2时的示例如下所示。



参考

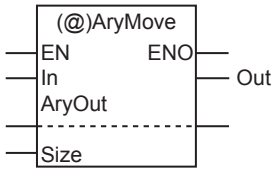
- 将常数代入变量时，请使用 “MOVE指令(P.2-346)”。
- 将变量值复制至其它变量时，请使用 “MemCopy指令(P.2-355)”。

使用注意事项

- 请将InOut1[ ]和InOut2[ ]元素的数据类型设为相同。如果不同，则编连时会发生异常。
- “Size”的值为0时，“Out”的值变为TRUE，InOut1[ ]与InOut2[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，InOut1[ ]与InOut2[ ]不变。
  - “Size”的值超出InOut1[ ]或InOut2[ ]的数组区域时。
  - InOut1[ ]、InOut2[ ]为STRING型的数组，且存在字符串长度超过另一方大小的元素时。
  - InOut1[ ]、InOut2[ ]为STRING型的数组，且存在不以NULL字符结尾的要素时。

AryMove

传送多个数组元素。传送源和传送目标的数据类型可不同。

指令	名称	FB/FUN	图形表现	ST表现
AryMove	数组的传送	FUN		AryMove(In、AryOut、Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	传送对象数组	输入	传送数组	遵从数据类型	–	(*)
Size	传送元素数量		传送元素数量			1
AryOut[ ] 数组	传送结果数组	输入输出	传送结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

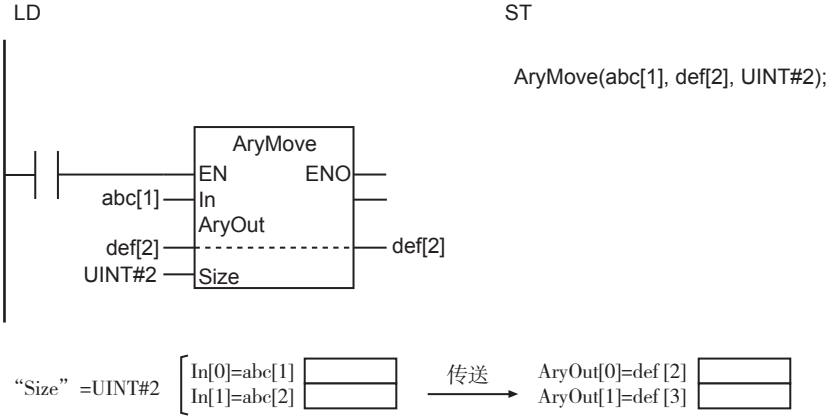
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
Size							○													
AryOut[ ] 数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
Out	○																			

功能

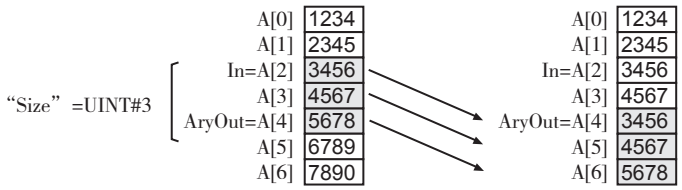
将从传送对象数组In[ ]的In[0]开始的 “Size”个的元素传送至从传送结果数组AryOut[ ]的AryOut[0]开始的元素。
In[ ]与AryOut[ ]的数据类型允许不同。

“Size”=UINT#2时的示例如下所示。



参考

- In[ ]和AryOut[ ]的数据类型相同时，可高速处理MemCopy指令。
- 也可给 In[ ] 和 AryOut[ ] 指定相同的数组。此时，传送源元素与传送目标元素允许重叠。In[0]=A[2]、AryOut[0]=A[4]、“Size”=UINT#3时的示例如下所示。

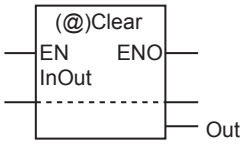


使用注意事项

- In[ ]和AryOut[ ]的数据类型不同时，请设为无论下述哪一种数据类型群，AryOut[ ]的有效范围均包含In[ ]的有效范围的组合。
  - BYTE、WORD、DWORD、LWORD
  - USINT、UINT、UDINT、ULINT、SINT、INT、DINT、LINT、REAL、LREAL
- In[ ]为将结构体作为元素的数组时，请将In[ ]和AryOut[ ]的数据类型设为相同。
- “Size”的值为0时，“Out”的值为TRUE，AryOut[ ]不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Size”的值超过In[ ]或AryOut[ ]的大小时。
  - In[ ]、AryOut[ ]为STRING型的数组，且传送元素的字符串长度超过AryOut[ ]元素的大小时。

Clear

初始化变量。

指令	名称	FB/FUN	图形表现	ST表现
Clear	初始化	FUN		Clear(InOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut	初始化对象	输入输出	初始化对象	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素																			
Out	○																			

功能

使初始化对象 “InOut”的值初始化。
设定变量的初始值属性时，初始化为该值。未设定初始值属性时，变为各数据类型默认的初始值。
“InOut”为外部变量时，与与其对应的全局变量的初始值属性无关，为“InOut”数据类型的默认初始值。

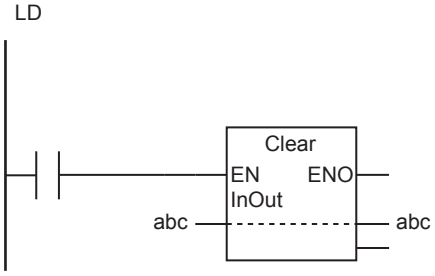
各数据类型默认的初始值如下表所示。

数据类型	初始值
BOOL	FALSE
BYTE、WORD、DWORD、LWORD	16#0
USINT、UINT、UDINT、ULINT、SINT、INT、DINT、LINT、REAL、LREAL	0
TIME	T#0ms
DATE	D#1970–1–1
TOD	TOD#0:0:0
DT	DT#1970–1–1–0:0:0
STRING	''

“InOut”为整个数组、数组的1个元素、整个结构体、结构体的1个结构要素时，按下表所示进行处理。

“InOut”	处理内容
整个数组	使数组的所有元素初始化。
数组的1个元素	仅使该元素初始化。
整个结构体	使结构体的所有结构要素初始化。
结构体的1个结构要素	仅使该结构要素初始化。

示例如下所示。使变量abc的值初始化。例如，abc=INT#100时，abc的值变为INT#0。



ST

```
Clear(abc);
```

使“InOut”的值初始化。
abc的数据类型为INT，因此abc的值变为INT#0。

参考

- “InOut”为作为堆栈使用的数组时，请在执行本指令的同时，将管理堆栈保存数量的变量值清零。
- 通过本指令使凸轮数据变量初始化时，数值并非为通过 MC_SaveCamTable 指令进行保存的值，而是归零。

使用注意事项

- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 使枚举体的变量初始化时，请设定初始值属性。未设定初始值属性时，枚举体的变量值归零。
- 请勿执行符合以下所有条件的处理。否则会导致动作异常。
  - 将BOOL型数组的1个元素作为输入输出变量传输至FUN或FB。
  - 在上述FUN或FB内执行Clear指令。
  - 传输至Clear指令的参数为接收了上述BOOL型数组的1个元素的输入输出变量。

Copy**ToNum(位串→带符号整数)

将位串的内容直接复制到带符号整数。

指令	名称	FB/FUN	图形表现	ST表现
Copy**To Num	位模式复制(位串→带符号整数)组	FUN	(@)Copy**ToNum EN ENO In Out **为位串的数据类型名称	Out:=Copy**ToNum(In); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	复制源	输入	复制源	遵从数据类型	–	0
Out	复制目标	输出	复制目标	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out	与 “In”的数据类型相同大小的带符号整数的数据类型																			

功能

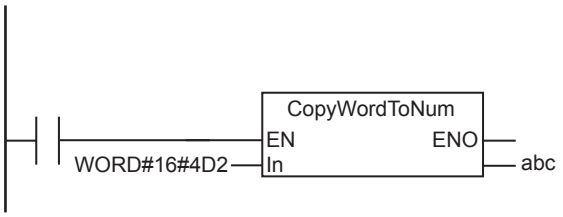
不加工而直接将复制源 “In”的内容复制至复制目标 “Out”。

根据 “In”与 “Out”的数据类型组合，指令名称分为4种。

“In”	“Out”	指令名称
BYTE	SINT	CopyByteToNum
WORD	INT	CopyWordToNum
DWORD	DINT	CopyDwordToNum
LWORD	LINT	CopyLwordToNum

通过CopyWordToNum指令，使 “In”=WORD#16#4D2时的示例如下所示。

LD



ST

```
abc:=CopyWordToNum(WORD#16#4D2);
```

Copy**To***(位串→实数)

将位串的内容直接复制到实数。

指令	名称	FB/FUN	图形表现	ST表现
Copy**To***	位模式复制(位串→实数)组	FUN	(@)CopyDwordToReal EN　ENO In　Out (@)CopyLwordToLreal EN　ENO In　Out	Out:=CopyDwordToReal(In); 或 Out:=CopyLwordToLreal(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	复制源	输入	复制源	遵从数据类型	–	0
Out	复制目标	输出	复制目标	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In				○	○															
Out	“In”的数据类型为DWORD时，则为REAL;为LWORD时，则为LREAL																			

功能

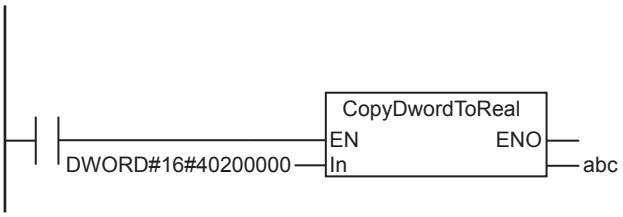
不加工而直接将复制源 “In”的内容复制至复制目标 “Out”。

根据 “In”与 “Out”的数据类型组合，指令名称分为2种。

“In”	“Out”	指令名称
DWORD	REAL	CopyDwordToReal
LWORD	LREAL	CopyLwordToLreal

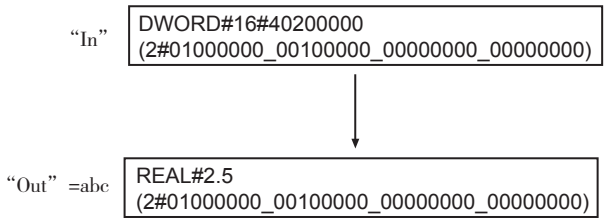
通过CopyDwordToReal指令，使 “In”=DWORD#16#40200000时的示例如下所示。

LD



ST

```
abc:=CopyDwordToReal(DWORD#16#40200000);
```



CopyNumTo**(带符号整数→位串)

将带符号整数的内容直接复制到位串。

指令	名称	FB/FUN	图形表现	ST表现
CopyNumTo**	位模式复制(带符号整数→位串)组	FUN	(@)CopyNumTo** EN ENO In Out **为位串的数据类型名称	Out:=CopyNumTo**(In); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	复制源	输入	复制源	遵从数据类型	–	0
Out	复制目标	输出	复制目标	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In										○	○	○	○							
Out	与 “In” 的数据类型相同大小的位串的数据类型																			

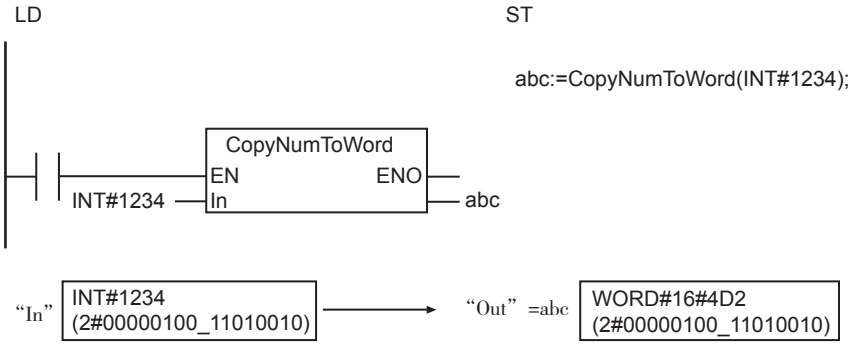
功能

不加工而直接将复制源 “In” 的内容复制至复制目标 “Out”。

根据 “In” 与 “Out” 的数据类型组合，指令名称分为4种。

“In”	“Out”	指令名称
SINT	BYTE	CopyNumToByte
INT	WORD	CopyNumToWord
DINT	DWORD	CopyNumToDword
LINT	LWORD	CopyNumToLword

通过CopyNumToWord指令，使 “In”=INT#1234时的示例如下所示。



CopyNumTo**(带符号整数→实数)

将带符号整数的内容直接复制到实数。

指令	名称	FB/FUN	图形表现	ST表现
CopyNumTo**	位模式复制(带符号整数→实数)组	FUN	(@)CopyNumToReal EN ENO In Out (@)CopyNumToLreal EN ENO In Out	Out:=CopyNumToReal(In); 或 Out:=CopyNumToLreal(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	复制源	输入	复制源	遵从数据类型	–	0
Out	复制目标	输出	复制目标	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In												○	○							
Out	“In”的数据类型为DINT时，则为REAL;为LINT时，则为LREAL																			

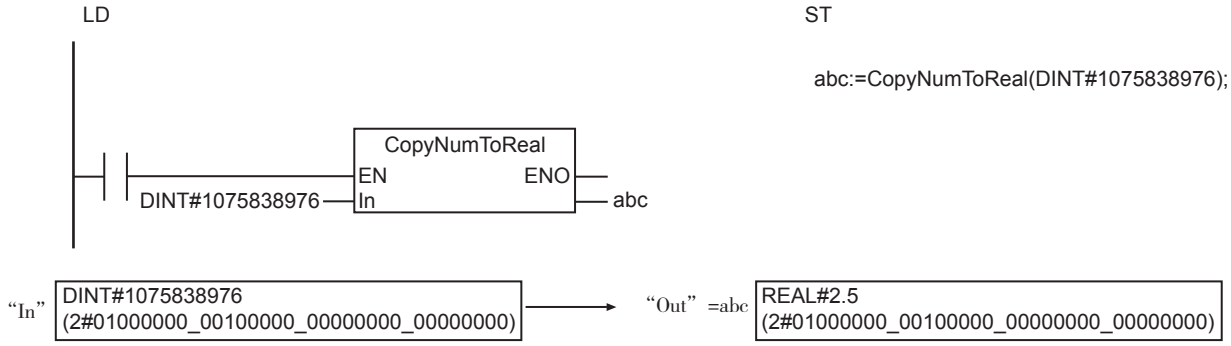
功能

不加工而直接将复制源 “In”的内容复制至复制目标 “Out”。

根据 “In”与 “Out”的数据类型组合，指令名称分为2种。

“In”	“Out”	指令名称
DINT	REAL	CopyNumToReal
LINT	LREAL	CopyNumToLreal

通过CopyNumToReal指令，使 “In” =DINT#1075838976时的示例如下所示。



Copy**To***(实数→位串)

将实数的内容直接复制到位串。

指令	名称	FB/FUN	图形表现	ST表现
Copy**To***	位模式复制(实数→位串)组	FUN	(@)CopyRealToDword EN ENO In Out (@)CopyLrealToLword EN ENO In Out	Out:=CopyRealToDword(In); 或 Out:=CopyLrealToLword(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	复制源	输入	复制源	遵从数据类型	–	0.0
Out	复制目标	输出	复制目标	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out	“In”的数据类型为REAL时，则为DWORD;为LREAL时，则为LWORD																			

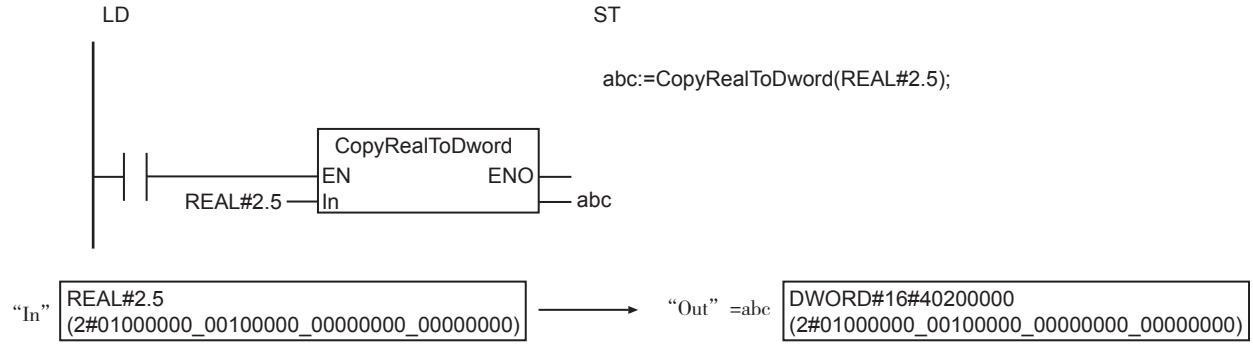
功能

不加工而直接将复制源 “In”的内容复制至复制目标 “Out”。

根据 “In”与 “Out”的数据类型组合，指令名称分为2种。

“In”	“Out”	指令名称
REAL	DWORD	CopyRealToDword
LREAL	LWORD	CopyLrealToLword

CopyRealToDword指令下，“In”=REAL#2.5时的示例如下所示。



Copy**ToNum(实数→带符号整数)

将实数的内容直接复制到带符号整数。

指令	名称	FB/FUN	图形表现	ST表现
Copy**ToNum	位模式复制(实数→带符号整数)组	FUN	(@)CopyRealToNum EN ENO In Out (@)CopyLrealToNum EN ENO In Out	Out:=CopyRealToNum(In); 或 Out:=CopyLrealToNum(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	复制源	输入	复制源	遵从数据类型	–	0.0
Out	复制目标	输出	复制目标	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out	“In”的数据类型为REAL时，则为DINT;为LREAL时，则为LINT																			

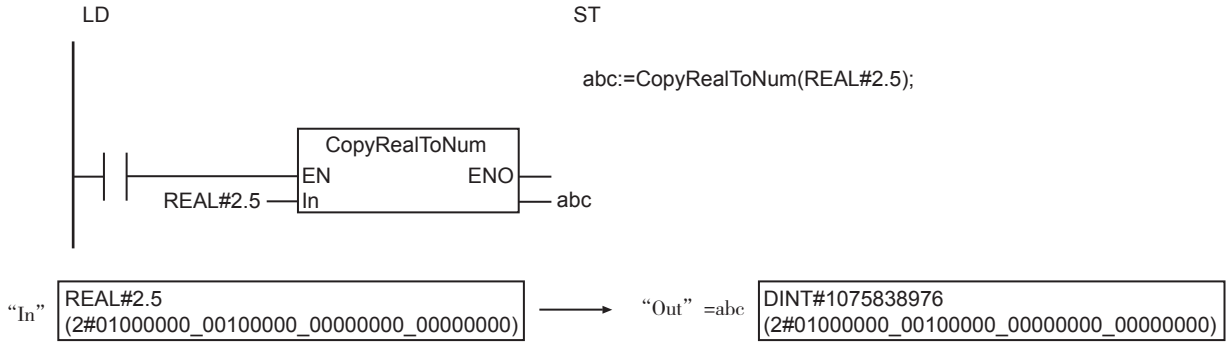
功能

不加工而直接将复制源 “In”的内容复制至复制目标 “Out”。

根据 “In”与 “Out”的数据类型组合，指令名称分为2种。

“In”	“Out”	指令名称
REAL	DINT	CopyRealToNum
LREAL	LINT	CopyLrealToNum

CopyRealToNum指令下，“In”=REAL#2.5时的示例如下所示。

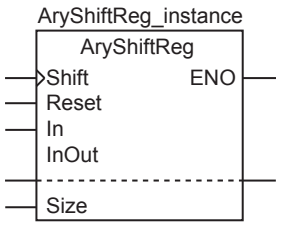


移位指令

指令	名称	页码
AryShiftReg	移位寄存器	2–380
AryShiftRegLR	左右移位寄存器	2–382
ArySHL/ArySHR	数组左移N个元素/ 数组右移N个元素	2–385
SHL/SHR	左移N位/右移N位	2–388
NSHLC/NSHRC	N位数据带CY左移N位/ N位数据带CY右移N位	2–390
ROL/ROR	左旋转N位组/右旋转N位组	2–392

AryShiftReg

将数组元素组成的整个位串左移1位，并将输入值插入最低位。

指令	名称	FB/FUN	图形表现	ST表现
AryShiftReg	移位寄存器	FB		AryShiftReg_instance(Shift、 Reset、 In、 InOut、 Size);

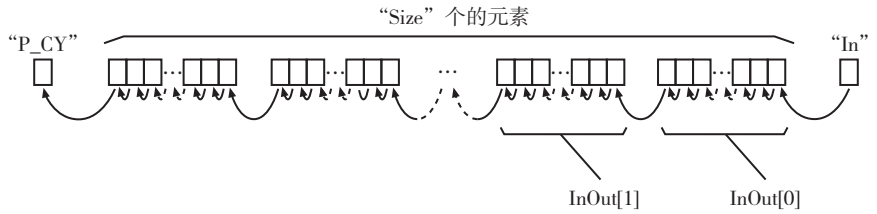
变量

	名称	输入/输出	内容	有效范围	单位	初始值
Shift	移位	输入	FALSE→TRUE时执行移位	遵从数据类型	–	FALSE
Reset	复位		TRUE：执行复位			
In	输入值		插入InOut[ ]的最低位的值			
Size	位串数组的元素数量		用于InOut[ ]内移位寄存器的元素数量			1
InOut[ ]数组	位串数组	输入输出	位串数组	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Shift	○																			
Reset	○																			
In	○																			
Size							○													
InOut[ ]数组	○	○	○	○	○															

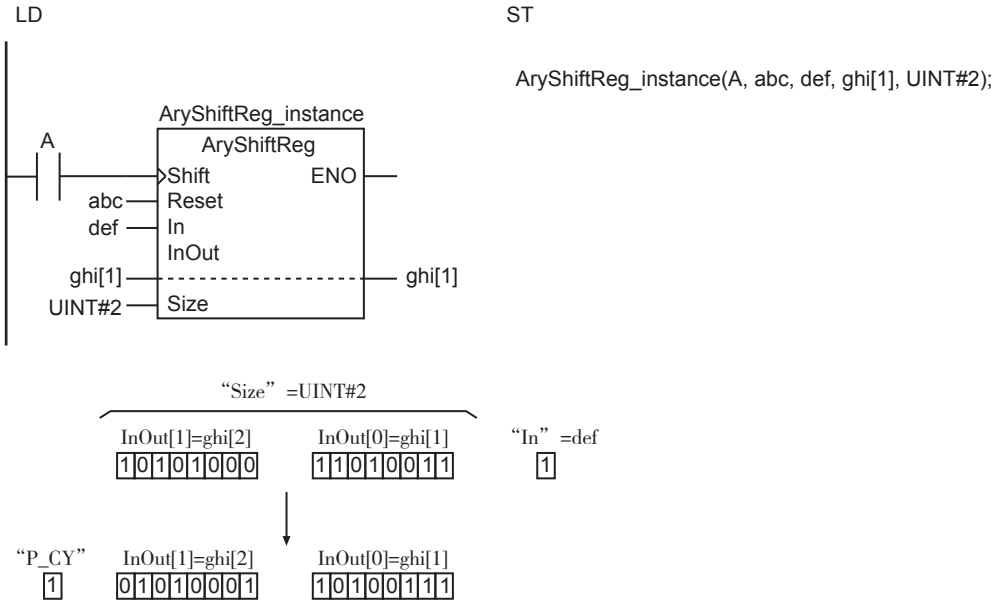
功能

“Shift”为FALSE→TRUE时，将从位串数组InOut[ ]的InOut[0]开始的 “Size”个的数组元素向左(高位方向)移1位。
将输入值 “In”插入最低位。将溢出位串数组的最高位输出至进位(CY)标志 “P_CY”。



“Reset”为TRUE时，将FALSE设定至从InOut[0]开始的 “Size”个的元素的所有位与进位(CY)标志。

InOut[ ]为BYTE型数组、“Size”=UINT#2时的示例如下所示。



相关的系统定义变量

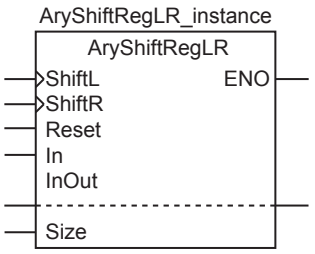
变量名称	名称	数据类型	内容
P_CY	进位(CY)标志	BOOL	进位标志中保存的值

使用注意事项

- “Reset”为TRUE时，即使 “Shift”由FALSE→TRUE，也不执行移位。
- “Shift”由FALSE→TRUE，移位动作正常进行或 “Reset”变为TRUE，复位动作正常进行时，ENO变为TRUE。
- “Size”的值为0时，InOut[ ]不变。
- 以下情况时会发生异常。ENO为FALSE，InOut[ ]不变。
  - “Size”的值超过InOut[ ]的数组区域时。

AryShiftRegLR

将数组元素组成的整个位串左移(或右移)1位，并将输入值插入最低位(或最高位)。

指令	名称	FB/FUN	图形表现	ST表现
AryShiftRegLR	左右移位寄存器	FB		AryShiftRegLR_instance(ShiftL、 ShiftR、 Reset、 In、 InOut、 Size);

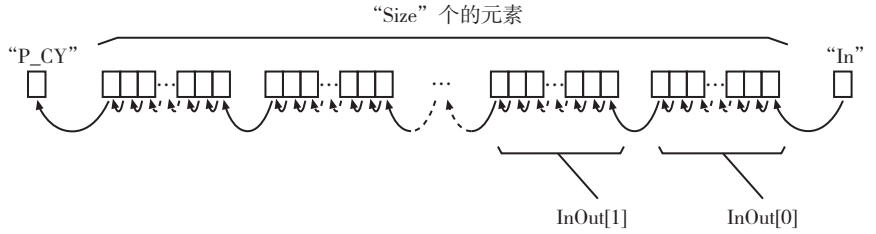
变量

	名称	输入/输出	内容	有效范围	单位	初始值
ShiftL	左移位	输入	FALSE→TRUE时执行左移位	遵从数据类型	–	FALSE
ShiftR	右移位		FALSE→TRUE时执行右移位			
Reset	复位		TRUE：执行复位			
In	输入值		插入InOut[ ]的最低位或最高位的值			
Size	位串数组的元素数量		用于InOut[ ]内移位寄存器的元素数量			1
InOut[ ]数组	位串数组	输入输出	位串数组	遵从数据类型	–	–

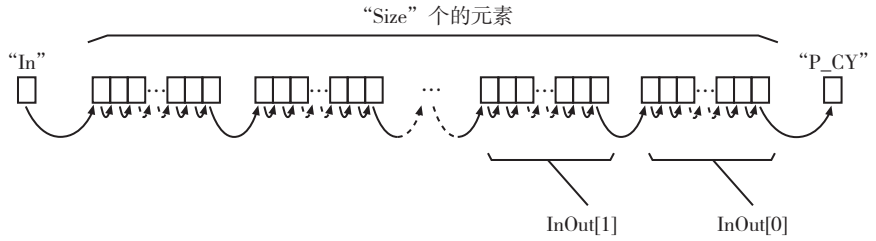
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
ShiftL	○																			
ShiftR	○																			
Reset	○																			
In	○																			
Size							○													
InOut[ ]数组	○	○	○	○	○															

功能

“ShiftL”由FALSE→TRUE时，将从位串数组InOut[ ]的InOut[0]开始的 “Size”个的数组元素向左移1位。将输入值 “In”插入最低位。将溢出位串数组的最高位输出至进位(CY)标志 “P_CY”。

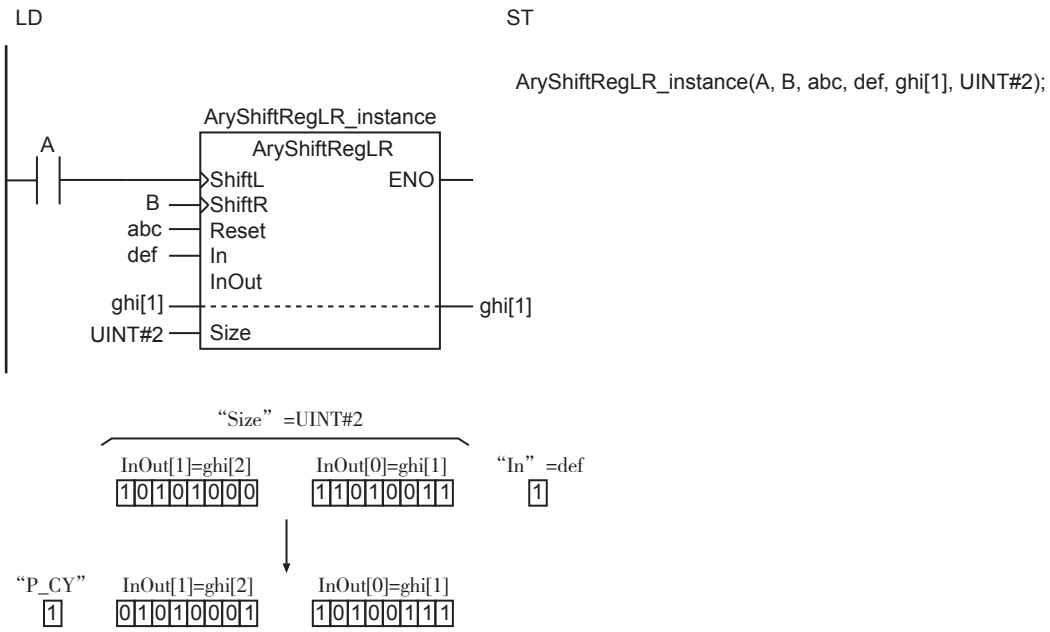


“ShiftR”由FALSE→TRUE时，向右移1位，将 “In”插入最高位。将溢出位串数组的最低位输出至进位(CY)标志 “P_CY”。



“Reset”为TRUE时，将FALSE设定至InOut[0]以后的 “Size”个的元素的所有位与 “P_CY”。

“InOut”为BYTE型数组、“Size”=UINT#2、“ShiftL”由FALSE→TRUE时的示例如下所示。



相关的系统定义变量

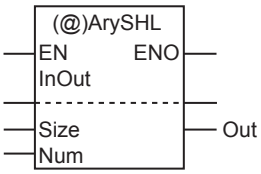
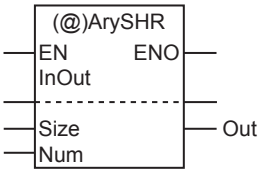
变量名称	名称	数据类型	内容
P_CY	进位(CY)标志	BOOL	进位标志中保存的值

使用注意事项

- “Reset”为TRUE时，即使 “ShiftL”及 “ShiftR”由FALSE→TRUE，也不执行移位。
- “ShiftL”及 “ShiftR”同时由FALSE→TRUE时，不执行移位。
- “Shift”由FALSE→TRUE，移位动作正常进行或 “Reset”变为TRUE，复位动作正常进行时，ENO变为TRUE。
- “Size”的值为0时，InOut[ ]不变。
- 以下情况时会发生异常。ENO为FALSE，InOut[ ]不变。
  - “Size”的值超过InOut[ ]的数组区域时。

ArySHL/ArySHR

对多个数组元素进行移位。
ArySHL：向左(高位方向)移位。
ArySHR：向右(低位方向)移位。

指令	名称	FB/FUN	图形表现	ST表现
ArySHL	数组的N元素左移位	FUN		ArySHL(InOut、Size、Num);
ArySHR	数组右移N个元素	FUN		ArySHR(InOut、Size、Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Size	移位对象的元素数量	输入	移位对象的元素数量	遵从数据类型	–	1
Num	移位元素数量		移位元素数量			
InOut[ ]数组	移位对象数组	输入输出	移位对象数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Size							○													
Num							○													
InOut[ ]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定将结构体作为元素的数组																			
Out	○																			

功能

针对移位对象数组InOut[ ]的高位 “Size”个的元素，进行移位元素数量 “Num”个的移位。
通过移位删除溢出值。
此外，将InOut[ ]的数据类型的初始值保存至备用元素。给InOut[ ]设定初始值属性时，初始化为该值。未设定初始值属性时，变为各数据类型默认的初始值。InOut[ ]为将结构体作为元素的数组时，使各元素结构体的所有结构要素初始化。

各数据类型默认的初始值如下表所示。

数据类型	初始值
BOOL	FALSE
BYTE、WORD、DWORD、LWORD	16#0
USINT、UINT、UDINT、ULINT、SINT、INT、DINT、LINT、REAL、LREAL	0
TIME	T#0ms
DATE	D#1970-1-1
TOD	TOD#0:0:0
DT	DT#1970-1-1-0:0:0
STRING	''

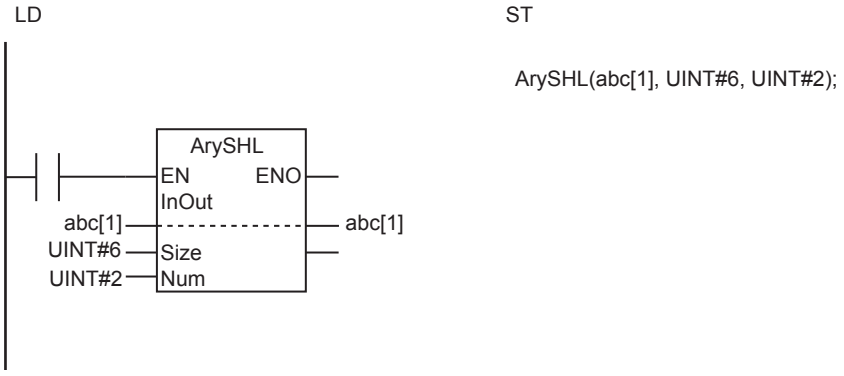
- **ArySHL**

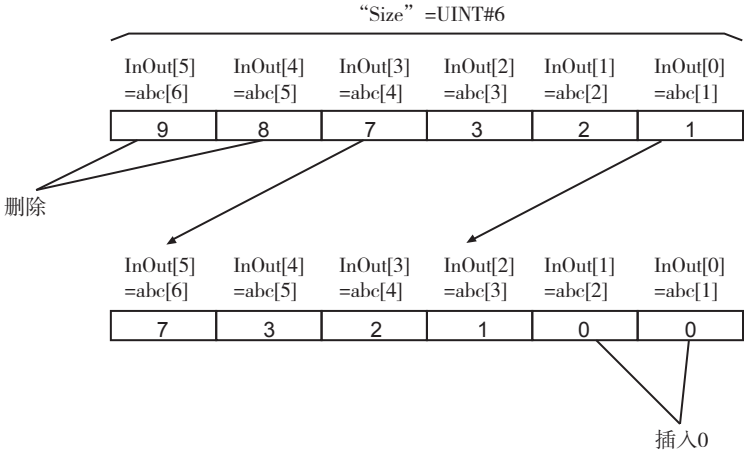
向左(数组的高位方向)移位。

- **ArySHR**

向右(数组的低位方向)移位。

ArySHL指令下，“Size”=UINT#6、“Num”=UINT#2时的示例如下所示。





参考

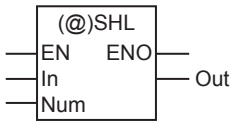
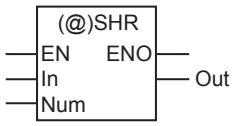
InOut[ ]为BOOL型时，与将 “Size”位的位串移 “Num”位相同。

使用注意事项

- “Num”的值为0时，不进行移位动作。
- “Num”的值大于 “Size”时，从InOut[0]开始，使InOut[“Size”–1]的所有值初始化。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，InOut[ ]不变。
  - “Size”的值超过InOut[ ]的数组区域时。

SHL/SHR

对位串进行多位移位。
SHL：向左(高位方向)移位。
SHR：向右(低位方向)移位。

指令	名称	FB/FUN	图形表现	ST表现
SHL	左移N位	FUN		Out:=SHL(In、Num);
SHR	右移N位	FUN		Out:=SHR(In、Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	移位对象	输入	移位对象	遵从数据类型	–	*1
Num*2	移位数量		移位位数	0～“In”的位数	位	1
Out	处理结果	输出	处理结果	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Num缩记为N。例如，可记述为Out：=SHL(In：=BYTE#16#89、N：=ULINT#2);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Num						○ *1			○ *1											
Out	与“In”相同的数据类型																			

*1 CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时，为ULINT型。CPU单元Ver.1.01以下或Sysmac Studio Ver.1.02以下时，为USINT型。

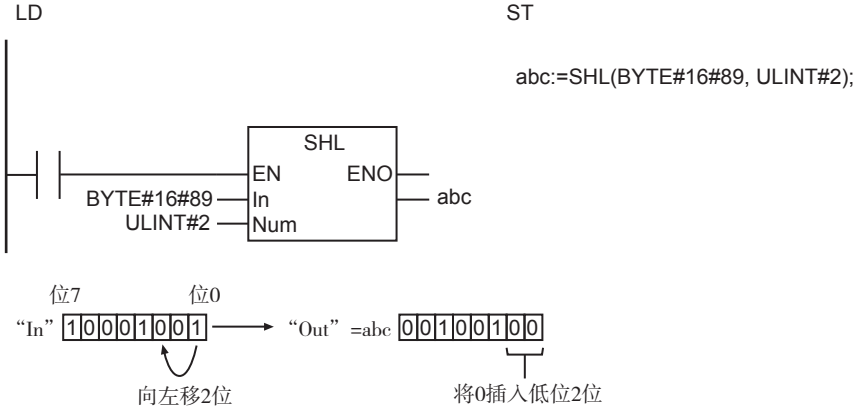
功能

使位串型的移位对象“In”的移位量仅为移位数量“Num”的位数。删除溢出位，从相反侧向插入位添零。

- SHL

从右向左(从低位到高位)移位。

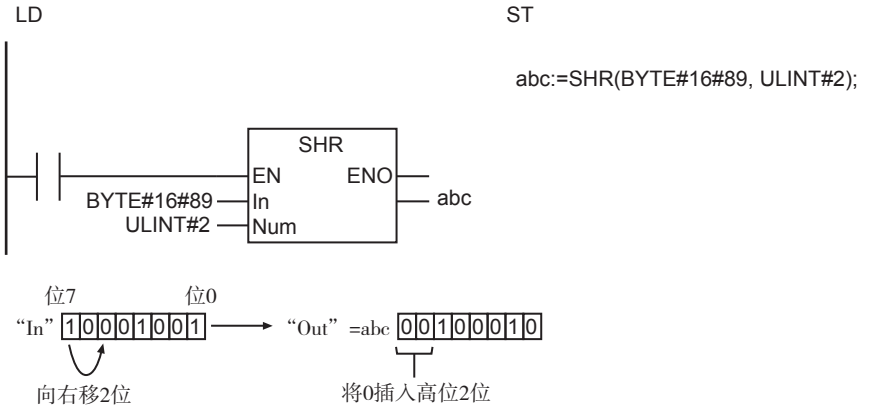
“In”=BYTE#16#89、“Num”=ULINT#2时的示例如下所示。



- SHR

从左向右(从高位到低位)移位。

“In”=BYTE#16#89、“Num”=ULINT#2时的示例如下所示。



参考

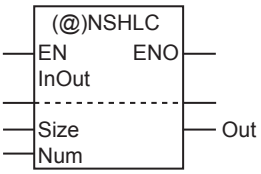
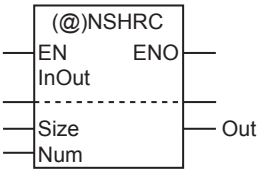
ROL/ROR指令将溢出位的值从相反侧插入。

使用注意事项

- 请将 “In”与 “Out”的数据类型设为相同。
- “Num”的值为0时，不会发生异常，将 “In”的值直接代入 “Out”。
- “Num”的值大于 “In”的位数时，不会发生异常，“Out”的值变为16#0。

NSHLC/NSHRC

带进位(CY)标志，对位串数组进行多位移位。
NSHLC：向左(高位方向)移位。
NSHRC：向右(低位方向)移位。

指令	名称	FB/FUN	图形表现	ST表现
NSHLC	N位数据带CY左移N位	FUN		NSHLC(InOut、Size、Num);
NSHRC	N位数据带CY右移N位	FUN		NSHRC(InOut、Size、Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Size	移位对象的位数	输入	移位对象的位数	遵从数据类型	位	1
Num	移位位数		移位位数			
InOut[ ]数组	移位对象数组	输入输出	移位对象的位串数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Size						○														
Num						○														
InOut[ ]数组	○	○	○	○	○															
Out	○																			

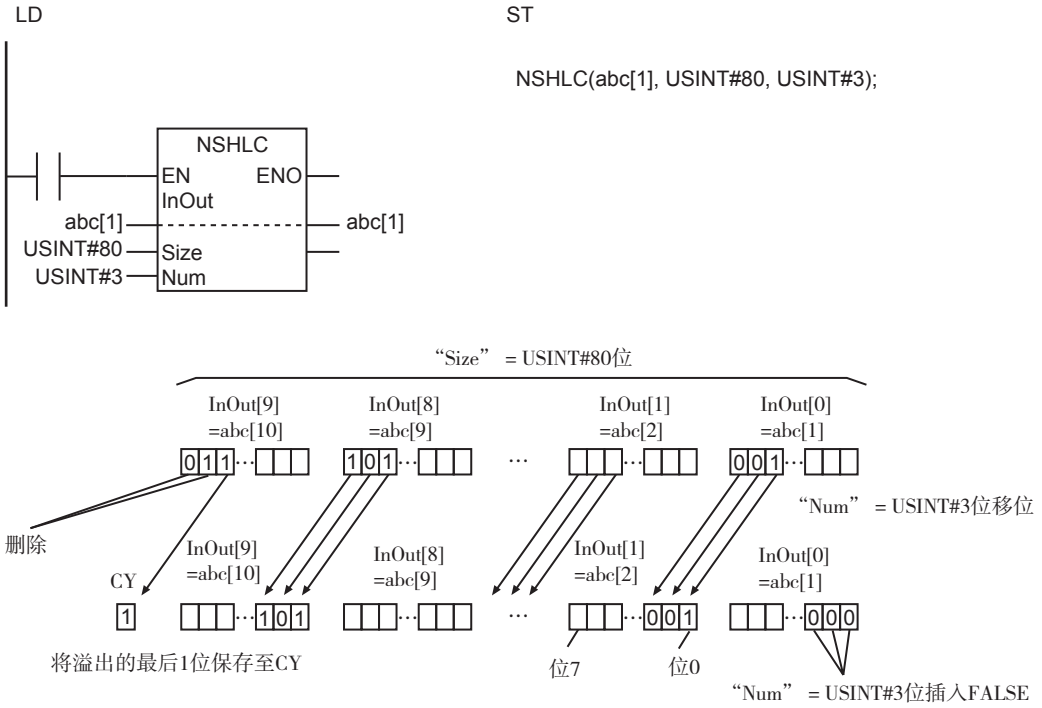
功能

针对从移位对象数组InOut[ ]的InOut[0]开始的 “Size”个的数组元素，进行 “Num”位移位。将溢出位的最后值输出至进位(CY)标志。从相反侧向插入位添零。

- NSHLC
  向左(从数组的低位到高位)移位。

- NSHRC
  向右(从数组的高位到低位)移位。

NSHLC指令下，InOut[ ]为BYTE型数组、“Size”=USINT#80、“Num”=USINT#3时的示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
P_CY	进位(CY)标志	BOOL	进位标志中保存的值

使用注意事项

- “Num”的值为0时，不进行移位动作。
- “Num”的值大于 “Size”时，从InOut[0]的位0开始，“Size”位的值变为FALSE。此外，进位(CY)标志的值同样变为FALSE。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，InOut[ ]不变。
  - “Size”的值超过InOut[ ]的数组区域时。

ROL/ROR

对位串进行多位旋转。
ROL：向左(高位方向)旋转。
ROR：向右(低位方向)旋转。

指令	名称	FB/FUN	图形表现	ST表现
ROL	左旋转N位	FUN	(@)ROL EN ENO In Out Num	Out:=ROL(In、Num);
ROR	右旋转N位	FUN	(@)ROR EN ENO In Out Num	Out:=ROR(In、Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	旋转对象	输入	旋转对象	遵从数据类型	–	*1
Num*2	旋转数量		旋转位数	0～“In”的位数	位	1
Out	处理结果	输出	处理结果	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 在Ver.1.03以上的Sysmac Studio中，进行ST表现以明确表示变量名称和参数名称之间的对应关系时，可将Num缩记为N。例如，可记述为Out：=ROL(In：=BYTE#16#89、N：=ULINT#2);。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Num						○ *1			○ *1											
Out	与“In”相同的数据类型																			

*1 CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时，为ULINT型。CPU单元Ver.1.01以下或Sysmac Studio Ver.1.02以下时，为USINT型。

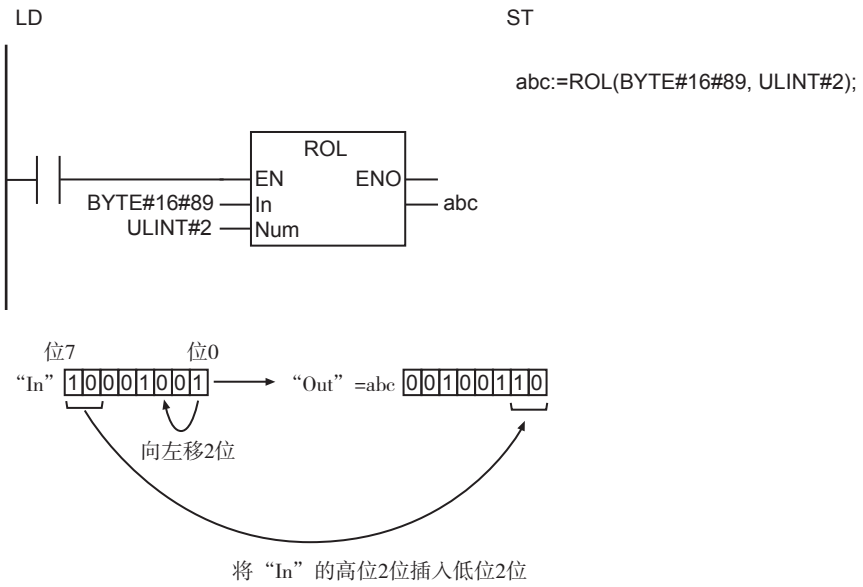
功能

使位串型的旋转对象 “In”的旋转量仅为旋转数量 “Num”的位数。从相反侧插入溢出位。

- ROL

从右向左(从低位到高位)旋转。

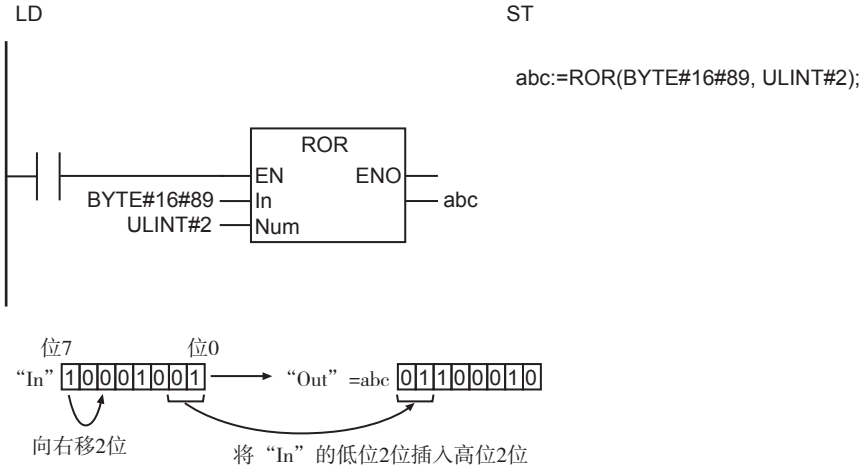
“In”=BYTE#16#89、“Num”=ULINT#2时的示例如下所示。



- ROR

从左向右(从高位到低位)旋转。

“In”=BYTE#16#89、“Num”=ULINT#2时的示例如下所示。



参考

SHL/SHR指令删除溢出位，从相反侧向插入位添零。

使用注意事项

- 请将 “In”与 “Out”的数据类型设为相同。
- “Num”的值为0时，不会发生异常，将 “In”的值直接代入 “Out”。
- “Num”的值超出 “In”的位数时，不会发生异常，仅 “Num”的值旋转。例如，“In”为WORD型、“Num”的值为USINT#1与USINT#17时，“Out”的值相同。

数据转换指令

Swap

更换16位值的高位字节和低位字节。

指令	名称	FB/FUN	图形表现	ST表现
Swap	字节交换	FUN	(@)Swap EN　ENO In　Out	Out:=Swap(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0
Out	转换结果	输出	转换结果	遵从数据类型	–	–

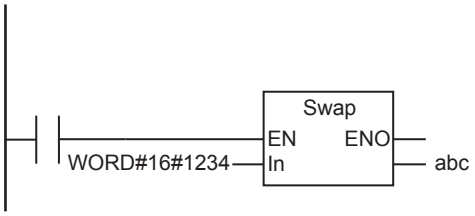
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In			○																	
Out			○																	

功能

更换转换对象 “In”的高位字节和低位字节，代入转换结果 “Out”。

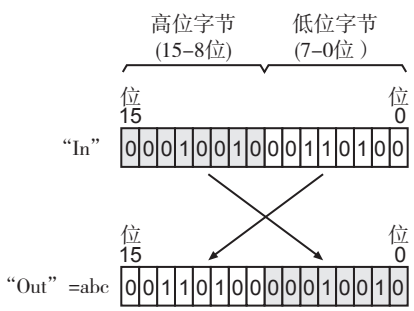
“In”=WORD#16#1234时的示例如下所示。

LD



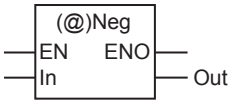
ST

```
abc:=Swap(WORD#16#1234);
```



Neg

对数值的符号进行取反。

指令	名称	FB/FUN	图形表现	ST表现
Neg	符号取反	FUN		Out:=Neg(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	(*)
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○	○	○					
Out						○	○	○	○	○	○	○	○	○	○					

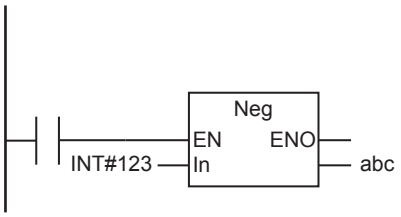
功能

对转换对象 “In”的符号进行取反。
如下所示，处理内容因 “In”的数据类型而异。

“In”的数据类型	“Out”的值
带符号整数 SINT，INT，DINT，LINT	对 “In”的所有位进行取反后加1 (与 “In” ×(–1)意义相同)
无符号整数 USINT，UINT，UDINT，ULINT	对 “In”的所有位进行取反后加1
实数 REAL，LREAL	“In” ×(–1)

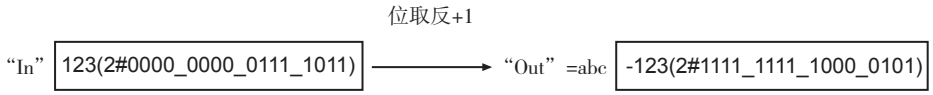
“In”=INT#123时的示例如下所示。

LD

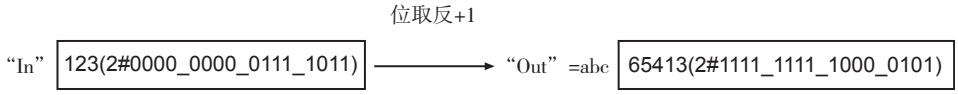


ST

```
abc:=Neg(INT#123);
```



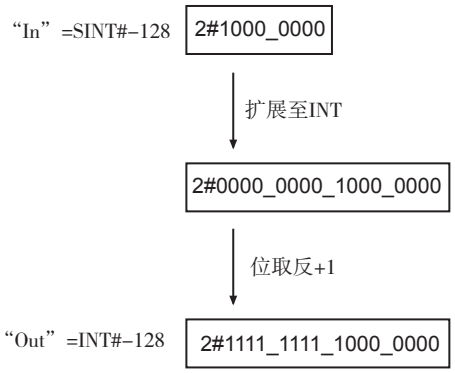
“In”=UINT#123时的示例如下所示。



使用注意事项

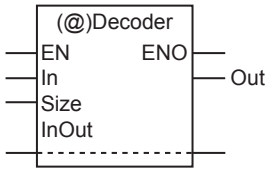
“In”和 “Out”的数据类型不同时，请将 “Out”的有效范围设为包含 “In”的有效范围。否则，虽不会发生异常，但 “Out”的值为错误值。

例如，“In”的值为SINT#–128，“Out”的数据类型为INT时，“Out的值不是INT#128，而是INT#–128。



Decoder

将最大256位组成的数组元素指定的1位设置为TRUE，其它位设置为FALSE。

指令	名称	FB/FUN	图形表现	ST表现
Decoder	位译码器	FUN		Decoder(In, Size, InOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换位位置	输入	待转换的位位置	遵从数据类型	–	0
Size	转换位数		待转换的位数	0 ~ 8	位	1
InOut[ ]数组	转换对象数组	输入输出	转换对象数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

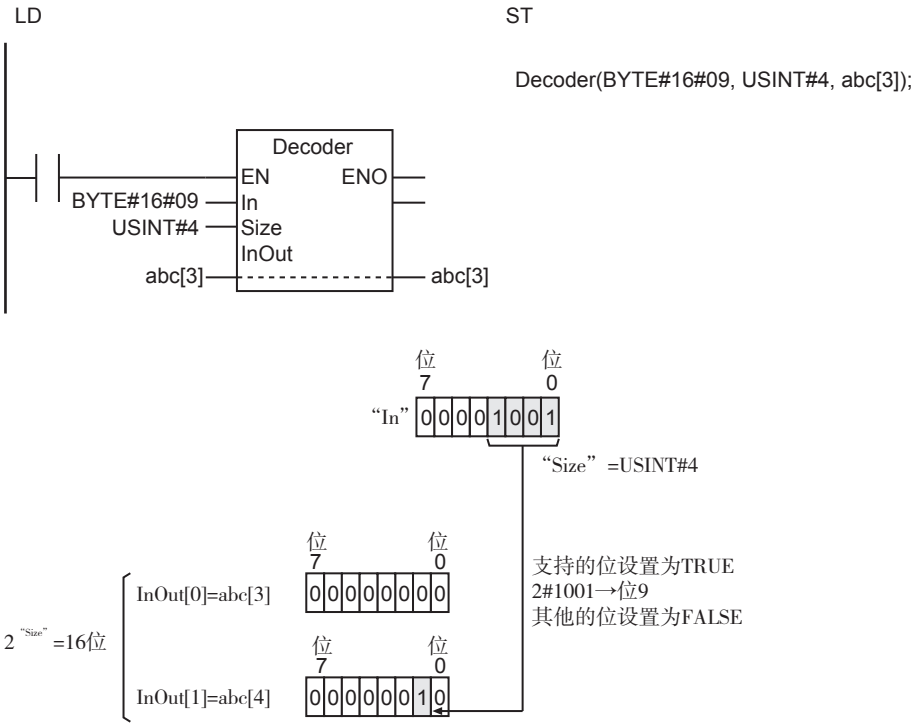
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○																		
Size						○														
InOut[ ]数组	○	○	○	○	○															
Out	○																			

功能

将转换对象数组InOut[ ]从InOut[0]开始2“Size”位大小的数组元素的指定位设为TRUE。其他位设为FALSE。
值为TRUE的位位置由转换位位置 “In”的低位 “Size”位指定。
传输至InOut[ ]的输入输出参数务必也像array[3]那样加上元素编号后再指定。

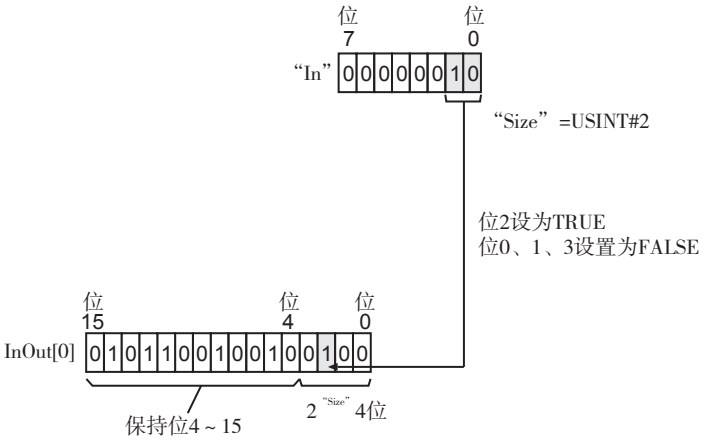
以 “In”=BYTE#16#09，“Size”=USINT#4，InOut[ ]为BYTE型数组为例进行说明。
“In”的低位 “Size”位的值16#9转换为10进制后为9。因此，从InOut[ ]的最低位起的第9位为TRUE，其他位为FALSE。

InOut[ ]是BYTE型数组，从最低位起的第9位是InOut[1]的位1。因此，InOut[1]的位1为TRUE，InOut[0]的所有位和InOut[1]的位1以外的位为FALSE。



InOut[ ]的元素的位数大于以 “Size”表示的位数时，保持剩余位的值。以 “In”=BYTE#16#02，“Size”=USINT#2，InOut[ ]为WORD型数组为例进行说明。

“Size”=USINT#2，因此InOut[0]的低位4位设置值。保持InOut[0]的剩余位4 ~ 15的值。



参考

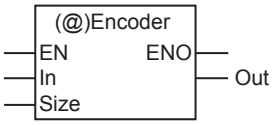
在最多由256位组成的数组元素中，计算TRUE的位位置时，请使用 “Encoder指令(P.2–402)”。

使用注意事项

- “Size”的值为0时，InOut[ ]的所有位为FALSE。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，InOut[ ]不变。
  - “Size”超过有效范围时。
  - 2“Size”的值超过InOut[ ]的数组元素的位数时。

Encoder

计算最大256位组成的数组元素中，值为TRUE的位位置。

指令	名称	FB/FUN	图形表现	ST表现
Encoder	位编码器	FUN		Out:=Encoder(In, Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	转换对象数组	输入	转换对象数组	遵从数据类型	–	(*)
Size	转换位数		待转换的位数	0 ~ 8	位	1
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组	○	○	○	○	○															
Size						○														
Out		○																		

功能

计算转换对象数组In[ ]在任意范围中值为TRUE的位位置。

在以In[0]开始的2“Size”位的范围中计算位位置。以2进制表示此范围中值为TRUE的位位置，在转换结果“Out”的低位 “Size”位中保存。“Out”的剩余位中保存FALSE。

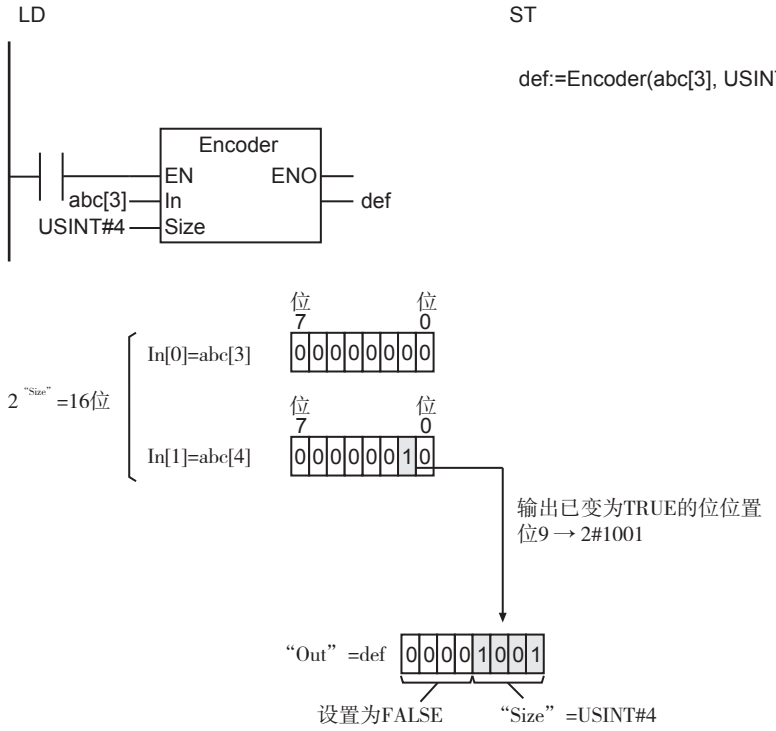
如果指定范围内有多个值为TRUE的位，则在这些位中计算最高位的位位置。

传输至In[ ]的输入参数务必也像array[3]那样加上元素编号后再指定。

表示 “Size”=USINT#4，In[ ]为BYTE型数组时的示例。

“Size”=USINT#4，因此从In[0]开始的2^4=16位是计算TRUE的位位置的对象范围。下图中，该范围的第9位有值为TRUE的位。

“Size”=USINT#4，因此在 “Out”的低位4位中保存已计算的9=2#1001。“Out”的高位4位中保存FALSE。



参考

在最多由256位组成的数组元素中，将1位设置为TRUE，其他位设置为FALSE时，请使用 “Decoder指令(P.2–399)”。

使用注意事项

- “Size”的值为0时，“Out”的所有位为FALSE。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”超过有效范围时。
  - 2 “Size”的值超过In[ ]的数组元素的位数时。
  - In[ ]的位中，“Size”指定的所有的值为FALSE时。

BitCnt

对位串内的值为TRUE的位的总数进行计数。

指令	名称	FB/FUN	图形表现	ST表现
BitCnt	位计数	FUN	(@)BitCnt EN ENO In Out	Out:=BitCnt(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	计数对象	输入	对值为TRUE的位进行计数的对象	遵从数据类型	–	(*)
Out	计数结果	输出	值为TRUE的位数	0～“In”的位数	–	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

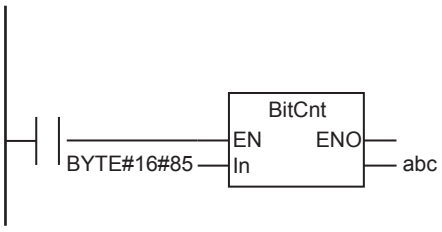
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out						○														

功能

对计数对象 “In”中值为TRUE的位的总数进行计数。

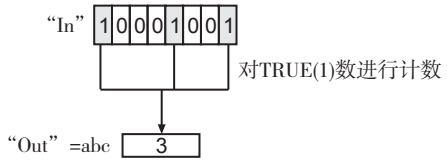
“In”为BYTE型，值为BYTE#16#85时的示例如下所示。

LD



ST

```
abc:=BitCnt(BYTE#16#85);
```



ColmToLine_**

提取各数组元素指定位置的位的值，并作为位串输出。

指令	名称	FB/FUN	图形表现	ST表现
ColmToLine_**	位串→位行转换组	FUN	(@)ColmToLine_** EN ENO In Out Size Pos **为位串的数据类型名称	Out:=ColmToLine_**(In, Size, Pos); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	转换对象数组	输入	转换对象数组	遵从数据类型	–	(*)
Size	转换对象的元素数		In[ ]中作为转换对象的元素数	0 ~ “Out”的位数		1
Pos	转换位位置		待转换的位位置	0 ~ In[ ]的位数−1		0
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○	○	○	○															
Size						○														
Pos						○														
Out		○	○	○	○															

功能

提取数组各元素指定位置的位的值，将其排列后作为位串输出。

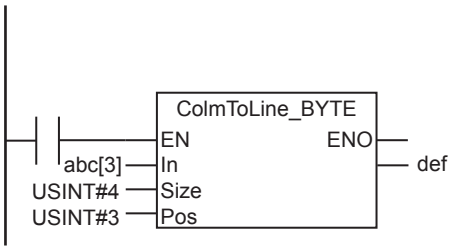
首先，提取转换对象数组In[ ]从In[0]开始的 “Size”个元素。然后，仅提取上述元素的第 “Pos”位的值。将其排列成 “Size”位的位串，并保存在转换结果 “Out”的最低位的位位置。“Out”的剩余位中保存FALSE。

指令名称因 “Out”的数据类型而异。例如，“Out”为BYTE型时，指令名称为ColmToLine_BYTE。

传输至In[ ]的输入参数务必也像array[3]那样加上元素编号后再指定。

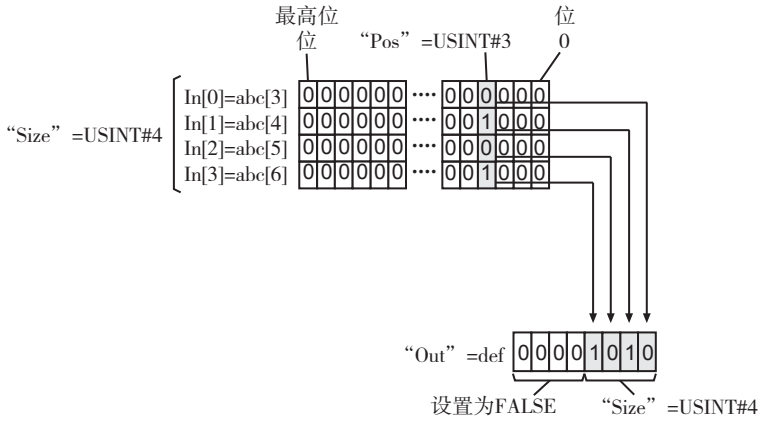
ColmToLine_BYTE指令下，“Pos”=USINT#3，“Size”=USINT#4时的示例如下所示。

LD



ST

```
def:=ColmToLine_BYTE(abc[3], USINT#4, USINT#3);
```



参考

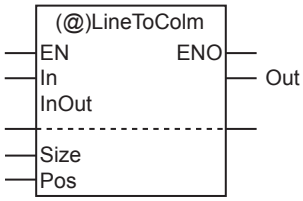
将位串输出至各数组元素指定的位位置时，请使用 “LineToColm指令(P.2–407)”。

使用注意事项

- “Size”的值为0时，“Out”的所有位为FALSE。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过有效范围时。
  - “Pos”的值超过有效范围时。
  - “Size”的值超过In[ ]的数组区域时。

LineToColm

分解位串，输出至数组元素指定的位位置。

指令	名称	FB/FUN	图形表现	ST表现
LineToColm	位行→位串转换	FUN		LineToColm(In, InOut, Size, Pos);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	(*)
Size	转换结果的元素数		转换结果的元素数	0 ~ “In”的位数		1
Pos	转换位位置		待转换的位位置	0 ~ InOut[ ]的位数–1		0
InOut[ ]数组	转换结果数组	输入输出	转换结果	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

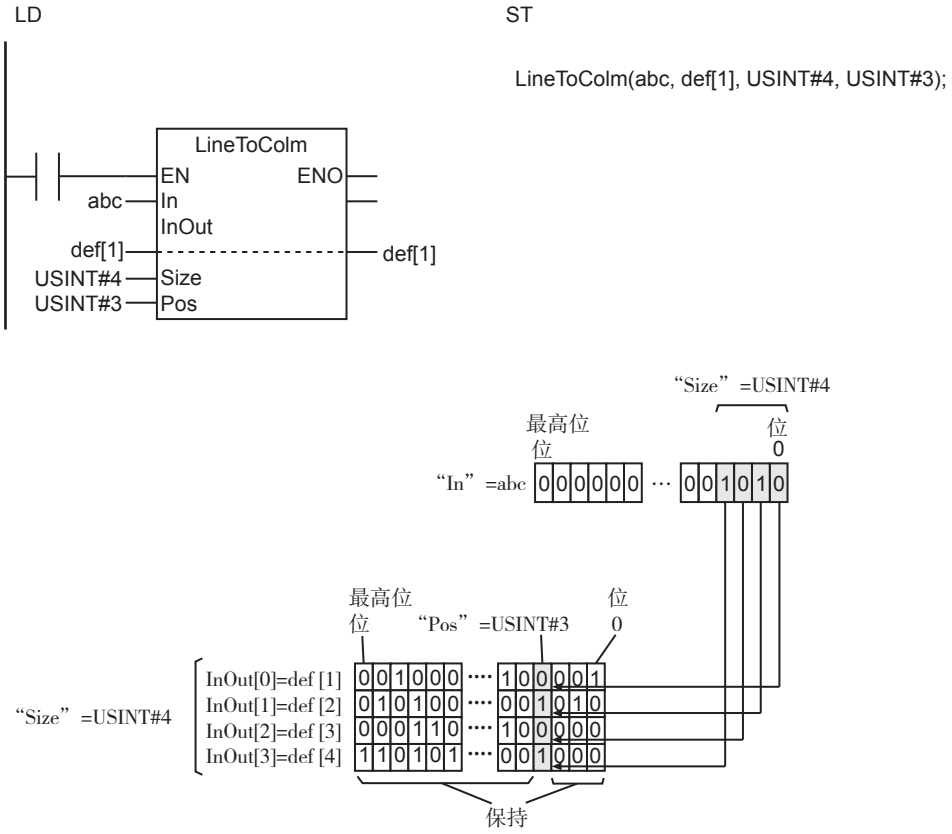
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Size						○														
Pos						○														
InOut[ ]数组		○	○	○	○															
Out	○																			

功能

分解位串，输出至数组元素指定的位位置。

首先，从转换对象 “In”的最低位提取 “Size”位，并分解成每个位。
然后，将其保存在转换结果数组 InOut[ ] 从 InOut[0] 开始的各元素的第 “Pos”位中。保存的数组元素有 “Size”个。
未保存值的位，进行值的保持。

“Pos”=USINT#3、“Size”=USINT#4时的示例如下所示。



参考

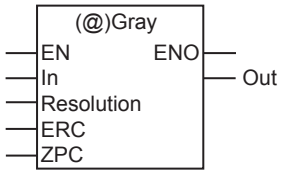
提取各数组元素指定位置的位的值，输出至位串时，请使用 “ColmToLine_**指令(P.2–405)”。

使用注意事项

- “Size”的值为0时，InOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，InOut[ ]不变。
  - “Size”的值超过有效范围时。
  - “Pos”的值超过有效范围时。
  - “Size”的值超过InOut[ ]的数组区域时。

Gray

将格雷码转换为角度。

指令	名称	FB/FUN	图形表现	ST表现
Gray	格雷码转换	FUN		Out:=Gray(In, Resolution, ERC, ZPC);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象的格雷码	遵从数据类型	–	0
Resolution	分辨率		分辨率	_R256, _R1B ~ _R15B, _R360, _R720, _R1024		_R256
ERC	编码器余数补偿值		编码器余数补偿值	0 ~ “Resolution”的分辨率		0
ZPC	原点补偿值		原点补偿值			
Out	转换结果	输出	转换结果	(*)	°	–

* 0 ~ 3.59999999999999e+2

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In			○																	
Resolution	枚举体_eGRY_RESOLUTION　枚举元素参阅功能说明																			
ERC							○													
ZPC							○													
Out															○					

功能

将格雷码表现的旋转编码器的输出值 “In”转换为角度值。转换结果 “Out”的单位为° 。

“Resolution”的数据类型为枚举体_eGRY_RESOLUTION。枚举元素的含义如下所示。

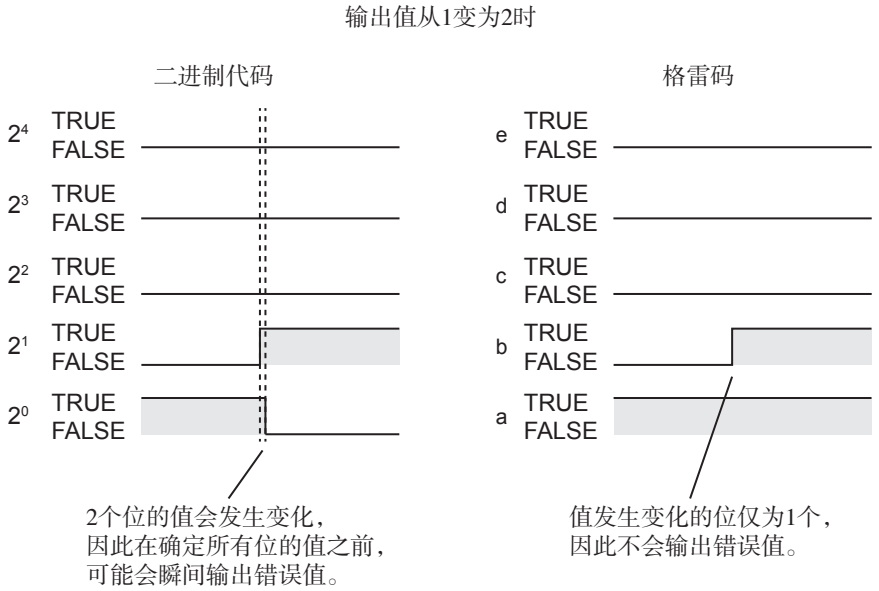
枚举元素	含义
_R256	256
_R1B	1位(2)
_R2B	2位(4)
_R3B	3位(8)
_R4B	4位(16)
_R5B	5位(32)
_R6B	6位(64)
_R7B	7位(128)
_R8B	8位(256)
_R9B	9位(512)
_R10B	10位(1024)
_R11B	11位(2048)
_R12B	12位(4096)
_R13B	13位(8192)
_R14B	14位(16384)
_R15B	15位(32768)
_R360	360
_R720	720
_R1024	1024

格雷码

格雷码是一种名为交替二进制代码的2进制数。其特征在于，像0和1、1和2那样，差为1的数值的代码必定有1个位的值不同。格雷码用于绝对值编码器的输出等用途。
以下列出了4位的二进制代码和格雷码。

10进制	二进制代码				格雷码			
	2^3	2^2	2^1	2^0	d	c	b	a
0	0	0	0	0	0	0	0	0
1	0	0	0	1	0	0	0	1
2	0	0	1	0	0	0	1	1
3	0	0	1	1	0	0	1	0
4	0	1	0	0	0	1	1	0
5	0	1	0	1	0	1	1	1
6	0	1	1	0	0	1	0	1
7	0	1	1	1	0	1	0	0
8	1	0	0	0	1	1	0	0
9	1	0	0	1	1	1	0	1
10	1	0	1	0	1	1	1	1
11	1	0	1	1	1	1	1	0
12	1	1	0	0	1	0	1	0
13	1	1	0	1	1	0	1	1
14	1	1	1	0	1	0	0	1
15	1	1	1	1	1	0	0	0

如果使用格雷码，编码器的输出值增加或减少1时，仅有1个位的值发生变化，因此可防止瞬间输出错误的值。因格雷码和二进制代码的差异导致的编码器输出值变化的差异如下所示。

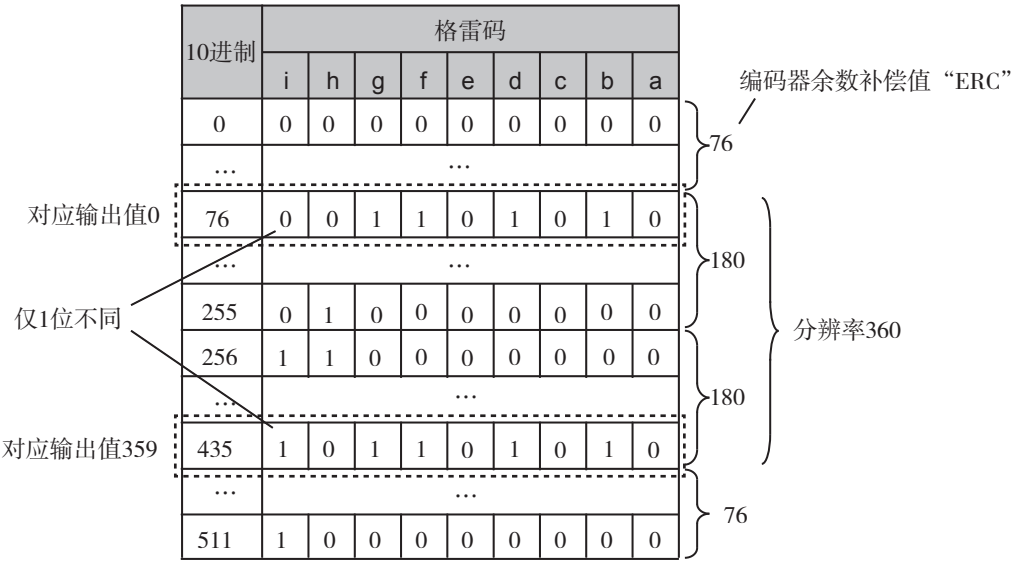


编码器余数补偿值 “ERC”

ERC”为指定格雷码范围的补偿值，编码器的分辨率不是2的乘幂时，以使相对于编码器输出最大值和最小值的格雷码的差异仅有1位。

例如，使用分辨率为360的绝对值编码器。使用9位格雷码。9位可表现的范围为0 ~ 511。此时，自0 ~ 511的中心向前后各 180 的范围，即在 76 ~ 435 的范围内使用格雷码。因此，表示输出值为 0 时的格雷码为001101010(10进制为76)，表示输出值为359时的格雷码为101101010(10进制为435)，两者相差仅1位。

此时，编码器余数补偿值 “ERC”的值为76。



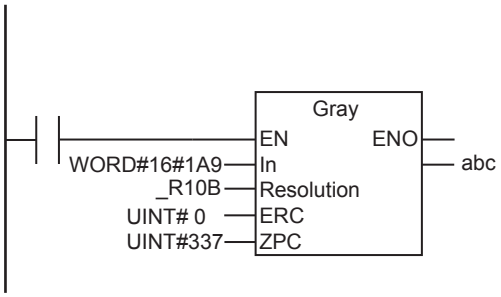
原点补偿值 “ZPC”

移动旋转编码器的原点角度时，需设定 “ZPC”。例如，将分辨率为256的旋转编码器的原点移动90°时，“ZPC”的值设为256 × (90/360)=64。

记述示例

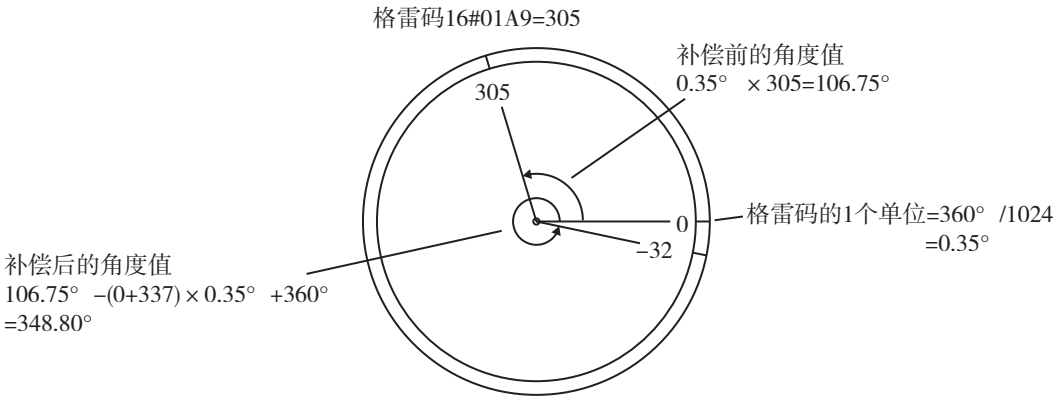
“In”=WORD#16#1A9，“Resolution”=_R10B，“ERC”=UINT#0，“ZPC”=UINT#337时的示例如下所示。
首先，分辨率为10位，因此格雷码的1个单位为360° /1024=0.35° 。
格雷码16#01A9相应的10进制数的值为305。因此，补偿前的角度值为0.35° × 305=106.75° 。
“ERC”的值为0，“ZPC”的值为337。因此，对其补偿后的角度值为106.75° −(0+337) × 0.35° =−11.20° 。
“Out”的值的范围为大于0，因此为−11.20° +360° =348.80° 。“Out”的值为LREAL#348.8。

LD



ST

```
abc:=Gray(WORD#16#1A9, _R10B, UINT#0, UINT#337);
```



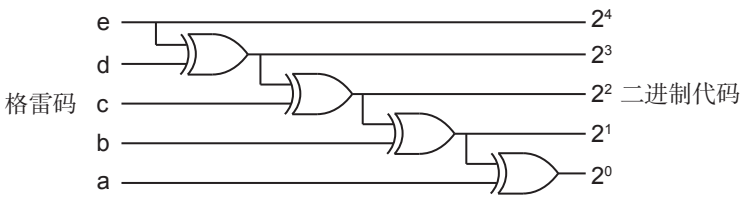
参考

请参阅所使用的旋转编码器的使用说明书等后再确定为 “Resolution”和 “ERC”指定的值。

从格雷码到二进制代码的转换

可通过以下处理进行从格雷码到二进制代码的转换。图中的逻辑符号表示异或。

5位时的转换电路



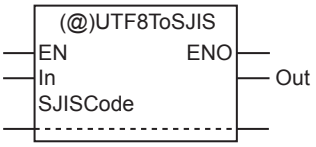
使用注意事项

以下情况时会发生异常。“ENO”变为FALSE，“Out”不变。

- “Resolution”的值超过有效范围时。
- “ERC”的值大于 “Resolution”指定的分辨率时。
- “ZPC”的值大于 “Resolution”指定的分辨率时。
- 将 “In”转换为位串后的值小于 “ERC”的值时。
- 通过 “ERC”补偿位串后的值大于 “Resolution”指定的分辨率时。

UTF8ToSJIS

将文字代码UTF-8的字符串转换为文字代码SJIS的BYTE型数组。

指令	名称	FB/FUN	图形表现	ST表现
UTF8ToSJIS	字符代码转换(UTF-8→SJIS)	FUN		Out:=UTF8ToSJIS(In, SJISCode);

变量

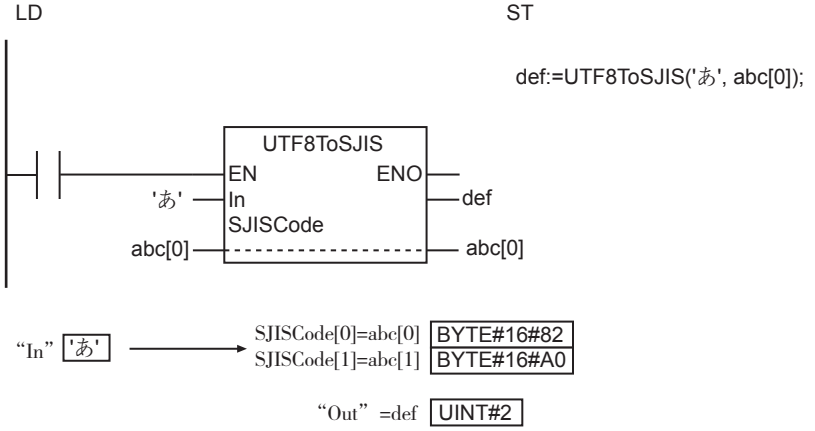
	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象字符串	输入	待转换的字符串	遵从数据类型	-	''
SJIS Code[ ] 数组	SJIS数组	输入输出	字符代码SJIS的数组	遵从数据类型	-	-
Out	转换元素数	输出	保存在SJISCode[ ]中的元素数	0 ~ 1985	-	-

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
SJIS Code[ ] 数组		○																		
Out							○													

功能

将字符代码UTF-8的转换对象字符串 “In”转换为字符代码SJIS的BYTE型数组SJISCode[ ]。将转换后的数据分割成每个字节，从SJISCode[0]开始依次保存。
转换元素数 “Out”中保存已保存在SJISCode[ ]中的转换后的数据的元素数。

“In” ='あ'时的示例如下所示。



使用注意事项

- 不转换 “In”结尾的NULL字符。也不对转换元素数进行计数。
- “In”为仅有NULL字符的字符串时，“Out”的值变为0，SJISCode[ ]不变。
- SJISCode[ ]的第 “Out”号之后的元素不变。例如，转换元素数为5时，SJISCode[5]之后的元素不变。
- 以下情况时会发生异常。“ENO”变为FALSE，“Out”、SJISCode[ ]不变。
  - 转换结果的元素数超过SJISCode[ ]连接的输出参数的范围大小时。
  - “In”的内容为无法转换的字符时。

版本相关信息

本指令可用于Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio。

SJISToUTF8

将文字代码SJIS的BYTE型数组转换为文字代码UTF-8的字符串。

指令	名称	FB/FUN	图形表现	ST表现
SJISToUTF8	字符代码转换(SJIS→UTF-8)	FUN	(@)SJISToUTF8 EN ENO In Out Size	Out:=SJISToUTF8(In, Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	转换对象SJIS数组	输入	待转换的字符代码SJIS的数组(*1)	遵从数据类型	-	(*2)
Size	SJIS数组元素数		待转换的In[ ]的元素数			-
Out	转换字符串	输出	转换为字符代码UTF-8的字符串	遵从数据类型	-	-

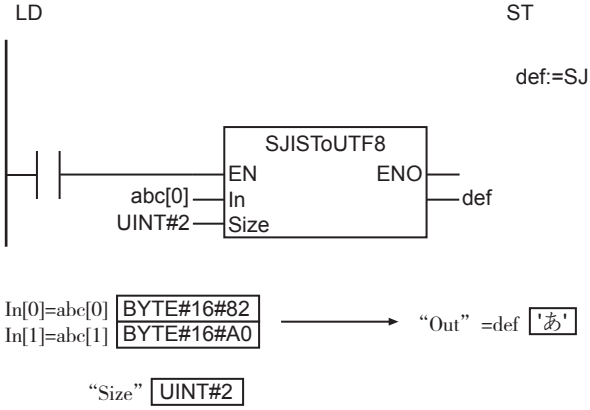
*1 包含NULL字符(BYTE#16#00)时的最大元素数为1986。不含NULL字符时的最大元素数为1985。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○																		
Size							○													
Out																				○

功能

将字符代码SJIS的BYTE型数组In[ ]的元素转换为字符代码UTF-8的字符串。
转换对象为In[0]之后的 “Size”个元素。但如果中途有NULL字符(BYTE#16#00)，则转换到该处为止。
转换后的字符串保存在转换字符串 “Out”中。“Out”的结尾带NULL字符。

In[0]=BYTE#16#82，In[1]=BYTE#16#A0，“Size”=UINT#2时的示例如下所示。



使用注意事项

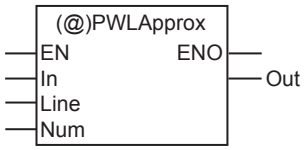
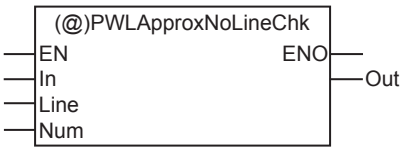
- “Size”的值为0时，“Out”的值仅带NULL字符。
- 以下情况时会发生异常。“ENO”变为FALSE，“Out”不变。
  - “Size”的值超过In[ ]的数组大小时。
  - In[ ]的内容为无法转换的字符时。

版本相关信息

本指令可用于Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio。

PWLApprox/ PWLApproxNoLineChk

对整数、实数进行折线近似计算。
PWLApprox ：检查折线数据是否有效。
PWLApproxNoLineChk：不检查折线数据是否有效。

指令	名称	FB/ FUN	图形表现	ST表现
PWLApprox	折线近似转换 (带折线检查)	FUN		Out:=PWLApprox(In, Line, Num);
PWLApprox NoLineChk	折线近似转换 (无折线检查)	FUN		Out:=PWLApproxNoLineChk(In, Line, Num);

变量

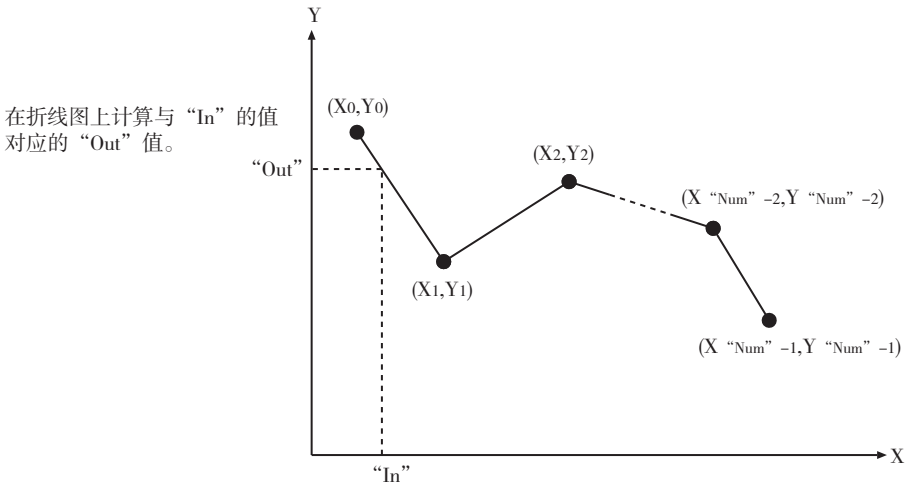
	名称	输入/ 输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	(*)
Line[ ] 数组	折线数据数组		折线数据数组			
Num	折线数据数		折线数据数			1
Out	转换结果	输出	转换结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○	○	○					
Line[ ] 数组	将与 “In”相同的数据类型作为元素的数组																			
Num							○													
Out						○	○	○	○	○	○	○	○	○	○					

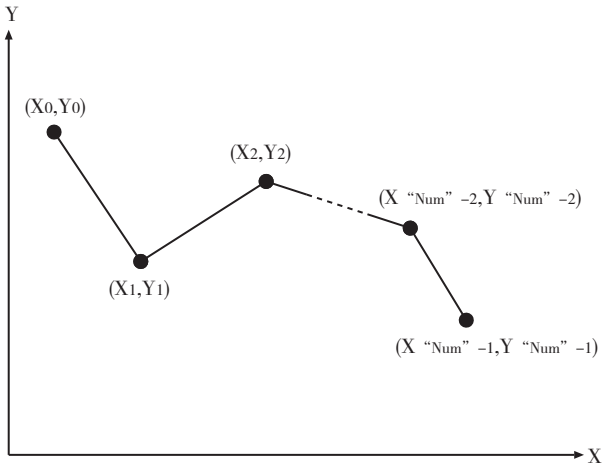
功能

根据折线数据数组Line[ ]的、Line[0,0]之后的(“Num” × 2)个元素组成的折线数据，对转换对象 “In”进行近似转换。
如图所示，将与折线数据上X坐标 “In”对应的Y坐标代入转换结果 “Out”。



折线数据Line[ ]的元素和折线数据数 “Num”

Line[ ] 为 2 维或 3 维数组，请将第 1 维的元素数设为 2。如图所示，请将折线数据的各点坐标值 (X_0,Y_0)、(X_1,Y_1)……作为Line[ ]的各元素。折线数据数 “Num”为用于折线近似计算的Line[ ]的元素数的1/2。



Line[ ]为2维数组时

Line[0,0]	X0
Line[0,1]	Y0
Line[1,0]	X1
Line[1,1]	Y1
Line[2,0]	X2
Line[2,1]	Y2
:	:
Line[“Num” −1,0]	X “Num” −1
Line[“Num” −1,1]	Y “Num” −1

Line[ ]为3维时

Line[0,0,0]	X0
Line[0,0,1]	Y0
Line[0,1,0]	X1
Line[0,1,1]	Y1
Line[0,2,0]	X2
Line[0,2,1]	Y2
:	:
Line[0, “Num” −1,0]	X “Num” −1
Line[0, “Num” −1,1]	Y “Num” −1

记述示例

根据元素数为4的折线数据数组abc[ ]，对 “In”=LREAL#3.0进行近似转换时的示例如下所示。
“Num”=UINT#4，abc[ ]的元素值如下所示。

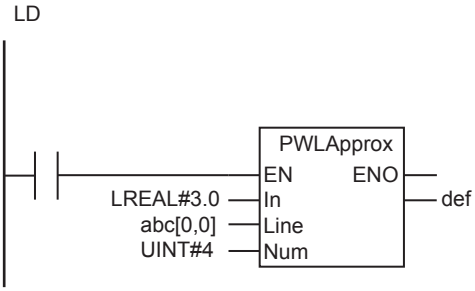
abc[0.0]=X_0=LREAL#1.0、abc[0,1]=Y_0=LREAL#5.0、

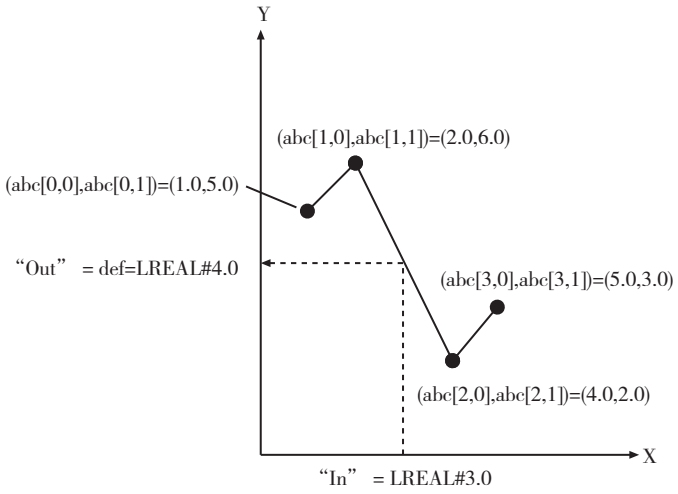
abc[1.0]=X_1=LREAL#2.0、abc[1,1]=Y_1=LREAL#6.0、

abc[2.0]=X_2=LREAL#4.0、abc[2,1]=Y_2=LREAL#2.0、

abc[3.0]=X_3=LREAL#5.0、abc[3,1]=Y_3=LREAL#3.0

转换后，“Out”的值为LREAL#4.0。





PWLApprox指令和PWLApproxNoLineChk指令的区别

PWLApprox指令和PWLApproxNoLineChk指令的区别在于是否对“In”和Line[ ]的有效性进行检查及其处理时间。规格分别如下所示。

指令	检查内容	数据无效时的处理	处理时间
PWLApprox	• 应将Line[ ]的元素按X坐标升序排列 • “In”、Line[ ]为实数时，“In”和Line[ ]的元素不可为非数、正无穷大、负无穷大	• 会发生异常。 • ENO的值变为FALSE。 • “Out”的值不变。	长
PWLApprox NoLineChk	不检查	• 不会发生异常。 • ENO的值变为TRUE。 • 有时不向 “Out”输出有效值。	短

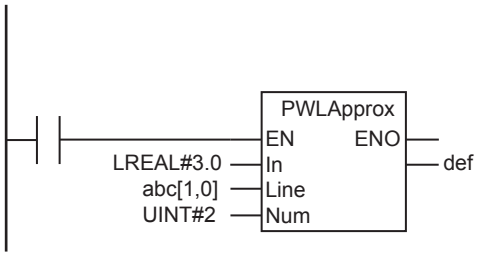
PWLApproxNoLineChk指令和PWLLineChk指令

PWLApproxNoLineChk虽然不检查 “In”和Line[ ]的有效性，但具有处理时间短的特点。因此，如果保证了这些输入变量的有效性，则与PWLApprox指令相比，使用PWLApproxNoLineChk指令更有效。
另一个为 “PWLLineChk指令(P.2–423)”检查Line[ ]的元素是否按X坐标升序排列的指令。因此，还有这样的方法，即通常使用PWLApproxNoLineChk指令，仅不保证Line[ ]的元素是否按X坐标升序排列时，与PWLLineChk指令同时使用，缩短处理时间。

参考

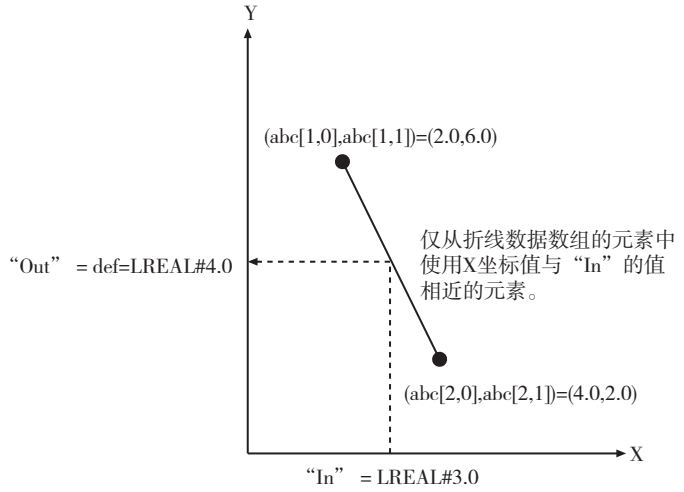
通过缩小用于近似转换的折线数据数组的元素范围，也可缩短处理时间。
上述示例中，“In”的值为LREAL#3.0，因此如果仅对折线数据数组的元素中X坐标值接近3.0
(abc[1,0],abc[1,1])=(2.0,6.0)
(abc[2,0],abc[2,1])=(4.0,2.0)
的4个元素进行近似转换，则缩短处理时间。
此时，“Num”=UINT#2，将传输至Line[ ]的abc[ ]的元素设为abc[1,0]。
转换结果保持 “Out”=LREAL#4.0不变。

LD



ST

```
def:=PWLApprox(LREAL#3.0, abc[1,0], UINT#2);
```



使用注意事项

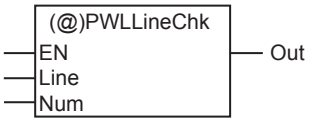
- In”的值小于Line[0,0]的值(即X_1的值)时，“Out”的值为Line[0,1](即Y_1的值)。
- “In”的值大于Line[“Num”−1,0](即X “Num”的值)时，“Out”的值为Line[“Num”−1,1](即Y “Num”的值)。
- Line[ ]为2维或3维数组，请将第1维的元素数设为2。
- “Num”的值为0时，“Out”的值为0。
- 以下情况时，PWLApprox指令会发生异常。ENO变为FALSE，“Out”不变。PWLApproxNoLineChk指令不会发生异常。
  - 折线数据的X坐标不按升序，即$X_1<X_2<\cdots\cdots<$X “Num”的顺序排列。
  - “In”和Line[ ]为实数型，且这些值为非数、正无穷大、负无穷大时。
- 以下情况时，无论PWLApprox指令还是PWLApproxNoLineChk指令均会发生异常。ENO变为FALSE，“Out”不变。
  - “Num”的值超过Line[ ]的数组区域时。
  - “In”的值超过Line[ ]指定的折线数据的X坐标的范围时。

版本相关信息

PWLApproxNoLineChk指令可用于Ver.1.03以上的CPU单元和Ver.1.04以上的Sysmac Studio。

PWLLineChk

判定折线近似转换(无折线数据检查)指令使用的折线数据是否按X坐标升序排列。

指令	名称	FB/FUN	图形表现	ST表现
PWLLineChk	折线数据检查	FUN		Out:=PWLLineChk(Line, Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Line[ ] 数组	折线数据数组	输入	折线数据数组	遵从数据类型	–	(*)
Num	折线数据数		折线数据数			1
Out	判定结果	输出	判定结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

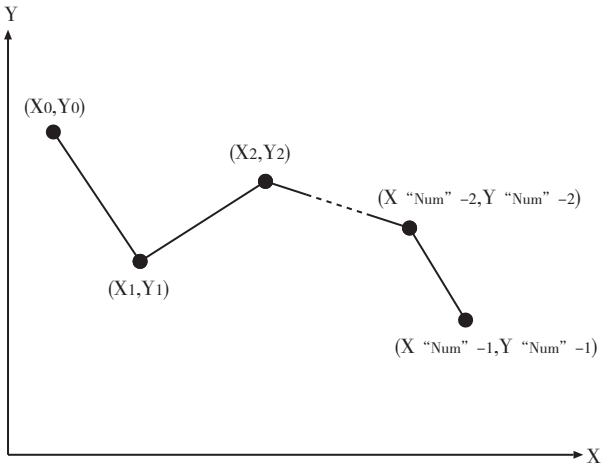
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Line[ ] 数组						○	○	○	○	○	○	○	○	○	○					
Num							○													
Out	○																			

功能

判定折线近似转换(无折线数据检查)PWLApproxNoLineChk指令使用的折线数据数组Line[ ]的元素是否按X坐标升序排列。如果按升序排列，判定结果 “Out”的值为TRUE，否则 “Out”的值为FALSE。

折线数据Line[ ]的元素和折线数据数 “Num”

Line[ ] 为 2 维或 3 维数组，请将第 1 维的元素数设为 2。如图所示，请将折线数据的各点坐标值 (X_0,Y_0)、(X_1,Y_1)……作为Line[ ]的各元素。折线数据数 “Num”为用于折线近似计算的Line[ ]的元素数的1/2。



Line[ ]为2维数组时

Line[0,0]	X_0
Line[0,1]	Y_0
Line[1,0]	X_1
Line[1,1]	Y_1
Line[2,0]	X_2
Line[2,1]	Y_2
:	:
Line[“Num” −1,0]	X “Num” −1
Line[“Num” −1,1]	Y “Num” −1

Line[ ]为3维时

Line[0,0,0]	X_0
Line[0,0,1]	Y_0
Line[0,1,0]	X_1
Line[0,1,1]	Y_1
Line[0,2,0]	X_2
Line[0,2,1]	Y_2
:	:
Line[0, “Num” −1,0]	X “Num” −1
Line[0, “Num” −1,1]	Y “Num” −1

记述示例

判定元素数为4的折线数据数组abc[ ]是否按X坐标升序排列的示例如下所示。“Num”=UINT#4，abc[ ]的元素值如下所示。

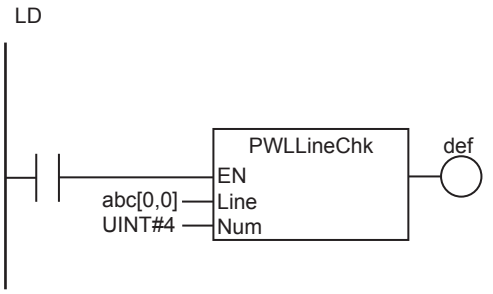
abc[0.0]=X_0=LREAL#1.0、abc[0,1]=Y_0=LREAL#5.0、

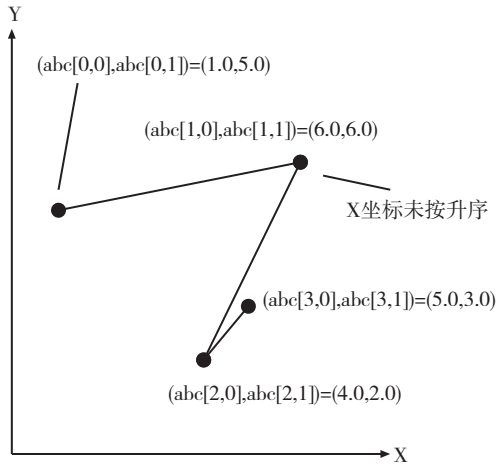
abc[1.0]=X_1=LREAL#6.0、abc[1,1]=Y_1=LREAL#6.0、

abc[2.0]=X_2=LREAL#4.0、abc[2,1]=Y_2=LREAL#2.0、

abc[3.0]=X_3=LREAL#5.0、abc[3,1]=Y_3=LREAL#3.0

X坐标未按升序排列，因此 “Out”的值为FALSE。





参考

- 本指令请与PWLApproxNoLineChk指令一起使用。PWLApproxNoLineChk指令的详情请参阅 “PWLApprox/ PWLApproxNoLineChk指令(P.2–418)”。
- 每次折线近似转换均进行折线数据检查时，请使用PWLApprox指令。PWLApprox指令的详情也请参阅 “PWLApprox/ PWLApproxNoLineChk指令(P.2–418)”。与PWLApprox指令相比，PWLApproxNoLineChk指令的处理时间较短。

使用注意事项

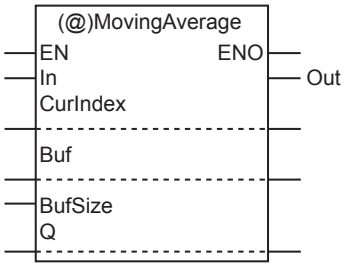
- Line[ ]为2维或3维数组，请将第1维的元素数设为2。
- 以下情况时会发生异常。“Out”为FALSE。
  - “Num”的值超过Line[ ]的数组区域时。
  - Line[ ]为实数型，且元素的值为非数、正无穷大、负无穷大时。

版本相关信息

本指令可用于Ver.1.03以上的CPU单元和Ver.1.04以上的Sysmac Studio。

MovingAverage

计算出移动平均值。

指令	名称	FB/FUN	图形表现	ST表现
Moving Average	移动平均	FUN		Out:=MovingAverage(In, CurIndex, Buf, BufSize, Q);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	输入值	输入	用于平均值计算的数值	遵从数据类型	–	(*)
BufSize	最大元素数		用于平均值计算的最大元素数			1
CurIndex	输入值保存位置	输入输出	保存 “In” 的Buf[ ]的位置	遵从数据类型	–	–
Buf[ ]数组	输入值保存数组		保存 “In” 的数组			
Q	计算完成标志		TRUE：保存至Buf[ ]的数值数大于 “BufSize” FALSE：保存至Buf[ ]的数值数为小于 “BufSize”			
Out	运算结果	输出	运算结果	遵从数据类型	–	–

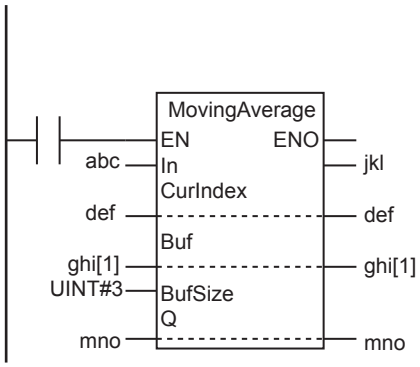
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○	○	○					
BufSize							○													
CurIndex							○													
Buf[ ]数组	将与 “In” 相同的数据类型作为元素的数组																			
Q	○																			
Out						○	○	○	○	○	○	○	○	○	○					

功能

每次执行本指令时，均将输入值 “In”保存在输入值保存数组Buf [ ]中。再将已保存的值的平均值保存在运算结果 “Out”中。用于平均值计算的最大元素数由 “BufSize”指定。
以 “BufSize”=UINT#3为例，说明处理的步骤。示例如下所示。

LD



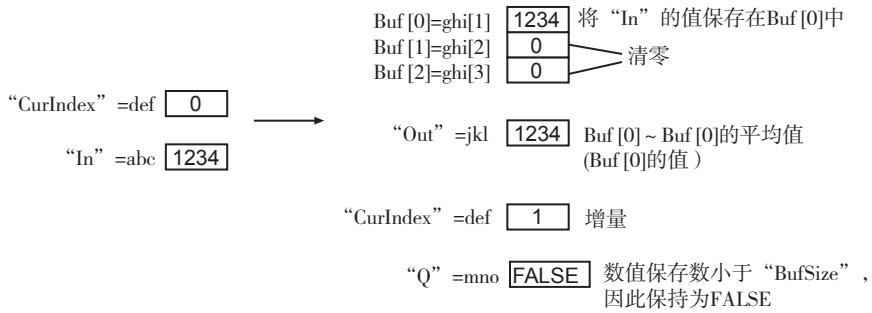
ST

```
jkl:=MovingAverage(abc, def, ghi[1], UINT#3, mno);
```

初次数值输入

为输入值保存位置 “CurIndex”设定0，执行本指令。
将输入值保存数组Buf [ ]的Buf [0]到Buf [“BufSize”–1]清零后，在Buf [0]中保存首次的输入值 “In”的值。
计算完成标志 “Q”的值变为FALSE。这表示保存至Buf [ ]的数值数未达到 “BufSize”。
“Q”的值为FALSE的期间，对从Buf [0]开始的 “CurIndex”+1个数值进行平均值计算。将运算结果保存在 “Out”中。
然后，对 “CurIndex”的值进行增量。

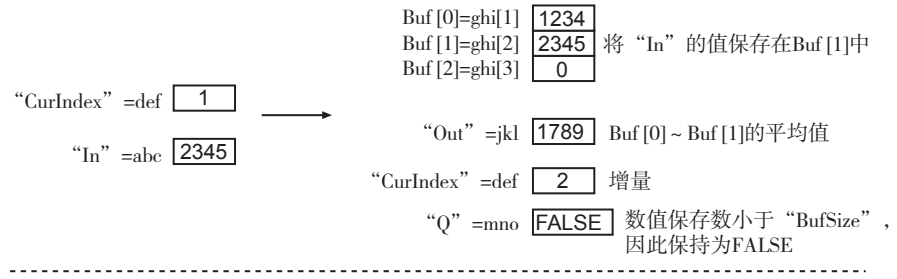
第1次执行指令



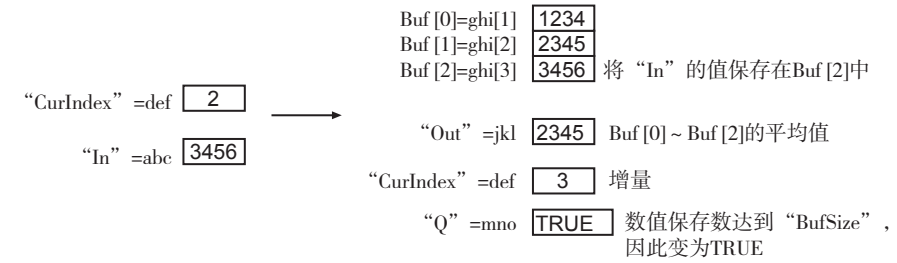
到第 “BufSize”次为止的数值输入

每次执行本指令时，“In”的值均保存在Buf [“CurIndex”]中。从Buf [0]开始对 “CurIndex”+1个数值进行平均值计算，并将运算结果保存在 “Out”中。
执行次数达到 “BufSize”时，“Q”的值变为TRUE。

第2次执行指令



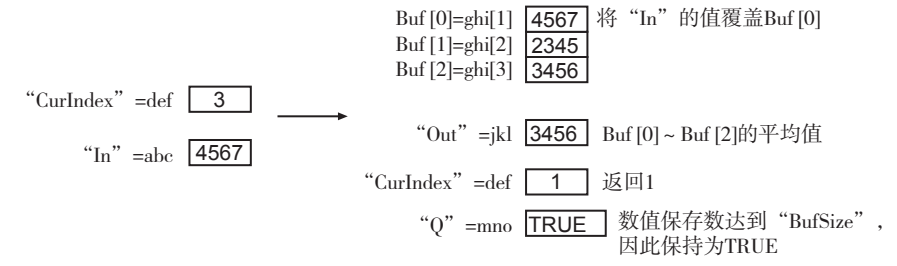
第3次执行指令



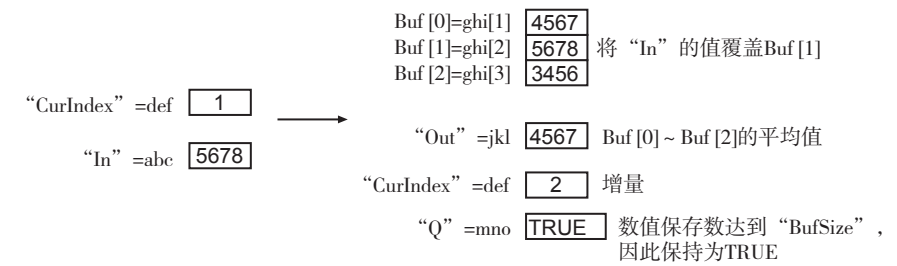
第 “BufSize”次后的数值输入

每次执行本指令时，在Buf [0]至Buf [“BufSize”−1]之间，定期覆盖 “In”的值。对该时刻的Buf [0]至Buf [“BufSize”−1]进行平均值计算，并将运算结果保存在 “Out”中。
“CurIndex”的值在 “BufSize”之后返回1，并从该处开始再进行增量。“Q”的值保持为TRUE。

第4次执行指令



第5次执行指令

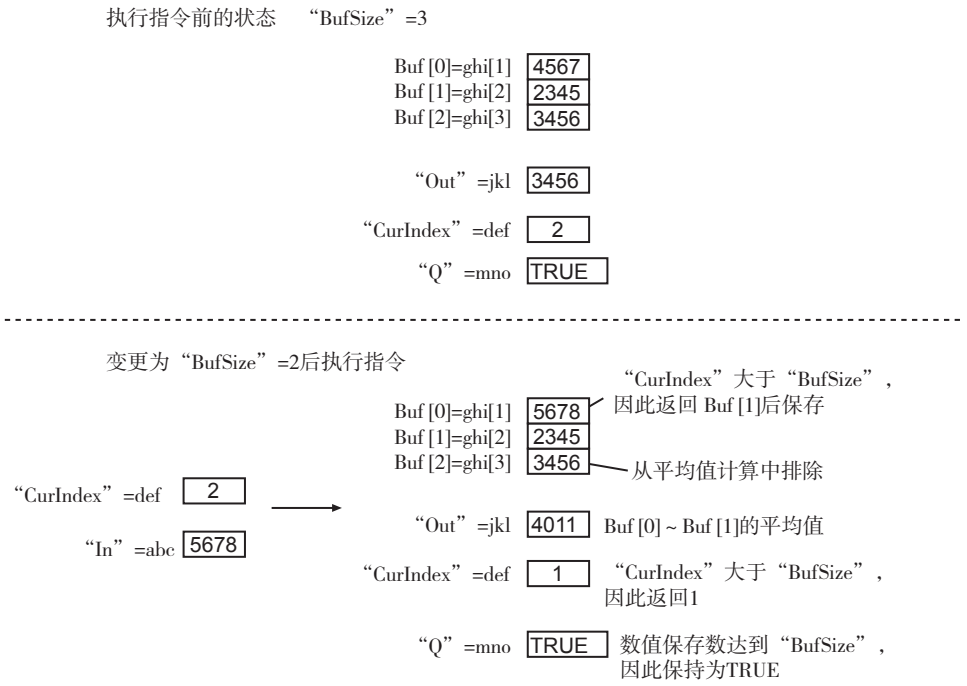


保存值的初始化

如果将 “CurIndex”的值设为0后再执行本指令，则从Buf [0]至Buf [“BufSize” –1]的值会先变为0，并重新将该时刻的 “In”的值保存在Buf [0]中。
“CurIndex”的值为1，“Q”的值为FALSE。

“BufSize”值的变更

变更 “BufSize”的值后执行本指令时，将按照变更后的 “BufSize”和该时刻的 “CurIndex”的值动作。

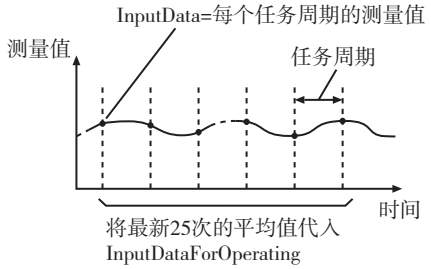


使用注意事项

- 请将 “In”和Buf [ ]的元素的数据类型设为相同。
- 请务必将Buf [ ]的大小设为大于 “BufSize”。
- 即使运算结果超过 “Out”的有效范围，也不会发生异常，但 “Out”中保存错误值。
- “BufSize”的值为0时，“Out”、“CurIndex”的值为0。“Q”的值变为TRUE。
- 变更 “BufSize”的值时，请务必将 “CurIndex”的值设为0，进行保存值的初始化。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “BufSize”的值超过Buf [ ]的大小时。

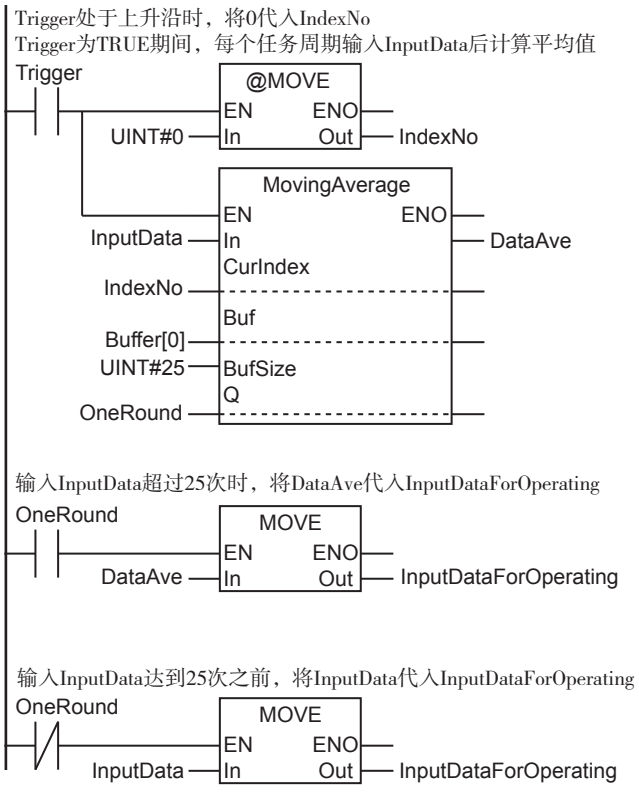
示例程序

这是消除传感器等模拟量输入数据中含有的噪声等外部干扰影响的示例。
将最新25次的输入数据InputData的平均值DataAve代入下个工序的输入InputDataForOperating。
InputData在执行条件Trigger的值为TRUE的期间，每个任务周期进行输入。
InputData输入达到25次前，由于数据量不足，因此将最新1次的InputData值代替平均值代入InputDataForOperating。
Trigger的值由FALSE变为TRUE时，清除平均值，重新开始进行InputData的输入。



LD

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
InputData	INT	10	输入值
Buffer	ARRAY[0..24] OF INT	[25(0)]	输入值保存数组
DataAve	INT	0	平均值
OneRound	BOOL	FALSE	25次输入标志
IndexNo	UINT	0	输入值保存位置
InputDataForOperating	INT	0	下个工序输入



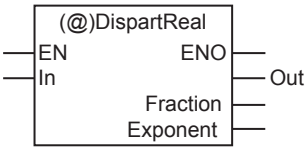
ST

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
Operating	BOOL	FALSE	处理中
OperatingStart	BOOL	FALSE	处理开始
Buffer	ARRAY[0..24] OF INT	[25(0)]	输入值保存数组
InputData	INT	10	输入值
DataAve	INT	0	平均值
OneRound	BOOL	FALSE	25次输入标志
IndexNo	UINT	0	输入值保存位置
InputDataForOperating	INT	0	下个工序输入

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    OperatingStart:=TRUE;
    Operating:=TRUE;
END_IF;
LastTrigger:=Trigger;

// 清除平均值
IF (OperatingStart=TRUE) THEN
    IndexNo:=UINT#0;
    OperatingStart:=FALSE;
END_IF;

// 移动平均处理
IF (Operating=TRUE) THEN
    DataAve:=MovingAverage(
        In          :=InputData,
        CurIndex    :=IndexNo,
        Buf         :=Buffer[0],
        BufSize     :=UINT#25,
        Q           :=OneRound);

    IF (OneRound=TRUE) THEN
        // 代入最新25次的输入值的平均值
        InputDataForOperating:=DataAve;
  ELSE
        // 代入最新1次的输入值
        InputDataForOperating:=InputData;
    END_IF;
END_IF;

// 移动平均处理结束
IF (Trigger=FALSE) THEN
    Operating:=FALSE;
END_IF;
```

DispartReal

将实数分离为带符号尾数部分和指数部分。

指令	名称	FB/FUN	图形表现	ST表现
DispartReal	实数的尾数、指数分离	FUN		Out:=DispartReal(In, Fraction, Exponent);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	实数	输入	待分离的实数	遵从数据类型	–	(*1)
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
Fraction	带符号的尾数部分		带符号的尾数部分	(*2)		
Exponent	指数部分		指数部分	(*3)		

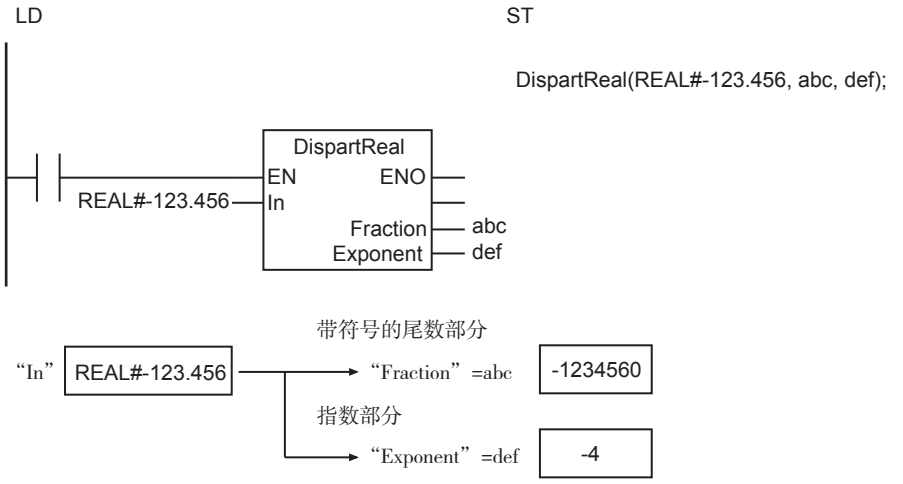
*1 省略输入参数时，初始值不适用。编连时会发生异常。
*2 有效范围因 “In”“Fraction”的数据类型的组合而异。详情请参阅功能说明。
*3 “In”为REAL型时，有效范围为–44 ~ 32； “In”为LREAL型时，有效范围为–322 ~ 294。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In														○	○					
Out	○																			
Fraction	“In”的数据类型为REAL时，则为DINT；为LREAL时，则为LINT																			
Exponent											○									

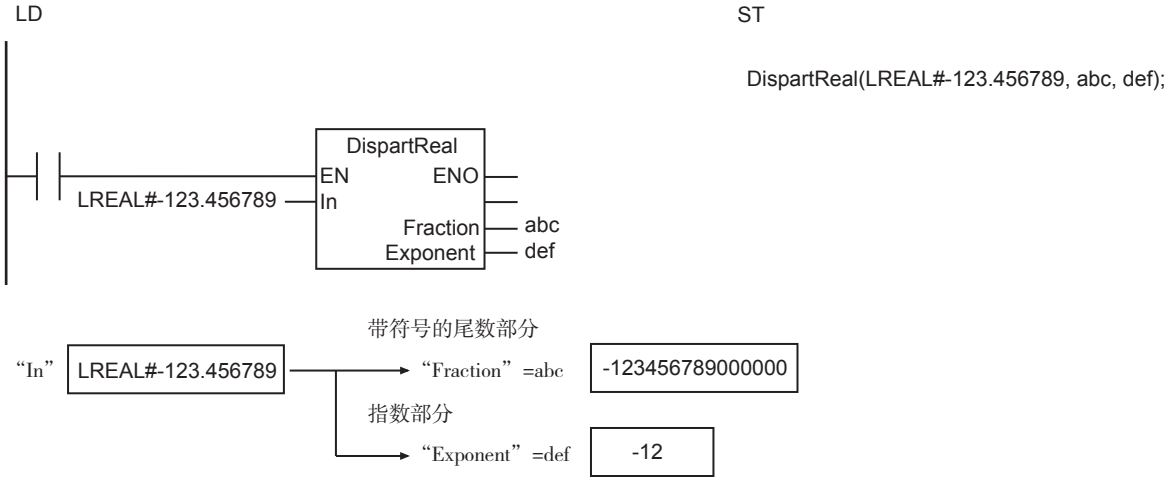
功能

将实数 “In”分离为带符号尾数部分 “Fraction”和指数部分 “Exponent”。

“In”为REAL型时，“Fraction”为7位整数。“In”为LREAL型时，“Fraction”为15位整数。
“In”为REAL型，值为REAL#–123.456时的示例如下所示。



“In”为LREAL型，值为LREAL#–123.456789时的示例如下所示。



如下所示，根据 “In”和 “Fraction”的数据类型的组合，“Fraction”的有效范围有所差异。

“In”的数据类型	“Fraction”的数据类型	“Fraction”的有效范围
REAL	DINT	–9999999 ~ 9999999
LREAL	LINT	–999999999999999 ~ 999999999999999

参考

组合带符号的尾数部分和指数部分来获取实数时，请使用 “UniteReal指令(P.2–435)”。

使用注意事项

- 转换为整数时，可能因 “In” 的值而产生误差。
- “In” 的有效位数大于 “Fraction” 的有效位数时，进行五舍五入，以将其控制在 “Fraction” 的有效范围内。五舍五入是指如下所示的处理。

小数部分的值	处理	例
小于0.5	舍去	1.49→1 −1.49→−1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2 −1.50→−2 −2.50→−2
大于0.5	进位	1.51→2 −1.51→−2

- 以下情况时会发生异常。ENO变为FALSE，“Fraction” “Exponent” 不变。
  - “In” 的值为非数或无穷大时。

UniteReal

将带符号的尾数部分和指数部分组合成实数。

指令	名称	FB/FUN	图形表现	ST表现
UniteReal	尾数、指数组合成实数	FUN	(@)UniteReal EN ENO Fraction Exponent Out	Out:=UniteReal(Fraction, Exponent);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Fraction	带符号的尾数部分	输入	带符号的尾数部分	遵从数据类型	–	(*)
Exponent	指数部分		指数部分			0
Out	实数	输出	实数	遵从数据类型	–	–

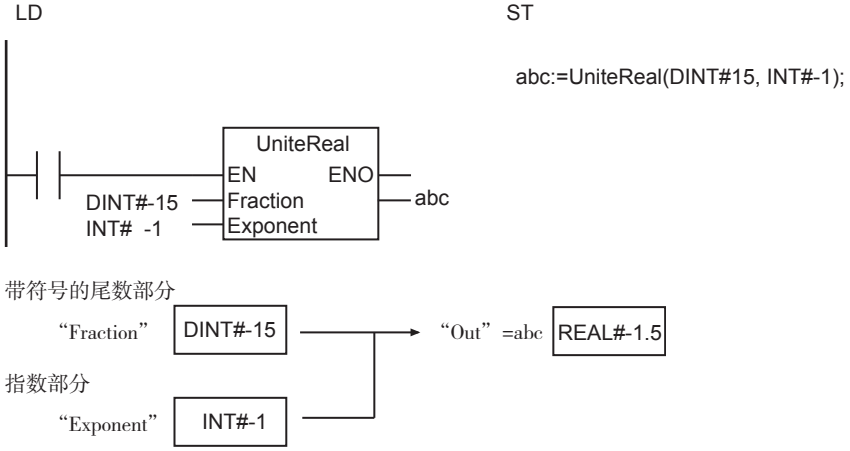
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Fraction												○	○							
Exponent											○									
Out	“Fraction”的数据类型为DINT时，“Out”的数据类型为REAL， “Fraction”的数据类型为LINT时，“Out”的数据类型为LREAL																			

功能

组合带符号尾数部分 “Fraction”和指数部分 “Exponent”，获取实数 “Out”。

“Fraction”=DINT#–15，“Exponent”=INT#–1时的示例如下所示。



参考

将实数分离为带符号尾数部分和指数部分时，请使用 “DispartReal指令(P.2–432)”。

使用注意事项

- 将整数转换为实数时，可能 “Fraction”和 “Exponent”的值而产生误差。
- 组合结果超过 “Out”的有效范围时，如果 “Exponent”为正数，则 “Out”的值为与 “Fraction”具有相同符号的无穷大。如果 “Exponent”为负数，则 “Out”的值为0。

NumToDecString/ NumToHexString

NumToDecString :将整数转换为固定长度的10进制字符串格式。
NumToHexString :将整数转换为固定长度的16进制字符串格式。

指令	名称	FB/ FUN	图形表现	ST表现
NumToDec String	固定长度10进制字符串转换	FUN	(@)NumToDecString EN ENO In Out L Fill	Out:=NumToDecString(In, L, Fill);
NumToHex String	固定长度16进制字符串转换	FUN	(@)NumToHexString EN ENO In Out L Fill	Out:=NumToHexString(In, L, Fill);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	整数	输入	整数	遵从数据类型	–	(*)
L	字符数		“Out”的字符数	0 ~ 1985		1
Fill	添加字符		添加字符	_BLANK, _ZERO		_BLANK
Out	字符串	输出	字符串	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○							
L							○													
Fill	枚举体_eFILL_CHR　枚举元素参阅功能说明																			
Out																				○

功能

- **NumToDecString**
  将整数 “In”转换为UTF-8半角英数字的10进制格式的字符串。“In”为负数时，在开头加上'-'(减号)。

- **NumToHexString**
  将整数 “In”转换为UTF-8半角英数字的16进制格式的字符串。“In”为负数时，以2的补数形式表现(位取反+1)。

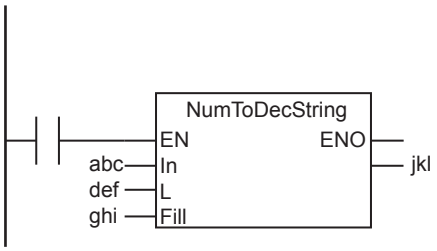
上述指令均将字符串 “Out”的字符数设为 “L” 。字符数不足时，将添加字符 “Fill”指定的字符添加在高位上。转换结果的字符数多于 “L”时，从转换结果的最低位仅将 “L”字符代入 “Out”。
“Out”的结尾带NULL字符。字符数中不包含NULL字符。

“Fill”的数据类型为枚举体_eFILL_CHR。枚举元素的含义如下所示。

枚举元素	含义
_BLANK	' '(半角空白字符)
_ZERO	'0'

NumToDecString指令的几个示例如下所示。

LD



ST

```
jkl:=NumToDecString(abc, def, ghi);
```

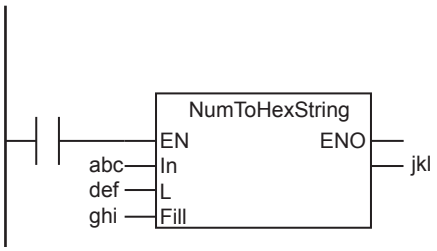
“In” =abc=INT#128、 “L” =def=UINT#8、 “Fill” =ghi = _BLANK
“Out” =jkl | | | | | 1 | 2 | 8 |

“In” =abc=INT#−128、 “L” =def=UINT#8、 “Fill” =ghi = _BLANK
“Out” =jkl | | | | - | 1 | 2 | 8 |

“In” =abc=INT#−128、 “L” =def=UINT#8、 “Fill” = ghi=_ZERO
“Out” =jkl | - | 0 | 0 | 0 | 0 | 1 | 2 | 8 |

NumToHexString指令的几个示例如下所示。

LD



ST

```
jkl:=NumToHexString(abc, def, ghi);
```

“In” =abc=INT#128、 “L” =def=UINT#8、 “Fill” =ghi = _BLANK
“Out” =jkl | | | | | | 8 | 0 |

“In” =abc=INT#128、 “L” =def=UINT#8、 “Fill” = ghi=_ZERO
“Out” =jkl | 0 | 0 | 0 | 0 | 0 | 8 | 0 |

“In” =abc=INT#−128、 “L” =def=UINT#8、 “Fill” = ghi=_BLANK
“Out” =jkl | F | F | F | F | F | 8 | 0 |

使用注意事项

- “L”的值为0时，“Out”的值不变。
- 转换结果的字符数多于 “L”时，从转换结果的低位将 “L”位保存在 “Out”中。示例如下所示。

指令	“In”的值	“L”的值	“Out”的值
NumToDecString	128	2	28
NumToHexString			80

- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “L”的值超过有效范围时。
  - “Fill”的值超过有效范围时。

HexStringToNum_**

将16进制字符串格式转换为整数。

指令	名称	FB/FUN	图形表现	ST表现
HexStringToNum_**	16进制字符串→数值转换组	FUN	(@)HexStringToNum_** EN ENO In Out **为整数的数据类型名称	Out:=HexStringToNum_**(In); **为整数的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	16进制字符串	输入	16进制字符串	遵从数据类型	–	''
Out	整数	输出	整数	遵从数据类型	–	–

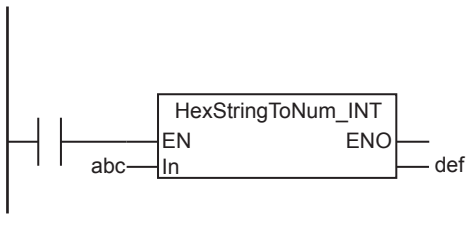
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out						○	○	○	○	○	○	○	○							

功能

将16进制格式的字符串 “In”转换为整数。忽略高位的半角空格(16#20)和'0'(16#30)。同时忽略字符串中的下划线(16#5F)。
指令名称因 “Out”的数据类型而异。例如，“Out”为INT型时，指令名称为HexStringToNum_INT。

几个示例如下所示。

LD



ST

```
def:=HexStringToNum_INT(abc);
```

“In” =abc 8 0 ⟶ “Out” =def=128

“In” =abc - 8 0 ⟶ “Out” =def=-128

“In” =abc - 0 0 0 0 0 F ⟶ “Out” =def=-15

使用注意事项

- 即使转换结果超过 “Out”的有效范围，也不会发生异常，但 “Out”中保存错误值。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的内容为无法进行数值转换的字符时。

FixNumToString

将带符号的固定小数点数转换为10进制字符串格式。

指令	名称	FB/FUN	图形表现	ST表现
FixNumTo String	固定小数点数→字符串转换	FUN	(@)FixNumToString EN ENO In Out Zero	Out:=FixNumToString(In, Zero);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	固定小数点数	输入	带符号的固定小数点数	遵从数据类型	–	0
Zero	零显示		小数点后的位数不到3时的显示 TRUE ：补 “0” FALSE：不补 “0”			TRUE
Out	10进制字符串	输出	10进制字符串	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In				○																
Zero	○																			
Out																				○

功能

将带符号的固定小数点数 “In”转换为10进制字符串。转换步骤如下所示。

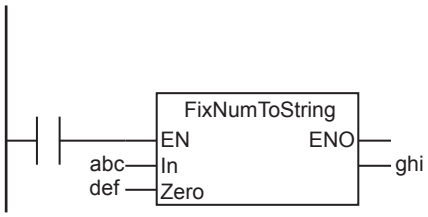
1 将16进制格式的 “In”转换为10进制格式。

2 将该值除以1,000。

“In”的小数点后位数不到3时，零显示 “Zero”指定 “Out”的小数点后第3位前是否补 “0”。如果“Zero”的值为TRUE，则补'0'。
“Out”的结尾带NULL字符。

几个示例如下所示。

LD



ST

```
ghi:=FixNumToString(abc, def);
```

“In”=abc	“Out”=ghi	
	“Zero”=def=TRUE	“Zero”=def=FALSE
16#0001462C (10#83500)	'83.500'	'83.5'
16#00051AA4 (10#334500)	'334.500'	'334.5'
16#0003BEFC (10#245500)	'245.500'	'245.5'

参考

固定小数点数的格式与欧姆龙的视觉传感器FZ系列的固定小数点输出的格式通用。

StringToFixNum

将10进制字符串转换为带符号的固定小数点数形式。

指令	名称	FB/FUN	图形表现	ST表现
StringToFixNum	字符串→固定小数点数转换	FUN	(@)StringToFixNum EN ENO In Out	Out:=StringToFixNum(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	10进制字符串	输入	10进制字符串	遵从数据类型	–	''
Out	固定小数点数	输出	固定小数点数	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out				○																

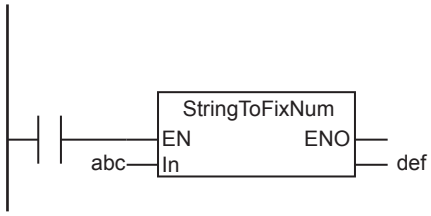
功能

将10进制字符串 “In” 转换为固定小数点数。转换步骤如下所示。

1. 将 “In” 表现的数值乘以1,000。
2. 该值的小数点后舍去。
3. 将该值转换为32位的16进制格式(DWORD型)。

几个值的示例如下所示。

LD



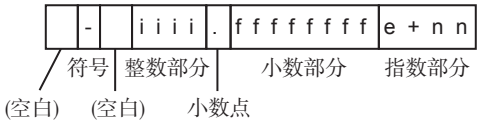
ST

```
def:=StringToFixNum(abc);
```

“In” =abc	“Out” =def
'83.5'	16#0001462C (10#83500)
'334.5'	16#00051AA4 (10#334500)

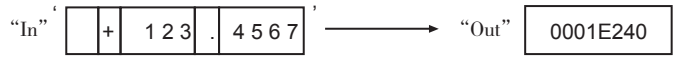
“In” =abc	“Out” =def
'245.5'	16#0003BEFC (10#245500)

“In”的字符串格式如下所示。

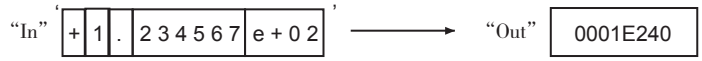


名称	格式
符号	• 以忽略字符串开头出现的连续的' '(空格16#20)后出现的单个'+'或'–'为符号。 • 可省略符号。 • 同时忽略符号后的连续空格。
整数部分	• 将符号后至小数点前的连续的数值('0' ~ '9')作为整数部分。有时也省略符号。有时符号和整数部分之间会有连续的空格。 • 无小数点和小数部分时，指数部分前为整数部分。 • 无小数点、小数部分和指数部分时，到字符串结尾为止为整数部分。 • 不可省略整数部分。 • 整数部的最大位数为最大字符串长度1986减去符号、小数点、小数部分、指数部分以及符号前后空格的总字节数后的位数。
小数点	• 整数部分后的单个'.'(点)为小数点。 • 无小数部时，请省略小数点。
小数部分	• 将小数点后至指数部分前的连续的数值('0' ~ '9')作为小数部分。 • 无指数部分时，到字符串结尾为止为小数部分。 • 可省略小数部分。无小数点时，也无小数部分。 • 小数部分的最大位数为15位。
指数部分	• 小数部分后的单个'e'或'E'，单个'+'或'–'到字符串结尾的连续的数值('0' ~ '9')为指数部分。 • 无小数部分时，小数点后的字符串为指数部分。 • 无小数点和小数部分时，整数后的上述字符串为指数部分。 • 可省略指数部分。 • 指数部分的数字部分的最大位数为3。

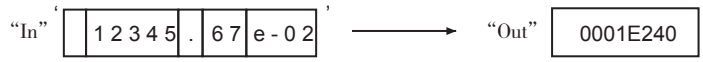
(例1) 有符号、小数点、小数部分，无指数部分



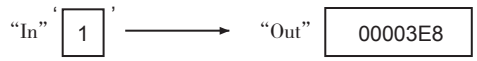
(例2) 有符号、小数点、小数部分、指数部分



(例3) 无符号，有小数点、小数部分、指数部分



(例4) 无符号、小数点、小数部分、指数部分



参考

固定小数点数的格式与欧姆龙的视觉传感器FZ系列的固定小数点输出的格式通用。

使用注意事项

- “In”的小数点后4位之后舍去。
- 忽略 “In”字符串中的下划线(16#5F)。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的内容为无法进行数值转换的字符时。
  - “In”的内容有小数点，但无小数部分时。

DtToString

将日期时间转换为字符串格式。

指令	名称	FB/FUN	图形表现	ST表现
DtToString	日期时间→字符串转换	FUN	(@)DtToString EN ENO In Out	Out:=DtToString(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	日期时间	输入	日期时间	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
Out	字符串	输出	字符串	30字节 (29个半角英数字字符+结尾NULL字符)	-	-

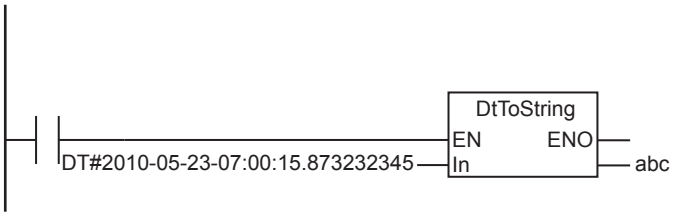
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Out																				○

功能

将日期时间 “In”转换为字符串。字符串 “Out”的结尾带NULL字符。

“In”的值为2010年5月23日7时0分15.873232345秒时的示例如下所示。变量abc的值为'2010-05-23-07:00:15.873232345'。

LD



ST

```
abc:=DtToString(DT#2010-05-23-07:00:15.873232345);
```

将日期时间“In”转换为字符串。
“In”的值为2010年5月23日7时0分15.873232345秒，因此abc的值为'2010-05-23-07:00:15.873232345'。

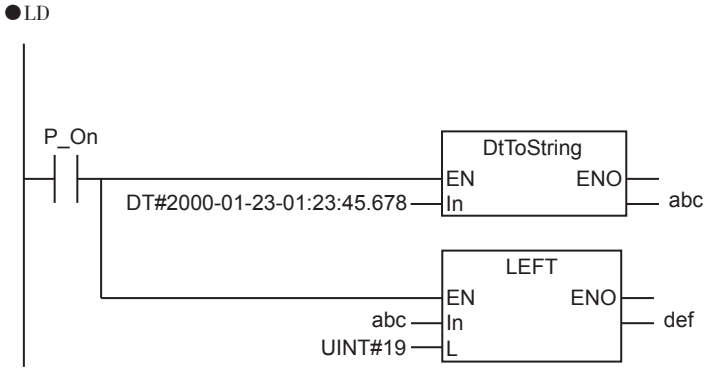
“In” DT#2010-05-23-07:00:15.873232345 —字符串转换→ “Out”=abc '2010-05-23-07:00:15.873232345'

参考

“Out”以纳秒为单位表现。获取以秒或毫秒为单位的字符串时，请同时使用本指令和 “LEFT/RIGHT指令(P.2–540)”。

获取以秒为单位的字符串时的示例如下所示。

●LD



●ST

```
def:=LEFT(DtToString(DT#2000-01-23-01:23:45.678), UINT#19);
```

DateToString

将日期转换为字符串格式。

指令	名称	FB/FUN	图形表现	ST表现
DateToString	日期→字符串转换	FUN	(@)DateToString EN ENO In Out	Out:=DateToString(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	日期	输入	日期	遵从数据类型	年月日	D#1970-1-1
Out	字符串	输出	字符串	11字节 (10个半角英数字字符 + 结尾NULL字符)	-	-

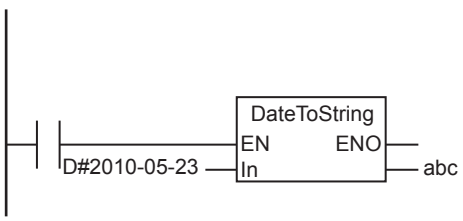
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																	○			
Out																				○

功能

将日期 “In”转换为字符串。“Out”的结尾带NULL字符。

“In”的值为2010年5月23日时的示例如下所示。变量abc的值为'2010-05-23'。

LD



ST

```
abc:=DateToString(D#2010-05-23);
```

将日期“In”转换为字符串。
“In”的值为2010年5月23日，因此abc的值为'2010-05-23'。

“In” D#2010-05-23 —字符串转换→ “Out”=abc '2010-05-23'

TodToString

将时刻转换为字符串格式。

指令	名称	FB/FUN	图形表现	ST表现
TodToString	时刻→字符串转换	FUN	(@)TodToString EN ENO In Out	Out:=TodToString(In);

变量

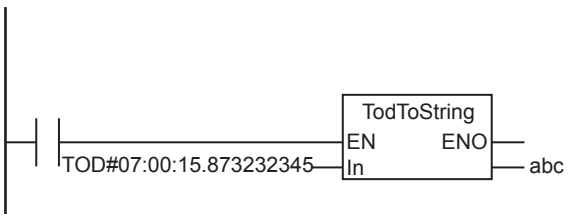
	名称	输入/输出	内容	有效范围	单位	初始值
In	时刻	输入	时刻	遵从数据类型	时分秒	TOD#0:0:0
Out	字符串	输出	字符串	19字节 (18个半角英数字字符 + 结尾NULL字符)	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																		○		
Out																				○

功能

将时刻 “In”转换为字符串。“Out”的结尾带NULL字符。
“In”的值为7时0分15.873232345秒时的示例如下所示。变量abc的值为'07:00:15.873232345'。

LD



ST

```
abc:=TodToString(TOD#07:00:15.873232345);
```

将时刻“In”转换为字符串。
“In”的值为7时0分15.873232345秒，因此abc的值为'07:00:15.873232345'。

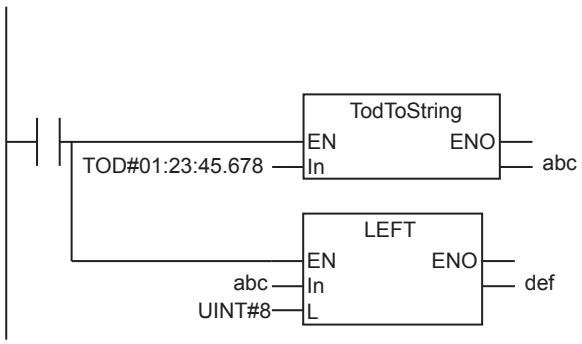
“In” TOD#07:00:15.873232345 字符串转换 “Out”=abc '07:00:15.873232345'

参考

“Out”以纳秒为单位表现。获取以秒或毫秒为单位的字符串时，请同时使用本指令和 “LEFT/RIGHT指令(P.2–540)”。

获取以秒为单位的字符串时的示例如下所示。

●LD



●ST

```
def:=LEFT(TodToString(TOD#01:23:45.678), UINT#8);
```

GrayToBin_**/BinToGray_**

GrayToBin_** :将格雷码转换为位串。
BinToGray_** :将位串转换为格雷码。

指令	名称	FB/FUN	图形表现	ST表现
GrayToBin_**	格雷码→BIN码转换组	FUN	(@)GrayToBin_** EN ENO In Out **为位串的数据类型名称	Out:=GrayToBin_**(In); **为位串的数据类型名称
BinToGray_**	BIN码→格雷码转换组	FUN	(@)BinToGray_** EN ENO In Out **为位串的数据类型名称	Out:=BinToGray_**(In); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	0
Out	转换结果	输出	转换结果	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Out	与 “In” 相同的数据类型																			

功能

- GrayToBin_**

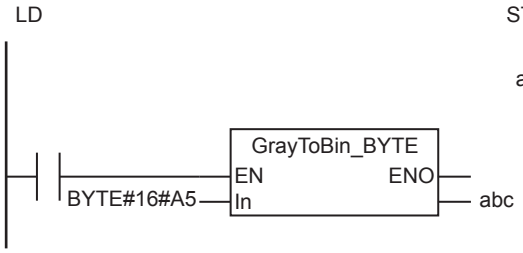
将格雷码的转换对象 “In” 转换为位串。以 “In”、“Out” 为BYTE型为例，对转换步骤进行说明。

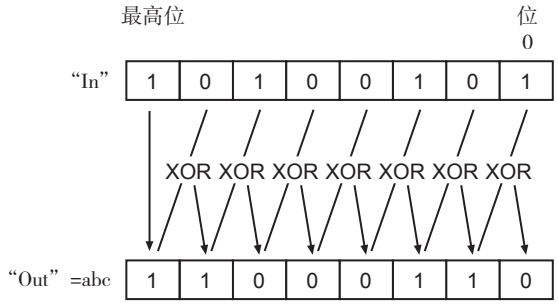
1 “In” 的最高位(位7)直接作为 “Out” 的最高位(位7)。

2 将 “In” 的位6的值和 “Out” 的位7的值的异或作为 “Out” 的位6。

3 同样地，如下所示，确定 “Out” 的最低位(位0)为止的值。

GrayToBin_BYTE指令下，“In”=BYTE#16#A5时的示例如下所示。





- BinToGray_**

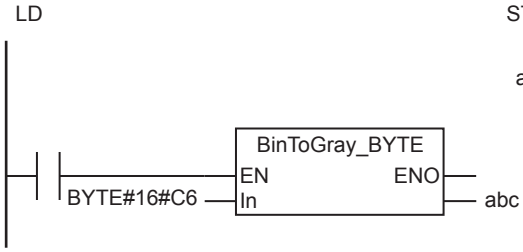
将位串的转换对象 “In”转换为格雷码。以 “In”、“Out”为BYTE型为例，对转换步骤进行说明。

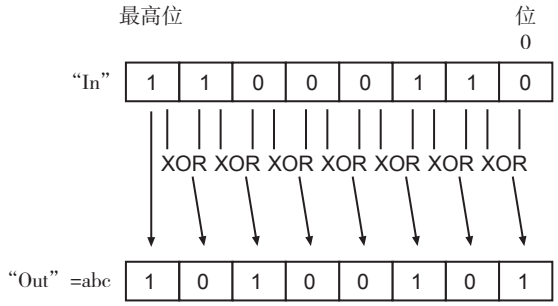
1 “In”的最高位(位7)直接作为 “Out”的最高位(位7)。

2 将 “In”的位7的值和 “In”的位6的值的异或作为 “Out”的位6。

3 同样地，如下所示，确定 “Out”的最低位(位0)为止的值。

BinToGray_BYTE指令下，“In”=BYTE#16#C6时的示例如下所示。





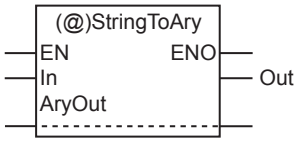
指令名称因 “In”、“Out”的数据类型而异。例如，“In”、“Out”为WORD型时，指令名称为GrayToBin_WORD或BinToGray_WORD。

使用注意事项

请将 “In” 和 “Out” 的数据类型设为相同。

StringToAry

将字符串转换为BYTE型数组。

指令	名称	FB/FUN	图形表现	ST表现
StringToAry	字符串→数组转换	FUN		Out:=StringToAry(In, AryOut);

变量

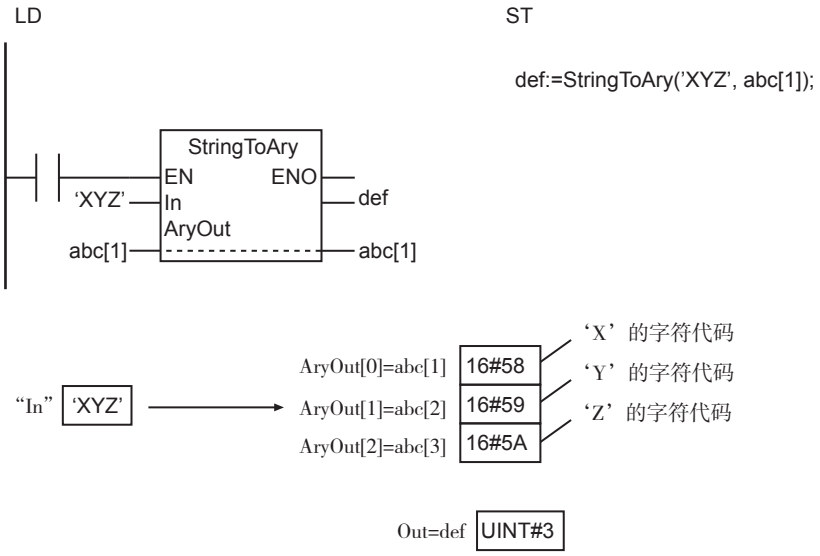
	名称	输入/输出	内容	有效范围	单位	初始值
In	字符串	输入	字符串	遵从数据类型	–	''
AryOut[ ] 数组	BYTE型数组	输入输出	BYTE型数组	遵从数据类型	–	–
Out	转换后的字节数	输出	转换后的字节数	0 ~ 1985	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
AryOut[ ] 数组		○																		
Out							○													

功能

将字符串 “In”的字符码视为数值，按字符保存在BYTE型数组AryOut[ ]中。“Out”中保存转换后的字节数。

“In”='XYZ'时的示例如下所示。

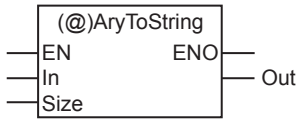


使用注意事项

- “In”结尾的NULL字符不保存在AryOut[ ]中。
- “In”为仅有NULL字符的字符串时，“Out”的值变为0，AryOut[ ]不变。
- 以下情况时会发生异常。ENO变为FALSE，“Out”和AryOut[ ]不变。
  - “In”的字节数大于AryOut[ ]的元素数时。

AryToString

将BYTE型数组转换为字符串。

指令	名称	FB/FUN	图形表现	ST表现
AryToString	数组→字符串转换	FUN		Out:=AryToString(In, Size);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	BYTE型数组	输入	BYTE型数组 元素数最大为1985	遵从数据类型	–	(*)
Size	转换元素数		待转换的In[ ]的元素数	0 ~ 1985		1
Out	字符串	输出	字符串	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

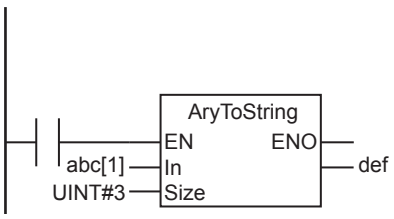
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○																		
Size							○													
Out																				○

功能

将BYTE型的数组In[0]之后的各元素的值视为字符码，保存在字符串 “Out”中。“Out”的结尾带NULL字符。
为 “Size”指定待转换的In[ ]的元素数。In[0] ~ In[“Size”–1]中含有NULL字符时，仅该处前的内容保存在 “Out”中。

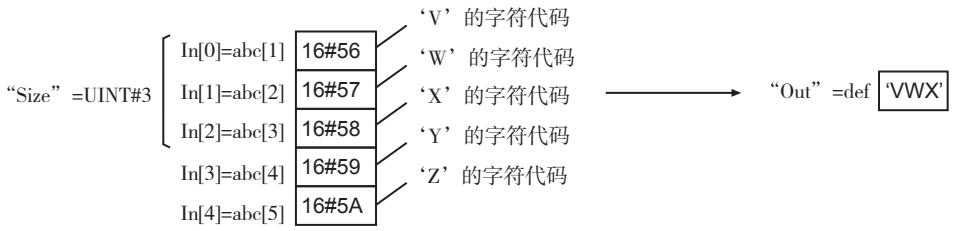
“Size”=UINT#3时的示例如下所示。

LD



ST

```
def:=AryToString(abc[1], UINT#3);
```

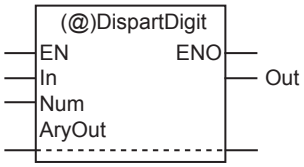


使用注意事项

- “Size”的值为0时，“Out”为仅含有NULL字符的字符串。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过In[ ]的数组区域时。

DispartDigit

以4位为单位分离位串。

指令	名称	FB/FUN	图形表现	ST表现
DispartDigit	4位分离	FUN		DispartDigit(In, Num, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	分离对象	输入	分离对象的位串	遵从数据类型	–	(*)
Num	分离位数		待分离的位数	0 ~ “In”的位数		1
AryOut[ ]数组	分离结果数组	输入输出	分离结果数组	16#00 ~ 16#0F	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

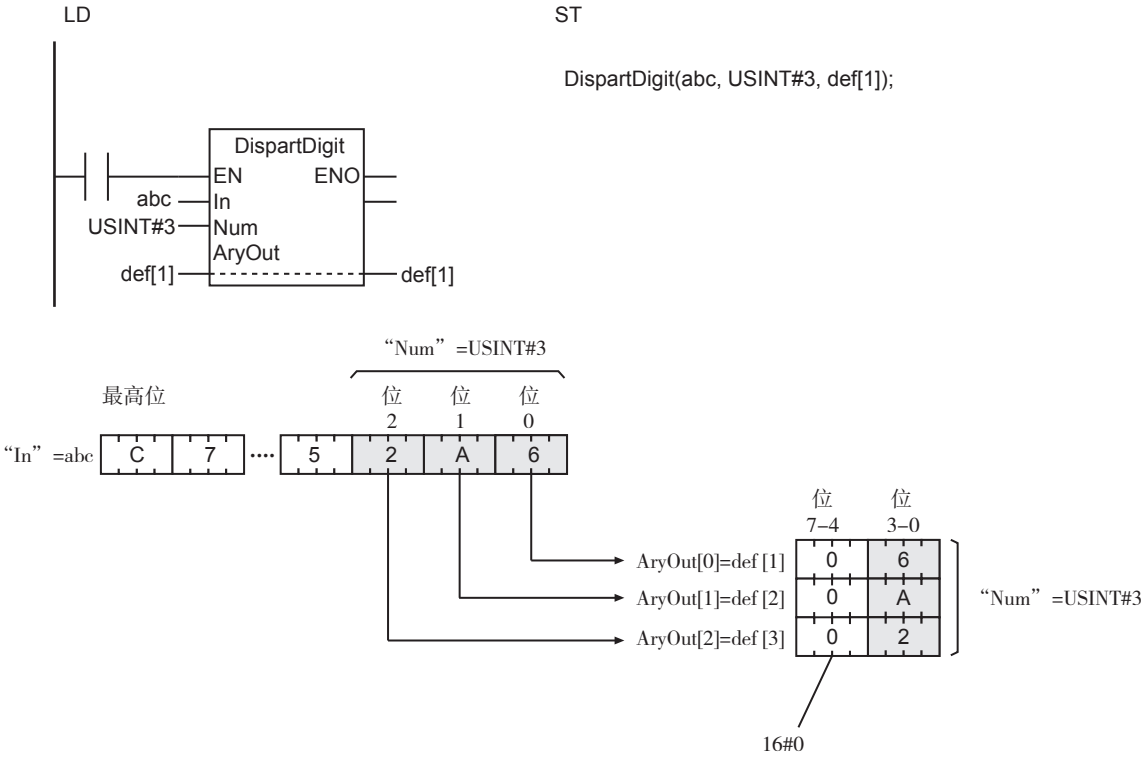
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Num						○														
AryOut[ ]数组		○																		
Out	○																			

功能

以4位为单位(数位单位)将分离对象的 “In”进行分离，保存在分离结果数组AryOut[ ]中。

首先，以4位为单位分离 “In”。然后，将最低位的4位保存在AryOut[0]中。此时，在BYTE型的AryOut[0]的位4到位7中添加16#0。
对分离位数 “Num”数进行相同处理。

“Num”=USINT#3时的示例如下所示。



参考

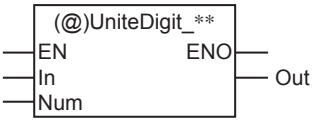
按每4位组合数组的各元素时，请使用 “UniteDigit_**指令(P.2–461)”。

使用注意事项

- “Num”的值为0时，AryOut[ ]的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Num”的值超过有效范围时。
  - “Num”的值超过AryOut[ ]的数组区域时。

UniteDigit_**

将以4位为单位的数据组合成位串。

指令	名称	FB/FUN	图形表现	ST表现
UniteDigit_**	4位组合组	FUN	 **为位串的数据类型名称	Out:=UniteDigit_**(In, Num); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	组合对象数组	输入	待组合的数组	遵从数据类型	–	(*)
Num	组合位数		待组合的位数	0 ~ “Out”的位数		1
Out	组合结果	输出	组合结果的位串	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○																		
Num						○														
Out		○	○	○	○															

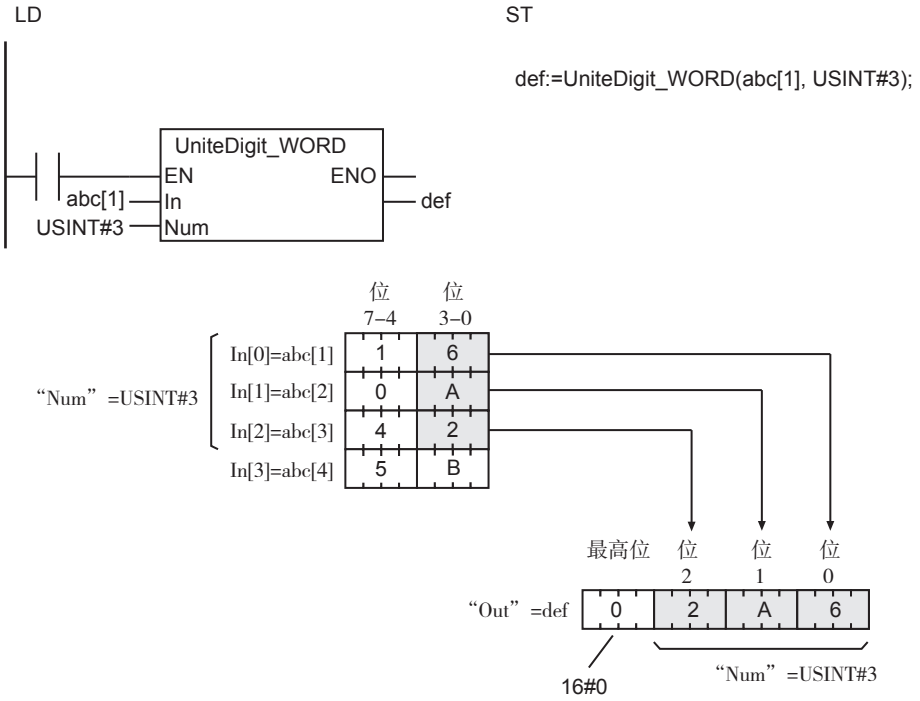
功能

按低位4位将组合对象数组In[ ]的各元素(1个数位=4位)进行组合，生成组合结果的位串 “Out”。

待组合的数组元素数由组合位数 “Num”指定。首先，将In[0] ~ In[“Num”–1]的每低位4位进行组合，生成 “Num”位的位串。其高位上，将组合{(“Out”的位数)– “Num”个}的16#0的位串作为 “Out”。

指令名称因 “Out”的数据类型而异。例如，“Out”为WORD型时，指令名称为UniteDigit_WORD。

UniteDigit_WORD指令下，“Num”=USINT#3时的示例如下所示。



参考

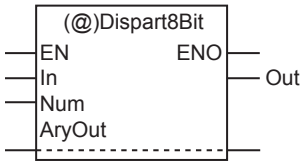
以4位为单位分离位串时，请使用 “DispartDigit指令(P.2–459)”。

使用注意事项

- “Num”的值为0时，“Out”的值为0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Num”的值超过有效范围时。
  - “Num”的值超过In[ ]的数组区域时。

Dispart8Bit

以1字节为单位分离位串。

指令	名称	FB/FUN	图形表现	ST表现
Dispart8Bit	以字节为单位的数据分离	FUN		Dispart8Bit(In, Num, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	分离对象	输入	分离对象的位串	遵从数据类型	–	(*)
Num	分离字节数		待分离的字节数	0～“In”的字节数		1
AryOut[ ]数组	分离结果数组	输入输出	分离结果数组	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

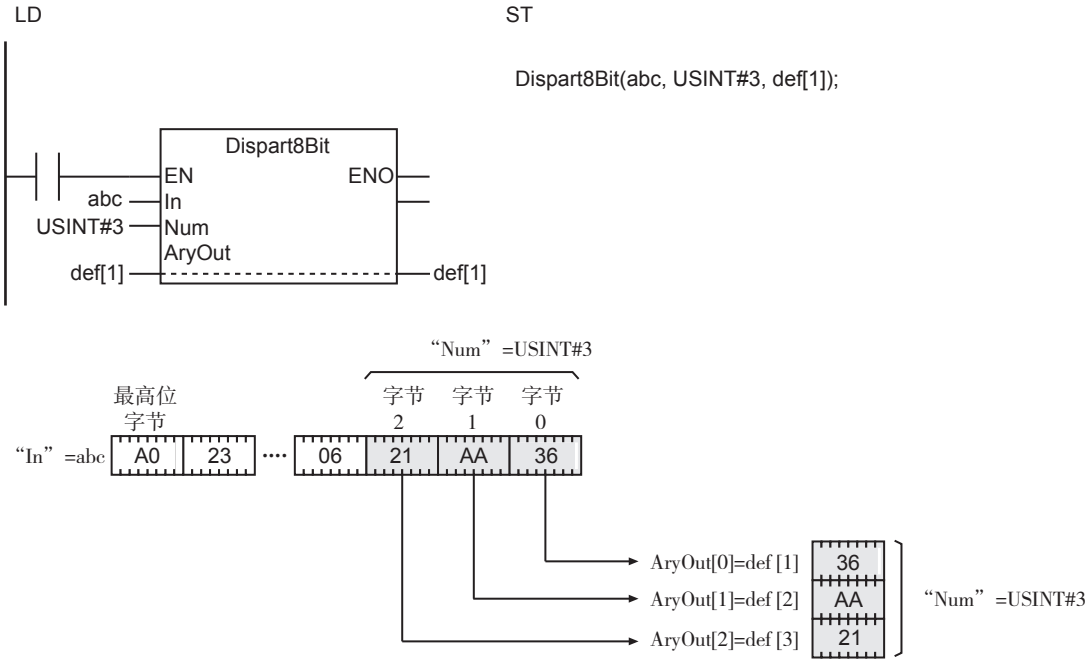
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Num						○														
AryOut[ ]数组		○																		
Out	○																			

功能

以1字节为单位将分离对象 “In”进行分离，保存在分离结果数组AryOut[ ]中。

首先，以1字节为单位分离 “In”。然后，将最低位字节保存在AryOut[0]中。再将下1个字节保存在AryOut[1]中。对分离字节数 “Num”数进行相同处理。

“Num”=USINT#3时的示例如下所示。



参考

按每个字节组合数组的各元素时，请使用 “Unite8Bit_**指令(P.2-465)”。

使用注意事项

- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE，AryOut[ ]不变。
  - “Num”的值超过有效范围时。
  - “Num”的值超过AryOut[ ]的数组区域时。

Unite8Bit_**

将以1字节为单位的数据组合成位串。

指令	名称	FB/FUN	图形表现	ST表现
Unite8Bit_**	以字节为单位的数据组合组	FUN	(@)Unite8Bit_** EN ENO In Out Num **为位串的数据类型名称	Out:=Unite8Bit_**(In, Num); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	组合对象数组	输入	待组合的数组	遵从数据类型	–	(*)
Num	组合字节数		待组合的字节数	0～“Out”的字节数		1
Out	组合结果	输出	组合结果的位串	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○																		
Num						○														
Out		○	○	○	○															

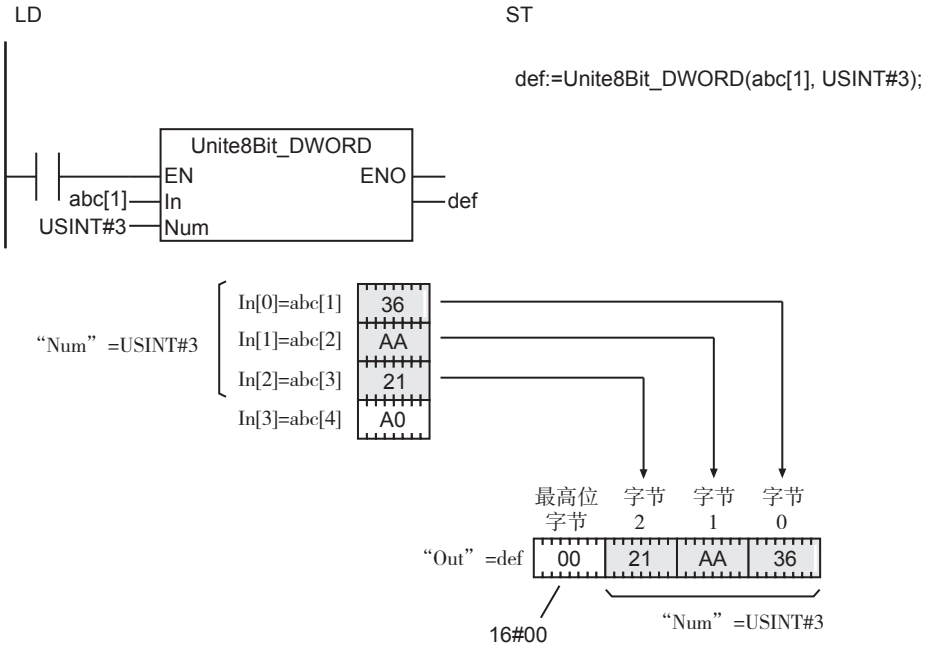
功能

将组合对象数组In[ ]的各元素进行组合，生成组合结果的位串 “Out”。

待组合的数组元素数由组合字节数 “Num” 指定。首先，将In[0]～In[“Num” –1]进行组合，生成 “Num” 字节的位串。其高位上，将组合{(“Out” 的字节数)– “Num” 个}的16#00的位串作为 “Out”。

指令名称因 “Out” 的数据类型而异。例如，“Out” 为DWORD型时，指令名称为Unite8Bit_DWORD。

Unite8Bit_DWORD指令下，“Num”=USINT#3时的示例如下所示。



参考

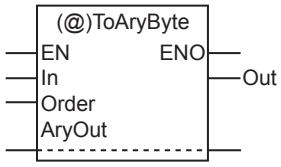
以1字节为单位分离位串时，请使用 “Dispart8Bit指令(P.2–463)”。

使用注意事项

- “Num”的值为0时，“Out”的值为0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Num”的值超过有效范围时。
  - “Num”的值超过In[ ]的数组区域时。

ToAryByte

以1字节为单位分割变量后，保存至BYTE型数组。

指令	名称	FB/FUN	图形表现	ST表现
ToAryByte	转换为字节数组	FUN		Out:=ToAryByte(In, Order, AryOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换对象	遵从数据类型	–	(*)
Order	转换顺序		待转换的顺序	_LOW_HIGH, _HIGH_LOW		_LOW _HIGH
AryOut[ ] 数组	转换结果数组	输入输出	转换结果数组	遵从数据类型	–	–
Out	转换结果元素数	输出	转换结果的元素数	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素																			
Order	枚举体_eBYTE_ORDER 枚举元素参阅功能说明																			
AryOut[ ] 数组		○																		
Out							○													

功能

以 1 字节为单位分割转换对象 “In”，从转换结果数组 AryOut[ ] 的 AryOut[0] 起依次保存。再将已保存在 AryOut[ ]中的元素数保存在转换结果元素数 “Out”中。

以1字节为单位对 “In”的值进行转换时的顺序由转换顺序 “Order”指定。“Order”的数据类型为枚举体_eBYTE_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_LOW_HIGH	低位字节在前，高位字节在后
_HIGH_LOW	高位字节在前，低位字节在后

“In”的数据类型为2字节以上的大小时

“In”的数据类型为2字节以上的大小时，以1字节为单位分割 “In”，保存在AryOut[ ]中。
2字节以上大小的数据类型如下所示。

种类	数据类型
位串	WORD、DWORD、LWORD
整数	UINT、UDINT、ULINT、INT、DINT、LINT
实数	REAL、LREAL
时刻、持续时间、日期、字符串	TIME、DATE、TOD、DT、2字节以上的STRING
其它	枚举体、 所有元素的合计大小为2字节以上的整个数组、 2字节以上的数组的1个元素、 所有结构要素的合计大小为2字节以上的整个结构体、 2字节以上的结构体的1个结构要素

处理步骤如下所示。

1 首先，以1个字(2字节)为单位分割 “In”的值。

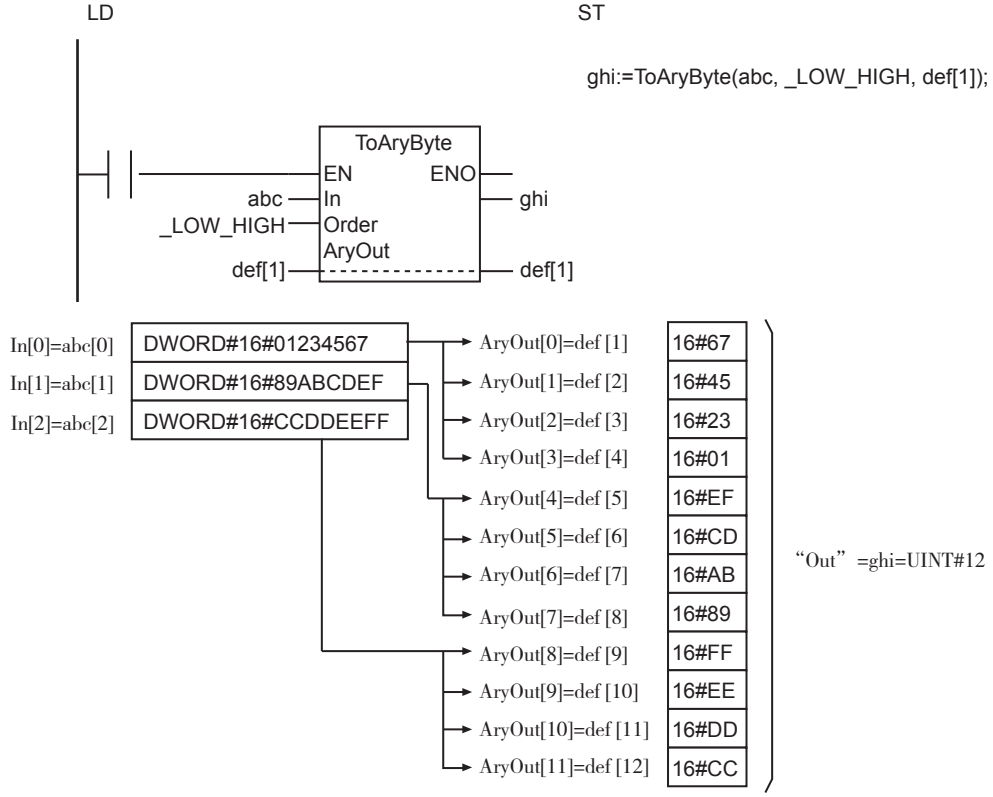
2 再以1字节为单位分割其中最低位的字(2字节)。

3 如果 “Order”的值为_LOW_HIGH，则将其中的低位字节保存在AryOut[0]中，将高位字节保存在AryOut[1]中。如果 “Order”的值为_HIGH_LOW，则将高位字节保存在AryOut[0]中，将低位字节保存在AryOut[1]中。

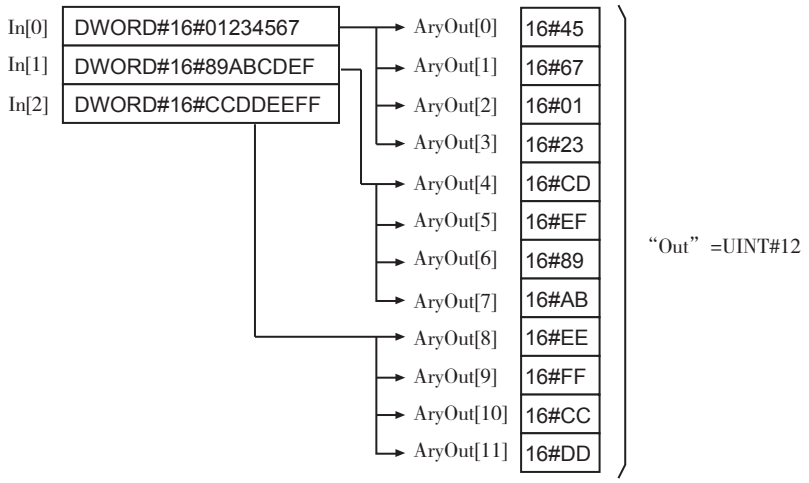
4 同样地，以1字节为单位分割下1个字，保存在AryOut[2]和AryOut[3]中。

5 直至 “In”值的最后均进行相同的处理。“In”为数组时，直至 “In”最后的元素均进行相同的处理。

“In”为元素数为3的DWORD型整个数组，“Order”=_LOW_HIGH时的示例如下所示。



“In”同上，“Order”=_HIGH_LOW时的示例如下所示。



“In”的数据类型为1字节的大小时

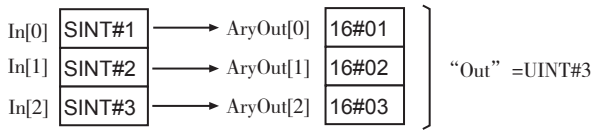
“In”的数据类型为1字节的大小时，仍以1字节为单位将 “In”保存在AryOut[ ]中。
1字节大小的数据类型如下所示。

种类	数据类型
位串	BYTE
整数	USINT、SINT
实数	无
时刻、持续时间、日期、字符串	1字节的STRING
其它	所有元素的合计大小为1字节的整个数组、 1字节的数组的1个元素、 所有结构要素的合计大小为1字节的整个结构体、 1字节的结构体的1个结构要素

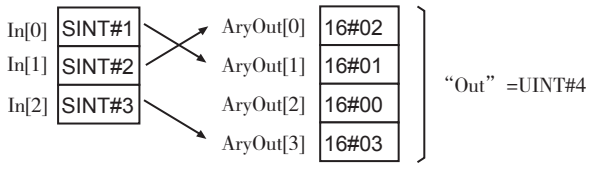
保存方法如下所示。

“Order”的值	“In”是否为数组	保存至AryOut[ ]的方法
_LOW_HIGH	非数组	将 “In”的值保存在AryOut[0]中。
	数组	将In[i]的值保存在AryOut[i]中。
_HIGH_LOW	非数组	将 “In”的值保存在AryOut[1]中。 将16#00保存在AryOut[0]中。
	数组	将In[i](i为偶数)保存在AryOut[i+1]中。 将In[i](i为奇数)保存在AryOut[i−1]中。 如果In[ ]的元素数n为奇数，则最后将16#00保存在AryOut[n−1]中。

“In”为元素数为3的SINT型数组，“Order”=_LOW_HIGH时的示例如下所示。



“In”同上，“Order”=_HIGH_LOW时的示例如下所示。

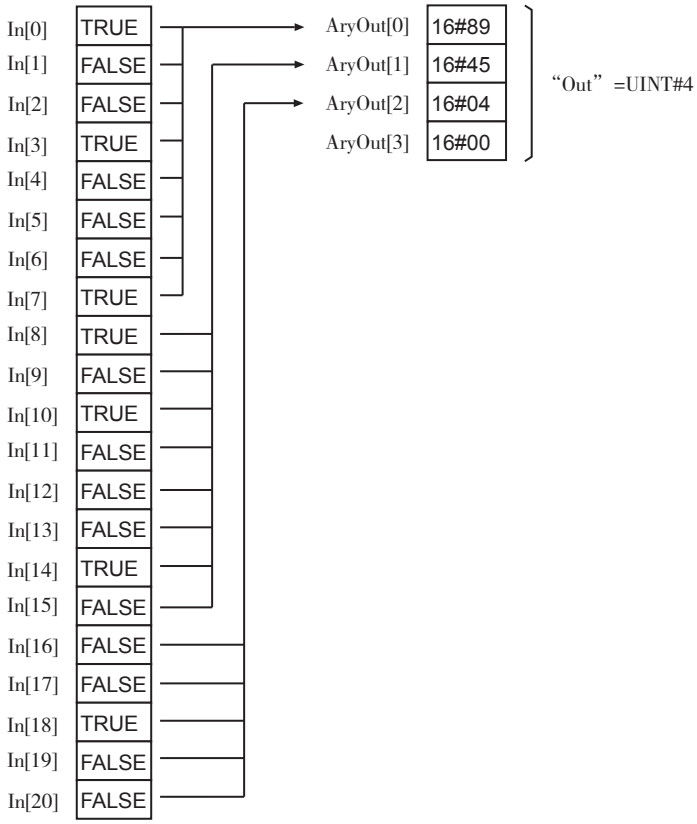


“In”的数据类型为BOOL时

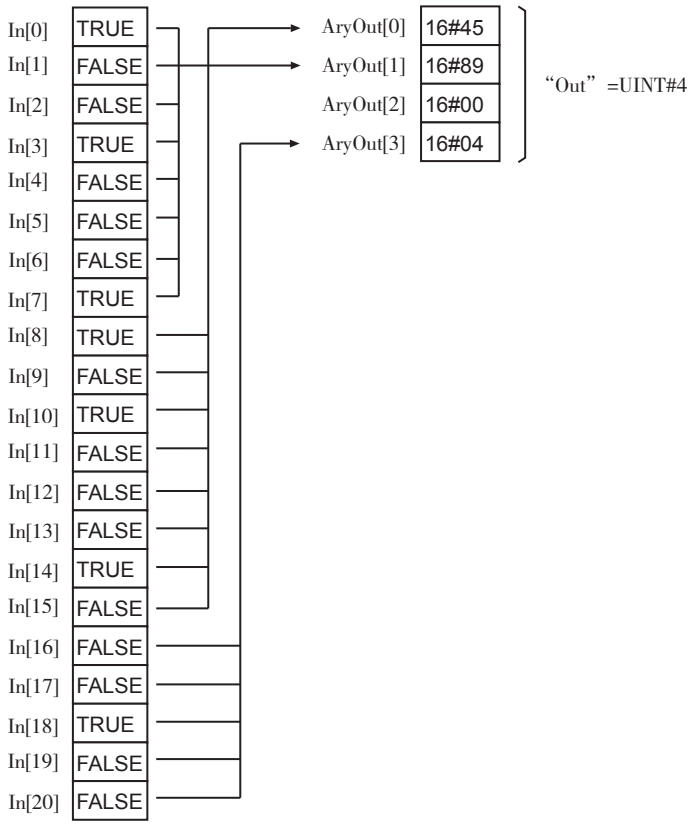
“In”的数据类型为BOOL(1位)时，保存至AryOut[ ]的方法如下所示。

“Order”的值	“In”是否为数组	保存至AryOut[ ]的方法
_LOW_HIGH	非数组	将 “In”的值与16#00的逻辑和保存在AryOut[0]中。
	数组	连接In[0]～In[7]的值，保存在AryOut[0]中。 连接In[8]～In[15]的值，保存在AryOut[1]中。之后以相同的方法保存。 In[ ]不足8个时，为高位添加FALSE。 务必将 “Out”的值设为偶数。不足的位的值均为FALSE。
_HIGH_LOW	非数组	将 “In”的值与16#00的逻辑和保存在AryOut[1]中。 将16#00保存在AryOut[0]中。
	数组	连接In[0]～In[7]的值，保存在AryOut[1]中。 连接In[8]～In[15]的值，保存在AryOut[0]中。 之后以相同的方法保存。 “Out”的值必定为偶数。不足的位的值均为FALSE。

“In”为BOOL型，数组的元素数为21，“Order”=_LOW_HIGH时的示例如下所示。



“In”同上，“Order”=_HIGH_LOW时的示例如下所示。

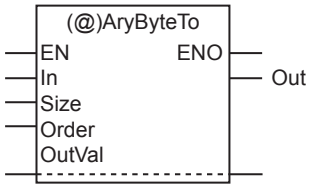


使用注意事项

- 请务必将传输至 “In”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “In”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- “In”为STRING型时，不将字符串转换为数值。将变量的内容视为位串，转换为字节数组。
- “In”为结构体时，根据构成的不同，各结构要素间的调整用区域可能会插入AryOut[ ]中。
- “Order”的值为_HIGH_LOW，“In”的合计大小为奇数字节时，在 “In”的最后添加16#00，将其设为偶数字节，再开始转换。
- 以下情况时会发生异常。ENO变为FALSE，“Out”和AryOut[ ]不变。
  - “Order”的值超过有效范围时。
  - 转换结果超过AryOut[ ]的数组区域时。

AryByteTo

组合BYTE型数组元素后，保存至变量。

指令	名称	FB/FUN	图形表现	ST表现
AryByteTo	从字节数组转换	FUN		AryByteTo(In, Size, Order, OutVal);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	转换对象数组	输入	待转换的数组	遵从数据类型	–	(*)
Size	转换元素数		In[ ]的待转换的元素数			1
Order	转换顺序		待转换的顺序	_LOW_HIGH, _HIGH_LOW		_LOW_HIGH
OutVal	转换结果	输入输出	转换结果	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○																		
Size							○													
Order	枚举体_eBYTE_ORDER 枚举元素参阅功能说明																			
OutVal	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素																			
Out	○																			

功能

按照转换结果 “OutVal”的数据类型的大小对转换对象数组In[ ]开头的 “Size”个元素进行组合，并保存在 “OutVal”中。

组合In[ ]的元素时的顺序由转换顺序 “Order”指定。“Order”的数据类型为枚举体_eBYTE_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_LOW_HIGH	低位字节在前，高位字节在后
_HIGH_LOW	高位字节在前，低位字节在后

“OutVal”的数组类型为2字节以上的大小时

“OutVal”的数据类型为2字节以上的大小时，对In[ ]进行组合的仅为 “OutVal”的数据类型的大小部分，保存在 “OutVal”中。
2字节以上大小的数据类型如下所示。

种类	数据类型
位串	WORD、DWORD、LWORD
整数	UINT、UDINT、ULINT、INT、DINT、LINT
实数	REAL、LREAL
时刻、持续时间、日期、字符串	TIME、DATE、TOD、DT、2字节以上的STRING
其它	枚举体、 所有元素的合计大小为2字节以上的整个数组、 2字节以上的数组的1个元素、 所有结构要素的合计大小为2字节以上的整个结构体、 2字节以上的结构体的1个结构要素

处理步骤如下所示。

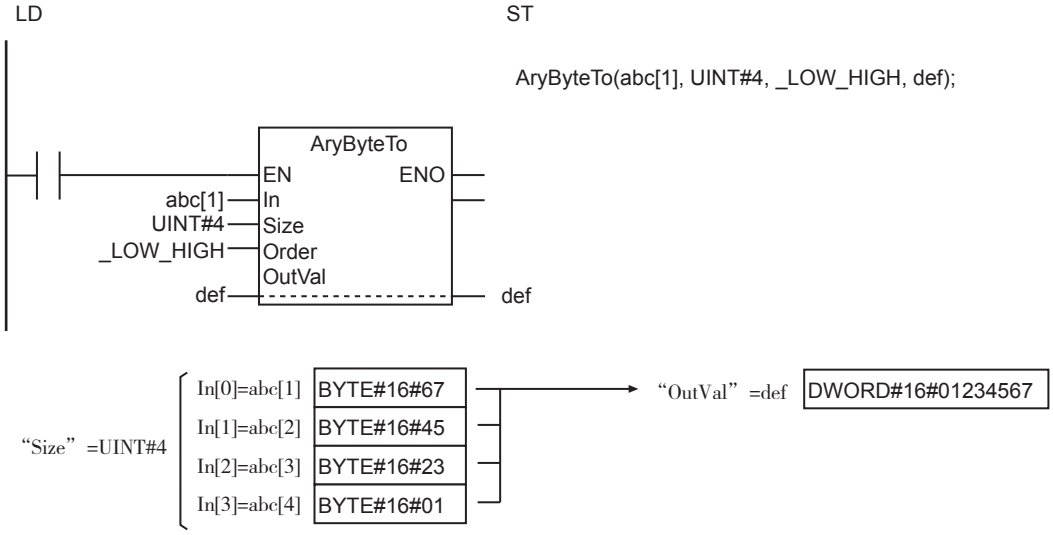
1 按照“Order”的值组合In[0]和In[1]，生成1字(2字节)数据。如果“Order”的值为_LOW_HIGH，则高位字节为In[1]，低位字节为In[0]。如果 “Order”的值为_HIGH_LOW，则高位字节为In[0]，低位字节为In[1]。

2 同样地，In[2]和In[3]之后也进行组合，生成多个1字数据。

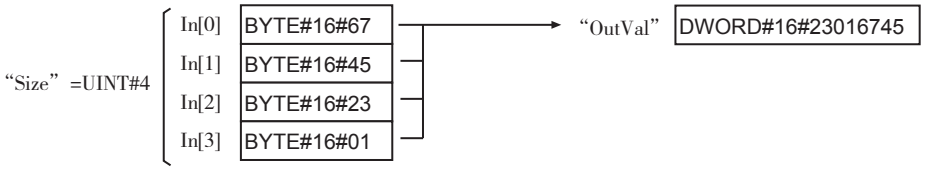
3 对已生成的多个1字数据再进行组合，以符合 “OutVal”的数据类型的大小。例如，“OutVal”为DWORD型时，组合4个1字数据。

4 将已生成的数据保存在 “OutVal”中。

“OutVal”为DWORD型，“Size”=UINT#4，“Order”=_LOW_HIGH时的示例如下所示。



“OutVal”同上，“Size”=UINT#4，“Order”=_HIGH_LOW时的示例如下所示。



“OutVal”的数据类型为1字节的大小时

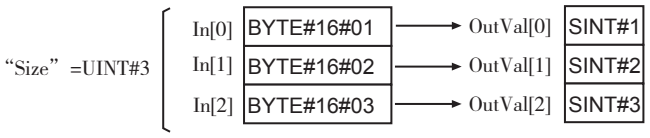
“OutVal”的数据类型为1字节的大小时，仍以1字节为单位将In[ ]保存在 “OutVal”中。
1字节大小的数据类型如下所示。

种类	数据类型
位串	BYTE
整数	USINT、SINT
实数	无
时刻、持续时间、日期、字符串	1字节的STRING
其它	所有元素的合计大小为1字节的整个数组、 1字节的数组的1个元素、 所有结构要素的合计大小为1字节的整个结构体、 1字节的结构体的1个结构要素

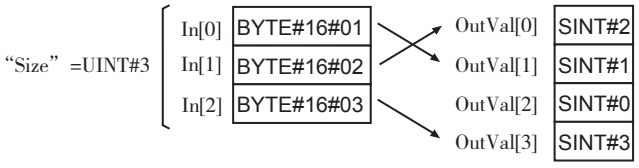
保存方法如下所示。

“Order”的值	“OutVal”是否为数组	保存至 “OutVal”的方法
_LOW_HIGH	非数组	将In[0]的值保存在 “OutVal”中。
	数组	将In[i]的值保存在OutVal[i]中。
_HIGH_LOW	非数组	将In[1]的值保存在 “OutVal”中。
	数组	将In[i](i为偶数)保存在OutVal[i+1]中。 将In[i](i为奇数)保存在OutVal[i−1]中。 如果 “Size”的值为奇数，直至OutVal[“Size”]均保存， OutVal[“Size”−1]中保存16#00。

“OutVal”为元素数为3的SINT型数组，“Size”=UINT#3、“Order”=_LOW_HIGH时的示例如下所示。



“OutVal”和 “Size”同上，“Order”=_HIGH_LOW时的示例如下所示。

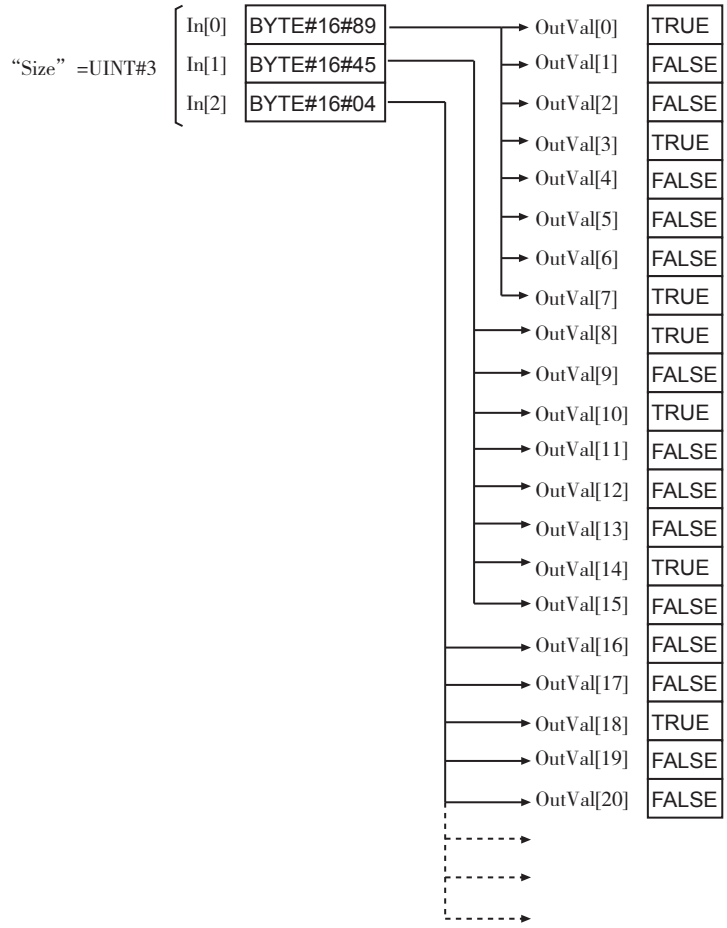


“OutVal”的数据类型为BOOL时

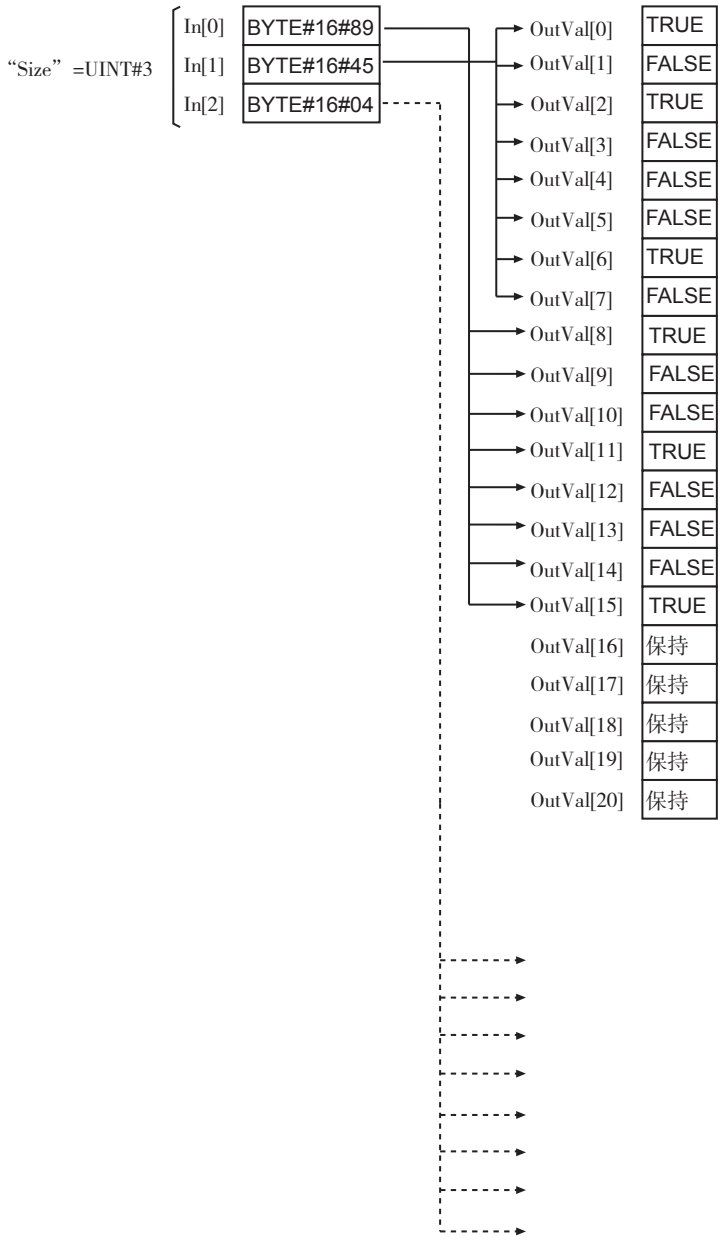
“OutVal”的数据类型为BOOL(1位)时，保存至 “OutVal”的方法如下所示。

“Order”的值	“OutVal”是否为数组	保存至 “OutVal”的方法
_LOW_HIGH	非数组	将In[0]的位0的值保存在 “OutVal”中。
	数组	分解In[0]的值，保存在OutVal[0] ~ OutVal[7]中。 分解In[1]的值，保存在OutVal[8] ~ OutVal[15]中。 之后以相同的方法保存。 剩余的位舍弃。
_HIGH_LOW	非数组	将In[1]的位0的值保存在 “OutVal”中。
	数组	分解In[0]的值，保存在OutVal[8] ~ OutVal[15]中。 分解In[1]的值，保存在OutVal[0] ~ OutVal[7]中。 之后以相同的方法保存。 剩余的位舍弃。

OutVal[ ]为元素数为21的BOOL型数组，“Size”=UINT#3，” Order “=_LOW_HIGH时的示例如下所示。



OutVal[ ]和 “Size”同上，“Order”=_HIGH_LOW时的示例如下所示。



使用注意事项

- “OutVal”为整个结构体时，根据构成的不同，部分In[ ]的值可能会插入各要素间的调整用区域中。
- “Size”的值为0时，“Out”的值为TRUE，“OutVal”不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“OutVal”不变。
  - “Order”的值超过有效范围时。
  - “Size”的值超过In[ ]的元素数时。

SizeOfAry

获取数组元素数。

指令	名称	FB/FUN	图形表现	ST表现
SizeOfAry	获取数组元素数	FUN	(@)SizeOfAry EN ENO In Out	Out:=SizeOfAry(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	数组	输入	数组	遵从数据类型	–	(*)
Out	元素数	输出	元素数	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
Out							○													

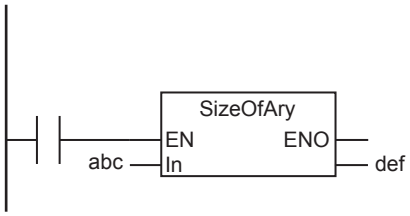
功能

获取数组In[ ]的元素数。
输入参数请指定像array那样的数组名称，而不是像array[0]那样的数组元素名称。

动作示例如下所示。

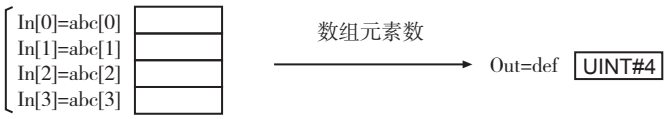
名称	数据类型
abc	ARRAY[0..3] OF INT

LD



ST

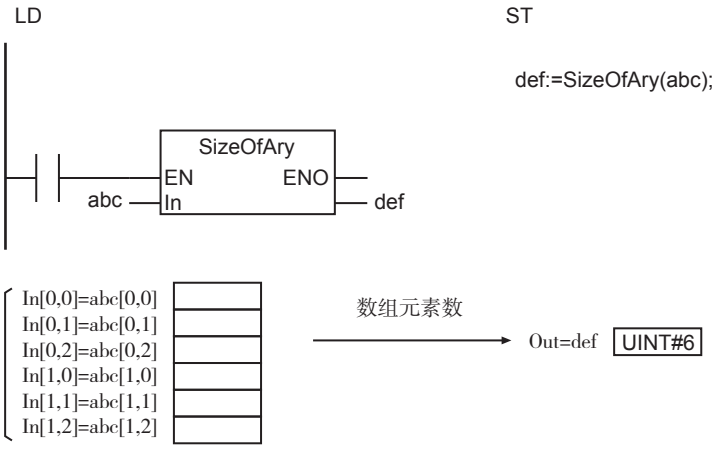
```
def:=SizeOfAry(abc);
```



参考

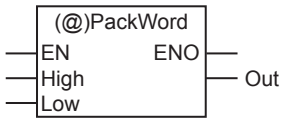
In[ ]也可为2维以上的数组。此时，“Out”中保存In[ ]的所有元素数。例如，传输至In[ ]的输入参数的数组为ARRAY[0..1,0..2]时，“Out”的值为UINT#6。

名称	数据类型
abc	ARRAY[0..1,0..2] OF BOOL



PackWord

将2个单字节数据合并成1个2字节数据。

指令	名称	FB/FUN	图形表现	ST表现
PackWord	合并2字节	FUN		Out:=PackWord(High,Low);

版本相关信息

本指令可用于CPU单元Ver.1.12以上且Sysmac Studio Ver.1.16以上。

变量

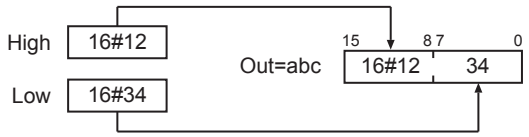
	名称	输入/输出	内容	有效范围	单位	初始值
High	字节数据H	输入	位15–8中保存的字节数据	遵从数据类型	–	0
Low	字节数据L		位7–0中保存的字节数据	遵从数据类型	–	0
Out	合并数据	输出	2字节数据	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
High		○																		
Low		○																		
Out			○																	

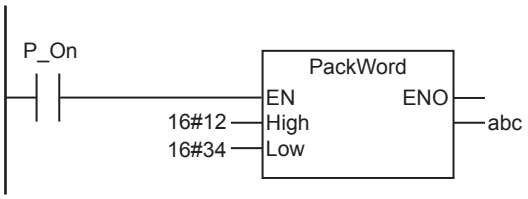
功能

将2个单字节数据合并成1个2字节数据。
将 “High”指定的数据保存在位15–8中，将 “Low”指定的数据保存在位7–0中。

“High”=16#12、“Low”=16#34时的示例如下所示。
变量abc为16#1234。



- LD

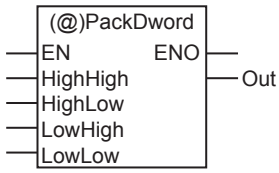


- ST

```
abc:=PackWord(16#12,16#34);
```

PackDword

将4个单字节数据合并成1个4字节数据。

指令	名称	FB/FUN	图形表现	ST表现
PackDword	合并4字节	FUN		Out:=PackDword(HighHigh, HighLow, LowHigh, LowLow);

版本相关信息

本指令可用于CPU单元Ver.1.12以上且Sysmac Studio Ver.1.16以上。

变量

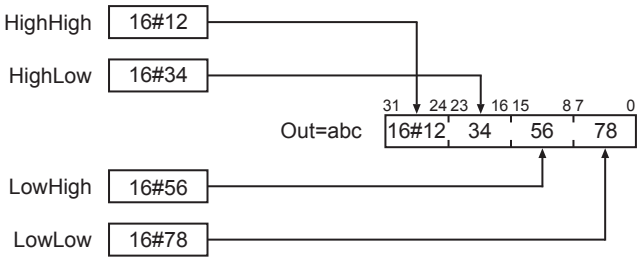
	名称	输入/输出	内容	有效范围	单位	初始值
HighHigh	字节数据HH	输入	位31–24中保存的字节数据	遵从数据类型	–	0
HighLow	字节数据HL		位23–16中保存的字节数据	遵从数据类型	–	0
LowHigh	字节数据LH		位15–8中保存的字节数据	遵从数据类型	–	0
LowLow	字节数据LL		位7–0中保存的字节数据	遵从数据类型	–	0
Out	合并数据	输出	4字节数据	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
HighHigh		○																		
HighLow		○																		
LowHigh		○																		
LowLow		○																		
Out				○																

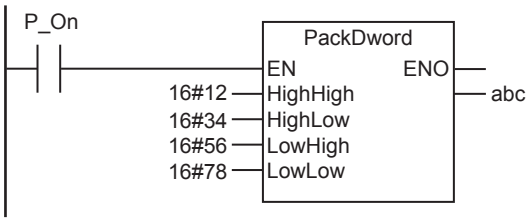
功能

将4个单字节数据合并成1个4字节数据。
将 “HighHigh”指定的数据保存在位31–24中，将 “HighLow”指定的数据保存在位23–16中，将“LowHigh”指定的数据保存在位15–8中,将 “LowLow”指定的数据保存在位7–0中。

“HighHigh”=16#12、“HighLow”=16#34、“LowHigh”=16#56、“LowLow”=16#78时的示例如下所示。
变量abc为16#12345678。



- LD



- ST

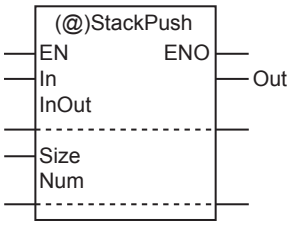
```
abc:=PackDword(16#12,16#34,16#56,16#78);
```

堆栈/表指令

指令	名称	页码
StackPush	堆栈数据保存	2–484
StackFIFO/StackLIFO	先入先出/后入先出	2–493
StackIns	堆栈数据插入	2–496
StackDel	堆栈数据删除	2–498
RecSearch	记录检索	2–500
RecRangeSearch	范围指定记录检索	2–505
RecSort	记录排序	2–510
RecNum	记录数获取	2–515
RecMax/RecMin	记录最大值检索/ 记录最小值检索	2–517

StackPush

将值保存至堆栈中。

指令	名称	FB/FUN	图形表现	ST表现
StackPush	堆栈数据保存	FUN		StackPush(In, InOut, Size, Num);

变量

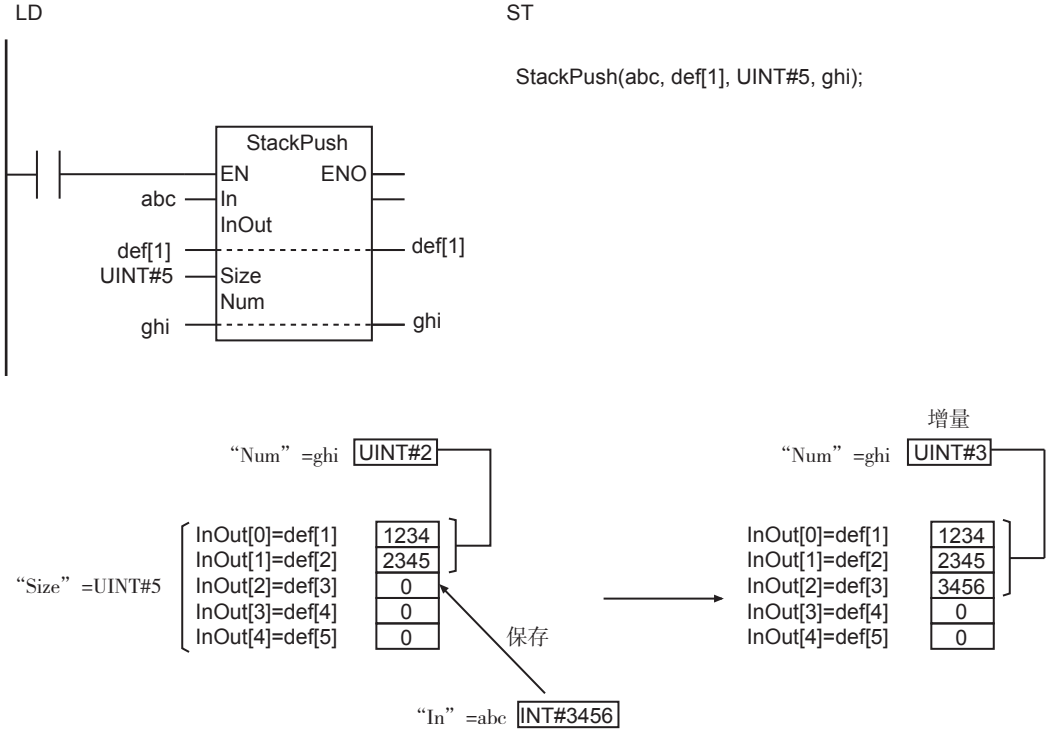
	名称	输入/输出	内容	有效范围	单位	初始值
In	输入值	输入	输入至堆栈的值、结构体、结构体的1个结构要素	遵从数据类型	–	–
Size	堆栈的元素数		堆栈的数组元素数			1
InOut[]数组	堆栈数组	输入输出	构成堆栈的数组	遵从数据类型	–	–
Num	堆栈的保存元素数		保存在堆栈中的元素数			
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个结构体、结构体的1个结构要素																			
Size							○													
InOut[ ]数组	将与 “In”相同的数据类型作为元素的数组																			
Num							○													
Out	○																			

功能

视为当前已将保存元素数 “Num”个元素保存至堆栈数组InOut[]。将输入值 “In”覆盖下一元素InOut[“Num”]。
然后，对 “Num”进行增量。
在堆栈的元素数 “Size”中指定InOut[]中用作堆栈的元素数。

“Size”=UINT#5、“Num”=UINT#2时的示例如下所示。



参考

抽取堆栈的最低位或最高位的值时，请使用 “StackFIFO/StackLIFO指令(P.2–493)”。

使用注意事项

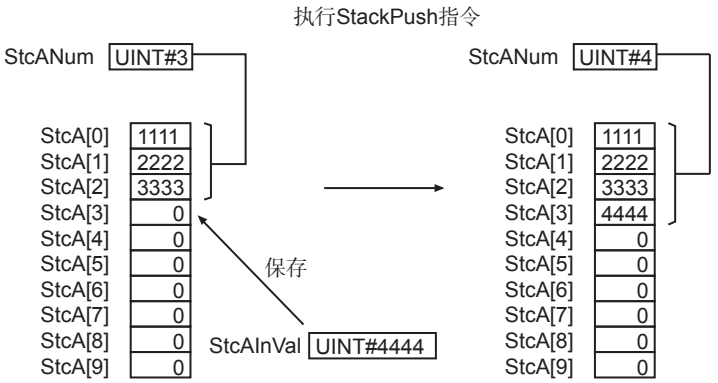
- 请将 “In”和InOut[]的元素的数据类型设为相同。如果不同，则编连时会发生异常。
- 将数组中的元素传输至InOut[]时，该元素之后为处理对象。
- “Size”的值为0时，InOut[]、“Num”的值不变。
- 请务必将传输至 “In”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “In”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，InOut[]不变。
  - “Size”的值不为0，“Num” ≥ “Size”时。
  - “Size”的值超过InOut[]的数组区域时。
  - “In”、InOut[]为STRING型，“In”的字节数超过InOut[]的大小时。

示例程序

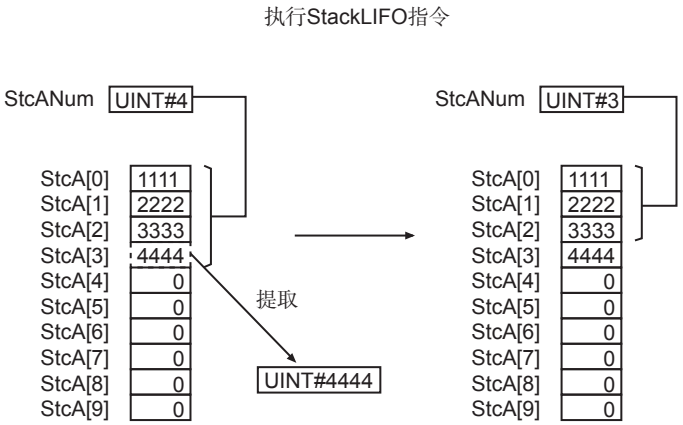
将数组变量StcA[0..9]作为堆栈。首先，将3个数据(UINT#1111、UINT#2222、UINT#3333)保存至堆栈。

StcA[0]	1111
StcA[1]	2222
StcA[2]	3333
StcA[3]	0
StcA[4]	0
StcA[5]	0
StcA[6]	0
StcA[7]	0
StcA[8]	0
StcA[9]	0

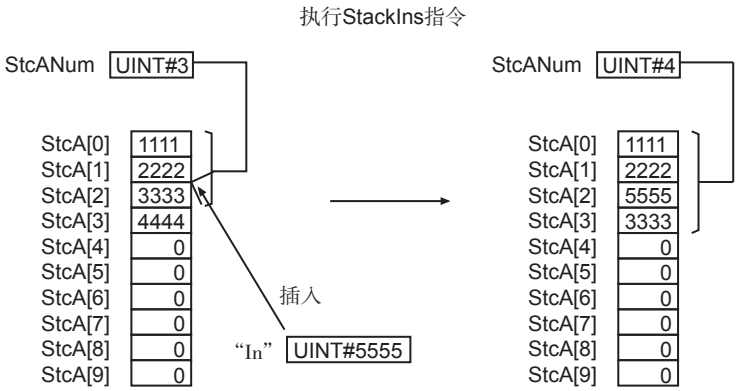
使用StackPush指令，将1个新数据(UINT#4444)保存至堆栈的最高位 StcA[3]。堆栈内的数据数变为4个。



然后，使用StackLIFO指令，抽取堆栈的最高位 StcA[3]的1个数据。堆栈的数据数变为3个。

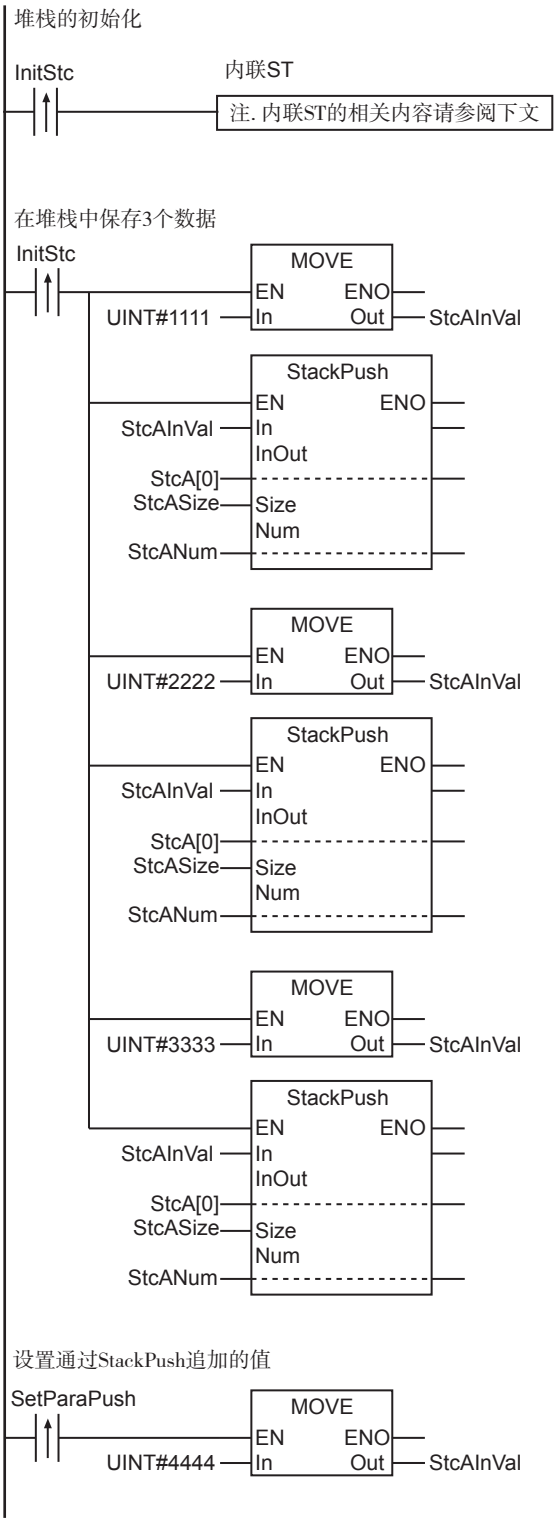


最后，使用StackIns指令在StcA[1]与StcA[2]之间插入1个数据(UINT#5555)。堆栈的数据数变为4个。

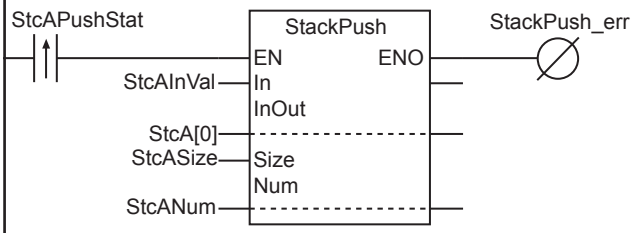


LD

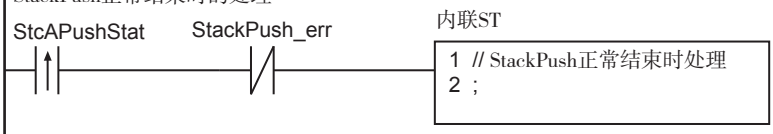
名称	数据类型	初始值	注释
InitStc	BOOL	FALSE	堆栈的初始化条件
StcANum	UINT	0	堆栈的保存元素数
StcA	ARRAY[0..9] OF UINT	[10(0)]	堆栈数组
StcASize	UINT	0	堆栈的元素数
SetParaPush	BOOL	FALSE	设置StcAInVal的执行条件
StcAInVal	UINT	0	通过StackPush追加的值
StcAPushStat	BOOL	FALSE	StackPush的执行条件
StackPush_err	BOOL	FALSE	StackPush的异常标志
StcALIFOStat	BOOL	FALSE	StackLIFO的执行条件
StcAOutVal	UINT	0	通过StackLIFO抽取的值
StackLIFO_err	BOOL	FALSE	StackLIFO的异常标志
SetParaIns	BOOL	FALSE	设置StcAInsVal和StcAOffset的执行条件
StcAInsVal	UINT	0	通过StackIns插入的值
StcAOffset	UINT	0	StackIns的偏置值
StcAInsStat	BOOL	FALSE	StackIns的执行条件
StackIns_err	BOOL	FALSE	StackIns的异常标志


堆栈的初始化
InitStc
内联ST
注. 内联ST的相关内容请参阅下文
在堆栈中保存3个数据
InitStc
MOVE
EN
ENO
UINT#1111
In
Out
StcAInVal
StackPush
EN
ENO
StcAInVal
In
InOut
StcA[0]
StcASize
Size
Num
StcANum
MOVE
EN
ENO
UINT#2222
In
Out
StcAInVal
StackPush
EN
ENO
StcAInVal
In
InOut
StcA[0]
StcASize
Size
Num
StcANum
MOVE
EN
ENO
UINT#3333
In
Out
StcAInVal
StackPush
EN
ENO
StcAInVal
In
InOut
StcA[0]
StcASize
Size
Num
StcANum
设置通过StackPush追加的值
SetParaPush
MOVE
EN
ENO
UINT#4444
In
Out
StcAInVal

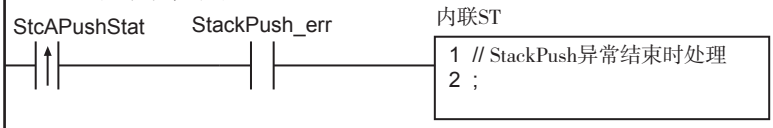
通过StackPush指令追加数据



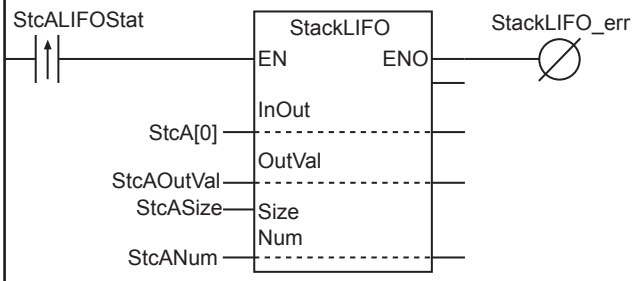
StackPush正常结束时的处理



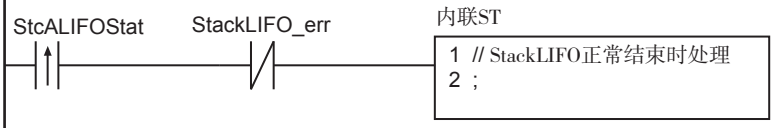
StackPush异常结束时的处理



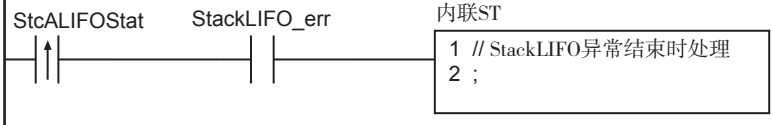
通过StackLIFO指令提取数据



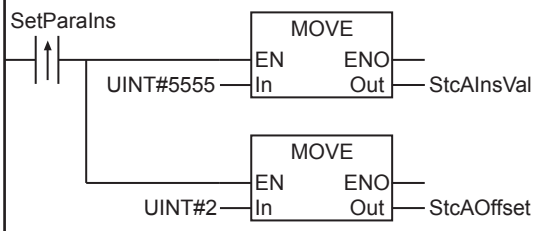
StackLIFO正常结束时的处理

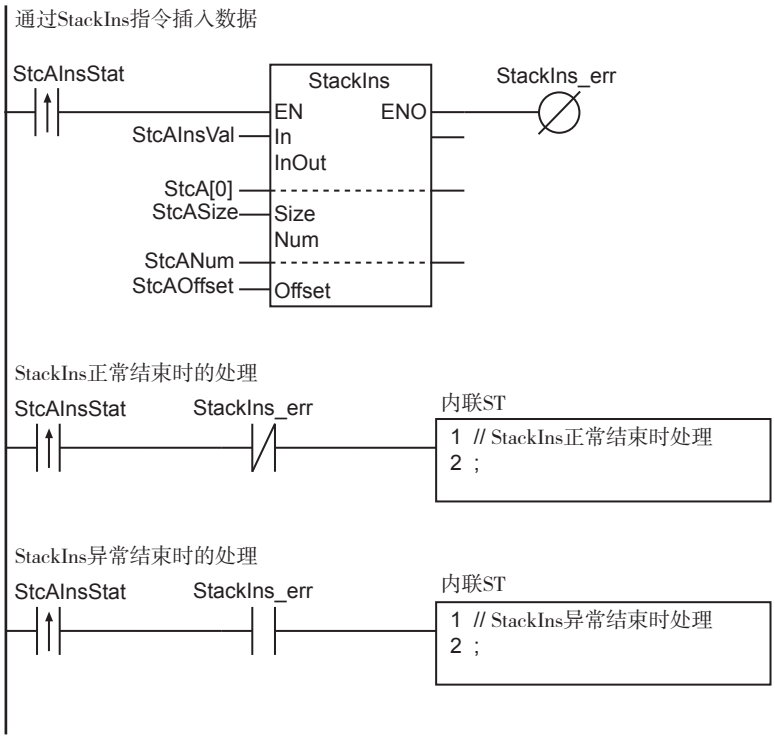


StackLIFO异常结束时的处理



设置通过StackIns插入的值和偏置





● 内联ST的内容

```
StcANum:=0;
Clear(StcA);
StcASize:=SizeOfAry(StcA);
```

ST

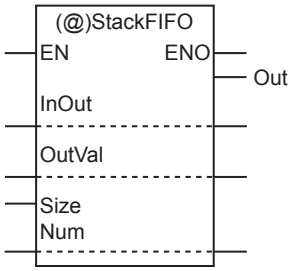
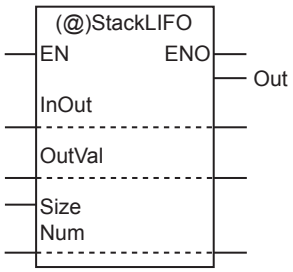
名称	数据类型	初始值	注释
InitStc	BOOL	FALSE	堆栈的初始化条件
preInitStc	BOOL	FALSE	上个任务周期的InitStc的值
StcANum	UINT	0	堆栈的保存元素数
StcA	ARRAY[0..9] OF UINT	[10(0)]	堆栈数组
StcASize	UINT	0	堆栈的元素数
StcAPushStat	BOOL	FALSE	StackPush的执行条件
preStcAPushStat	BOOL	FALSE	上个任务周期的StcAPushStat的值
StcAInVal	UINT	0	通过StackPush追加的值
StcAPush_OK	BOOL	FALSE	StackPush的正常结束标志
StcAPushNormalEnd	BOOL	FALSE	StackPush的正常结束时处理
StcAPushErrorEnd	BOOL	FALSE	StackPush的异常结束时处理
StcALIFOStat	BOOL	FALSE	StackLIFO的执行条件
preStcALIFOStat	BOOL	FALSE	上个任务周期的StcALIFOStat的值
StcAOutVal	UINT	0	通过StackLIFO抽取的值
StcALIFO_OK	BOOL	FALSE	StackLIFO的正常结束标志
StcALIFONormalEnd	BOOL	FALSE	StackLIFO的正常结束时处理
StcALIFOErrorEnd	BOOL	FALSE	StackLIFO的异常结束时处理
StcAInsStat	BOOL	FALSE	StackIns的执行条件
preStcAInsStat	BOOL	FALSE	上个任务周期的StcAInsStat的值
StcAInsVal	UINT	0	通过StackIns插入的值
StcAOffset	UINT	0	StackIns的偏置值
StcAIns_OK	BOOL	FALSE	StackIns的正常结束标志
StcAInsNormalEnd	BOOL	FALSE	StackIns的正常结束时处理
StcAInsErrorEnd	BOOL	FALSE	StackIns的异常结束时处理

```
// 堆栈的初始化
IF ( (InitStc=TRUE) AND (preInitStc=FALSE) ) THEN
   StcANum:=0;
   Clear(StcA);
   StcASize:=SizeOfAry(StcA);
END_IF;

// 在堆栈中保存3个数据
IF ( (InitStc=TRUE) AND (preInitStc=FALSE) ) THEN
   StackPush(In:=UINT#1111, InOut:=StcA[0], Size:=StcASize, Num:=StcANum);
   StackPush(In:=UINT#2222, InOut:=StcA[0], Size:=StcASize, Num:=StcANum);
   StackPush(In:=UINT#3333, InOut:=StcA[0], Size:=StcASize, Num:=StcANum);
END_IF;

preInitStc:=InitStc;

// 通过StackPush指令追加数据
IF ( (StcAPushStat=TRUE) AND (preStcAPushStat=FALSE) ) THEN
   StcAInVal:=UINT#4444;
   StackPush(
      In     :=StcAInVal,    // 待追加的值
      InOut:=StcA[0],        // 堆栈数组的首个元素
      Size   :=StcASize,     // 堆栈的元素数
      Num  :=StcANum,    // 堆栈的保存元素数
      ENO   =>StcAPush_OK); // 正常结束标志
   IF (StcAPush_OK=TRUE) THEN
      StcAPushNormalEnd:=TRUE; // 正常结束时处理
   ELSE
      StcAPushErrorEnd:=TRUE;   // 异常结束时处理
   END_IF;
END_IF;
```

```
preStcAPushStat:=StcAPushStat;
// 通过StackLIFO指令提取数据
IF ( (StcALIFOStat=TRUE) AND (preStcALIFOStat=FALSE) ) THEN
  StackLIFO(
    InOut  :=StcA[0],          // 堆栈数组的首个元素
    OutVal :=StcAOutVal,       // 从堆栈中提取的值
    Size   :=StcASize,         // 堆栈的元素数
    Num    :=StcANum,          // 堆栈的保存元素数
    ENO    =>StcALIFO_OK);     // 正常结束标志
  IF (StcALIFO_OK=TRUE) THEN
    StcALIFONormalEnd:=TRUE;   // 正常结束时处理
  ELSE
    StcALIFOErrorEnd:=TRUE;    // 异常结束时处理
  END_IF;
END_IF;
preStcALIFOStat:=StcALIFOStat;

// 通过StackIns指令插入数据
IF ( (StcAInsStat=TRUE) AND (preStcAInsStat=FALSE) ) THEN
  StcAInsVal:=UINT#5555;
  StcAOffset:=UINT#2;
  StackIns(
    In     :=StcAInsVal,       // 插入堆栈的值
    InOut  :=StcA[0],          // 堆栈数组的首个元素
    Size   :=StcASize,         // 堆栈的元素数
    Num    :=StcANum,          // 堆栈的保存元素数
    Offset:=StcAOffset,        // 待插入的偏置位置
    ENO    =>StcAIns_OK);      // 正常结束标志
  IF (StcAIns_OK=TRUE) THEN
    StcAInsNormalEnd:=TRUE;    // 正常结束时处理
  ELSE
    StcAInsErrorEnd:=TRUE;     // 异常结束时处理
  END_IF;
END_IF;
preStcAInsStat:=StcAInsStat;
```

StackFIFO/StackLIFO

StackFIFO :提取堆栈最低位的值。
StackLIFO :提取堆栈最高位的值。

指令	名称	FB/FUN	图形表现	ST表现
StackFIFO	先入先出	FUN		StackFIFO(InOut, OutVal, Size, Num);
StackLIFO	后入先出	FUN		StackLIFO(InOut, OutVal, Size, Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Size	堆栈的元素数	输入	堆栈的数组元素数	遵从数据类型	–	1
InOut[]数组	堆栈数组	输入输出	构成堆栈的数组	遵从数据类型	–	–
OutVal	输出值		从堆栈输出的值、整个结构体			
Num	堆栈的保存元素数		保存在堆栈中的元素数			
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Size							○													
InOut[]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
OutVal	与InOut[]的元素相同的数据类型																			
Num							○													
Out	○																			

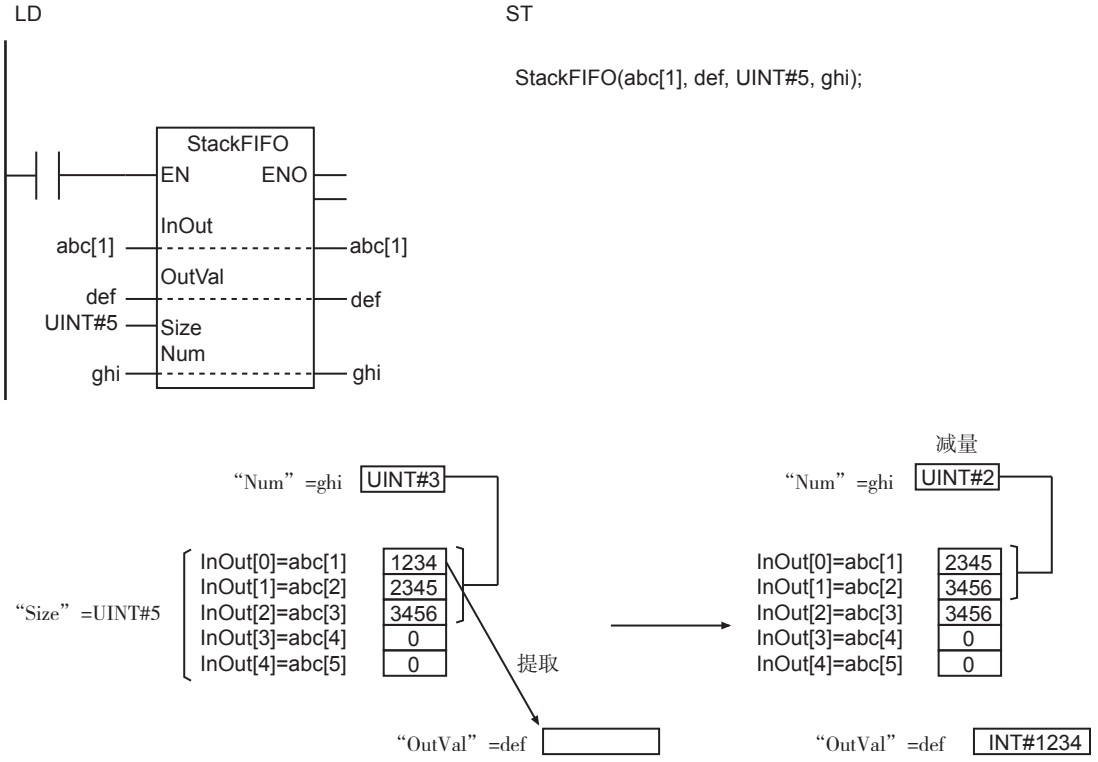
功能

视为当前已将保存元素数“Num”个元素保存至堆栈数组InOut[]。从该处提取值，代入输出值“OutVal”。
在堆栈的元素数 “Size”中指定InOut[]中用作堆栈的元素数。

- StackFIFO

提取堆栈最低位的值。将InOut[0]的值代入 “OutVal”。
然后，将InOut[1]之后的 “Num”–1个元素向堆栈数组的低位方向各移1位。
最后，对 “Num”进行减量。

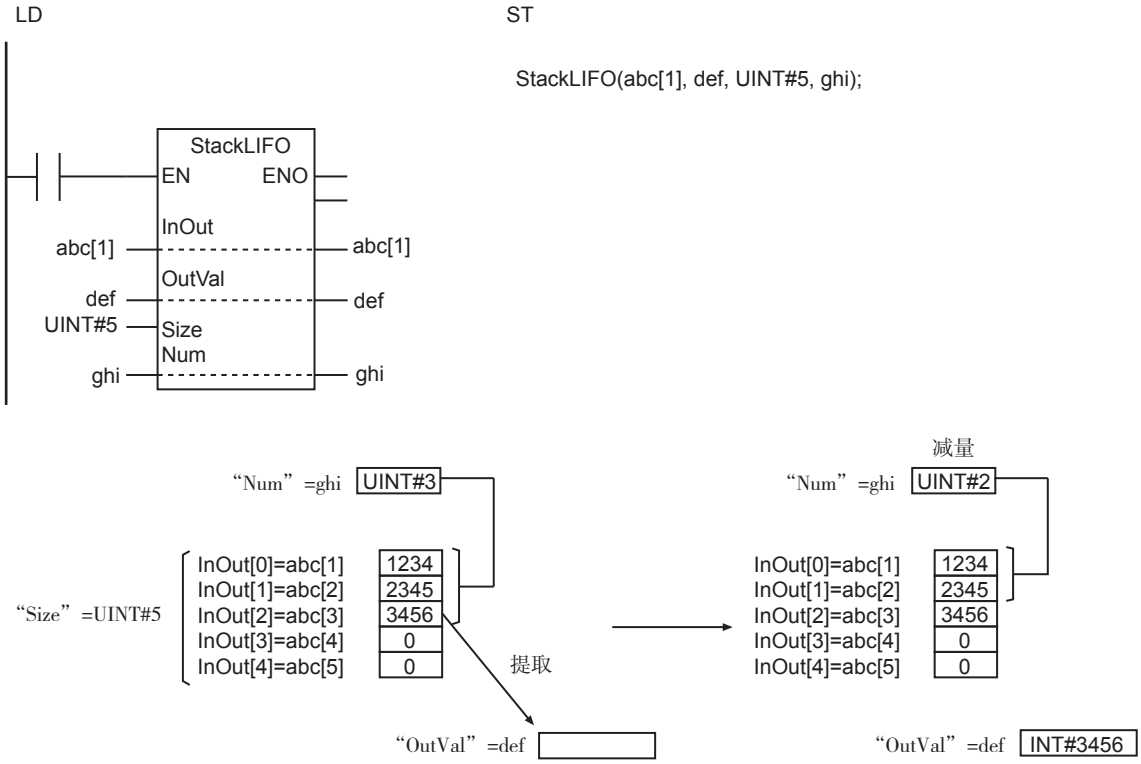
“Size”=UINT#5、“Num”=UINT#3时的示例如下所示。



● StackLIFO

提取堆栈最高位的值。将InOut[“Num”-1]的值代入 “OutVal”。
对 “Num”进行减量。

“Size”=UINT#5、“Num”=UINT#2时的示例如下所示。



使用注意事项

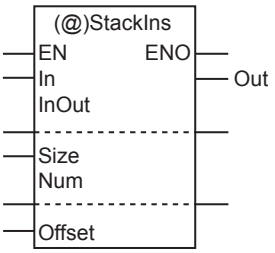
- 请将InOut[]和 “OutVal”的数据类型设为相同。如果不同，则编连时会发生异常。
- 将数组中的元素传输至InOut[]时，该元素之后为处理对象。
- “Size”或 “Num”为0时，InOut[]、“Num”、“OutVal”的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“OutVal”不变。
  - “Num”和 “Size”的值不为0，“Num” > “Size”时。
  - “Size”的值超过InOut[]的数组区域时。
  - InOut[]为STRING型数组，所有元素均未以NULL字符结尾时。
  - InOut[]为STRING型数组，元素的字节数超过 “OutVal”的大小时。

示例程序

🕮 请参阅 “StackPush指令(P.2-484)”的示例程序。

StackIns

将值插入堆栈的任意位置。

指令	名称	FB/FUN	图形表现	ST表现
StackIns	堆栈数据插入	FUN		StackIns(In, InOut, Size, Num, Offset);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	插入值	输入	插入堆栈的值、整个结构体、结构体的1个结构要素	遵从数据类型	–	(*)
Size	堆栈的元素数		堆栈的数组元素数			1
Offset	偏置位置		插入 “In” 的堆栈的偏置位置			0
InOut[]数组	堆栈数组	输入输出	构成堆栈的数组	遵从数据类型	–	–
Num	堆栈的保存元素数		保存在堆栈中的元素数			
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

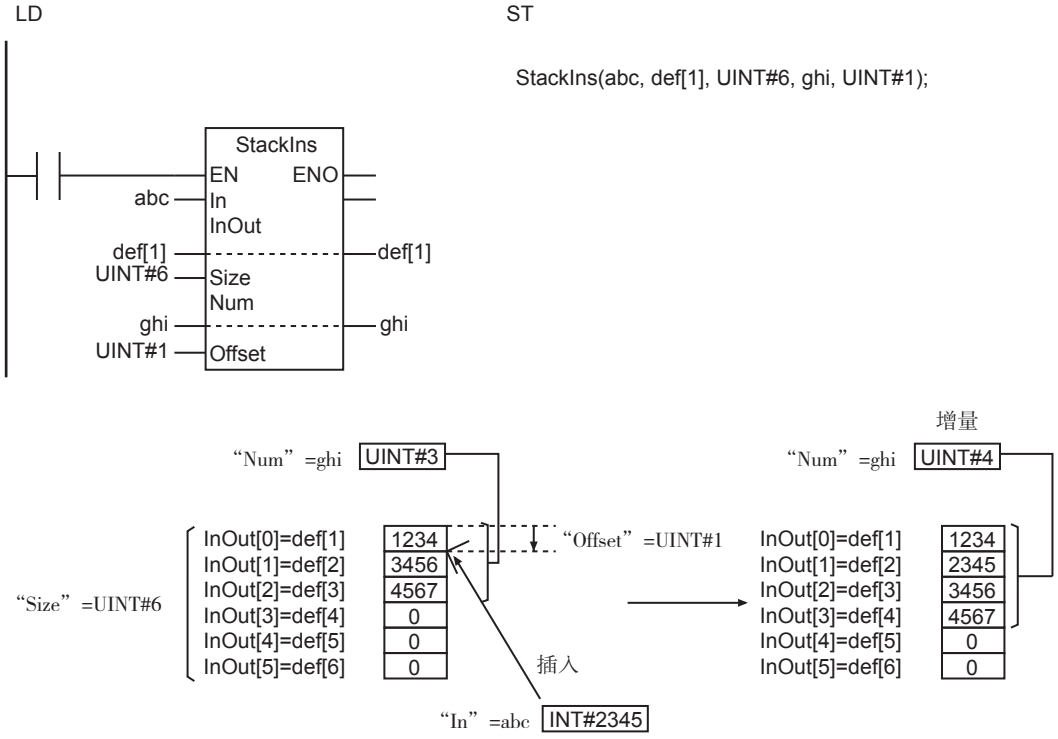
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个结构体、结构体的1个结构要素																			
Size							○													
Offset							○													
InOut[]数组	将与 “In” 相同的数据类型作为元素的数组																			
Num							○													
Out	○																			

功能

视为当前已将保存元素数“Num”个元素保存至堆栈数组InOut[]。将插入值“In”插入由偏置位置“Offset”确定的位置InOut[“Offset”]。
将更高位的元素即InOut[“Offset”] ~ InOut[“Num” −1]向堆栈数组的高位方向各移1位。
然后，对 “Num”进行增量。
在堆栈的元素数 “Size”中指定InOut[]中用作堆栈的元素数。

“Size” =UINT#6、“Num” =UINT#3、“Offset” =UINT#1时的示例如下所示。



使用注意事项

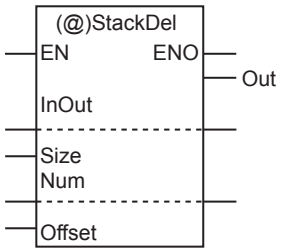
- 请将 “In”和InOut[]的数据类型设为相同。如果不同，则编连时会发生异常。
- 将数组中的元素传输至InOut[]时，该元素之后为处理对象。
- “Size”的值为0时，InOut[]、“Num”的值不变。
- 请务必将传输至 “In”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “In”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，InOut[]不变。
  - “Size”的值不为0，不满足 “Size” > “Num” ≥ “Offset”时。
  - “Size”的值超过InOut[]的数组区域时。
  - “In”、InOut[]为STRING型，“In”的字节数超过InOut[]的大小时。

示例程序

🕮 请参阅 “StackPush指令(P.2−484)”的示例程序。

StackDel

删除堆栈任意位置的值。

指令	名称	FB/FUN	图形表现	ST表现
StackDel	堆栈数据删除	FUN		StackDel(InOut, Size, Num, Offset);

变量

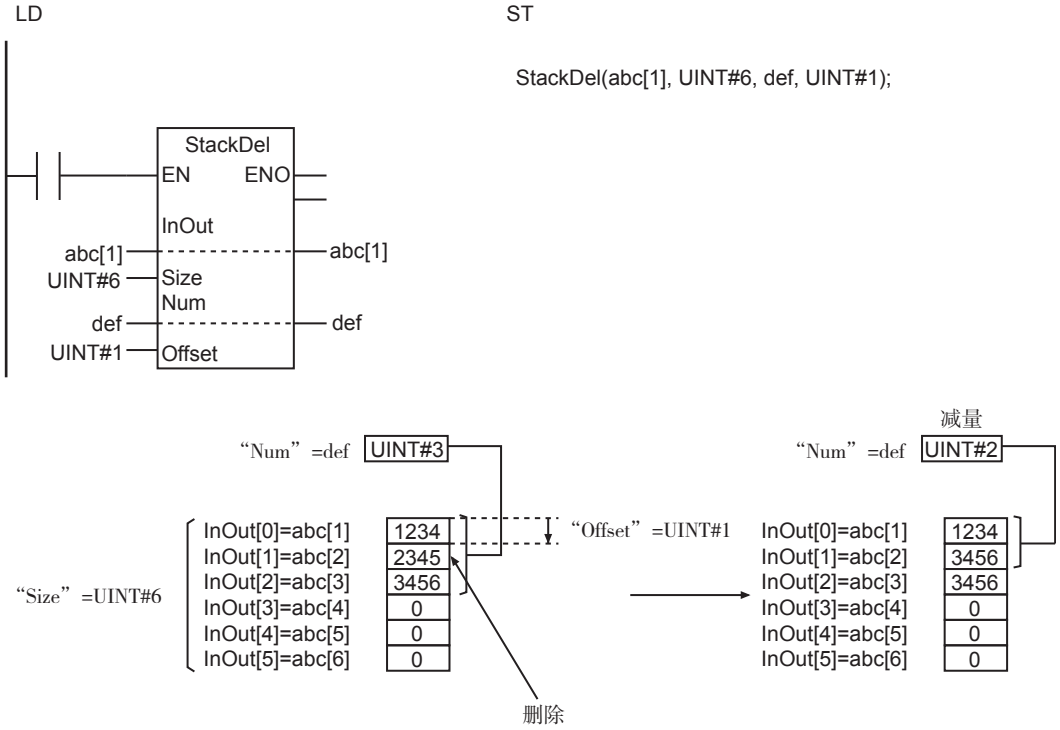
	名称	输入/输出	内容	有效范围	单位	初始值
Size	堆栈的元素数	输入	堆栈的数组元素数	遵从数据类型	–	1
Offset	偏置位置		待删除的堆栈的偏置位置			0
InOut[]数组	堆栈数组	输入输出	构成堆栈的数组	遵从数据类型	–	–
Num	堆栈的保存元素数		保存在堆栈中的元素数			
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Size							○													
Offset							○													
InOut[]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
Num							○													
Out	○																			

功能

视为当前已将保存元素数 “Num”个元素保存至堆栈数组InOut[]。删除由偏置位置 “Offset”确定的位置InOut[“Offset”]的值。
将更高位的元素即InOut[“Offset”+1]～InOut[“Num”−1]向堆栈数组的低位方向各移1位。
然后，对 “Num”进行减量。
在堆栈的元素数 “Size”中指定InOut[]中用作堆栈的元素数。

“Size”=UINT#6、“Num”=UINT#3、“Offset”=UINT#1时的示例如下所示。

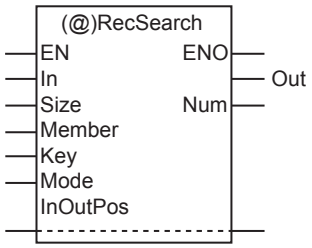


使用注意事项

- 将数组中的元素传输至InOut[]时，该元素之后为处理对象。
- “Size”或 “Num”的值为0时，InOut[]、“Num”的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，InOut[]不变。
  - “Num”和 “Size”的值不为0，不满足 “Size”≥ “Num”> “Offset”时。
  - “Size”的值超过InOut[]的数组区域时。

RecSearch

在以结构体为元素的数组中，按指定方法检索与检索关键词一致的要素。

指令	名称	FB/FUN	图形表现	ST表现
RecSearch	记录检索	FUN		Out:=RecSearch(In, Size, Member, Key, Mode, InOutPos, Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[] 数组	检索对象数组	输入	作为检索对象的以结构体为元素的数组	–	–	(*)
Size	检索对象的元素数		检索对象的数组元素数	遵从数据类型		1
Member	检索对象结构要素		In[]的结构体的检索对象结构要素			(*)
Key	检索关键词		检索值			
Mode	检索方法		检索方法	_LINEAR, _BIN_ASC, _BIN_DESC		_LINEAR
InOutPos[] 数组	一致元素的元素编号	输入输出	一致元素的元素编号	遵从数据类型	–	–
Out	检索结果	输出	TRUE：有一致的元素 FALSE：无一致的元素	遵从数据类型	–	–
Num	一致元素的个数		一致元素的个数			

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[] 数组	指定以结构体为元素的数组																			
Size							○													
Member						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
																(*)				
	与In[]的检索对象结构要素相同的数据类型																			
Key	与 “Member” 相同的数据类型																			
Mode	枚举体_eSEARCH_MODE 枚举元素参阅功能说明																			
InOutPos[] 数组							○													
Out	○																			
Num							○													

* TIME型、DATE型、TOD型、DT型、STRING型在CPU单元Ver.1.01以上且Sysmac Studio Ver.1.02以上时可指定。

功能

在以结构体为元素的数组In[]的 “Size” 个元素即In[0] ~ In[“Size” -1]中，检索结构体的检索对象结构要素 “Member” 的值与检索关键字 “Key” 一致的内容。
将In[]的任一元素的检索对象结构要素作为自变量传输至 “Member”。
如果存在一致的元素，则检索结果 “Out” 的值变为TRUE。将一致元素的元素编号、一致元素数分别代入 InOutPos[0]、“Num” 中。有 2 个以上一致元素时，将 In[] 中最低位的一致元素的元素编号代入InOutPos[0]。
如果不存在一致元素，则 “Out” 的值变为FALSE，InOutPos[0]和 “Num” 变为0。

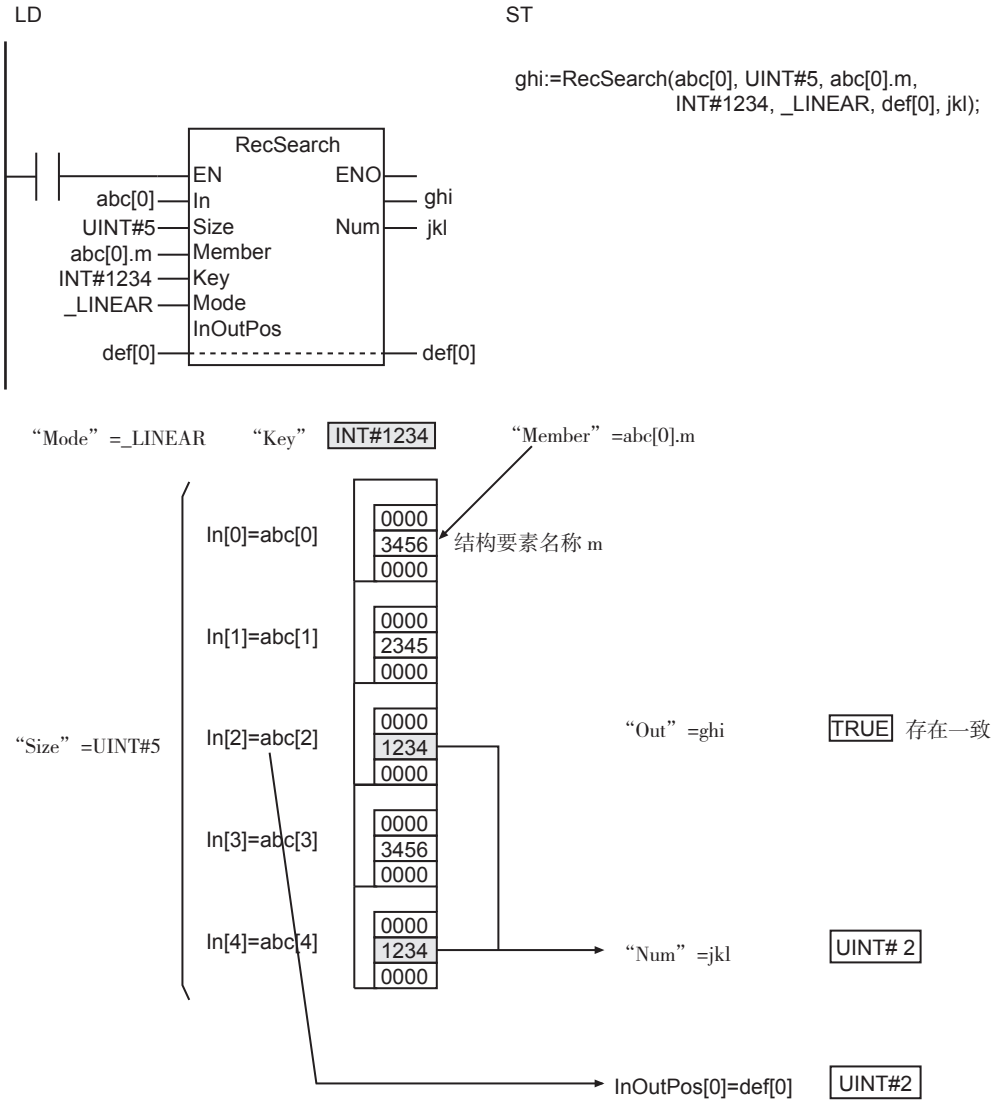
传输至In[]的输入参数务必也像array[3]那样加上元素编号后再指定。

检索方法 “Mode” 的数据类型为枚举体_eSEARCH_MODE。枚举元素的含义如下所示。

枚举元素	含义
_LINEAR	线性检索
_BIN_ASC	升序二分检索
_BIN_DESC	降序二分检索

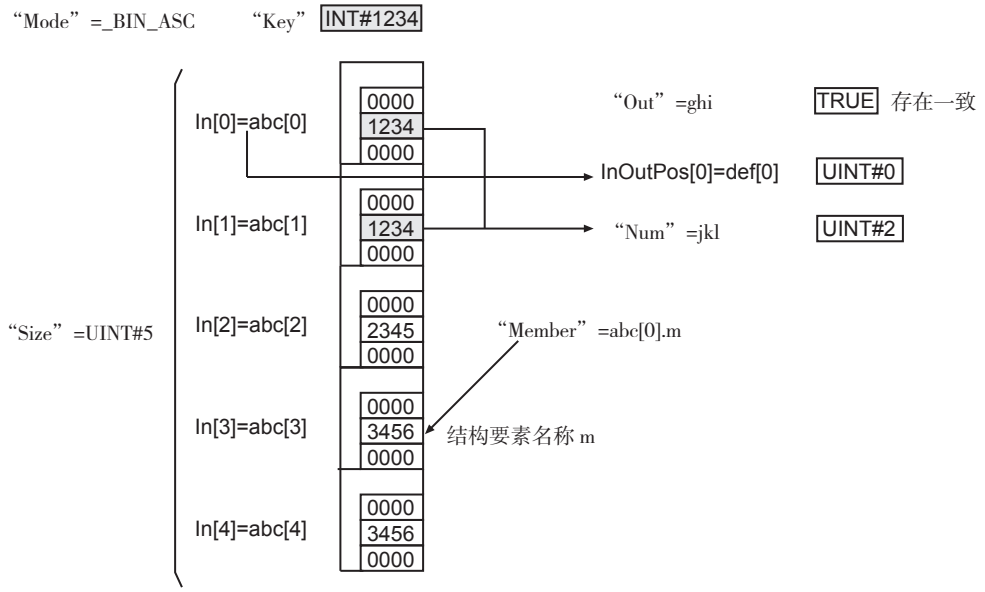
线性检索时，从In[]的首个元素起依次进行检索。

“Size”=UINT#5、“Key”=INT#1234、“Mode”=_LINEAR时的示例如下所示。



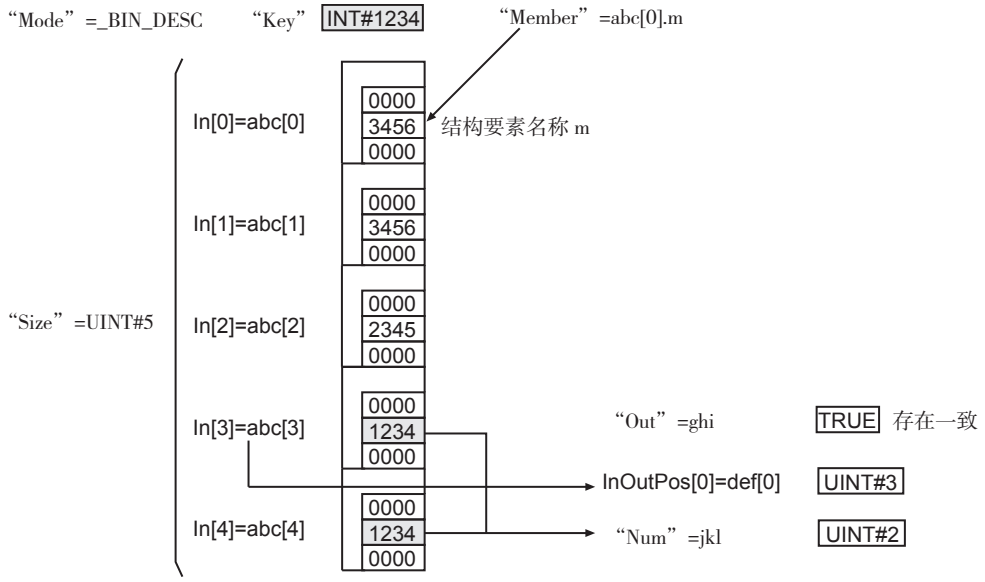
升序二分检索时，执行本指令前需事先将传输至In[]的输入参数的数组元素进行升序排列。再通过执行本指令进行二分检索。

对与上述相同的示例进行升序二分检索后，数组元素的顺序和处理结果如下所示。



降序二分检索时，执行本指令前需事先将传输至In[]的输入参数的数组元素进行降序排列。再通过执行本指令进行二分检索。

对与上述相同的示例进行降序二分检索后，数组元素的顺序和处理结果如下所示。



参考

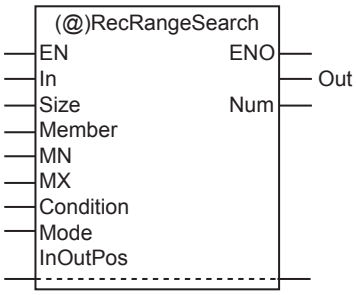
- 整个In[]可为更高位的结构体的结构要素。
  (例) In[0]=str0.str1[0]
- In[]也可为2维以上的数组。此时，将与检索条件一致的元素的第1维元素编号代入InOutPos[0]，将第2维元素编号代入InOutPos[1]。
- In[]为3维数组时，将与检索条件一致的元素的第1维元素编号代入InOutPos[0]，将第2维元素编号代入InOutPos[1]，将第3维元素编号代入InOutPos[2]。
- 检索TIME型、DT型、TOD型时，请符合"Member"和"Key"的精度。有 "TruncTime指令(P.2-642)"、"TruncDt指令(P.2-646)"、"TruncTod指令(P.2-650)"，以符合值的精度。

使用注意事项

- In[]请设为以结构体为元素的数组。否则，编连时会发生异常。
- 请将 “key”和 “Member”的数据类型设为相同。如果不同，则编连时会发生异常。
- 将数组中的元素传输至In[]时，该元素之后为处理对象。
- “Member”为实数时，会因数值而产生误差，因此结果可能会出现意外。
- “Key”为实数时，“Key”请勿指定为非数。
- “Size”的值为0时，“Out”的值为FALSE，“Num”的值为0。InOutPos[]不变。
- “Mode”的值为_BIN_ASC(或_BIN_DESC)时，如果In[]的元素未按升序(或降序)排列，则无法获得正确结果。执行本指令前，请将元素升序(或降序)排列。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、InOutPos[]、“Num”不变。
  - “Mode”的值超过有效范围时。
  - “Size”的值超过In[]的数组区域时。
  - “Member”不是In[]的结构要素时。
  - InOutPos[]的数组大小小于In[]的维数时。
  - “Member”为STRING型，且未以NULL字符结尾时。

RecRangeSearch

在以结构体为元素的数组中，按指定方法检索与检索条件的范围一致的要素。

指令	名称	FB/FUN	图形表现	ST表现
RecRangeSearch	范围指定记录检索	FUN		Out:=RecRangeSearch(In, Size, Member, MN, MX, Condition, Mode, InOutPos, Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[] 数组	检索对象数组	输入	作为检索对象的以结构体为元素的数组	–	–	(*)
Size	检索对象的元素数		检索对象的数组元素数	遵从数据类型		1
Member	检索对象结构要素		In[]的结构体的检索对象结构要素			(*)
MN	检索条件下限值		检索条件下限值			
MX	检索条件上限值		检索条件上限值			
Condition	检索条件		检索条件	_EQ_BOTH, _EQ_MIN, _EQ_MAX, _NE_BOTH		_EQ_BOTH
Mode	检索方法		检索方法	_LINEAR, _BIN_ASC, _BIN_DESC		_LINEAR
InOutPos[] 数组	一致元素的元素编号	输入输出	一致元素的元素编号	遵从数据类型	–	–
Out	检索结果	输出	TRUE：有一致的元素 FALSE：无一致的元素	遵从数据类型	–	–
Num	一致元素的个数		一致元素的个数			

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[] 数组	指定以结构体为元素的数组																			
Size							○													
Member						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
																(*)				
	与In[]的检索对象结构要素相同的数据类型																			
MN	与 “Member” 相同的数据类型																			
MX	与 “Member” 相同的数据类型																			
Condition	枚举体_eSEARCH_CONDITION 枚举元素参阅功能说明																			
Mode	枚举体_eSEARCH_MODE 枚举元素参阅功能说明																			
InOutPos[] 数组							○													
Out	○																			
Num							○													

* TIME型、DATE型、TOD型、DT型、STRING型在CPU单元Ver.1.01以上且Sysmac Studio Ver.1.02以上时可指定。

功能

在以结构体为元素的数组In[]的 “Size”个元素即In[0]～In[“Size”-1]中，检索结构体的检索对象结构要素 “Member”的值与检索条件一致的内容。
检索条件和检索方法由 “Condition”和 “Mode”指定。详情将于下文阐述。
将In[]的任一元素的检索对象结构要素作为自变量传输至 “Member”。
如果存在与条件一致的元素，则检索结果 “Out”的值变为TRUE。将一致元素的元素编号、一致元素数分别代入InOutPos[0]、“Num”中。有2个以上一致元素时，将In[]中最低位的一致元素的元素编号代入InOutPos[0]。
如果不存在一致元素，则 “Out”的值变为FALSE，InOutPos[0]和 “Num”变为0。

传输至In[]的输入参数务必也像array[3]那样加上元素编号后再指定。

检索条件 “Condition”的数据类型为枚举体_eSEARCH_CONDITION。枚举元素的含义如下所示。

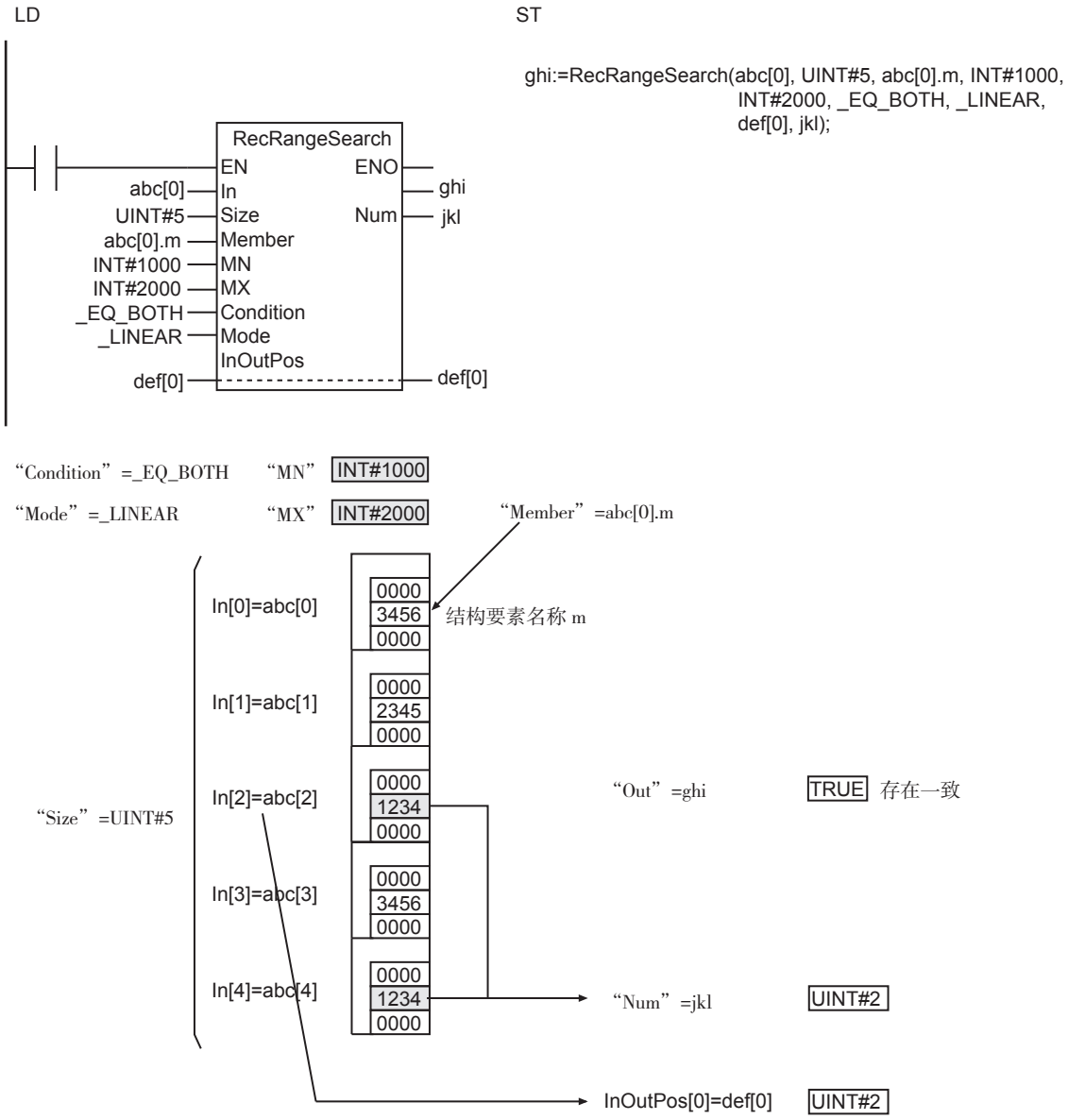
枚举元素	含义
_EQ_BOTH	“MN” ≦ “Member” ≦ “MX”
_EQ_MIN	“MN” ≦ “Member” < “MX”
_EQ_MAX	“MN” < “Member” ≦ “MX”
_NE_BOTH	“MN” < “Member” < “MX”

检索方法 “Mode”的数据类型为枚举体_eSEARCH_MODE。枚举元素的含义如下所示。

枚举元素	含义
_LINEAR	线性检索
_BIN_ASC	升序二分检索
_BIN_DESC	降序二分检索

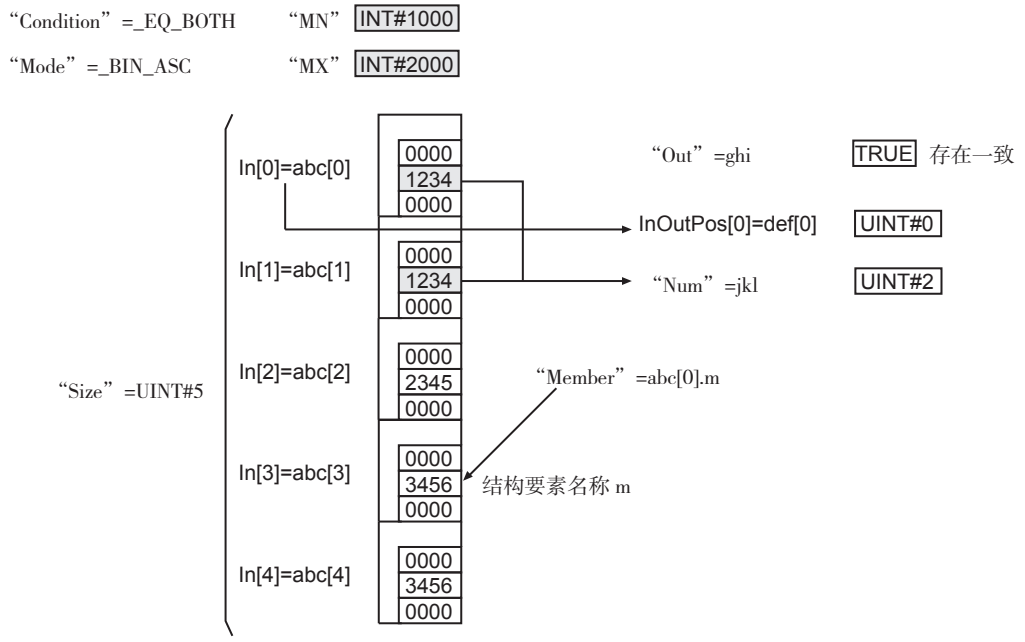
线性检索时，从In[]的首个元素起依次进行检索。

“Size”=UINT#5、“MN”=INT#1000、“MX”=INT#2000、“Condition”=_EQ_BOTH、“Mode”=_LINEAR时的示例如下所示。



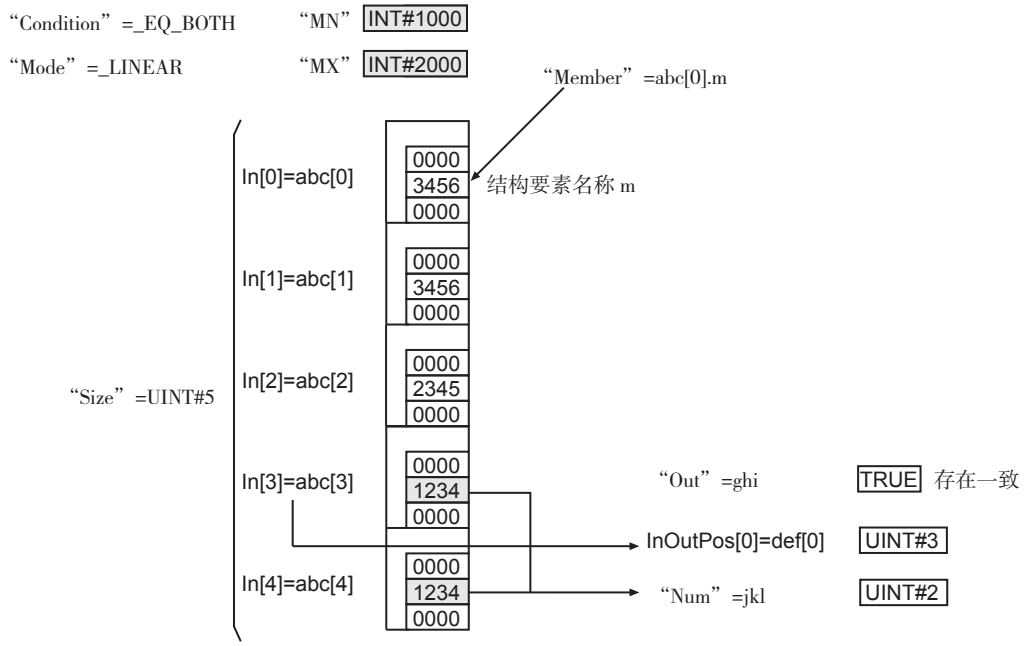
升序二分检索时，执行本指令前需事先将传输至In[]的输入参数的数组元素进行升序排列。再通过执行本指令进行二分检索。

对与上述相同的示例进行升序二分检索后，数组元素的顺序和处理结果如下所示。



降序二分检索时，执行本指令前需事先将传输至In[]的输入参数的数组元素进行降序排列。再通过执行本指令进行二分检索。

对与上述相同的示例进行降序二分检索后，数组元素的顺序和处理结果如下所示。



参考

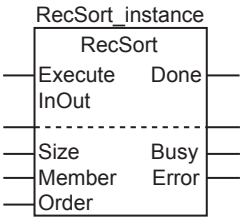
- 整个In[]可为更高位的结构体的结构要素。
  (例) In[0]=str0.str1[0]
- In[]也可为2维以上的数组。此时，将与检索条件一致的元素的第1维元素编号代入InOutPos[0]，将第2维元素编号代入InOutPos[1]。
- In[]为3 维数组时，将与检索条件一致的元素的第1维元素编号代入InOutPos[0]，将第2维元素编号代入InOutPos[1]，将第3维元素编号代入InOutPos[2]。
- 检索TIME型、DT型、TOD型时，请符合 “Member”、“MN”、“MX”的精度。有 “TruncTime指令(P.2–642)”、“TruncDt指令(P.2–646)”、“TruncTod指令(P.2–650)”，以符合值的精度。

使用注意事项

- 请将 “Member”、“MN”、“MX”的数据类型与In[]的检索对象结构要素的数据类型设为一致。否则，编连时会发生异常。
- In[]请设为以结构体为元素的数组。否则，编连时会发生异常。
- 将数组中的元素传输至In[]时，该元素之后为处理对象。
- “Member”为实数时，会因数值而产生误差，因此结果可能会出现意外。
- “MN”、“MX”为实数时，请勿指定非数。
- “Size”的值为0时，“Out”的值为FALSE，“Num”的值为0。InOutPos[]不变。
- “Mode”的值为_BIN_ASC(或_BIN_DESC)时，如果In[]的元素未按升序(或降序)排列，则无法获得正确结果。执行本指令前，请将元素升序(或降序)排列。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、InOutPos[]、“Num”不变。
  - “MN” > “MX”时。
  - “Condition”的值超过有效范围时。
  - “Mode”的值超过有效范围时。
  - “Size”的值超过In[]的数组区域时。
  - “Member”不是In[]的结构要素时。
  - InOutPos[]的数组大小小于In[]的维数时。

RecSort

将以结构体为元素的数组的要素进行分类。

指令	名称	FB/FUN	图形表现	ST表现
RecSort	记录排序	FB		RecSort_instance(Execute, InOut, Size, Member, Order, Done, Busy, Error);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Size	排序对象的元素数	输入	排序对象的数组元素数	遵从数据类型	–	1
Member	排序对象结构要素		In[]的结构体的排序对象结构要素			(*)
Order	排序顺序		排序顺序	_ASC, _DESC		_ASC
InOut[]数组	排序对象数组	输入输出	作为排序对象的以结构体为元素的数组	–	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Size							○													
Member						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	与InOut[]的检索对象结构要素相同的数据类型															(*)				
Order	枚举体_eSORT_ORDER 枚举元素参阅功能说明																			
InOut[]数组	指定以结构体为元素的数组																			

* TIME型、DATE型、TOD型、DT型、STRING型在CPU单元Ver.1.01以上且Sysmac Studio Ver.1.02以上时可指定。

功能

“Execute”的值为TRUE时，按照结构体的排序对象结构要素 “Member”的值，将以结构体为元素的数组InOut[]的 “Size”个元素即InOut[0] ~ InOut[“Size”−1]进行排序。排序顺序由 “Order”指定。详情将于下文阐述。
将In[]的任一元素的排序对象结构要素作为自变量传输至 “Member”。

传输至InOut[]的输入输出参数务必也像array[3]那样加上元素编号后再指定。

排序顺序 “Order”的数据类型为枚举体_eSORT_ORDER。枚举元素的含义如下所示。

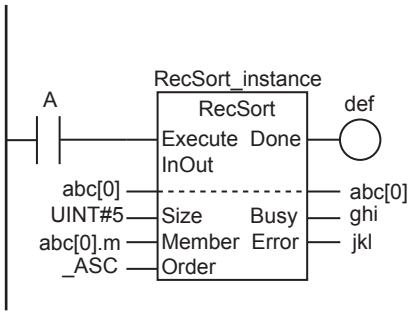
枚举元素	含义
_ASC	升序
_DESC	降序

整数、实数以外的数据类型的值的大小关系的判断如下表所示。

数据类型	大小关系
TIME	值较大者判断为大。
DATE、TOD、DT	对于日期和时刻，较后者判断为大。
STRING	📖 与 “LTascii/LEascii/GTascii/GEascii指令(P.2−105)”的规格相同。请参阅该处。

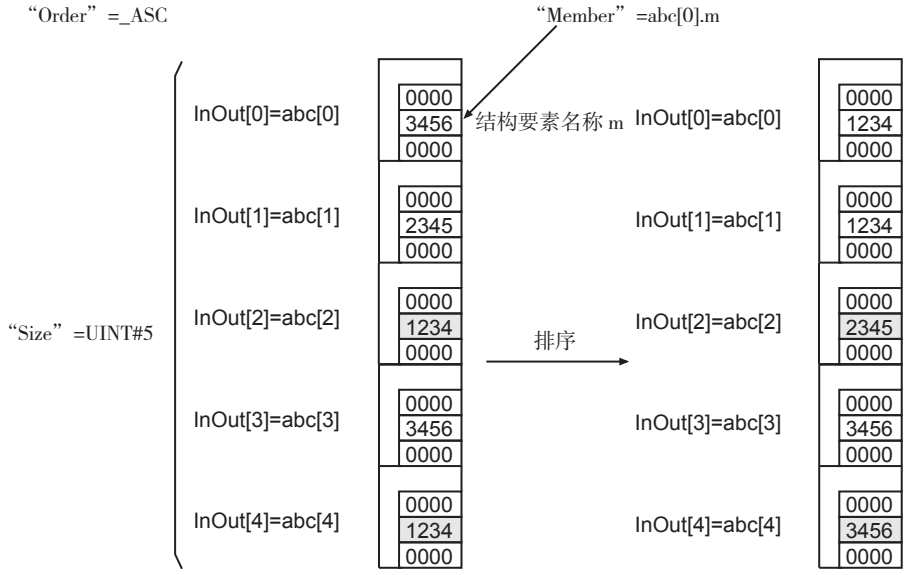
“Size”=UINT#5、“Order”=_ASC时的示例如下所示。

LD



ST

```
RecSort_instance(A, abc[0], UINT#5, abc[0].m, _ASC, def, ghi, jkl);
```



参考

- 执行本指令时如果电源断开，则可能会破坏InOut[]的内容。如果每次执行完本指令后均备份InOut[]的内容，则该内容即使受到破坏也可修复。请参阅示例程序。
- 对TIME型、DT型、TOD型进行排序时，请符合“Member”的精度。有 “TruncTime指令(P.2-642)”、“TruncDt指令(P.2-646)”、“TruncTod指令(P.2-650)”，以符合值的精度。

使用注意事项

- InOut[]请设为以结构体为元素的数组。否则，编连时会发生异常。
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- “Member”为实数时，会因数值而产生误差，因此结果可能会出现意外。
- 将数组中的元素传输至InOut[]时，该元素之后为处理对象。
- “Size”的值为0时，“Done”的值为TRUE，InOut[]不变。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Order”的值超过有效范围时。
  - “Size”的值超过InOut[]的数组区域时。
  - “Member”不是InOut[]的结构要素时。
  - “Member”为STRING型，且未以NULL字符结尾时。

示例程序

将以结构体MyStr为元素的数组Abc[]进行升序排序。排序对象结构要素为Abc[].m。
执行排序前需先按照Abc_backup[]的变量名称备份Abc[]，以确保处理时即使发生断电也不会丢失数据。如果发生断电，则将Abc_backup[]恢复为Abc[]，并重新排序。

全局变量的定义

数据类型

名称	数据类型	注释
MyStr	STRUCT	结构体
l	BOOL	结构要素
m	INT	结构要素
n	REAL	结构要素

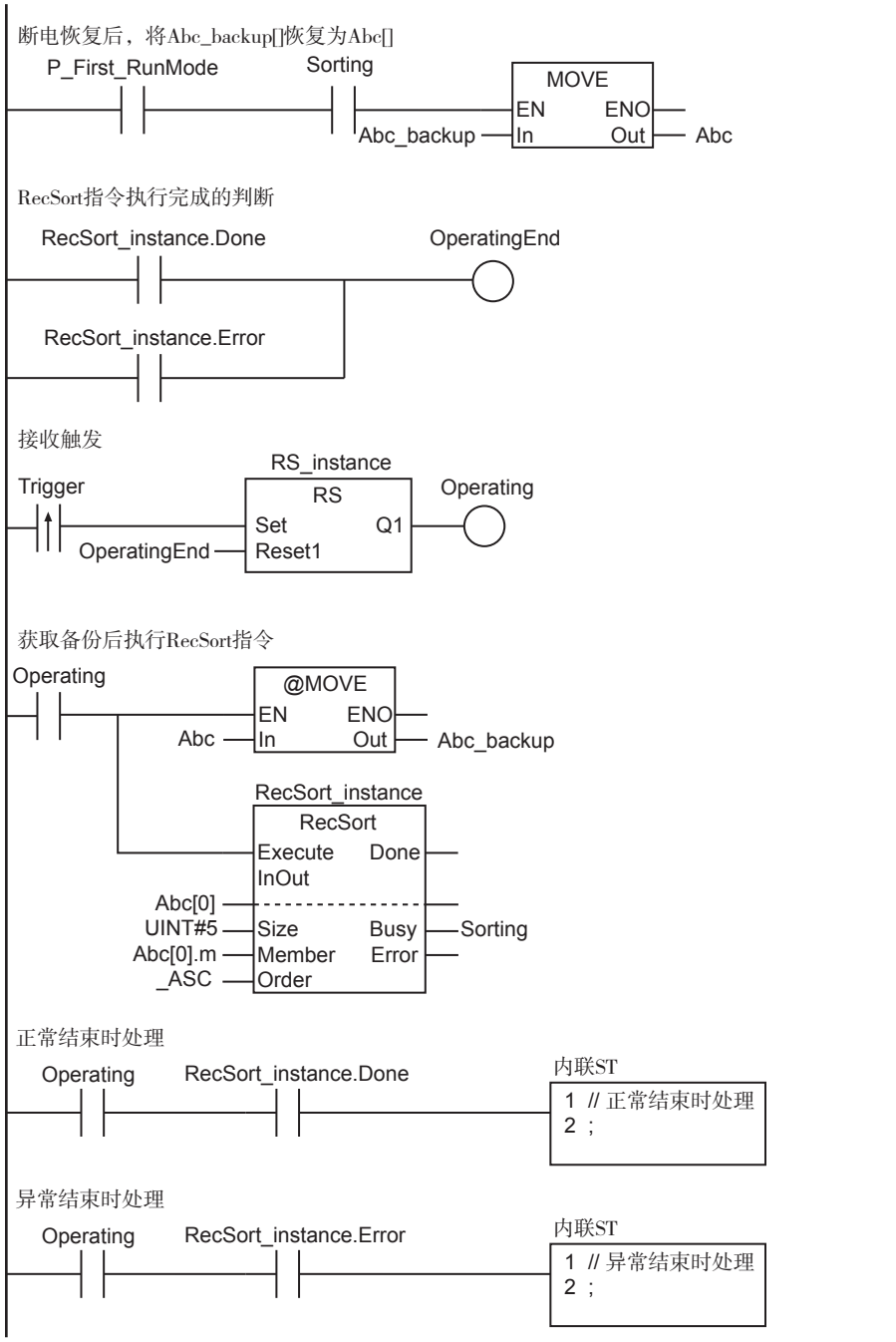
全局变量

名称	数据类型	初始值	保持	注释
Abc	ARRAY[0..4] OF MyStr	[5((l:=FALSE,m:=0,n:=0.0))]	☑	排序对象数组
Abc_backup	ARRAY[0..4] OF MyStr	[5((l:=FALSE,m:=0,n:=0.0))]	☑	Abc[]的备份

LD

内部变量	名称	数据类型	初始值	保持	注释
	Sorting	BOOL	FALSE	☑	处理中(带保持)
	OperatingEnd	BOOL	FALSE	☐	处理结束
	Trigger	BOOL	FALSE	☐	执行条件
	Operating	BOOL	FALSE	☐	处理中
	RS_instance	RS		☐	
	RecSort_instance	RecSort		☐	

外部变量	名称	数据类型	注释
	Abc	ARRAY[0..4] OF MyStr	排序对象数组
	Abc_backup	ARRAY[0..4] OF MyStr	Abc[]的备份



ST

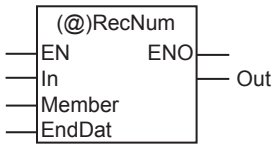
内部变量	名称	数据类型	初始值	保持	注释
	Sorting	BOOL	FALSE	☑	处理中(带保持)
	Trigger	BOOL	FALSE	☐	执行条件
	LastTrigger	BOOL	FALSE	☐	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	☐	处理开始
	Operating	BOOL	FALSE	☐	处理中
	RS_instance	RS		☐	
	RecSort_instance	RecSort		☐	

外部变量	名称	数据类型	注释
	Abc	ARRAY[0..4] OF MyStr	排序对象数组
	Abc_backup	ARRAY[0..4] OF MyStr	Abc[]的备份

```
// 断电恢复后，将Abc_backup[]恢复为Abc[]
IF ( (P_First_RunMode = TRUE) AND (Sorting = TRUE) ) THEN
   Abc:=Abc_backup;
END_IF;

// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
   OperatingStart:=TRUE;
   Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// RecSort指令初始化
IF (OperatingStart=TRUE) THEN
   Abc_backup:=Abc;
   RecSort_instance(
      Execute:=FALSE,     // 启动条件
      InOut   :=Abc[0],     // 排序对象数组
      Member :=Abc[0].m); // 排序对象结构要素
   OperatingStart:=FALSE;
END_IF;

// RecSort指令执行
IF (Operating=TRUE) THEN
   RecSort_instance(
      Execute:=TRUE,
      InOut   :=Abc[0],
      Size      :=UINT#5,
      Member :=Abc[0].m,
      Order    :=_ASC,
      Busy    =>Sorting);

   IF (RecSort_instance.Done=TRUE) THEN
      // 正常结束时处理
      Operating:=FALSE;
   END_IF;

   IF (RecSort_instance.Error=TRUE) THEN
      // 异常结束时处理
      Operating:=FALSE;
   END_IF;
END_IF;
```

RecNum

计算以结构体为元素的数组的结尾数据为止的记录数。

指令	名称	FB/FUN	图形表现	ST表现
RecNum	记录数获取	FUN		Out:=RecNum(In, Member, EndDat);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[] 数组	对象数组	输入	作为处理对象的以结构体为元素的数组	–	–	(*)
Member	对象结构要素		In[]的结构体的处理对象结构要素	遵从数据类型		
EndDat	结尾数据		结尾数据			
Out	记录数	输出	记录数	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[] 数组	指定以结构体为元素的数组																			
Member	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○ *1	○	○	○	○
	枚举体也可指定*2																			
	与In[]的处理对象结构要素相同的数据类型																			
EndDat	与 “Member”相同的数据类型																			
Out							○													

*1 TIME型、DATE型、TOD型、DT型在CPU单元Ver.1.01以上且Sysmac Studio Ver.1.02以上时可指定。

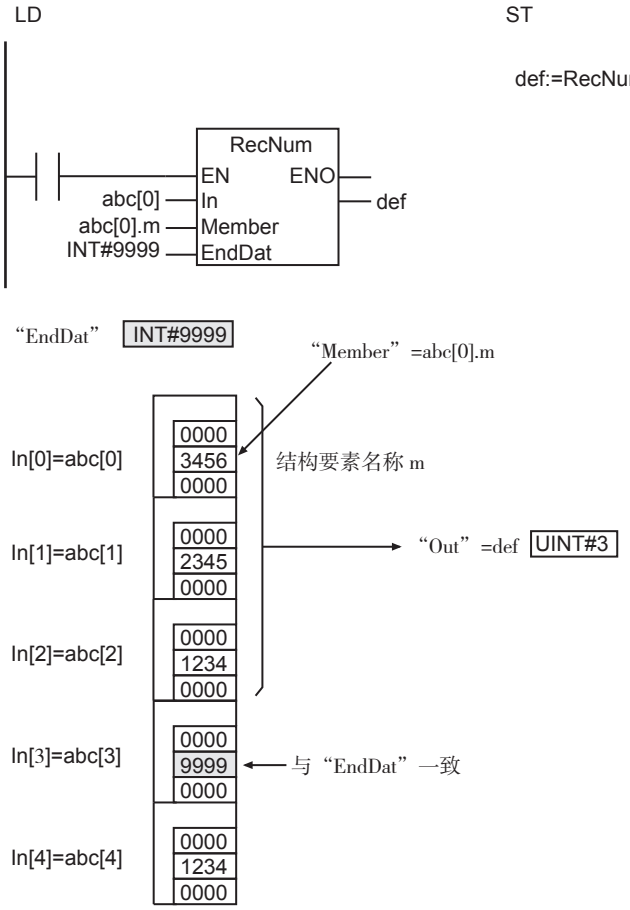
*2 CPU单元Ver.1.02以上且Sysmac Studio Ver.1.03以上时可指定。

功能

从以结构体为元素的数组In[]的开头起，检索对象结构要素 “Member”的值与结尾数据 “EndDat”一致的元素。最终，将与 “EndDat”一致的元素前的元素数(记录数)代入 “Out”。
将In[]的任一元素的对象结构要素作为自变量传输至 “Member”。

传输至In[]的输入参数务必也像array[3]那样加上元素编号后再指定。

“EndDat” =INT#9999时的示例如下所示。



参考

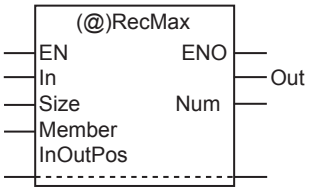
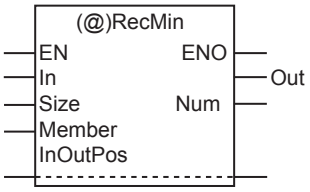
- 整个In[]可为更高位的结构体的结构要素。
  (例) In[0]=str0.str1[0]
- 检索 TIME 型、DT 型、TOD 型时，请符合 “Member” 和 “EndDat” 的精度。有 “TruncTime 指令(P.2-642)”、“TruncDt指令(P.2-646)”、“TruncTod指令(P.2-650)”，以符合值的精度。

使用注意事项

- In[]请设为以结构体为元素的数组。否则，编连时会发生异常。
- 请将 “Member” 和 “EndDat” 的数据类型设为相同。如果不同，则编连时会发生异常。
- In[]的结构要素中无与 “EndDat” 一致的结构要素时，将In[]的所有元素数代入 “Out”。
- “Member” 为实数时，会因数值而产生误差，因此结果可能会出现意外。
- “EndDat” 为实数时，“EndDat” 请勿指定为非数。
- 将数组中的元素传输至In[]时，该元素之后为处理对象。
- 以下情况时会发生异常。ENO变为FALSE，“Out” 不变。
  - “Member” 不是In[]的结构要素时。
  - “Member” 为STRING型，且未以NULL字符结尾时。

RecMax/RecMin

RecMax :在以结构体为元素的数组中，检索指定的结构要素的最大值。
RecMin :在以结构体为元素的数组中，检索指定的结构要素的最小值。

指令	名称	FB/FUN	图形表现	ST表现
RecMax	记录最大值检索	FUN		Out:=RecMax(In, Size,Member, InOutPos, Num);
RecMin	记录最小值检索	FUN		Out:=RecMin(In, Size, Member, InOutPos, Num);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[] 数组	检索对象数组	输入	作为检索对象的以结构体为元素的数组	–	–	(*)
Size	检索对象的元素数		检索对象的数组元素数	遵从数据类型		1
Member	检索对象结构要素		In[]的结构体的检索对象结构要素			(*)
InOutPos[] 数组	检索结果的元素编号	输入输出	检索结果的元素编号	遵从数据类型	–	–
Out	检索结果	输出	检索结果	遵从数据类型	–	–
Num	检索个数		检索个数			

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[] 数组	指定以结构体为元素的数组																			
Size							○													
Member						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	与In[]的检索对象结构要素相同的数据类型															(*)				
InOutPos[] 数组							○													
Out						○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
																(*)				

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Num							○													

* TIME型、DATE型、TOD型、DT型、STRING型在CPU单元Ver.1.01以上且Sysmac Studio Ver.1.02以上时可指定。

功能

在以结构体为元素的数组In[]的 “Size”个元素即In[0]～In[“Size”−1]中，检索结构体的检索对象结构要素 “Member”的值。
将In[]的任一元素的检索对象结构要素作为自变量传输至 “Member”。
将检索结果的元素编号、检索元素数分别代入InOutPos[0]、“Num”。有2个以上检索结果时，将In[]中最低位的检索结果的元素编号代入InOutPos[0]。

传输至In[]的输入参数务必也像array[3]那样加上元素编号后再指定。

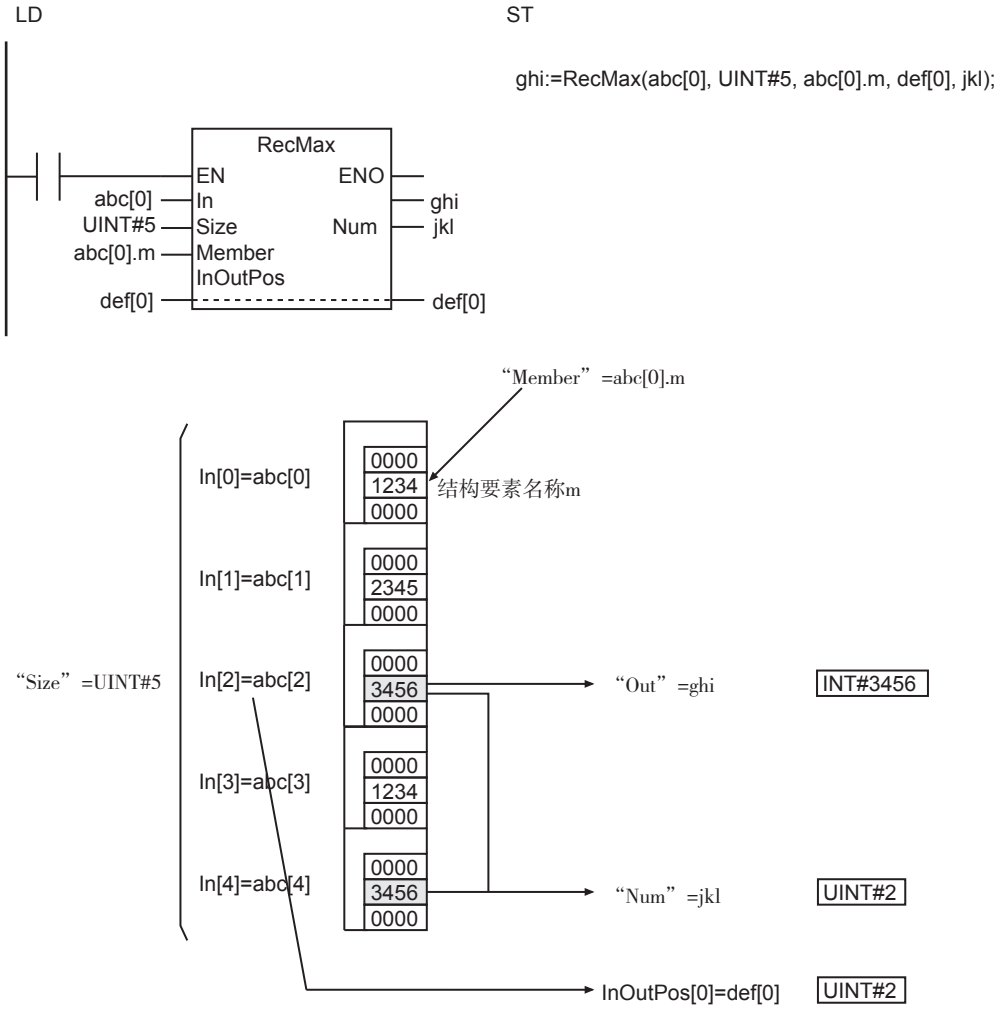
整数、实数以外的数据类型的值的大小关系的判断如下表所示。

数据类型	大小关系
TIME	值较大者判断为大。
DATE、TOD、DT	对于日期和时刻，较后者判断为大。
STRING	🕮 与 “LTascii/LEascii/GTascii/GEascii指令(P.2−105)”的规格相同。请参阅该处。

- **RecMax**
  检索最大值。检索结果 “Out”中代入检索对象结构要素的最大值。

- **RecMin**
  检索最小值。检索结果 “Out”中代入检索对象结构要素的最小值。

RecMax指令下，“Size”=UINT#5时的示例如下所示。



参考

- 整个In[]可为更高位的结构体的结构要素。
  (例)　In[0]=str0.str1[0]
- In[]也可为2维以上的数组。此时，将与检索条件一致的元素的第1维元素编号代入InOutPos[0]，将第2维元素编号代入InOutPos[1]。
- In[]为3维数组时，将与检索条件一致的元素的第1维元素编号代入InOutPos[0]，将第2维元素编号代入InOutPos[1]，将第3维元素编号代入InOutPos[2]。
- 检索TIME型、DT型、TOD型时，请符合 “Member”的精度。有 “TruncTime指令(P.2–642)”、“TruncDt指令(P.2–646)”、“TruncTod指令(P.2–650)”，以符合值的精度。

使用注意事项

- “Member”和“Out”的数据类型不同时，请从下列数据类型中选择，以使“Out”的有效范围包含“Member”的有效范围。
  - USINT,UINT,UDINT,ULINT,SINT,INT,DINT,LINT,REAL,LREAL
- “Member”为实数时，会因数值而产生误差，因此结果可能会出现意外。
- 将数组中的元素传输至In[]时，该元素之后为处理对象。
- “In”为枚举体时，请务必将传输至“In”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “Size”的值为0时，“Out”、“Num”的值变为0。但“Member”为STRING型、“Size”的值为0时，“Out”为仅含有NULL字符的字符串。InOutPos[]的值不变。
- 以下情况时会发生异常。ENO变为FALSE，“Out”、InOutPos[]、“Num”不变。
  - “Size”的值超过In[]的数组区域时。
  - “Member”不是In[]的结构要素时。
  - InOutPos[]的数组大小小于In[]的维数时。
  - “Member”为STRING型，且未以NULL字符结尾时。

FCS指令

指令	名称	页码
StringSum	和值计算	2–522
StringLRC	LRC值计算(字符串)	2–524
StringCRCCCITT	CRC–CCITT值计算(字符串)	2–526
StrCRC16	CRC–16值计算(字符串)	2–528
AryLRC_**	LRC值计算(数组)组	2–530
AryCRCCCITT	CRC–CCITT值计算(数组)	2–532
AryCRC16	CRC–16值计算(数组)	2–534

StringSum

计算字符串的和值。

指令	名称	FB/FUN	图形表现	ST表现
StringSum	和值计算	FUN	(@)StringSum EN ENO In Out Size	Out:=StringSum(In, Size);

变量

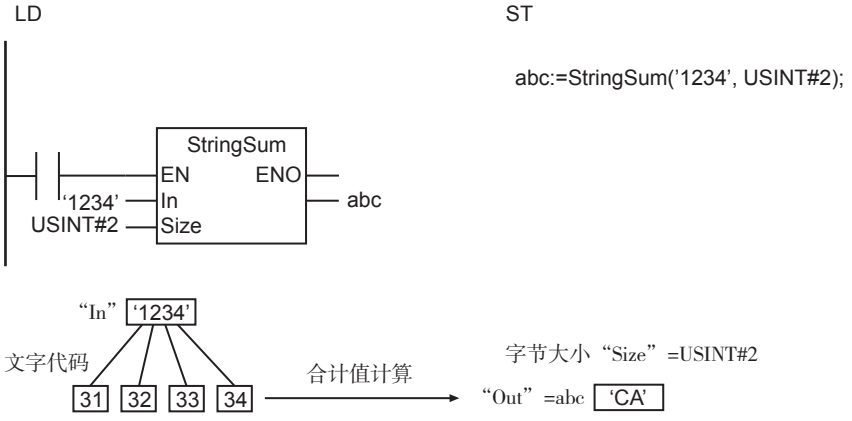
	名称	输入/输出	内容	有效范围	单位	初始值
In	对象字符串	输入	对象字符串	遵从数据类型	–	''
Size	字节大小		和值的字节大小	1，2	字节	1
Out	和值	输出	和值	“Size”显示的字节数	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Size						○														
Out																				○

功能

计算对象字符串“In”的和值(各字符的文字代码的总和)。和值“Out”为字节大小“Size”指定的字节数。“Out”以16进制的字符串表现，结尾保存NULL字符。

“In”='1234'，“Size”=USINT#2时的示例如下所示。



上例中，如果 “Size”=USINT#1，则 “Out”的值为'A'。

使用注意事项

- “In”的文字代码的总和超过 “Size”可表现的位数时，舍去高位。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过有效范围时。
  - “In”的字节数为0(仅NULL字符)时。

StringLRC

计算字符串的LRC值(水平奇偶校验)。

指令	名称	FB/FUN	图形表现	ST表现
StringLRC	LRC值计算(字符串)	FUN	(@)StringLRC EN ENO In Out	Out:=StringLRC(In);

变量

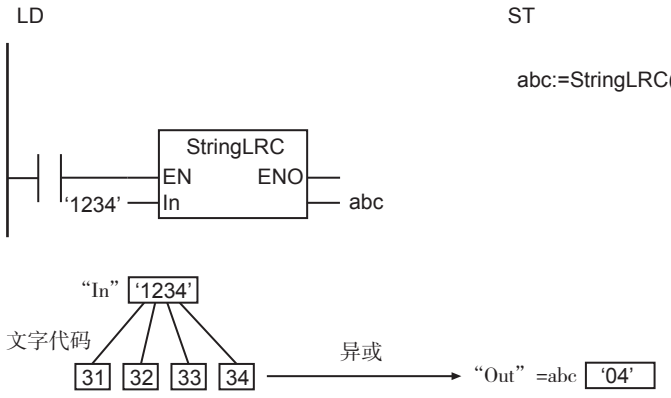
	名称	输入/输出	内容	有效范围	单位	初始值
In	对象字符串	输入	对象字符串	遵从数据类型	–	''
Out	LRC值	输出	LRC值	最大3字节(2个半角英数字字符+结尾NULL字符)	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out																				○

功能

计算对象字符串 “In”的LRC值(水平奇偶校验)。LRC值是指 “In”的各字符的文字代码的异或。LRC值 “Out”以16进制的字符串表现，结尾保存NULL字符。

“In”='1234'时的示例如下所示。



使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “In”的字节数为0(仅NULL字符)时。

StringCRCCCITT

按照XMODEM方式计算字符串的CRC-CCITT值。

指令	名称	FB/FUN	图形表现	ST表现
String CRCCCITT	CRC-CCITT值计算 (字符串)	FUN	(@)StringCRCCCITT EN ENO In Out Initial OutOrder	Out:=StringCRCCCITT(In, Initial, OutOrder);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	对象字符串	输入	对象字符串	遵从数据类型	-	"
Initial	初始值		CRC-CCITT值计算的初始值			0
OutOrder	字节顺序		处理 “In”的字节顺序	_LOW_HIGH, _HIGH_LOW		_HIGH_LOW
Out	CRC-CCITT值	输出	CRC-CCITT值	5字节(4个半角英数字字符+结尾NULL字符)	-	-

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Initial			○																	
OutOrder	枚举体_eBYTE_ORDER 枚举元素参阅功能说明																			
Out																				○

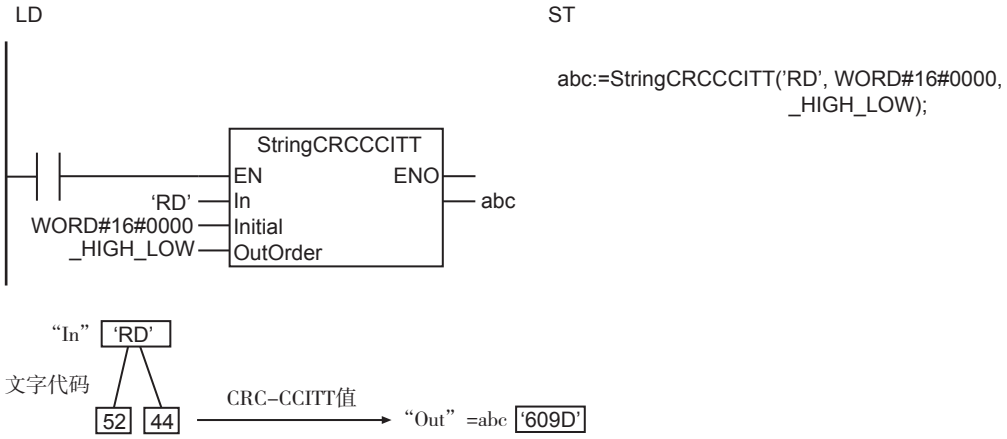
功能

按照XMODEM方式计算对象字符串 “In”的CRC-CCITT值。
CRC-CCITT值 “Out”以16进制的字符串表现，结尾保存NULL字符。

计算CRC-CCITT值时的初始值由 “Initial”指定。字节顺序由 “OutOrder”指定。
“OutOrder”的数据类型为枚举体_eBYTE_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_LOW_HIGH	低位字节在前，高位字节在后
_HIGH_LOW	高位字节在前，低位字节在后

“In”='RD'，“Initial”=WORD#16#0000，“OutOrder”=_HIGH_LOW时的示例如下所示。



使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “OutOrder”的值超过有效范围时。
- “In”的字节数为0(仅NULL字符)时。

StringCRC16

按照MODBUS方式计算字符串的CRC-16值。

指令	名称	FB/FUN	图形表现	ST表现
StringCRC16	CRC-16值计算(字符串)	FUN	(@)StringCRC16 EN ENO In Out Initial OutOrder	Out:=StringCRC16(In, Initial, OutOrder);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	对象字符串	输入	对象字符串	遵从数据类型	-	"
Initial	初始值		CRC-16值计算的初始值			16#FFFF
OutOrder	字节顺序		处理 “In”的字节顺序	_LOW_HIGH, _HIGH_LOW		_LOW _HIGH
Out	CRC-16值	输出	CRC-16值	5字节(4个半角英数字字符+结尾NULL字符)	-	-

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Initial			○																	
OutOrder	枚举体_eBYTE_ORDER 枚举元素参阅功能说明																			
Out																				○

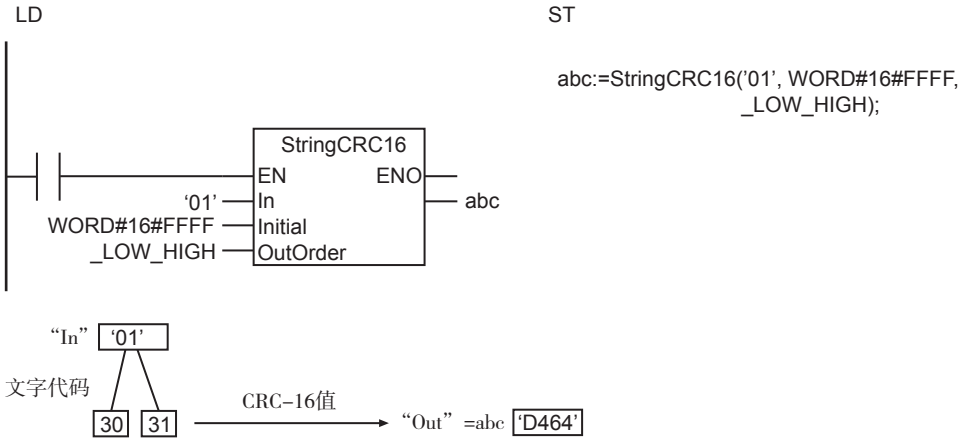
功能

按照MODBUS方式计算对象字符串 “In”的CRC-16值。
CRC-16值 “Out”以16进制的字符串表现，结尾保存NULL字符。

计算CRC-16值时的初始值由 “Initial”指定。字节顺序由 “OutOrder”指定。
“OutOrder”的数据类型为枚举体_eBYTE_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_LOW_HIGH	低位字节在前，高位字节在后
_HIGH_LOW	高位字节在前，低位字节在后

“In”='01'，“Initial”=WORD#16#FFFF，“OutOrder”=_LOW_HIGH时的示例如下所示。



使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “OutOrder”的值超过有效范围时。
- “In”的字节数为0(仅NULL字符)时。

AryLRC_**

计算数组的LRC值。

指令	名称	FB/FUN	图形表现	ST表现
AryLRC_**	LRC值计算(数组)组	FUN	(@)AryLRC_** EN ENO In Out Size **为位串的数据类型名称	Out:=AryLRC_**(In, Size); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	–	(*)
Size	运算对象的元素数		In[ ]的元素数			1
Out	LRC值	输出	LRC值	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

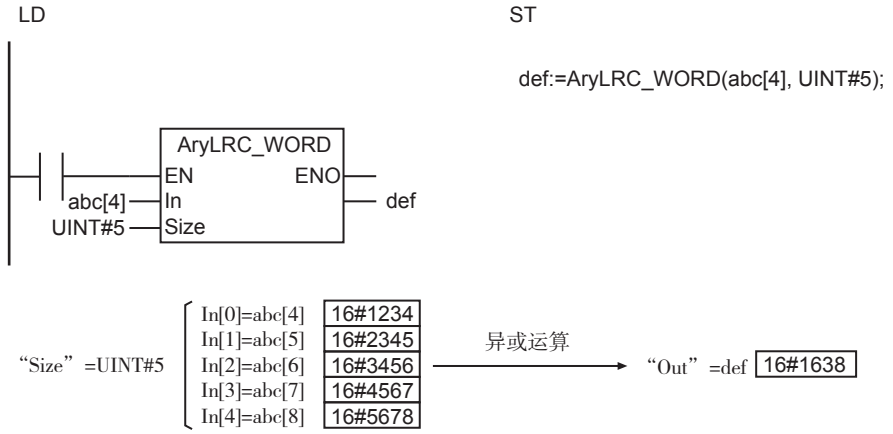
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○	○	○	○															
Size							○													
Out	与In[ ]相同的数据类型																			

功能

计算运算对象数组In[ ]的In[0]之后的 “Size”个数组元素的LRC值(异或)。
指令名称因In[ ]的数据类型而异。例如，In[ ]为WORD型时，指令名称为AryLRC_WORD。

传输至In[ ]的输入参数务必也像array[3]那样加上元素编号后再指定。

AryLRC_WORD指令下，“Size”=UINT#5时的示例如下所示。

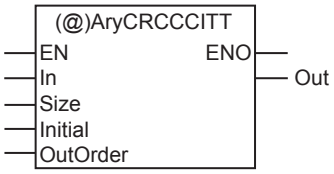


使用注意事项

- 请将In[ ]和 “Out”的数据类型设为相同。
- “Size”的值为0时，“Out”的值为16#00。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过In[ ]的数组区域时。

AryCRCCCITT

按照XMODEM方式计算数组的CRC-CCITT值。

指令	名称	FB/FUN	图形表现	ST表现
AryCRCCCITT	CRC-CCITT值计算(数组)	FUN		Out:=AryCRCCCITT(In, Size, Initial, OutOrder);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	-	(*)
Size	运算对象的元素数		In[ ]的元素数			1
Initial	初始值		CRC-CCITT值计算的初始值			0
OutOrder	字节顺序		处理 “In”的字节顺序	_LOW_HIGH, _HIGH_LOW		_HIGH _LOW
Out	CRC-CCITT值	输出	CRC-CCITT值	遵从数据类型	-	-

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	
Size							○													
Initial			○																	
OutOrder	枚举体_eBYTE_ORDER 枚举元素参阅功能说明																			
Out			○																	

功能

按照XMODEM方式计算运算对象数组In[ ]的In[0]之后的 “Size”个数组元素的CRC-CCITT值。

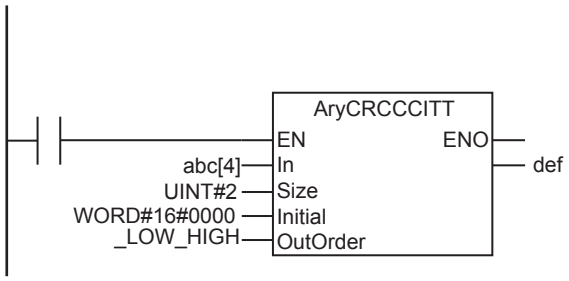
计算CRC-CCITT值时的初始值由 “Initial”指定。字节顺序由 “OutOrder”指定。
“OutOrder”的数据类型为枚举体_eBYTE_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_LOW_HIGH	低位字节在前，高位字节在后
_HIGH_LOW	高位字节在前，低位字节在后

传输至In[ ]的输入参数务必也像array[3]那样加上元素编号后再指定。

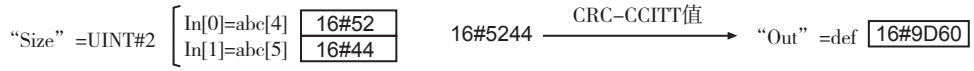
“Size”=UINT#2，“Initial”=WORD#16#0000，“OutOrder”=_LOW_HIGH时的示例如下所示。

LD



ST

```
def:=AryCRCCCITT(abc[4], UINT#2,
                 WORD#16#0000,
                 _LOW_HIGH);
```



使用注意事项

- “Size”的值为0时，“Out”的值为WORD#16#0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “OutOrder”的值超过有效范围时。
  - “Size”的值超过In[ ]的数组区域时。

AryCRC16

按照MODBUS方式计算数组的CRC-16值。

指令	名称	FB/FUN	图形表现	ST表现
AryCRC16	CRC-16值计算(数组)	FUN	(@)AryCRC16 EN ENO In Out Size Initial OutOrder	Out:=AryCRC16(In, Size, Initial, OutOrder);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In[ ] 数组	运算对象数组	输入	运算对象数组	遵从数据类型	–	(*)
Size	运算对象的元素数		In[ ]的元素数			1
Initial	初始值		CRC-16值计算的初始值			16#FFFF
OutOrder	字节顺序		处理 “In”的字节顺序	_LOW_HIGH, _HIGH_LOW		_LOW _HIGH
Out	CRC-16值	输出	CRC-16值	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In[ ] 数组		○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	
Size							○													
Initial			○																	
OutOrder	枚举体_eBYTE_ORDER 枚举元素参阅功能说明																			
Out			○																	

功能

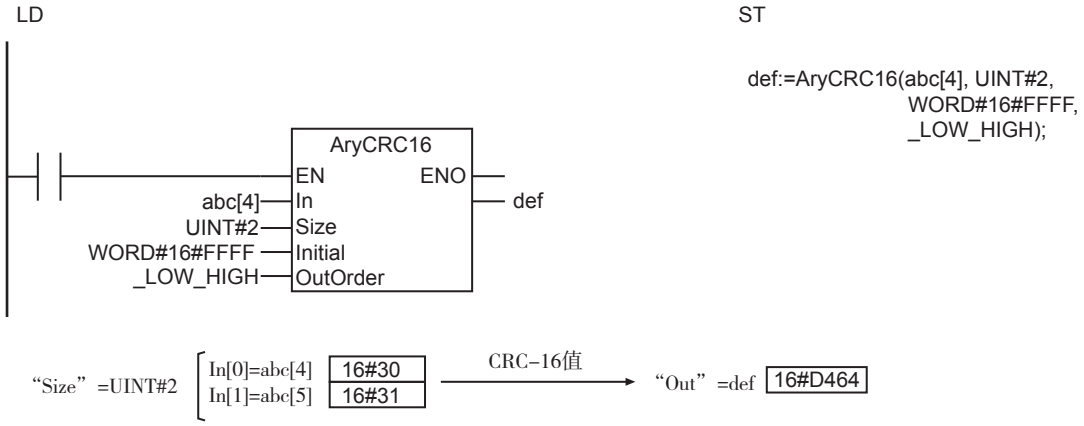
按照MODBUS方式计算运算对象数组In[ ]的In[0]之后的 “Size”个数组元素的CRC-16值。

计算CRC-16值时的初始值由 “Initial”指定。字节顺序由 “OutOrder”指定。
“OutOrder”的数据类型为枚举体_eBYTE_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_LOW_HIGH	低位字节在前，高位字节在后
_HIGH_LOW	高位字节在前，低位字节在后

传输至In[ ]的输入参数务必也像array[3]那样加上元素编号后再指定。

“Size”=UINT#2，“Initial”=WORD#16#FFFF，“OutOrder”=_LOW_HIGH时的示例如下所示。



使用注意事项

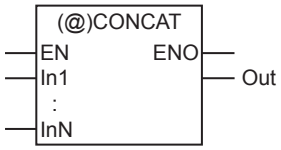
- “Size”的值为0时，“Out”的值为WORD#16#0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “OutOrder”的值超过有效范围时。
  - “Size”的值超过In[ ]的数组区域时。

字符串指令

指令	名称	页码
CONCAT	字符串・连接	2–538
LEFT/RIGHT	字符串・从左侧提取 字符串・从右侧提取	2–540
MID	字符串・从任意位置提取	2–542
FIND	字符串・检索	2–544
LEN	字符串・长度检测	2–546
REPLACE	字符串・置换	2–547
DELETE	字符串・删除	2–549
INSERT	字符串・插入	2–551
GetByteLen	字符串・获取字节数	2–553
ClearString	字符串・清除	2–554
ToUCase/ToLCase	字符串・大写字母转换 字符串・小写字母转换	2–555
TrimL/TrimR	字符串・左侧调整 字符串・右侧调整	2–557
AddDelimiter	带分隔符的字符串写入	2–559
SubDelimiter	字符串读取分隔符删除	2–569

CONCAT

连接2 ~ 5个字符串。

指令	名称	FB/FUN	图形表现	ST表现
CONCAT	字符串・连接	FUN		Out:=CONCAT(In1, …, InN);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1 ~ InN	连接对象	输入	待连接的字符串　N为2 ~ 5	遵从数据类型	–	''(*)
Out	连接结果	输出	连接结果字符串	遵从数据类型	–	–

* 省略了连接到InN的输入参数时，初始值不适用，编连时会发生异常。例如N=3时，如果省略与In1和In2连接的输入参数，则初始值适用；但如果省略与In3连接的输入参数，则编连时会发生异常。

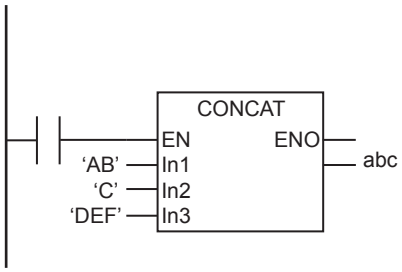
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1 ~ InN																				○
Out																				○

功能

将2～5个连接对象字符串 “In1” ～ “InN” 按此顺序连接。结尾带NULL字符。

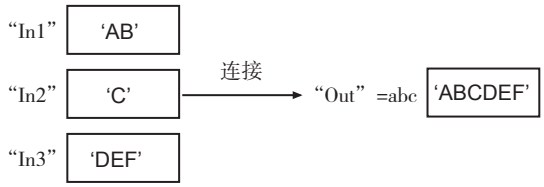
“In1”='AB'，“In2”='C'，“In3”='DEF'时的示例如下所示。变量abc的值为'ABCDEF'。

LD



ST

```
abc:=CONCAT('AB', 'C', 'DEF');
```

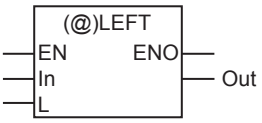
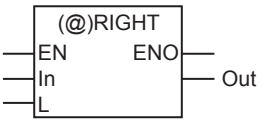


使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

LEFT/RIGHT

从字符串提取指定数量的字符串。
LEFT ：从字符串的左侧(开头)提取。
RIGHT：从字符串的右侧(结尾)提取。

指令	名称	FB/FUN	图形表现	ST表现
LEFT	字符串・从左侧提取	FUN		Out:=LEFT(In, L);
RIGHT	字符串・从右侧提取	FUN		Out:=RIGHT(In, L);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	提取对象	输入	待提取的对象字符串	遵从数据类型	–	''
L	字符数		待提取的字符数	0 ~ 1985		1
Out	提取结果	输出	提取结果字符串	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
L							○													
Out																				○

功能

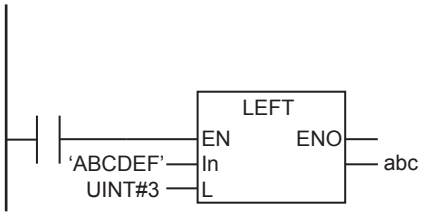
从提取对象字符串“In”中提取字符数“L”指定字符数的字符串。提取结果“Out”的结尾带NULL字符。

- LEFT

从“In”的左侧(开头)提取。

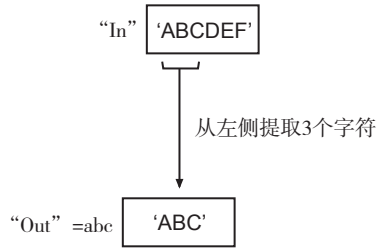
“In”='ABCDEF'，“L”=UINT#3时的示例如下所示。变量abc的值为'ABC'。

LD



ST

```
abc:=LEFT('ABCDEF', UINT#3);
```

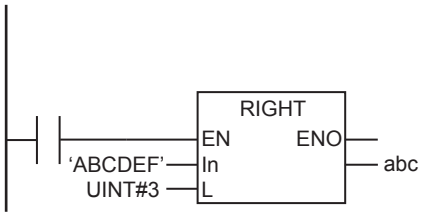


- RIGHT

从“In”的右侧(结尾)提取。

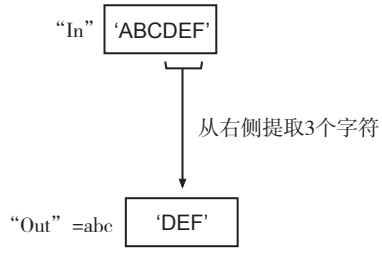
“In”='ABCDEF'，“L”=UINT#3时的示例如下所示。变量abc的值为'DEF'。

LD



ST

```
abc:=RIGHT('ABCDEF', UINT#3);
```

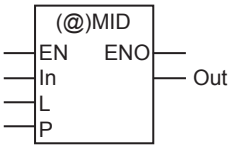


使用注意事项

- “L”的值大于“In”的字符数或在有效范围内时，不会发生异常，将“In”的所有字符复制到“Out”。
- “L”的值为0时，不会发生异常，仅将NULL字符代入“Out”。
- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的文字代码异常时。

MID

从字符串的任意位置提取指定数量的字符串。

指令	名称	FB/FUN	图形表现	ST表现
MID	字符串・从任意位置提取	FUN		Out:=MID(In, L, P);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	提取对象	输入	待提取的对象字符串	遵从数据类型	–	''
L	字符数		待提取的字符数	0 ~ 1985		1
P	提取开始位置		待提取的开始位置	1 ~ 1985		
Out	提取结果	输出	提取结果字符串	遵从数据类型	–	–

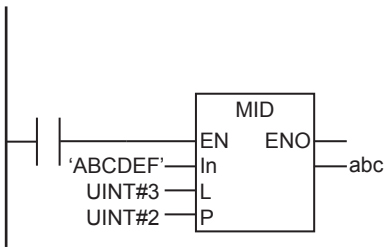
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
L							○													
P							○													
Out																				○

功能

从对象字符串 “In”中提取字符数 “L”指定数量的字符串。提取的开始位置由开始位置 “P”指定。提取结果 “Out”的结尾带NULL字符。

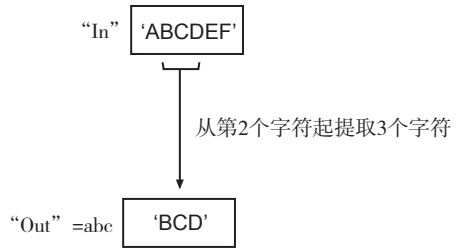
“In”='ABCDEF'，“L”=UINT#3，“P”=UINT#2时的示例如下所示。变量abc的值为'BCD'。

LD



ST

```
abc:=MID('ABCDEF', UINT#3, UINT#2);
```



使用注意事项

- “L”的值为0时，不会发生异常，仅将NULL字符代入 “Out”。
- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的文字代码异常时。
  - “In”从 “P”值所示位置起无 “L”值所示数量的字符时。
  - “P”的值为0时。

FIND

检索字符串中指定字符串的位置。

指令	名称	FB/FUN	图形表现	ST表现
FIND	字符串・检索	FUN	(@)FIND EN ENO In1 Out In2	Out:=FIND(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	检索对象	输入	检索的对象字符串	遵从数据类型	–	''
In2	检索字符串		待检索的字符串			
Out	检索结果	输出	检索结果	0 ~ 1985	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																				○
In2																				○
Out							○													

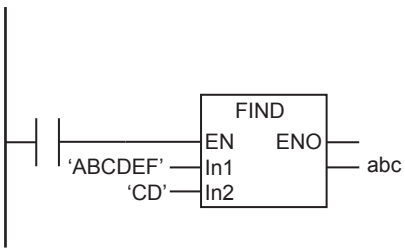
功能

对检索对象“In1”中的检索字符串“In2”的位置进行检索。检索结果“Out”中代入“In2”位于“In1”开头起的第几个字符。

“In1”中不存在“In2”时，“Out”的值为0。

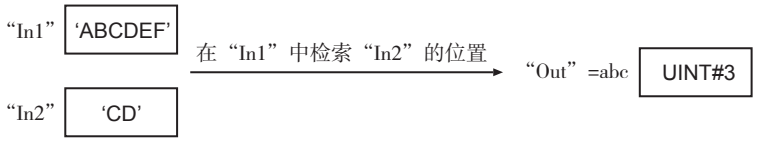
“In1”='ABCDEF'，“In2”='CD'时的示例如下所示。变量abc的值为UINT#3。

LD



ST

```
abc:=FIND('ABCDEF', 'CD');
```



使用注意事项

- 请将 “In2”的字符数设为小于 “In1”的字符数。否则，“Out”的值为0。
- “In1”中含有多个 “In2”时，“Out”的值为从 “In1”的开头起检索首个 “In2”的位置。
- “In1”、“In2”的值均仅有NULL字符时，“Out”的值为1。
- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In1”或 “In2”的文字代码异常时。

LEN

计算字符串的字符数。

指令	名称	FB/FUN	图形表现	ST表现
LEN	字符串・长度检测	FUN	(@)LEN EN ENO In Out	Out:=LEN(In);

变量

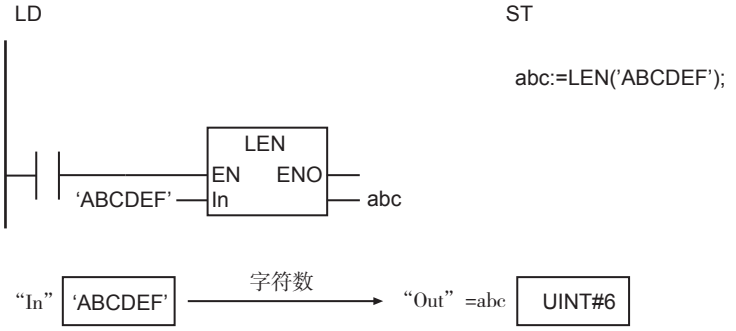
	名称	输入/输出	内容	有效范围	单位	初始值
In	长度检测对象	输入	长度检测的对象字符串	遵从数据类型	–	''
Out	检测结果	输出	长度检测结果	0 ~ 1985	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out							○													

功能

计算长度检测对象字符串 “In”的字符数。字符数中不包含 “In”结尾的NULL字符。

“In”='ABCDEF'时的示例如下所示。变量abc的值为UINT#6。

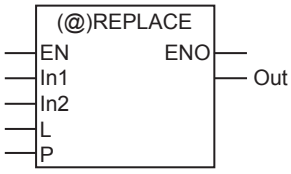


使用注意事项

- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的文字代码异常时。

REPLACE

用其它字符串置换部分字符串。

指令	名称	FB/FUN	图形表现	ST表现
REPLACE	字符串・置换	FUN		Out:=REPLACE(In1, In2, L, P);

变量

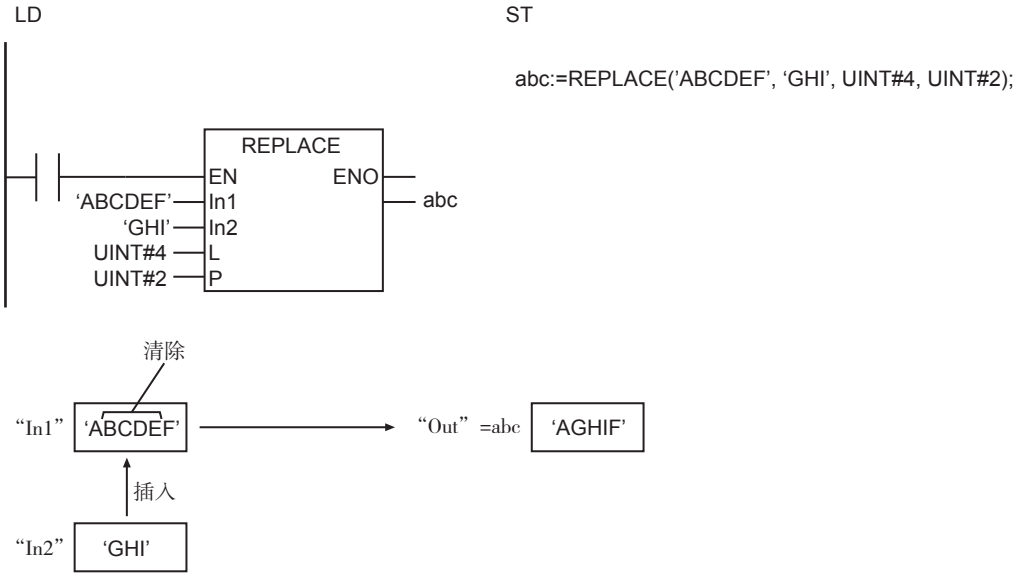
	名称	输入/输出	内容	有效范围	单位	初始值
In1	置换对象	输入	置换的对象字符串	遵从数据类型	–	''
In2	置换字符串		待置换的字符串			
L	字符数		待置换的字符数	0 ~ 1985		1
P	置换开始位置		开始置换的位置	1 ~ 1985		
Out	置换结果	输出	置换后的字符串	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																				○
In2																				○
L							○													
P							○													
Out																				○

功能

以置换字符串 “In2” 置换部分置换对象字符串 “In1”。
首先，从 “In1” 的置换开始位置 “P” 值所示的位置起删除字符数 “L” 值的字符。在空白位置嵌入 “In2”。置换结果 “Out” 的结尾带NULL字符。

“In1” ='ABCDEF'，“In2” ='GHI'，“P” =UINT#2，“L” =UINT#4 时的示例如下所示。变量 abc 的值为 'AGHIF'。

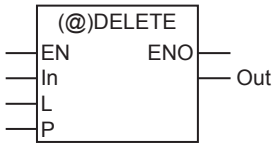


使用注意事项

- “L” 的值为0时，不会发生异常，将 “In1” 的所有字符复制到 “Out”。
- “In2” 的值为0时，删除从 “In1” 开头起的 “P” 字符后的 “L” 字符。
- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out” 不变。
  - “In1” 的文字代码异常时。
  - “In1” 从 “P” 值所示的位置起无 “L” 值所示数量的字符时。
  - “P” 的值为0时。
  - 置换后的字符串长度超过1986个字节的大小时。

DELETE

删除部分或全部字符串。

指令	名称	FB/FUN	图形表现	ST表现
DELETE	字符串・删除	FUN		Out:=DELETE(In, L, P);

变量

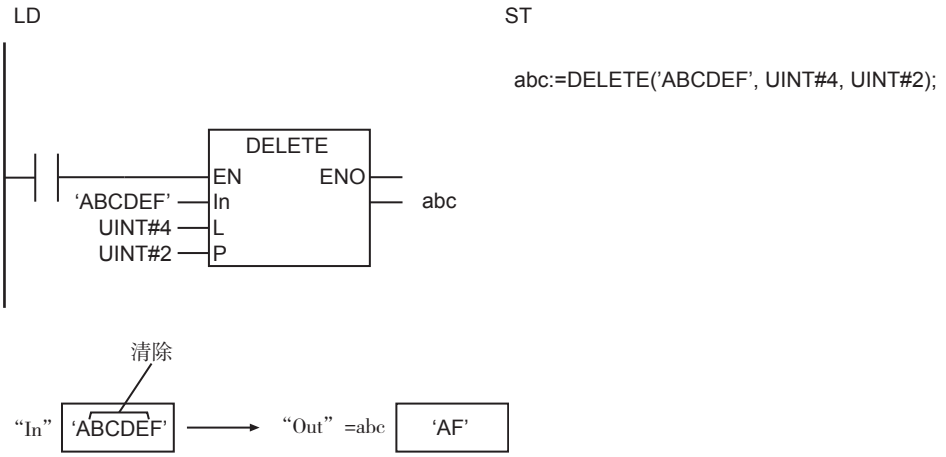
	名称	输入/输出	内容	有效范围	单位	初始值
In	删除对象	输入	删除的对象字符串	遵从数据类型	–	''
L	字符数		待删除的字符数	0 ~ 1985		1
P	删除开始位置		开始删除的位置	1 ~ 1985		
Out	删除结果	输出	删除后的字符串	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
L							○													
P							○													
Out																				○

功能

从删除对象字符串 “In”的删除开始位置 “P”值所示的位置起删除字符数 “L”值的字符。删除结果“Out”的结尾带NULL字符。

“In”='ABCDEF'，“L”=UINT#4，“P”=UINT#2时的示例如下所示。变量abc的值为'AF'。

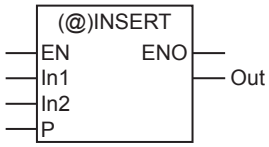


使用注意事项

- “L”的值为0时，不会发生异常，将 “In”的所有字符被复制到 “Out”。
- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的文字代码异常时。
  - “In”从 “P”值所示位置起无 “L”值所示数量的字符时。
  - “P”的值为0时。

INSERT

在字符串中插入其它字符串。

指令	名称	FB/FUN	图形表现	ST表现
INSERT	字符串・插入	FUN		Out:=INSERT(In1, In2, P);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	插入对象	输入	插入的对象字符串	遵从数据类型	–	''
In2	插入字符串		待插入的字符串			
P	插入开始位置		开始插入的位置	0 ~ 1985		0
Out	插入结果	输出	插入后的字符串	遵从数据类型	–	–

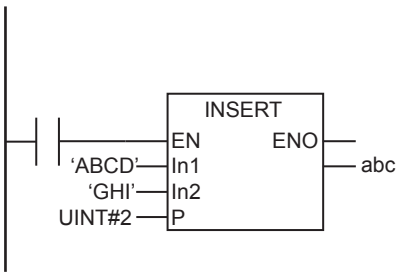
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																				○
In2																				○
P							○													
Out																				○

功能

在插入对象字符串 “In1”的插入开始位置 “P”值所示的位置，将插入字符串 “In2”插入。插入结果“Out”的结尾带NULL字符。

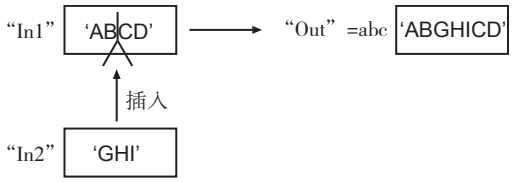
“In1”='ABCD'，“In2”='GHI'，“P”=UINT#2时的示例如下所示。变量abc的值为'ABGHICD'。

LD



ST

```
abc:=INSERT('ABCD', 'GHI', UINT#2);
```



参考

“P”的值为0时，“In2”后连接 “In1”。

使用注意事项

- 多字节字符也算作1个字符。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In1”的文字代码异常时。
  - “P”的值大于 “In1”的字符数时。
  - 插入后的字符串长度超过1986个字节的大小时。

GetByteLen

对字符串的字节数进行计数。

指令	名称	FB/FUN	图形表现	ST表现
GetByteLen	字符串・获取字节数	FUN	(@)GetByteLen EN ENO In Out	Out:=GetByteLen(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	计数对象	输入	对字节数进行计数的对象字符串	遵从数据类型	–	''
Out	字节数	输出	字节数	0 ~ 1985	字节	–

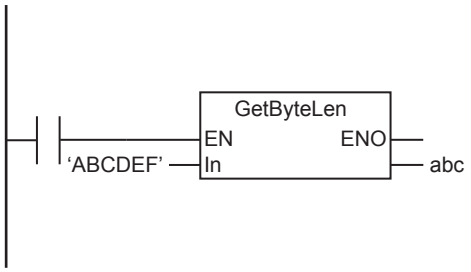
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out							○													

功能

对计数对象字符串 “In”的开头到结尾的字节数进行计数。字节数中不包含字符串结尾的NULL字符。

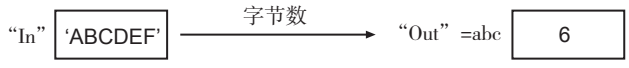
“In”='ABCDEF'时的示例如下所示。变量abc的值为6。

LD



ST

```
abc:=GetByteLen('ABCDEF');
```

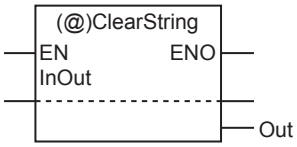


参考

“In”仅含ASCII字符时，本指令的结果与LEN指令的结果一致。

ClearString

清除字符串。

指令	名称	FB/FUN	图形表现	ST表现
ClearString	字符串・清除	FUN		ClearString(InOut);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
InOut	清除字符串	输入输出	待清除的字符串	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

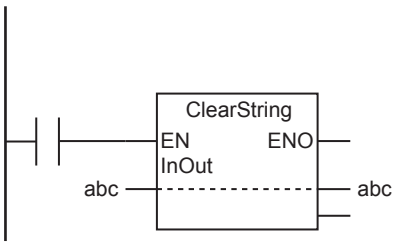
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
InOut																				○
Out	○																			

功能

将清除字符串 “InOut”清除。在 “InOut”的所有区域中保存NULL字符。

示例如下所示。STRING型变量abc的值均为NULL字符。

LD



ST

```
ClearString(abc);
```

在“InOut”的所有区域中保存NULL字符。

abc为5个字符的STRING型变量时

“InOut”=abc

NULL	NULL	NULL	NULL	NULL

使用注意事项

在ST程序中使用本指令时，不使用返回值 “Out”。

ToUCase/ToLCase

ToUCase：将字符串中的半角字母均转换为大写字母。
ToLCase：将字符串中的半角字母均转换为小写字母。

指令	名称	FB/FUN	图形表现	ST表现
ToUCase	字符串・大写字母转换	FUN	(@)ToUCase EN ENO In Out	Out:=ToUCase(In);
ToLCase	字符串・小写字母转换	FUN	(@)ToLCase EN ENO In Out	Out:=ToLCase(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	转换对象	输入	转换的对象字符串	遵从数据类型	-	''
Out	转换结果	输出	转换后的字符串	遵从数据类型	-	-

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out																				○

功能

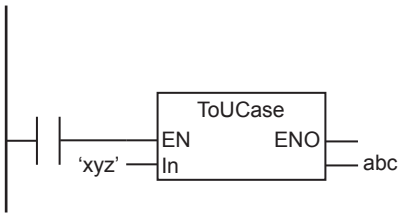
- **ToUCase**
  将转换对象字符串 “In”的半角字母均转换为大写字母。

- **ToLCase**
  将转换对象字符串 “In”的半角字母均转换为小写字母。

所有指令在输出时，字符串结尾均含NULL字符。非半角字母的字符不受影响。

ToUCase指令下，“In”='xyz'时的示例如下所示。变量abc的值为'XYZ'。

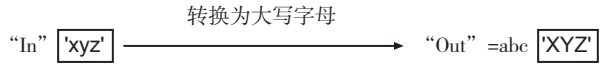
LD



ST

```
abc:=ToUCase('xyz');
```

将“In”的半角字母均转换为大写字母。

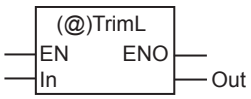


使用注意事项

- 全角字母不属于转换对象。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的文字代码异常时。

TrimL/TrimR

TrimL：删除字符串开头的空格。
TrimR：删除字符串末尾的空格。

指令	名称	FB/FUN	图形表现	ST表现
TrimL	字符串・左侧调整	FUN		Out:=TrimL(In);
TrimR	字符串・右侧调整	FUN		Out:=TrimR(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	删除对象	输入	删除的对象字符串	遵从数据类型	–	''
Out	删除结果	输出	删除后的字符串	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Out																				○

功能

- TrimL

  将删除对象字符串 “In”的开头到首字符之间的空格删除。开头无空格时，不进行任何处理。

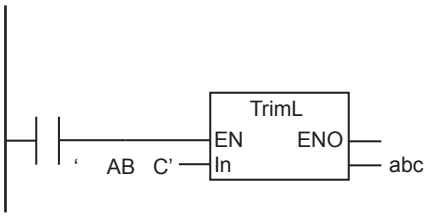
- TrimR

  将删除对象字符串 “In”的尾字符到末尾之间的空格删除。末尾无空格时，不进行任何处理。

所有指令在输出时，字符串结尾均含NULL字符。ASCII码的空格(16#20)与中文全角空格(16#E38080)均视为空格。

TrimL指令下，“In”=' AB C'时的示例如下所示。变量abc的值为'AB C'。

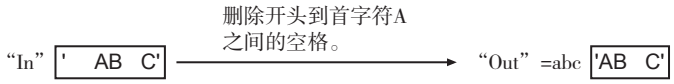
LD



ST

```
abc:=TrimL('  AB  C');
```

删除“In”的开头到首字符之间的空格。



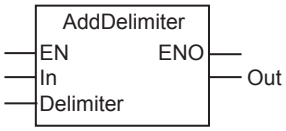
使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “In”的文字代码异常时。

AddDelimiter

将整个结构体的值转换为带分隔符的字符串。

指令	名称	FB/FUN	图形表现	ST表现
AddDelimiter	带分隔符的字符串写入	FUN		Out:=AddDelimiter(In,Delimiter);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	输入结构体	输入	作为字符串转换对象的整个结构体	遵从结构要素的数据类型	–	*1
Delimiter	分隔符		分隔符	_COMMA _TAB _SEMICOLON _SPACE	–	_COMMA
Out	返回值	输出	带分隔符的字符串	最大1986字节(1985个半角英数字字符 + 结尾NULL字符)	–	–

*1 省略输入参数时，初始值不适用。编译时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	整个结构体																			
Delimiter	枚举体_eDELIMITER　枚举元素参阅功能说明																			
Out																				○

功能

将输入结构体 “In”的结构要素的值从开头的结构要素起依次转换为字符串，由分隔符 “Delimiter”隔开后进行连接。将连接的字符串输出至返回值 “Out”。“Out”的结尾带NULL字符。

“Delimiter”的数据类型为枚举体_eDELIMITER。枚举元素的含义如下所示。

枚举元素	含义
_COMMA	','(逗号)
_TAB	'$T'(标签)
_SEMICOLON	';'(分号)
_SPACE	' '(空格)

根据数据类型的不同，“In”的结构要素的值的转换如下所示。

● 布尔型

FALSE转换为'0'，TRUE转换为'1'。

● 位串型

将以英数字表现的内容转换为16进制字符串。字符串中不包含表示16进制的前缀'16#'。
结构要素的值不到结构要素的数据类型的最大位数时，高位以'0'填充。即进行补零。
如下所示，字符串的字符数因数据类型而异。

结构要素的数据类型	字符数
BYTE	2个半角英数字字符
WORD	4个半角英数字字符
DWORD	8个半角英数字字符
LWORD	16个半角英数字字符

几个示例如下所示。

结构要素的值	转换后的字符串
BYTE#16#AB	'AB'
LWORD#16#0123	'0000000000000123'

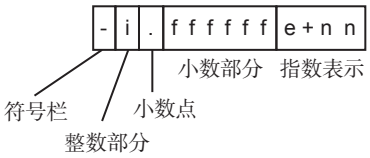
● 整数型

将整数的值转换为字符串。高位的0不输出为字符串。结构要素的值为负数时，在字符串的开头加上减号'-'。
几个示例如下所示。

结构要素的值	转换后的字符串
UINT#0012	'12'
LINT#-12	'-12'

● 实数型

转换结构要素的值后的字符串的构成如下所示。



项目	内容
符号栏	仅结构要素的值为负数时，加上'-'(减号)。 结构要素的值为正数时，不加'+'(加号)。
整数部分	始终以1个数位表示。
小数点	即使不是小数，也始终表示结构要素的值。
小数部分	结构要素为REAL型时，以6个数位表示；为LREAL型时，以14个数位表示。
指数表示	始终表示。'e'表示指数e。 nn的位数为2或3。 结构要素值的绝对值大于1.0时，nn的符号为'+'(加号)，小于1.0时，nn的符号为'-'(减号)。结构要素的值为0时，nn的符号为'+'(加号)。

结构要素的值为无穷大、非数时，字符串如下所示。

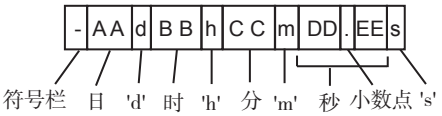
结构要素的值	字符串
+∞	'inf'
−∞	'-inf'
非数	'nan'或'-nan'

几个示例如下所示。

结构要素的值	转换后的字符串
REAL#3.14e1	'3.140000e+01'
REAL#-123.4567	'-1.234567e+02'
REAL#0	'0.000000e+00'
LREAL#0.00123456789	'1.23456789000000e-03'
LREAL#1.0e308	'1.00000000000000e+308'

● 持续时间型

转换结构要素的值后的字符串的构成如下所示。



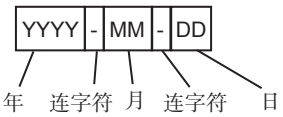
项目	内容
符号栏	仅结构要素的值为负数时，加上'-'(减号)。 结构要素的值为正数时，不加'+'(加号)。
日	始终表示。值的范围为0 ~ 106751。不在高位上补零。
时	始终以2个数位表示。值的范围为00 ~ 23。
分	始终以2个数位表示。值的范围为00 ~ 59。
秒	始终表示。DD的值始终以2个数位表示，范围为00 ~ 59。EE的值始终以9个数位表示，范围为000000000 ~ 999999999。
'd'、'h'、'm'、's'、小数点	始终表示。

几个示例如下所示。

结构要素的值	转换后的字符串
T#-180122000ms	'-2d02h02m02.000000000s'
T#100d2h3m5.678s	'100d02h03m05.678000000s'
T#2h3m5.678s	'0d02h03m05.678000000s'

● 日期型

转换结构要素的值后的字符串的构成如下所示。

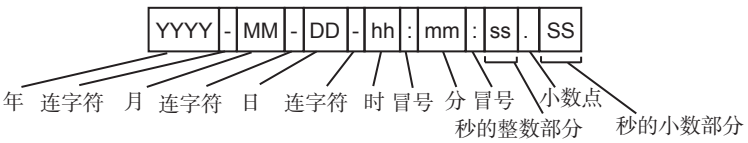


月、日分别转换为2个数位的字符串。
示例如下所示。

结构要素的值	转换后的字符串
D#2010-1-2	'2010-01-02'

● 日期时刻型

转换结构要素的值后的字符串的构成如下所示。

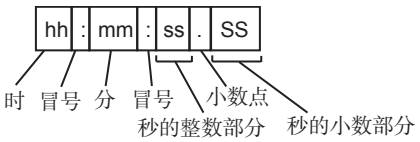


月MM、日DD、时hh、分mm、秒的整数部分ss分别转换为2个数位的字符串。秒的小数部分SS转换为9个数位的字符串。
示例如下所示。

结构要素的值	转换后的字符串
DT#1994-09-23-12:16:8.12	'1994-09-23-12:16:08.120000000'

● 时刻型

转换结构要素的值后的字符串的构成如下所示。



时hh、分mm、秒的整数部分ss分别转换为2个数位的字符串。秒的小数部分SS转换为9个数位的字符串。
示例如下所示。

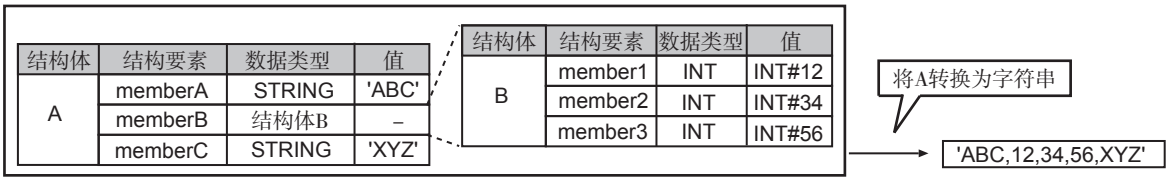
结构要素的值	转换后的字符串
TOD#2:16:28.12	'02:16:28.120000000'

● 字符串型

保持原有的字符串。但不含结尾的NULL字符。
例如，结构要素的值为结尾含NULL字符的'ABC'时，转换为结尾不含NULL字符的字符串'ABC'。

● 结构体

结构要素的数据类型从开头起依次转换结构要素的值，直至非结构体类型的层。按照各结构要素的数据类型的转换规则，将结构要素的值转换为字符串。
例如，结构体A的结构要素的数据类型中含有结构体B时，转换如下所示。本例中，将逗号用作分隔符。



● 枚举体

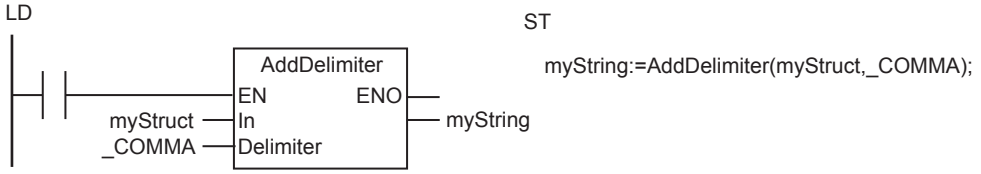
将枚举体的值转换为DINT型的值。
例如，枚举体Color获取的枚举元素为red、yellow、green。与各枚举元素对应的值：red为1；yellow为2；green为3。
枚举体Color的结构要素值为yellow时，字符串为'2'。

● 数组

各元素间由分隔符隔开。按照数组的数据类型的转换规则转换各元素的值。仅可转换为1维数组。
例如，假设存在INT型的数组myArray[0..2]。myArray[0]的值为INT#225，myArray[1]的值为INT#–128，myArray[2]的值为INT#0，分隔符为逗号时，字符串为 ‘225,–128,0'。

记述示例

将结构体 “myStruct”转换为字符串 “myString”时的描述示例如下所示。分隔符使用','(逗号)。



结构体	结构要素	数据类型	值
myStruct	member1	STRING	'XYZ'
	member2	INT	INT#12
	member3	WORD	WORD#16#00AB
	member4	REAL	REAL#3.14e0

转换为字符串
“Out” =myString 'XYZ,12,00AB,3.140000e+00'

参考

- 将值写入SD存储卡内的指定CSV文件时，如果同时使用“FilePuts指令(P.2-1338)”和本指令，则较为便捷。使用方法请参阅示例程序。
- 通过本指令转换的字符串可通过 “SubDelimiter指令(P.2-569)”读取为结构体的结构要素的值。

使用注意事项

- 请勿将分隔符用于部分 “In”的结构要素的值。如果将分隔符用于部分 “In”的结构要素的值，则通过SubDelimiter指令将字符串转换为结构体的结构要素的值时，会无法正确转换。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - 转换结果的字符串结尾含有NULL字符，且超过1986字节时
  - “In”中含有2维以上的数组的结构要素时
  - “In”中含有联合体的结构要素时

版本相关信息

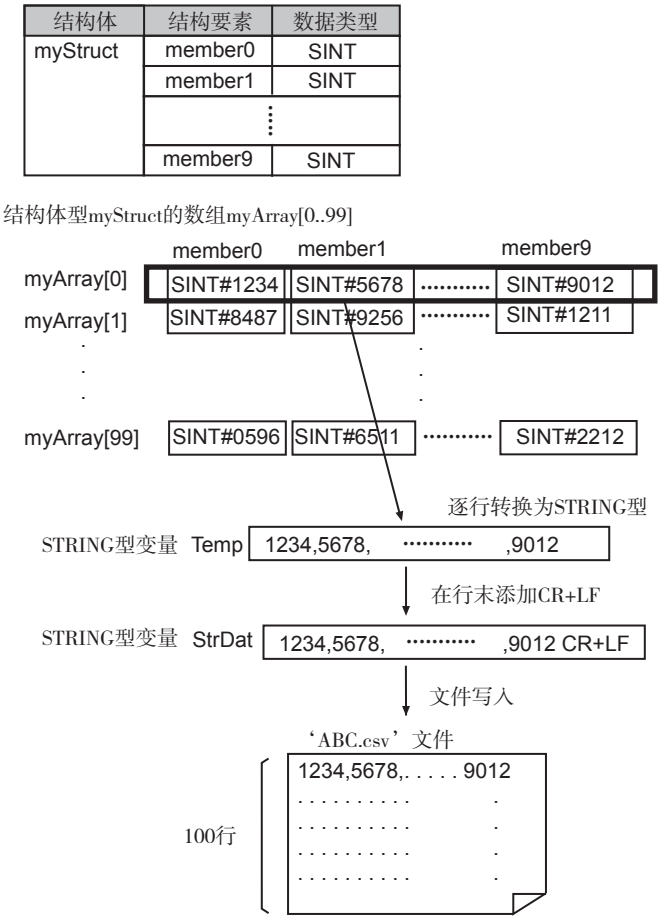
本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

假设存在结构要素中含有10个SINT型变量的结构体myStruct。以100行的CSV文件格式将结构体myStruct的数组myArray[0..99]的内容保存至SD存储卡的'ABC.csv'文件。
1行中排列10个数组元素的结构要素的值，作为字符串，各值之间插入逗号。行末添加换行代码CR + LF。

处理步骤如下所示。

1. 使用FileOpen指令，打开'ABC.csv'文件。
2. 通过AddDelimiter指令，仅将1行myArray[ ]的元素转换为STRING型的变量Temp。
3. 通过CONCAT指令，组合Temp和CR+LF，保存至STRING型的变量StrDat。
4. 将StrDat写入文件。
5. 在100行中重复执行上述2 ~ 4。
6. 使用FileClose指令，关闭文件。



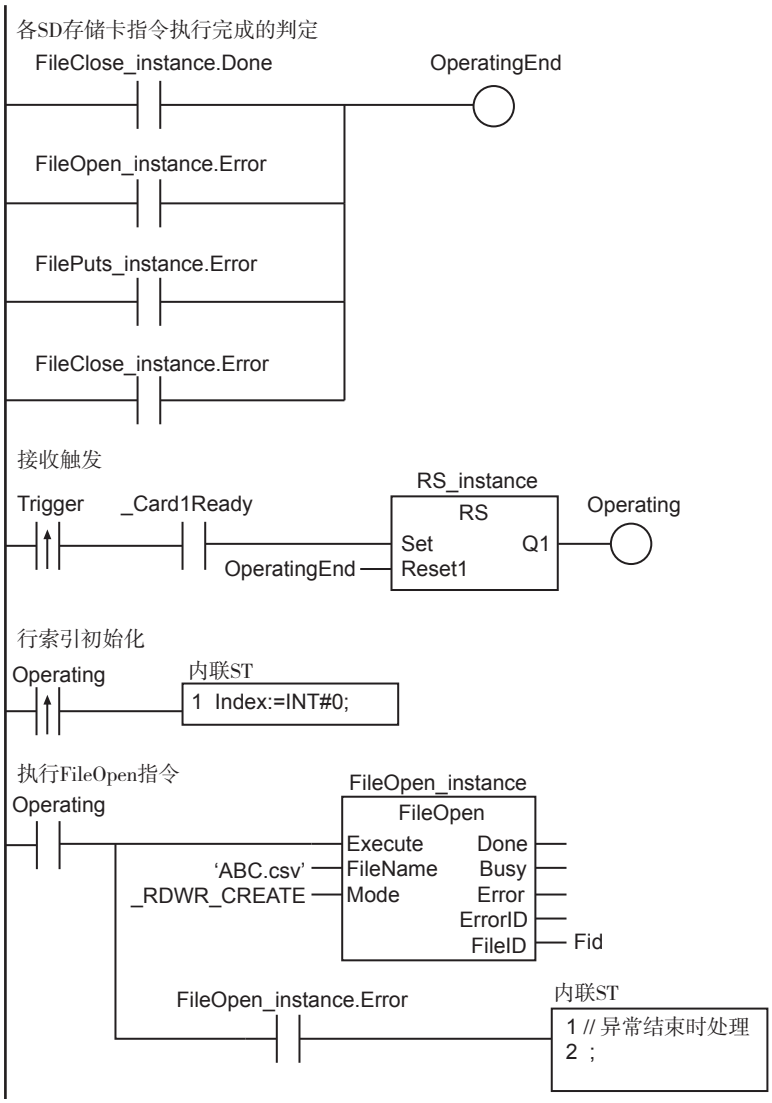
● 数据类型的定义

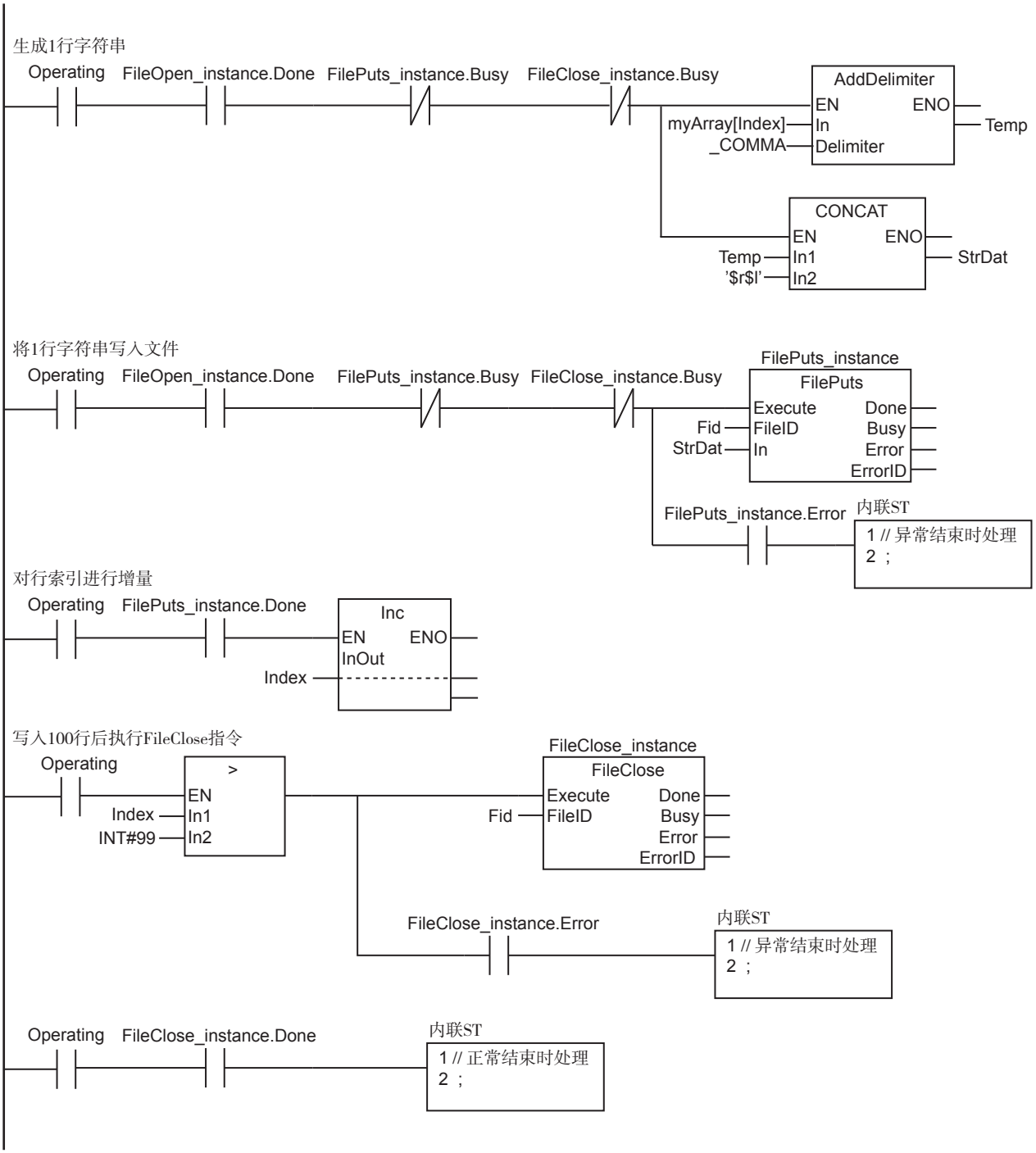
名称	数据类型	注释
myStruct	STRUCT	结构体
member0	SINT	结构要素
member1	SINT	结构要素
member2	SINT	结构要素
member3	SINT	结构要素
member4	SINT	结构要素
member5	SINT	结构要素
member6	SINT	结构要素
member7	SINT	结构要素
member8	SINT	结构要素
member9	SINT	结构要素

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	False	处理结束
	Trigger	BOOL	False	执行条件
	Operating	BOOL	False	处理中
	Index	INT	0	索引
	Fid	DWORD	16#0	文件ID
	StrDat	STRING[256]	''	字符串数据
	myArray	ARRAY[0..99] OF myStruct	[100((member0:=0,member1:=0, member2:=0,member3:=0,member4:=0, member5:=0,member6:=0,member7:=0, member8:=0,member9:=0))]	数值数据
	Temp	STRING[256]	''	临时
	RS_instance	RS		
	FileOpen_instance	FileOpen		
	FilePuts_instance	FilePuts		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志




生成1行字符串
Operating
FileOpen_instance.Done
FilePuts_instance.Busy
FileClose_instance.Busy
AddDelimiter
EN
ENO
myArray[Index]
In
Temp
_COMMA
Delimiter
CONCAT
EN
ENO
Temp
In1
StrDat
'rl'
In2
将1行字符串写入文件
Operating
FileOpen_instance.Done
FilePuts_instance.Busy
FileClose_instance.Busy
FilePuts_instance
FilePuts
Execute
Done
Fid
FileID
Busy
StrDat
In
Error
ErrorID
FilePuts_instance.Error
内联ST
1 // 异常结束时处理
2 ;
对行索引进行增量
Operating
FilePuts_instance.Done
Inc
EN
ENO
InOut
Index
写入100行后执行FileClose指令
Operating
>
EN
Index
In1
INT#99
In2
FileClose_instance
FileClose
Execute
Done
Fid
FileID
Busy
Error
ErrorID
FileClose_instance.Error
内联ST
1 // 异常结束时处理
2 ;
Operating
FileClose_instance.Done
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	False	执行条件
	LastTrigger	BOOL	False	上个任务周期的Trigger的值
	OperatingStart	BOOL	False	处理开始
	Operating	BOOL	False	处理中
	Stage	INT	0	状态变化
	Index	INT	0	索引
	Fid	DWORD	16#0	文件ID
	StrDat	STRING[256]	''	字符串数据
	myArray	ARRAY[0..99] OF myStruct	[100((member0:=0,member1:=0, member2:=0,member3:=0,member4:=0, member5:=0,member6:=0,member7:=0, member8:=0,member9:=0))]	数值数据
	Temp	STRING[256]	''	临时
	FileOpen_instance	FileOpen		
	FilePuts_instance	FilePuts		
	FileClose_instance	FileClose		

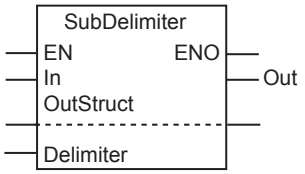
外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart  :=TRUE;
    Operating       :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileOpen_instance(Execute:=FALSE);
    FilePuts_instance(Execute:=FALSE);
    FileClose_instance(Execute:=FALSE);
    Stage           :=INT#1;
    Index           :=INT#0;     // 行索引初始化
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                          // 文件打开
        FileOpen_instance(
            Execute  :=TRUE,
            FileName:='ABC.csv',              // 文件名
            Mode     :=_RDWR_CREATE,          // 文件读取
            FileID  =>Fid);                   // 文件ID

        IF (FileOpen_instance.Done=TRUE) THEN
            Stage:=INT#2;              // 正常结束
        END_IF;

        IF (FileOpen_instance.Error=TRUE) THEN
            Stage:=INT#99;             // 异常结束
```

```
        END_IF;

    2 :                         // 创建1行字符串
        StrDat:=' ';

        Temp :=AddDelimiter(myArray[Index],_COMMA);
        StrDat:=CONCAT(In1:=Temp, In2:='$r$l');

        Stage:=INT#3;

    3 :                         // 字符串写入
        FilePuts_instance(
            Execute :=TRUE,
            FileID  :=Fid,
            In      :=StrDat);

        IF (FilePuts_instance.Done=TRUE) THEN
            Index:=Index+INT#1;

            IF (Index>INT#99) THEN  // 已写入100行
                Stage:=INT#4;
            ELSE
                FilePuts_instance(Execute:=FALSE);
                Stage:=INT#2;
            END_IF;
        END_IF;

        IF (FilePuts_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    4 :                         // 文件关闭
        FileClose_instance(
            Execute :=TRUE,
            FileID  :=Fid);             // 文件ID

        IF (FileClose_instance.Done=TRUE) THEN
            Operating:=FALSE;   // 正常结束
        END_IF;

        IF (FileClose_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    99 :                        // 异常结束处理
        Operating:=FALSE;
    END_CASE;
END_IF;
```

SubDelimiter

从字符串中读取由分隔符隔开的数据，保存为结构体结构要素的值。

指令	名称	FB/FUN	图形表现	ST表现
SubDelimiter	字符串读取分隔符删除	FUN		Out:=SubDelimiter(In, OutStruct, Delimiter);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	输入字符串	输入	以分割符隔开转换为结构体结构要素值的数据的字符串	最大1986字节(1985个半角英数字字符 + 结尾NULL字符)	–	''
Delimiter	分隔符		分隔符	_COMMA、_TAB、_SEMICOLON、_SPACE	–	_COMMA
OutStruct	保存位置结构体	输入输出	保存转换的数据的整个结构体	最大8192字节	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																				○
Delimiter	枚举体_eDELIMITER　枚举元素参阅功能说明																			
OutStruct	整个结构体																			
Out	○																			

功能

将输入字符串 “In”的分隔符 “Delimiter”隔开的字符串数据转换为保存位置结构体 “OutStruct”的结构要素的值，作为各结构要素的值从开头起依次保存。

“Delimiter”的数据类型为枚举体_eDELIMITER。枚举元素的含义如下所示。

枚举元素	含义
_COMMA	','(逗号)
_TAB	'$T'(标签)
_SEMICOLON	';'(分号)
_SPACE	' '(空格)

由 “In”的分隔符隔开的数据数多于 “OutStruct”的结构要素数时，忽略多余的数据。
由 “In”的分隔符隔开的数据数少于 “OutStruct”的结构要素数时，不变更数据不足部分的结构要素的值。
“OutStruct”的结构要素为结构体，即使“In”的数据少于该结构体的结构要素数，也保存中途之前的数据。
“OutStruct”的结构要素为数组，即使 “In”的数据少于该数组的元素数，也保存中途之前的数据。

由 “In”的分隔符隔开的数据为字符串型。根据 “OutStruct”的结构要素的数据类型的不同，字符串型的数据的转换如下所示。

● 布尔型

字符串型的数据为'FALSE'或'0'时，转换为FALSE。为'TRUE'或'1'时，转换为TRUE。
以下情况例外。

- 如果'0'或'1'之前有连续'0'，则忽略。
- 'FALSE'、'TRUE'不区分大小写字符。

字符串型的数据为'FALSE'、'TRUE'、'0'、'1'以外的值时，无法转换。

● 位串型

转换规则与 “STRING_TO_**(字符串→位串转换组)指令(P.2-297)”相同。
不是表示16进制数值的数据时，无法转换。

● 整数型

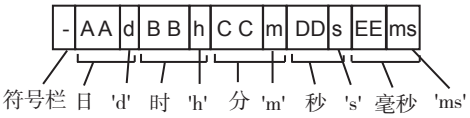
转换规则与 “STRING_TO_**(字符串→整数转换组)指令(P.2-295)”相同。
不是表示整数型数值的数据时，无法转换。

● 实数型

转换规则与 “STRING_TO_**(字符串→实数转换组)指令(P.2-299)”相同。
不是表示实数型数值的数据时，无法转换。

● 持续时间型

将下列结构的数据转换为持续时间型的值。



项目	内容
符号栏	无'+'(加号)或符号栏时，结构要素的值为正数。 '–'(减号)时，结构要素的值为负数。
日	AA值的小数点后12位之后舍去。
时	BB值的小数点后12位之后舍去。
分	CC值的小数点后11位之后舍去。
秒	DD值的小数点后10位之后舍去。
毫秒	EE值的小数点后7位之后舍去。

(注) 1 符号栏、天、时、分、秒、毫秒的值之前含有' '(空格)时，忽略' '(空格)。
2 AA、BB、CC、DD、EE值的字符之间含有单独的'_'(下划线)时，忽略'_'(下划线)。
3 天、时、分、秒、毫秒的值为使用'.'(句号)的实数时，也可转换。
4 如果数据中含有天、时、分、秒、毫秒中的任意一项，则省略其它也可转换。
5 天、时、分、秒、毫秒的值前带有'0'时，也可转换。

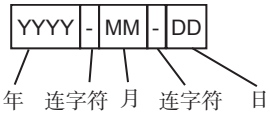
以下情况时无法转换。

- 非上述结构时
- 符号栏与天之间含有'_'(下划线)时
- '.'(句号)和'_'(下划线)连续时

例如，字符串型的数据为'–0.5d48h0.123456789ms'时，结构要素的值为T#–2d12h0m0s0.123456ms(T#–216000000.123456ms)。

● 日期型

将下列结构的数据转换为日期型的值。



以下情况例外。

- 年、月、日的值之前含有' '(空格)时，忽略' '(空格)。
- 年、月、日的值的字符之间含有单独的'_'(下划线)时，忽略'_'(下划线)。
- 年、月、日的值之前带有'0'时，也可转换。

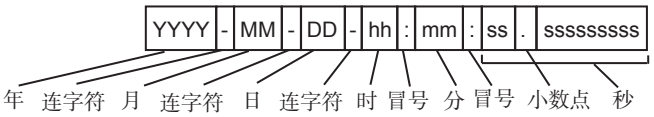
以下情况时无法转换。

- 非上述结构时
- 不存在的日期时

例如，字符串型的数据为'1970–1–01'时，结构要素的值为D#1970–01–01。

● 日期时刻型

将下列结构的数据转换为日期时刻型的值。



项目	内容
年、月、日	表示日期的年月日。
时	值的范围为0 ~ 23。
分	值的范围为0 ~ 59。
秒	值的范围为0 ~ 59.999999999。值为整数时，无需小数点。
连字符、冒号	始终需要。

(注) 1 年、月、日、时、分、秒的值之前含有' '(空格)时，忽略' '(空格)。
2 年、月、日、时、分、秒的值的字符之间含有单独的'_'(下划线)时，忽略'_'(下划线)。
3 年、月、日、时、分、秒的值前带有'0'时，也可转换。

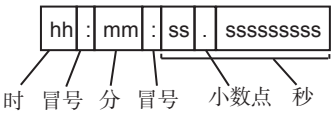
以下情况时无法转换。

- 非上述结构时
- 不存在的日期时

例如，字符串型的数据为'1970–01–23–4:56:07.89'时，结构要素的值为DT#1970–01–23–04:56:07.89。

● 时刻型

将下列结构的数据转换为时刻型的值。



项目	内容
时	值的范围为0 ~ 23。
分	值的范围为0 ~ 59。
秒	值的范围为0 ~ 59.999999999。值为整数时，无需小数点'.'(句号)。
冒号	始终需要。

(注) 1 时、分、秒的值之前含有' '(空格)时，忽略' '(空格)。
2 时、分、秒的值的字符之间含有单独的'_'(下划线)时，忽略'_'(下划线)。
3 时、分、秒的值之前带有'0'时，也可转换。

以下情况时无法转换。

- 非上述结构时
- '.'(句号)和'_'(下划线)连续时

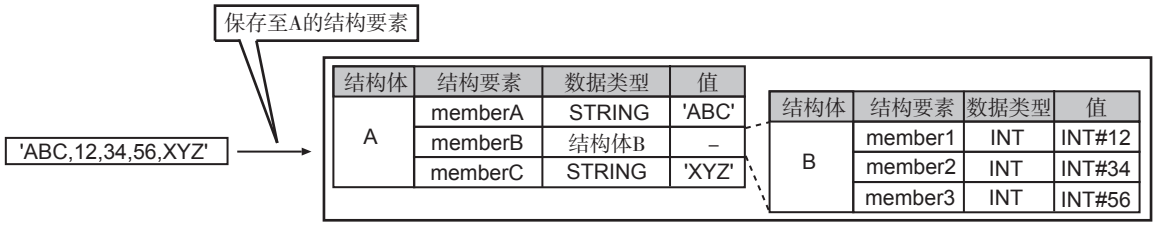
例如，字符串型的数据为'12:23:34.567'时，结构要素的值为TOD#12:23:34.567。

● 字符串型

数据结尾带有的NULL字符的值为结构要素的值。字符串超过结构要素的数据大小时，无法转换。
例如，字符串型的数据为结尾无NULL字符的'ABC'时，结构要素的值为结尾带NULL字符的'ABC'。

● 结构体

按照结构要素的数据类型的转换规则转换字符串型的数据。将转换的值从开头起依次保存为结构要素的值，直至结构要素为非结构体的数据类型的层。例如，结构体A的结构要素中含有结构体B时，转换如下所示。



● 枚举体

将表示 DINT 型变量的字符串型的数据转换为枚举体的枚举元素。以与整数型相同的规则转换为 DINT 型，将DINT型的值作为枚举体的值，转换为枚举体的值相应的枚举元素。

字符串型的数据不是表示DINT型数值的字符串时，无法转换。

例如，枚举体Color获取的枚举元素为red、yellow、green。与各枚举元素对应的值：red为1；yellow为2；green为3。数据为'3'时，结构要素的值为green。

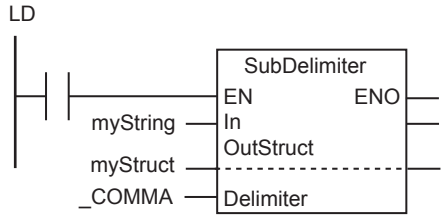
● 数组

将由分隔符隔开的数据转换为各元素的值。转换规则遵从数组的数据类型。仅结构要素为 1 维数组时可转换。

例如，假设结构要素中含有BYTE型的数组myString[0..3]。将字符串'AA，BB，CC，DD'中由逗号隔开的数据转换为各元素的值时，myString[0] 为 BYTE#16#AA，mySring[1] 为 BYTE#16#BB，myString[2] 为 BYTE#16#CC，myString[3]为BYTE#16#DD。

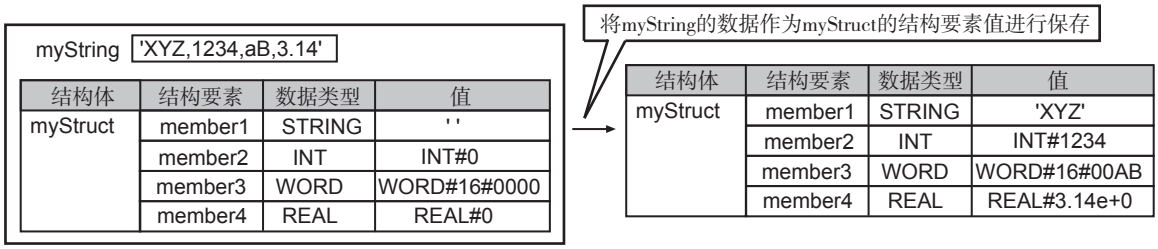
记述示例

读取字符串 “myString”中由逗号隔开的数据，保存为结构体 “myStruct”的结构要素的值时的示例如下所示。



ST

```
SubDelimiter(myString,myStruct,_COMMA);
```



参考

- 从SD存储卡内的指定CSV文件中读取值时，如果同时使用“FileGets指令(P.2-1330)”和本指令，则较为便捷。使用方法请参阅示例程序。
- 将由 “AddDelimiter指令(P.2-559)”转换的字符串恢复为结构体的数据时，使用本指令。

使用注意事项

- “In”中含有连续分隔符时，不存在由分隔符隔开的数据。数据不存在时，“OutStruct”的结构要素的值不确定。
- 请勿在 “In”中将分隔符用于分隔符以外的用途。即使用于分隔符以外的用途，本指令也识别为分隔符进行动作。
- “OutStruct”中含有字符串型的结构要素时，请勿使 “In”中相应数据的结尾带NULL字符。如果在“In”的结尾以外使用NULL文字，则仅可转换NULL字符之前的字符串。
- “OutStruct”中含有枚举体的结构要素时，请将 “In”的相应部分设为表示定义为枚举元素值的数据。即使枚举体变量的值不是定义为枚举元素的值，也不会发生异常。
- 以下情况时会发生异常。ENO为FALSE，“OutStruct”的值不确定。
  - 无法转换为 “OutStruct”的结构要素的数据类型的值时
  - 转换结果超过 “OutStruct”的结构要素的数据类型的值的有效范围时
  - “OutStruct”中含有2维以上的数组的结构要素时
  - “OutStruct”中含有联合体的结构要素时
  - “OutStruct”的大小大于8192byte时

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

文件名为'ABC.csv'的文件中已保存由换行代码CR进行换行的多行字符串。1行字符串由逗号隔开，表示多个数据。
从该文件中逐行读取字符串，将由逗号隔开的数据作为myStruct的数组变量myArray[ ]的结构要素的值从开头起依次保存。假设myStruct为结构要素中含有5个STRING型变量的结构体。
读取文件至最后(读取直至文件结尾EOF)时，结束处理。

处理步骤如下所示。

1 使用FileOpen指令，打开'ABC.csv'文件。

2 使用FileGets指令，读取1行文件内容。

3 使用SubDelimiter指令，将由逗号隔开的字符串保存为myArray[ ]的结构要素的值。

4 继续执行上述2～3，直至读取文件结尾EOF。

5 使用FileClose指令，关闭文件。

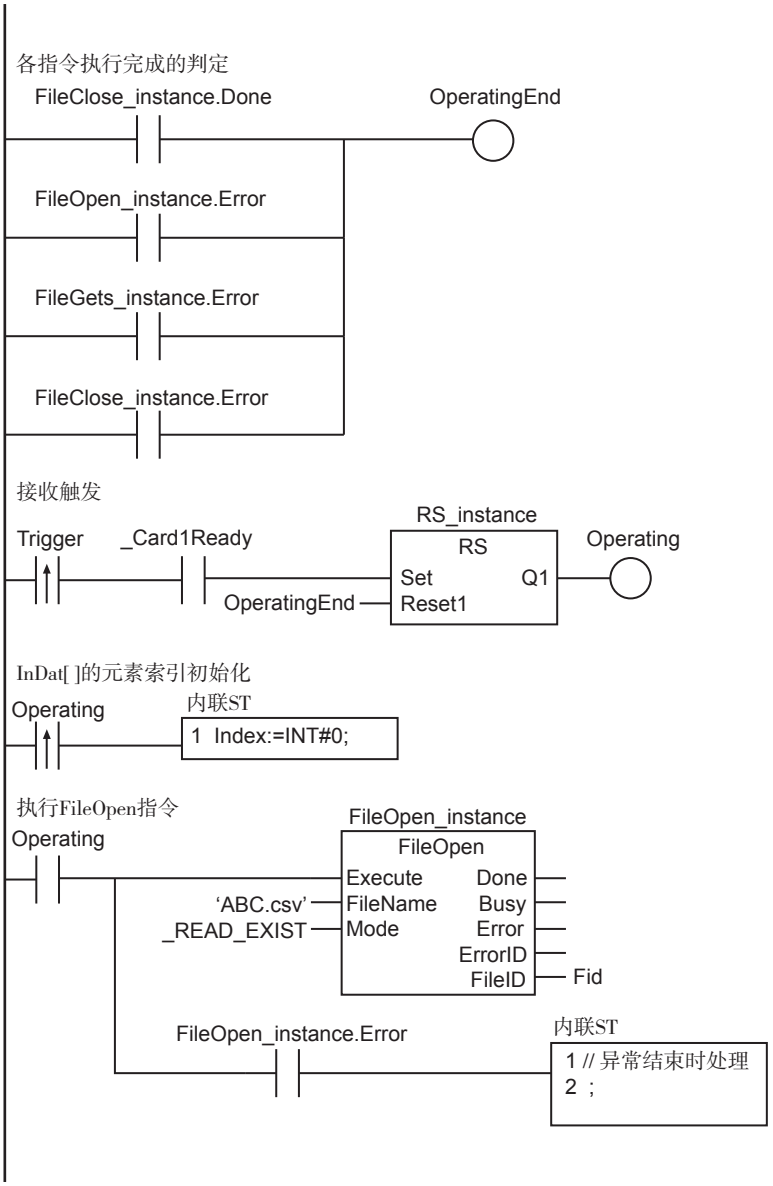
● 数据类型的定义

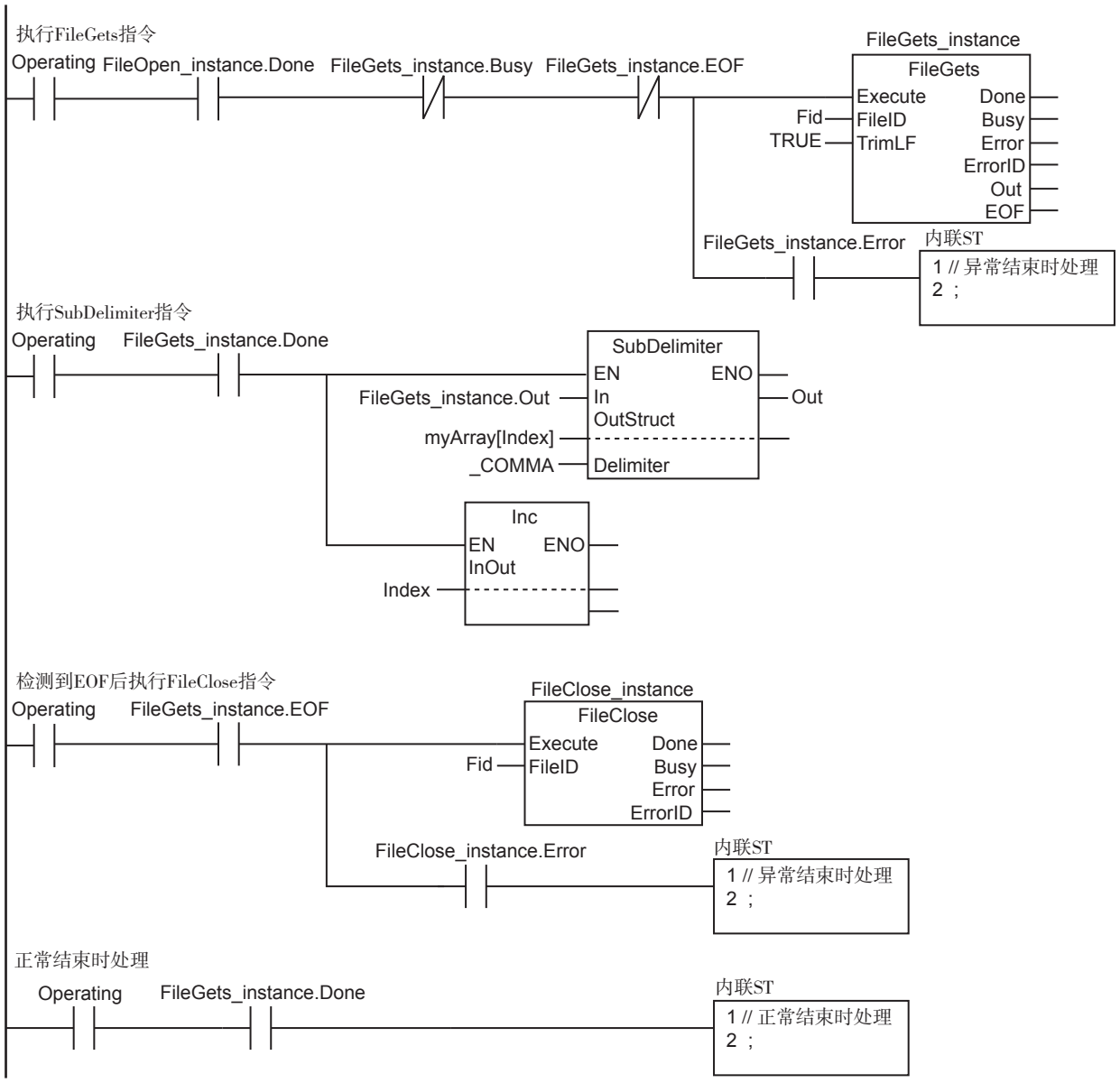
名称	数据类型	注释
myStruct	STRUCT	结构体
member0	STRING	结构要素
member1	STRING	结构要素
member2	STRING	结构要素
member3	STRING	结构要素
member4	STRING	结构要素

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	False	处理结束
	Trigger	BOOL	False	执行条件
	Operating	BOOL	False	处理中
	Index	INT	0	myArray[ ]的要素索引
	Fid	DWORD	16#0	文件ID
	myArray	ARRAY[0..999] OF myStruct	[1000((member0:='',member1:='',member2:='',member3:='',member4:=''))]	整数数据
	RS_instance	RS		
	FileOpen_instance	FileOpen		
	FileGets_instance	FileGets		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志




执行FileGets指令
Operating
FileOpen_instance.Done
FileGets_instance.Busy
FileGets_instance.EOF
FileGets_instance
FileGets
Execute
Done
Fid
FileID
Busy
TRUE
TrimLF
Error
ErrorID
Out
EOF
FileGets_instance.Error
内联ST
1 // 异常结束时处理
2 ;
执行SubDelimiter指令
Operating
FileGets_instance.Done
SubDelimiter
EN
ENO
FileGets_instance.Out
In
Out
OutStruct
myArray[Index]
_COMMA
Delimiter
Inc
EN
ENO
InOut
Index
检测到EOF后执行FileClose指令
Operating
FileGets_instance.EOF
FileClose_instance
FileClose
Execute
Done
Fid
FileID
Busy
Error
ErrorID
FileClose_instance.Error
内联ST
1 // 异常结束时处理
2 ;
正常结束时处理
Operating
FileGets_instance.Done
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	False	执行条件
	LastTrigger	BOOL	False	上个任务周期的Trigger的值
	OperatingStart	BOOL	False	处理开始
	Operating	BOOL	False	处理中
	myArray	ARRAY[0..999] OF myStruct	[1000((member0:='',member1:='',member2:='',member3:='',member4:=''))]	整数数据
	Stage	INT	0	状态变化
	Index	INT	0	myArray[ ]的要素索引
	Fid	DWORD	16#0	文件ID
	FileOpen_instance	FileOpen		
	FileGets_instance	FileGets		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

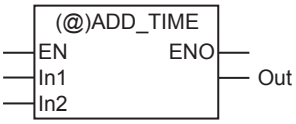
```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileOpen_instance(Execute:=FALSE);
    FileGets_instance(Execute:=FALSE);
    FileClose_instance(Execute:=FALSE);
    Stage          :=INT#1;
    Index          :=INT#0;
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                         // 文件打开
        FileOpen_instance(
            Execute  :=TRUE,
            FileName :='ABC.csv',          // 文件名
            Mode     :=_READ_EXIST,        // 文件读取
            FileID   =>Fid);               // 文件ID

        IF (FileOpen_instance.Done=TRUE) THEN
            Stage:=INT#2;                  // 正常结束
        END_IF;

        IF (FileOpen_instance.Error=TRUE) THEN
            Stage:=INT#99;                 // 异常结束
        END_IF;
```

```
    2 :                         // 字符串读取
        FileGets_instance(
            Execute :=TRUE,
            FileID  :=Fid,
            TrimLF :=TRUE);

        IF (FileGets_instance.Done=TRUE) THEN
            // 将读取的字符串保存为myArray[ ]的结构要素值
            SubDelimiter(FileGets_instance.Out,myArray[Index],_COMMA);
            Index:=Index+INT#1;

            // 到达文件结尾
            IF (FileGets_instance.EOF=TRUE) THEN
                Stage:=INT#3;           // 正常结束
            ELSE
                FileGets_instance(Execute:=FALSE);
            END_IF;
        END_IF;

        IF (FileGets_instance.Error=TRUE) THEN
            Stage:=INT#99;              // 异常结束
        END_IF;

    3 :                         // 文件关闭
        FileClose_instance(
            Execute :=TRUE,
            FileID  :=Fid);             // 文件ID

        IF (FileClose_instance.Done=TRUE) THEN
            Operating:=FALSE;           // 正常结束
        END_IF;

        IF (FileClose_instance.Error=TRUE) THEN
            Stage:=INT#99;              // 异常结束
        END_IF;

    99 :                        // 异常结束处理
        Operating:=FALSE;
    END_CASE;
END_IF;
```

时间/时刻指令

ADD_TIME

对时间和时间进行加法运算。

指令	名称	FB/FUN	图形表现	ST表现
ADD_TIME	时间加法	FUN		Out:=ADD_TIME(In1, In2);

变量

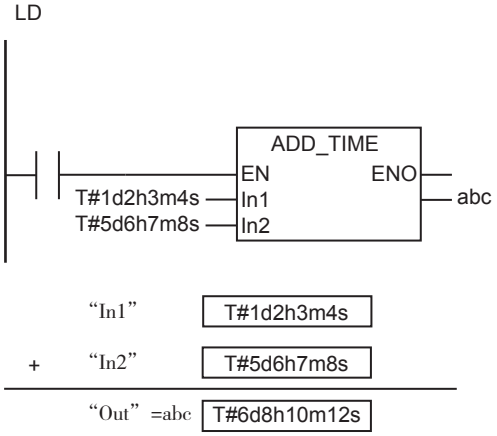
	名称	输入/输出	内容	有效范围	单位	初始值
In1	被加法时间	输入	被加法时间	遵从数据类型	ns	T#0s
In2	加法时间		加法时间			
Out	加法结果时间	输出	加法结果时间	遵从数据类型	ns	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																○				
In2																○				
Out																○				

功能

对时间 “In1”和 “In2”进行加法运算。加法运算结果 “Out”也是时间。

“In1”=T#1d2h3m4s、“In2”=T#5d6h7m8s时的示例如下所示。



使用注意事项

加法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。

- T#106751d_23h_47m_16s_854.775807ms + T#0.000001ms
  → T#–106751d_23h_47m_16s_854.775808ms
- T#–106751d_23h_47m_16s_854.775808ms + T#–0.000001ms
  → T#106751d_23h_47m_16s_854.775807ms

ADD_TOD_TIME

将时刻加上时间。

指令	名称	FB/FUN	图形表现	ST表现
ADD_TOD_TIME	时刻和时间的加法	FUN	(@)ADD_TOD_TIME EN ENO In1 Out In2	Out:=ADD_TOD_TIME(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被加法时刻	输入	被加法时刻	遵从数据类型	时分秒	TOD#0:0:0
In2	加法时间		加法时间		ns	T#0s
Out	加法结果时刻	输出	加法结果时刻	遵从数据类型	时分秒	–

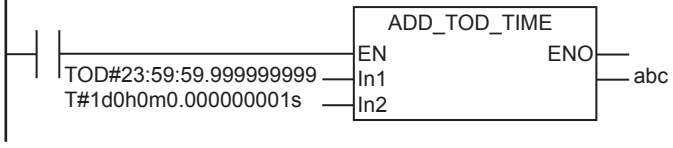
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																		○		
In2																○				
Out																		○		

功能

将时刻 “In1”加上时间 “In2”。加法运算结果 “Out”为时刻。

“In1”=TOD#23:59:59.999999999、“In2”=T#1d0h0m0.000000001s时的示例如下所示。

LD



ST

```
abc:=ADD_TOD_TIME(TOD#23:59:59.999999999,
                  T#1d0h0m0.000000001s);
```

	“In1”	TOD#23:59:59.999999999
+	“In2”	T#1d0h0m0.000000001s
	“Out”=abc	TOD#0:0:0.000000000

使用注意事项

加法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。

- TOD#23:59:59.999999999 + T#0.000001ms → TOD#0:0:0.000000000
- TOD#0:0:0.000000000 + T#−0.000001ms → TOD#23:59:59.999999999

ADD_DT_TIME

将日期时刻加上时间。

指令	名称	FB/FUN	图形表现	ST表现
ADD_DT_TIME	日期时刻和时间的加法	FUN	(@)ADD_DT_TIME EN ENO In1 Out In2	Out:=ADD_DT_TIME(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被加法日期时刻	输入	被加法日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
In2	加法时间		加法时间		ns	T#0s
Out	加法结果日期时刻	输出	加法结果日期时刻	遵从数据类型	年月日时分秒	–

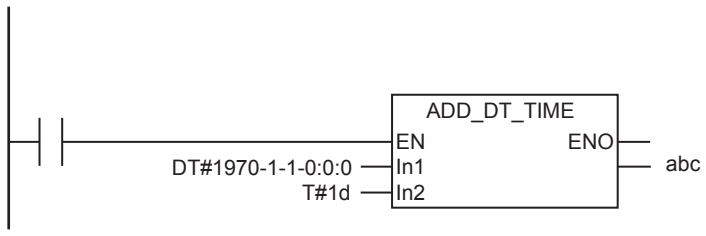
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																			○	
In2																○				
Out																			○	

功能

将日期时刻 “In1” 加上时间 “In2”。加法运算结果 “Out” 为日期时刻。闰年也加进去。

“In1” =DT#1970-1-1-0:0:0、“In2” =T#1d时的示例如下所示。

LD



ST

```
abc:=ADD_DT_TIME(DT#1970-1-1-0:0:0, T#1d);
```

	“In1”	DT#1970-1-1-0:0:0
+	“In2”	T#1d
	“Out” =abc	DT#1970-1-2-0:0:0

相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

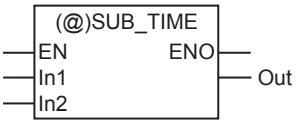
使用注意事项

加法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。

- DT#2554–7–21–23:34:33.709551615 + T#0.000001ms → DT#1970–1–1–0:0:0
- DT#1970–1–1–0:0:0 + T#–0.000001ms → DT#2554–7–21–23:34:33.709551615

SUB_TIME

将时间减去时间。

指令	名称	FB/FUN	图形表现	ST表现
SUB_TIME	时间减法	FUN		Out:=SUB_TIME(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减法时间	输入	被减法时间	遵从数据类型	ns	T#0s
In2	减法时间		减法时间			
Out	减法结果时间	输出	减法结果时间	遵从数据类型	ns	–

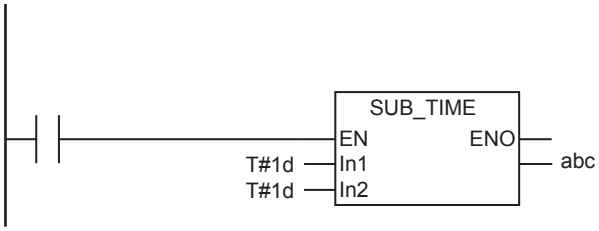
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																○				
In2																○				
Out																○				

功能

将时间 “In1”减去时间 “In2”。减法运算结果 “Out”也是时间。

“In1” = “In2” =T#1d时的示例如下所示。

LD



ST

```
abc:=SUB_TIME(T#1d, T#1d);
```

“In1” T#1d

\- “In2” T#1d

“Out” =abc T#0s

使用注意事项

减法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。

- T#106751d_23h_47m_16s_854.775807ms – T#–0.000001ms
  → T#–106751d_23h_47m_16s_854.775808ms
- T#–106751d_23h_47m_16s_854.775808ms – T#0.000001ms
  → T#106751d_23h_47m_16s_854.775807ms

SUB_TOD_TIME

将时刻减去时间。

指令	名称	FB/FUN	图形表现	ST表现
SUB_TOD_TIME	时刻和时间的减法	FUN	(@)SUB_TOD_TIME EN ENO In1 Out In2	Out:=SUB_TOD_TIME(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减法时刻	输入	被减法时刻	遵从数据类型	时分秒	TOD#0:0:0
In2	减法时间		减法时间		ns	T#0s
Out	减法结果时刻	输出	减法结果时刻	遵从数据类型	时分秒	–

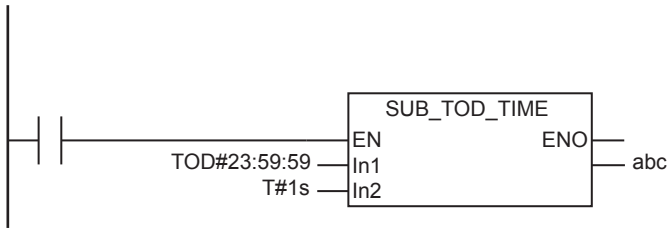
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																		○		
In2																○				
Out																		○		

功能

将时刻 “In1”减去时间 “In2”。减法运算结果 “Out”为时刻。

“In1”=TOD#23:59:59、“In2”=T#1s时的示例如下所示。

LD



ST

```
abc:=SUB_TOD_TIME(TOD#23:59:59, T#1s);
```

	“In1”	TOD#23:59:59
-	“In2”	T#1s
	“Out” =abc	TOD#23:59:58

使用注意事项

减法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。

- TOD#23:59:59.999999999 – T#–0.000001ms → TOD#0:0:0
- TOD#0:0:0 – T#0.000001ms → TOD#23:59:59.999999999

SUB_TOD_TOD

将时刻减去时刻。

指令	名称	FB/FUN	图形表现	ST表现
SUB_TOD_TOD	时刻减法	FUN	(@)SUB_TOD_TOD EN ENO In1 Out In2	Out:=SUB_TOD_TOD(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减法时刻	输入	被减法时刻	遵从数据类型	时分秒	TOD#0:0:0
In2	减法时刻		减法时刻			
Out	减法结果时间	输出	减法结果时间	遵从数据类型	ns	–

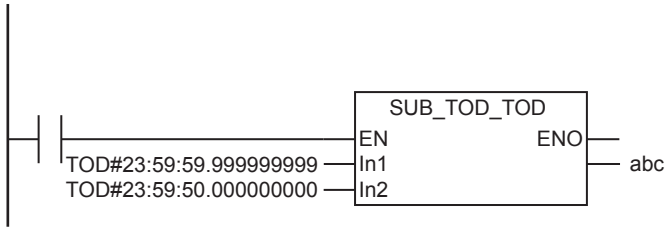
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																		○		
In2																		○		
Out																○				

功能

将时刻 “In1”减去时刻 “In2”。减法运算结果 “Out”为时间。

“In1”=TOD#23:59:59.999999999、“In2”=TOD#23:59:50.000000000时的示例如下所示。

LD



ST

```
abc:=SUB_TOD_TOD(TOD#23:59:59.999999999,
                 TOD#23:59:50.000000000);
```

	“In1”	TOD#23:59:59.999999999
-	“In2”	TOD#23:59:50.000000000
	“Out” =abc	T#0d0h0m9.999999999s

SUB_DATE_DATE

将日期减去日期。

指令	名称	FB/FUN	图形表现	ST表现
SUB_DATE_DATE	日期减法	FUN	(@)SUB_DATE_DATE EN ENO In1 Out In2	Out:=SUB_DATE_DATE(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减法日期	输入	被减法日期	遵从数据类型	年月日	D#1970-1-1
In2	减法日期		减法日期			
Out	减法结果时间	输出	减法结果时间	遵从数据类型	ns	–

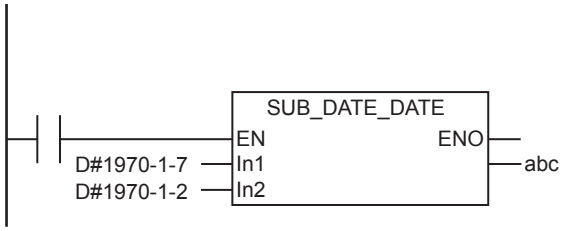
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																	○			
In2																	○			
Out																○				

功能

将日期 “In1”减去日期 “In2”。减法运算结果 “Out”为时间。

“In1”=D#1970–1–7、“In2”=D#1970–1–2时的示例如下所示。

LD



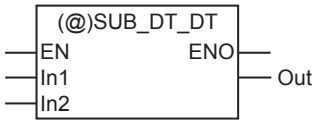
ST

```
abc:=SUB_DATE_DATE(D#1970-1-7, D#1970-1-2);
```

	“In1”	D#1970-1-7
-	“In2”	D#1970-1-2
	“Out”=abc	T#5d0h0m0.000000000s

SUB_DT_DT

将日期时刻减去日期时刻。

指令	名称	FB/FUN	图形表现	ST表现
SUB_DT_DT	日期时刻减法	FUN		Out:=SUB_DT_DT(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减法日期时刻	输入	被减法日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
In2	减法日期时刻		减法日期时刻			
Out	减法结果时间	输出	减法结果时间	遵从数据类型	ns	–

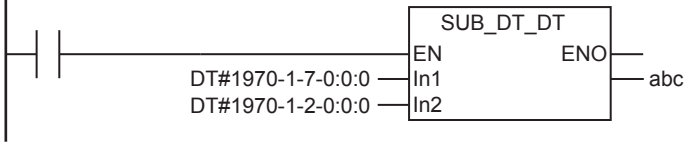
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																			○	
In2																			○	
Out																○				

功能

将日期时刻 “In1”减去日期时刻 “In2”。减法运算结果 “Out”为时间。

“In1”=DT#1970-1-7-0:0:0、“In2”=DT#1970-1-2-0:0:0时的示例如下所示。

LD



ST

```
abc:=SUB_DT_DT(DT#1970-1-7-0:0:0,
               DT#1970-1-2-0:0:0);
```

	“In1”	DT#1970-1-7-0:0:0
-	“In2”	DT#1970-1-2-0:0:0
	“Out” =abc	T#5d

相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

使用注意事项

运算结果超过 “Out”的有效范围时，“Out”的值为错误值。

SUB_DT_TIME

将日期时刻减去时间。

指令	名称	FB/FUN	图形表现	ST表现
SUB_DT_TIME	日期时刻和时间的减法	FUN	(@)SUB_DT_TIME EN ENO In1 Out In2	Out:=SUB_DT_TIME(In1, In2);

变量

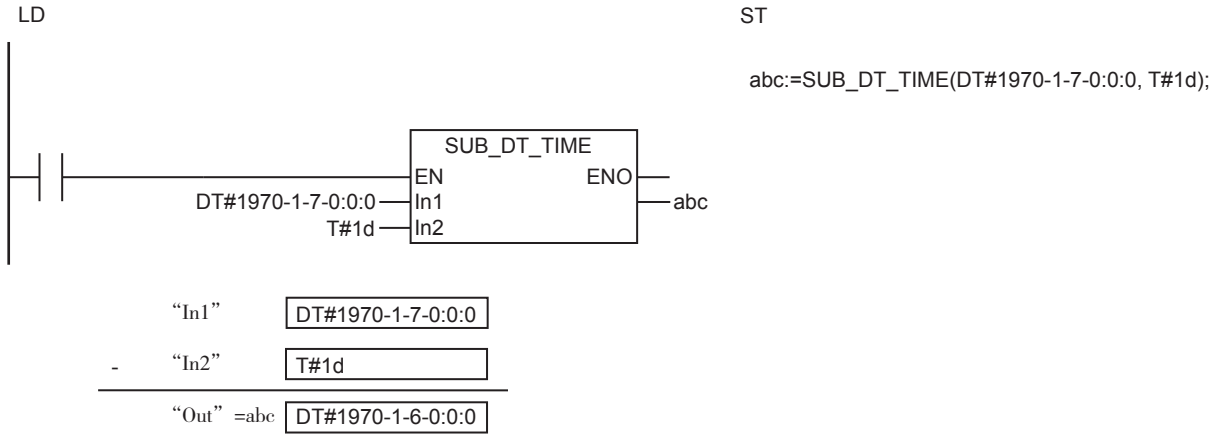
	名称	输入/输出	内容	有效范围	单位	初始值
In1	被减法日期时刻	输入	被减法日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
In2	减法时间		减法时间		ns	T#0s
Out	减法结果日期时刻	输出	减法结果日期时刻	遵从数据类型	年月日时分秒	-

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																			○	
In2																○				
Out																			○	

功能

将日期时刻 “In1”减去时间 “In2”。减法运算结果 “Out”为日期时刻。闰年也加进去。

“In1”=DT#1970-1-7-0:0:0、“In2”=T#1d时的示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

使用注意事项

减法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。

- DT#2554–7–21–23:34:33.709551615 – T#–0.000001ms → DT#1970–1–1–0:0:0
- DT#1970–1–1–0:0:0 – T#0.000001ms → DT#2554–7–21–23:34:33.709551615

MULTIME

按指定乘数对时间进行乘法运算。

指令	名称	FB/FUN	图形表现	ST表现
MULTIME	时间乘法	FUN	(@)MULTIME EN ENO In1 Out In2	Out:=MULTIME(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被乘法时间	输入	被乘法时间	遵从数据类型	ns	T#0s
In2	乘数		乘数		–	(*)
Out	乘法结果时间	输出	乘法结果时间	遵从数据类型	ns	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

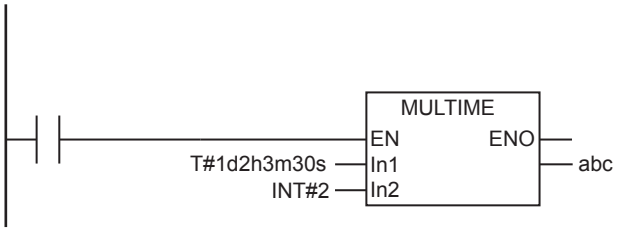
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																○				
In2						○	○	○	○	○	○	○	○	○	○					
Out																○				

功能

对时间 “In1”和乘数 “In2”进行乘法运算。乘法运算结果 “Out”为时间。

“In1”=T#1d2h3m30s、“In2”=INT#2时的示例如下所示。

LD



ST

```
abc:=MULTIME(T#1d2h3m30s, INT#2);
```

“In1” T#1d2h3m30s

× “In2” INT#2

“Out” =abc T#2d4h7m

使用注意事项

- “In2”为实数时，乘法运算结果小于1ns时，采用五舍五入。五舍五入是指如下表所示的处理。

低于1ns的数值	处理	例
小于0.5	舍去	1.49→1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2
大于0.5	进位	1.51→2

- “In2”的值为0、+∞、−∞、非数时，“Out”的值如下所示。

“In2”的值	“Out”的值	
	除右侧以外	NX1P2
0	T#0s	T#0s
+∞	T#−106751d23h47m16.854775808s	T#−0d0h0m0s1e−6ms
−∞	T#−106751d23h47m16.854775808s	T#−0d0h0m0s1e−6ms
非数	T#−106751d23h47m16.854775808s	T#0s

- 乘法运算结果超出 “Out”的有效范围时，不会发生异常，执行如下动作。
  - T#53375d_23h_53m_38s_427.387904ms * USINT#2
    → T#−106751d_23h_47m_16s_854.775808ms
  - T#−53375d_23h_53m_38s_427.387905ms * USINT#2
    → T#106751d_23h_47m_16s_854.775806ms

DIVTIME

按指定除数对时间进行除法运算。

指令	名称	FB/FUN	图形表现	ST表现
DIVTIME	时间除法	FUN	(@)DIVTIME EN ENO In1 Out In2	Out:=DIVTIME(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	被除法时间	输入	被除法时间	遵从数据类型	ns	T#0s
In2	除数		除数		–	(*)
Out	除法结果时间	输出	除法结果时间	遵从数据类型	ns	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

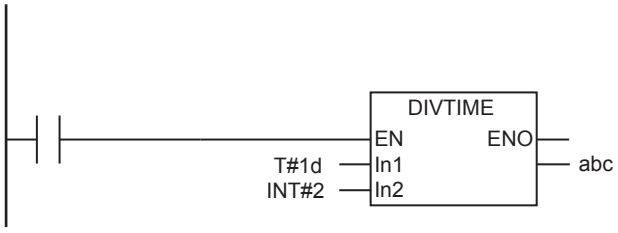
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																○				
In2						○	○	○	○	○	○	○	○	○	○					
Out																○				

功能

将时间 “In1”除以除数 “In2”。除法运算结果 “Out”为时间。

“In1”=T#1d、“In2”=INT#2时的示例如下所示。

LD



ST

```
abc:=DIVTIME(T#1d, INT#2);
```

	“In1”	T#1d
/	“In2”	INT#2
	“Out” =abc	T#12h

使用注意事项

- “In2”的值为0、+∞、−∞、非数时，“Out”的值如下所示。

“In2”的值	“Out”的值	
	除右侧以外	NX1P2
0.0	T#−106751d23h47m16.854775808s	T#0d_0h_0m_0s_1e−006
+∞	T#0s	T#0s
−∞	T#0s	T#0s
非数	T#−106751d23h47m16.854775808s	T#0s

- “In2”为实数时，可能产生数ns的误差。
- “In2”为实数时，除法运算结果小于1ns时，采用五舍五入。五舍五入是指如下表所示的处理。

低于1ns的数值	处理	例
小于0.5	舍去	1.49→1
0.5	个位的值为偶数时舍去，为奇数时进位	1.50→2 2.50→2
大于0.5	进位	1.51→2

- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In2”为整数型、值为0时。

CONCAT_DATE_TOD

组合日期和时刻。

指令	名称	FB/FUN	图形表现	ST表现
CONCAT_DATE_TOD	日期和时刻的组合	FUN	(@)CONCAT_DATE_TOD EN ENO In1 Out In2	Out:=CONCAT_DATE_TOD(In1, In2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In1	日期	输入	日期	遵从数据类型	年月日	D#1970-1-1
In2	时刻		时刻		时分秒	TOD#0:0:0
Out	组合结果日期时刻	输出	组合结果日期时刻	遵从数据类型	年月日时分秒	–

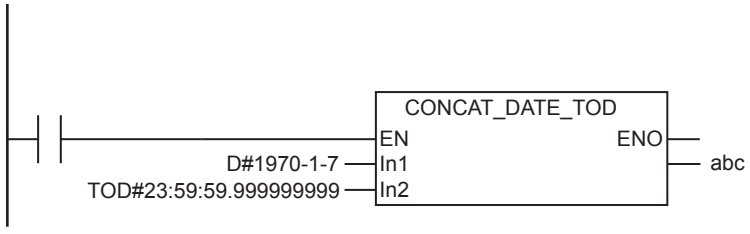
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In1																	○			
In2																		○		
Out																			○	

功能

组合日期 “In1”和时刻 “In2”。组合结果 “Out”为日期时刻。

“In1”=D#1970-1-7、“In2”=TOD#23:59:59.999999999时的示例如下所示。

LD



ST

```
abc:=CONCAT_DATE_TOD(D#1970-1-7,
                     TOD#23:59:59.999999999);
```

	“In1”	D#1970-1-7
+	“In2”	TOD#23:59:59.999999999
	“Out”=abc	DT#1970-1-7-23:59:59.999999999

相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- 组合结果超过 “Out”的有效范围(“In1”的值为D#2554–7–21、“In2”的值大于TOD#23:34:33.709551615)时。

DT_TO_TOD

从日期时刻中截取时刻。

指令	名称	FB/FUN	图形表现	ST表现
DT_TO_TOD	从日期时刻中截取时刻	FUN	(@)DT_TO_TOD EN ENO In Out	Out:=DT_TO_TOD(In);

变量

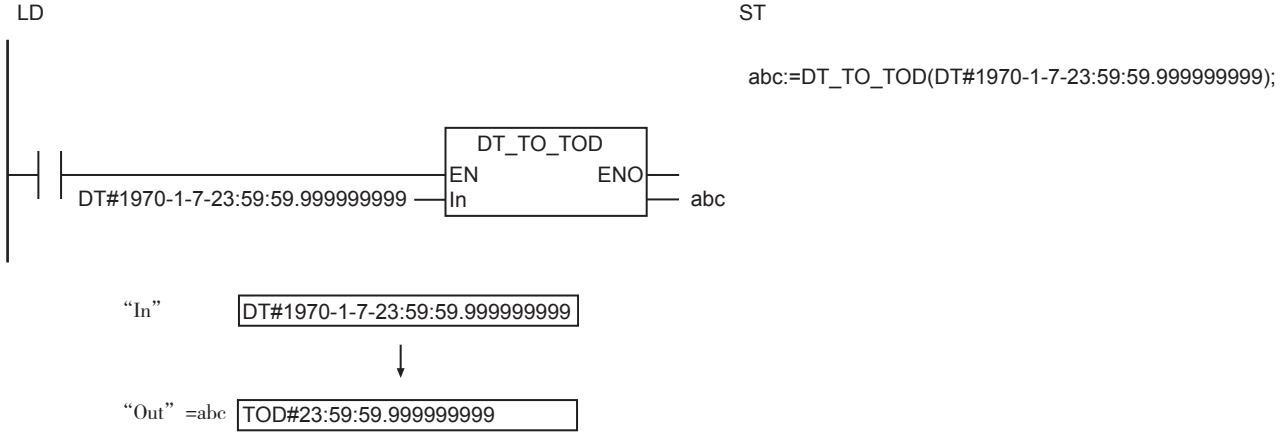
	名称	输入/输出	内容	有效范围	单位	初始值
In	日期时刻	输入	日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
Out	时刻	输出	时刻	遵从数据类型	时分秒	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Out																		○		

功能

从日期时刻 “In”中仅截取时刻部分。

“In”=DT#1970-1-7-23:59:59.999999999时的示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

DT_TO_DATE

从日期时刻中截取日期。

指令	名称	FB/FUN	图形表现	ST表现
DT_TO_DATE	从日期时刻中截取日期	FUN	(@)DT_TO_DATE EN ENO In Out	Out:=DT_TO_DATE(In);

变量

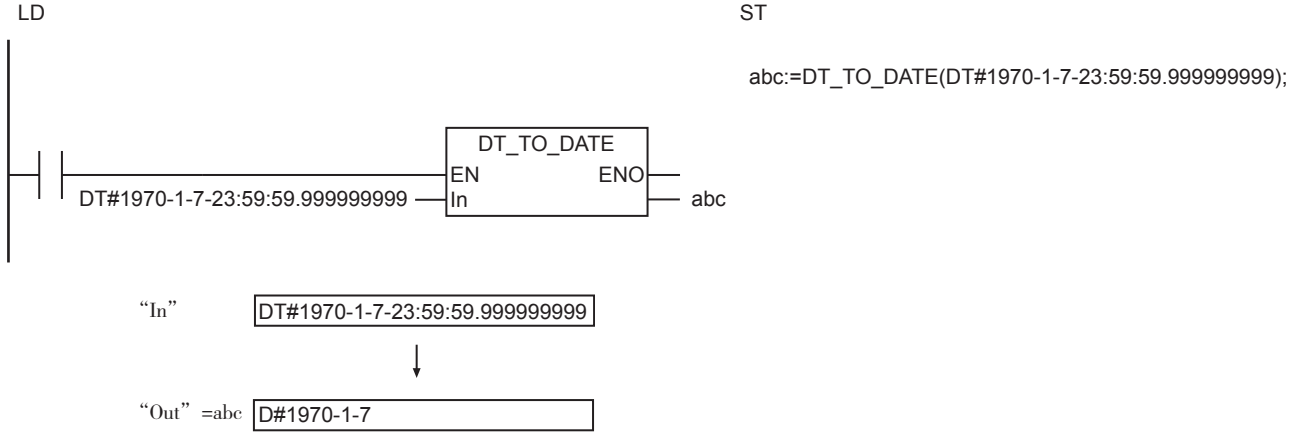
	名称	输入/输出	内容	有效范围	单位	初始值
In	日期时刻	输入	日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
Out	日期	输出	日期	遵从数据类型	年月日	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Out																	○			

功能

从日期时刻 “In”中仅截取日期部分。

“In”=DT#1970-1-7-23:59:59.999999999时的示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

SetTime

补偿系统时刻。

指令	名称	FB/FUN	图形表现	ST表现
SetTime	时钟补偿	FUN	(@)SetTime EN ENO In Out	SetTime(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	补偿时刻数据	输入	用于补偿系统时刻的当前日期时刻	*1	年月日时分秒	DT#1970-1-1-0:0:0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

*1 有效范围为GMT(格林尼治标准时间)。
NX系列CPU单元的有效范围为DT#1970-01-01-00:00:00.000000000 ~ DT#2069-12-31-23:59:59.999999999(1970年1月1日0时0分0.000000000秒 ~ 2069年12 月31日23时59分59.999999999秒)。
NJ系列CPU单元的有效范围为DT#1970-01-01-00:00:00.000000000 ~ DT#2106-02-06-23:59:59.999999999(1970年1月1日0时0分0.000000000秒 ~ 2106年2月6 日23时59分59.999999999秒)。

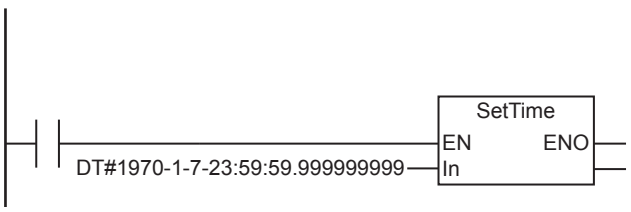
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Out	○																			

功能

将日期时刻 “In”的值设置为系统时刻。

“In”=DT#1970-1-7-23:59:59.999999999时的示例如下所示。

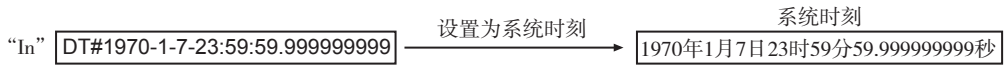
LD



ST

```
SetTime(DT#1970-1-7-23:59:59.999999999);
```

将“In”的值设置为系统时刻。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

参考

除了本指令外，系统时刻的补偿方法还有如下两种。

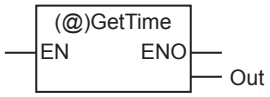
- 使用Sysmac Studio的方法
- 使用NTP功能的方法

使用注意事项

- 请将 “In”指定为已设定时间区的标准时间，而非GMT(格林尼治标准时间)。
- 不能将 “In”设定为低于GMT的1970–1–1–0:0:0.000000000的时刻。
- 更新内置时钟时，会发生时滞现象。因此，如果在执行本指令后立即获取时刻，可获取更新前的时刻。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值超过有效范围时。
  - “In”的值低于GMT的1970–1–1–0:0:0.000000000时。

GetTime

读取当前时刻。

指令	名称	FB/FUN	图形表现	ST表现
GetTime	时刻获取	FUN		Out:=GetTime();

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	当前时刻	输出	当前时刻	*1	年月日时分秒	–

*1 有效范围为GMT(格林尼治标准时间)。
NX系列CPU单元的有效范围为DT#1970–01–01–00:00:00.000000000 ~ DT#2069–12–31–23:59:59.999999999(1970年1月1日0时0分0.000000000秒 ~ 2069年12 月31日23时59分59.999999999秒)。
NJ系列CPU单元的有效范围为DT#1970–01–01–00:00:00.000000000 ~ DT#2106–02–06–23:59:59.999999999(1970年1月1日0时0分0.000000000秒 ~ 2106年2月6 日23时59分59.999999999秒)。
NY系列控制器的有效范围为DT#2000–01–01–00:00:00.000000000 ~ DT#2099–12–31–23:59:59.999999999(2000年1月1日0时0分0.000000000秒 ~ 2099年12 月31日23时59分59.999999999秒)。

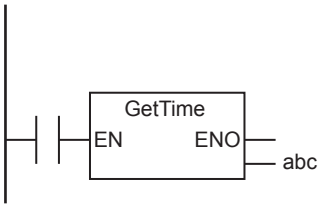
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out																			○	

功能

读取当前时刻。
当前时刻为已设定时间区的标准时间，而非GMT(格林尼治标准时间)。

示例如下所示。向变量abc中代入当前时刻。

LD



ST

```
abc:=GetTime();
```

读取当前时刻，代入abc。

相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

参考

- 将当前时刻转换为系统时刻(1970年1月1日0时0分0秒起的秒数)时，请使用 “DtToSec指令(P.2–613)”。
- 将当前时刻分解为年月日时分秒时，请使用 “DtToDateStruct指令(P.2–638)”。
- 欲读取星期时，请使用 “GetDayOfWeek指令(P.2–634)”。

DtToSec

将日期时刻转换为1970年1月1日0时0分0秒起的秒数。

指令	名称	FB/FUN	图形表现	ST表现
DtToSec	日期时刻→秒转换	FUN	(@)DtToSec EN ENO In Out	Out:=DtToSec(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	日期时刻	输入	日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
Out	秒	输出	1970年1月1日0时0分0秒起的秒数	0 ~ 18446744073	秒	-

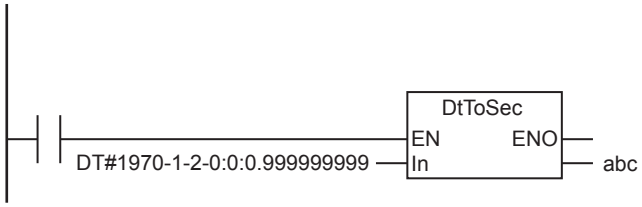
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Out													○							

功能

将日期时刻“In”转换为1970年1月1日0时0分0秒起的秒数。转换后的数值单位为秒。舍去小于1秒的值。

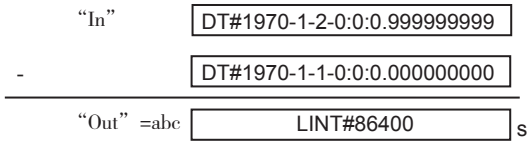
“In”=DT#1970-1-2-0:0:0.999999999时的示例如下所示。

LD



ST

```
abc:=DtToSec(DT#1970-1-2-0:0:0.999999999);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

参考

欲将1970年1月1日0时0分0秒起的秒数转换为日期时刻时，请使用 “SecToDt指令(P.2–618)”。

DateToSec

将日期转换为1970年1月1日0时0分0秒起的秒数。

指令	名称	FB/FUN	图形表现	ST表现
DateToSec	日期→秒转换	FUN	(@)DateToSec EN ENO In Out	Out:=DateToSec(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	日期	输入	日期	遵从数据类型	年月日	D#1970-1-1
Out	秒	输出	1970年1月1日0时0分0秒起的秒数	0 ~ 18446659200	秒	-

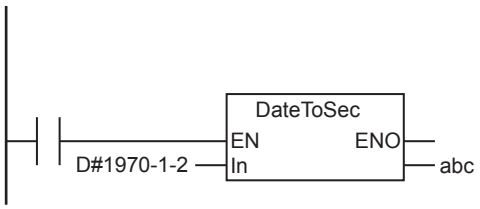
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																	○			
Out													○							

功能

将日期 “In”的0时0分0秒的时刻转换为1970年1月1日0时0分0秒起的秒数。转换后的数值单位为秒。

“In”=D#1970–1–2时的示例如下所示。

LD



ST

```
abc:=DateToSec(D#1970-1-2);
```

“In”	D#1970-1-2	
-	DT#1970-1-1-0:0:0.000000000	
“Out”=abc	LINT#86400	s

参考

欲将1970年1月1日0时0分0秒起的秒数转换为日期时，请使用 “SecToDate指令(P.2–620)”。

TodToSec

将时刻转换为0时0分0秒起的秒数。

指令	名称	FB/FUN	图形表现	ST表现
TodToSec	时刻→秒转换	FUN	(@)TodToSec EN ENO In Out	Out:=TodToSec(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	时刻	输入	时刻	遵从数据类型	时分秒	TOD#0:0:0
Out	秒	输出	0时0分0秒起的秒数	0 ~ 86399	秒	–

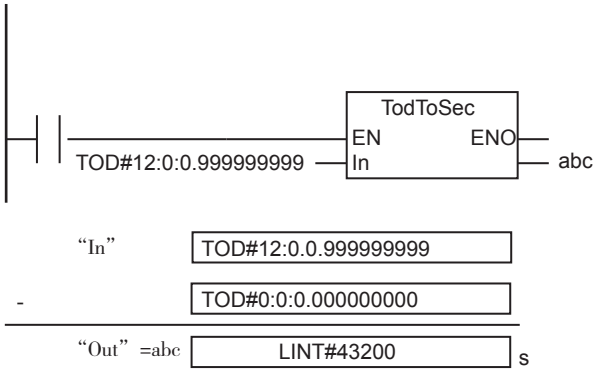
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																		○		
Out													○							

功能

将时刻 “In”转换为0时0分0秒起的秒数。转换后的数值单位为秒。舍去小于1秒的值。

“In”=TOD#12:0:0.999999999时的示例如下所示。

LD



ST

```
abc:=TodToSec(TOD#12:0:0.999999999);
```

参考

欲将0时0分0秒起的秒数转换为时刻时，请使用 “SecToTod指令(P.2–622)”。

SecToDt

将1970年1月1日0时0分0秒起的秒数转换为日期时刻。

指令	名称	FB/FUN	图形表现	ST表现
SecToDt	秒→日期时刻转换	FUN	(@)SecToDt EN ENO In Out	Out:=SecToDt(In);

变量

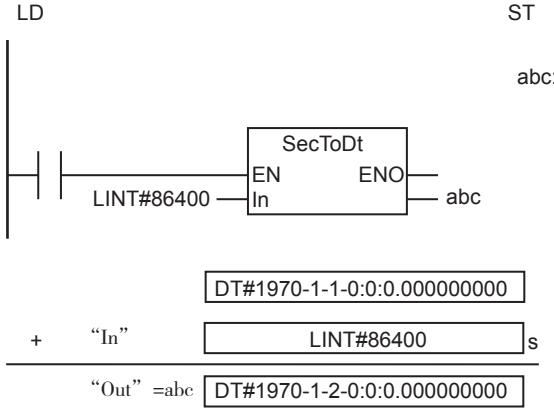
	名称	输入/输出	内容	有效范围	单位	初始值
In	秒	输入	1970年1月1日0时0分0秒起的秒数	0 ~ 18446744073	秒	0
Out	日期时刻	输出	日期时刻	遵从数据类型	年月日时分秒	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In													○							
Out																			○	

功能

将1970年1月1日0时0分0秒起的秒数 “In”转换为日期时刻。

“In”=LINT#86400时的示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

参考

欲将日期时刻转换为1970年1月1日0时0分0秒起的秒数时，请使用 “DtToSec指令(P.2-613)”。

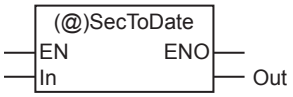
使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “In”的值超过有效范围时。

SecToDate

将1970年1月1日0时0分0秒起的秒数转换为日期。

指令	名称	FB/FUN	图形表现	ST表现
SecToDate	秒→日期转换	FUN		Out:=SecToDate(In);

变量

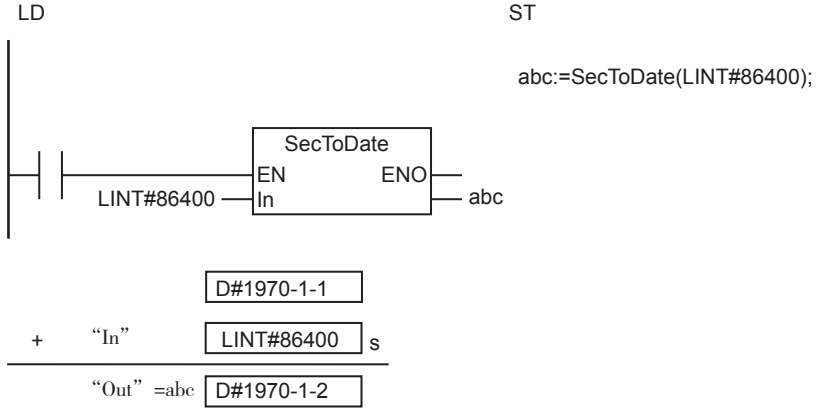
	名称	输入/输出	内容	有效范围	单位	初始值
In	秒	输入	1970年1月1日0时0分0秒起的秒数	0 ~ 18446744073	秒	0
Out	日期	输出	日期	遵从数据类型	年月日	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In													○							
Out																	○			

功能

将1970年1月1日0时0分0秒起的秒数 “In”转换为日期。舍去小于1日的值。

“In”=LINT#86400时的示例如下所示。



参考

欲将日期转换为1970年1月1日0时0分0秒起的秒数时，请使用 “DateToSec指令(P.2–615)”。

使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “In”的值超过有效范围时。

SecToTod

将0时0分0秒起的秒数转换为时刻。

指令	名称	FB/FUN	图形表现	ST表现
SecToTod	秒→时刻转换	FUN	(@)SecToTod EN ENO In Out	Out:=SecToTod(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	秒	输入	0时0分0秒起的秒数	遵从数据类型(*)	秒	0
Out	时刻	输出	时刻	遵从数据类型	时分秒	–

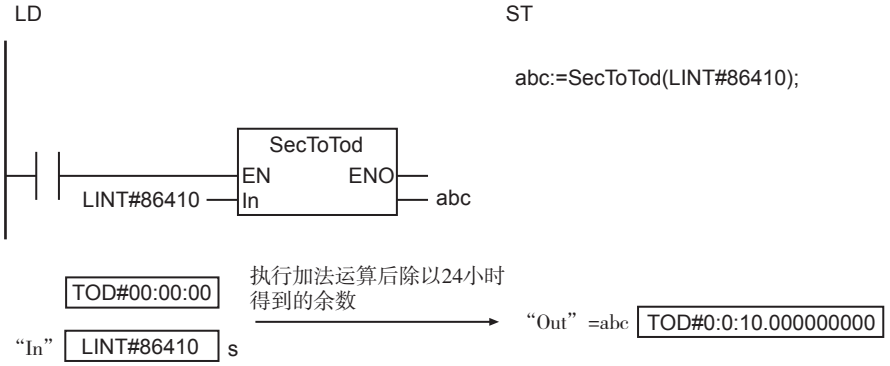
* 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In													○							
Out																		○		

功能

将0时0分0秒起的秒数 “In”转换为时刻。“In”的值为24小时以上时，将 “In”除以24小时得到的余数转换为时刻。

“In”=LINT#86410时的示例如下所示。



参考

欲将时刻转换为0时0分0秒起的秒数时，请使用 “TodToSec指令(P.2–617)”。

使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “In”的值超过有效范围时。

TimeToNanoSec

将时间转换为纳秒数。

指令	名称	FB/FUN	图形表现	ST表现
TimeToNanoSec	时间→纳秒转换	FUN	(@)TimeToNanoSec EN ENO In Out	Out:=TimeToNanoSec(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	时间	输入	时间	遵从数据类型	ns	T#0s
Out	秒数	输出	秒数	(*)	ns	–

* –9223372036854775808 ~ 9223372036854775807

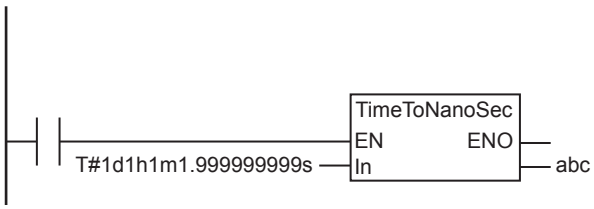
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																○				
Out													○							

功能

将时间 “In”转换为纳秒数。

“In”=T#1d1h1m1.999999999s时的示例如下所示。

LD



ST

```
abc:=TimeToNanoSec(T#1d1h1m1.999999999s);
```

纳秒数

“In” T#1d1h1m1.999999999s → “Out”=abc LINT#90061000000000 ns

参考

欲将纳秒转换为时间时，请使用 “NanoSecToTime指令(P.2–626)”。

TimeToSec

将时间转换为秒数。

指令	名称	FB/FUN	图形表现	ST表现
TimeToSec	时间→秒转换	FUN	(@)TimeToSec EN ENO In Out	Out:=TimeToSec(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	时间	输入	时间	遵从数据类型	ns	T#0s
Out	秒数	输出	秒数	−9223372036 ~ 9223372036	秒	−

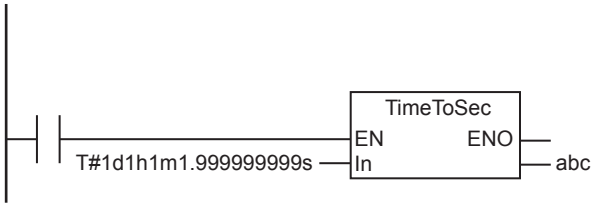
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																○				
Out													○							

功能

将时间 “In”转换为秒数。舍去小于1秒的值。

“In”=T#1d1h1m1.999999999s时的示例如下所示。

LD



ST

```
abc:=TimeToSec(T#1d1h1m1.999999999s);
```

参考

欲将秒转换为时间时，请使用 “SecToTime指令(P.2–627)”。

使用注意事项

“In”的单位为纳秒，“Out”的单位为秒。

NanoSecToTime

将纳秒数转换为时间。

指令	名称	FB/FUN	图形表现	ST表现
NanoSecToTime	纳秒→时间转换	FUN	(@)NanoSecToTime EN ENO In Out	Out:=NanoSecToTime(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	秒数	输入	秒数	(*)	ns	0
Out	时间	输出	时间	遵从数据类型	ns	–

* –9223372036854775808 ~ 9223372036854775807

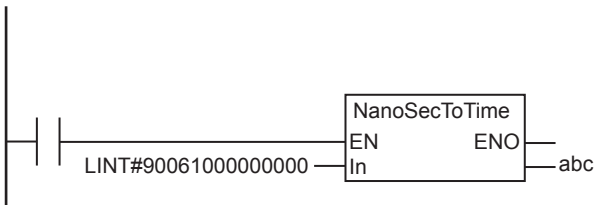
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In													○							
Out																○				

功能

将纳秒数 “In”转换为时间。

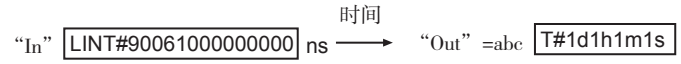
“In”=LINT#90061000000000时的示例如下所示。

LD



ST

```
abc:=NanoSecToTime(LINT#90061000000000);
```



参考

欲将时间转换为纳秒时，请使用 “TimeToNanoSec指令(P.2–624)”。

SecToTime

将秒数转换为时间。

指令	名称	FB/FUN	图形表现	ST表现
SecToTime	秒→时间转换	FUN	(@)SecToTime EN ENO In Out	Out:=SecToTime(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	秒数	输入	秒数	–9223372036 ~ 9223372036	秒	0
Out	时间	输出	时间	遵从数据类型	ns	–

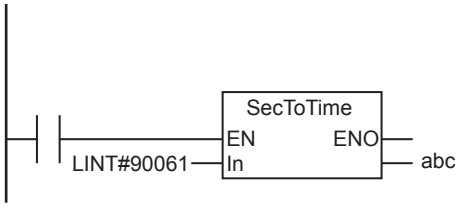
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In													○							
Out																○				

功能

将秒数 “In”转换为时间。

“In”=LINT#90061时的示例如下所示。

LD



ST

```
abc:=SecToTime(LINT#90061);
```

“In” LINT#90061 s —时间→ “Out”=abc T#1d1h1m1s

参考

欲将时间转换为秒时，请使用 “TimeToSec指令(P.2–625)”。

使用注意事项

- “In”的单位为秒，“Out”的单位为纳秒。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “In”的值超过有效范围时。

ChkLeapYear

判定指定的年份是否为闰年。

指令	名称	FB/FUN	图形表现	ST表现
ChkLeapYear	闰年判别	FUN	(@)ChkLeapYear EN ENO In Out	Out:=ChkLeapYear(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	公历年	输入	公历年	1970 ~ 2554	年	1970
Out	判定结果	输出	TRUE：是闰年 FALSE：非闰年	遵从数据类型	–	–

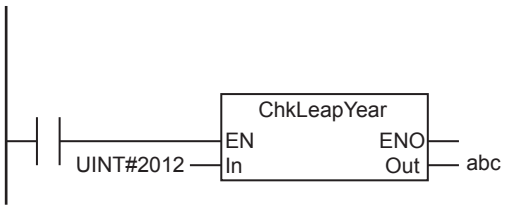
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In							○													
Out	○																			

功能

判断公历年 “In”是否为闰年。如果是闰年，则判定结果 “Out”的值为TRUE。否则为FALSE。

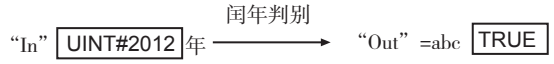
“In”=UINT#2012时的示例如下所示。

LD



ST

```
abc:=ChkLeapYear(UINT#2012);
```



使用注意事项

“In”的值超过有效范围时，不会发生异常，“Out”的值为错误值。

GetDaysOfMonth

获取指定月的天数。

指令	名称	FB/FUN	图形表现	ST表现
GetDaysOfMonth	月的天数获取	FUN	(@)GetDaysOfMonth EN ENO Year Out Month	Out:=GetDaysOfMonth(Year, Month);

变量

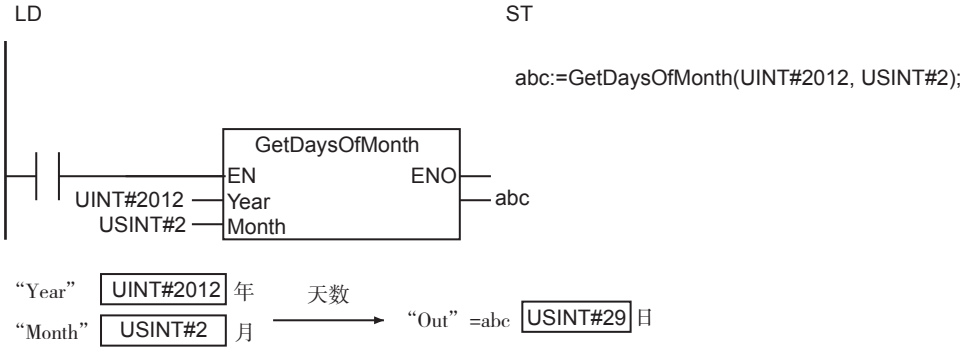
	名称	输入/输出	内容	有效范围	单位	初始值
Year	年	输入	公历年	1970 ~ 2554	年	1970
Month	月		月	1 ~ 12	月	1
Out	天数	输出	天数	28 ~ 31	日	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Year							○													
Month						○														
Out						○														

功能

获取 “Year”表示年的天数、“Month”表示月的天数。

“Year”=UINT#2012、“Month”=USINT#2时的示例如下所示。



使用注意事项

- “Year”的值超过有效范围时，不会发生异常，“Out”的值为错误值。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Month”的值超过有效范围时。

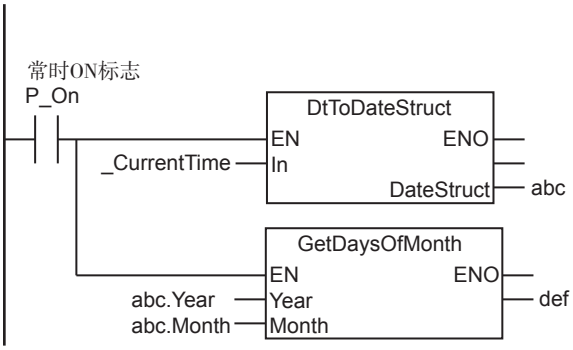
示例程序

获取本月的天数。

LD

内部变量	名称	数据类型	初始值	注释
	abc	_sDT	(Year:=0, Month:=0, Day:=0, Hour:=0, Min:=0, Sec:=0, NSec:=0)	日期时刻
	def	USINT	0	本月的天数

外部变量	名称	数据类型	常数	注释
	_CurrentTime	DATE_AND_TIME	☑	系统时刻



ST

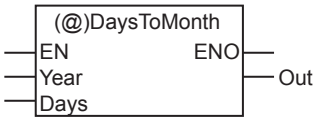
内部变量	名称	数据类型	初始值	注释
	abc	_sDT	(Year:=0, Month:=0, Day:=0, Hour:=0, Min:=0, Sec:=0, NSec:=0)	日期时刻
	def	USINT	0	本月的天数

外部变量	名称	数据类型	常数	注释
	_CurrentTime	DATE_AND_TIME	☑	系统时刻

```
DtToDateStruct(_CurrentTime, abc);
def:=GetDaysOfMonth(abc.Year, abc.Month);
```

DaysToMonth

根据从1月1日起的天数，计算出该日为哪月。

指令	名称	FB/FUN	图形表现	ST表现
DaysToMonth	天数→月转换	FUN		Out:=DaysToMonth(Year, Days);

变量

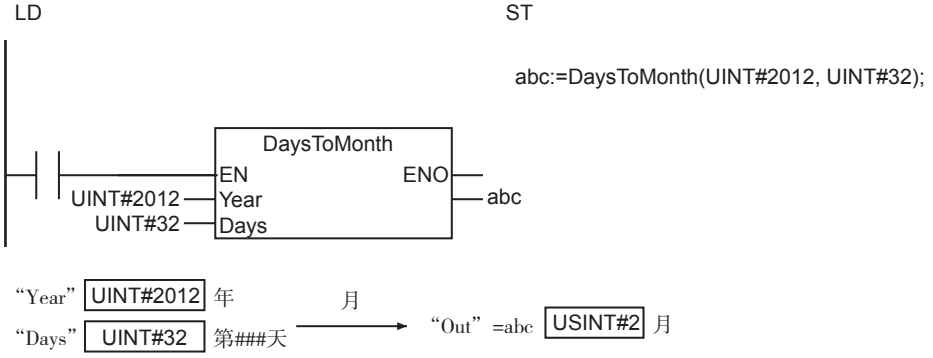
	名称	输入/输出	内容	有效范围	单位	初始值
Year	年	输入	公历年	1970 ~ 2554	年	1970
Days	天数		1月1日起的天数	1 ~ 365 “Year”为闰年时则为1 ~ 366	日	1
Out	月	输出	月	1 ~ 12	月	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Year							○													
Days							○													
Out						○														

功能

根据从年 “Year”的1月1日起的天数 “Days”，计算出该日为哪月。

“Year”=UINT#2012，“Days”=UINT#32时的示例如下所示。

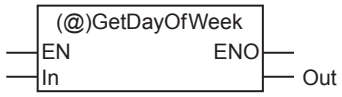


使用注意事项

- “Year”的值超过有效范围时，不会发生异常，“Out”的值为错误值。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Days”的值超过有效范围时。

GetDayOfWeek

获取指定年月日的星期。

指令	名称	FB/FUN	图形表现	ST表现
GetDayOfWeek	星期获取	FUN		Out:=GetDayOfWeek(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	年月日	输入	年月日	遵从数据类型	年月日	(*)
Out	星期	输出	星期	_MON,_TUE, _WED,_THU, _FRI,_SAT, _SUN	星期	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																	○		○	
Out	枚举体_eDAYOFWEEK　枚举元素参阅功能说明																			

功能

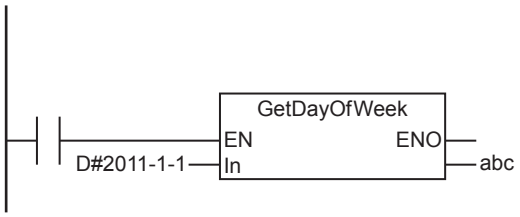
获取年月日 “In”表示年月日的星期。

“Out”的数据类型为枚举体_eDAYOFWEEK。枚举元素的含义如下所示。

枚举元素	含义
_MON	星期一
_TUE	星期二
_WED	星期三
_THU	星期四
_FRI	星期五
_SAT	星期六
_SUN	星期天

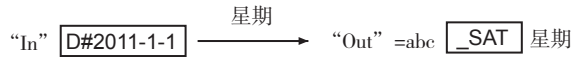
“In”=D#2011–1–1时的示例如下所示。

LD



ST

```
abc:=GetDayOfWeek(D#2011-1-1);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

GetWeekOfYear

计算指定年月日为当年的第几周。

指令	名称	FB/FUN	图形表现	ST表现
GetWeekOfYear	周获取	FUN	(@)GetWeekOfYear EN ENO In Out	Out:=GetWeekOfYear(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	年月日	输入	年月日	遵从数据类型	年月日	(*)
Out	周	输出	当年的第几周	1 ~ 54	周	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

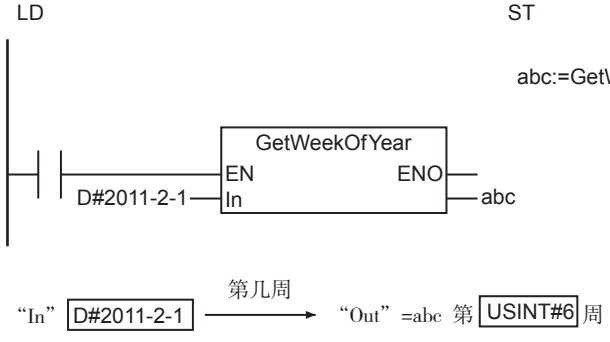
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																	○		○	
Out						○														

功能

计算年月日 “In”表示年月日为当年的第几周。
以星期一到星期天为1周，在星期天变为星期一时对周数进行正计数。

1月1日必定是第1周。
例如，假设1月1日为星期四，则从1月1日(星期四)到1月4日(星期天)为第1周；1月5日(星期一)到1月11日(星期天)为第2周。

“In”=D#2011–2–1时的示例如下所示。

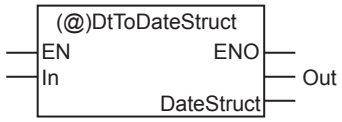


相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

DtToDateStruct

将日期时间分解为 “年”、“月”、“日”、“ 时”、“ 分”、“秒”、“纳秒”。

指令	名称	FB/FUN	图形表现	ST表现
DtToDateStruct	时刻分解	FUN		Out:=DtToDateStruct(In, DateStruct);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	日期时刻	输入	日期时刻	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
DateStruct	日期时刻		分解为 “年”、“月”、“日”、“ 时”、“ 分”、“秒”、“纳秒” 的日期时刻	–		

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Out	○																			
DateStruct	结构体_sDT 详情参阅功能说明																			

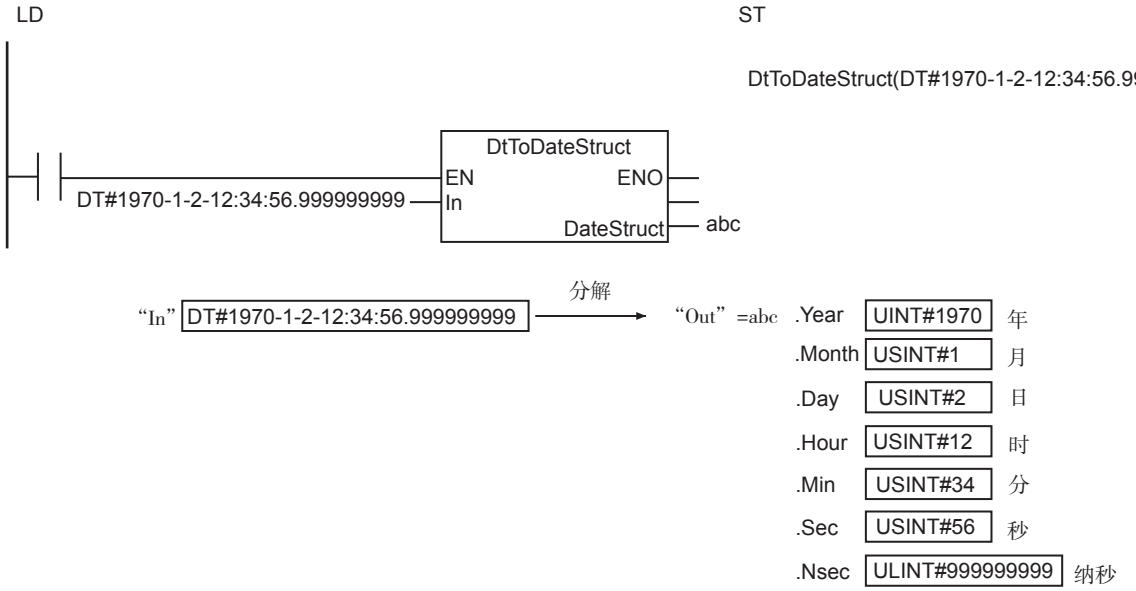
功能

将日期时刻 “In” 分解为 “年”、“月”、“日”、“ 时”、“ 分”、“秒”、“纳秒”。

已分解日期时刻 “DateStruct” 的数据类型为结构体_sDT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DateStruct	日期时刻	分解为 “年”、“月”、“日”、“ 时”、“ 分”、“秒”、“纳秒” 的日期时刻	_sDT	–	–	–
Year	年	年	UINT	1970 ~ 2554	年	–
Month	月	月	USINT	1 ~ 12	月	
Day	日	日	USINT	1 ~ 31	日	
Hour	时	时	USINT	0 ~ 23	时	
Min	分	分	USINT	0 ~ 59	分	
Sec	秒	秒	USINT	0 ~ 59	秒	
Nsec	纳秒	纳秒	ULINT	0 ~ 999999999	纳秒	

“In”=DT#1970–1–2–12:34:56.999999999时的示例如下所示。

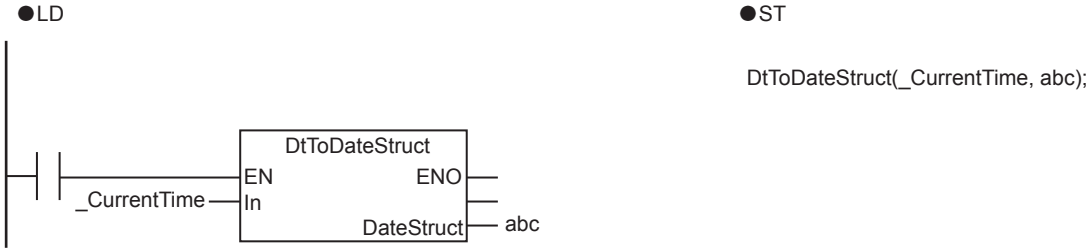


相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

参考

- 组合分解的 “年” 、“月” 、“日” 、“ 时” 、“ 分” 、“秒” 、“纳秒”设为日期时间时，请使用 “DateStructToDt指令(P.2–640)”。
- 求得当前时刻的示例如下所示。

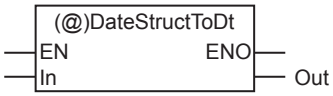


使用注意事项

在ST程序中使用本指令时，不使用返回值 “Out”。

DateStructToDt

组合分解为 “年”、“月”、“日”、“时”、“分”、“秒”、“纳秒”的日期时刻。

指令	名称	FB/FUN	图形表现	ST表现
DateStructToDt	时刻组合	FUN		Out:=DateStructToDt(In);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	日期时刻	输入	分解为“年”、“月”、“日”、“时”、“分”、“秒”、“纳秒”的日期时刻	–	–	–
Out	日期时刻	输出	日期时刻	遵从数据类型	年月日时分秒	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In	结构体_sDT 详情参阅功能说明																			
Out																			○	

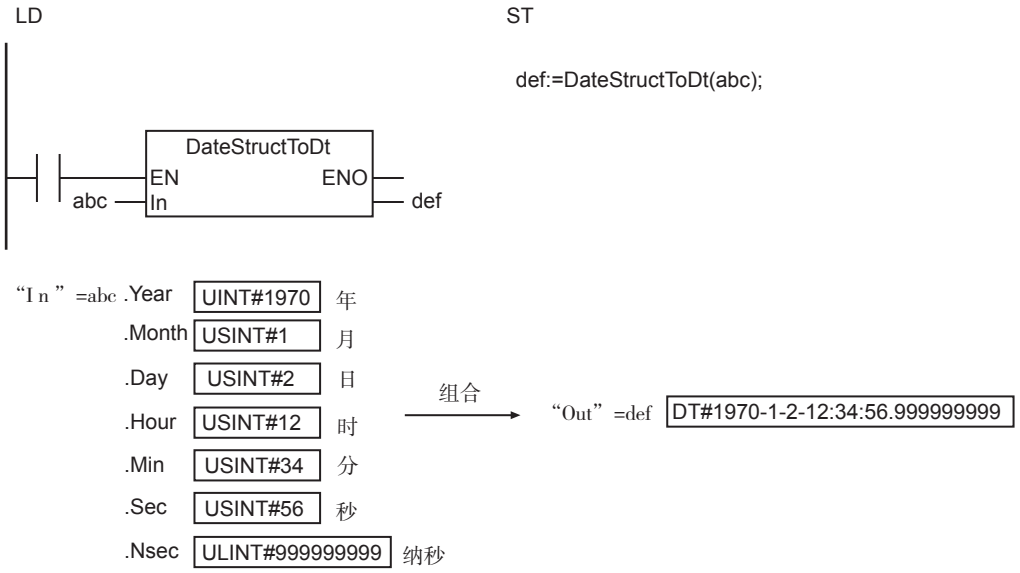
功能

组合分解为 “年”、“月”、“日”、“时”、“分”、“秒”、“纳秒”的日期时刻 “In”。

“In”的数据类型为结构体_sDT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
In	日期时刻	分解为“年”、“月”、“日”、“时”、“分”、“秒”、“纳秒”的日期时刻	_sDT	–	–	–
Year	年	年	UINT	1970 ~ 2554	年	1970
Month	月	月	USINT	1 ~ 12	月	1
Day	日	日	USINT	1 ~ 31	日	
Hour	时	时	USINT	0 ~ 23	时	0
Min	分	分	USINT	0 ~ 59	分	
Sec	秒	秒	USINT	0 ~ 59	秒	
Nsec	纳秒	纳秒	ULINT	0 ~ 999999999	纳秒	

“In”各元素的值：“Year”=UINT#1970、“Month”=USINT#1、“Day”=USINT#2、“Hour”USINT#=12、“Min”=USINT#34、“Sec”=USINT#56、“Nsec”=ULINT#999999999时的示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CurrentTime	系统时刻	DT	系统内置时钟的时刻。1970年1月1日0时0分0秒起的秒数。

参考

欲将日期时刻分解为 “年”、“月”、“日”、“时”、“分”、“秒”、“纳秒”时，请使用
“DtToDateStruct指令(P.2-638)”。

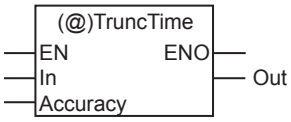
使用注意事项

以下情况时会发生异常。ENO变为FALSE，“Out”不变。

- “In”的任一结构要素的值超过有效范围时。
- 处理结果超过 “Out”的有效范围时。

TruncTime

舍去TIME型变量中小于指定单位的值。

指令	名称	FB/FUN	图形表现	ST表现
TruncTime	时间舍去	FUN		Out:=TruncTime(In, Accuracy);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	对象时间	输入	时间舍去的对象	遵从数据类型	ns	T#0s
Accuracy	舍去单位		不舍去的最小时间单位	_NANOSEC, _MICROSEC, _MILLISEC, _SEC	–	_NANOSEC
Out	舍去后时间	输出	舍去后的时间	遵从数据类型	ns	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																○				
Accuracy	枚举体_eSUBSEC　枚举元素参阅功能说明																			
Out																○				

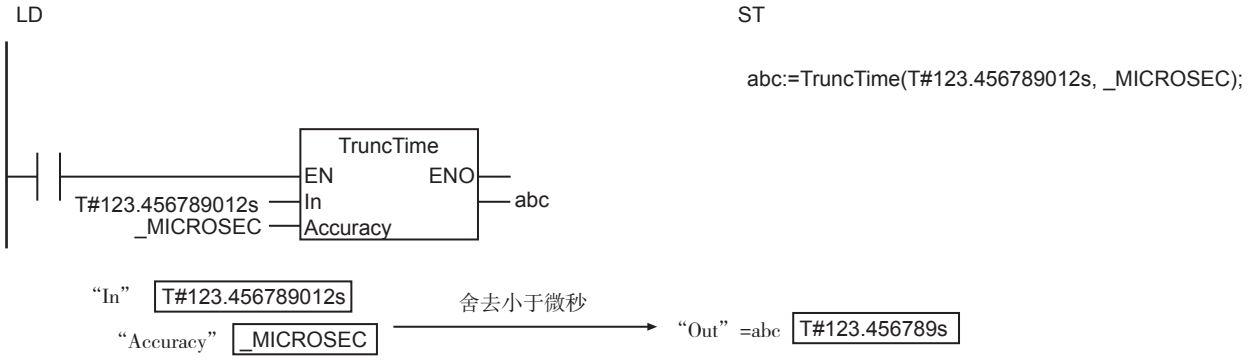
功能

舍去对象时间 “In”中小于舍去单位 “Accuracy”的值。舍去后的值保存在舍去后时间 “Out”中。

“Accuracy”的数据类型为枚举体_eSUBSEC。枚举元素的含义如下所示。

枚举元素	含义
_NANOSEC	纳秒
_MICROSEC	微秒
_MILLISEC	毫秒
_SEC	秒

“In”=TIME#123.456789012s，“Accuracy”=_MICROSEC时的示例如下所示。



参考

使用 “EQ，=指令(P.2–94)”等对不同时间比较大小时，使用本指令使两者的精度一致。

使用注意事项

版本相关信息

本指令可用于Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio。

示例程序

是确认传感器输出的ON时间是否在阈值以上的示例程序。
动作模式分为阈值设定模式和执行模式，分别执行如下动作。

动作模式	动作
阈值设定	测量传感器输出的ON时间，将该值设为阈值。
执行	测量传感器输出的ON时间，将该值与阈值比较。如果ON时间在阈值以上，则可判断为正常。

按照以ms为单位的精度对时间进行比较。因此，使用TruncTime指令，舍去小于测量时间的ms的值。
当前的动作模式保存在变量RecentMode中。判断结果保存在变量Result中。Result的值为TRUE表示正常，FALSE表示异常。

全局变量的定义

数据类型 枚举体

名称	枚举值	注释
Mode		动作模式
SET	0	阈值设定
EXEC	1	执行

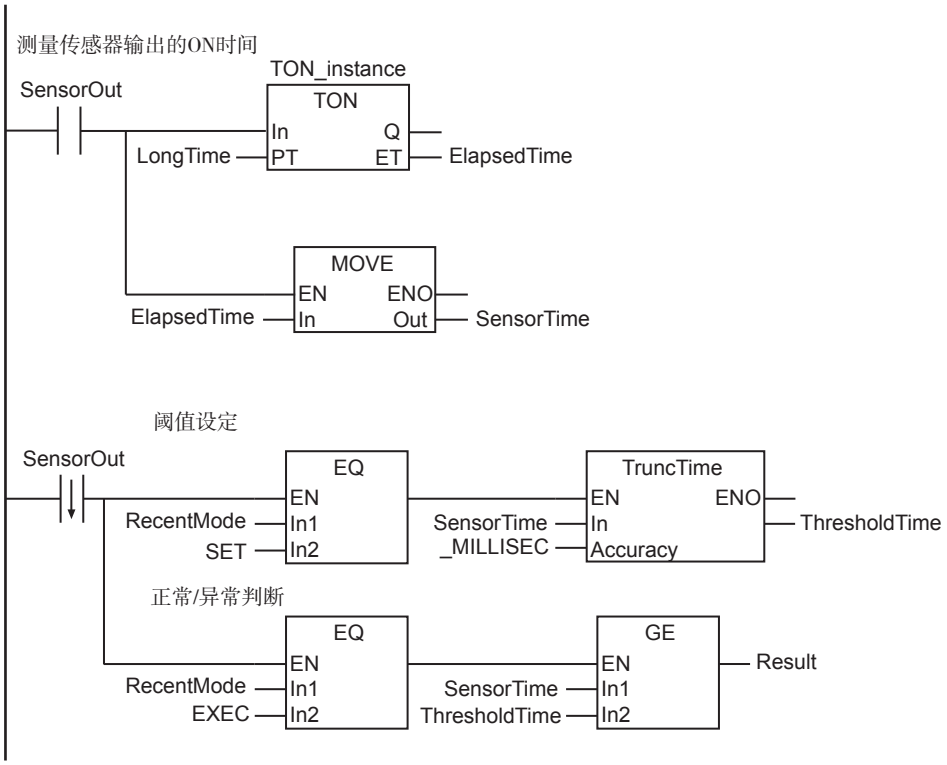
全局变量

名称	数据类型	初始值	注释
RecentMode	Mode	SET	当前的动作模式

LD

内部变量	名称	数据类型	初始值	注释
	SensorOut	BOOL	FALSE	传感器输出
	ElapsedTime	TIME	T#0s	经过时间
	SensorTime	TIME	T#0s	传感器的ON时间
	LongTime	TIME	T#1h	比传感器的ON时间足够长的时间
	ThresholdTime	TIME	T#0s	阈值
	Result	BOOL	FALSE	判断结果　TRUE：正常　FALSE：异常
	TON_instance	TON		

外部变量	名称	数据类型	注释
	RecentMode	Mode	当前的动作模式



ST

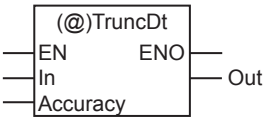
内部变量	名称	数据类型	初始值	注释
	SensorOut	BOOL	FALSE	传感器输出
	ElapsedTime	TIME	T#0s	经过时间
	SensorTime	TIME	T#0s	传感器的ON时间
	LongTime	TIME	T#1h	比传感器的ON时间足够长的时间
	SensorDone	BOOL	FALSE	传感器输出OFF标志
	ThresholdTime	TIME	T#0s	阈值
	Result	BOOL	FALSE	判断结果　TRUE：正常　FALSE：异常
	TON_instance	TON		
	F_TRIG_instance	F_TRIG		

外部变量	名称	数据类型	注释
	RecentMode	Mode	当前的动作模式

```
// 执行TON指令
TON_instance(
    In:=SensorOut,        // 定时器输入
    PT:=LongTime,         // 设定时间
    ET=>ElapsedTime);  // 经过时间

// 将TON的经过时间设为传感器的ON时间
IF (SensorOut=TRUE) THEN
    SensorTime:=ElapsedTime;
END_IF;

// 传感器输出的下降沿检测
F_TRIG_instance(Clk:=SensorOut, Q=>SensorDone);
Result:=FALSE;

//   阈值设定
IF (SensorDone=TRUE AND RecentMode=SET) THEN
    ThresholdTime:=TruncTime(
                        In          :=SensorTime,
                        Accuracy:=_MILLISEC);  //精度 ms单位
// 正常/异常判断
ELSIF (SensorDone=TRUE AND RecentMode=EXEC) THEN
    IF (SensorTime >= ThresholdTime) THEN
        Result:=TRUE;
    END_IF;
END_IF;
```

TruncDt

舍去DT型变量中小于指定单位的值。

指令	名称	FB/FUN	图形表现	ST表现
TruncDt	日期时刻舍去	FUN		Out:=TruncDt(In, Accuracy);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	对象日期时刻	输入	日期时刻舍去的对象	遵从数据类型	年月日时分秒	DT#1970-1-1-0:0:0
Accuracy	舍去单位		不舍去的最小时刻单位	_NANOSEC, _MICROSEC, _MILLISEC, _SEC	–	_NANOSEC
Out	舍去后日期时刻	输出	舍去后的日期时刻	遵从数据类型	年月日时分秒	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																			○	
Accuracy	枚举体_eSUBSEC　枚举元素参阅功能说明																			
Out																			○	

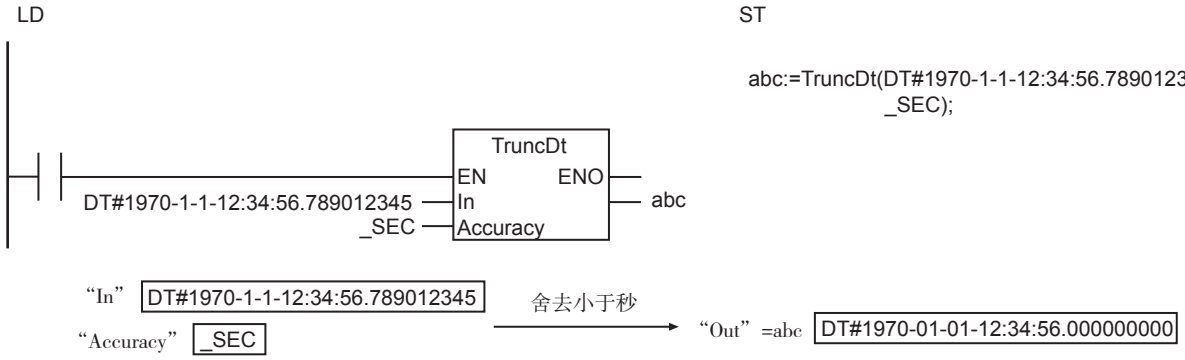
功能

舍去对象日期时刻“In”中小于舍去单位“Accuracy”的值。舍去后的值保存在舍去后日期时刻“Out”中。

“Accuracy”的数据类型为枚举体_eSUBSEC。枚举元素的含义如下所示。

枚举元素	含义
_NANOSEC	纳秒
_MICROSEC	微秒
_MILLISEC	毫秒
_SEC	秒

“In”=DT#1970-1-1-12:34:56.789012345，“Accuracy”=_SEC时的示例如下所示。



参考

使用 “EQ，=指令(P.2-94)”等对不同日期时刻比较大小时，使用本指令使两者的精度一致。

使用注意事项

版本相关信息

本指令可用于Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio。

示例程序

是对传感器输出为ON的日期时刻和该时的电压值进行记录的示例程序。
按照ms单位的精度记录日期时刻。

传感器输出保存在SensorOut中，电压值保存在Voltage中。利用GetTime指令获取当前的日期时刻。
要记录的日期时刻和电压值，以将其作为结构要素的结构体Recent的形式，依次保存到堆栈Stack中。

全局变量的定义

数据类型

名称	数据类型	注释
Record	STRUCT	结构体
DandT	DT	日期时刻
Voltage	REAL	电压值

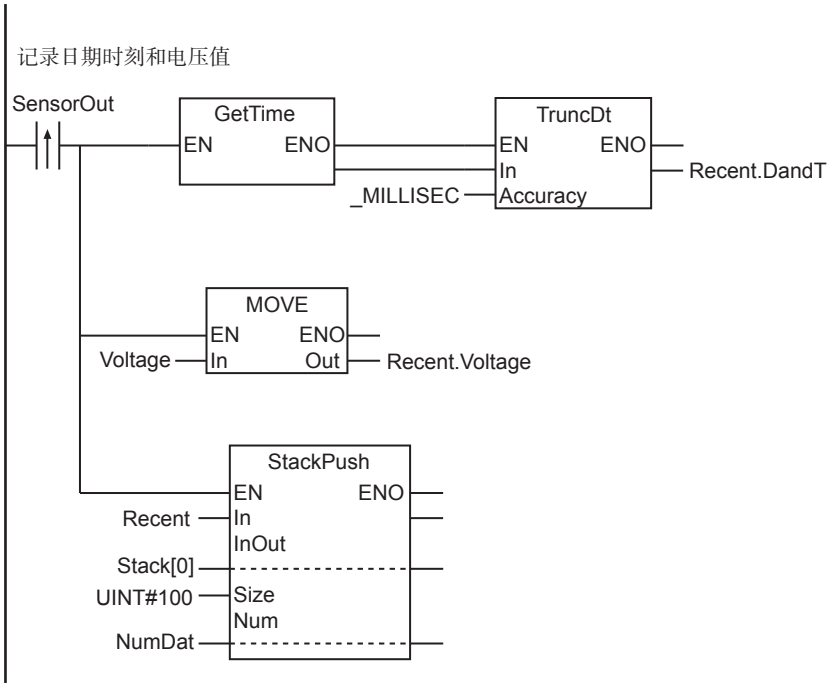
全局变量

名称	数据类型	初始值	注释
Recent	Record	(DandT:=DT#1970-1-1-0:0:0,Voltage:=0.0)	当前值
Stack	ARRAY[0..99] OF Record	[100((DandT:=DT#1970-1-1-0:0:0,Voltage:=0.0))]	堆栈

LD

内部变量	名称	数据类型	初始值	注释
	SensorOut	BOOL	FALSE	传感器输出
	Voltage	REAL	0.0	电压值
	NumDat	UINT	UINT#0	当前的记录数据数

外部变量	名称	数据类型	注释
	Recent	Record	当前值
	Stack	ARRAY[0..99] OF Record	堆栈



ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	触发
	SensorOut	BOOL	FALSE	传感器输出
	Voltage	REAL	0.0	电压值
	NumDat	UINT	UINT#0	当前的记录数据数
	R_TRIG_instance	R_TRIG		

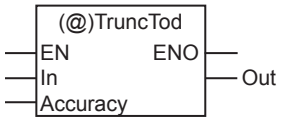
外部变量	名称	数据类型	注释
	Recent	Record	当前值
	Stack	ARRAY[0..99] OF Record	堆栈

```
// 传感器输出为ON后启动触发
R_TRIG_instance(SensorOut, Trigger);

IF (Trigger=TRUE) THEN
    // 按照ms单位的精度保存当前的日期时刻
    Recent.DandT:=TruncDt(
                    In        :=GetTime(),      // 日期时刻获取
                    Accuracy :=_MILLISEC);   // 精度   ms单位
```

```
    // 获取电压值的当前值
    Recent.Voltage:=Voltage;

    // 将日期时刻和电压值保存在堆栈中
    StackPush(
        In     :=Recent,      // 日期时刻和电压值
        InOut :=Stack[0],     // 堆栈数组
        Size   :=UINT#100,    // 堆栈数组的要素数100
        Num   :=NumDat);      // 当前的记录数据数
END_IF;
```

TruncTod

舍去TOD型变量中小于指定单位的值。

指令	名称	FB/FUN	图形表现	ST表现
TruncTod	时刻舍去	FUN		Out:=TruncTod(In, Accuracy);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	对象时刻	输入	时刻舍去的对象	遵从数据类型	时分秒	TOD#0:0:0
Accuracy	舍去单位		不舍去的最小时刻单位	_NANOSEC, _MICROSEC, _MILLISEC, _SEC	–	_NANOSEC
Out	舍去后时刻	输出	舍去后的时刻	遵从数据类型	时分秒	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In																		○		
Accuracy	枚举体_eSUBSEC　枚举元素参阅功能说明																			
Out																		○		

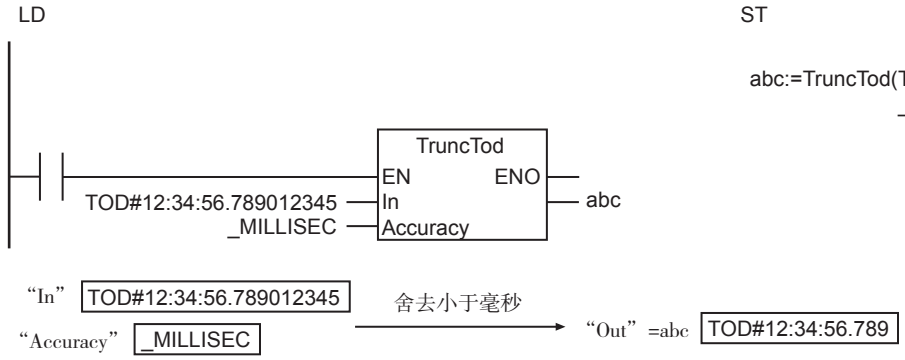
功能

舍去对象时刻 “In”中小于舍去单位 “Accuracy”的值。舍去后的值保存在舍去后时刻 “Out”中。

“Accuracy”的数据类型为枚举体_eSUBSEC。枚举元素的含义如下所示。

枚举元素	含义
_NANOSEC	纳秒
_MICROSEC	微秒
_MILLISEC	毫秒
_SEC	秒

“In”=TOD#12:34:56.789012345，“Accuracy”=_MILLISEC时的示例如下所示。



参考

使用 “EQ，=指令(P.2–94)”等对不同时刻比较大小时，使用本指令使两者的精度一致。

使用注意事项

版本相关信息

本指令可用于Ver.1.01以上的CPU单元和Ver.1.02以上的Sysmac Studio。

示例程序

是对传感器输出为ON的时刻和该时的电压值进行记录的示例程序。
按照秒单位的精度记录时刻。

传感器输出保存在SensorOut中，电压值保存在Voltage中。组合GetTime指令和DT_TO_TOD指令获取当前时刻。
要记录的时刻和电压值，以将其作为结构要素的结构体Recent的形式，依次保存到堆栈Stack中。

全局变量的定义

数据类型

名称	数据类型	注释
Record	STRUCT	结构体
TofD	TOD	时刻
Voltage	REAL	电压值

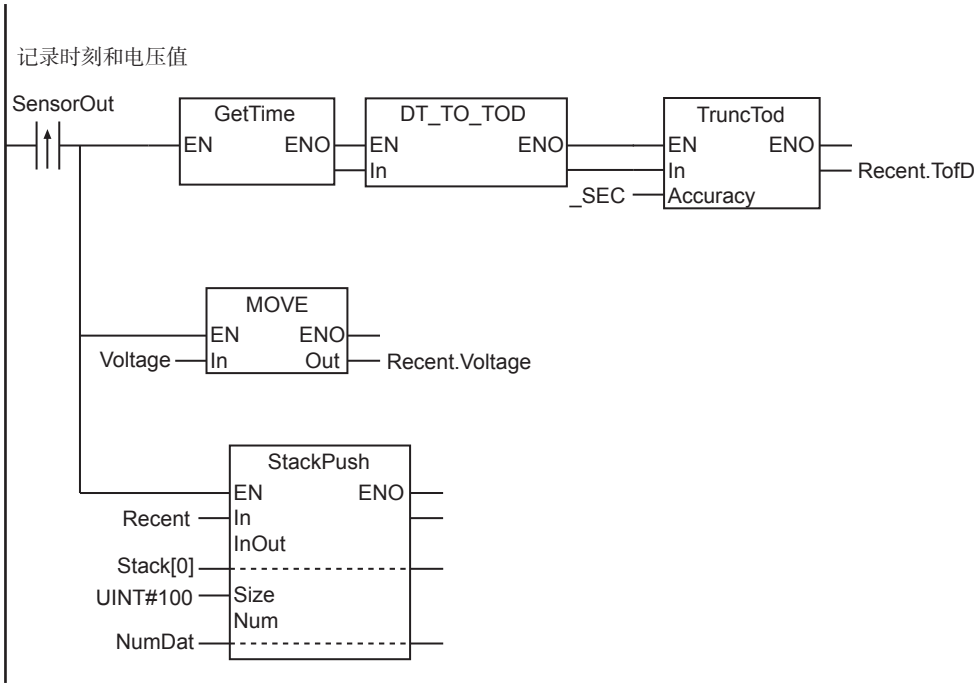
全局变量

名称	数据类型	初始值	注释
Recent	Record	(TofD:=TOD#0:0:0,Voltage:=0.0)	当前值
Stack	ARRAY[0..99] OF Record	[100((TofD:=TOD#0:0:0,Voltage:=0.0))]	堆栈

LD

内部变量	名称	数据类型	初始值	注释
	SensorOut	BOOL	FALSE	传感器输出
	Voltage	REAL	0.0	电压值
	NumDat	UINT	UINT#0	当前的记录数据数

外部变量	名称	数据类型	注释
	Recent	Record	当前值
	Stack	ARRAY[0..99] OF Record	堆栈



ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	触发
	SensorOut	BOOL	FALSE	传感器输出
	TmpTod	TOD	TOD#0:0:0	临时变量
	Voltage	REAL	0.0	电压值
	NumDat	UINT	UINT#0	当前的记录数据数
	R_TRIG_instance	R_TRIG		

外部变量	名称	数据类型	注释
	Recent	Record	当前值
	Stack	ARRAY[0..99] OF Record	堆栈

```
// 传感器输出为ON后启动触发
R_TRIG_instance(SensorOut, Trigger);

IF (Trigger=TRUE) THEN
    // 按照秒单位的精度保存当前的时刻
    TmpTod      :=DT_TO_TOD(GetTime()); // 获取时刻
    Recent.TofD :=TruncTod(
```

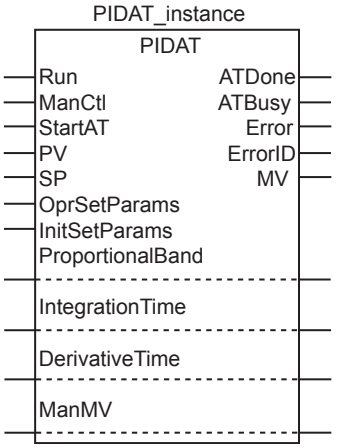
```
                In          :=TmpTod,
                Accuracy :=_SEC);  // 精度 秒单位

    // 获取电压值的当前值
    Recent.Voltage:=Voltage;

    // 将时刻和电压值保存在堆栈中
    StackPush(
        In     :=Recent,    // 时刻和电压值
        InOut:=Stack[0],   // 堆栈数组
        Size   :=UINT#100,// 堆栈数组的要素数100
        Num  :=NumDat);  // 当前的记录数据数
END_IF;
```

模拟控制指令

指令	名称	页码
PIDAT	带自动调谐的PID运算	2–656
PIDAT_HeatCool	带自动调谐的加热冷却PID运算	2–682
TimeProportionalOut	分时比例输出	2–719
LimitAlarm_**	上下限警报组	2–734
LimitAlarmDv_**	上下限偏差警报组	2–738
LimitAlarmDvStbySeq_**	带待机时序的上下限偏差警报组	2–742
ScaleTrans	比例转换	2–756
AC_StepProgram	步程序	2–759

PIDAT

执行带自动调谐的PID运算(目标值滤波器型2自由度PID控制)。

指令	名称	FB/FUN	图形表现	ST表现
PIDAT	带自动调谐的PID运算	FB		PIDAT_instance(Run, ManCtl, StartAT, PV, SP, OprSetParams, InitSetParams, ProportionalBand, IntegrationTime, DerivativeTime, ManMV, ATDone, ATBusy, Error, ErrorID, MV);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Run	执行条件	输入	TRUE：执行 FALSE：停止	遵从数据类型	–	FALSE
ManCtl	手动/自动切换		TRUE：手动运行 FALSE：自动运行			
StartAT	自动调谐执行条件		TRUE：执行 FALSE：中止			
PV	测量值(当前值)		测量值(当前值)	(*1)		0
SP	目标值		目标值			
OprSet Params	运行中设定参数		运行中的各种设定参数	–		–
InitSet Params	初始设定参数		初始设定参数			
Proportional Band	比例带	输入输出	比例带	0.01 ~ 1000.00	%FS	–
Integration Time	积分时间		积分时间 值越大则积分效应越小。 为0时，无积分动作。	T#0.0000s ~ T#10000.0000s (*2)	s	
Derivative Time	微分时间		微分时间 值越大则微分效应越大。 为0时，无微分动作。	T#0.0000s ~ T#10000.0000s (*2)		
ManMV	手动操作量		手动操作量	–320 ~ 320	%	

	名称	输入/输出	内容	有效范围	单位	初始值
ATDone	自动调谐正常结束	输出	TRUE：正常结束 FALSE：(*3)	遵从数据类型	–	–
ATBusy	自动调谐执行中		TRUE：执行中 FALSE：非执行中			
MV	操作量		操作量	–320 ~ 320	%	

*1 输入范围下限 “InitSetParams.RngLowLmt”的值 ~ 输入范围上限 “InitSetParams.RngUpLmt”的值。
*2 小于0.0001s时舍去。
*3 异常结束或不执行自动调谐即处于PID运算状态或未处于PID运算状态。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Run	○																			
ManCtl	○																			
StartAT	○																			
PV														○						
SP														○						
OprSet Params	结构体_sOPR_SET_PARAMS 详情参阅功能说明																			
InitSet Params	结构体_sINIT_SET_PARAMS 详情参阅功能说明																			
Proportional Band														○						
Integration Time																○				
Derivative Time																○				
ManMV														○						
ATDone	○																			
ATBusy	○																			
MV														○						

功能

进行PID运算，控制温控器等的操作量。
如果执行条件 “Run”的值变为TRUE，则开始PID运算。此后，“Run”的值为TRUE的期间，定期重复进行测量值 “PV”的获取→PID运算→操作量 “MV”的输出。
如果 “Run”的值变为FALSE，则结束PID运算。

带自动调谐功能，可自动计算PID常数的最佳值。
如果自动调谐执行条件 “StartAT”的值变为TRUE，则执行PID常数的自动调谐。

结构体的规格

运行中设定参数 “OprSetParams”的数据类型为结构体_sOPR_SET_PARAMS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
OprSetParams	运行中设定参数	运行中的各种设定参数	_sOPR_SET_PARAMS	–	–	–
MVLowLmt	操作量限位下限值	对 “MV”进行限位控制的下限值	REAL	–320 ~ 320 (*)	%	0
MVUpLmt	操作量限位上限值	限位控制 “MV”的上限值	REAL			100
ManResetVal	手动复位值	比例动作时，偏差为0时的 “MV”的值	REAL	–320 ~ 320		0
MVTrackSw	MV跟踪开关	TRUE：ON FALSE：OFF	BOOL	遵从数据类型	–	FALSE
MVTrackVal	MV跟踪值	进行MV跟踪时 “MV”的值	REAL	–320 ~ 320	%	0
StopMV	停止时操作量	停止执行指令时 “MV”的值	REAL			
ErrorMV	异常时操作量	发生异常时 “MV”的值	REAL			
Alpha	2自由度PID参数 α	目标值滤波器系数 α 为0时，目标值滤波器无效	REAL	0.00 ~ 1.00	–	0.65
ATCalcGain	自动调谐计算增益	自动调谐结果的调整系数。 值越大则越重视稳定性，越小则越重视快速响应性。	REAL	0.1 ~ 10.0		1.0
ATHystrs	自动调谐滞后	极限环滞后的大小	REAL		%FS	0.2

*“MVLowLmt” < “MVUpLmt”

初始设定参数 “InitSetParams”的数据类型为结构体_sINIT_SET_PARAMS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
InitSetParams	初始设定参数	初始设定参数	_sINIT_SET_PARAMS	–	–	–
SampTime	采样周期	进行PID运算的周期	TIME	T#0.0001s ~ T#100.0000s	s	T#0.1s
RngLowLmt	输入范围下限	“PV”和 “SP”的下限值	REAL	–32000 ~ 32000(*)	–	0
RngUpLmt	输入范围上限	“PV”和 “SP”的上限值	REAL			100
DirOpr	动作方向	TRUE：正动作 FALSE：逆动作	BOOL	遵从数据类型		FALSE

*“RngLowLmt” < “RngUpLmt”

各变量的含义

以下对本指令各变量的含义作详细说明。

- **“Run”(执行条件)**

本指令的执行条件。

值为TRUE的期间，进行PID运算。如果值变为FALSE，则停止PID运算。

- **“ManCtl”(手动/自动切换)**

本指令有手动运行和自动运行2种模式。
根据 “ManCtl”的值，确定在哪种模式下运行。

“ManCtl”的值	运行模式	“MV”的值
TRUE	手动	“ManMV”的值 (不进行PID运算)
FALSE	自动	根据PID运算计算出的数值

- **“StartAT”(自动调谐执行条件)**

PID常数自动调谐的执行条件。
在“StartAT”的值为TRUE的状态下，如果“Run”的值由FALSE变为TRUE，则运行开始时执行自动调谐。
“Run”的值为TRUE且处于PID运算状态时，如果“StartAT”的值由FALSE变为TRUE，则在运行过程中开始自动调谐。
无论哪种情况，如果在自动调谐过程中 “StartAT”的值由TRUE变为FALSE，则中止自动调谐。
自动调谐功能的详情将于下文阐述。

- **“PV”(测量值)**

控制对象的测量值。

- **“SP”(目标值)**

控制对象的目标值。

- **“MVLowLmt”(操作量限位下限值)、“MVUpLmt”(操作量限位上限值)**

可限位控制 “MV”的值。
“MVLowLmt”和 “MVUpLmt”为限位控制的下限值和上限值。
请务必设定为 “MVLowLmt”< “MVUpLmt”。

通过PID运算得出的操作量	“MV”的值
小于 “MVLowLmt”	“MVLowLmt”
大于 “MVLowLmt”，小于 “MVUpLmt”	通过PID运算得出的操作量
大于 “MVUpLmt”	“MVUpLmt”

为 “MV”设置停止时操作量“StopMV”、异常时操作量 “ErrorMV”或手动操作量 “ManMV”中的任意一个时，不属于限位控制的对象。

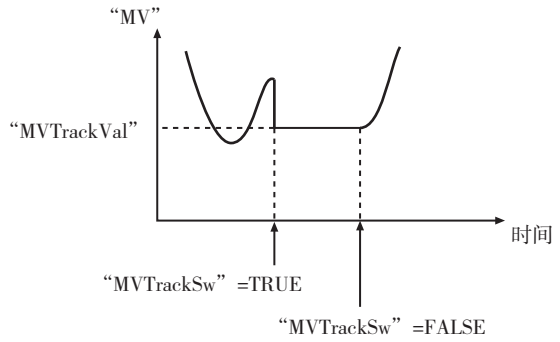
- **“ManResetVal”(手动复位值)**

通过比例动作偏差(“PV”与 “SP”之差)变为0时的 “MV”的值。
根据 “ManResetVal”的值，确定比例动作范围的位置。
进行积分动作时，手动复位值的设定无效。因此，“ManResetVal”的设定在 “IntegrationTime”的值为 0 时有效。

- **“MVTrackSw”(MV跟踪开关)**

自动运行时，为 “MV”强制设置来自外部的输入值(MV跟踪值)的功能称为MV跟踪。
“MVTrackSw”的值为TRUE的期间，执行MV跟踪。

如果“MVTrackSw”的值变为FALSE，则 “MV”的值返回PID运算结果。此时，“MV”的值在无波动(无突变)状态下变化。



- **“MVTrackVal”(MV跟踪值)**

  通过MV跟踪，为 “MV”强制设置的值。

  通过 “MVLowLmt”和 “MVUpLmt”，限位控制 “MVTrackVal”的值。

- **“StopMV”(停止时操作量)**

  “Run”的值变为FALSE，停止执行本指令时 “MV”的值。

- **“ErrorMV”(异常时操作量)**

  发生异常(“Err”的值为TRUE)时 “MV”的值。

  如果 “ErrorMV”的值超过有效范围(–320 ~ 320)，则异常时的 “MV”值变为0。

- **“Alpha”(2自由度PID参数α)**

  确定目标值滤波器系数的参数。

  详情请参阅目标值滤波器型2自由度PID控制的说明。

  一般请将 “Alpha”的值设定为0.65。

- **“ATCalcGain”(自动调谐计算增益)**

  使通过执行自动调谐计算出的PID常数反映为实际PID常数时的系数。

  如果指定1.00，则直接反映自动调谐的结果。

  重视稳定性时增大 “ATCalcGain”的值，重视快速响应性时减小 “ATCalcGain”的值。

- **“ATHystrs”(自动调谐滞后)**

  表示自动调谐时极限环滞后的大小。

  可对较小的“ATHystrs”值进行高精度的调谐。测量值不稳定而难以正常调谐时，增大“ATHystrs”的值。

  详情请参阅自动调谐的说明。

- **“SampTime”(采样周期)**

  进行PID运算的周期的最小值。

  详情请参阅PID运算执行时间的说明。执行PID运算后的经过时间小于 “SampTime”时，不执行下一PID运算。

- **“RngLowLmt”(输入范围下限)、“RngUpLmt”(输入范围上限)**

  “PV”和 “SP”的下限值和上限值。

  与 “PV”和 “SP”连接的参数值偏离该范围时，发生异常。

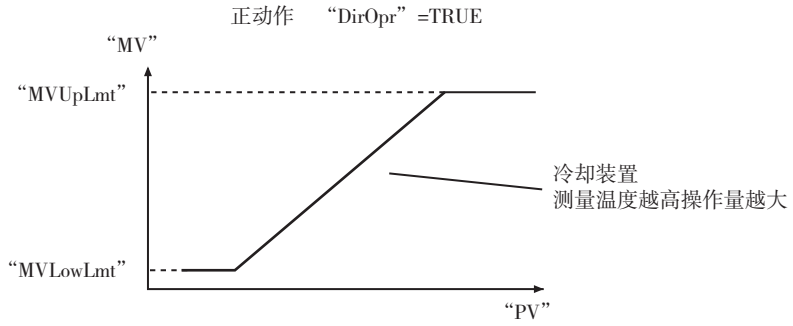
  请务必设定为使 “RngLowLmt”< “RngUpLmt”。

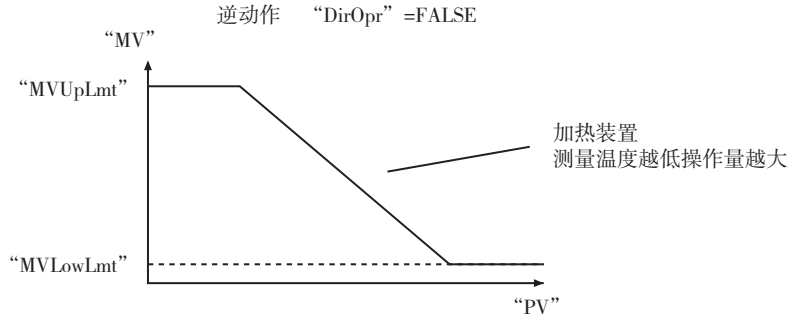
- **“DirOpr”(动作方向)**

  根据 “PV”的大小，指定增大或减小 “MV”。
  分别称为正动作、逆动作。

“DirOpr”的值	含义	“MV”的值
TRUE	正动作	“PV”越大则 “MV”的值越大
FALSE	逆动作	“PV”越大则 “MV”的值越小

  正动作和逆动作的差异将在温度控制时进行说明。
  控制冷却装置的操作量时为正动作。即测量温度越高，则越要增大冷却装置的操作量。相对地，控制加热装置的操作量时为逆动作。测量温度越低，则越要增大加热装置的操作量。





- **“ProportionalBand”(比例带)**

  3个PID常数中的1个。详情请参阅比例动作的说明。
  如果 “ProportionalBand”的值较大，则偏置增大。如果较小，则会发生震荡。

- **“IntegrationTime”(积分时间)**

  3个PID常数中的1个。详情请参阅积分动作的说明。
  “IntegrationTime”的值越大，则积分动作越弱。

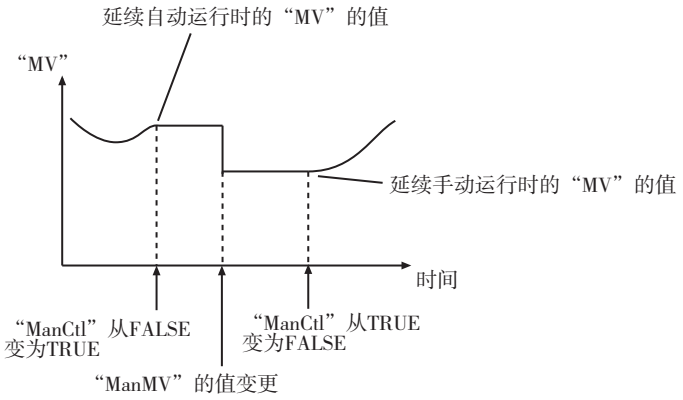
- **“DerivativeTime”(微分时间)**

  3个PID常数中的1个。详情请参阅微分动作的说明。
  “DerivativeTime”的值越大，则微分动作越强。

- **“ManMV”(手动操作量)**

  手动运行(“ManCtl”=TRUE)时 “MV”的值。
  从自动运行切换至手动运行时，直接延续自动运行状态下的“MV”值。切换至手动运行后变更 “ManMV”的值时，为 “MV”设置 “ManMV”的值。
  从手动运行切换至自动运行时，也延续手动运行状态下的 “MV”值。

“ManMV”的值可超过 “MVLowLmt” ~ “MVUpLmt”的范围。



- **“ATDone”(自动调谐正常结束)**

  表示自动调谐正常结束的标志。

  自动调谐正常结束后，“StartAT”的值为TRUE期间，变为TRUE。

  以下情况会变为FALSE。

  - 自动调谐异常结束时。
  - 自动调谐执行中(“ATBusy”的值为TRUE)。
  - 不执行自动调谐即处于PID运算状态。
  - 非PID运算状态(“Run”的值为FALSE)。
  - “StartAT”的值为FALSE。

- **“ATBusy”(自动调谐执行中)**

  表示正在执行自动调谐的标志。

  正在执行自动调谐时，变为TRUE。其余情况下，均变为FALSE。

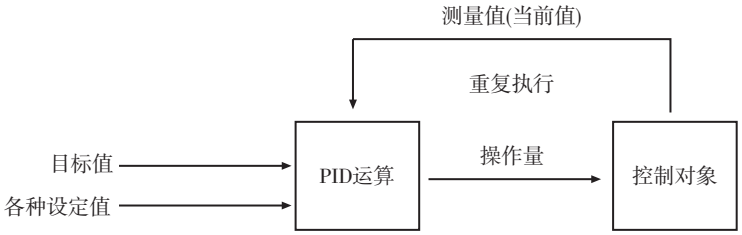
- **“MV”(操作量)**

  给予控制对象的操作量。

PID运算的概要

PID运算是一种重复调节操作量的反馈控制的运算方法，用于测量控制对象的当前值，并使该值接近目标值。

因此，本指令的输入为测量值、目标值及用于运算的各种设定值，输出为操作量。通过定期重复测量→运算→操作量输出，使当前值接近目标值。

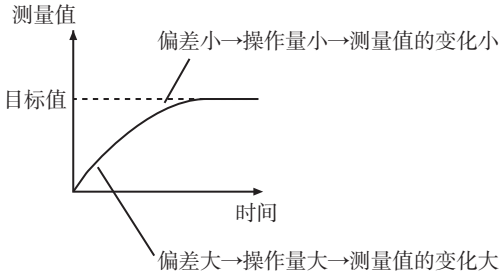


比例动作(P)、积分动作(I)、微分动作(D)

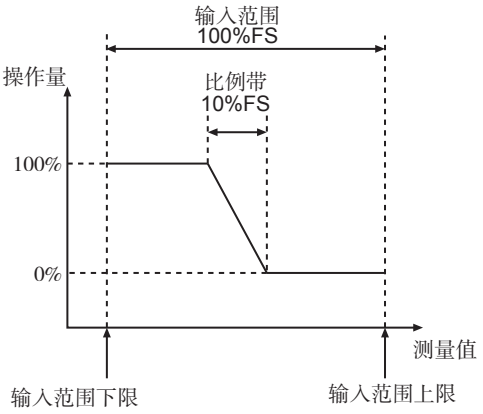
通过合成比例动作(Proportional action)、积分动作(Integral action)、微分动作(Derivative action)，进行PID运算。以下分别进行说明。

● 比例动作(P)

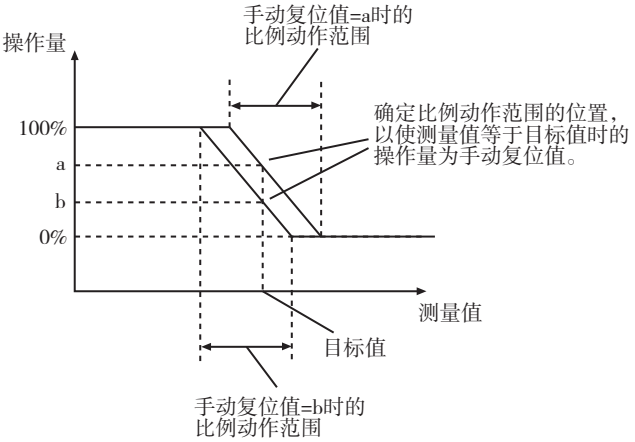
与测量值和目标值之差(偏差)成正比，增大操作量绝对值的控制方法。控制对象的测量值的变化如下图所示。



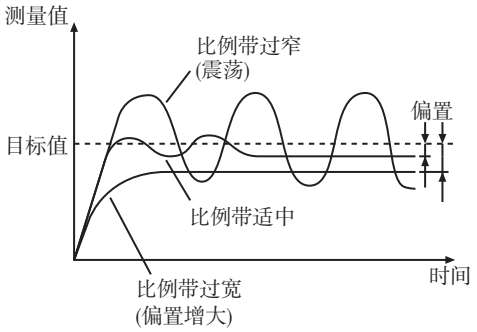
比例动作的设定值的其中之一有比例带。
比例带是指进行比例动作的测量值的宽度。测量值偏离比例动作范围时，将操作量设置为100%或0%。
通过输入范围中进行比例动作的范围的比例(%FS)表现比例带。比例带的值为10%FS时的图如下所示。



比例动作的另一设定值为手动复位值。
手动复位值是指偏差为0时的操作量。通过手动复位值确定测量值和操作量的图中比例动作范围的位置。
手动复位值与比例动作范围的关系如下所示。
确定比例动作范围的位置，以使测量值与目标值一致时的操作量与手动复位值一致。



如果手动复位值不恰当，则完全无法将偏差设为0。该残留的偏差称为偏置(残留偏差)。如果比例带较窄，则可减小偏置。但如果过窄，则会发生测量值超过目标值而不停止的现象。该现象称为过冲。测量值不稳定，在目标值附近振动的现象称为震荡。

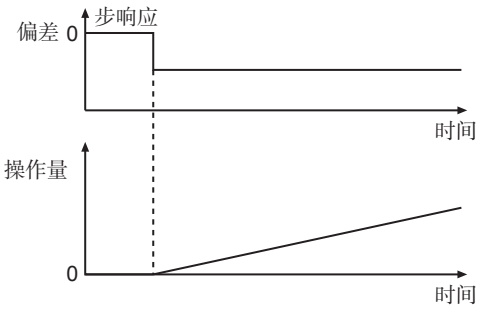


● 积分动作(I)

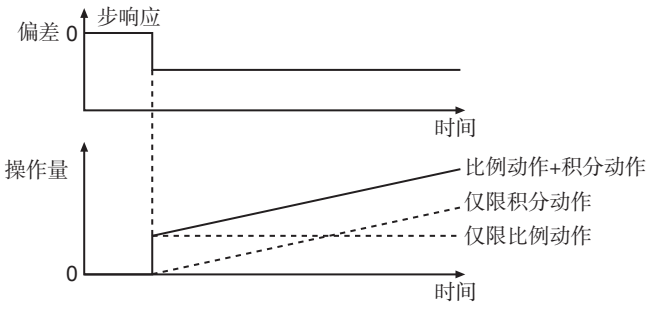
仅通过比例动作将偏置设为0时，需仔细调整比例带和手动复位值。偏置的大小因外部干扰而变化时，须频繁对其进行调整。同时使用积分动作和比例动作可节省时间。积分动作为通过时间轴累积(积分)偏差，与该值成正比，增大操作量绝对值的控制方法。
进行积分动作时，手动复位值的设定无效。

发生阶梯式偏差时，基于积分动作的操作量的变化如左下图所示。同时使用积分动作和比例动作时的操作量的变化如右下图所示。

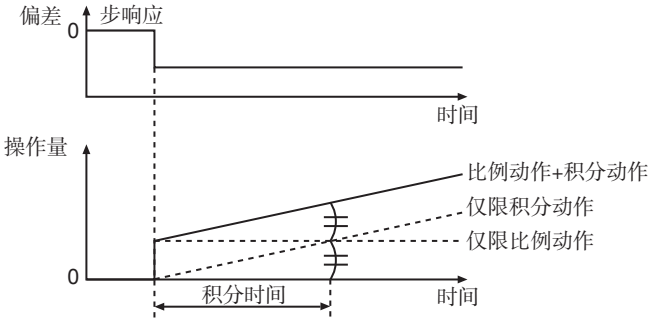
基于积分动作的操作量



基于同时使用比例动作和积分动作的操作量



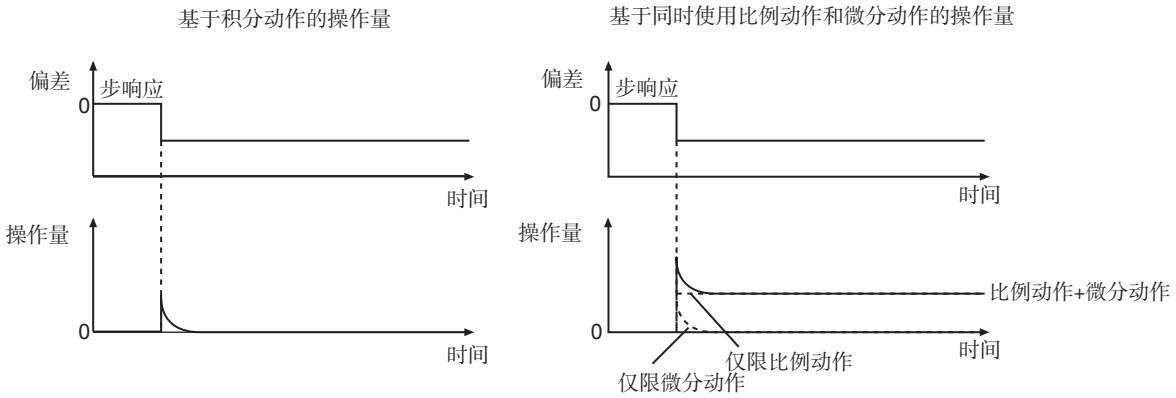
积分动作的设定值带有积分时间。这是针对阶梯式偏差，使基于积分动作的操作量与基于比例动作的操作量相等之前的时间。积分时间越短，则积分动作越强。如果积分时间较短，则将偏置设为0之前的时间相应缩短，否则会发生震荡。



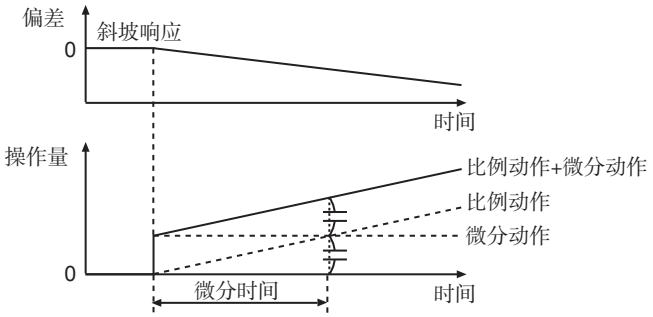
● 微分动作(D)

如果同时使用积分动作和比例动作，则将偏置设为0，可使测量值与目标值一致。但发生外部干扰，测量值突然变化时，需一定的时间使其复原。微分动作用于即使发生外部干扰，仍在短时间内使测量值返回目标值的用途。微分动作为针对时间轴对偏差进行微分处理，与该大小成正比，增大操作量绝对值的控制方法。即测量值的变化越大，则基于微分动作的操作量绝对值越大。

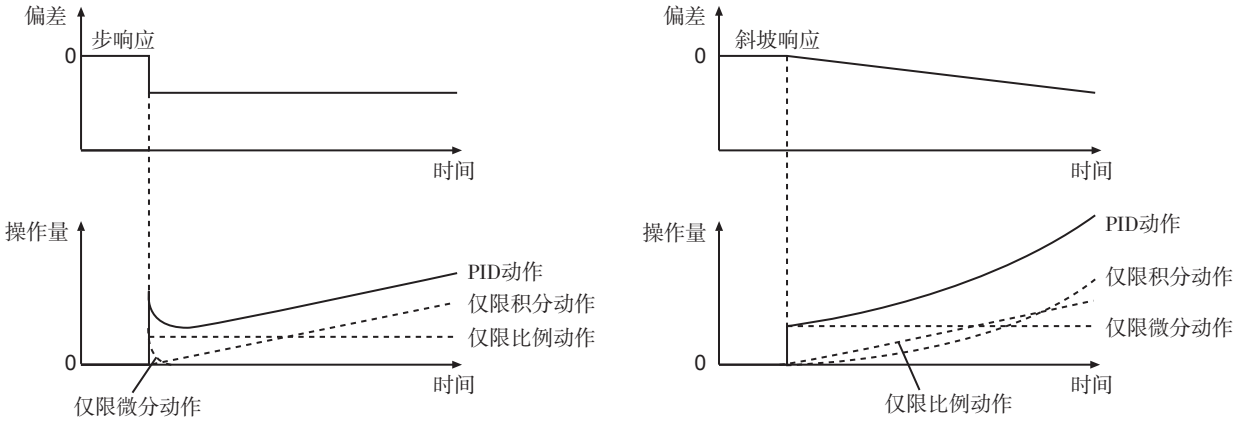
发生阶梯式偏差时，基于微分动作的操作量的变化如下所示。还表示同时使用比例动作和微分动作时的操作量的变化。



微分动作的设定值带有微分时间。这是针对斜坡式偏差，使基于微分动作的操作量与基于比例动作的操作量相等之前的时间。微分时间越长，则微分动作越强。如果微分时间较长，则应对外部干扰的速度变快，否则会发生震荡。



比例动作、积分动作、微分动作的操作量均相加后即为PID动作的操作量。相对于阶梯式偏差、斜坡式偏差的PID动作的操作量的变化如下所示。

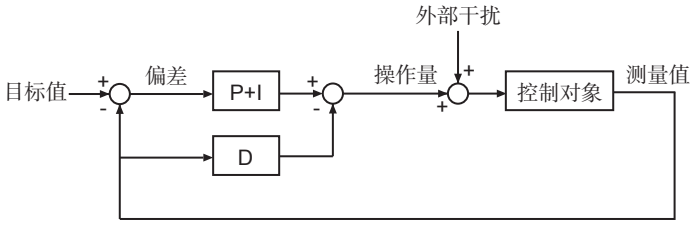


目标值滤波器型2自由度PID控制

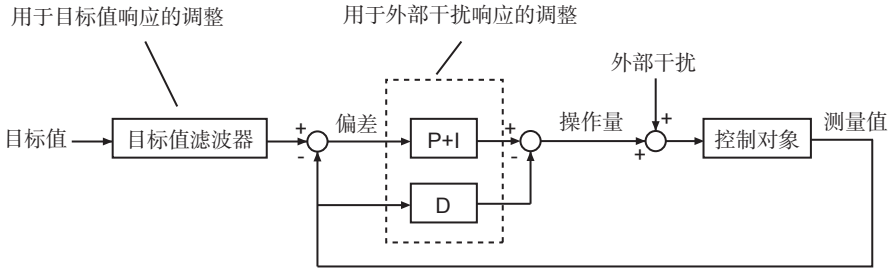
PID控制时需调整的主要设定值有比例带、积分时间、微分时间3种。三者统称为PID常数。
PID常数的值会对PID控制的以下2种性能产生影响。

- 目标值响应性能　：目标值变化时，对其进行跟踪的性能
- 外部干扰响应性能：发生外部干扰，导致测量值变化较大时，对其进行修正的性能

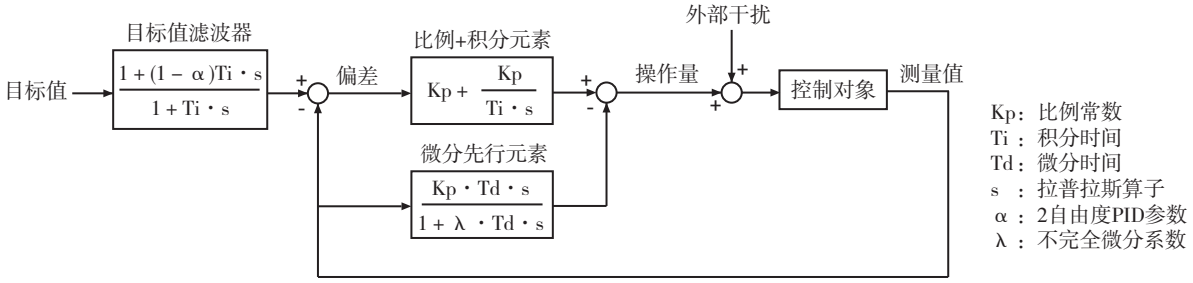
基本PID控制的框图如下所示。目标值和外部干扰的输入位置不同，因此难以查找目标值响应性能和外部干扰响应性能的最佳PID常数。即如果重视目标值响应性能来设定PID常数，则应对外部干扰的速度变慢；如果重视外部干扰响应性能来设定PID常数，则会发生过冲。



2自由度PID控制可兼顾目标值响应性能与外部干扰响应性能。2自由度是指独立具有2种调整目标值响应性能和外部干扰响应性能的参数。框图如下所示。设置新的目标值滤波器，赋予其另一调整参数。
调整PID常数，使外部干扰响应性能最优化。再通过目标值滤波器加工目标值，以使该设定也最适用于目标值响应。可独立调整PID常数值和目标值滤波器的设定值，因此可同时提高目标值响应性能和外部干扰响应性能。



本指令各模块的算术表达式如下所示。通过积分时间和2自由度PID参数α调整目标值滤波器的值(乘以目标值的系数)。α的值最好为0.65，一般不会变更。α的值越小，则目标值滤波器的影响越小。



运行开始的步骤

需使用合适的PID常数运行本指令。因此，运行开始的方法有下述2种。

● 不确定PID常数的合适值时

运行开始时进行自动调谐，计算PID常数的合适值。
在将 “StartAT”的值设为TRUE的状态下，使 “Run”的值从FALSE变为TRUE。
首先，执行自动调谐，使用计算出的PID常数，开始PID运算。

● 已明确PID常数的合适值时

将PID常数的合适值设定为 “ProportinalBand”、“IntegrationTime”、“DerivativeTime”，使 “Run”的值从FALSE变为TRUE。

“ProportinalBand”、“IntegrationTime”、“DerivativeTime”为输入输出变量。无法将常数作为输入参数。请务必定义合适的变量，将代入的值作为输入参数。

可在运行过程中变更PID常数。还可在运行过程中执行自动调谐。此时，将 “StartAT”的值设为TRUE。

控制状态与操作量

如下所示，根据控制状态，确定操作量 “MV”。

控制状态	变量值					操作量 “MV”
	手动/自动切换 “ManCtl”	执行条件 “Run”	异常结束 “Error”	手动跟踪开关 “Man TrackSw”	自动调谐执行中 “ATBusy”	
异常结束	FALSE	TRUE	TRUE	–	FALSE	异常时操作量 “ErrorMV”
自动运行中 MV跟踪			FALSE	TRUE		MV跟踪值 “MVTrackVal”
自动运行中 自动调谐执行中				FALSE	TRUE	重复操作量限位上限制和操作量限位下限制
自动运行中 不执行自动调谐					FALSE	通过该时刻的PID常数计算出的值
停止执行指令		FALSE	–	–		停止时操作量 “StopMV”
手动运行中	TRUE	–				手动操作量 “ManMV”

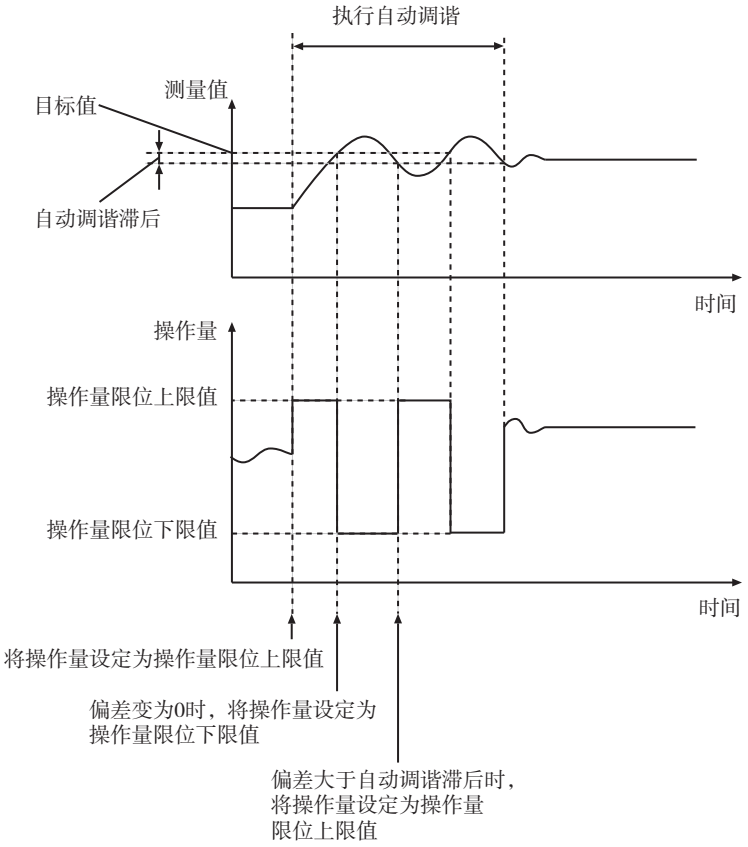
自动调谐

由于极少调整2自由度PID参数 α，因此本指令主要调整的设定值为PID常数。

本指令带有自动调谐PID常数的功能。使用极限环法作为自动调谐的方法。极限环法是指强制将操作量暂时设为操作量限位上限值和操作量限位下限值，根据此时的测量值的变化情况计算 PID 常数最佳值的方法。

目标值>测量值时如果有自动调谐执行指示，则首先将操作量设定为操作量限位上限值。然后，偏差变为0时，将操作量设定为操作量限位下限值。偏差大于自动调谐滞后的值时，将操作量设定为操作量限位上限值。重复2次上述操作，计算PID常数的最佳值。

目标值<测量值时如果有自动调谐执行指示，则首先将操作量设定为操作量限位下限值。然后，按照与上述相同的步骤计算PID常数的最佳值。



“Run”的值为TRUE且处于PID运算状态时，如果 “StartAT”的值由FALSE变为TRUE，则在运行过程中执行自动调谐。在 “StartAT”的值为TRUE的状态下，如果 “Run”的值由FALSE变为TRUE，则运行开始时执行。

如果自动调谐正常结束，则立即反映计算出的PID常数的最佳值。

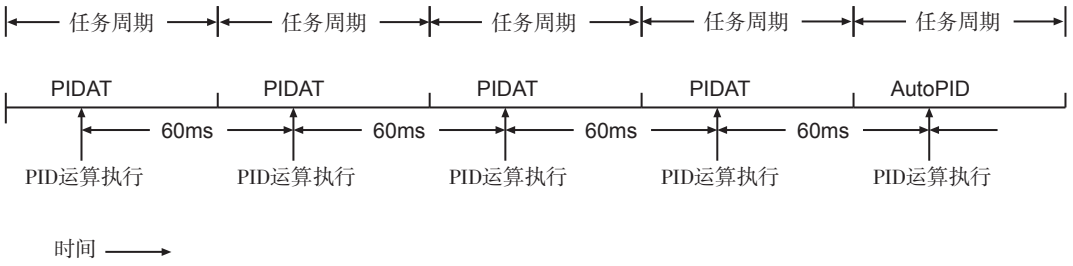
自动调谐过程中(“ATBusy”=TRUE)，如果将 “StartAT”的值设为FALSE，则中止自动调谐。此时，通过开始自动调谐之前的PID常数重新开始PID运算。

PID运算的执行时间

定期重复执行PID运算。在用户程序中执行本指令时，执行PID运算。
执行上次PID运算后的经过时间小于采样周期 “SampTime”时不执行。
执行上次PID运算后的经过时间大于 “SampTime”时执行，剩余的时间(上次执行后的经过时间–“SampTime”)转入下一次。详情请参阅下图。

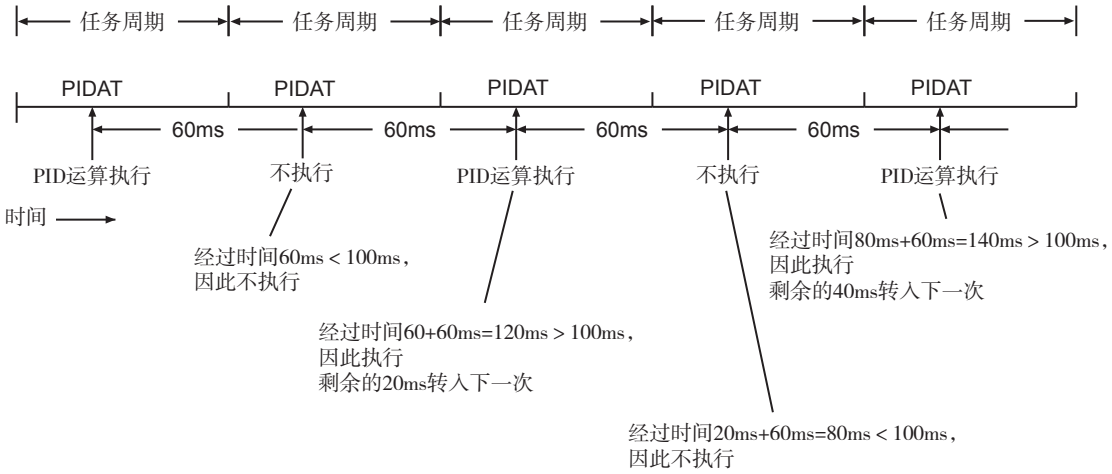
任务周期=60ms　“SampTime” < 60ms时

任务周期≥“SampTime”，因此1个任务周期必定执行1次。



任务周期=60ms　“SampTime”=100ms时

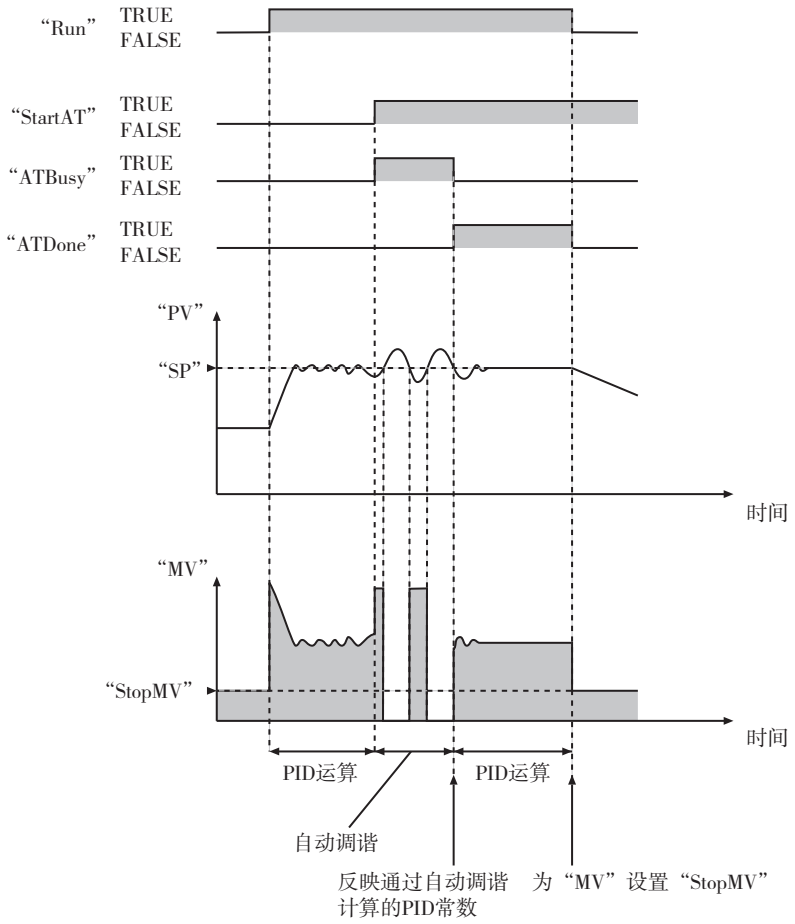
任务周期 < “SampTime”，因此可能不执行。



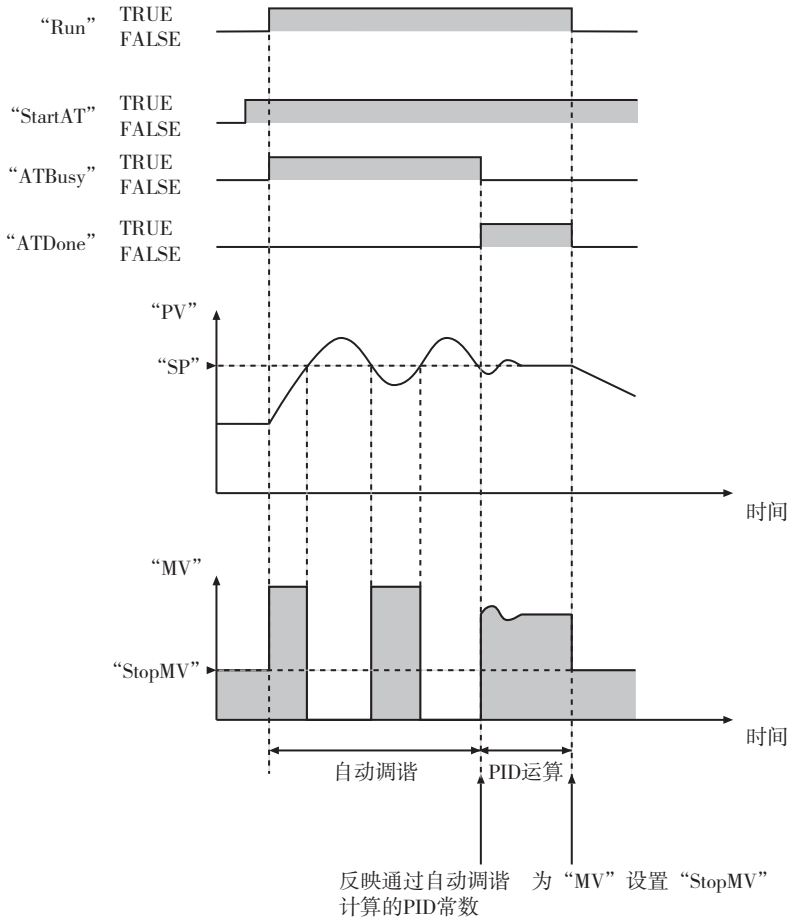
时序图

表示有数个变量时各变量的时序图。

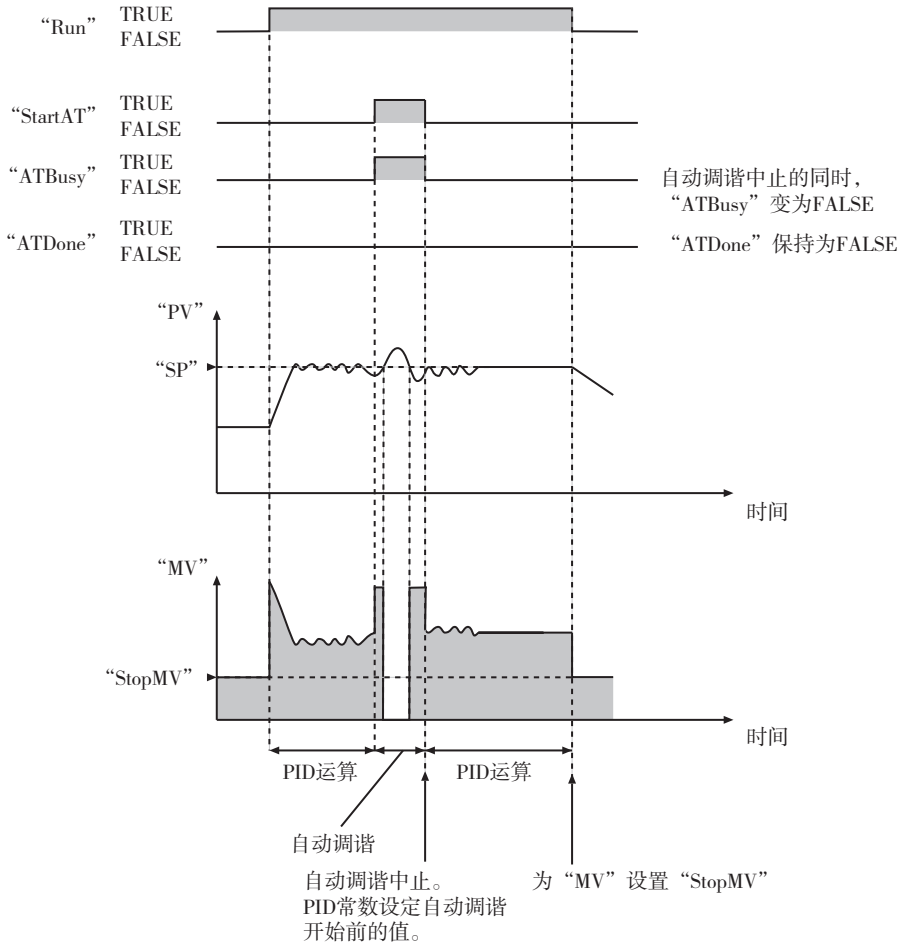
● 在自动运行过程中执行自动调谐时



● 在执行本指令的开头执行自动调谐时



● 中止自动调谐时

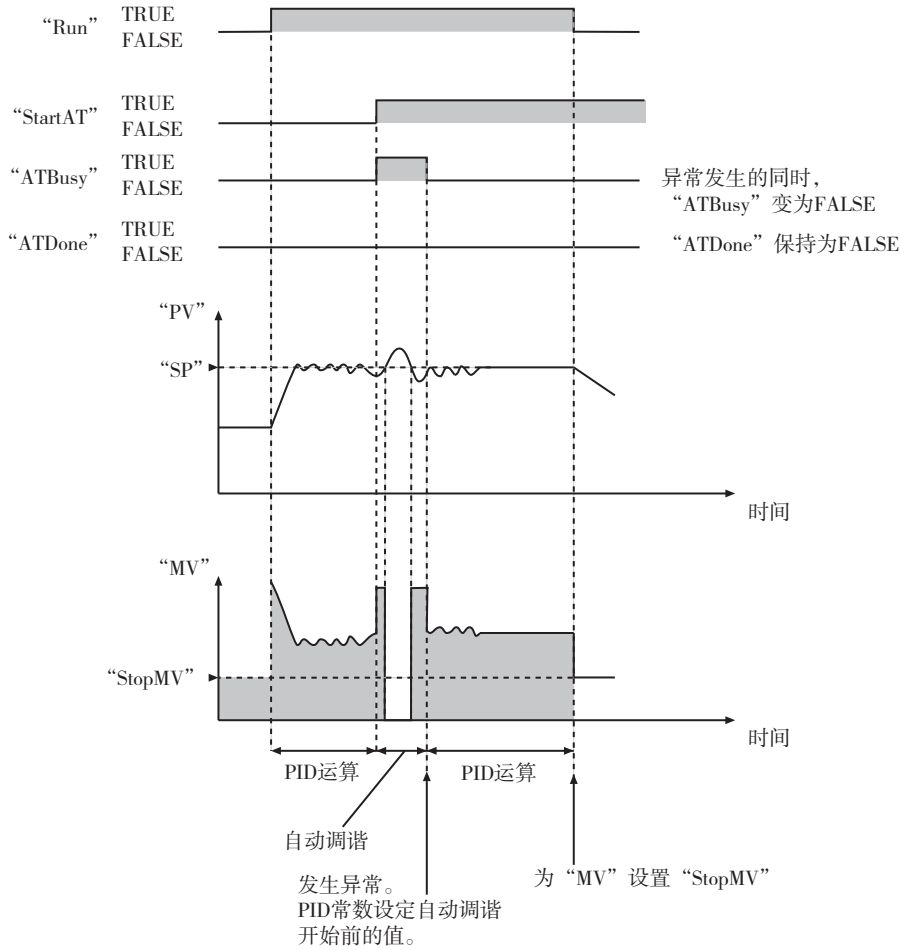


● 在自动调谐过程中发生自动调谐异常时

以下情况时会发生自动调谐异常，中止自动调谐。

- 操作量等于操作量限位上限值且偏差变为0之前的时间超过19999s。
- 操作量等于操作量限位下限值且偏差大于 “ATHystrs”之前的时间超过19999s。

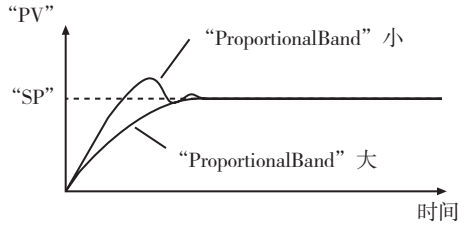
如果中止自动调谐，则通过开始自动调谐之前的PID常数重新开始PID运算。



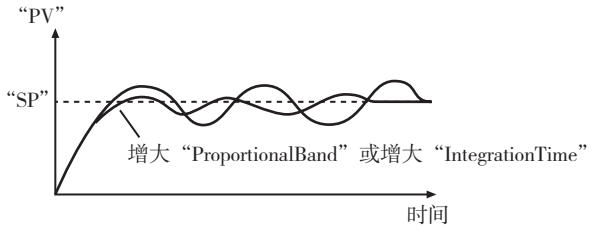
参考

PID常数的调整方法

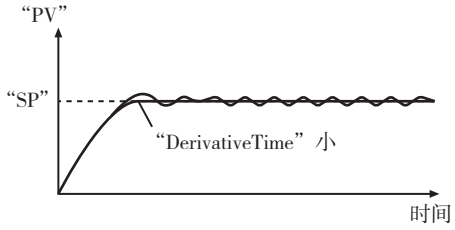
- 即使在控制对象稳定前需些许时间也不希望发生震荡时，应增大 "ProportionalBand"的值。反之，即使发生些许震荡也希望尽快稳定时，应减小 "ProportionalBand"的值。



- 重复震荡时，增大 “ProportionalBand”或 “IntegrationTime”。



- 在短周期内重复震荡时，减小 “DerivativeTime”。



温度调节时的PID常数初始值

将本指令用于温度调节时，PID常数的初始值请参考下述值。请使用除PID常数以外的各变量的初始值。

变量	参考初始值(*)
“ProportinalBand”	10%FS
“IntegrationTime”	233s
“DerivativeTime”	40s

*执行自动调谐后，请遵从该结果。

使用注意事项

- “PV”和 “SP”的值需大于 “RngLowLmt”的值且小于 “RngUpLmt”的值。因此，如下所示，请符合这些变量的单位系统。

单位系统	“PV”、“SP”的值	“RngLowLmt”、“RngUpLmt”的值
%FS	“PV” =(测量值的物理量−MIN)/(MAX−MIN)×100 “SP” =(目标值的物理量−MIN)/(MAX−MIN)×100 (*)	“RngLowLmt” =0 “RngUpLmt” =100
物理量	“PV” =测量值的物理量 “SP” =目标值的物理量	“RngLowLmt” =MIN “RngUpLmt” =MAX(*)

*MAX：输入范围上限的物理量、MIN：输入范围下限的物理量

- 如下所示，根据控制状态，确认可否变更各变量。

变量	控制状态		
	停止执行指令(*1)	运行中 (不执行自动调谐) (*2)	运行中 (自动调谐执行中) (*3)
“Run”	○	○	○
“ManCtl”	○	○	○
“StartAT”	○	○	○
“PV”	○	○	○
“SP”	○	○	×
“MVLowLmt”	○	○	×
“MVUpLmt”	○	○	×

变量	控制状态		
	停止执行指令(*1)	运行中 (不执行自动调谐) (*2)	运行中 (自动调谐执行中) (*3)
“ManResetVal”	○	○	×
“MVTrackSw”	○	○	×
“MVTrackVal”	○	○	×
“StopMV”	○	○	○
“ErrorMV”	○	○	○
“Alpha”	○	○	×
“ATCalcGain”	○	○	×
“ATHystrs”	○	○	×
“SampTime”	○	×	×
“RngLowLmt”	○	×	×
“RngUpLmt”	○	×	×
“DirOpr”	○	×	×
“ProportinalBand”	○	○	×
“IntegrationTime”	○	○	×
“DerivativeTime”	○	○	×
“ManMV”	○	○	○

*1 “ManCtl”=TRUE、“Run”=FALSE、“Error”=TRUE或 “MVTrackSw”=TRUE
*2 “MacCtl”=FALSE、“Run”=TRUE、“Error”=FALSE、“MVTrackSw”=FALSE且 “ATBusy”=FALSE
*3 “MacCtl”=FALSE、“Run”=TRUE、“Error”=FALSE、“MVTrackSw”=FALSE且 “ATBusy”=TRUE

- “SampTime”小于100ns时舍去。
- “ManCtl”的值为TRUE时，如果将 “StartAT”的值设为TRUE，则接下来 “ManCtl”的值变为FALSE时，开始自动调谐。
- 如果 “ErrorMV”的值超过有效范围(–320 ~ 320)，则异常时的 “MV”值变为0。
- 自动调谐过程中如果将 “ManCtl”的值设为TRUE，则中止自动调谐。
- 即使发生自动调谐异常，“Error”的值也不变为TRUE。
- 以下情况时会发生异常。“Error”变为TRUE，将异常ID代入 “ErrorID”。“ATDone”、“ATBusy”为FALSE。如果 “ManCtl”和 “Run”的值均为FALSE，则为 “MV”设置 “ErrorMV”的值。“ErrorMV”的值超过有效范围时，“MV”的值变为0。

异常的内容	“ErrorID”的值
任一输入变量超过有效范围	16#0400
“RngLowLmt” ≥ “RngUpLmt”	16#0401
“MVLowLmt” ≥ “MVUpLmt”	

- 如需在上述以外的条件下使异常停止，则请在发生异常时设置将 “Run”的值设为FALSE的处理。
- 因 “PV”或 “SP”的值超过有效范围而发生异常后，即使将值变更到有效范围内，仍保持5秒的异常状态。即5秒内 “Error”的值保持为FALSE。
- 解除异常后，如果 “Run”的值为 TRUE，则自动重新开始 PID 运算。“Run”和 “StartAT”的值均为TRUE时，自动重新开始自动调谐。
- 按采样周期判定有无异常。

示例程序

使用PIDAT指令，进行温度控制。将PID AT指令的操作量转换为分时比例输出，输出至加热器。此处已使用定时器指令，以转换为分时比例输出。转换为分时比例输出时，如果使用TimeProportionalOut指令，则请参阅 “TimeProportionalOut指令(P.2-719)”的示例程序。

规格

通过下表所示规格，进行温度控制。

项目	规格
输入类型	K型热电偶
输入单元	绝缘型高分辨率多功能输入单元 CJ1W-PH41U
输出单元	晶体管输出单元 CJ1W-OD212
目标温度	90℃
PID运算的采样周期	100ms
输出控制周期	1s

构成・设定

输入单元CJ1W-PH41U的设定如下所示。

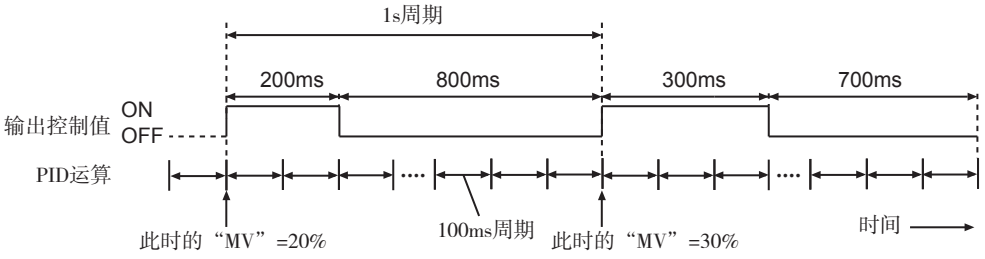
项目名称	设定值
Input1:Input signal type	K(1)

I/O映射设定如下所示。

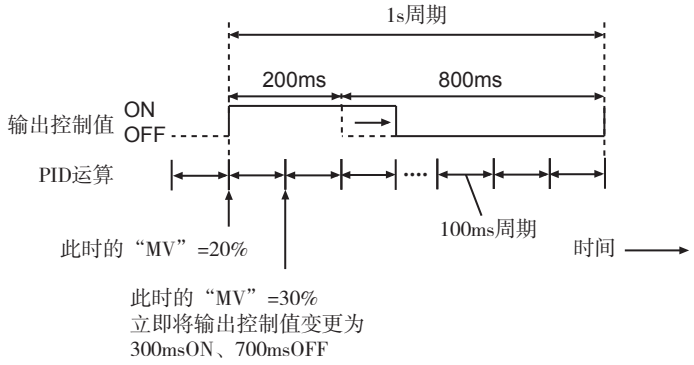
单元	端口	说明	变量
CJ1W-PH41U	Ch1_AIInPV	输入No.1测量值(INT型)	AI1
CJ1W-OD212	Ch1_Out00	输出CH1触点00	DO1

处理内容

- 获取PIDAT指令的操作量 “MV”，控制至温控器的输出。至温控器的输出可分为ON或OFF的2种。
- PIDAT指令的采样周期 “InitSetParams.SampTime”的值为100ms。任务周期远短于100ms。因此，在约100ms的周期内刷新 “MV”的值。
- 输出控制周期为1s。通过分时比例输出控制该期间的输出控制值的ON时间和OFF时间。例如，获取的“MV”的值为20%时，在200ms内，至温控器的输出设为ON；在接下来的800ms内，设为OFF。在1s周期内重复上述操作。



- 最新的 “MV”值小于确定了输出控制值时的 “MV”值时，输出控制值不变。最新的 “MV”值大于确定了输出控制值时的 “MV”值时，立即将该值反映为输出控制值。例如，假设 “MV”值为20%时确定输出控制值 (ON200ms、OFF800ms)。100ms 后，新的 “MV”值为 30% 时，输出控制值立即变为ON300ms、OFF700ms。



- 执行自动调谐，“MV”的值变为100%时，与控制周期无关，均立即将输出控制值设为ON。

应用程序

全局变量的定义

全局变量

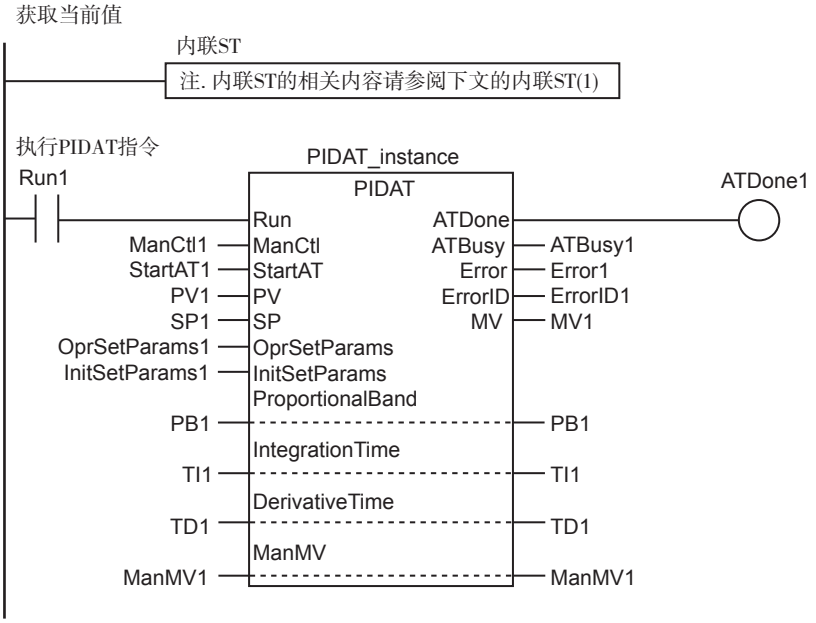
名称	数据类型	分配对象*1	注释
AI1	INT	IOBus://rack#0/slot#0/Ch1_AIInPV	输入No.1测量值(INT型)
DO1	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out00	输出CH1触点00

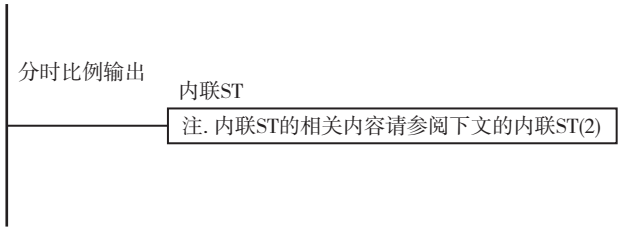
*1 将CJ1W-PH41U安装至机架编号0的插槽编号0，将CJ1W-OD212安装至机架编号0的插槽编号1时的分配对象。

(注) 通过I/O映射设定，自动生成分配至各单元端口的全局变量。

LD

名称	数据类型	初始值	保持	注释
Run1	BOOL	FALSE	□	执行条件
ManCtl1	BOOL	FALSE	□	手动/自动切换
StartAT1	BOOL	FALSE	□	自动调谐执行条件
PV1	REAL	0.0	□	测量值(当前值)
SP1	REAL	90	□	目标值
OprSetParams1	_sOPR_SET_PARAMS	(MVLowLmt:=0.0，MVUpLmt:=100.0，ManResetVal:=0.0，MVTrackSw:=FALSE，MVTrackVal:=0.0，StopMV:=0.0，ErrorMV:=0.0，Alpha:=0.65，ATCalcGain:=1.0，ATHystrs:=0.2)	□	运行中设定参数
InitSetParams1	_sINIT_SET_PARAMS	(SampTime:=T#100ms，RngLowLmt:=0.0，RngUpLmt:=1000.0，DirOpr:=FALSE)	□	初始设定参数
PB1	REAL	10	☑	比例带
TI1	TIME	T#0S	☑	积分时间
TD1	TIME	T#0S	☑	微分时间
ManMV1	REAL	0.0	□	手动操作量
ATDone1	BOOL	FALSE	□	自动调谐正常结束
ATBusy1	BOOL	FALSE	□	自动调谐执行中
Error1	BOOL	FALSE	□	异常
ErrorID1	WORD	16#0	□	异常ID
MV1	REAL	0.0	□	操作量
PulseOnTime	TIME	T#0s	□	控制输出ON时间
PulseCycTime	TIME	T#1s	□	控制周期
ResetPulse	BOOL	FALSE	□	定时器复位
PIDAT_instance	PIDAT		□	
TOF_instance	TOF		□	
TON_instance	TON		□	





● 内联ST (1)的内容

```
PV1:=INT_TO_REAL(AI1)/REAL#10.0;    // 将当前值AI1转换为实数
                                    // CJ1W-PH41U的输出为测量温度的10倍，因此除以10.0
```

● 内联ST (2)的内容

```
PulseOnTime:=MULTIME(PulseCycTime, MV1/REAL#100.0);          // 输出控制值ON时间计算
TOF_instance(In:=BOOL#FALSE, PT:=PulseOnTime, Q=>DO1);       // 通过TOF指令切换ON/OFF
TON_instance(In:=BOOL#TRUE, PT:=PulseCycTime, Q=>ResetPulse); // 通过TON指令测量定时器复位时间
IF (ResetPulse=BOOL#TRUE) THEN                               // 定时器复位
    TOF_instance(In:=BOOL#TRUE);
    TON_instance(In:=BOOL#FALSE);
END_IF;
IF ( (ATBusy1=BOOL#TRUE) & (MV1=REAL#100.0) ) THEN           // 通过自动调谐，MV1=100%
    DO1:=BOOL#TRUE;                                          // 如果是这种情况，则立即将输出控制值设为ON
END_IF;
```

ST

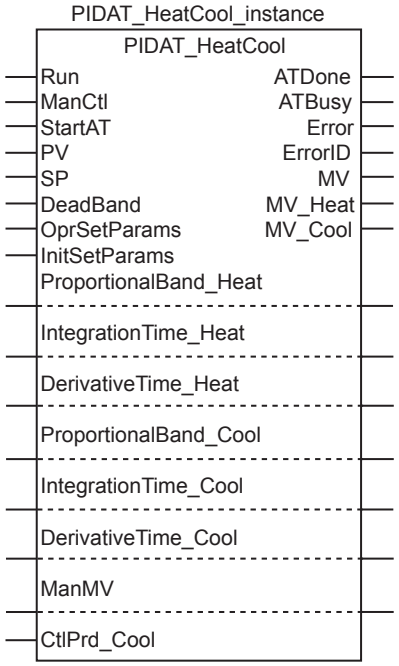
名称	数据类型	初始值	保持	注释
Run1	BOOL	FALSE	☐	执行条件
ManCtl1	BOOL	FALSE	☐	手动/自动切换
StartAT1	BOOL	FALSE	☐	自动调谐执行条件
PV1	REAL	0.0	☐	测量值(当前值)
SP1	REAL	90	☐	目标值
OprSetParams1	_sOPR_SET_PARAMS	(MVLowLmt:=0.0，MVUpLmt:=100.0，ManResetVal:=0.0，MVTrackSw:=FALSE，MVTrackVal:=0.0，StopMV:=0.0，ErrorMV:=0.0，Alpha:=0.65，ATCalcGain:=1.0，ATHystrs:=0.2)	☐	运行中设定参数
InitSetParams1	_sINIT_SET_PARAMS	(SampTime:=T#100ms，RngLowLmt:=0.0，RngUpLmt:=1000.0, DirOpr:=FALSE)	☐	初始设定参数
PB1	REAL	10	☑	比例带
TI1	TIME	T#0S	☑	积分时间
TD1	TIME	T#0S	☑	微分时间
ManMV1	REAL	0.0	☐	手动操作量
ATDone1	BOOL	FALSE	☐	自动调谐正常结束
ATBusy1	BOOL	FALSE	☐	自动调谐执行中
Error1	BOOL	FALSE	☐	异常
ErrorID1	WORD	16#0	☐	异常ID
MV1	REAL	0.0	☐	操作量
PulseOnTime	TIME	T#0S	☐	控制输出ON时间
PulseCycTime	TIME	T#1S	☐	控制周期
ResetPulse	BOOL	FALSE	☐	定时器复位
PIDAT_instance	PIDAT		☐	
TOF_instance	TOF		☐	
TON_instance	TON		☐	

```
// 将当前值AI1转换为实数
PV1:=INT_TO_REAL(AI1)/REAL#10.0;          // CJ1W-PH41U的输出为测量温度的10倍，因此除以10.0

// 执行PIDAT指令
PIDAT_instance(
    Run              :=Run1,
    ManCtl           :=ManCtl1,
    StartAT          :=StartAT1,
    PV               :=PV1,
    SP               :=SP1,
    OprSetParams     :=OprSetParams1,
    InitSetParams    :=InitSetParams1,
    ProportionalBand :=PB1,
    IntegrationTime  :=TI1,
    DerivativeTime   :=TD1,
    ManMV            :=ManMV1,
    ATDone           =>ATDone1,
    ATBusy           =>ATBusy1,
    Error            =>Error1,
    ErrorID          =>ErrorID1,
    MV               =>MV1);
```

```
// 分时比例输出
PulseOnTime:=MULTIME(PulseCycTime, MV1/REAL#100.0);              // 输出控制值ON时间计算
TOF_instance(In:=BOOL#FALSE, PT:=PulseOnTime, Q=>DO1);           // 通过TOF指令切换ON/OFF
TON_instance(In:=BOOL#TRUE, PT:=PulseCycTime, Q=>ResetPulse);    // 通过TON指令测量定时器复位时间
IF (ResetPulse=BOOL#TRUE) THEN                                   // 定时器复位
    TOF_instance(In:=BOOL#TRUE);
    TON_instance(In:=BOOL#FALSE);
END_IF;
IF ( (ATBusy1=BOOL#TRUE) & (MV1=REAL#100.0) ) THEN               // 通过自动调谐，MV1=100%
    DO1:=BOOL#TRUE;                                              // 如果是这种情况，则立即将输出控制值设为ON
END_IF;
```

PIDAT_HeatCool

执行带自动调谐的加热冷却PID运算(目标值滤波器型2自由度PID控制)。

指令	名称	FB/FUN	图形表现	ST表现
PIDAT_HeatCool	带自动调谐的加热冷却PID运算	FB		PIDAT_HeatCool_instance(Run, ManCtl, StartAT, PV, SP, DeadBand, OprSetParams, InitSetParams, ProportionalBand_Heat, IntegrationTime_Heat, DerivativeTime_Heat, ProportionalBand_Cool, IntegrationTime_Cool, DerivativeTime_Cool, ManMV, CtlPrd_Cool, ATDone, ATBusy, Error, ErrorID, MV, MV_Heat, MV_Cool);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Run	执行条件	输入	TRUE：执行 FALSE：停止	遵从数据类型	–	FALSE
ManCtl	手动/自动切换		TRUE：手动运行 FALSE：自动运行			
StartAT	自动调谐执行条件		TRUE：执行 FALSE：中止			
PV	测量值(当前值)		测量值(当前值)	*1		0
SP	目标值		目标值			
DeadBand	死区		设定死区或重叠区	–320.0 ~ 320.0	%	
OprSet Params	运行中设定参数		运行中的各种设定参数	–	–	–
InitSet Params	初始设定参数		初始设定参数			
CtlPrd _Cool	冷却操作的控制周期		分时比例输出 “MV_Cool”时的控制周期	T#0.1s ~ T#100s		T#20s
Proportional Band_Heat	加热操作的比例带	输入输出	加热操作的比例带	0.01 ~ 1000.00	%FS	–
IntegrationTime_Heat	加热操作的积分时间		加热操作的积分时间 值越大则积分效应越小。 为0时，无积分动作。	T#0.0000s ~ T#10000.0000s*2	s	
Derivative Time_Heat	加热操作的微分时间		加热操作的微分时间 值越大则微分效应越大。 为0时，无微分动作。	T#0.0000s ~ T#10000.0000s*2		
Proportional Band_Cool	冷却操作的比例带		冷却操作的比例带	0.01 ~ 1000.00	%FS	
IntegrationTime_Cool	冷却操作的积分时间		冷却操作的积分时间 值越大则积分效应越小。 为0时，无积分动作。	T#0.0000s ~ T#10000.0000s*2	s	
Derivative Time_Cool	冷却操作的微分时间		冷却操作的微分时间 值越大则微分效应越大。 为0时，无微分动作。	T#0.0000s ~ T#10000.0000s*2		
ManMV	手动操作量		手动操作量	–320 ~ 320	%	
ATDone	自动调谐正常结束	输出	TRUE：正常结束 FALSE：*3	遵从数据类型	–	–
ATBusy	自动调谐执行中		TRUE：执行中 FALSE：非执行中			
MV	操作量		操作量		%	
MV_Heat	加热操作的操作量		加热操作的操作量	0 ~ 320		
MV_Cool	冷却操作的操作量		冷却操作的操作量	0 ~ 320		

*1 输入范围下限 “InitSetParams.RngLowLmt”的值 ~ 输入范围上限 “InitSetParams.RngUpLmt”的值。
*2 小于0.0001s时舍去。
*3 异常结束或不执行自动调谐即处于PID运算状态或未处于PID运算状态。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Run	○																			
ManCtl	○																			
StartAT	○																			
PV														○						
SP														○						
DeadBand														○						
OprSet Params	结构体_sOPR_SET_PARAMS　详情参阅功能说明																			
InitSet Params	结构体_sINIT_SET_PARAMS　详情参阅功能说明																			
CtlPrd _Cool																○				
Proportional Band_Heat														○						
IntegrationTime_Heat																○				
Derivative Time_Heat																○				
Proportional Band_Cool														○						
IntegrationTime_Cool																○				
Derivative Time_Cool																○				
ManMV														○						
ATDone	○																			
ATBusy	○																			
MV														○						
MV_Heat														○						
MV_Cool														○						

功能

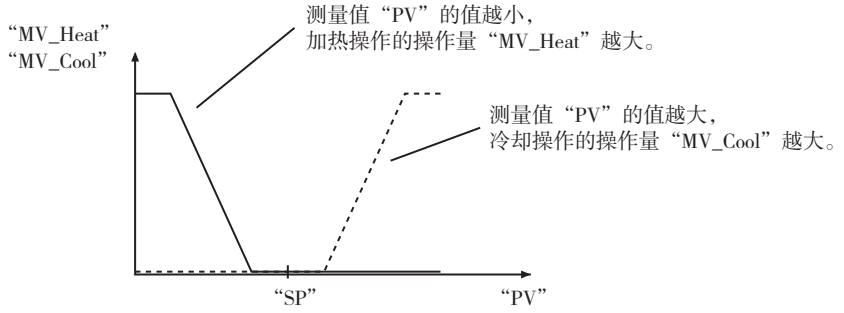
进行加热冷却PID运算，控制温控器等的操作量。
如果执行条件 “Run”的值变为TRUE，则开始加热冷却PID运算。此后，“Run”的值为TRUE的期间，定期重复进行测量值 “PV”的获取→加热冷却PID运算→加热操作的操作量 “MV_Heat”和冷却操作的操作量 “MV_Cool”的输出。
如果 “Run”的值变为FALSE，则结束加热冷却PID运算。

带自动调谐功能，可自动计算加热操作及冷却操作PID常数的各自最佳值。
如果自动调谐执行条件 “StartAT”的值变为TRUE，则执行加热操作及冷却操作PID常数的自动调谐。

PIDAT_HeatCool指令与PIDAT指令的不同

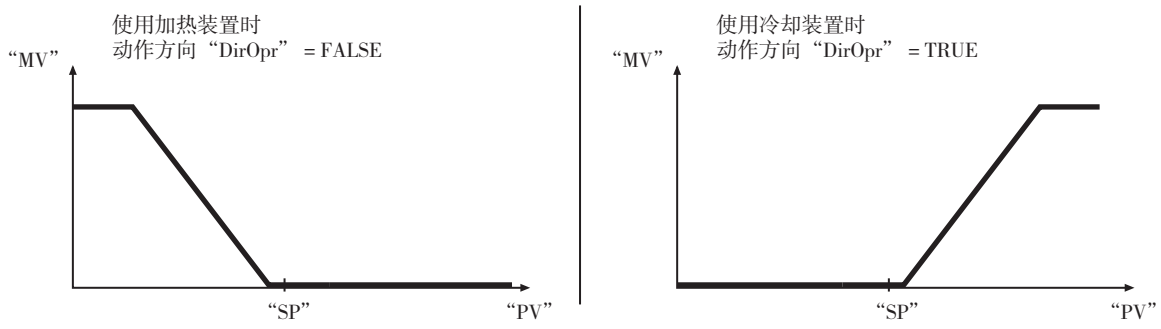
● PIDAT_HeatCool指令

PIDAT_HeatCool指令是使用加热装置和冷却装置进行温度调节时使用的指令。因此，将输出加热操作的操作量 “MV_Heat”和冷却操作的操作量 “MV_Cool”2种操作量。此外，利用自动调谐功能，分别计算加热操作PID常数的最佳值和冷却操作PID常数的最佳值。



● PIDAT指令

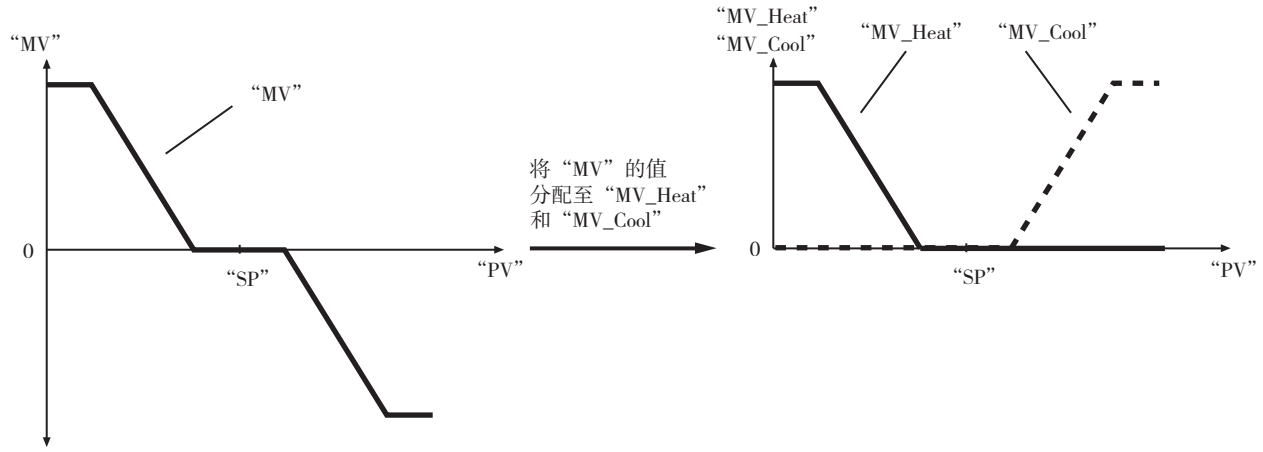
PIDAT指令是使用加热装置或冷却装置进行温度调节时使用的指令。因此，输出的操作量 “MV”只有1种。此外，带动作方向参数 “DirOpr”，决定对加热装置输出操作量或对冷却装置输出操作量。PIDAT_HeatCool指令不使用 “DirOpr”。



操作量“MV”和加热装置的操作量“MV_Heat”、冷却装置的操作量“MV_Cool”

与 PID 指令相同，使用加热装置或冷却装置进行温度调节时的操作量为 “MV” 。本指令也首先使用与 PIDAT 指令相同的方法计算 “MV”。然后，将 “MV” 分配至加热装置的操作量和冷却装置的操作量为 “MV_Heat” 和 “MV_Cool”。

将 “MV” 的值分配至 “MV_Heat” 和 “MV_Cool” 的概念图如下所示。“MV_Cool” 的值为 “MV” 负值的绝对值。



上图为概念图。“MV_Heat” 、“MV_Cool” 的实际值与 “MV” 值的正数部分、负数部分不完全一致。“MV_Heat”、“MV_Cool” 的值根据 “MV” 的值，使用特殊计算公式进行计算。

结构体的规格

运行中设定参数 “OprSetParams”的数据类型为结构体_sOPR_SET_PARAMS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
OprSetParams	运行中设定参数	运行中的各种设定参数	_sOPR_SET_PARAMS	–	–	–
MVLowLmt	操作量限位下限值	对 “MV_Heat”、“MV_Cool”进行限位控制的下限值	REAL	–320 ~ 320*1	%	–100
MVUpLmt	操作量限位上限值	对 “MV_Heat”、“MV_Cool”进行限位控制的上限值	REAL			100
ManResetVal	手动复位值	不使用	REAL	–320 ~ 320		0
MVTrackSw	MV跟踪开关	MV跟踪开关 TRUE ：ON FALSE：OFF	BOOL	遵从数据类型	–	FALSE
MVTrackVal	MV跟踪值	进行MV跟踪时 “MV”的值	REAL	–320 ~ 320	%	0
StopMV	停止时操作量	停止执行指令时 “MV”的值	REAL			
ErrorMV	异常时操作量	发生异常时 “MV”的值	REAL			
Alpha	2自由度PID参数α	目标值滤波器系数α 为0时，目标值滤波器无效	REAL	0.00 ~ 1.00	–	0.65
ATCalcGain	自动调谐计算增益	自动调谐结果的调整系数。 值越大则越重视稳定性，越小则越重视快速响应性。	REAL	0.1 ~ 10.0		0.8
ATHystrs	自动调谐滞后	极限环滞后的大小	REAL	0.01 ~ 10.0	%FS	0.05

*1 “MVLowLmt” < “MVUpLmt”

初始设定参数 “InitSetParams”的数据类型为结构体_sINIT_SET_PARAMS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
InitSetParams	初始设定参数	初始设定参数	_sINIT_SET_PARAMS	–	–	–
SampTime	采样周期	进行PID运算的周期	TIME	T#0.0001s ~ T#100.0000s	s	T#0.05s
RngLowLmt	输入范围下限	“PV”和 “SP”的下限值	REAL	–32000 ~ 32000*1	–	0
RngUpLmt	输入范围上限	“PV”和 “SP”的上限值	REAL			100
DirOpr	动作方向	不使用	BOOL	遵从数据类型		FALSE

*1 “RngLowLmt” < “RngUpLmt”

各变量的含义

以下对本指令各变量的含义作详细说明。

● “Run”(执行条件)

本指令的执行条件。
值为TRUE的期间，进行加热冷却PID运算。如果值变为FALSE，则停止加热冷却PID运算。

● “ManCtl”(手动/自动切换)

本指令有手动运行和自动运行2种模式。
根据 “ManCtl”的值，确定在哪种模式下运行。

“ManCtl”的值	运行模式	“MV”的值
TRUE	手册	“ManMV”的值 (不进行加热冷却PID运算)
FALSE	自动	根据加热冷却PID运算计算出的数值

● “StartAT”(自动调谐执行条件)

PID常数自动调谐的执行条件。
在 “StartAT”的值为TRUE的状态下，如果 “Run”的值由FALSE变为TRUE，则运行开始时执行自动调谐。
“Run”的值为TRUE且处于加热冷却PID运算状态时，如果 “StartAT”的值由FALSE变为TRUE，则在运行过程中开始自动调谐。
无论哪种情况，如果在自动调谐过程中 “StartAT”的值由TRUE变为FALSE，则中止自动调谐。
自动调谐功能的详情将于下文阐述。

● “PV”(测量值)

控制对象的测量值。

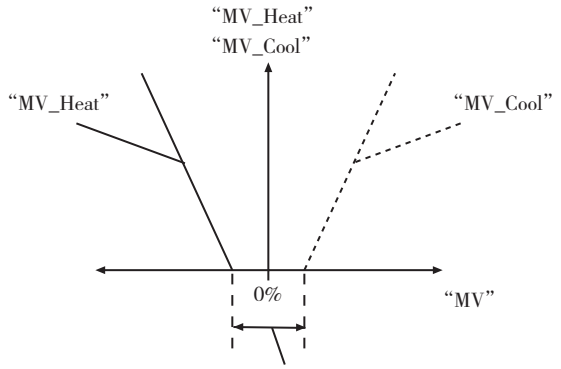
● “SP”(目标值)

控制对象的目标值。

- **“DeadBand”(死区)**

“DeadBand”确定如何将 “MV”的值分配至 “MV_Heat”和 “MV_Cool” 。“DeadBand”的值以“MV”=0为中心，表示不执行加热操作和冷却操作的 “MV”值的范围。
“MV”的值与 “MV_Heat”、“MV_Cool”值的关系如下所述。

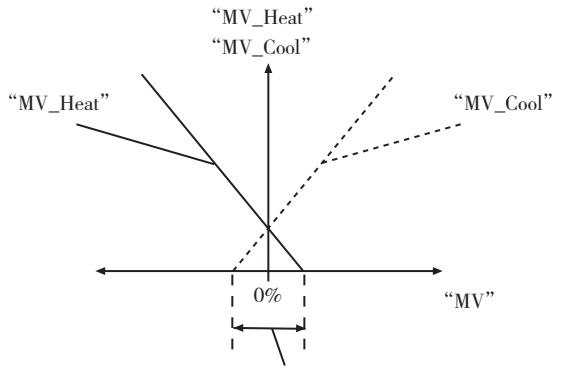
“MV”的值	“MV_Heat”的值	“MV_Cool”的值
死区内	0	0
大于死区	0	正值。“MV”的值越大则“MV_Cool”的值越大。
小于死区	正值。“MV”的值越小则“MV_Heat”的值越大。	0



“MV”的值在该范围内时，
“MV_Heat”和“MV_Cool”为0%。

也可给 “DeadBand”设定负值。“DeadBand”为负值时，若 “MV”的值在死区内，则加热操作和冷却操作均执行。“DeadBand”为负值时，“MV”的值与 “MV_Heat”、“MV_Cool”的关系如下所述。

“MV”的值	“MV_Heat”的值	“MV_Cool”的值
死区内	正值。“MV”的值越小则“MV_Heat”的值越大。	正值。“MV”的值越大则“MV_Cool”的值越大。
大于死区	0	正值。“MV”的值越大则“MV_Cool”的值越大。
小于死区	正值。“MV”的值越小则“MV_Heat”的值越大。	0



“MV”的值在该范围内时，
“MV_Heat”和“MV_Cool”两者的值为正。

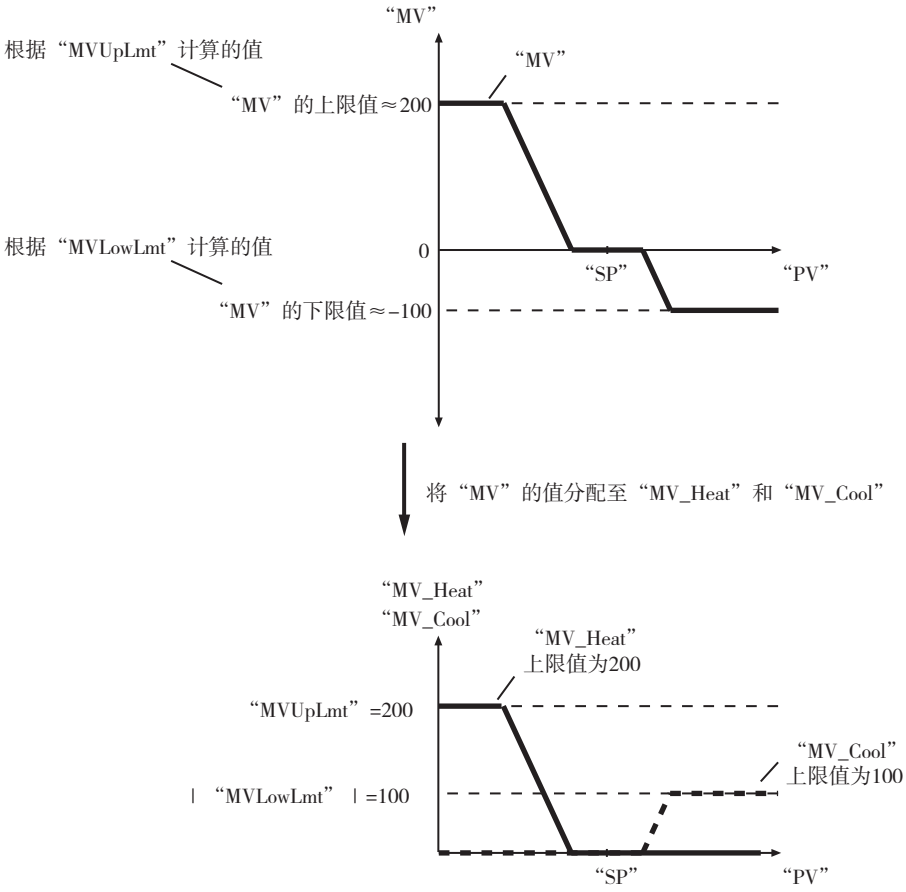
- **“MVLowLmt”(操作量限位下限值)、“MVUpLmt”(操作量限位上限值)**

可限位控制 “MV_Heat”、“MV_Cool”的值。确定 “MV_Heat”、“MV_Cool”的下限值和上限值的为 “MVLowLmt”和 “MVUpLmt”。“MV_Heat”和 “MV_Cool”的值按照以下步骤进行计算。

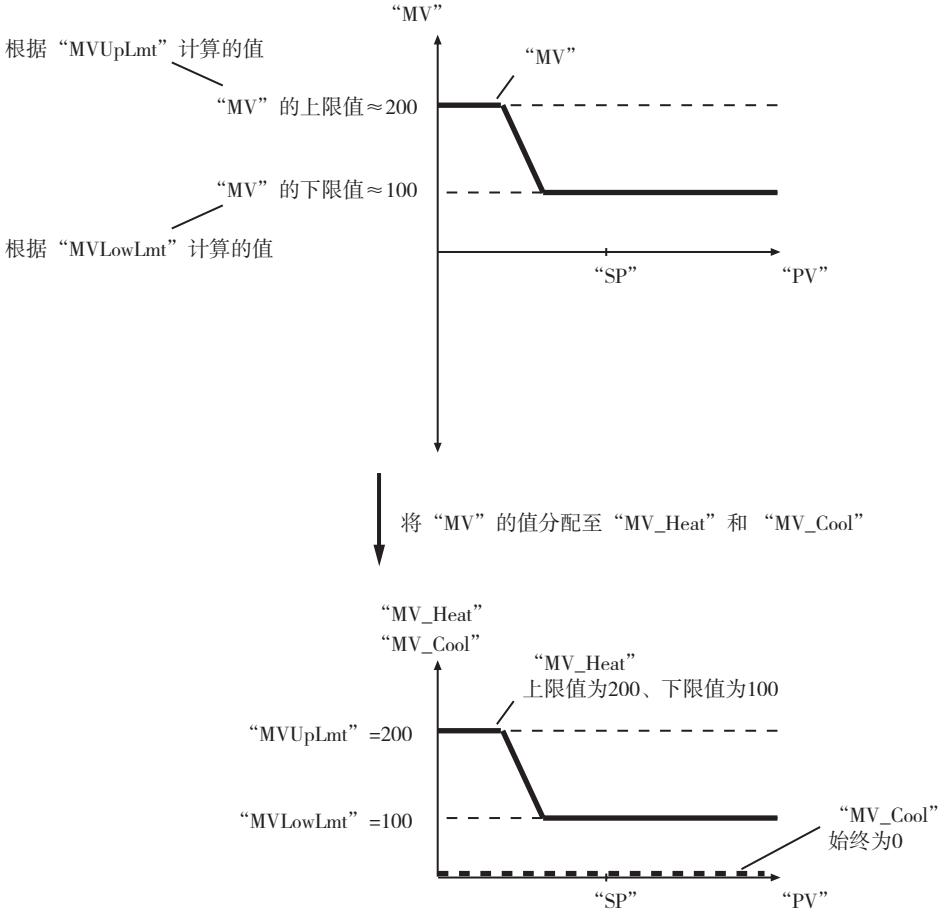
1 **“MV”通过加热冷却PID运算进行计算。“MV”的上限值和下限值根据 “MVLowLmt”和“MVU p Lmt”，使用特殊计算公式进行计算。**

2 **分配 “MV”，计算 “MV_Heat”和 “MV_Cool”。**

“MVLowLmt”=−100、“MVUpLmt”=200 时，“MV”与 “MV_Heat”、“MV_Cool”的关系如下图所示。计算出的 “MV_Heat”的上限值为200、下限值为0，“MV_Cool”的上限值为100、下限值为0。即 “MV_Heat”的上限值与 “MVU p Lmt”的值一致，“MV_Cool”的上限值与 “MVLowLmt”的绝对值一致。



“MVLowLmt”=100、“MVUpLmt”=200时，“MV”与 “MV_Heat”、“MV_Cool”的关系如下图所示。计算出的 “MV_Heat”的上限值为200、下限值为100，“MV_Cool”的值始终为0。即 “MV_Heat”的上限值和下限值分别与 “MVU p Lmt”和 “MVLowLmt”一致。



如上所述，根据 “MVLowLmt”和 “MVUpLmt”为正数还是负数，“MV_Heat”和 “MV_Cool”的上限值、下限值变化如下。

“MVLowLmt”的值	“MVUpLmt”的值	“MV_Heat”		“MV_Cool”	
		下限值	上限值	下限值	上限值
正	正	“MVLowLmt”	“MVUpLmt”	0	0
负	正	0	“MVUpLmt”	0	“MVLowLmt”的绝对值
负	负	0	0	“MVUpLmt”的绝对值	“MVLowLmt”的绝对值

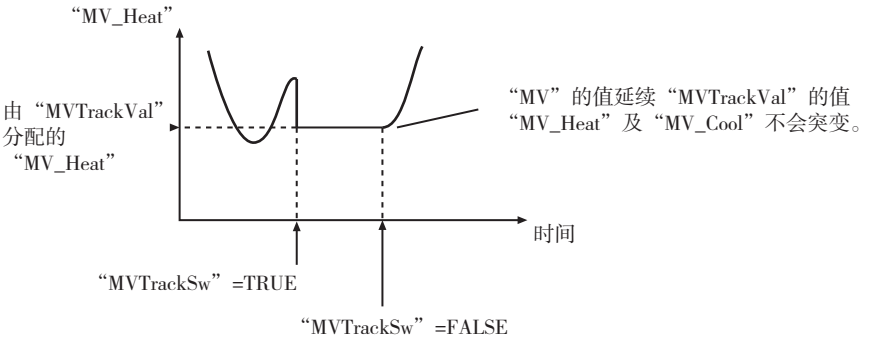
“MVLowLmt”、“MVUpLmt”请务必设定为 “MVLowLmt”< “MVUpLmt”。
为 “MV”设置停止时操作量 “StopMV”、异常时操作量 “ErrorMV”或手动操作量 “ManMV”中的任意一个时，不属于限位控制的对象。

- **“ManResetVal”(手动复位值)**

本指令不使用。即使设定数值也会被忽略。

- **“MVTrackSw”(MV跟踪开关)**

自动运行时，为 “MV”强制设置来自外部的输入值(MV跟踪值) “MVTrackVal”的功能称为MV跟踪。
“MVTrackSw”的值为TRUE的期间，执行MV跟踪。
如果 “MVTrackSw”的值变为FALSE，则 “MV”的值返回加热冷却PID运算结果。此时的 “MV”值将直接延续 “MVTrackVal”的值。因此，“MV_Heat”及 “MV_Cool”的值不会突变。



- **“MVTrackVal”(MV跟踪值)**

通过MV跟踪，为 “MV”强制设置的值。
通过 “MVLowLmt”和 “MVUpLmt”，限位控制 “MVTrackVal”的值。

- **“StopMV”(停止时操作量)**

“Run”的值变为FALSE，停止执行本指令时 “MV”的值。

- **“ErrorMV”(异常时操作量)**

发生异常(“Error”的值为TRUE)时 “MV”的值。
如果 “ErrorMV”的值超过有效范围(–320 ~ 320)，则异常时的 “MV”值变为0。

- **“Alpha”(2自由度PID参数α)**

确定目标值滤波器系数的参数。
详情请参阅 “PIDAT指令(P.2–656)”的示例目标值滤波器型2自由度PID控制的说明。
一般请将 “Alpha”的值设定为0.65。

- **“ATCalcGain”(自动调谐计算增益)**

使通过执行自动调谐计算出的PID常数反映为实际PID常数时的系数。
如果指定1.00，则直接反映自动调谐的结果。
重视稳定性时增大 “ATCalcGain”的值，重视快速响应性时减小 “ATCalcGain”的值。

- **“ATHystrs”(自动调谐滞后)**

表示自动调谐时极限环滞后的大小。
可对较小的 “ATHystrs”值进行高精度的调谐。测量值不稳定而难以正常调谐时，增大 “ATHystrs”的值。
详情请参阅 “PIDAT指令(P.2–656)”中自动调谐的说明。

- **“SampTime”(采样周期)**

进行加热冷却PID运算的周期的最小值。
详情请参阅加热冷却PID运算执行时间的说明。执行加热冷却PID运算后的经过时间小于“SampTime”时，不执行下一加热冷却PID运算。

- **“RngLowLmt”(输入范围下限)、“RngUpLmt”(输入范围上限)**

“PV”和“SP”的下限值和上限值。
与“PV”和“SP”连接的参数值偏离该范围时，发生异常。
请务必设定为使“RngLowLmt”<“RngUpLmt”。

- **“DirOpr”(动作方向)**

本指令不使用。即使设定数值也会被忽略。

- **“CtlPrd_Cool”(控制周期)**

组合本指令与 “TimeProportionalOut指令(P.2-719)”，分时比例输出“MV_Cool”时的控制周期。请设定与TimeProportionalOut指令的控制周期“CtlPrd”相同的值。
不对“MV_Cool”进行分时比例输出时，请设定初始值T#20s。

- **“ProportionalBand_Heat”、“ProportionalBand_Cool”(比例带)**

3个PID常数中的1个。详情请参阅 “PIDAT指令(P.2-656)”中比例动作的说明。
如果“ProportionalBand_Heat”、“ProportionalBand_Cool”的值较大，则偏置增大。如果较小，则会发生震荡。

- **“IntegrationTime_Heat”、“IntegrationTime_Cool”(积分时间)**

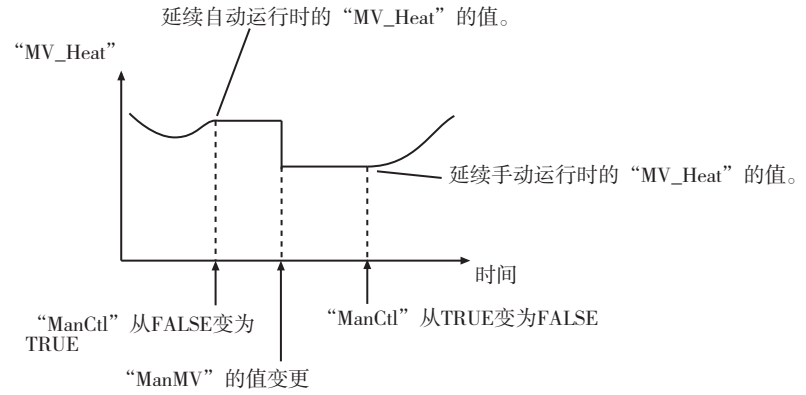
3个PID常数中的1个。详情请参阅 “PIDAT指令(P.2-656)”中积分动作的说明。
“IntegrationTime_Heat”、“IntegrationTime_Cool”的值越大，则积分动作越弱。

- **“DerivativeTime_Heat”、“DerivativeTime_Cool”(微分时间)**

3个PID常数中的1个。详情请参阅 “PIDAT指令(P.2-656)”中微分动作的说明。
“DerivativeTime_Heat”、“DerivativeTime_Cool”的值越大，则微分动作越强。

- **“ManMV”(手动操作量)**

手动运行(“ManCtl”=TRUE)时 “MV”的值。
切换至手动运行后变更 “ManMV”的值时，为 “MV”设置 “ManMV”的值。
从自动运行切换至手动运行后，如自动运行时的 “MV_Heat”值为正值，则 “MV”值为 “MV_Heat”的值，此外则为 “MV_Cool”的值。
同样，从手动运行切换至自动运行后，如手动运行时的 “MV_Heat”值为正值，则 “MV”值为 “MV_Heat”的值，此外则为 “MV_Cool”的值。
“ManMV”的值可超过 “MVLowLmt” ~ “MVUpLmt”的范围。



- **“ATDone”(自动调谐正常结束)**

表示自动调谐正常结束的标志。
自动调谐正常结束后，“StartAT”的值为TRUE期间，变为TRUE。
以下情况会变为FALSE。

- 自动调谐异常结束时。
- 自动调谐执行中(“ATBusy”的值为TRUE)。
- 不执行自动调谐即处于加热冷却PID运算状态。
- 非加热冷却PID运算状态(“Run”的值为FALSE)。
- “StartAT”的值为FALSE。

- **“ATBusy”(自动调谐执行中)**

表示正在执行自动调谐的标志。
正在执行自动调谐时，变为TRUE。其余情况下，均变为FALSE。

- **“MV”(操作量)**

通过加热冷却PID运算计算出的操作量。分配 “MV”，计算 “MV_Heat”和 “MV_Cool”。

- **“MV_Heat”(加热操作的操作量)**

给予加热装置的操作量。

- **“MV_Cool”(冷却操作的操作量)**

给予冷却装置的操作量。

加热冷却PID运算

PID运算的详情请参阅 “PIDAT指令(P.2-656)”。
加热冷却PID运算是使用加热操作的PID常数和冷却操作的PID常数计算操作量的运算。上一次的运算结果为 “MV”≤0时使用加热操作的PID常数进行运算，“MV”>0时则使用冷却操作的PID常数进行运算。

比例动作(P)、积分动作(I)、微分动作(D)

比例动作(P)、积分动作(I)、微分动作(D)的详情请参阅 “PIDAT指令(P.2-656)”。

目标值滤波器型2自由度PID控制

目标值滤波器型2自由度PID控制的详情请参阅 “PIDAT指令(P.2-656)”。

运行开始的步骤

需使用最佳PID常数运行本指令。因此，运行开始的方法有下述2种。

● 不确定PID常数的最佳值时

不确定PID常数的最佳值时，运行开始时进行自动调谐，计算PID常数的最佳值。
在将 “StartAT”的值设为TRUE的状态下，使 “Run”的值从FALSE变为TRUE。
首先，执行自动调谐，使用计算出的PID常数，开始加热冷却PID运算。

● 已明确PID常数的最佳值时

将PID常数的最佳值设定为 “ProportionalBand_Heat”、“IntegrationTime_Heat”、“DerivativeTime_Heat”及 “ProportionalBand_Cool”、“IntegrationTime_Cool”、“DerivativeTime_Cool”，使 “Run”的值从FALSE变为TRUE。
“ProportionalBand_Heat”、“IntegrationTime_Heat”、“DerivativeTime_Heat”及“ProportionalBand_Cool”、“IntegrationTime_Cool”、“DerivativeTime_Cool”为输入输出变量。无法将常数作为输入参数。请务必定义合适的变量，将代入的值作为输入参数。

可在运行过程中变更PID常数。还可在运行过程中执行自动调谐。此时，将“StartAT”的值设为TRUE。

控制状态与操作量

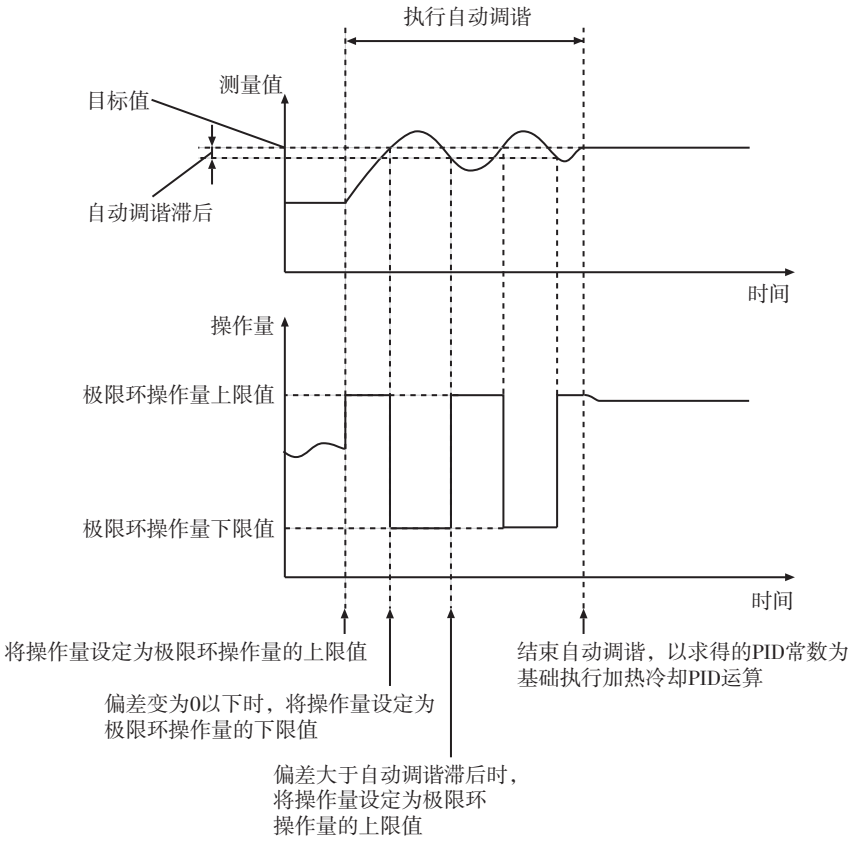
如下所示，根据控制状态，确定操作量 “MV”。

控制状态	变量值					操作量 “MV”
	手动/自动切换 “ManCtl”	执行条件 “Run”	异常结束 “Error”	MV跟踪开关 “MV TrackSw”	自动调谐执行中 “ATBusy”	
异常结束	FALSE	TRUE	TRUE	–	FALSE	异常时操作量 “ErrorMV”
自动运行中 MV跟踪			FALSE	TRUE		MV跟踪值 “MVTrackVal”
自动运行中 自动调谐执行中				FALSE	TRUE	重复操作量限位上限制和操作量限位下限制
自动运行中 不执行自动调谐					FALSE	通过该时刻的PID常数计算出的值
停止执行指令		FALSE	–	–		停止时操作量 “StopMV” *1
手动运行中	TRUE	–				手动操作量 “ManMV” *2

*1 “StopMV”的值超过有效范围时，“MV”的值变为0。
*2 “ManMV”的值超过有效范围时，“MV”的值变为0。

自动调谐

由于极少调整2自由度PID参数α，因此本指令主要调整的设定值为PID常数。
本指令带有自动调谐PID常数的功能。使用极限环法作为自动调谐的方法。极限环法是指强制将操作量暂时设为极限环操作量的上限值和下限值，根据此时的测量值的变化情况计算PID常数最佳值的方法。
如果有自动调谐执行指示，则首先将操作量设定为极限环操作量的上限值。然后，偏差变为0以下时，将操作量设定为极限环操作量的下限值。偏差大于自动调谐滞后的值时，将操作量设定为极限环操作量的上限值。重复2次半的上述操作，计算PID常数的最佳值。
极限环操作量的上限值和下限值根据各参数值进行计算。



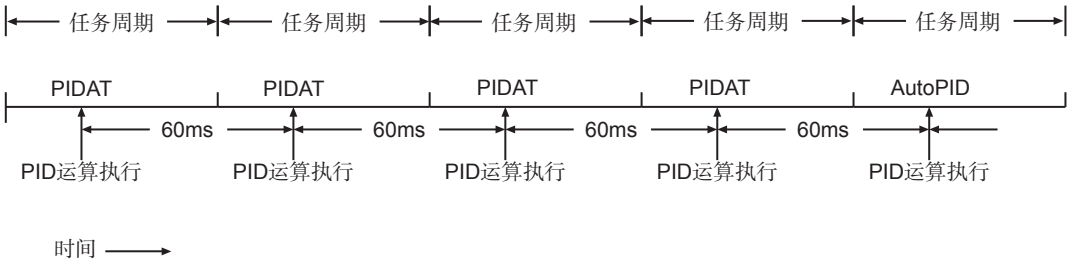
“Run”的值为TRUE且处于加热冷却PID运算状态时，如果 “StartAT”的值由FALSE变为TRUE，则在运行过程中执行自动调谐。在 “StartAT”的值为TRUE的状态下，如果 “Run”的值由FALSE变为TRUE，则运行开始时执行。
如果自动调谐正常结束，则立即反映计算出的PID常数的最佳值。
自动调谐过程中(“ATBusy” =TRUE)，如果将 “StartAT”的值设为FALSE，则中止自动调谐。此时，通过开始自动调谐之前的PID常数重新开始加热冷却PID运算。

加热冷却PID运算的执行时间

定期重复执行加热冷却PID运算。在用户程序中执行本指令时，执行加热冷却PID运算。
执行上次加热冷却PID运算后的经过时间小于采样周期 “SampTime”时不执行。
执行上次加热冷却PID运算后的经过时间大于 “SampTime”时执行，剩余的时间(上次执行后的经过时间–“SampTime”)转入下一次。详情请参阅下图。
即使在因PrgStop指令及MC指令而未执行本指令的情况下，在下图的 “执行PID运算”时，上一次执行加热冷却PID运算起的经过时间变为0。

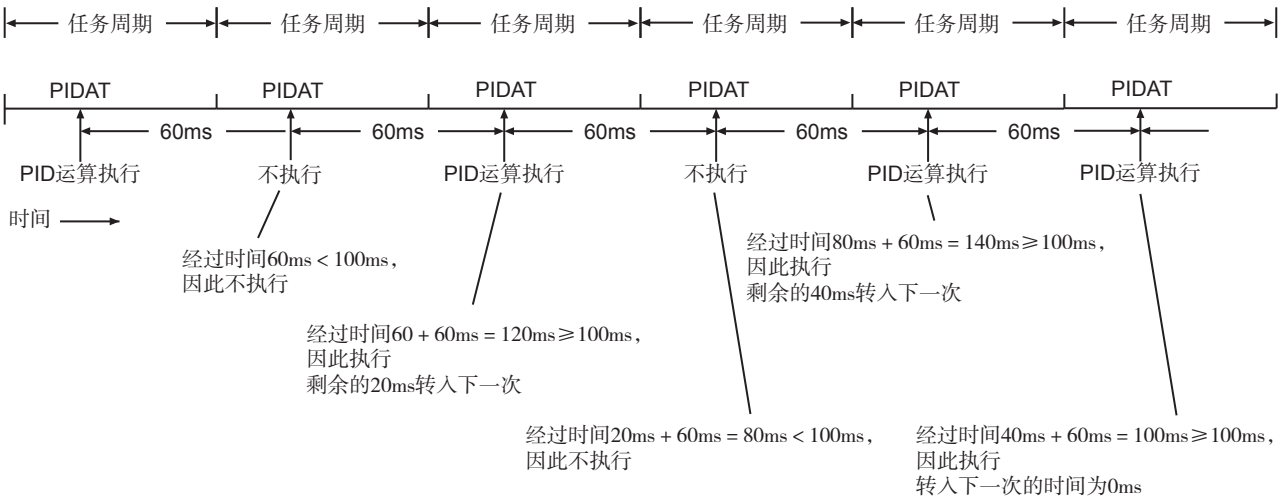
任务周期 = 60ms “SampTime”<60ms时

任务周期≥“SampTime”，因此1个任务周期必定执行1次。



任务周期 = 60ms “SampTime”=100ms时

任务周期<“SampTime”，因此可能不执行。

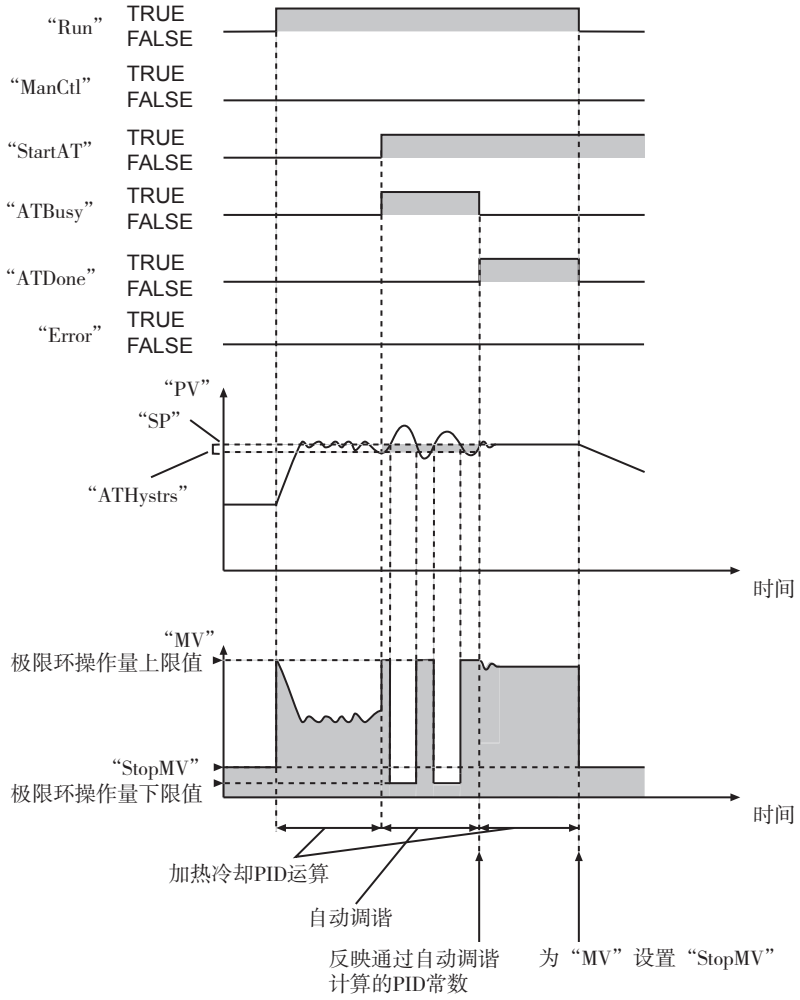


时序图

表示有数个变量时各变量的时序图。

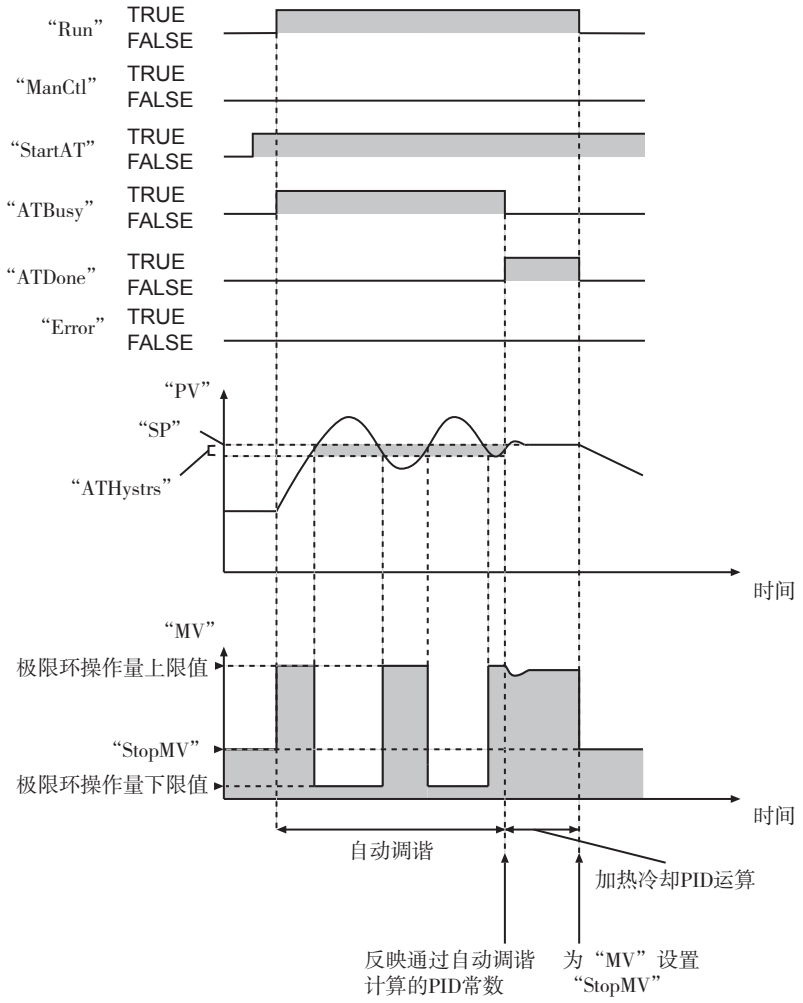
● 在自动运行过程中执行自动调谐时

- 下图情况下，“ManCtl”的值为FALSE，因此在“Run”的值为FALSE期间，“MV”的值为“StopMV”。
- 将 “Run”的值设为TRUE时，将根据PID常数输出 “MV”。
- 将 “StartAT”的值设为TRUE时，将执行自动调谐。“ATBusy”的值变为TRUE。
- 自动调谐结束时，“ATBusy”的值变为FALSE，“ATDone”的值变为TRUE。
- 自动调谐结束后，将根据自动调谐计算出的PID常数输出 “MV”。
- 将“Run”的值设为FALSE时，“MV”的值将变为 “StopMV”。此外，“ATDone”的值变为FALSE。



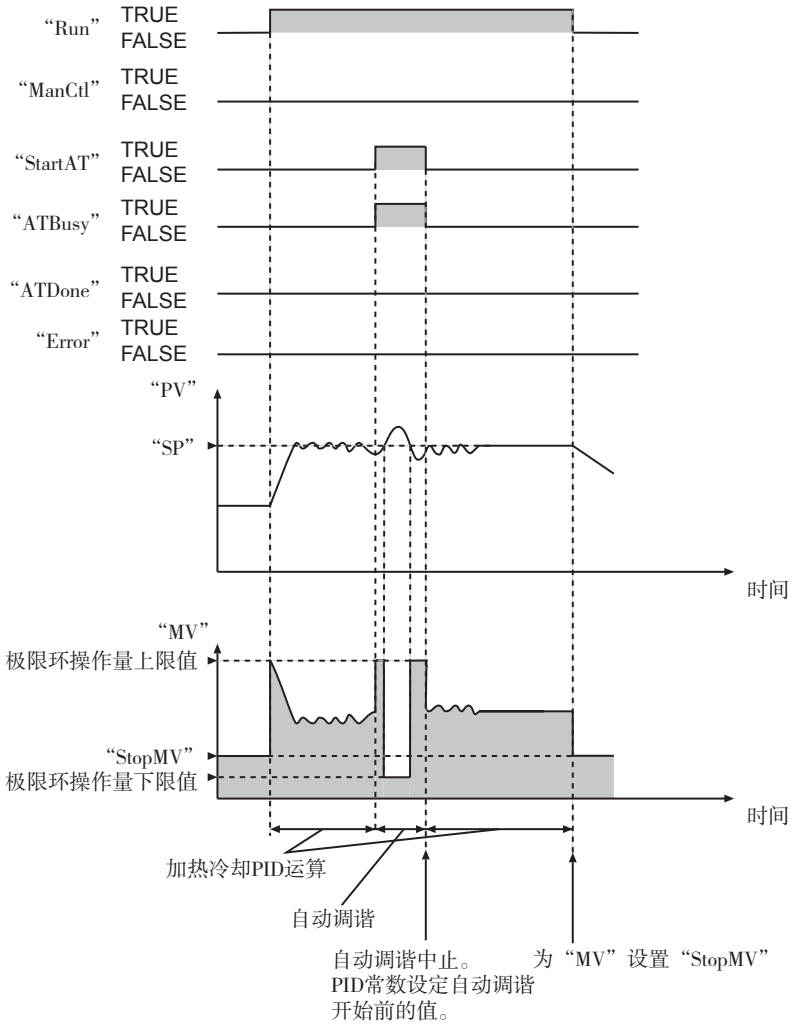
● 在执行本指令的开头执行自动调谐时

- 下图情况下，“ManCtl”的值为FALSE，因此在“Run”的值为FALSE期间，“MV”的值为“StopMV”。
- “Run”的值为FALSE期间，即使将 “StartAT”的值设为TRUE，也不会执行自动调谐。
- 将 “StartAT”和 “Run”的值均设为TRUE时，将执行自动调谐。“ATBusy”的值变为TRUE。
- 自动调谐结束时，“ATBusy”的值变为FALSE，“ATDone”的值变为TRUE。
- 自动调谐结束后，将根据自动调谐计算出的PID常数输出 “MV”。



● 中止自动调谐时

- 下图情况下，“ManCtl”的值为FALSE，因此在“Run”的值为FALSE期间，“MV”的值为“StopMV”。
- 将 “Run”的值设为TRUE时，将根据PID常数输出 “MV”。
- 将 “StartAT”的值设为TRUE时，将执行自动调谐。“ATBusy”的值变为TRUE。
- 自动调谐过程中，如果将“StartAT”的值设为FALSE，则中止自动调谐。“ATBusy”的值变为FALSE。
- 自动调谐中止后，将根据自动调谐开始前的PID常数输出 “MV”。
- 将 “Run”的值设为FALSE时，“MV”的值将变为 “StopMV”。
- 已中止自动调谐，因此 “ATDone”的值不会变为TRUE。



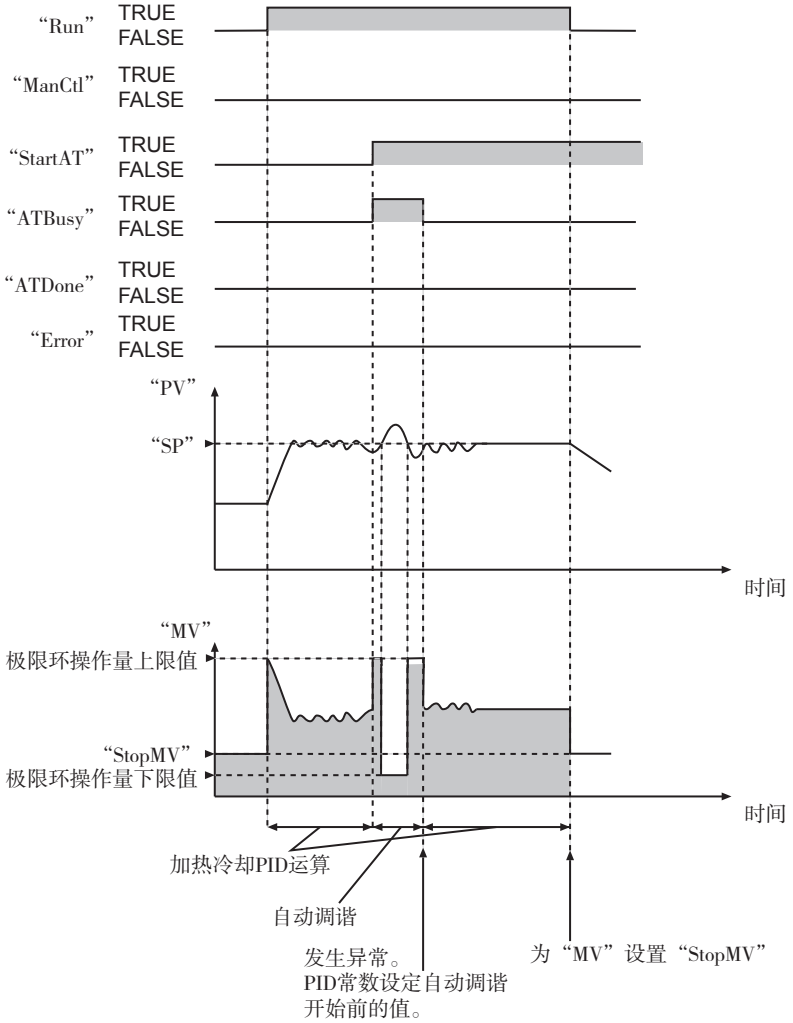
● 在自动调谐过程中发生自动调谐异常时

以下情况时会发生自动调谐异常，中止自动调谐。

- 操作量等于极限环操作量的上限值且偏差变为0之前的时间超过19999s。
- 操作量等于极限环操作量的下限值且偏差大于 “ATHystrs”之前的时间超过19999s。

即使发生自动调谐异常，“Error”的值也不变为TRUE。此外，也不会记录在事件日志中。
如果中止自动调谐，则通过开始自动调谐之前的PID常数重新开始加热冷却PID运算。

- 下图情况下，“ManCtl”的值为FALSE，因此在“Run”的值为FALSE期间，“MV”的值为“StopMV”。
- 将 “Run”的值设为TRUE时，将根据PID常数输出 “MV”。
- 将 “StartAT”的值设为TRUE时，将执行自动调谐。“ATBusy”的值变为TRUE。
- 执行自动调谐的过程中发生自动调谐异常时，将立即中止自动调谐。“ATBusy”的值变为FALSE。
- 即使发生自动调谐异常，“Error”的值也不变为TRUE。
- 自动调谐中止后，将根据自动调谐开始前的PID常数输出 “MV”。
- 将 “Run”的值设为FALSE时，“MV”的值将变为 “StopMV”。
- 已中止自动调谐，因此 “ATDone”的值不会变为TRUE。



参考

PID常数的调整方法

PID常数的调整方法请参阅 “PIDAT指令(P.2–656)”。

温度调节时的PID常数初始值

将本指令用于温度调节时，PID常数的初始值请参考下述值。请使用除PID常数以外的各变量的初始值。

变量	参考初始值*1
“ProportionalBand_Heat”、“ProportionalBand_Cool”	10%FS
“IntegrationTime_Heat”、“IntegrationTime_Cool”	233s
“DerivativeTime_Heat”、“DerivativeTime_Cool”	40s

*1 执行自动调谐后，请遵从该结果。

使用注意事项

- “PV”和 “SP”的值需大于 “RngLowLmt”的值且小于 “RngUpLmt”的值。因此，如下所示，请符合这些变量的单位系统。

单位系统	“PV”、“SP”的值	“RngLowLmt”、“RngUpLmt”的值
%FS	“PV”=(测量值的物理量 – MIN)/(MAX – MIN) × 100 “SP”=(目标值的物理量 – MIN)/(MAX – MIN) × 100 *1	“RngLowLmt”=0 “RngUpLmt” =100
物理量	“PV”=测量值的物理量 “SP”=目标值的物理量	“RngLowLmt”=MIN “RngUpLmt” = MAX*1

*1 MAX：输入范围上限的物理量、MIN：输入范围下限的物理量

• 如下所示，根据控制状态，确认可否变更各变量。

变量	控制状态		
	停止执行指令*1	运行中 (不执行自动调谐) *2	运行中 (自动调谐执行中) *3
“Run”	○	○	○
“ManCtl”	○	○	○
“StartAT”	○	○	○
“DeadBand”	○	○	○
“PV”	○	○	○
“SP”	○	○	×*4
“MVLowLmt”	○	○	×*4
“MVUpLmt”	○	○	×*4
“ManResetVal” *5	–	–	–
“MVTrackSw”	○	○	×*4
“MVTrackVal”	○	○	×*4
“StopMV”	○	○	○
“ErrorMV”	○	○	○
“Alpha”	○	○	×*4
“ATCalcGain”	○	○	×*4
“ATHystrs”	○	○	×*4
“CtlPrdCool”	○	○	×*4
“SampTime”	○	×*6	×*4
“RngLowLmt”	○	×*6	×*4
“RngUpLmt”	○	×*6	×*4
“DirOpr” *5	–	–	–
“ProportinalBand _Heat”	○	○	×*7
“IntegrationTime _Heat”	○	○	×*7
“DerivativeTime _Heat”	○	○	×*7
“ProportinalBand _Cool”	○	○	×*7
“IntegrationTime _Cool”	○	○	×*7
“DerivativeTime _Cool”	○	○	×*7
“ManMV”	○	○	○

*1 “ManCtl” =TRUE、“Run” =FALSE、“Error” =TRUE或 “MVTrackSw” =TRUE
*2 “ManCtl” =FALSE、“Run” = TRUE、“Error” =FALSE、“MVTrackSw” = FALSE且 “ATBusy” =FALSE
*3 “ManCtl” =FALSE、“Run” = TRUE、“Error” =FALSE、“MVTrackSw” = FALSE且 “ATBusy” =TRUE
*4 使用执行自动调谐前的值，执行自动调谐。
*5 本指令不使用。可变更值，但会被忽略。
*6 使用执行运行前的值，执行运行。
*7 可变更值，但会被忽略。自动调谐完成时，将被自动调谐计算出的值所改写。

- “SampTime”的值小于100ns时舍去。
- “ManCtl”的值为TRUE时，如果将 “StartAT”的值设为TRUE，则接下来 “ManCtl”的值变为FALSE时，开始自动调谐。
- 如果 “ErrorMV”的值超过有效范围(–320 ~ 320)，则异常时的 “MV”值变为0。
- 自动调谐过程中如果将 “ManCtl”的值设为TRUE，则中止自动调谐。
- 即使发生自动调谐异常，“Error”的值也不变为TRUE。此外，也不会记录在事件日志中。
- 以下情况时会发生异常。“Error”变为TRUE，将异常ID代入 “ErrorID”。“ATDone”、“ATBusy”为FALSE。如果 “ManCtl”和 “Run”的值均为FALSE，则为 “MV”设置 “ErrorMV”的值。“ErrorMV”的值超过有效范围时，“MV”的值变为0。

异常的内容	“ErrorID”的值
任一输入变量超过有效范围	16#0400
“RngLowLmt” ≥ “RngUpLmt”	16#0401
“MVLowLmt” ≥ “MVUpLmt”	

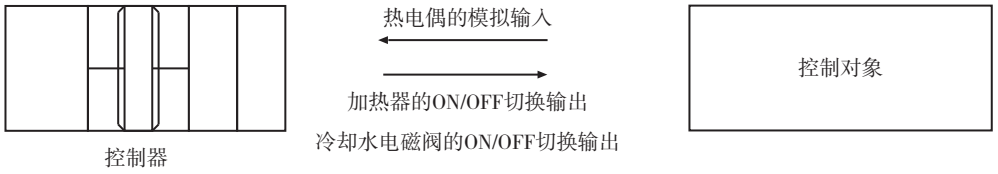
- 如需在上述以外的条件下使异常停止，则请在发生异常时设置将 “Run”的值设为FALSE的处理。
- 因 “PV”或 “SP”的值超过有效范围而发生异常后，即使将值变更到有效范围内，仍保持5秒的异常状态。即5秒内 “Error”的值保持为FALSE。
- 解除异常后，如果 “Run”的值为TRUE，则自动重新开始加热冷却PID运算。“Run”和 “StartAT”的值均为TRUE时，自动重新开始自动调谐。
- 按采样周期判定有无异常。
- 按以下条件执行备份、恢复时，自动调谐计算出的PID常数将恢复为备份前的值。请予以注意。
  - 对输入输出参数指定保持变量。
  - 依次执行备份、自动调谐、恢复。
- 从自动运行切换成手动运行时，“MV_Heat”和 “MV_Cool”中的正值将延续数值而无波动(无突变)。因此，另一个值则可能会突变。

版本相关信息

本指令可用于CPU单元Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

使用PIDAT_HeatCool指令，进行温度控制。来自控制对象的输入中，有1个热电偶模拟输入。对控制对象的输出中，有加热用数字输出和冷却用数字输出两个输出。加热用数字输出为加热器的ON/OFF切换，冷却用数字输出为冷却水电磁阀的ON/OFF切换。



单元构成

连接的单元如下所述。

- C1JW-AD04U 绝缘型多功能输入单元
- C1JW-OC201 继电器触点输出单元

I/O映射

各单元的I/O映射设定如下。

● C1JW-AD04U

端口	说明	R/W	数据类型	变量	变量注释	变量种类
Ch1_AIInPV	输入No.1测量值	R	INT	J01_Ch1_AIInPV	热电偶输入	全局变量

● CJ1W-OC201

端口	说明	R/W	数据类型	变量	变量注释	变量种类
Ch1_Out00	输出CH1触点00	RW	BOOL	J02_Ch1_Out00	加热装置用输出	全局变量
Ch1_Out04	输出CH1触点04	RW	BOOL	J02_Ch1_Out04	冷却装置用输出	全局变量

触摸屏的规格

本示例程序以控制器连接触摸屏为前提。基于触摸屏的输入输出信息如下。

输入/输入输出/输出	信息
输入	本示例程序的执行标志 手动/自动切换标志 目标值 自动调谐执行标志 死区 初始设定参数 运行中设定参数
输入输出	加热操作的比例带、积分时间、微分时间 冷却操作的比例带、积分时间、微分时间 手动操作量
输出	测量值 自动调谐正常结束标志 自动调谐执行中标志 异常发生标志 操作量 加热操作的操作量 冷却操作的操作量

转换为操作量的分时比例输出

该示例中，加热装置用输出和冷却装置用输出均为ON/OFF数字输出。因此，加热装置的操作量和冷却装置的操作量均需转换成分时比例输出。将操作量转换为分时比例输出需使用 “TimeProportionalOut 指令(P.2–719)”。

但在自动调谐的过程中，PIDAT_HeatCool 指令的输出 “MV_Heat”、“MV_Cool”变化后，必须立即改变加热装置用输出、冷却装置用输出，因此无法使用TimeProportionalOut指令。若使用TimeProportionalOut指令，加热装置用输出、冷却装置用输出将只按用户设定的控制周期变化。因此，在本示例程序中，使用定时器指令将自动调谐中的操作量转换为分时比例输出。

应用程序

全局变量的定义

全局变量

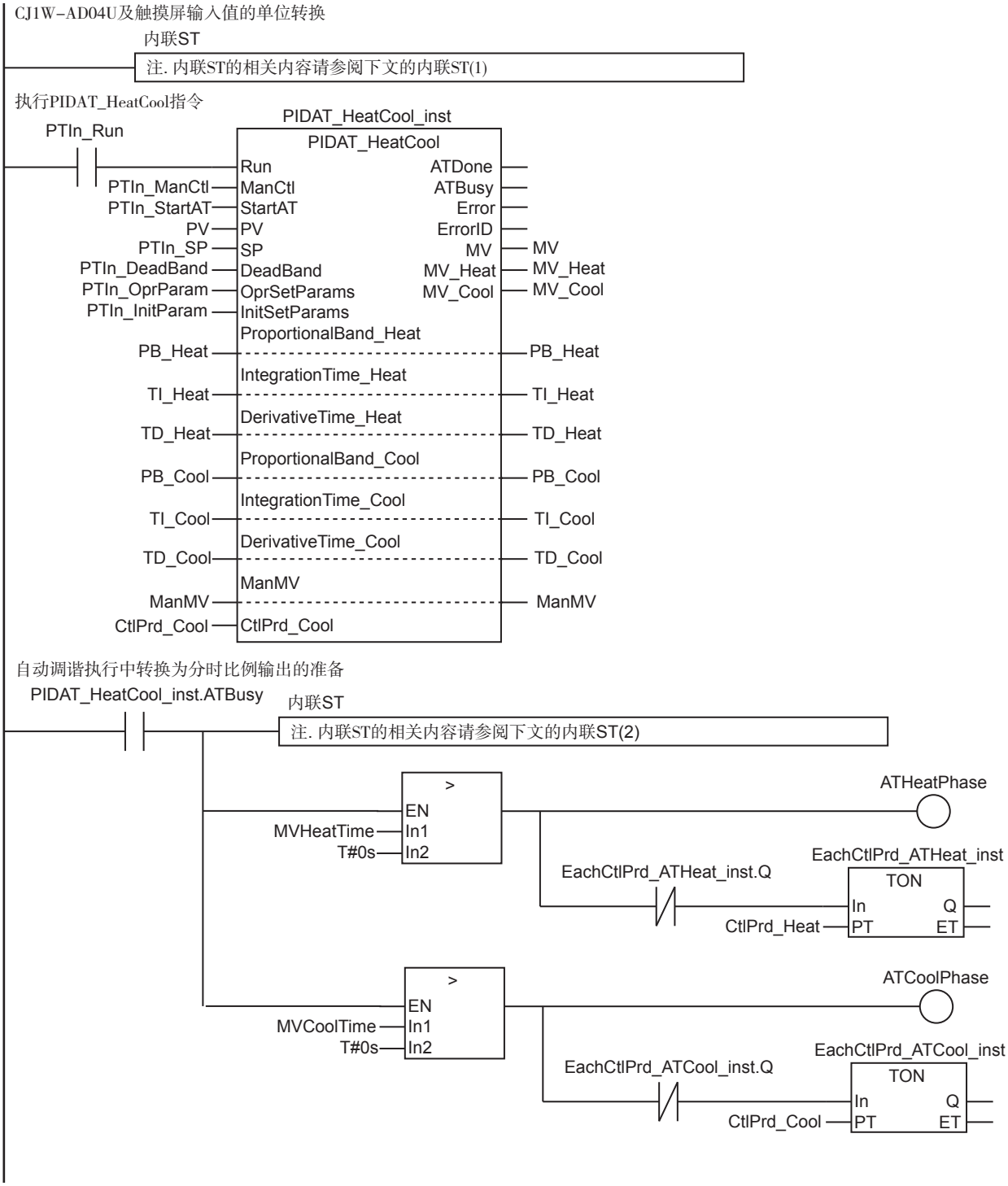
名称	数据类型	初始值	分配对象	保持	网络公布	注释
J01_Ch1_AIInPV	INT	0	IOBus://rack#0/slot#0/Ch1_AIInPV	☐	不公布	来自C1JW-AD04U的热电偶输入
J02_Ch1_Out00	BOOL	FALSE	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out00	☐	不公布	对C1JW-OC201的加热用输出
J02_Ch1_Out04	BOOL	FALSE	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out04	☐	不公布	对C1JW-OC201的冷却用输出
PTIn_Run	BOOL	FALSE		☑	输入	通过触摸屏输入本示例程序执行标志
PTIn_ManCtl	BOOL	FALSE		☑	输入	通过触摸屏输入手动/自动切换标志
PTIn_SP	REAL			☑	输入	通过触摸屏输入目标值
PTIn_StartAT	BOOL	FALSE		☑	输入	通过触摸屏输入自动调谐执行标志
PTIn_DeadBand	REAL	0		☑	输入	通过触摸屏输入死区
PTIn_InitParam	_sINIT_SET_PARAMS	(SampTime := T#100ms, RngLowLmt := 0.0, RngUpLmt := 100.0, DirOpr := False)		☑	输入	通过触摸屏输入初始设定参数
PTIn_InitSetOpr_SampTime	LINT	100		☑	输入	通过触摸屏输入采样周期(以ms为单位)
PTIn_OprParam	_sOPR_SET_PARAMS	(MVLowLmt := -100, MVUpLmt := 100, ManResetVal := 0.0, MVTrackSw := False, MVTrackVal := 0.0, StopMV := 0.0, ErrorMV := 0.0, Alpha := 0.65, ATCalcGain := 1.0, ATHystrs := 0.2)		☑	输入	通过触摸屏输入运行中设定参数
PTOut_PV	REAL	0		☐	输出	对触摸屏输出测量值
PT_PB_Heat	REAL	1		☑	输入	通过触摸屏输入输出加热操作的比例带
PT_TI_Heat	LINT	1000		☑	输入	通过触摸屏输入输出加热操作的积分时间(以ms为单位)
PT_TD_Heat	LINT	1000		☑	输入	通过触摸屏输入输出加热操作的微分时间(以ms为单位)
PT_PB_Cool	REAL	1		☑	输入	通过触摸屏输入输出冷却操作的比例带
PT_TI_Cool	LINT	1000		☑	输入	通过触摸屏输入输出冷却操作的积分时间(以ms为单位)
PT_TD_Cool	LINT	1000		☑	输入	通过触摸屏输入输出冷却操作的微分时间(以ms为单位)

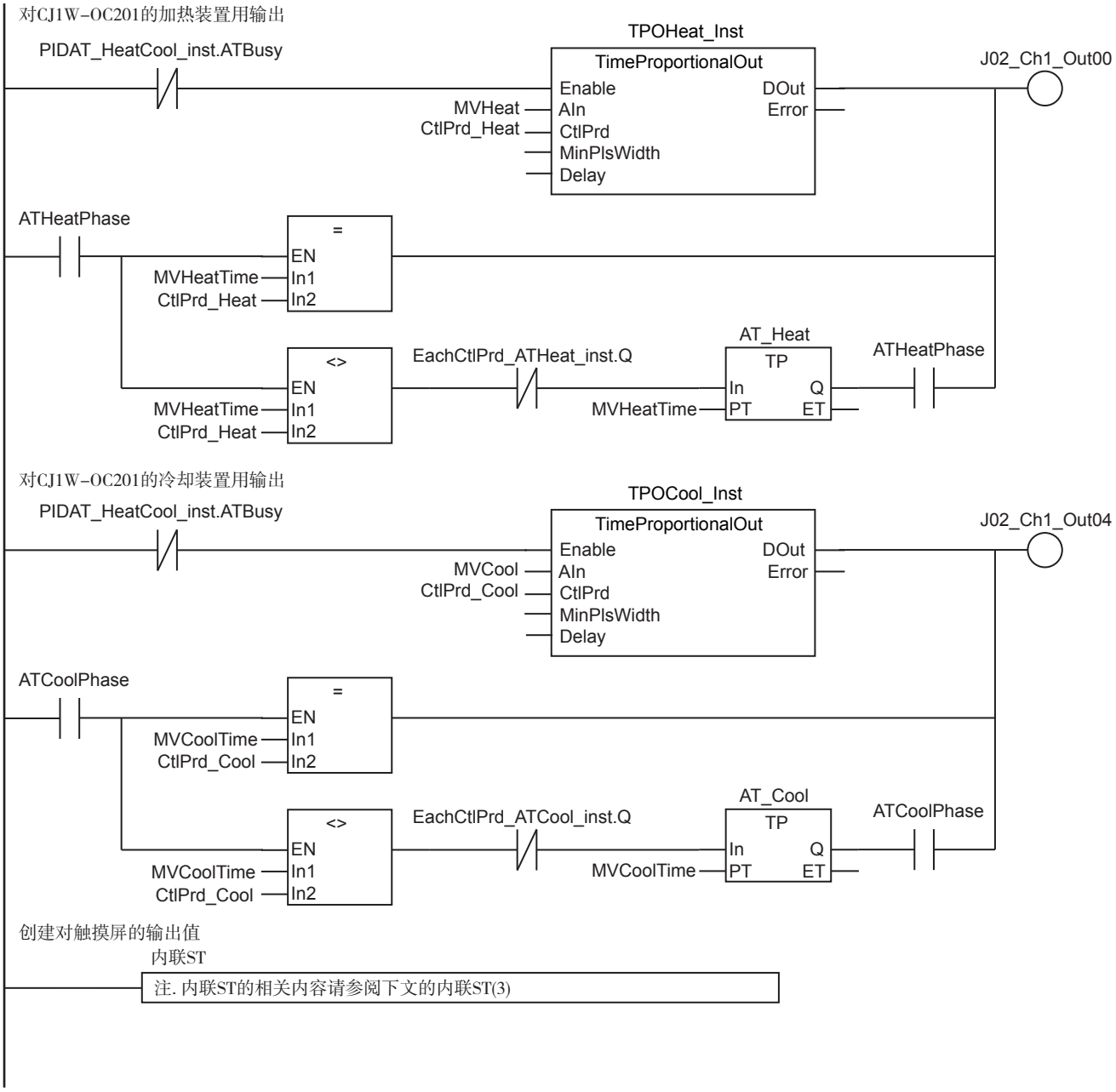
名称	数据类型	初始值	分配对象	保持	网络公布	注释
PT_ManMV	REAL	0		☑	输入	通过触摸屏输入输出手动操作量
PTOut_ATDone	BOOL	FALSE		☐	输出	对触摸屏输出自动调谐正常结束标志
PTOut_ATBusy	BOOL	FALSE		☐	输出	对触摸屏输出自动调谐执行中标志
PTOut_Error	BOOL	FALSE		☐	输出	对触摸屏输出异常结束标志
PTOut_MV	REAL	0		☐	输出	对触摸屏输出操作量
PTOut_MVHeat	REAL	0		☐	输出	对触摸屏输出加热操作的操作量
PTOut_MVCool	REAL	0		☐	输出	对触摸屏输出冷却操作的操作量

LD

内部变量	名称	数据类型	初始值	注释
	PB_Heat	REAL	0	加热操作的比例带
	PB_Cool	REAL	0	冷却操作的比例带
	MV	REAL	0	操作量
	MV_Heat	REAL	0	加热操作的操作量
	MV_Cool	REAL	0	冷却操作的操作量
	PIDAT_HeatCool_inst	PIDAT_HeatCool		PIDAT_HeatCool指令的实例
	TI_Heat	TIME	T#0s	加热操作的积分时间
	TI_Cool	TIME	T#0s	冷却操作的积分时间
	TD_Heat	TIME	T#0s	加热操作的微分时间
	TD_Cool	TIME	T#0s	冷却操作的微分时间
	ManMV	REAL	0	手动操作量
	CtlPrd_Cool	TIME	T#20s	冷却操作的控制周期
	CtlPrd_Heat	TIME	T#2s	加热操作的控制周期
	TPOHeat_inst	TimeProportionalOut		加热操作用TimeProportionalOut指令的实例
	TPOCool_inst	TimeProportionalOut		冷却操作用TimeProportionalOut指令的实例
	ATHeatPhase	BOOL	FALSE	自动调谐的加热操作中标志
	ATCoolPhase	BOOL	FALSE	自动调谐的冷却操作中标志
	MVHeatTime	TIME	T#0s	自动调谐中的加热操作时间
	MVCoolTime	TIME	T#0s	自动调谐中的冷却操作时间
	AT_Heat_inst	TP		自动调谐中的加热操作的操作量输出用TP指令实例
	AT_Cool_inst	TP		自动调谐中的冷却操作的操作量输出用TP指令实例
	EachCtlPrd_ATHeat_inst	TON		自动调谐中的加热操作的操作量输出用TON指令实例
	EachCtlPrd_ATCool_inst	TON		自动调谐中的冷却操作的操作量输出用TON指令实例
	PV	REAL	0	测量值

外部变量	名称	数据类型	注释
	J01_Ch1_AIInPV	INT	来自C1JW-AD04U的热电偶输入
	J02_Ch1_Out00	BOOL	对C1JW-OC201的加热用输出
	J02_Ch1_Out04	BOOL	对C1JW-OC201的冷却用输出
	PTIn_Run	BOOL	通过触摸屏输入本示例程序执行标志
	PTIn_ManCtl	BOOL	通过触摸屏输入手动/自动切换标志
	PTIn_SP	REAL	通过触摸屏输入目标值
	PTIn_StartAT	BOOL	通过触摸屏输入自动调谐执行标志
	PTIn_DeadBand	REAL	通过触摸屏输入死区
	PTIn_InitParam	_sINIT_SET_PARAMS	通过触摸屏输入初始设定参数
	PTIn_InitSetOpr_SampTime	LINT	通过触摸屏输入采样周期(以ms为单位)
	PTIn_OprParam	_sOPR_SET_PARAMS	通过触摸屏输入运行中设定参数
	PTOut_PV	REAL	对触摸屏输出测量值
	PT_PB_Heat	REAL	通过触摸屏输入输出加热操作的比例带
	PT_TI_Heat	LINT	通过触摸屏输入输出加热操作的积分时间(以ms为单位)
	PT_TD_Heat	LINT	通过触摸屏输入输出加热操作的微分时间(以ms为单位)
	PT_PB_Cool	REAL	通过触摸屏输入输出冷却操作的比例带
	PT_TI_Cool	LINT	通过触摸屏输入输出冷却操作的积分时间(以ms为单位)
	PT_TD_Cool	LINT	通过触摸屏输入输出冷却操作的微分时间(以ms为单位)
	PT_ManMV	REAL	通过触摸屏输入输出手动操作量
	PTOut_ATDone	BOOL	对触摸屏输出自动调谐正常结束标志
	PTOut_ATBusy	BOOL	对触摸屏输出自动调谐执行中标志
	PTOut_Error	BOOL	对触摸屏输出异常结束标志
	PTOut_MV	REAL	对触摸屏输出操作量
	PTOut_MVHeat	REAL	对触摸屏输出加热操作的操作量
	PTOut_MVCool	REAL	对触摸屏输出冷却操作的操作量


CJ1W-AD04U及触摸屏输入值的单位转换
内联ST
注. 内联ST的相关内容请参阅下文的内联ST(1)
执行PIDAT_HeatCool指令
PIDAT_HeatCool_inst
PTIn_Run
PIDAT_HeatCool
Run
ATDone
PTIn_ManCtl
ManCtl
ATBusy
PTIn_StartAT
StartAT
Error
PV
PV
ErrorID
PTIn_SP
SP
MV
MV
PTIn_DeadBand
DeadBand
MV_Heat
MV_Heat
PTIn_OprParam
OprSetParams
MV_Cool
MV_Cool
PTIn_InitParam
InitSetParams
ProportionalBand_Heat
PB_Heat
PB_Heat
IntegrationTime_Heat
TI_Heat
TI_Heat
DerivativeTime_Heat
TD_Heat
TD_Heat
ProportionalBand_Cool
PB_Cool
PB_Cool
IntegrationTime_Cool
TI_Cool
TI_Cool
DerivativeTime_Cool
TD_Cool
TD_Cool
ManMV
ManMV
ManMV
CtlPrd_Cool
CtlPrd_Cool
自动调谐执行中转换为分时比例输出的准备
PIDAT_HeatCool_inst.ATBusy
内联ST
注. 内联ST的相关内容请参阅下文的内联ST(2)
>
EN
MVHeatTime
In1
T#0s
In2
ATHeatPhase
EachCtlPrd_ATHeat_inst
EachCtlPrd_ATHeat_inst.Q
TON
In
Q
CtlPrd_Heat
PT
ET
>
EN
MVCoolTime
In1
T#0s
In2
ATCoolPhase
EachCtlPrd_ATCool_inst
EachCtlPrd_ATCool_inst.Q
TON
In
Q
CtlPrd_Cool
PT
ET



● 内联ST(1)的内容

```
// CJ1W-AD04U及触摸屏输入值的单位转换
PV := INT_TO_REAL(J01_Ch1_AIInPV)/REAL#10.0;

PTIn_InitParam.SampTime := NanoSecToTime(PTIn_InitSetOpr_SampTime*1000000);

PB_Heat := PT_PB_Heat;
TI_Heat  := NanoSecToTime(PT_TI_Heat*1000000);
TD_Heat := NanoSecToTime(PT_TD_Heat*1000000);
PB_Cool  := PT_PB_Cool;
TI_Cool   := NanoSecToTime(PT_TI_Cool*1000000);
TD_Cool := NanoSecToTime(PT_TD_Cool*1000000);

ManMV := PT_ManMV;
```

● 内联ST(2)的内容

```
MVHeatTime := MULTIME(CtlPrd_Heat,(MV_Heat/100));
MVCoolTime := MULTIME(CtlPrd_Cool,(MV_Cool/100));
```

● 内联ST(3)的内容

```
// 创建对触摸屏的输出值
PTOut_PV := PV;

PTOut_ATDone := PIDAT_HeatCool_inst.ATDone;
PTOut_ATBusy := PIDAT_HeatCool_inst.ATBusy;
PTOut_Error     := PIDAT_HeatCool_inst.Error;

PTOut_MV        := PIDAT_HeatCool_inst.MV;
PTOut_MVHeat := PIDAT_HeatCool_inst.MV_Heat;
PTOut_MVCool := PIDAT_HeatCool_inst.MV_Cool;

PT_PB_Heat := PB_Heat;
PT_TI_Heat  := TimeToNanoSec( TI_Heat )/1000000;
PT_TD_Heat := TimeToNanoSec( TD_Heat )/1000000;
PT_PB_Cool  := PB_Cool;
PT_TI_Cool   := TimeToNanoSec( TI_Cool )/1000000;
PT_TD_Cool  := TimeToNanoSec( TD_Cool )/1000000;

PT_ManMV := ManMV;
```

ST

内部变量	名称	数据类型	初始值	注释
	PB_Heat	REAL	0	加热操作的比例带
	PB_Cool	REAL	0	冷却操作的比例带
	MV	REAL	0	操作量
	MV_Heat	REAL	0	加热操作的操作量
	MV_Cool	REAL	0	冷却操作的操作量
	PIDAT_HeatCool_inst	PIDAT_HeatCool		PIDAT_HeatCool指令的实例
	TI_Heat	TIME	T#0s	加热操作的积分时间
	TI_Cool	TIME	T#0s	冷却操作的积分时间
	TD_Heat	TIME	T#0s	加热操作的微分时间
	TD_Cool	TIME	T#0s	冷却操作的微分时间
	ManMV	REAL	0	手动操作量
	CtlPrd_Cool	TIME	T#20s	冷却操作的控制周期
	CtlPrd_Heat	TIME	T#2s	加热操作的控制周期
	TPOHeat_inst	TimeProportionalOut		加热操作用TimeProportionalOut指令的实例
	TPOCool_inst	TimeProportionalOut		冷却操作用TimeProportionalOut指令的实例
	ATHeatPhase	BOOL	FALSE	自动调谐的加热操作中标志
	ATCoolPhase	BOOL	FALSE	自动调谐的冷却操作中标志
	MVHeatTime	TIME	T#0s	自动调谐中的加热操作时间
	MVCoolTime	TIME	T#0s	自动调谐中的冷却操作时间
	AT_Heat_inst	TP		自动调谐中的加热操作的操作量输出用TP指令实例
	AT_Cool_inst	TP		自动调谐中的冷却操作的操作量输出用TP指令实例
	EachCtlPrd_ATHeat_inst	TON		自动调谐中的加热操作的操作量输出用TON指令实例
	EachCtlPrd_ATCool_inst	TON		自动调谐中的冷却操作的操作量输出用TON指令实例
	PV	REAL	0	测量值

外部变量	名称	数据类型	注释
	J01_Ch1_AIInPV	INT	来自C1JW-AD04U的热电偶输入
	J02_Ch1_Out00	BOOL	对C1JW-OC201的加热用输出
	J02_Ch1_Out04	BOOL	对C1JW-OC201的冷却用输出
	PTIn_Run	BOOL	通过触摸屏输入本示例程序执行标志
	PTIn_ManCtl	BOOL	通过触摸屏输入手动/自动切换标志
	PTIn_SP	REAL	通过触摸屏输入目标值
	PTIn_StartAT	BOOL	通过触摸屏输入自动调谐执行标志
	PTIn_DeadBand	REAL	通过触摸屏输入死区
	PTIn_InitParam	_sINIT_SET_PARAMS	通过触摸屏输入初始设定参数
	PTIn_InitSetOpr_SampTime	LINT	通过触摸屏输入采样周期 (以ms为单位)
	PTIn_OprParam	_sOPR_SET_PARAMS	通过触摸屏输入运行中设定参数
	PTOut_PV	REAL	对触摸屏输出测量值
	PT_PB_Heat	REAL	通过触摸屏输入输出加热操作的比例带
	PT_TI_Heat	LINT	通过触摸屏输入输出加热操作的积分时间(以ms为单位)
	PT_TD_Heat	LINT	通过触摸屏输入输出加热操作的微分时间(以ms为单位)
	PT_PB_Cool	REAL	通过触摸屏输入输出冷却操作的比例带
	PT_TI_Cool	LINT	通过触摸屏输入输出冷却操作的积分时间(以ms为单位)
	PT_TD_Cool	LINT	通过触摸屏输入输出冷却操作的微分时间(以ms为单位)
	PT_ManMV	REAL	通过触摸屏输入输出手动操作量
	PTOut_ATDone	BOOL	对触摸屏输出自动调谐正常结束标志
	PTOut_ATBusy	BOOL	对触摸屏输出自动调谐执行中标志
	PTOut_Error	BOOL	对触摸屏输出异常结束标志
	PTOut_MV	REAL	对触摸屏输出操作量
	PTOut_MVHeat	REAL	对触摸屏输出加热操作的操作量
	PTOut_MVCool	REAL	对触摸屏输出冷却操作的操作量

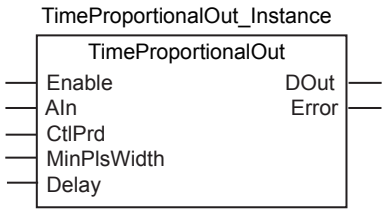
```
// CJ1W-AD04U及触摸屏输入值的单位转换
PV := INT_TO_REAL(J01_Ch1_AIInPV)/REAL#10.0;

PTIn_InitParam.SampTime := NanoSecToTime(PTIn_InitSetOpr_SampTime*1000000);

PB_Heat := PT_PB_Heat;
TI_Heat  := NanoSecToTime(PT_TI_Heat*1000000);
TD_Heat := NanoSecToTime(PT_TD_Heat*1000000);
PB_Cool  := PT_PB_Cool;
TI_Cool   := NanoSecToTime(PT_TI_Cool*1000000);
TD_Cool  := NanoSecToTime(PT_TD_Cool*1000000);
ManMV   := PT_ManMV;

// 执行PIDAT_HeatCool指令
PIDAT_HeatCool_inst(Run                 :=PTIn_Run,
                    ManCtl              :=PTIn_ManCtl,
                    StartAT             :=PTIn_StartAT,
                    PV                  :=PV,
                    SP                  :=PTIn_SP,
                    DeadBand            :=PTIn_DeadBand,
                    OprSetParams        :=PTIn_OprParam,
                    InitSetParams       :=PTIn_InitParam,
                    ProportionalBand_Heat   :=PB_Heat,
                    IntegrationTime_Heat    :=TI_Heat,
```

```
                DerivativeTime_Heat   :=TD_Heat,
                ProportionalBand_Cool :=PB_Cool,
                IntegrationTime_Cool  :=TI_Cool,
                DerivativeTime_Cool   :=TD_Cool,
                ManMV                 :=ManMV,
                CtlPrd_Cool           :=CtlPrd_Cool,
                MV                    =>MV,
                MV_Heat               =>MV_Heat,
                MV_Cool               =>MV_Cool);

//自动调谐执行中转换为分时比例输出的准备
IF PIDAT_HeatCool_inst.ATBusy THEN
  MVHeatTime := MULTIME(CtlPrd_Heat, (MV_Heat/100) );
  MVCoolTime := MULTIME(CtlPrd_Cool, (MV_Cool/100) );
END_IF;

ATHeatPhase := PIDAT_HeatCool_inst.ATBusy & (MVHeatTime>T#0s);
EachCtlPrd_ATHeat_inst(In := ATHeatPhase & NOT(EachCtlPrd_ATHeat_inst.Q),
                       PT:= CtlPrd_Heat);

ATCoolPhase := PIDAT_HeatCool_inst.ATBusy & (MVCoolTime>T#0s);
EachCtlPrd_ATCool_inst(In  := ATCoolPhase & NOT(EachCtlPrd_ATCool_inst.Q),
                       PT:= CtlPrd_Cool);

// 对CJ1W-OC201的加热装置用输出
TPOHeat_inst(Enable :=NOT(PIDAT_HeatCool_inst.ATBusy),
             AIn    :=MV_Heat,
             CtlPrd :=CtlPrd_Heat );
AT_Heat_inst(In := ATHeatPhase & (MVHeatTime<>CtlPrd_Heat) & NOT(EachCtlPrd_ATHeat_inst.Q) ,
             PT:= MVHeatTime);
J02_Ch1_Out00 :=( TPOHeat_inst.DOut ) OR
                ( ATHeatPhase & (MVHeatTime=CtlPrd_Heat)) OR
                ( AT_Heat_inst.Q & ATHeatPhase );

// 对CJ1W-OC201的冷却装置用输出
TPOCool_inst(Enable :=NOT(PIDAT_HeatCool_inst.ATBusy),
             AIn    :=MV_Cool,
             CtlPrd :=CtlPrd_Cool );
AT_Cool_inst(In:= ATCoolPhase & (MVCoolTime<>CtlPrd_Cool) & NOT(EachCtlPrd_ATCool_inst.Q) ,
             PT:= MVCoolTime);
J02_Ch1_Out04 :=( TPOCool_inst.DOut ) OR
                ( ATCoolPhase & (MVCoolTime=CtlPrd_Cool)) OR
                ( AT_Cool_inst.Q & ATCoolPhase );
```

```
// 创建对触摸屏的输出值
PTOut_PV := PV;

PTOut_ATDone := PIDAT_HeatCool_inst.ATDone;
PTOut_ATBusy := PIDAT_HeatCool_inst.ATBusy;
PTOut_Error     := PIDAT_HeatCool_inst.Error;

PTOut_MV       := PIDAT_HeatCool_inst.MV;
PTOut_MVHeat := PIDAT_HeatCool_inst.MV_Heat;
PTOut_MVCool := PIDAT_HeatCool_inst.MV_Cool;

PT_PB_Heat := PB_Heat;
PT_TI_Heat  := TimeToNanoSec(TI_Heat)/1000000;
PT_TD_Heat := TimeToNanoSec(TD_Heat)/1000000;
PT_PB_Cool  := PB_Cool;
PT_TI_Cool   := TimeToNanoSec(TI_Cool)/1000000;
PT_TD_Cool  := TimeToNanoSec(TD_Cool)/1000000;

PT_ManMV   := ManMV;
```

TimeProportionalOut

将操作量转换为分时比例输出。

指令	名称	FB/FUN	图形表现	ST表现
TimeProportionalOut	分时比例输出	FB		TimeProportionalOut_instance(Enable, AIn, CtlPrd, MinPlsWidth, Delay, DOut, Error);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	动作指令	输入	TRUE：执行 FALSE：分时比例输出复位	遵从数据类型	–	FALSE
AIn	操作量		操作量	0 ~ 100	%	0
CtlPrd	控制周期		分时比例输出的控制周期	T#0.1s ~ T#100s	s	T#2s
MinPlsWidth	最小脉冲宽度		最小脉冲宽度	0 ~ 50	%	1
Delay	延迟		ON延迟时间	0 ~ 100	%	0
DOut	分时比例输出	输出	TRUE：分时比例输出ON FALSE：分时比例输出OFF	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
AIn														○						
CtlPrd																○				
MinPlsWidth														○						
Delay														○						
DOut	○																			

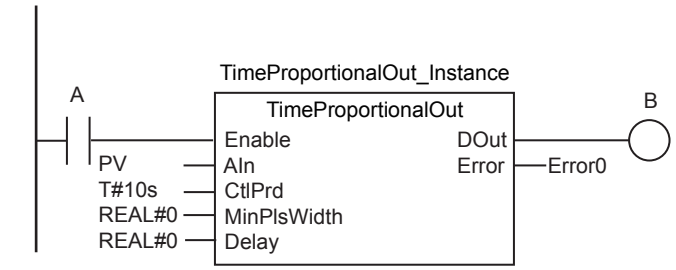
功能

将PID控制等的操作量转换为分时比例输出。分时比例输出是指使操作量与ON、OFF的时间比成正比进行转换的输出。

动作指令 “Enable”为 TRUE 期间，将操作量 “AIn”的值转换为控制周期 “CtlPrd”的分时比例输出 “DOut”。“Enable”为FALSE时，将分时比例输出复位。“DOut”和 “Error”为FALSE。“Enable”由FALSE变为TRUE时，反映 “CtlPrd”、“MinPlsWidth”和 “Delay”的值。

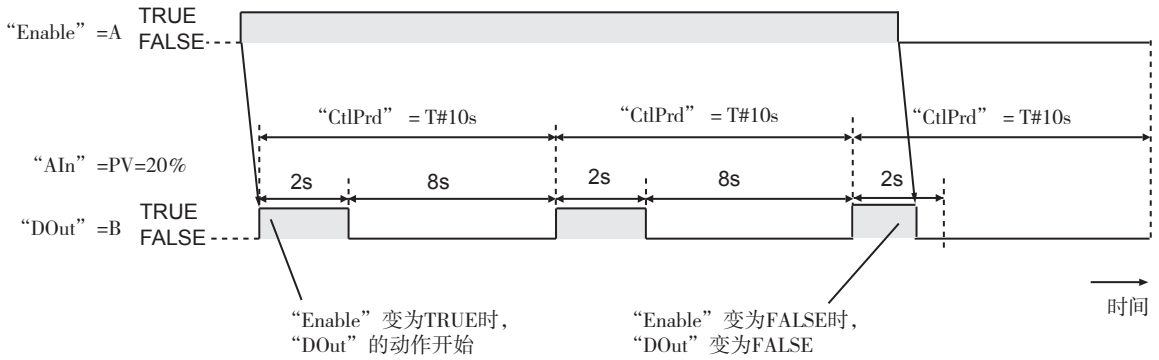
“CtlPrd”的值为10s，“AIn”的值为20%时的记述示例如下所示。“Enable”为TRUE期间，“DOut”在2s内变为TRUE，在接下来的8s内变为FALSE。在10s周期内重复上述操作。

LD



ST

```
TimeProportionalOut_instance(A,PV,T#10s,REAL#0,REAL#0,B,Error0);
```



分时比例输出 “DOut”的分辨率

将 “AIn”的值转换为 “DOut”的最小单位称为 “DOut”的分辨率。
“AIn”值的分辨率高于 “DOut”的分辨率时，根据 “DOut”的分辨率，将取整后的 “AIn”值转换为“DOut”。

“DOut”的分辨率通过下式计算。

“DOut”的分辨率(%)=任务周期 ÷ “CtlPrd” × 100

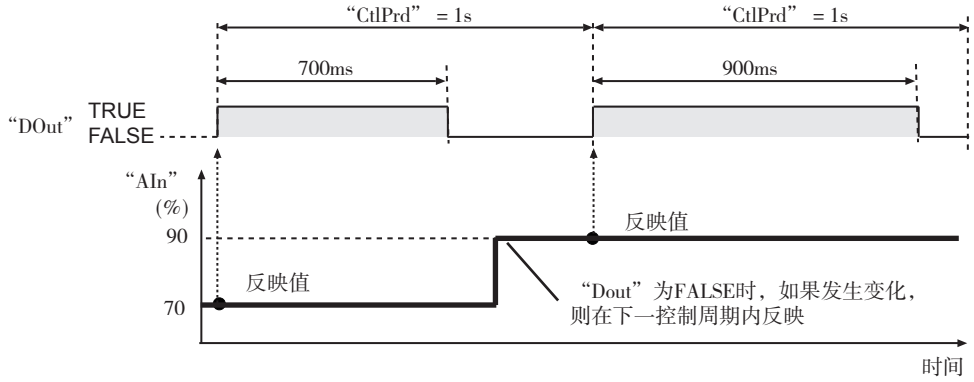
例如，任务周期为1ms，“CtlPrd”的值为1s时，“DOut”的分辨率变为0.1%。此时，对 “AIn”值的小数点后2位之后的数值进行取整。

操作量 “AIn”值的反映时间

“AIn”值的反映时间因 “DOut”为FALSE或TRUE而异。

● “DOut”为FALSE时

“DOut”为FALSE期间，“AIn”的值发生变化时，在下一控制周期内反映。

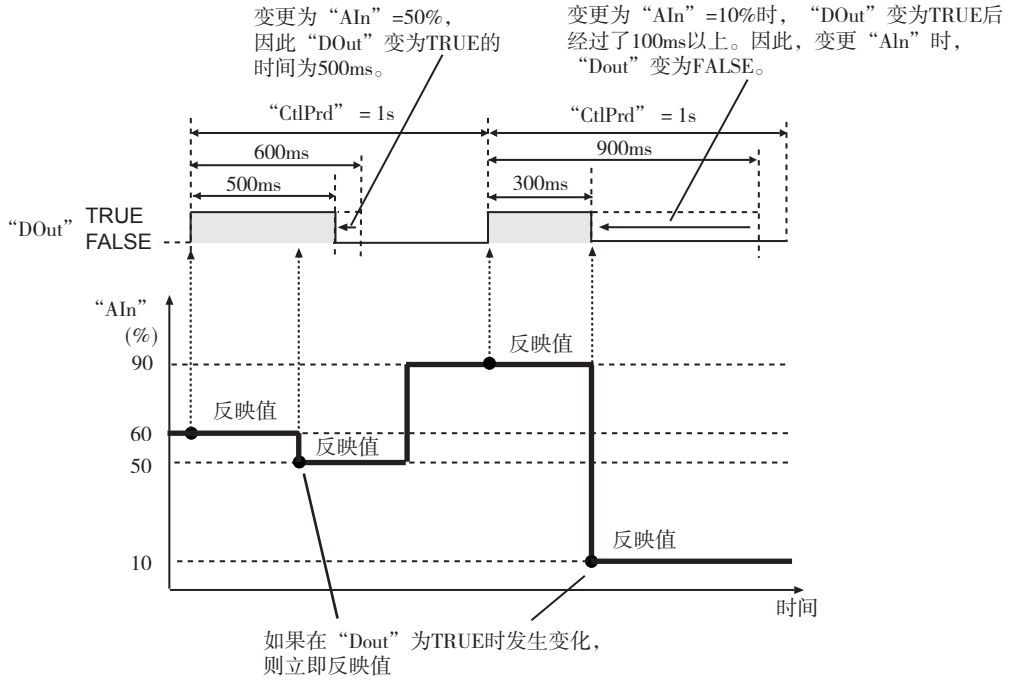


● “DOut”为TRUE时

“DOut”为TRUE期间，“AIn”的值发生变化时，直接反映。

例如，控制周期 “CtlPrd”的值为1s时，动作如下所示。

- 控制周期开始时，“AIn”的值为60%的情况下，如果 “DOut”为TRUE期间 “AIn”的值变为50%，则仅500ms内 “DOut”变为TRUE。
- 假设控制周期开始时，“AIn”的值为90%的情况下，“DOut”变为TRUE，经过300ms后，“AIn”的值变为10%。此时，相当于已经过100ms的10%，因此 “DOut”立即变为FALSE。



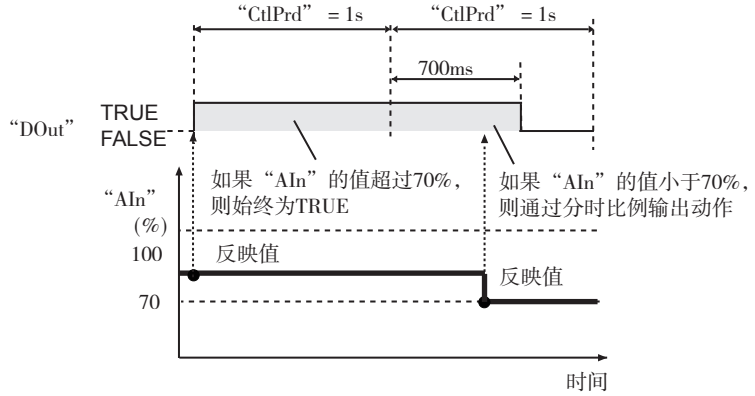
基于最小脉冲宽度 “MinPlsWidth”的分时比例输出 “DOut”的动作

最小脉冲宽度是指保持 “DOut”为TRUE或FALSE值的时间最小值。通过设定最小脉冲宽度“MinPlsWidth”，具有降低 “DOut”抖动的效果。例如，可通过在冷却控制中减少切换风扇ON/OFF的次数来降低消耗功率。

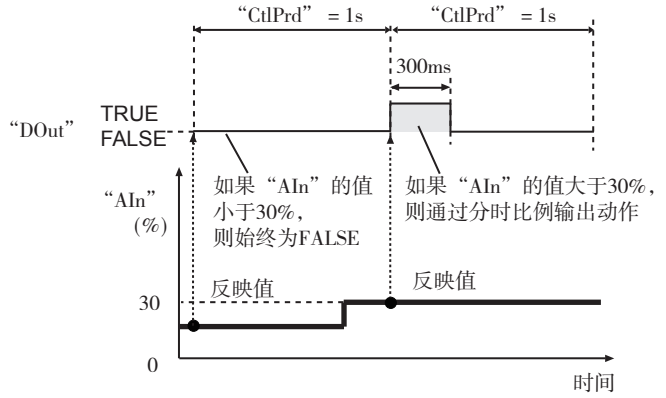
根据 “MinPlsWidth”与 “AIn”值的关系，“DOut”的动作如下所示。

“MinPlsWidth”与 “AIn”值的关系	“DOut”的动作
“AIn” < “MinPlsWidth”	始终为FALSE
“MinPlsWidth” ≤ “AIn” ≤100– “MinPlsWidth”	通过分时比例输出
“AIn” > 100– “MinPlsWidth”	始终为TRUE

例如，“MinPlsWidth”为30%时，“DOut”的动作如下图所示。
“AIn”的值超过70%期间，始终为TRUE；如果小于70%，则变为分时比例输出。



“AIn”的值小于30%期间，始终为FALSE；如果大于30%，则变为分时比例输出。



基于延迟 “Delay”的分时比例输出 “DOut”的动作

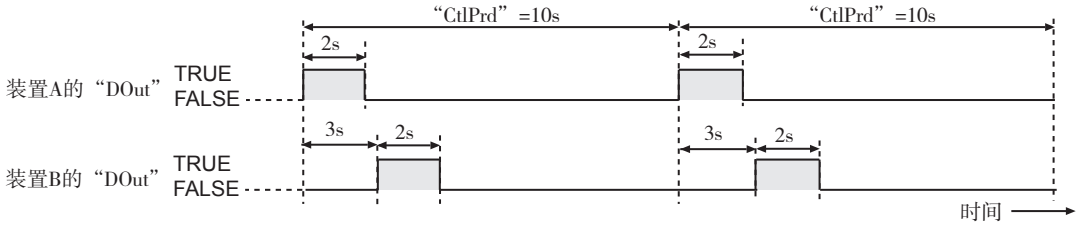
从控制周期开始，经过设定时间后，将 “DOut”设为TRUE的操作称为延迟。使用多个本指令时，可通过设定 “Delay”，将各 “DOut”变为TRUE的时间错开。因此，具有降低 “DOut”同时变为ON的可能性的效果。例如，使多个加热器动作时，通过 “Delay”将至各加热器的输出变为ON的时间错开，可降低同时使用的功率。

从控制周期开始，经过延迟 “Delay”的比例时间后，“DOut”变为TRUE。

例如，对相同控制周期的装置A、装置B设定如下所示的值。

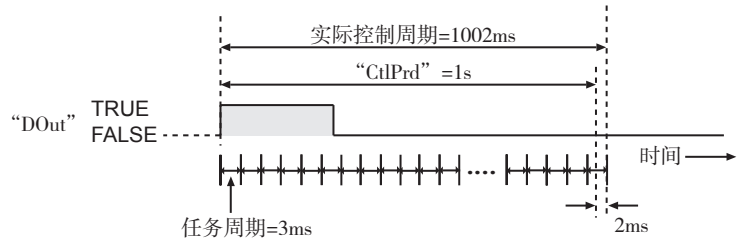
装置	“Delay”的值	“AIn”的值	“CtlPrd”的值
装置A	0%	20%	10s
装置B	30%		

装置A的 “DOut”在控制周期开始时变为TRUE。装置B的 “DOut”在控制周期开始3s后变为TRUE。

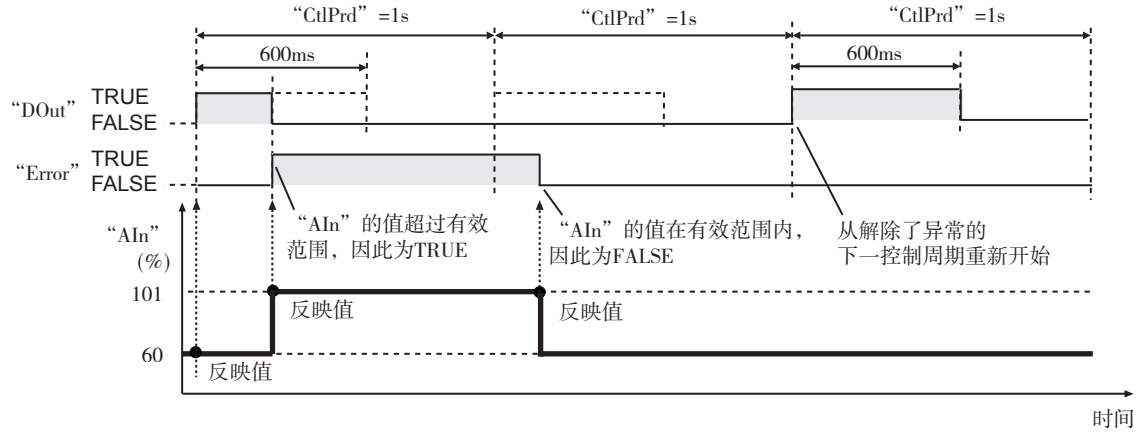


使用注意事项

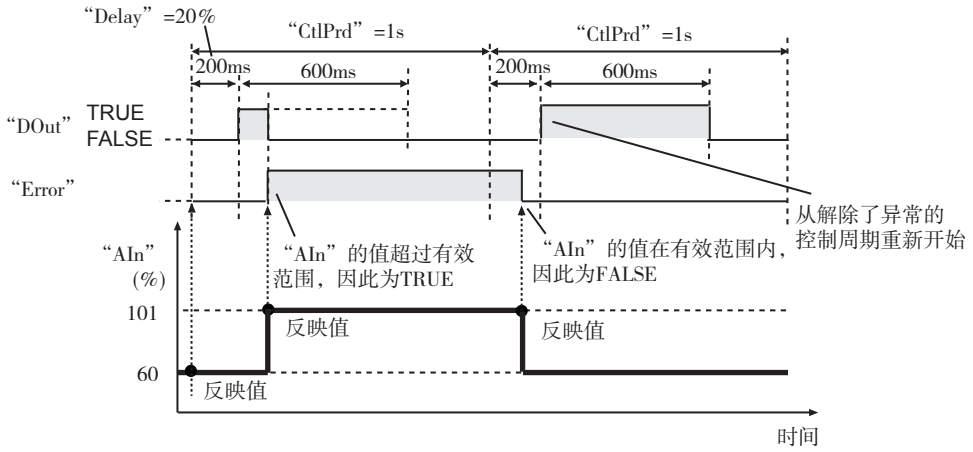
- 请将控制周期 “CtlPrd”的值设定为分配了程序的任务周期的整数倍。设定了非任务周期整数倍的值时，控制周期 “CtlPrd”结束后至下一任务执行时间前为实际控制周期。例如，任务周期为3ms，“CtlPrd”的值为1s时，“CtlPrd”结束后至下一任务执行时间前的1002ms为实际控制周期。



- 请设定任务周期和控制周期 “CtlPrd”的值，以使 “DOut”的分辨率低于0.1%。如果 “DOut”的分辨率超过 0.1%，则 “DOut”变为 TRUE 的比例与 “AIn”值的误差会增大，从而导致控制性能降低。例如，“CtlPrd”为10s时，请将任务周期设定为小于10ms。
- 如需使用多个本指令同步各控制周期时，则请在相同程序中使用本指令。如果在不同程序中使用本指令，则会因程序的执行时间而导致控制周期偏移。
- “Enable”的值变为TRUE至 “DOut”开始动作的时间差不定。
- “AIn”、“CtlPrd”、“MinPlsWidth”、“Delay”的值超过有效范围时，会发生异常。“Error”为TRUE，“DOut”为FALSE。如下所示，“AIn”的值超过有效范围时，“DOut”的动作因解除异常的时间而异。
  - 经过 “DOut”变为TRUE的时间后解除异常时，“DOut”从下一控制周期起重新开始分时比例输出。



- 在 “DOut”变为TRUE的时间前解除异常时，“DOut”从解除异常的控制周期起重新开始分时比例输出。

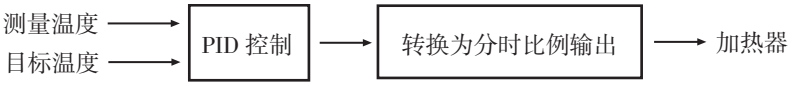


版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

进行上下限警报、带上下限偏差警报的4点温度控制。使用PID控制方式，将PID控制的操作量转换为分时比例输出，输出至加热器。



规格

通过下表所示规格，进行温度控制。

项目	规格
输入单元	绝缘型高分辨率多功能输入单元 CJ1W-PH41U
输入类型	均为K型热电偶
输出单元	晶体管输出单元 CJ1W-OD212
目标温度	100℃
上限温度	200℃
下限温度	0℃
上下限警报的滞后	5℃
上限偏差温度	50℃
下限偏差温度	50℃
上下限偏差警报的滞后	3℃
PID运算的采样周期	100ms
输出控制周期	1s

构成・设定

输入单元CJ1W–PH41U的设定如下所示。

项目名称	设定值
Input1:Input signal type	K(1)
Input2:Input signal type	K(1)
Input3:Input signal type	K(1)
Input4:Input signal type	K(1)

I/O映射设定如下所示。

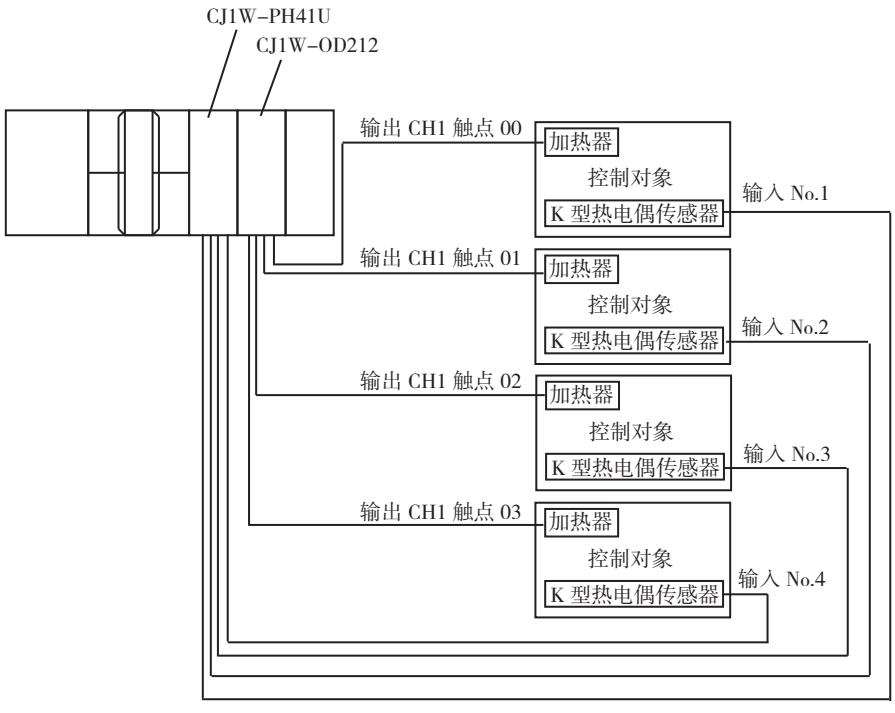
单元	端口	说明	变量
CJ1W–PH41U	Ch1_AIInPV	输入No.1测量值(INT型)	AI1
	Ch2_AIInPV	输入No.2测量值(INT型)	AI2
	Ch3_AIInPV	输入No.3测量值(INT型)	AI3
	Ch4_AIInPV	输入No.4测量值(INT型)	AI4
CJ1W–OD212	Ch1_Out00	输出CH1触点00	DO1
	Ch1_Out01	输出CH1触点01	DO2
	Ch1_Out02	输出CH1触点02	DO3
	Ch1_Out03	输出CH1触点03	DO4

4点温度控制的输入与输出的对应关系如下所示。

输入	输出
AI1	DO1
AI2	DO2
AI3	DO3
AI4	DO4

将待分配程序的任务周期设定为1ms。

● 结构图

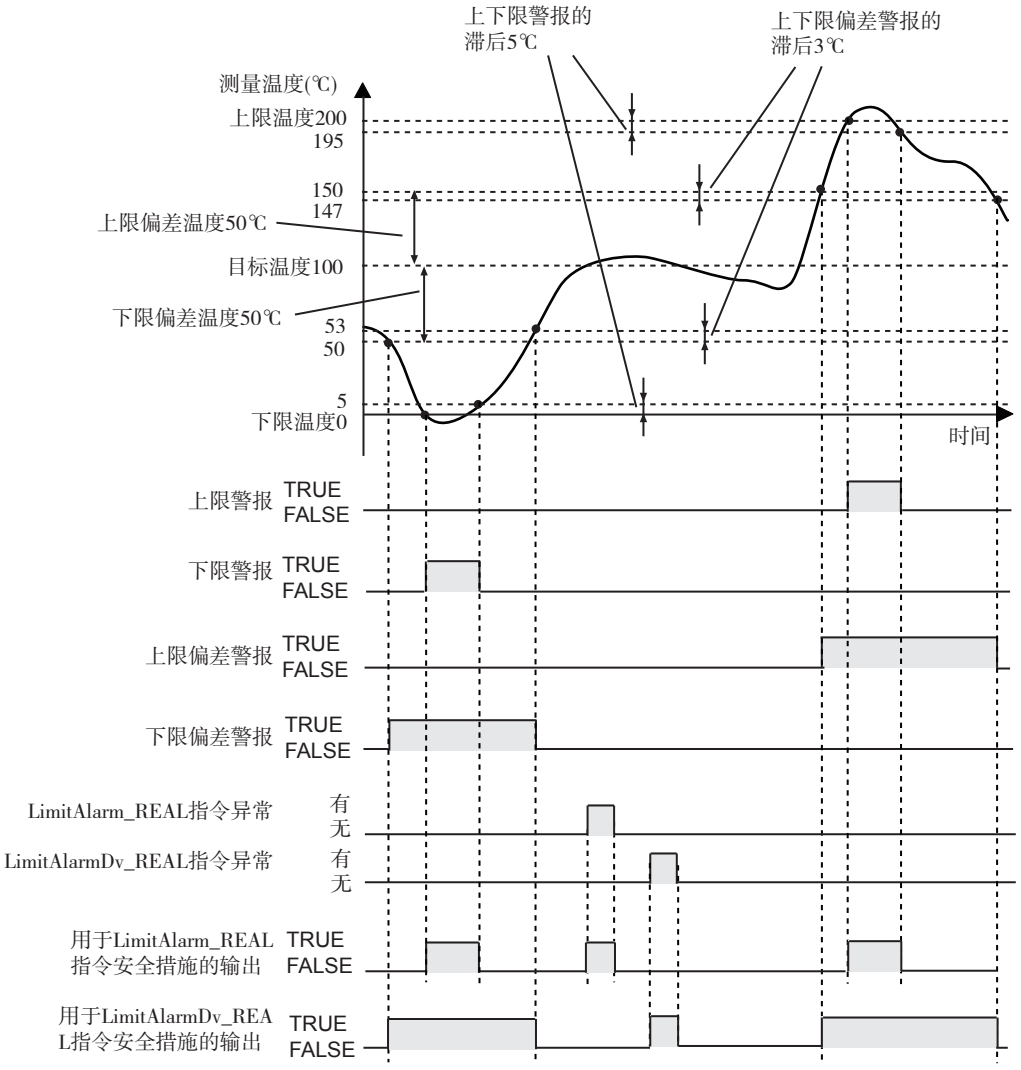


处理内容

进行4点以下处理。

1. 获取测量温度。
2. 使用LimitAlarm_REAL指令，针对测量温度输出上下限警报。
3. LimitAlarm_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出。
4. 使用LimitAlarmDv_REAL指令，针对测量温度与目标温度的偏差，输出上下限偏差警报。
5. LimitAlarmDv_REAL指令下发生异常或输出上下限偏差警报时，进行用于安全措施的输出。
6. 使用PIDAT指令进行温度控制。
7. 使用TimeProportionalOut指令，通过分时比例将操作量输出至加热器。

- 上下限警报、上下限偏差警报的动作



应用程序

全局变量的定义

全局变量

名称	数据类型	分配对象*1	注释
AI1	INT	IOBus://rack#0/slot#0/Ch1_AIInPV	输入No.1测量值(INT型)
AI2	INT	IOBus://rack#0/slot#0/Ch2_AIInPV	输入No.2测量值(INT型)
AI3	INT	IOBus://rack#0/slot#0/Ch3_AIInPV	输入No.3测量值(INT型)
AI4	INT	IOBus://rack#0/slot#0/Ch4_AIInPV	输入No.4测量值(INT型)
DO1	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out00	输出CH1触点00
DO2	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out01	输出CH1触点01
DO3	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out02	输出CH1触点02
DO4	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out03	输出CH1触点03

*1 将CJ1W-PH41U安装至机架编号0的插槽编号0，将CJ1W-OD212安装至机架编号0的插槽编号1时的分配对象。

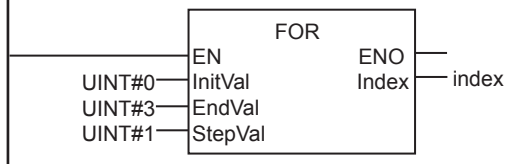
(注) 通过设定I/O映射，自动生成分配至各单元端口的全局变量。

LD

名称	数据类型	初始值	保持	注释
index	UINT	0	□	循环索引
LimitAlarm_ON	BOOL	True	□	执行上下限警报
LimitAlarmDv_ON	BOOL	True	□	执行上下限偏差警报
TimeProportional Out_ON	BOOL	True	□	执行分时比例输出
AI	INT	0	□	当前值
PV	ARRAY[0..3] OF REAL	[4(0.0)]	□	测量值
SP	ARRAY[0..3] OF REAL	[4(100)]	□	目标温度
DOut_TPO	BOOL	False	□	分时比例输出
HighVal	ARRAY[0..3] OF REAL	[4(200)]	□	上下限警报的上限设定值
LowVal	ARRAY[0..3] OF REAL	[4(0.0)]	□	上下限警报的下限设定值
Hystrs_LimitAlarm	ARRAY[0..3] OF REAL	[4(5)]	□	上下限警报的滞后
Q_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	上下限警报输出
HighAlm	ARRAY[0..3] OF BOOL	[4(False)]	□	上限警报
LowAlm	ARRAY[0..3] OF BOOL	[4(False)]	□	下限警报
Error_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	LimitAlarm_REAL指令异常
Alm_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	用于上下限警报安全措施的输出
DvHighVal	ARRAY[0..3] OF REAL	[4(50)]	□	上下限偏差警报的上限偏差设定值
DvLowVal	ARRAY[0..3] OF REAL	[4(50)]	□	上下限偏差警报的下限偏差设定值
Q_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	输出上下限偏差警报
HighAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	上限偏差警报
LowAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	下限偏差警报
Error_LimitAlarm Dv	ARRAY[0..3] OF BOOL	[4(False)]	□	LimitAlarmDv_REAL指令异常

名称	数据类型	初始值	保持	注释
Hystrs_LimitAlarmDv	ARRAY[0..3] OF REAL	[4(3)]	☐	上下限偏差警报的滞后
Alm_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	☐	用于上下限偏差警报安全措施的输出
Run	ARRAY[0..3] OF BOOL	[4(False)]	☐	执行条件
ManCtl	ARRAY[0..3] OF BOOL	[4(False)]	☐	手动/自动切换
StartAT	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行条件
OprSetParams	_sOPR_SET_PARAMS	(MVLowLmt:=0.0，MVUpLmt:=100.0，ManResetVal:=0.0，MVTrackSw:=False，MVTrackVal:=0.0，StopMV:=0.0，ErrorMV:=0.0，Alpha:=0.65，ATCalcGain:=1.0，ATHystrs:=0.2)	☐	运行中设定参数
InitSetParams	_sINIT_SET_PARAMS	(SampTime:=T#100ms，RngLowLmt:=-10.0，RngUpLmt:=1000.0，DirOpr:=False)	☐	初始设定参数
PB	ARRAY[0..3] OF REAL	[4(10)]	☑	比例带
TI	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	积分时间
TD	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	微分时间
ManMV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	手动操作量
ATDone	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐正常结束
ATBusy	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行中
Error_PIDAT	ARRAY[0..3] OF BOOL	[4(False)]	☐	PIDAT指令异常
ErrorID	ARRAY[0..3] OF WORD	[4(16#0)]	☐	PIDAT指令的异常ID
MV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	操作量
CtlPrd	ARRAY[0..3] OF TIME	[4(T#1s)]	☐	控制周期
MinPlsWidth	ARRAY[0..3] OF REAL	[4(0.0)]	☐	最小脉冲宽度
Delay	ARRAY[0..3] OF REAL	[4(0.0)]	☐	ON延迟时间
Error_TimeProportionalOut	ARRAY[0..3] OF BOOL	[4(False)]	☐	TimeProportionalOut指令异常
LimitAlarm_REAL_instance	LimitAlarm_REAL		☐	
LimitAlarmDv_REAL_instance	LimitAlarmDv_REAL		☐	
PIDAT_instance	PIDAT		☐	
TimeProportionalOut_instance	TimeProportionalOut		☐	

4点温度控制

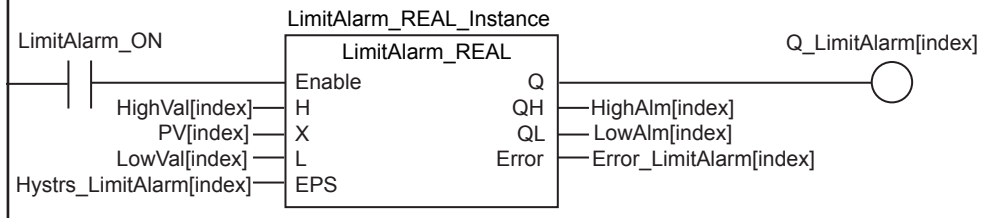


获取当前值

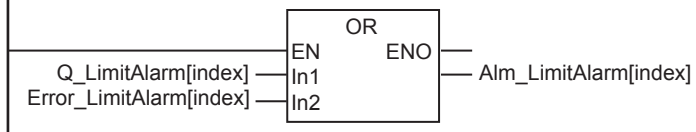
内联ST

注. 内联ST的相关内容请参阅下文的内联ST(1)

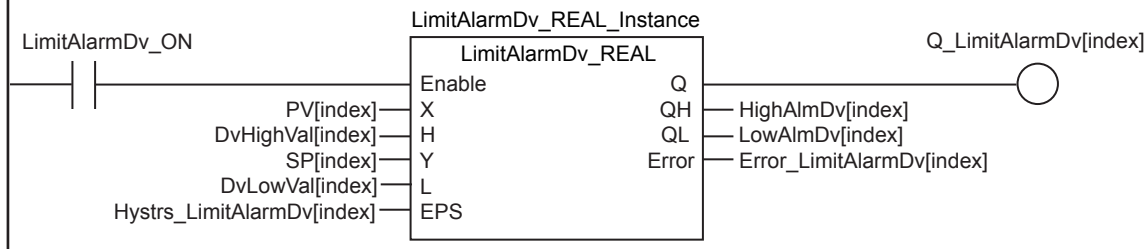
上下限警报



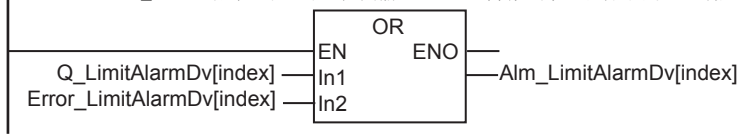
LimitAlarm_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出

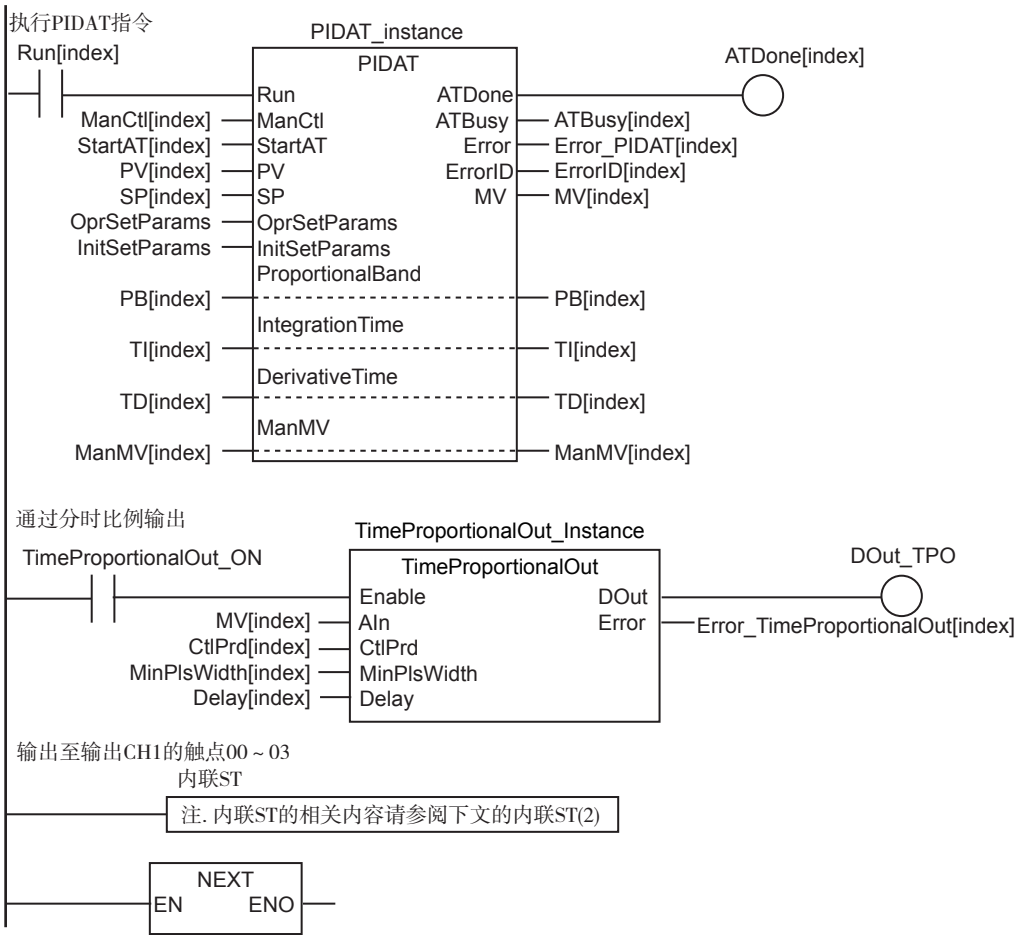


上下限偏差警报



LimitAlarmDv_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出





● 内联ST (1)的内容

```
//获取输入1 ~ 4的值
CASE index OF
    INT#0:
        AI:=AI1;
    INT#1:
        AI:=AI2;
    INT#2:
        AI:=AI3;
    ELSE
        AI:=AI4;
END_CASE;

//将当前值AI转换为实数
PV[index]:=INT_TO_REAL(AI)/REAL#10.0; // CJ1W-PH41U的输出为测量温度的10倍，因此除以10.0
```

● 内联ST (2)的内容

```
//输出至输出CH1的触点00 ~ 03
CASE index OF
    INT#0:
        DO1:=DOut_TPO;
    INT#1:
        DO2:=DOut_TPO;
    INT#2:
        DO3:=DOut_TPO;
    ELSE
        DO4:=DOut_TPO;
END_CASE;
```

ST

名称	数据类型	初始值	保持	注释
index	UINT	0	☐	循环索引
LimitAlarm_ON	BOOL	True	☐	执行上下限警报
LimitAlarmDv_ON	BOOL	True	☐	执行上下限偏差警报
TimeProportional Out_ON	BOOL	True	☐	执行分时比例输出
AI	INT	0	☐	当前值
PV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	测量值
SP	ARRAY[0..3] OF REAL	[4(100)]	☐	目标温度
DOut_TPO	BOOL	False	☐	分时比例输出
HighVal	ARRAY[0..3] OF REAL	[4(200)]	☐	上下限警报的上限设定值
LowVal	ARRAY[0..3] OF REAL	[4(0.0)]	☐	上下限警报的下限设定值
Hystrs_LimitAlarm	ARRAY[0..3] OF REAL	[4(5)]	☐	上下限警报的滞后
Q_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	☐	上下限警报输出
HighAlm	ARRAY[0..3] OF BOOL	[4(False)]	☐	上限警报
LowAlm	ARRAY[0..3] OF BOOL	[4(False)]	☐	下限警报
Error_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	☐	LimitAlarm_REAL指令异常
Alm_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	☐	用于上下限警报安全措施的输出
DvHighVal	ARRAY[0..3] OF REAL	[4(50)]	☐	上下限偏差警报的上限偏差设定值
DvLowVal	ARRAY[0..3] OF REAL	[4(50)]	☐	上下限偏差警报的下限偏差设定值
Q_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	☐	输出上下限偏差警报
HighAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	☐	上限偏差警报
LowAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	☐	下限偏差警报
Error_LimitAlarm Dv	ARRAY[0..3] OF BOOL	[4(False)]	☐	LimitAlarmDv_REAL指令异常
Hystrs_LimitAlarm Dv	ARRAY[0..3] OF REAL	[4(3)]	☐	上下限偏差警报的滞后
Alm_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	☐	用于上下限偏差警报安全措施的输出
Run	ARRAY[0..3] OF BOOL	[4(False)]	☐	执行条件
ManCtl	ARRAY[0..3] OF BOOL	[4(False)]	☐	手动/自动切换
StartAT	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行条件
OprSetParams	_sOPR_SET_PARAMS	(MVLowLmt:=0.0，MVUpLmt:=100.0，ManResetVal:=0.0，MVTrackSw:=False，MVTrackVal:=0.0，StopMV:=0.0，ErrorMV:=0.0，Alpha:=0.65，ATCalcGain:=1.0，ATHystrs:=0.2)	☐	运行中设定参数
InitSetParams	_sINIT_SET_PARAMS	(SampTime:=T#100ms，RngLowLmt:=-10.0，RngUpLmt:=1000.0，DirOpr:=False)	☐	初始设定参数
PB	ARRAY[0..3] OF REAL	[4(10)]	☑	比例带
TI	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	积分时间
TD	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	微分时间
ManMV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	手动操作量
ATDone	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐正常结束
ATBusy	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行中

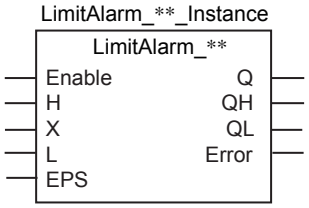
名称	数据类型	初始值	保持	注释
Error_PIDAT	ARRAY[0..3] OF BOOL	[4(False)]	□	PIDAT指令异常
ErrorID	ARRAY[0..3] OF WORD	[4(16#0)]	□	PIDAT指令的异常ID
MV	ARRAY[0..3] OF REAL	[4(0.0)]	□	操作量
CtlPrd	ARRAY[0..3] OF TIME	[4(T#1s)]	□	控制周期
MinPlsWidth	ARRAY[0..3] OF REAL	[4(0.0)]	□	最小脉冲宽度
Delay	ARRAY[0..3] OF REAL	[4(0.0)]	□	ON延迟时间
Error_Time ProportionalOut	ARRAY[0..3] OF BOOL	[4(False)]	□	TimeProportionalOut指令异常
LimitAlarm_REAL _instance	LimitAlarm_REAL		□	
LimitAlarmDv _REAL_instance	LimitAlarmDv_REAL		□	
PIDAT_instance	PIDAT		□	
TimeProportional Out_instance	TimeProportionalOut		□	

```
//4点温度控制
FOR index:=UINT#0 TO UINT#3 BY UINT#1 DO

 //获取输入1～4的值
   CASE index OF
      INT#0:
         AI:=AI1;
      INT#1:
         AI:=AI2;
      INT#2:
         AI:=AI3;
      ELSE
         AI:=AI4;
   END_CASE;

//将当前值AI转换为实数
PV[index]:=INT_TO_REAL(AI)/REAL#10.0;   // CJ1W-PH41U的输出为测量温度的10倍，因此除以10.0

//上下限警报
LimitAlarm_REAL_instance(
  Enable :=LimitAlarm_ON,
  H      :=HighVal[index],
  X      :=PV[index],
  L      :=LowVal[index],
  EPS    :=Hystrs_LimitAlarm[index],
  Q      =>Q_LimitAlarm[index],
  QH     =>HighAlm[index],
  QL     =>LowAlm[index],
  Error  =>Error_LimitAlarm[index]);

//LimitAlarm_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出
Alm_LimitAlarm[index]:=Q_LimitAlarm[index] OR Error_LimitAlarm[index];

//上下限偏差警报
LimitAlarmDv_REAL_instance(
  Enable :=LimitAlarmDv_ON,
  X      :=PV[index],
  H      :=DvHighVal[index],
  Y      :=SP[index],
  L      :=DvLowVal[index],
```

```
    EPS     :=Hystrs_LimitAlarmDv[index],
    Q       =>Q_LimitAlarmDv[index],
    QH      =>HighAlmDv[index],
    QL      =>LowAlmDv[index],
    Error   =>Error_LimitAlarmDv[index]);

  //LimitAlarmDv_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出
  Alm_LimitAlarmDv[index]:=Q_LimitAlarmDv[index] OR Error_LimitAlarmDv[index];

  // 执行PIDAT指令
  PIDAT_instance(
    Run                 :=Run[index],
    ManCtl              :=ManCtl[index],
    StartAT             :=StartAT[index],
    PV                  :=PV[index],
    SP                  :=SP[index],
    OprSetParams        :=OprSetParams,
    InitSetParams       :=InitSetParams,
    ProportionalBand    :=PB[index],
    IntegrationTime     :=TI[index],
    DerivativeTime      :=TD[index],
    ManMV               :=ManMV[index],
    ATDone              =>ATDone[index],
    ATBusy              =>ATBusy[index],
    Error               =>Error_PIDAT[index],
    ErrorID             =>ErrorID[index],
    MV                  =>MV[index]);

  // 通过分时比例输出
  TimeProportionalOut_instance(
    Enable          :=TimeProportionalOut_ON,
    AIn             :=MV[index],
    CtlPrd          :=CtlPrd[index],
    MinPlsWidth     :=MinPlsWidth[index],
    Delay           :=Delay[index],
    DOut            =>DOut_TPO,
    Error           =>Error_TimeProportionalOut[index]);

  //输出至输出Ch1的触点00 ~ 03
  CASE index OF
      INT#0:
          DO1:=DOut_TPO;
      INT#1:
          DO2:=DOut_TPO;
      INT#2:
          DO3:=DOut_TPO;
      ELSE
          DO4:=DOut_TPO;
  END_CASE;

END_FOR;
```

LimitAlarm_**

输入值小于下限设定值或超过上限设定值时输出警报。

指令	名称	FB/FUN	图形表现	ST表现
LimitAlarm_**	上下限警报组	FB	 **为REAL、LREAL中的任意一个	LimitAlarm_**_instance(Enable, H, X, L, EPS, Q, QH, QL, Error); **为REAL、LREAL中的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	动作指令	输入	TRUE：执行 FALSE：警报复位	遵从数据类型	–	FALSE
H	上限设定值		输入值的上限设定值			0
X	输入值		监控对象值			
L	下限设定值		输入值的下限设定值			
EPS	滞后		警报的滞后	遵从数据类型*1		
Q	报警输出	输出	TRUE：上限警报和下限警报中的任意一个变为ON FALSE：上限警报和下限警报均变为OFF	遵从数据类型	–	–
QH	上限警报		TRUE：上限警报ON FALSE：上限警报OFF			
QL	下限警报		TRUE：下限警报ON FALSE：下限警报OFF			

*1　不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
H														○	○					
X	与 “H” 相同的数据类型																			
L	与 “H” 相同的数据类型																			
EPS	与 “H” 相同的数据类型																			
Q	○																			
QH	○																			
QL	○																			

功能

监控输入值是否为下限设定值到上限设定值之间的数值。输入值小于下限设定值或超过上限设定值时输出警报。用于通过温度控制监控测量温度等用途。

动作指令 “Enable”为TRUE期间，监控输入值 “X”的值。“X”的值超过上限设定值 “H”的值时，上限警报 “QH”为TRUE。“X”的值小于下限设定值 “L”的值时，下限警报 “QL”为TRUE。“QH”、“QL”中的任一值为TRUE时，警报输出 “Q”变为TRUE。“Enable”为TRUE期间，始终反映 “X”、“H”、“L”、滞后 “EPS”的值。

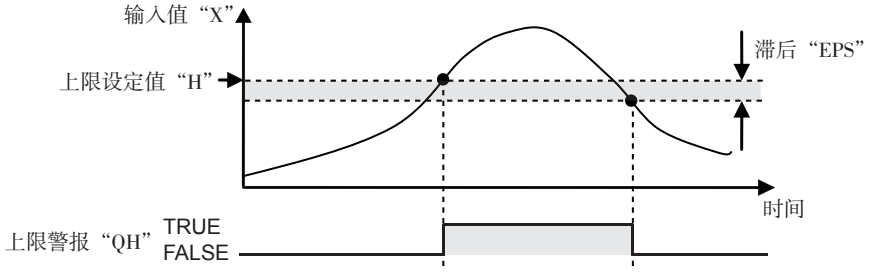
“Enable”为FALSE时，将警报复位。通过复位，“Q”、“QH”、“QL”均为FALSE。

“H”、“X”、“L”、“EPS”的数据类型为REAL、LREAL中的任意一个。指令名称因 “H”、“X”、“L”、“EPS”的数据类型而异。例如，指令名称为LimitAlarm_LREAL时，“H”、“X”、“L”、“EPS”的数据类型均为LREAL。

上限警报 “QH”的动作

上限警报 “QH”值的变化如下所示。可通过设定滞后，防止警报震荡。

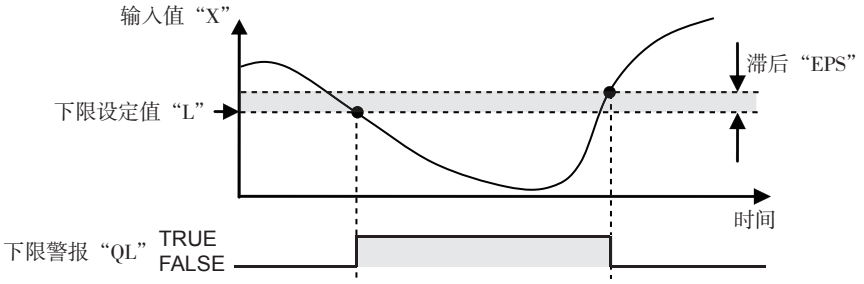
- 输入值 “X” > 上限设定值 “H”时，为TRUE。
- 输入值 “X” < 上限设定值 “H” –滞后 “EPS”时，为FALSE。



下限警报 “QL”的动作

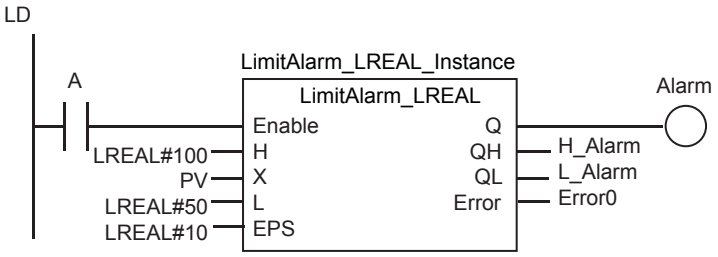
下限警报 “QL”值的变化如下所示。可通过设定滞后，防止警报震荡。

- 输入值 “X” < 下限设定值 “L”时，为TRUE。
- 输入值 “X” > 下限设定值 “L” +滞后 “EPS”时，为FALSE。

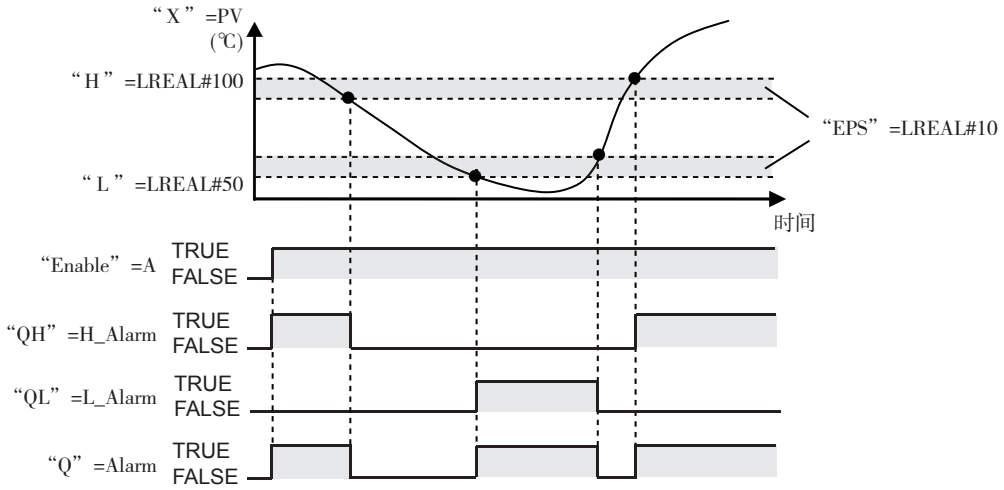


记述示例

上限设定值 “H”为100℃，下限设定值 “L”为50℃，滞后 “EPS”为10℃时的记述示例如下所示。

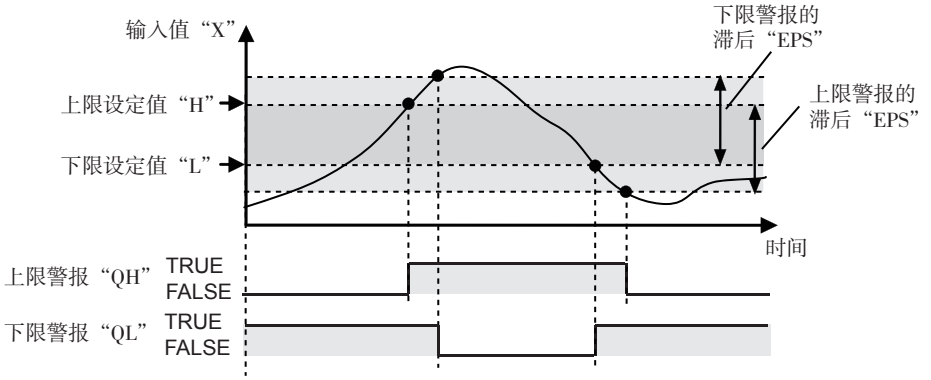



```
ST
 LimitAlarm_LREAL_instance(A,LREAL#100,PV,LREAL#50,LREAL#10,Alarm,H_Alarm,L_Alarm,Error0);
```

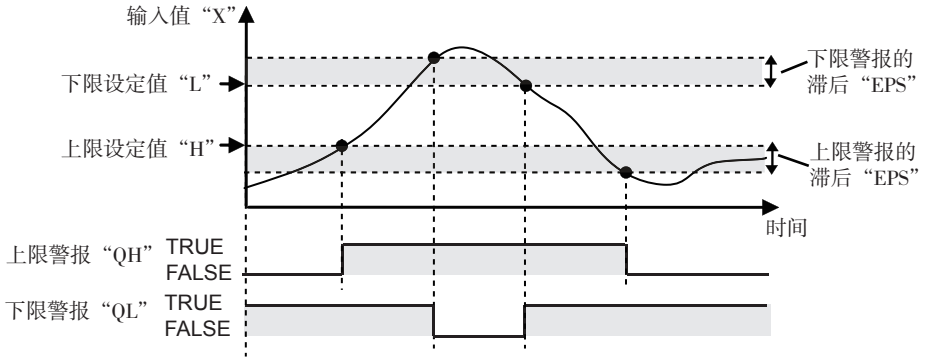


参考

- 如需缩短指令执行时间，则请使用LimitAlarm_REAL指令。
- 也可设定为 “H” – “L” < “EPS”。此时，“QH”和 “QL”可能均会变为TRUE。



- 也可设定为 “H” < “L”。此时，“QH”或 “QL”中的任意一个始终为TRUE。



使用注意事项

- “EPS”的值超过有效范围时，会发生异常。“Error”为TRUE，“Q”、“QH”、“QL”为FALSE。
- 可用于输出警报时关闭温度控制的输出等对本指令采取安全措施的用途。此时，即使 “Q”、“QH”及“QL”因异常变为FALSE，使用时仍请确保可采取安全措施。使用示例请参阅 “TimeProportionalOut指令(P.2–719)”的示例程序。

版本相关信息

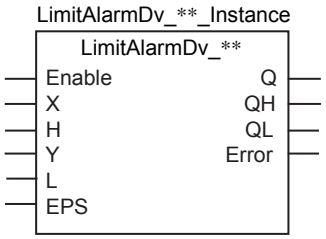
本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

请参阅 “TimeProportionalOut指令(P.2–719)”的示例程序。

LimitAlarmDv_**

输入值与基准值的偏差低于下限偏差设定值或超过上限偏差设定值时输出警报。

指令	名称	FB/FUN	图形表现	ST表现
LimitAlarmDv_**	上下限偏差警报组	FB	 **为REAL、LREAL中的任意一个	LimitAlarmDv_**instance(Enable, X, H, Y, L, EPS, Q, QH, QL, Error); **为REAL、LREAL中的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	动作指令	输入	TRUE：执行 FALSE：警报复位	遵从数据类型	–	FALSE
X	输入值		监控对象值			0
H	上限偏差设定值		相对于基准值的上方的偏差警报设定值			
Y	基准值		偏差的基准值			
L	下限偏差设定值		相对于基准值的下方的偏差警报设定值			
EPS	滞后		警报的滞后	遵从数据类型*1		
Q	偏差警报输出	输出	TRUE：上限偏差警报和下限偏差警报中的任意一个变为ON FALSE：上限偏差警报和下限偏差警报均变为OFF	遵从数据类型	–	–
QH	上限偏差警报		TRUE：上限偏差警报ON FALSE：上限偏差警报OFF			
QL	下限偏差警报		TRUE：下限偏差警报ON FALSE：下限偏差警报OFF			

*1 不含负数

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
X														○	○					
H	与“X”相同的数据类型																			
Y	与“X”相同的数据类型																			
L	与“X”相同的数据类型																			
EPS	与“X”相同的数据类型																			

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Q	○																			
QH	○																			
QL	○																			

功能

监控输入值与基准值的偏差是否超过下限偏差设定值或上限偏差设定值。偏差超过下限偏差设定值或上限偏差设定值时输出警报。用于通过温度控制监控测量温度与目标温度的偏差等用途。

动作指令 “Enable”为TRUE期间，监控输入值 “X”与基准值 “Y”的偏差。“X”上方与 “Y”的偏差超过上限偏差设定值 “H”的值时，上限偏差警报 “QH”为TRUE。“X”下方与 “Y”的偏差超过下限偏差设定值 “L”的值时，下限偏差警报 “QL”为TRUE。“QH”、“QL”中的任一值为TRUE时，警报输出“Q”变为TRUE。“Enable”为TRUE期间，始终反映“X”、“H”、“Y”、“L”、滞后 “EPS”的值。

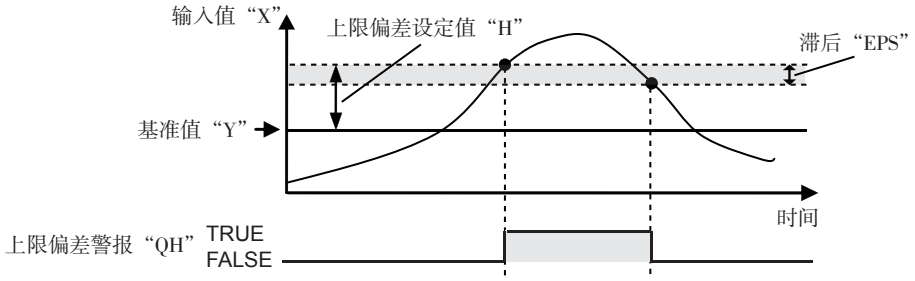
“Enable”为FALSE时，将警报复位。通过复位，“Q”、“QH”、“QL”均为FALSE。

“X”、“H”、“Y”、“L”、“EPS”的数据类型为REAL、LREAL中的任意一个。指令名称因 “X”、“H”、“Y”、“L”、“EPS”的数据类型而异。例如，指令名称为LimitAlarmDv_LREAL时，“X”、“H”、“Y”、“L”、“EPS”的数据类型均为LREAL。

上限偏差警报 “QH”的动作

上限偏差警报 “QH”为相对于基准值 “Y”的上方的偏差警报。“QH”值的变化如下所示。可通过滞后，防止偏差警报震荡。

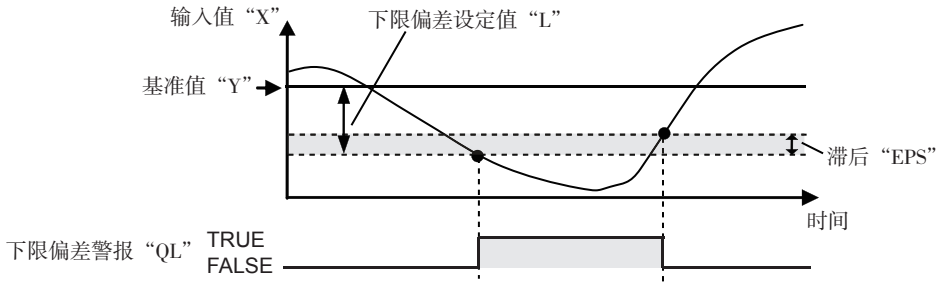
- 输入值 “X”–基准值 “Y” > 上限偏差设定值 “H”时，为TRUE。
- 输入值 “X”–基准值 “Y” < 上限偏差设定值 “H”–滞后 “EPS”时，为FALSE。



下限偏差警报 “QL”的动作

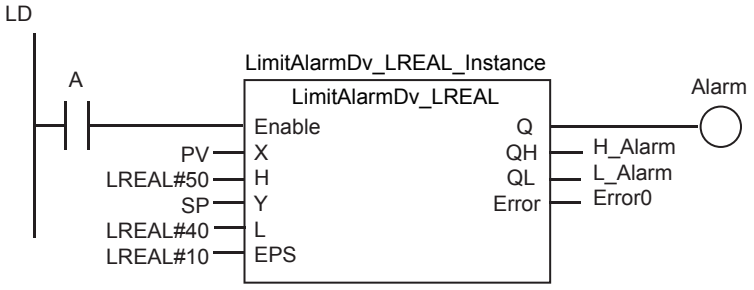
下限偏差警报 “QL”为相对于基准值 “Y”的下方的偏差警报。“QL”值的变化如下所示。可通过滞后，防止偏差警报震荡。

- –(输入值 “X”–基准值 “Y”)＞下限偏差设定值 “L”时，为TRUE。
- –(输入值 “X”–基准值 “Y”)＜下限偏差设定值 “L”–滞后 “EPS”时，为FALSE。



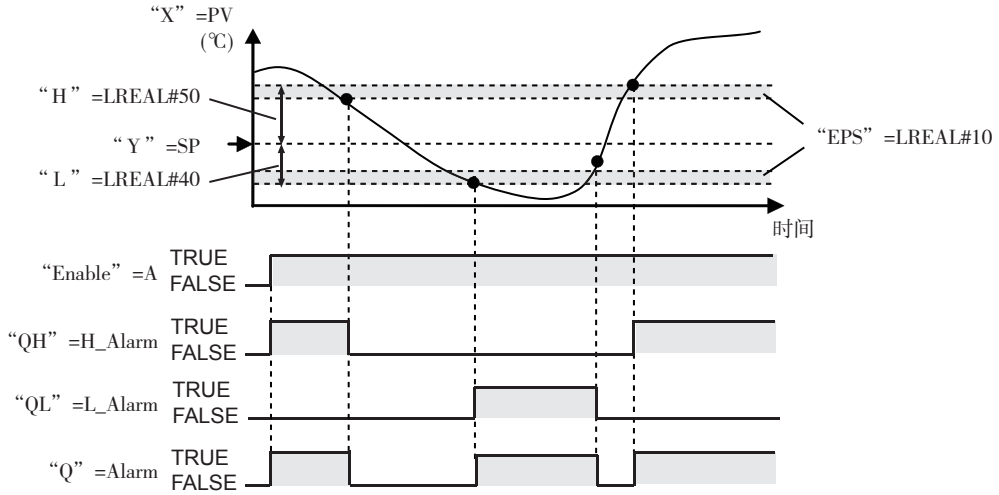
记述示例

上限偏差设定值 “H”为50℃，下限偏差设定值 “L”为40℃，滞后 “EPS”为10℃时的记述示例如下所示。



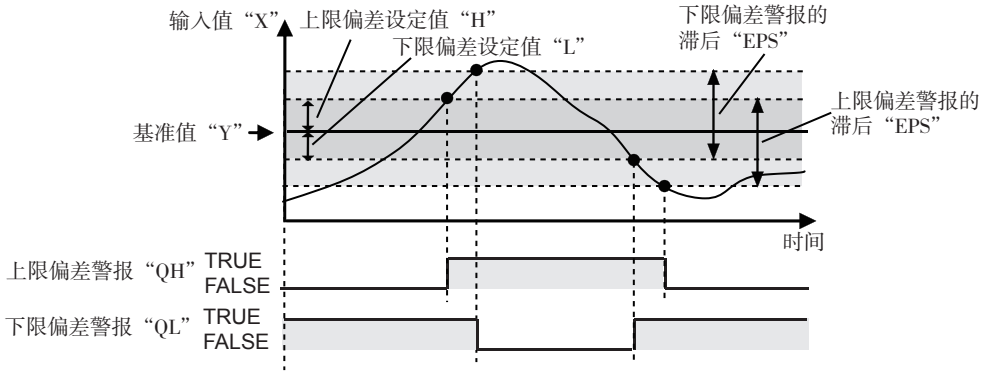
ST

```
LimitAlarmDv_LREAL_instance(A,PV,LREAL#50,SP,LREAL#40,LREAL#10,Alarm,H_Alarm,L_Alarm,Error0);
```

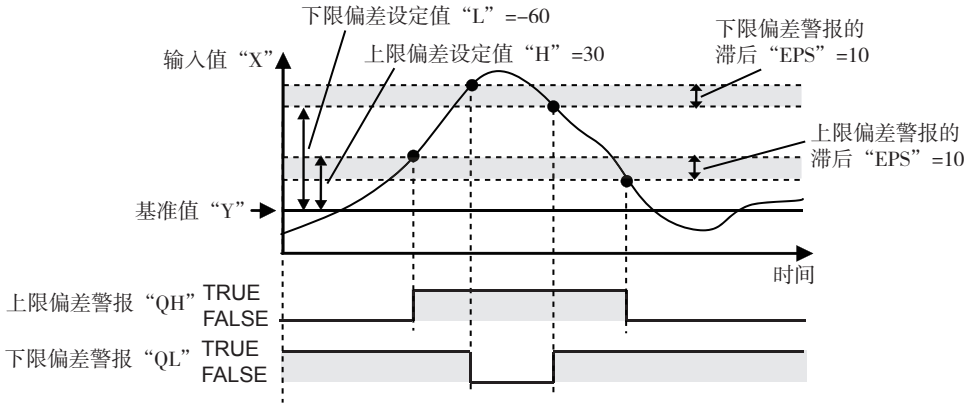


参考

- 如需缩短指令执行时间，则请使用LimitAlarmDv_REAL指令。
- 也可设定为 “H” + “L” < “EPS”。此时，“QH”和 “QL”可能均会变为TRUE。



- 也可设定为 “H” + “L” < 0。此时，“QH”和 “QL”中的任意一个始终为TRUE。例如，“L”的值为–60，“H”的值为30时，动作如下图所示。



使用注意事项

- “EPS”的值超过有效范围时，会发生异常。“Error”为TRUE，“Q”、“QH”、“QL”为FALSE。
- 可用于输出偏差警报时关闭温度控制的输出等对本指令采取安全措施的用途。此时，即使 “Q”、“QH”及 “QL”因异常变为FALSE，使用时仍请确保可采取安全措施。使用示例请参阅 “TimeProportionalOut指令(P.2–719)”的示例程序。

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

请参阅 “TimeProportionalOut指令(P.2–719)”的示例程序。

LimitAlarmDvStbySeq_**

带待机时序，输出上下限偏差警报。

指令	名称	FB/FUN	图形表现	ST表现
LimitAlarmDv StbySeq_**	带待机时序的上下限偏差警报组	FB	LimitAlarmDvStbySeq_**_Instance LimitAlarmDvStbySeq_** Enable, X, H, Y, L, EPS → Q, QH, QL, StbySeqFlag, Error **为REAL、LREAL中的任意一个	LimitAlarmDvStbySeq_**_instance(Enable, X, H, Y, L, EPS, Q, QH, QL, StbySeqFlag, Error); **为REAL、LREAL中的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	动作指令	输入	TRUE：执行 FALSE：警报复位	遵从数据类型	–	FALSE
X	输入值		偏差警报对象值			
H	上限偏差设定值		相对于基准值的上方的偏差警报设定值			0
Y	基准值		偏差的基准值			
L	下限偏差设定值		相对于基准值的下方的偏差警报设定值			
EPS	滞后		警报的滞后	遵从数据类型*1		
Q	偏差警报输出	输出	TRUE：上限偏差警报和下限偏差警报中的任意一个变为ON FALSE：上限偏差警报和下限偏差警报均变为OFF	遵从数据类型	–	–
QH	上限偏差警报		TRUE：上限偏差警报ON FALSE：上限偏差警报OFF			
QL	下限偏差警报		TRUE：下限偏差警报ON FALSE：下限偏差警报OFF			
StbySeqFlag	待机时序有效标志		TRUE：有效 FALSE：无效			

*1 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
X														○	○					

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
H	与 “X” 相同的数据类型																			
Y	与 “X” 相同的数据类型																			
L	与 “X” 相同的数据类型																			
EPS	与 “X” 相同的数据类型																			
Q	○																			
QH	○																			
QL	○																			
StbySeqFlag	○																			

功能

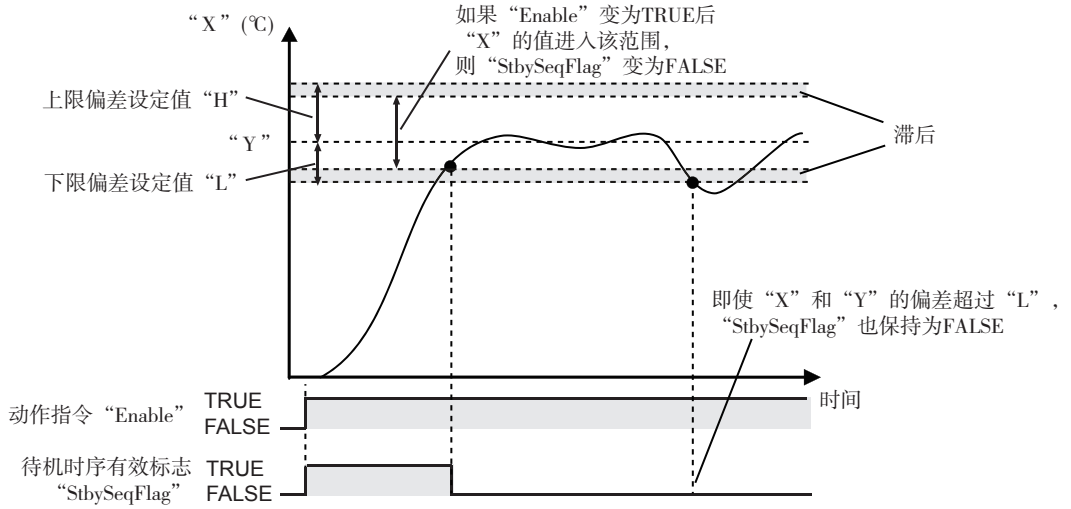
监控输入值与基准值的偏差是否超过下限偏差设定值或上限偏差设定值。偏差超过下限偏差设定值或上限偏差设定值时输出警报。不输出警报，直至偏差变为下限偏差设定值和上限偏差设定值以下的值。用于不输出偏差报警，直至通过温度控制稳定测量温度等用途。

动作指令 “Enable” 为 TRUE 期间，监控输入值 “X” 与基准值 “Y” 的偏差。待机时序有效标志 “StbySeqFlag” 为TRUE期间，不监控偏差。
“X” 上方与 “Y” 的偏差超过上限偏差设定值 “H” 的值时，上限偏差警报 “QH” 为TRUE。“X” 下方与 “Y” 的偏差超过下限偏差设定值 “L” 的值时，下限偏差警报 “QL” 为TRUE。“QH”、“QL” 中的任一值为TRUE时，警报输出 “Q” 变为TRUE。“Enable” 为TRUE期间，始终反映 “X”、“H”、“Y”、“L”、“EPS” 的值。

“Enable” 为FALSE时，将警报复位。通过复位，“Q”、“QH”、“QL”、“StbySeqFlag” 均为FALSE。

如果 “Enable”变为TRUE后满足以下所有条件，则 “StbySeqFlag”变为FALSE。如果 “StbySeqFlag”变为FALSE，则在 “Enable”由FALSE变为TRUE之前不会变为TRUE。

- 输入值 “X”–基准值 “Y” < 上限偏差设定值 “H”–滞后 “EPS”
- –(输入值 “X”–基准值 “Y”) < 下限偏差设定值 “L”–滞后 “EPS”

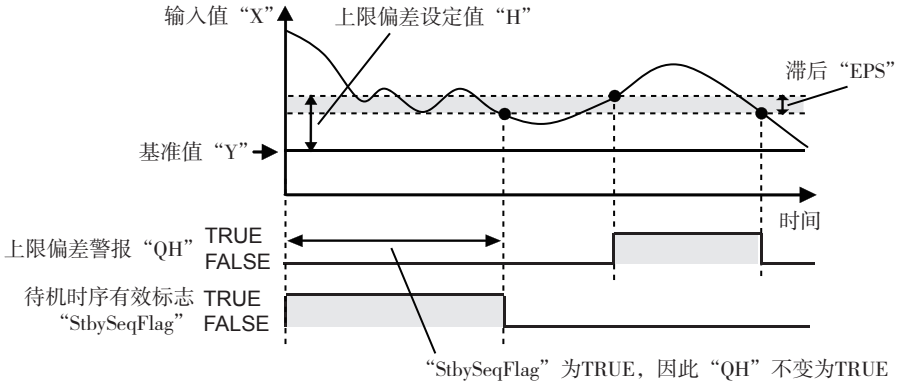


“X”、“H”、“Y”、“L”、“EPS”的数据类型为REAL、LREAL中的任意一个。指令名称因 “X”、“H”、“Y”、“L”、“EPS”的数据类型而异。例如，指令名称为LimitAlarmDvStbySeq_LREAL时，“X”、“H”、“Y”、“L”、“EPS”的数据类型均为LREAL。

上限偏差警报 “QH”的动作

上限偏差警报 “QH”为相对于基准值 “Y”的上方的偏差警报。“StbySeqFlag”为FALSE时，“QH”值的变化如下所示。可通过设定滞后，防止偏差警报震荡。

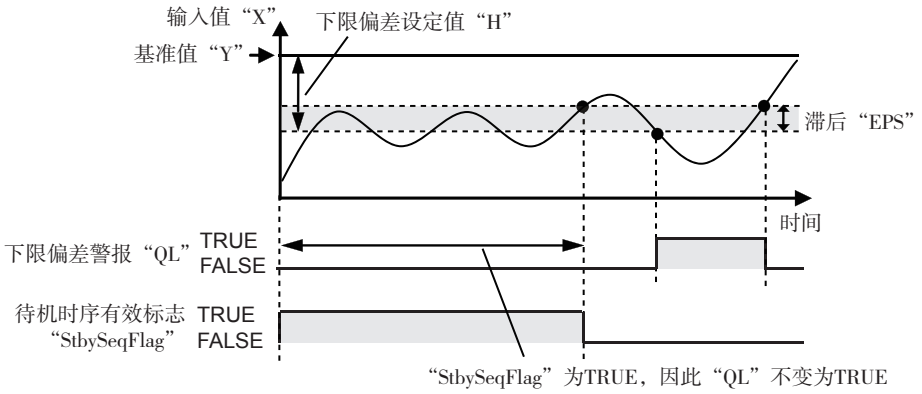
- 输入值 “X”–基准值 “Y” > 上限偏差设定值 “H”时，为TRUE。
- 输入值 “X”–基准值 “Y” < 上限偏差设定值 “H”–滞后 “EPS”时，为FALSE。



下限偏差警报 “QL”的动作

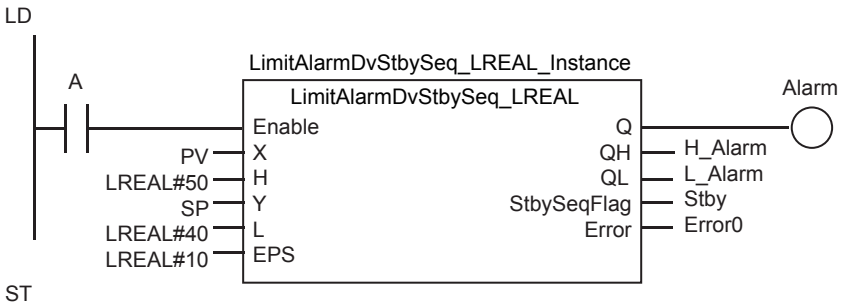
下限偏差警报 “QH”为相对于基准值 “Y”的下方的偏差警报。“StbySeqFlag”为FALSE时，“QL”值的变化如下所示。可通过设定滞后，防止偏差警报震荡。

- –(输入值 “X”–基准值 “Y”) > 下限偏差警报 “L”时，为TRUE。
- –(输入值 “X”–基准值 “Y”) < 下限偏差警报 “L”–滞后 “EPS”时，为FALSE。



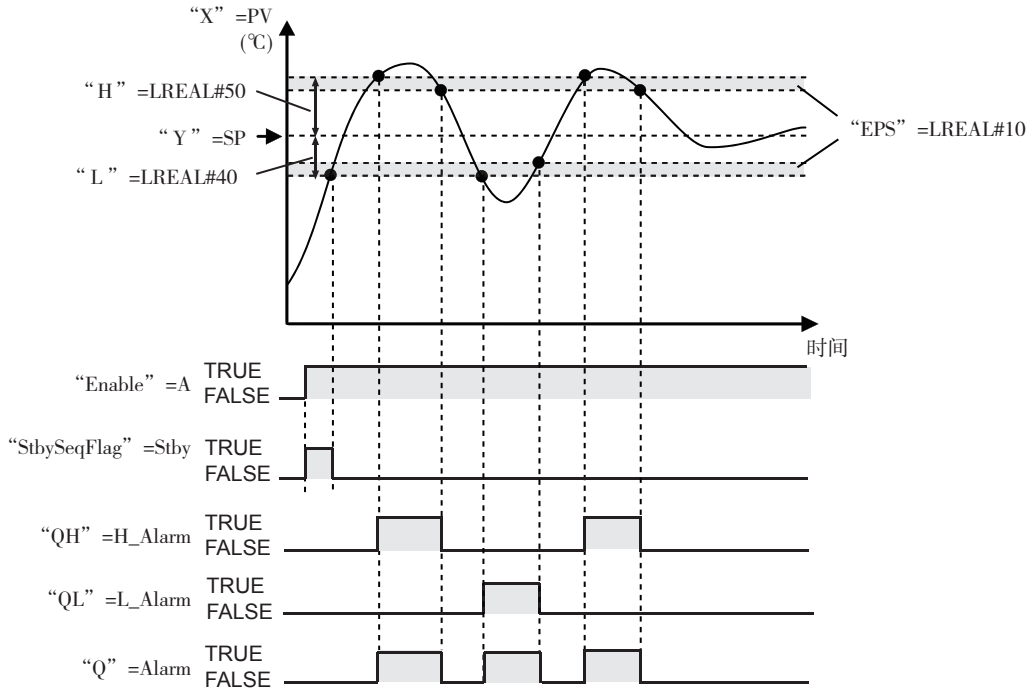
记述示例

上限偏差设定值“H”为50℃，下限偏差设定值“L”为40℃，滞后“EPS”为10℃时的记述示例如下所示。



ST

```
LimitAlarmDvStbySeq_LREAL_Instance(A,PV,LREAL#50,SP,LREAL#40,LREAL#10,Alarm,H_Alarm,L_Alarm,Stby,Error0);
```



参考

- 如需缩短指令执行时间，则请使用LimitAlarmDvStbySeq_REAL指令。
- 也可设定为 “H” + “L” < “EPS”。此时，“QH”和 “QL”可能均会变为TRUE。请参阅 “LimitAlarmDv_**指令(P.2-738)”。
- 也可设定为 “H” + “L” < 0。此时，“StbySeqFlag”为FALSE期间，“QH”和 “QL”的任意一个始终为TRUE。请参阅 “LimitAlarmDv_**指令(P.2-738)”。

使用注意事项

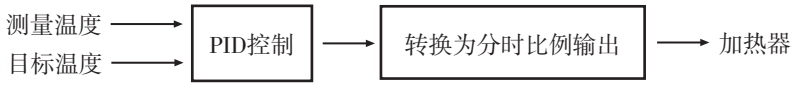
- “EPS”的值超过有效范围时，会发生异常。“Error”为TRUE，“Q”、“QH”、“QL”为FALSE。
- 可用于输出偏差警报时关闭温度控制的输出等对本指令采取安全措施的用途。此时，即使 “Q”、“QH”及 “QL”因异常变为FALSE，使用时仍请确保可采取安全措施。使用示例请参阅示例程序。

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

示例程序

进行上下限警报、带待机时序的上下限偏差警报的4点温度控制。使用PID控制方式，将PID控制的操作量转换为分时比例输出，输出至加热器。



规格

通过下表所示规格，进行温度控制。

项目	规格
输入单元	绝缘型高分辨率多功能输入单元 CJ1W-PH41U
输入类型	均为K型热电偶
输出单元	晶体管输出单元 CJ1W-OD212
目标温度	100℃
上限温度	200℃
下限温度	0℃
上下限警报的滞后	5℃
上限偏差温度	50℃
下限偏差温度	50℃
上下限偏差警报的滞后	3℃
PID运算的采样周期	100ms
输出控制周期	1s

构成・设定

输入单元CJ1W-PH41U的设定如下所示。

项目名称	设定值
Input1:Input signal type	K(1)
Input2:Input signal type	K(1)
Input3:Input signal type	K(1)
Input4:Input signal type	K(1)

I/O映射设定如下所示。

单元	端口	说明	变量
CJ1W-PH41U	Ch1_AIInPV	输入No.1测量值(INT型)	AI1
	Ch2_AIInPV	输入No.2测量值(INT型)	AI2
	Ch3_AIInPV	输入No.3测量值(INT型)	AI3
	Ch4_AIInPV	输入No.4测量值(INT型)	AI4
CJ1W-OD212	Ch1_Out00	输出CH1触点00	DO1
	Ch1_Out01	输出CH1触点01	DO2
	Ch1_Out02	输出CH1触点02	DO3
	Ch1_Out03	输出CH1触点03	DO4

4点温度控制的输入与输出的对应关系如下所示。

输入	输出
AI1	DO1
AI2	DO2
AI3	DO3
AI4	DO4

将待分配程序的任务周期设定为1ms。

● 结构图

请参阅 🕮 “TimeProportionalOut指令(P.2-719)”的示例程序。

处理内容

进行4点以下处理。

1 获取测量温度。

2 使用LimitAlarm_REAL指令，针对测量温度输出上下限警报。

3 LimitAlarm_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出。

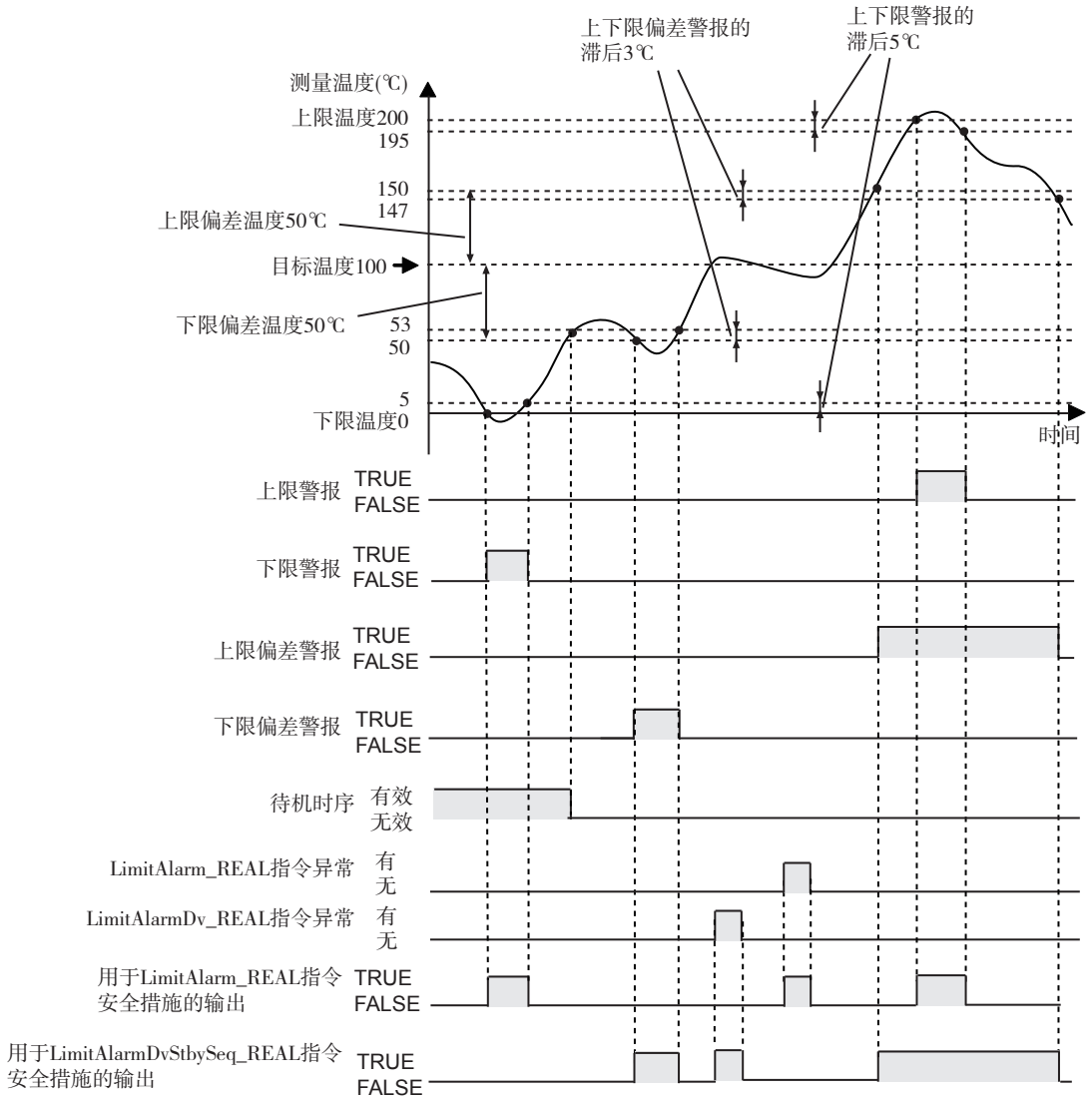
4 使用LimitAlarmDvStbySeq_REAL指令，相对测量温度与目标温度的偏差，输出带待机时序的上下限偏差警报。

5 LimitAlarmDvStbySeq_REAL 指令下发生异常或输出上下限偏差警报时，进行用于安全措施的输出。

6 使用PIDAT指令进行温度控制。

7 使用TimeProportionalOut指令，通过分时比例将操作量输出至加热器。

● 上下限警报、带待机时序的上下限偏差警报的动作



应用程序

全局变量的定义

全局变量

名称	数据类型	分配对象	注释
AI1	INT	IOBus://rack#0/slot#0/Ch1_AIInPV	输入No.1测量值(INT型)
AI2	INT	IOBus://rack#0/slot#0/Ch2_AIInPV	输入No.2测量值(INT型)
AI3	INT	IOBus://rack#0/slot#0/Ch3_AIInPV	输入No.3测量值(INT型)
AI4	INT	IOBus://rack#0/slot#0/Ch4_AIInPV	输入No.4测量值(INT型)
DO1	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out00	输出CH1触点00
DO2	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out01	输出CH1触点01
DO3	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out02	输出CH1触点02
DO4	BOOL	IOBus://rack#0/slot#1/Ch1_Out/Ch1_Out03	输出CH1触点03

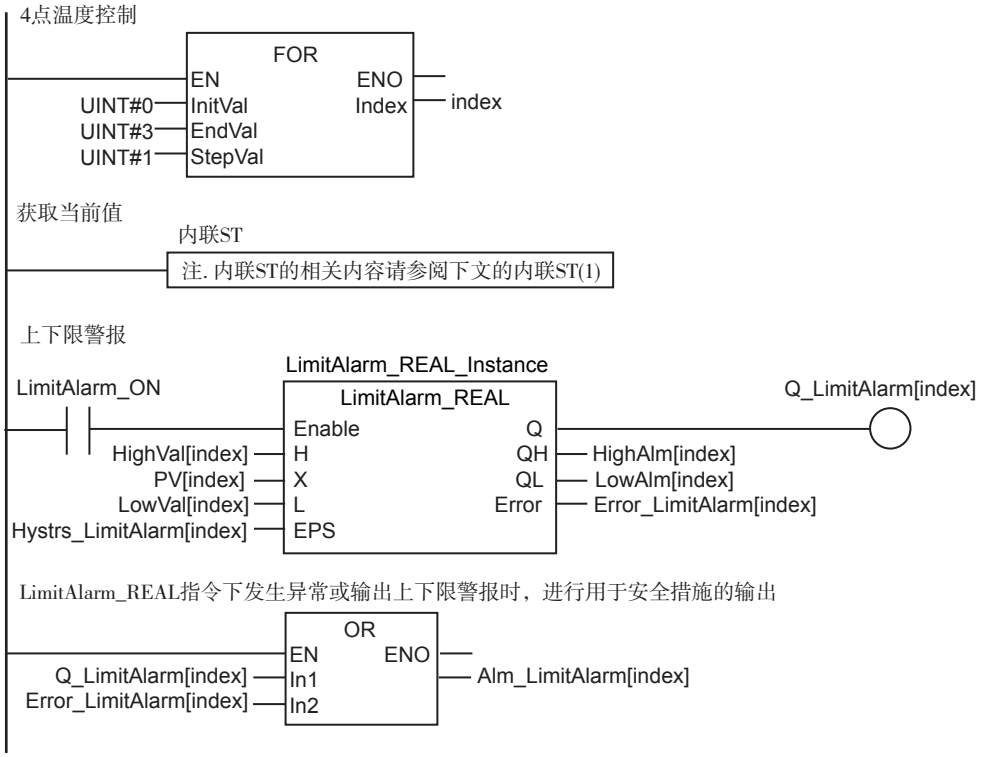
*1 将CJ1W-PH41U安装至机架编号0的插槽编号0，将CJ1W-OD212安装至机架编号0的插槽编号1时的分配对象。

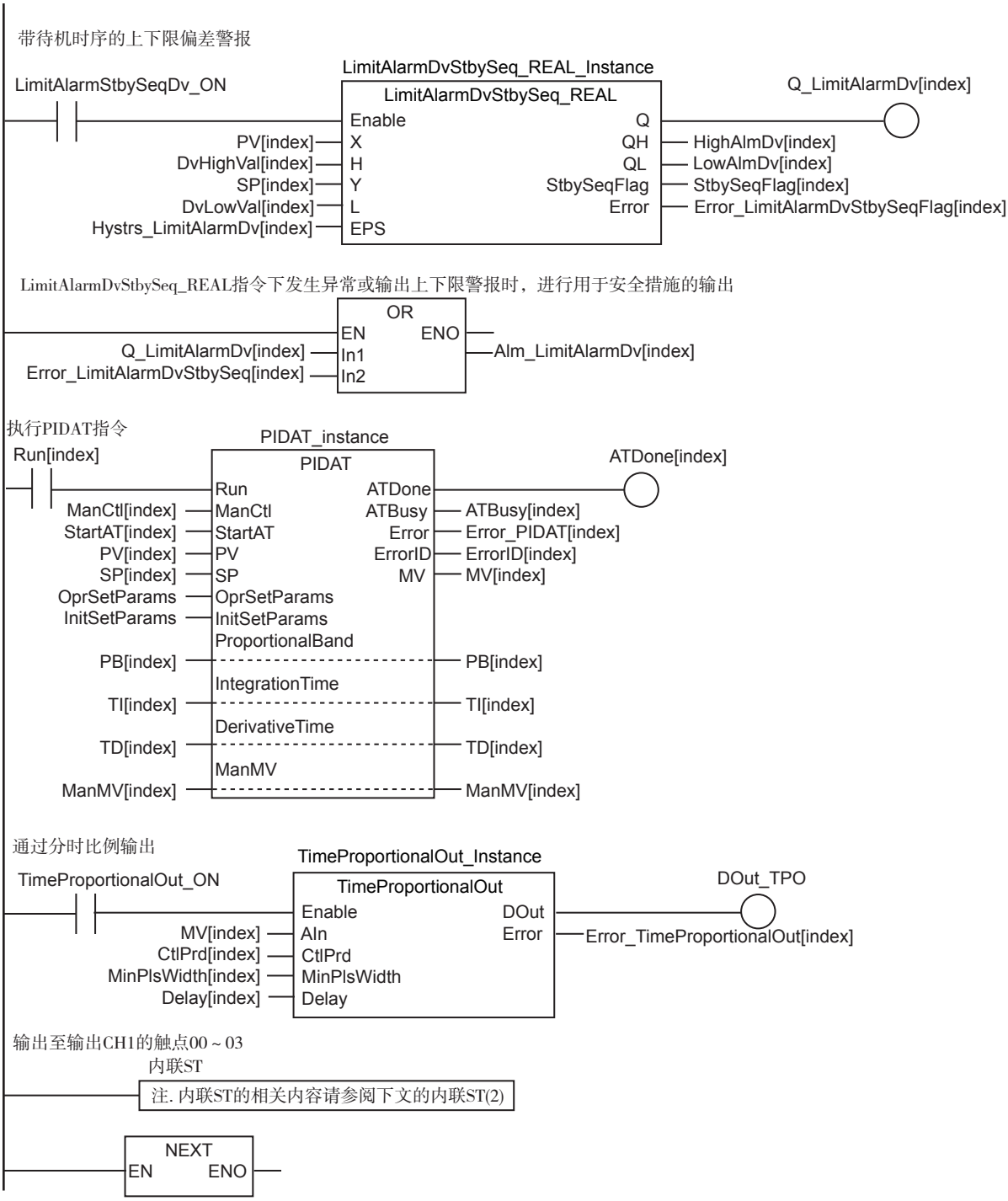
(注) 通过设定I/O映射，自动生成分配至各单元端口的全局变量。

LD

名称	数据类型	初始值	保持	注释
index	UINT	0	□	循环索引
LimitAlarm_ON	BOOL	True	□	执行上下限警报
LimitAlarmDvStbySeq_ON	BOOL	True	□	执行带待机时序的上下限偏差警报
TimeProportionalOut_ON	BOOL	True	□	执行分时比例输出
AI	INT	0	□	当前值
PV	ARRAY[0..3] OF REAL	[4(0.0)]	□	测量值
SP	ARRAY[0..3] OF REAL	[4(100)]	□	目标温度
DOut_TPO	BOOL	False	□	分时比例输出
HighVal	ARRAY[0..3] OF REAL	[4(200)]	□	上下限警报的上限设定值
LowVal	ARRAY[0..3] OF REAL	[4(0.0)]	□	上下限警报的下限设定值
Hystrs_LimitAlarm	ARRAY[0..3] OF REAL	[4(5)]	□	上下限警报的滞后
Q_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	上下限警报输出
HighAlm	ARRAY[0..3] OF BOOL	[4(False)]	□	上限警报
LowAlm	ARRAY[0..3] OF BOOL	[4(False)]	□	下限警报
Error_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	LimitAlarm_REAL指令异常
Alm_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	用于上下限警报安全措施的输出
DvHighVal	ARRAY[0..3] OF REAL	[4(50)]	□	上下限偏差警报的上限偏差设定值
DvLowVal	ARRAY[0..3] OF REAL	[4(50)]	□	上下限偏差警报的下限偏差设定值
Q_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	输出上下限偏差警报
HighAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	上限偏差警报
LowAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	下限偏差警报
StbySeqFlag	ARRAY[0..3] OF BOOL	[4(False)]	□	待机时序有效标志
Error_LimitAlarmDvStbySeq	ARRAY[0..3] OF BOOL	[4(False)]	□	LimitAlarmDvStbySeq_REAL指令异常
Hystrs_LimitAlarmDv	ARRAY[0..3] OF REAL	[4(3)]	□	上下限偏差警报的滞后
Alm_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	用于上下限偏差警报安全措施的输出
Run	ARRAY[0..3] OF BOOL	[4(False)]	□	执行条件
ManCtl	ARRAY[0..3] OF BOOL	[4(False)]	□	手动/自动切换
StartAT	ARRAY[0..3] OF BOOL	[4(False)]	□	自动调谐执行条件
OprSetParams	_sOPR_SET_PARAMS	(MVLowLmt:=0.0，MVUpLmt:=100.0，ManResetVal:=0.0，MVTrackSw:=False，MVTrackVal:=0.0，StopMV:=0.0，ErrorMV:=0.0，Alpha:=0.65，ATCalcGain:=1.0，ATHystrs:=0.2)	□	运行中设定参数
InitSetParams	_sINIT_SET_PARAMS	(SampTime:=T#100ms，RngLowLmt:=-10.0，RngUpLmt:=1000.0，DirOpr:=False)	□	初始设定参数
PB	ARRAY[0..3] OF REAL	[4(10)]	☑	比例带
TI	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	积分时间
TD	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	微分时间
ManMV	ARRAY[0..3] OF REAL	[4(0.0)]	□	手动操作量
ATDone	ARRAY[0..3] OF BOOL	[4(False)]	□	自动调谐正常结束

名称	数据类型	初始值	保持	注释
ATBusy	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行中
Error_PIDAT	ARRAY[0..3] OF BOOL	[4(False)]	☐	PIDAT指令异常
ErrorID	ARRAY[0..3] OF WORD	[4(16#0)]	☐	PIDAT指令的异常ID
MV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	操作量
CtlPrd	ARRAY[0..3] OF TIME	[4(T#1s)]	☐	控制周期
MinPlsWidth	ARRAY[0..3] OF REAL	[4(0.0)]	☐	最小脉冲宽度
Delay	ARRAY[0..3] OF REAL	[4(0.0)]	☐	ON延迟时间
Error_Time ProportionalOut	ARRAY[0..3] OF BOOL	[4(False)]	☐	TimeProportionalOut指令异常
LimitAlarm_REAL _instance	LimitAlarm_REAL		☐	
LimitAlarmDvStby Seq_REAL_ instance	LimitAlarmDvStbySeq _REAL		☐	
PIDAT_instance	PIDAT		☐	
TimeProportional Out_instance	TimeProportionalOut		☐	





● 内联ST (1)的内容

```
//获取输入1 ~ 4的值
CASE index OF
    INT#0:
        AI:=AI1;
    INT#1:
        AI:=AI2;
    INT#2:
        AI:=AI3;
    ELSE
        AI:=AI4;
END_CASE;

//将当前值AI转换为实数
```

```
PV[index]:=INT_TO_REAL(AI)/REAL#10.0; // CJ1W-PH41U的输出为测量温度的10倍，因此除以10.0
```

- 内联ST (2)的内容

```
//输出至输出CH1的触点00 ~ 03
CASE index OF
    INT#0:
        DO1:=DOut_TPO;
    INT#1:
        DO2:=DOut_TPO;
    INT#2:
        DO3:=DOut_TPO;
    ELSE
        DO4:=DOut_TPO;
END_CASE;
```

ST

名称	数据类型	初始值	保持	注释
index	UINT	0	□	循环索引
LimitAlarm_ON	BOOL	True	□	执行上下限警报
LimitAlarmDvStbySeq_ON	BOOL	True	□	执行带待机时序的上下限偏差警报
TimeProportionalOut_ON	BOOL	True	□	执行分时比例输出
AI	INT	0	□	当前值
PV	ARRAY[0..3] OF REAL	0.0	□	测量值
SP	ARRAY[0..3] OF REAL	[4(100)]	□	目标温度
DOut_TPO	BOOL	False	□	分时比例输出
HighVal	ARRAY[0..3] OF REAL	[4(200)]	□	上下限警报的上限设定值
LowVal	ARRAY[0..3] OF REAL	[4(0.0)]	□	上下限警报的下限设定值
Hystrs_LimitAlarm	ARRAY[0..3] OF REAL	[4(5)]	□	上下限警报的滞后
Q_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	上下限警报输出
HighAlm	ARRAY[0..3] OF BOOL	[4(False)]	□	上限警报
LowAlm	ARRAY[0..3] OF BOOL	[4(False)]	□	下限警报
Error_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	LimitAlarm_REAL指令异常
Alm_LimitAlarm	ARRAY[0..3] OF BOOL	[4(False)]	□	用于上下限警报安全措施的输出
DvHighVal	ARRAY[0..3] OF REAL	[4(50)]	□	上下限偏差警报的上限偏差设定值
DvLowVal	ARRAY[0..3] OF REAL	[4(50)]	□	上下限偏差警报的下限偏差设定值
Q_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	输出上下限偏差警报
HighAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	上限偏差警报
LowAlmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	下限偏差警报
StbySeqFlag	ARRAY[0..3] OF BOOL	[4(False)]	□	待机时序有效标志
Error_LimitAlarmDvStbySeq	ARRAY[0..3] OF BOOL	[4(False)]	□	LimitAlarmDvStbySeq_REAL指令异常
Hystrs_LimitAlarmDv	ARRAY[0..3] OF REAL	[4(3)]	□	上下限偏差警报的滞后
Alm_LimitAlarmDv	ARRAY[0..3] OF BOOL	[4(False)]	□	用于上下限偏差警报安全措施的输出
Run	ARRAY[0..3] OF BOOL	[4(False)]	□	执行条件
ManCtl	ARRAY[0..3] OF BOOL	[4(False)]	□	手动/自动切换

名称	数据类型	初始值	保持	注释
StartAT	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行条件
OprSetParams	_sOPR_SET_PARAMS	(MVLowLmt:=0.0，MVUpLmt:=100.0，ManResetVal:=0.0，MVTrackSw:=False，MVTrackVal:=0.0，StopMV:=0.0，ErrorMV:=0.0，Alpha:=0.65，ATCalcGain:=1.0，ATHystrs:=0.2)	☐	运行中设定参数
InitSetParams	_sINIT_SET_PARAMS	(SampTime:=T#100ms，RngLowLmt:= −10.0，RngUpLmt:=1000.0，DirOpr:=False)	☐	初始设定参数
PB	ARRAY[0..3] OF REAL	[4(10)]	☑	比例带
TI	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	积分时间
TD	ARRAY[0..3] OF TIME	[4(T#0S)]	☑	微分时间
ManMV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	手动操作量
ATDone	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐正常结束
ATBusy	ARRAY[0..3] OF BOOL	[4(False)]	☐	自动调谐执行中
Error_PIDAT	ARRAY[0..3] OF BOOL	[4(False)]	☐	PIDAT指令异常
ErrorID	ARRAY[0..3] OF WORD	[4(16#0)]	☐	PIDAT指令的异常ID
MV	ARRAY[0..3] OF REAL	[4(0.0)]	☐	操作量
CtlPrd	ARRAY[0..3] OF TIME	[4(T#1s)]	☐	控制周期
MinPlsWidth	ARRAY[0..3] OF REAL	[4(0.0)]	☐	最小脉冲宽度
Delay	ARRAY[0..3] OF REAL	[4(0.0)]	☐	ON延迟时间
Error_TimeProportionalOut	ARRAY[0..3] OF BOOL	[4(False)]	☐	TimeProportionalOut指令异常
LimitAlarm_REAL_instance	LimitAlarm_REAL		☐	
LimitAlarmDvStbySeq_REAL_instance	LimitAlarmDvStbySeq_REAL		☐	
PIDAT_instance	PIDAT		☐	
TimeProportionalOut_instance	TimeProportionalOut		☐	

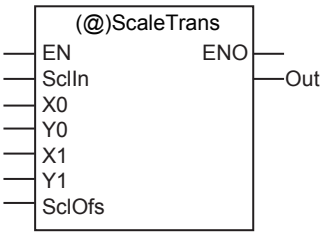
```
//4点温度控制
FOR index:=UINT#0 TO UINT#3 BY UINT#1 DO

    //获取输入1 ~ 4的值
    CASE index OF
        INT#0:
            AI:=AI1;
        INT#1:
            AI:=AI2;
        INT#2:
            AI:=AI3;
        ELSE
            AI:=AI4;
    END_CASE;

  //将当前值AI转换为实数
  PV[index]:=INT_TO_REAL(AI)/REAL#10.0;  // CJ1W-PH41U的输出为测量温度的10倍，因此除以10.0
```

```
//上下限警报
LimitAlarm_REAL_instance(
  Enable :=LimitAlarm_ON,
  H      :=HighVal[index],
  X      :=PV[index],
  L      :=LowVal[index],
  EPS    :=Hystrs_LimitAlarm[index],
  Q      =>Q_LimitAlarm[index],
  QH     =>HighAlm[index],
  QL     =>LowAlm[index],
  Error  =>Error_LimitAlarm[index]);

//LimitAlarm_REAL指令下发生异常或输出上下限警报时，进行用于安全措施的输出
Alm_LimitAlarm[index]:=Q_LimitAlarm[index] OR Error_LimitAlarm[index];

//带待机时序的上下限偏差警报
LimitAlarmDvStbySeq_REAL_instance(
  Enable       :=LimitAlarmDvStbySeq_ON,
  X            :=PV[index],
  H            :=DvHighVal[index],
  Y            :=SP[index],
  L            :=DvLowVal[index],
  EPS          :=Hystrs_LimitAlarmDv[index],
  Q            =>Q_LimitAlarmDv[index],
  QH           =>HighAlmDv[index],
  QL           =>LowAlmDv[index],
  StbySeqFlag  =>StbySeqFlag[index],
  Error        =>Error_LimitAlarmDvStbySeq[index]);

//LimitAlarmDvStbySeq_REAL指令下发生异常
//或输出上下限警报时，进行用于安全措施的输出
Alm_LimitAlarmDv[index]:=Q_LimitAlarmDv[index] OR Error_LimitAlarmDvStbySeq[index];

// 执行PIDAT指令
PIDAT_instance(
  Run              :=Run[index],
  ManCtl           :=ManCtl[index],
  StartAT          :=StartAT[index],
  PV               :=PV[index],
  SP               :=SP[index],
  OprSetParams     :=OprSetParams,
  InitSetParams    :=InitSetParams,
  ProportionalBand :=PB[index],
  IntegrationTime  :=TI[index],
  DerivativeTime   :=TD[index],
  ManMV            :=ManMV[index],
  ATDone           =>ATDone[index],
  ATBusy           =>ATBusy[index],
  Error            =>Error_PIDAT[index],
  ErrorID          =>ErrorID[index],
  MV               =>MV[index]);

// 通过分时比例输出
TimeProportionalOut_instance(
  Enable       :=TimeProportionalOut_ON,
  AIn          :=MV[index],
  CtlPrd       :=CtlPrd[index],
  MinPlsWidth  :=MinPlsWidth[index],
  Delay        :=Delay[index],
```

```
        DOut            =>DOut_TPO,
        Error           =>Error_TimeProportionalOut[index]);

    //输出至输出Ch1的触点00 ~ 03
    CASE index OF
        INT#0:
            DO1:=DOut_TPO;
        INT#1:
            DO2:=DOut_TPO;
        INT#2:
            DO3:=DOut_TPO;
        ELSE
            DO4:=DOut_TPO;
    END_CASE;

END_FOR;
```

ScaleTrans

将输入值从输入范围转换为输出范围。

指令	名称	FB/FUN	图形表现	ST表现
ScaleTrans	比例转换	FUN		Out :=ScaleTrans(SclIn,X0,Y0,X1, Y1,SclOfs);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
SclIn	输入值	输入	转换对象值	遵从数据类型	–	(*)
X0	输入范围下限值		输入范围的下限值			0
Y0	输出范围下限值		输出范围的下限值			
X1	输入范围上限值		输入范围的上限值			
Y1	输出范围上限值		输出范围的上限值			
SclOfs	偏置值		输出值叠加的偏置值			
Out	输出值	输出	比例转换后的值		–	–

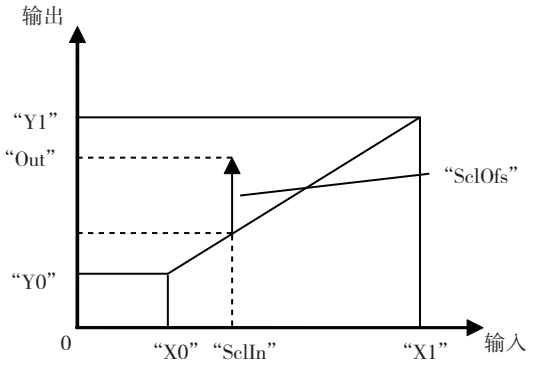
* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
SclIn														○	○					
X0														○	○					
X1														○	○					
Y0														○	○					
Y1														○	○					
SclOfs														○	○					
Out														○	○					

功能

将输入值 “SclIn”的值从输入范围转换为输出范围。
输入范围由输入范围下限值 “X0”和输入范围上限值 “X1”指定。输出范围由输出范围下限值 “Y0”和输出范围上限值 “Y1”指定。
将转换为输出范围的值加上偏置值 “SclOfs”得出的值，作为输出值 “Out”输出。“SclOfs”用于温度控制误差的调整等。
转换公式如下所示。

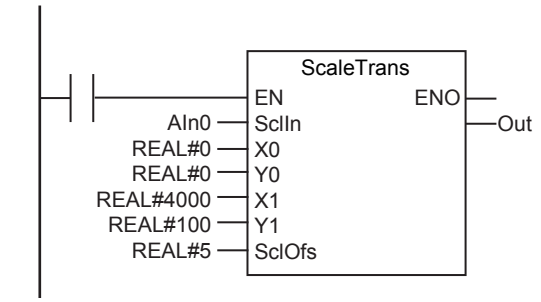
$$\text{“Out”} = \frac{\text{“Y1”} - \text{“Y0”}}{\text{“X1”} - \text{“X0”}} (\text{“SclIn”} - \text{“X0”}) + \text{“Y0”} + \text{“SclOfs”}$$



记述示例

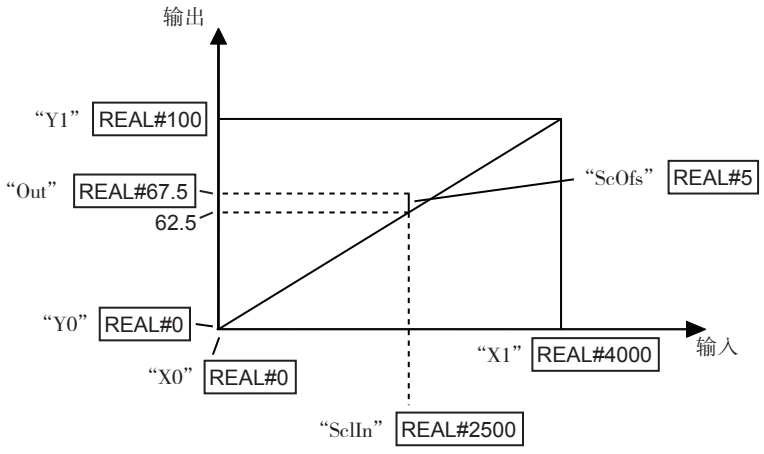
输入范围为0～4000、输出范围为0～100%，对输入值2500进行转换时的示例如下所示。将输出叠加的偏置值设为5%。
设 “SclIn”=REAL#2500、“X0”=REAL#0、“X1” = REAL#4000、“Y0”=REAL#0、“Y1”=REAL#100、“SclOfs”=REAL#5。
“Out”的值为REAL#67.5。

LD



ST

```
Out :=ScaleTrans(AIn0,REAL#0,REAL#0,REAL#4000,
                 REAL#100,REAL#5);
```



输入范围为0～4000、输出范围为0～100，对输入值2500进行转换则为62.5。
再加上偏置值5，则 “Out”=REAL#67.5。

参考

- 将“SclIn”比例转换到PIDAT指令的“PV”和“SP”的值范围时，请对“Y0”、“Y1”传输下列参数。

变量	参数
“Y0”	PIDAT指令的输入范围下限 “InitSetParams.RngLowLmt”
“Y1”	PIDAT指令的输入范围上限 “InitSetParams.RngUpLmt”

- 也可设定为 “X1” < “X0”、“Y1” < “Y0”。

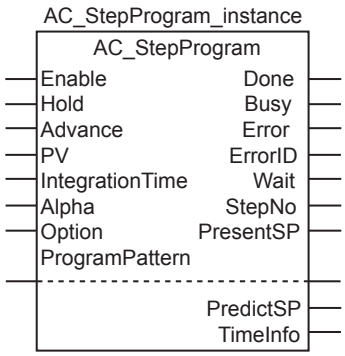
使用注意事项

版本相关信息

本指令可用于CPU单元Ver.1.05以上且Sysmac Studio Ver.1.06以上。

AC_StepProgram

根据指定程序模式，按任务周期计算当前目标值和预测目标值。

指令	名称	FB/FUN	图形表现	ST表现
AC_StepProgram	步程序	FB		AC_StepProgram_instance(Enable, Hold, Advance, PV, IntegrationTime, Alpha, Option, ProgramPattern, Done, Busy, Error, ErrorID, Wait, StepNo, PresentSP, PredictSP, TimeInfo);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	动作指令	输入	TRUE：执行 FALSE：非执行	遵从数据类型	–	FALSE
Hold	保持		TRUE：执行保持功能 FALSE：不执行保持功能			
Advance	前进		从FALSE上升到TRUE时，对要执行的步号进行增量			
PV	测量值 (当前值)		测量值(当前值) 与PIDAT指令的 “PV”相同。			0
Integration Time	积分时间		积分时间 与PIDAT指令的 “IntegrationTime”相同。	T#0.0000s ~ T#10000.0000s*1	s	T#0s
Alpha	2自由度PID参数α		2自由度PID参数α 与PIDAT指令的 “OprSetParams.Alpha”相同。	0.00 ~ 1.00	–	0.65
Option	选项		选项 详情请参阅功能说明	–		–
Program Pattern[]数组	程序模式	输入输出	程序模式	–	–	–

	名称	输入/输出	内容	有效范围	单位	初始值
Wait	等待中	输出	TRUE：等待中 FALSE：非等待中	遵从数据类型	–	–
StepNo	当前步号		当前执行中的步号	0 ~ 99		
PresentSP	当前目标值		求得的当前目标值	遵从数据类型		
PredictSP	预测目标值		求得的预测目标值			
TimeInfo	时间信息		旨在监视本指令进展状况的时间信息	–		

*1 小于0.0001s时舍去。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
Hold	○																			
Advance	○																			
PV														○						
Integration Time																○				
Alpha														○						
Option	结构体_sAC_STEP_OPTION 详情参阅功能说明																			
Program Pattern[] 数组*1*2*3	结构体_sAC_STEP_DATA 详情参阅功能说明 指定整个数组																			
Wait	○																			
StepNo						○														
PresentSP														○						
PredictSP														○						
TimeInfo	结构体_sAC_STEP_TIME 详情参阅功能说明																			

*1 数组的最大元素数为100。
*2 1维数组。指定2维以上数组的情况下，编连时会发生异常。
*3 开头的元素编号为0。指定开头元素编号非0的数组的情况下，编连时会发生异常。

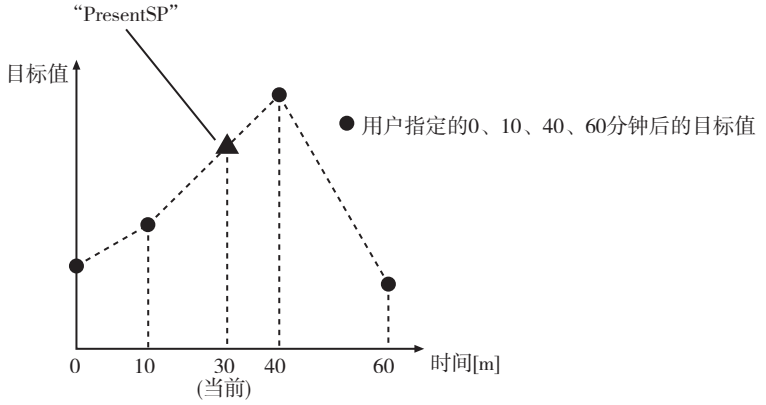
功能

与PIDAT指令联动，按周期计算进行温控器等的操作量控制时的当前目标值“PresentSP”和预测目标值“PredictSP”。

当前目标值是指当前任务周期的目标值。预测目标值是指，对当前目标值进行2自由度PID控制的延迟补偿后的目标值。将预测目标值“PredictSP”传输至PIDAT指令的目标值“SP”，可改善基于PIDAT指令的程序控制的随动性。

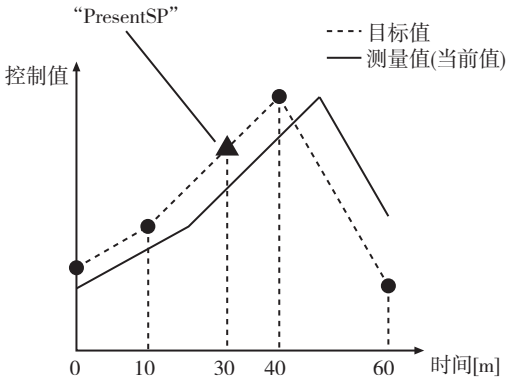
当前目标值 “PresentSP”

当前目标值 “PresentSP”是指当前任务周期的目标值。例如，用户按下图设定控制开始0、10、40、60分钟后的目标值。将当前时刻设为控制开始的30分钟后。利用本指令，对控制开始10分钟后和40分钟后的目标值进行线性插补，计算 “PresentSP”。

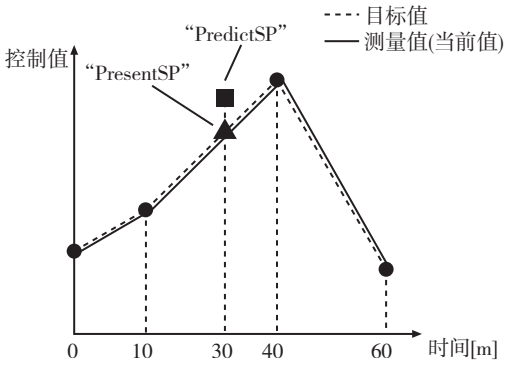


预测目标值 “PredictSP”

预测目标值 “PredictSP”是指，对 “PresentSP”进行2自由度PID控制的延迟补偿后的目标值。无补偿地将“PresentSP”传输至PIDAT指令的“SP”，则PIDAT指令的“PV”与目标值不一致。其情况如下图所示。



补偿该延迟的是 “PredictSP”。利用本命令，根据积分时间 “IntegrationTime”和 2 自由度 PID 参数 α “Alpha”的值，计算 “PredictSP”。将 “PredictSP”传输至PIDAT指令的 “SP”，可改善PIDAT指令的程序控制的随动性。



结构体的规格

选项 “Option”的数据类型为结构体_sAC_STEP_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	选项	选项	_sAC_STEP_OPTION	–	–	–
StartAtPV	当前值开始	TRUE：执行当前值开始功能 FALSE：不执行当前值开始功能	BOOL	遵从数据类型	–	FALSE
StartStepNo	开始步号	开始处理的步号	USINT	0 ~ 99	–	0
EndStepNo	结束步号	结束处理的步号*1	USINT			
Reserved	备用	备用	ARRAY[0..31] OF BYTE	遵从数据类型	–	32个元素全为0

*1 设定0时，将ProgramPattern[]的最大元素编号视为结束步号。

程序模式ProgramPattern[]的元素的数据类型为结构体_sAC_STEP_DATA。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Program Pattern	程序模式	程序模式	_sAC_STEP_DATA	–	–	–
ReachSP	到达目标值	该步的到达目标值	REAL	遵从数据类型		0
TimeWidth	时间宽度	该步的时间宽度*1	TIME		s	T#0s
WaitWidth	等待判定宽度	该步的等待判定宽度*2	REAL		–	0
WaitTime Limit	等待时间上限	该步的等待时间上限*1*3	TIME		s	T#0s

*1 分辨率为1个任务周期。
*2 设定0以下的值时，视为0。
*3 设定0以下的值时，视为T#0s。

时间信息 “TimeInfo”的数据类型为结构体_sAC_STEP_TIME。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
TimeInfo	时间信息	时间信息	_sAC_STEP_TIME	–	–	–
ProgramTime	程序时间	步0至 “EndStepNo”的“TimeWidth”的合计	TIME	0或正值	s	T#0s
ElapseTime	经过时间	开始执行本指令至当前的时间*1	TIME			
Progress Time	进度时间	开始执行本指令至当前的时间*2	TIME			
LeftTime	剩余时间	当前至全部处理结束的时间*2	TIME			
StepProgress Time	步内进度时间	执行中的步的开始至当前的时间*2	TIME			
StepLeftTime	步内剩余时间	当前至执行中的步的结束时间*2	TIME			

*1 含等待时间。不含保持时间。
*2 不含等待时间和保持时间。

各变量的含义

以下对本指令各变量的含义作详细说明。

● “Enable”(动作指令)

本指令的执行条件。
“Enable”的值从FALSE变为TRUE时，执行本指令。将 “Enable”的值设为FALSE时，停止执行本指令。

● “Hold”(保持)

用于执行保持功能的标志。
将 “Hold”的值设为TRUE时，执行保持功能。
保持功能的详情将于下文阐述。

● “Advance”(前进)

在执行本指令的过程中数值从FALSE变为TRUE时，处理将转至下一个步。
前进功能的详情将于下文阐述。

● “PV”(测量值、当前值)

控制对象的测量值。与PIDAT指令的 “PV”含义相同。

● “IntegrationTime”(积分时间)

与PIDAT指令的 “IntegrationTime”含义相同。

● “Alpha”(2自由度PID参数α)

与PIDAT指令的 “OprSetParams.Alpha”含义相同。

● “StartAtPV”(当前值开始)

当前值开始功能执行标志。
将 “StartAtPV”的值设为TRUE时，执行当前值开始功能。
当前值开始功能的详情将于下文阐述。

- “StartStepNo”(开始步号)、“EndStepNo”(结束步号)

  程序模式的步中，开始处理的步号和结束处理的步号。
  “EndStepNo”=0时，将ProgramPattern[]的最大元素编号视为结束步号。
  程序模式、步的详情将于下文阐述。

- “ReachSP”(到达目标值)

  程序模式内各步结束时应达到的目标值。
  程序模式、步的详情将于下文阐述。

- “TimeWidth”(时间宽度)

  程序模式内各步的时间宽度。
  程序模式、步的详情将于下文阐述。

- “WaitWidth”(等待判定宽度)

  判定程序模式内各步是否执行等待功能的阈值。
  等待功能的详情将于下文阐述。

- “WaitTimeLimit”(等待时间上限)

  程序模式内各步执行等待功能后等待时间的上限值。
  “WaitTimeLimit”的值为T#0s时，表示等待时间的上限值无限大。
  等待功能的详情将于下文阐述。

- “Wait”(等待中)

  表示是否正在执行等待功能的标志。
  “Wait”的值为TRUE，则正在执行等待功能。
  等待功能的详情将于下文阐述。

- “StepNo”(当前步号)

  当前执行中的步号。
  程序模式、步的详情将于下文阐述。

- “PresentSP”(当前目标值)

  前述的当前目标值。

- “PredictSP”(预测目标值)

  前述的预测目标值。

- “ProgramTime”(程序时间)

  程序模式的步0至 “EndStepNo”的 “TimeWidth”的合计值。
  程序模式、步的详情将于下文阐述。

- “ElapseTime”(经过时间)

  开始执行本指令至当前的时间。但是，含等待时间，不含保持时间。
  等待功能和保持功能的详情将于下文阐述。

- “ProgressTime”(进度时间)

  开始执行本指令至当前的时间。但是，不含等待时间和保持时间。
  等待功能和保持功能的详情将于下文阐述。

- **“LeftTime”(剩余时间)**

  当前至全部处理结束的时间。但是，不含等待时间和保持时间。
  等待功能和保持功能的详情将于下文阐述。

- **“StepProgressTime”(步内进度时间)**

  程序模式内执行中的步开始至当前的时间。但是，不含等待时间和保持时间。
  程序模式、步、等待功能、保持功能的详情将于下文阐述。

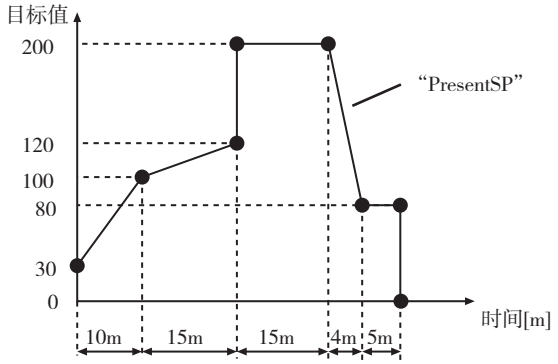
- **“StepLeftTime”(步内剩余时间)**

  程序模式内当前至执行中的步处理结束的时间。但是，不含等待时间和保持时间。
  程序模式、步、等待功能、保持功能的详情将于下文阐述。

程序模式

程序模式是指，将本指令执行开始到结束的处理分为多个步，按时间顺序排列每步的到达目标值和时间宽度。程序模式由以数据类型_sAC_STEP_DATA为元素的数组ProgramPattern[]表示。ProgramPattern[]的各元素对应各步。
程序模式的示例如下所示。ProgramPattern[] 各元素的 “ReachSP”和 “TimeWidth”的值如下表所示时，开始执行本指令起的时间和目标值之间的关系如下图所示。

	ProgramPattern[]的元素编号							
	0	1	2	3	4	5	6	7
步号	0	1	2	3	4	5	6	7
“ReachSP”的值	30	100	120	200	200	80	80	0
“TimeWidth”的值	T#0s	T#10m	T#15m	T#0s	T#15m	T#4m	T#5m	T#0s



对各步的目标值进行线性插补，算出各时间点的 “PresentSP”的值。上图实线所示为 “PresentSP”。按任务周期，输出各自在该时间点的 “PresentSP”的值。

- **“TimeWidth”的值和步的时间宽度之间的关系**

  “TimeWidth”的值和步的时间宽度之间的关系如下所示。

“TimeWidth”的值	步号	步的时间宽度
T#0s	0	视为T#0s。
	0以外	视为1个任务周期。
正值	–	“TimeWidth”的值为步的时间宽度。
负值	–	视为1个任务周期。

● 步的时间宽度不足1个任务周期时的动作

步的时间宽度的分辨率为1个任务周期。步的时间宽度不足1个任务周期时的动作如下所示。

步号	步的时间宽度	动作
0	T#0s	步0的 “ReachSP” 的值变为 “PresentSP” 的初始值。实际处理从步1开始。
	T#0s以外	仅1个任务周期进行该步的处理后，处理转至下一个步。
0以外	–	

开始步号 “StartStepNo” 和结束步号 “EndStepNo”

可将程序模式内的任意步设定为处理的开始步和结束步。在 “StartStepNo” 中设定开始步号，在 “EndStepNo” 中设定结束步号。
例如，设 “StartStepNo” =3、“EndStepNo” =6，执行本指令时，处理从步3开始，在步6结束。

● 指令执行中的 “StartStepNo”、“EndStepNo” 的值的变更

在执行指令的过程中，可变更 “StartStepNo” 和 “EndStepNo” 的值。变更值时，动作如下所示。

变量	要变更的步号	动作
“StartStepNo”	–	从变更后的 “StartStepNo” 的开头开始处理。
“EndStepNo”	变更为执行中的步号以上的值	变更后的 “EndStepNo” 结束时，处理结束。
	变更为执行中的步号以下的值	变更后，处理结束。 “Done” 的值变为TRUE。

等待功能

“PV” 的值可能会因控制对象的延迟而无法达到执行中的步的 “TimeWidth” 内的 “ReachSP” 的值。这种情况下，超出 “TimeWidth” 的时间宽度，保持执行中步的处理的功能，称为等待功能。
等待功能相关的变量为 ProgramPattern[] 内的等待判定值 “WaitWidth”、等待时间上限 “WaitTimeLimit” 和等待中 “Wait”。

● 等待功能执行的判定

在执行中的步的结束时间，如 “ReachSP” 和 “PV” 的差超出 “WaitWidth”，则执行等待功能。

● 等待功能结束的时间

从开始执行等待功能到 “WaitTimeLimit” 以内的时间内，“ReachSP” 和 “PV” 的差在 “WaitWidth” 以下时，在该时间点结束执行等待功能，处理移至下一个步。
从开始执行等待功能到 “WaitTimeLimit” 以内的时间内，“ReachSP” 和 “PV” 的差不在 “WaitWidth” 以下时，经过 “WaitTimeLimit” 的时间后，结束执行等待功能，处理移至下一个步。但，“WaitTimeLimit” 的值为T#0s时，等待时间的上限值无限大。因此，在 “ReachSP” 和 “PV” 的差为 “WaitWidth” 以下前，将无时间限制地持续执行等待功能。

● 等待功能执行的监视

可通过监视等待中 “Wait” 的值，判断是否正在执行等待功能。在执行等待功能的过程中，“Wait” 的值变为TRUE。等待功能执行结束后，“Wait” 的值变为FALSE。

● 等待功能执行中的计时

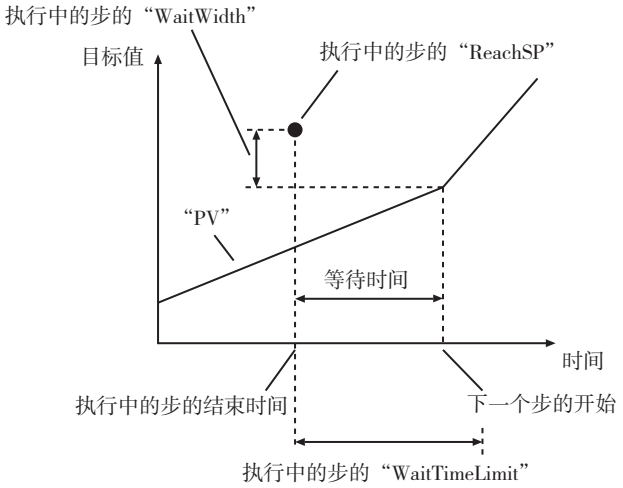
等待功能执行中时间相关的各变量的动作如下所示。

变量	动作
“ElapseTime”	继续计时。
“ProgressTime”	停止计时，保持等待功能执行开始时的值。等待功能执行结束后，从该值重新开始计时。
“LeftTime”	
“StepProgressTime”	变为执行中步的 “TimeWidth” 的值，保持该值。
“StepLeftTime”	变为0，保持该值。

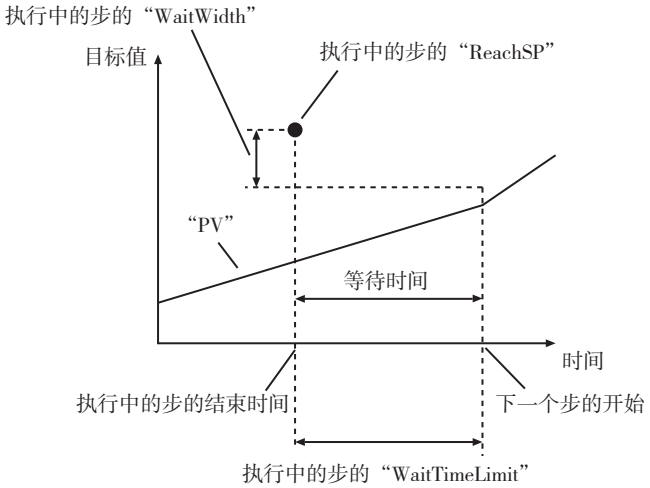
● 等待功能执行中的 “PresentSP” 和 “PredictSP”

等待功能执行中的 “PresentSP” 和 “PredictSP” 均保持 “ReachSP” 的值。

在 “WaitTimeLimit” 的时间以内，“ReachSP” 和 “PV” 的差为 “WaitWidth” 以下时 “PV” 的图表如下所示。即使到达执行中的步的结束时间，由于 “ReachSP” 和 “PV” 的差超出 “WaitWidth”，因此执行等待功能。“ReachSP” 和 “PV” 的差在 “WaitWidth” 以下时，开始下一个步的处理。



在 “WaitTimeLimit” 以内的时间内，“ReachSP” 和 “PV” 的差不在 “WaitWidth” 以下时 “PV” 的图表如下所示。经过 “WaitTimeLimit” 的时间后，开始下一个步的处理。



保持功能

保持功能是，将保持 “Hold”的值设为TRUE，强制保持执行中步的处理的功能。保持功能执行中，时间相关的所有变量的计时停止。
将 “Hold”的值设为FALSE时，重启时间相关的变量的计时。

● 保持功能执行中的计时

保持功能执行中，时间相关的各变量的动作如下所示。

变量	动作
“ElapseTime”	停止计时，保持开始执行保持功能时的值。保持功能执行结束后，从该值重新开始计时。
“ProgressTime”	
“LeftTime”	
“StepProgressTime”	
“StepLeftTime”	

● 保持功能执行中的 “PresentSP”和 “PredictSP”

保持功能执行中的 “PresentSP”，保持开始执行保持功能时的值。
保持功能执行中的 “PredictSP”的值与 “PresentSP”相同。

● 等待功能执行中的保持功能执行

在等待功能执行中执行保持功能时，等待功能的执行暂时结束。因此“Wait”的值变为FALSE。其后，在保持功能执行结束时，重新进行等待功能执行的判定。

当前值开始功能

当前值开始功能是从 “PV”的当前值和 “PresentSP”的值一致时开始处理的功能。在当前值开始“StartAtPV”的值设为TRUE的状态下，将 “Enable”的值设为TRUE时，执行当前值开始功能。
执行当前值开始功能时，将按下列步骤进行处理。

1 **获取 “PV”的当前值。**

2 **从步0到最终步，将检索到 “PV”的当前值和 “PresentSP”的值最初一致的时刻。**
但，“PresentSP”的值从步0的开头上升时，只能检索到 “PresentSP”的值刚刚下降之前。同样，“PresentSP”的值从步0的开头下降时，只能检索到 “PresentSP”的值刚刚上升之前。

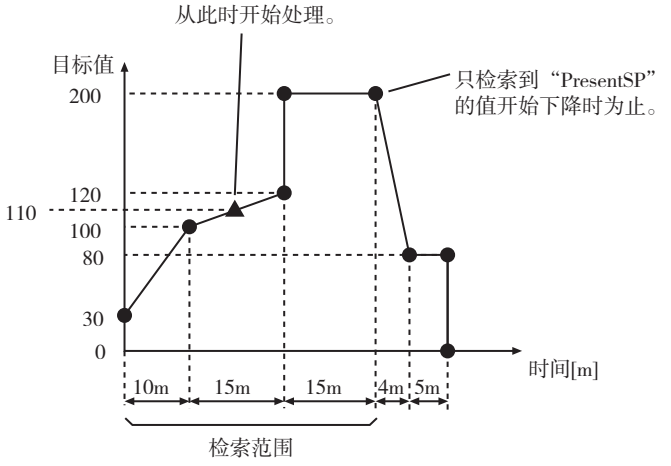
3 **从在上边检索到的时间点开始处理。**

在检索范围中，不存在 “PV”的当前值和 “PresentSP”的值一致的时间点时，从步0开始处理。

当前值开始功能执行示例如下所示。ProgramPattern[]的内容如下表所示。

	ProgramPattern[]的元素编号							
	0	1	2	3	4	5	6	7
步号	0	1	2	3	4	5	6	7
“ReachSP”的值	30	100	120	200	200	80	80	0
“TimeWidth”的值	T#0s	T#10m	T#15m	T#0s	T#15m	T#4m	T#5m	T#0s

“PresentSP”的值从步0的值开始上升。因此，只能检索到 “PresentSP”的值开始下降，开始处理后40m的时间点为止。
设本指令执行开始时 “PV”的值为110。此时，“PresentSP”=110，从下图所示的时间点开始处理。



● 当前值开始功能执行时的计时

当前值开始功能执行中，时间相关的各变量的动作如下所示。

变量	动作
“ElapseTime”	0。
“ProgressTime”	步0至已检索到的时间点的时间。
“LeftTime”	当前至 “EndStepNo”结束的时间。
“StepProgressTime”	执行中的步的开头至已检索到的时间点的时间。
“StepLeftTime”	当前至执行中的步的结束时间。

● 指令执行中的 “StartAtPV”的值的变更

本指令执行中即使变更 “StartAtPV”的值，也将被忽略。

前进功能

在执行本指令的过程中，“Advance”的值从FALSE变更为TRUE时，处理将移至下一个步的开头。

● 前进功能执行时的计时

前进功能执行中，时间相关的各变量的动作如下所示。

变量	动作
“ElapseTime”	继续计时。
“ProgressTime”	步0至执行中的步的 “TimeWidth”的合计。
“LeftTime”	下一个步至 “EndStepNo”结束的时间。
“StepProgressTime”	处理移至下一个步的开头，因此变为0。
“StepLeftTime”	下一个步的 “TimeWidth”的值。

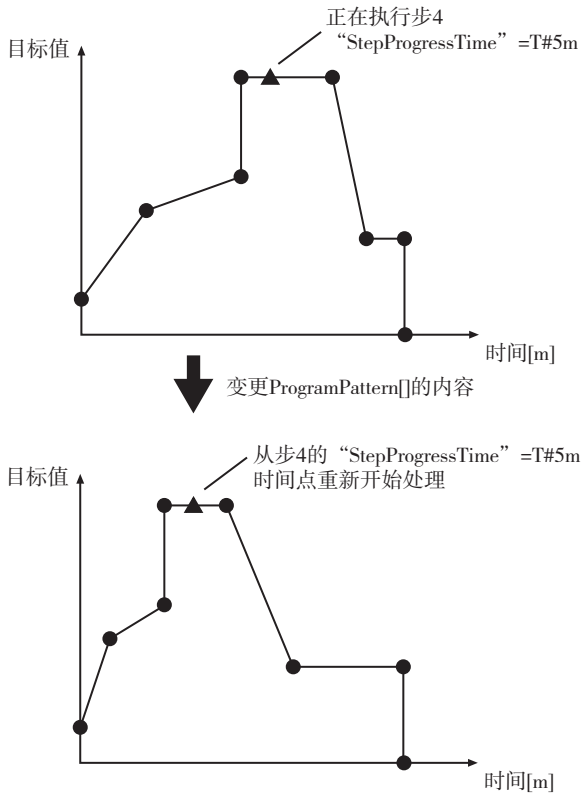
● “StartStepNo”的值变更和前进功能的并用

变更 “StartStepNo”的值的同时，将 “Advance”的值从FALSE变更为TRUE时，“StartStepNo”的值的变更优先。因此，处理移至 “StartStepNo”的开头。

指令执行中程序模式的变更

在执行指令的过程中，可变更ProgramPattern[]的内容。变更ProgramPattern[]的内容时，将重新计算“PresentSP”。从变更前执行的步号的 “StepProgressTime”的时间点重新开始处理。过去的步的内容可以变更。

例如，假设执行步4的过程中，已变更ProgramPattern[]的内容。假设变更前的 “StepProgressTime”的值为T#5m。变更后，从步4的 “StepProgressTime”=T#5m时间点重新开始处理。



变更后该步的 “TimeWidth”的值小于 “StepProgressTime”的值时，从下一个步的开头重新开始处理。

● 指令执行中变更程序模式时的计时

指令执行中变更程序模式时，时间相关的各变量的动作如下所示。

变量	动作
“ProgramTime”	变更后步0至 “EndStepNo”的 “TimeWidth”的合计。
“ElapseTime”	继续计时。
“ProgressTime”	变更后步0至执行中步号前1步的 “TimeWidth”的合计加上 “StepProgressTime”后的值。
“LeftTime”	变更后的当前至 “EndStepNo”结束的时间。
“StepProgressTime”	从变更前的值继续计时。
“StepLeftTime”	变更后的执行中的步的当前至步结束的时间。

● 等待功能执行中的程序模式变更

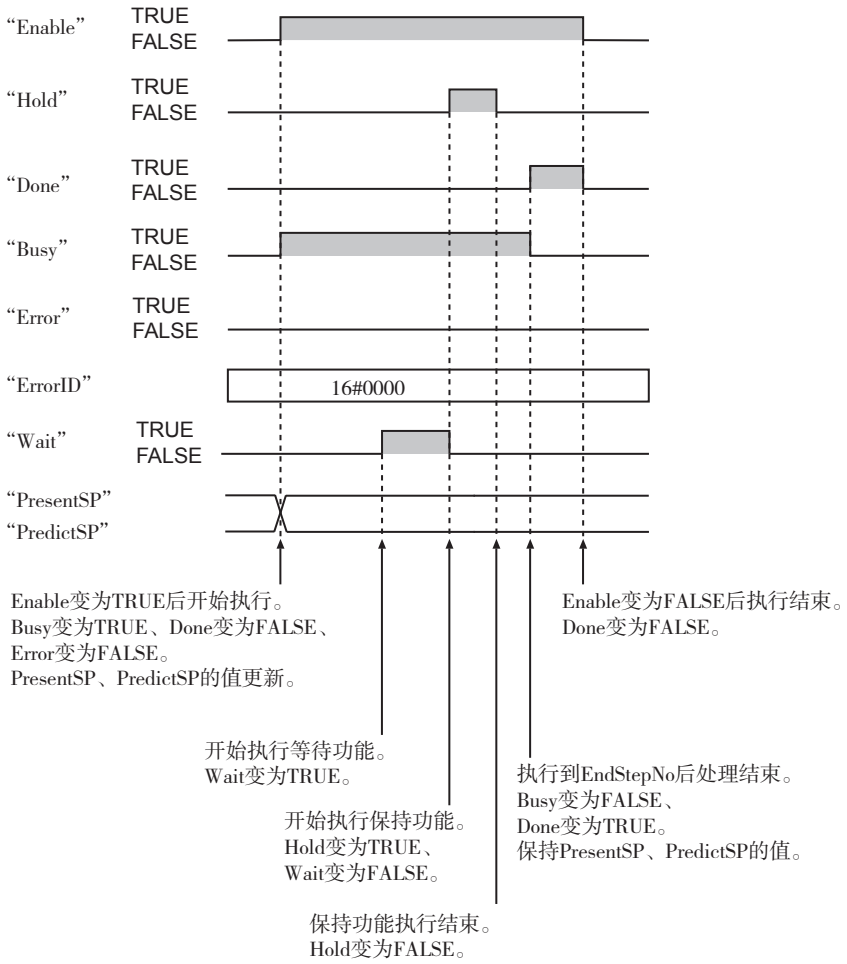
等待功能执行中变更程序模式后，对于新计算出的 “PresentSP”，重新进行等待功能执行的判定。但，“StepProgressTime”的值大于变更后的 “WaitTimeLimit”的值时，立即结束等待功能执行，移至下一个步进行处理。

● 保持功能执行中的程序模式变更

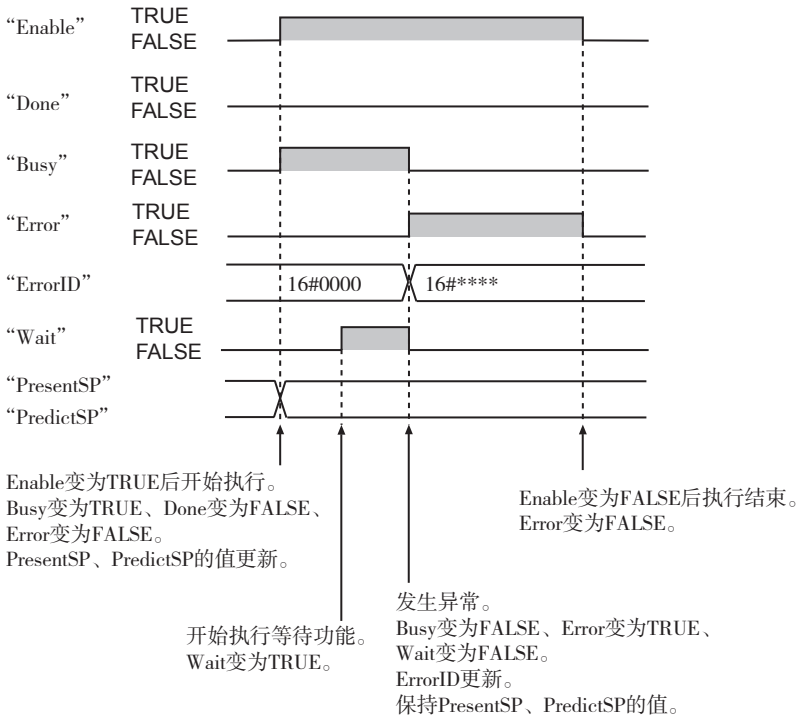
保持功能执行中变更程序模式后，对于新计算出的 “PresentSP”，继续执行保持功能。

时序图

正常时的时序图如下所示。



发生异常时的时序图如下所示。



使用注意事项

- 以下情况时会发生异常。“Error”变为TRUE，将异常ID代入 “ErrorID”。

异常的内容	“ErrorID”的值
“IntegrationTime”、“Alpha”、“StartStepNo”、“EndStepNo”中的任意一个值超过有效范围。	16#0400
ProgramPattern[]数组的最终元素编号超出99。	16#0416

版本相关信息

本指令可用于CPU单元Ver.1.06以上且Sysmac Studio Ver.1.07以上。

示例程序

使用AC_StepProgram指令的每步的最佳PID参数，进行温度控制的示例程序。

处理内容

本示例程序进行以下2个处理。

- 每步的最佳PID参数计算
- 基于程序模式的温度控制

下面对各处理进行说明。

● 每步的最佳PID参数计算

进行基于程序模式的温度控制之前，预先计算对于各步最佳的PID参数。计算PID参数时，使用PIDAT指令的自动调谐功能。
算出的PID参数保存在下标带步号的结构体类型数组PIDbank[]中。PIDbank[]元素的各个结构要素表示比例带、积分时间、微分时间的值。

处理步骤如下所示。

1 **用户将AC_StepProgram指令的动作指令 “ACSP_Enable”的值设为TRUE。**
AC_StepProgram指令启动，当前步号 “StepNo”的值变为0。

2 **用户将PIDAT指令的执行条件 “Run”的值设为TRUE。**
PIDAT指令启动。

3 **用户将自动调谐执行条件 “StartAT”的值从FALSE改设为TRUE。**
AC_StepProgram指令的保持 “Hold”的值变为TRUE，执行保持功能。
执行PIDAT指令的自动调谐功能，算出步0对应的最佳PID参数。

4 **自动调谐结束。**
PIDAT指令的自动调谐正常结束 “ATDone”的值变为TRUE。
算出的PID参数保存在PIDbank[0]中。

5 **用户将AC_StepProgram指令的 “Hold”的值设为FALSE。**
解除AC_StepProgram指令的保持功能。
稍过一段时间后，转至下一个步，“StepNo”的值变为1。

6 **用户仅按照步数反复执行上述3 ~ 5。**
所有步对应的最佳PID参数保存在PIDbank[]中。

● 基于程序模式的温度控制

使用各步对应的最佳PID参数，执行基于程序模式的温度控制。
处理步骤如下所示。

1 用户将AC_StepProgram指令的动作指令 “ACSP_Enable”的值设为TRUE。
AC_StepProgram指令启动，步号 “StepNo”的值变为0。

2 用户将PIDAT指令的执行条件 “Run”的值设为TRUE。
PIDAT指令启动。

3 按任务周期，输出PIDAT指令的操作量 “MV”。

4 使用TimeProportionalOut指令，执行 “MV”值对应的分时比例输出。

5 稍过一段时间后，移至下一个步。

6 反复执行上述3 ~ 5直至结束步。

Sysmac Studio的设定

执行本示例程序时，必须事先在Sysmac Studio中设定网络构成、I/O映射、数据类型定义。

● 网络设定

网络构成如下所示。在EtherCAT的节点地址1连接了下列构成的从站终端。分别分配以下设备名称。

单元编号	型号	单元	设备名称
0	NX-ECC201	EtherCAT耦合器单元	E001
1	NX-TS2101	温度输入单元	N1
2	NX-OD3121	数字输出单元	N2

● I/O映射

I/O映射设定如下所示。

位置	端口	说明	R/W	数据类型	变量	变量种类
Unit1	Ch1 Measured Value REAL*1	通道的测量值(REAL型)	R	REAL	N1_Ch1_Measured_Value_REAL	全局变量
Unit1	Ch2 Measured Value REAL*2	通道的测量值(REAL型)	R	REAL	N1_Ch2_Measured_Value_REAL	全局变量
Unit2	Output Bit 00	输出触点00	W	BOOL	N2_Output_Bit_00	全局变量
Unit2	Output Bit 01	输出触点01	W	BOOL	N2_Output_Bit_01	全局变量
Unit2	Output Bit 02	输出触点02	W	BOOL	N2_Output_Bit_02	全局变量
Unit2	Output Bit 03	输出触点03	W	BOOL	N2_Output_Bit_03	全局变量

*1 温度输入单元NX-TS2101的I/O入口，必须事先追加0x6003:01(Ch1 Measured Value REAL)。
*2 温度输入单元NX-TS2101的I/O入口，必须事先追加0x6003:02(Ch2 Measured Value REAL)。

● 数据类型定义

如下定义结构体型sPID_BANK。

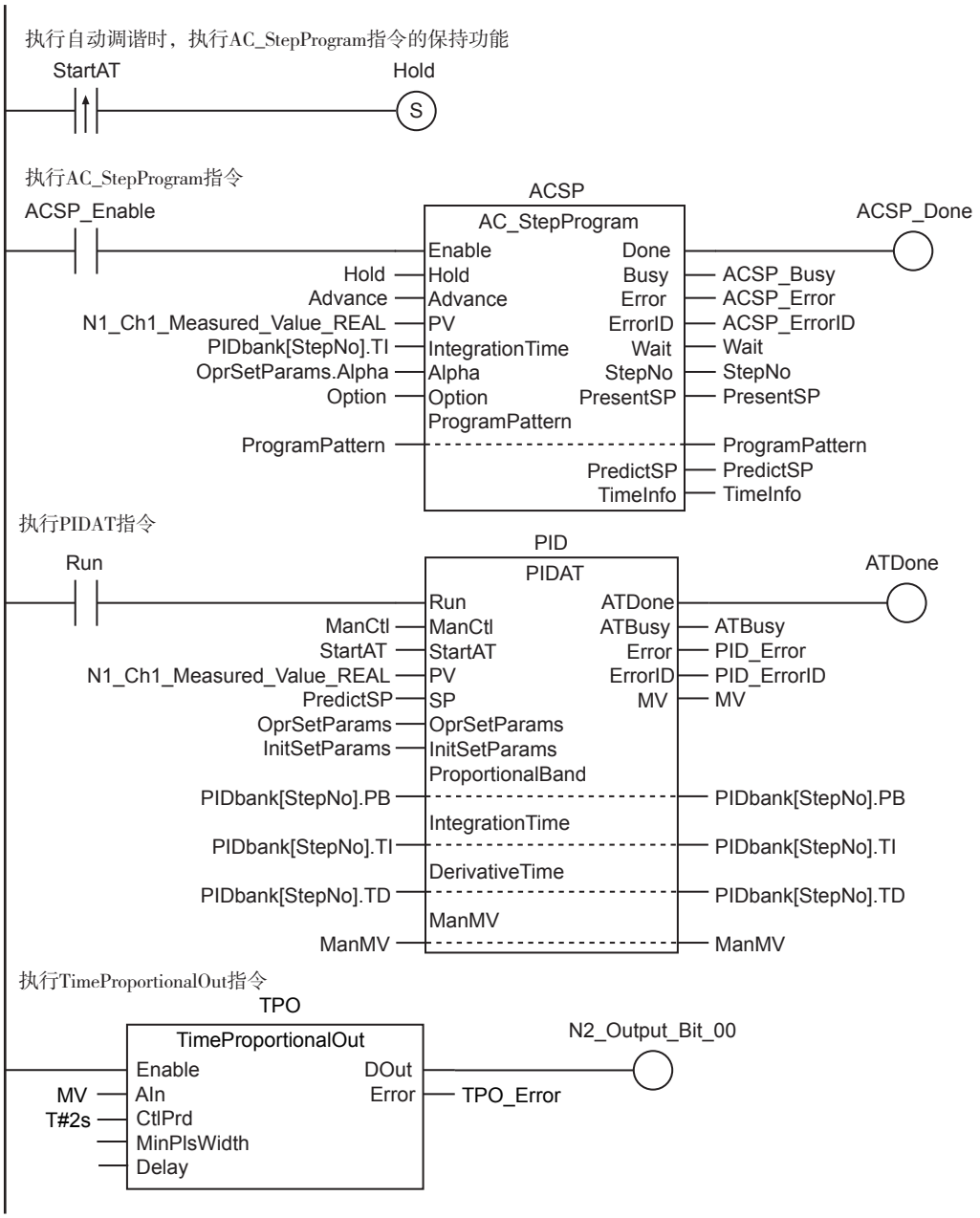
结构体型		名称	数据类型	注释
	▼	sPID_BANK	STRUCT	PID参数结构体
		PB	REAL	比例带
		TI	TIME	积分时间
		TD	TIME	微分时间

LD

内部变量	名称	数据类型	初始值	注释
	ACSP_Enable	BOOL	FALSE	AC_StepProgram的动作指令
	Hold	BOOL	FALSE	保持
	Advance	BOOL	FALSE	前进
	Option	_sAC_STEP_OPTION	(StartAtPV:=FALSE, StartStepNo:=0, EndStepNo:=7, Reserved:=[32(16#0)])	选项
	ProgramPattern	ARRAY[0..7] OF _sAC_STEP_DATA	[(ReachSP:=30.0, TimeWidth:=T#0s, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=100.0, TimeWidth:=T#10m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=120.0, TimeWidth:=T#15m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=150.0, TimeWidth:=T#0s, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=150.0, TimeWidth:=T#15m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=80.0, TimeWidth:=T#4m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=80.0, TimeWidth:=T#5m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=10.0, TimeWidth:=T#0s, WaitWidth:=3.0, WaitTimeLimit:=T#1m)]	程序模式

内部变量	名称	数据类型	初始值	注释
	ACSP_Busy	BOOL	FALSE	AC_StepProgram执行中
	ACSP_Error	BOOL	FALSE	AC_StepProgram错误
	ACSP_ErrorID	WORD	WORD#16#0	AC_StepProgram错误代码
	Wait	BOOL	FALSE	等待中
	StepNo	USINT	0	当前步号
	PresentSP	REAL	0.0	当前目标值
	PredictSP	REAL	0.0	预测目标值
	TimeInfo	_sAC_STEP_TIME	(ProgramTime:=T#0s, ElapseTime:=T#0s, ProgressTime:=T#0s, LeftTime:=T#0s, StepProgressTime:=T#0s, StepLeftTime:=T#0s)	时间信息
	ACSP_Done	BOOL	FALSE	AC_StepProgram执行完成
	Run	BOOL	FALSE	PIDAT指令执行条件
	ManCtl	BOOL	FALSE	手动/自动切换
	StartAT	BOOL	FALSE	自动调谐执行条件
	OprSetParams	_sOPR_SET _PARAMS	(MVLowLmt:=0.0, MVUpLmt:=100.0, ManResetVal:=0.0, MVTrackSw:=FALSE, MVTrackVal:=0.0, StopMV:=0.0, ErrorMV:=0.0, Alpha:=0.65, ATCalcGain:=1.0, ATHystrs:=0.2)	运行中设定参数
	InitSetParams	_sINIT_SET _PARAMS	(SampTime:=T#250ms, RngLowLmt:=-200.0, RngUpLmt:=1300.0, DirOpr:=FALSE)	初始设定参数
	ManMV	REAL	0.0	手动操作量
	ATBusy	BOOL	FALSE	自动调谐执行中
	PID_ErrorID	WORD	WORD#16#0	PIDAT错误代码
	PID_Error	BOOL	FALSE	PIDAT错误
	MV	REAL	0.0	操作量
	ATDone	BOOL	FALSE	自动调谐正常结束
	TPO_Error	BOOL	FALSE	TimeProportionalOut错误
	PIDbank	ARRAY[0..7] OF sPID_BANK	[8((PB:=10, TI:=T#233s, TD:=T#60s))]	最佳PID参数保存数组
	ACSP	AC_StepProgram		
	PID	PIDAT		
	TPO	TimeProportionalOut		

外部变量	名称	数据类型	常数	注释
	N1_Ch1_Measured_Value_REAL	REAL	☐	通道的测量值(REAL型)
	N2_Output_Bit_00	BOOL	☐	输出触点



ST

内部变量	名称	数据类型	初始值	注释
	ACSP_Enable	BOOL	FALSE	AC_StepProgram的动作指令
	Hold	BOOL	FALSE	保持
	Advance	BOOL	FALSE	前进
	Option	_sAC_STEP_OPTION	(StartAtPV:=FALSE, StartStepNo:=0, EndStepNo:=7, Reserved:=[32(16#0)])	选项
	ProgramPattern	ARRAY[0..7] OF _sAC_STEP_DATA	[(ReachSP:=30.0, TimeWidth:=T#0s, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=100.0, TimeWidth:=T#10m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=120.0, TimeWidth:=T#15m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=150.0, TimeWidth:=T#0s, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=150.0, TimeWidth:=T#15m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=80.0, TimeWidth:=T#4m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=80.0, TimeWidth:=T#5m, WaitWidth:=3.0, WaitTimeLimit:=T#1m), (ReachSP:=10.0, TimeWidth:=T#0s, WaitWidth:=3.0, WaitTimeLimit:=T#1m)]	程序模式
	ACSP_Busy	BOOL	FALSE	AC_StepProgram执行中
	ACSP_Error	BOOL	FALSE	AC_StepProgram错误
	ACSP_ErrorID	WORD	WORD#16#0	AC_StepProgram错误代码
	Wait	BOOL	FALSE	等待中
	StepNo	USINT	0	当前步号
	PresentSP	REAL	0.0	当前目标值
	PredictSP	REAL	0.0	预测目标值

内部变量	名称	数据类型	初始值	注释
	TimeInfo	_sAC_STEP_TIME	(ProgramTime:=T#0s, ElapseTime:=T#0s, ProgressTime:=T#0s, LeftTime:=T#0s, StepProgressTime:=T#0s, StepLeftTime:=T#0s)	时间信息
	ACSP_Done	BOOL	FALSE	AC_StepProgram执行完成
	Run	BOOL	FALSE	PIDAT指令执行条件
	ManCtl	BOOL	FALSE	手动/自动切换
	StartAT	BOOL	FALSE	自动调谐执行条件
	PreStartAT	BOOL	TRUE	上个任务周期的自动调谐执行条件
	OprSetParams	_sOPR_SET _PARAMS	(MVLowLmt:=0.0, MVUpLmt:=100.0, ManResetVal:=0.0, MVTrackSw:=FALSE, MVTrackVal:=0.0, StopMV:=0.0, ErrorMV:=0.0, Alpha:=0.65, ATCalcGain:=1.0, ATHystrs:=0.2)	运行中设定参数
	InitSetParams	_sINIT_SET _PARAMS	(SampTime:=T#250ms, RngLowLmt:=-200.0, RngUpLmt:=1300.0, DirOpr:=FALSE)	初始设定参数
	ManMV	REAL	0.0	手动操作量
	ATBusy	BOOL	FALSE	自动调谐执行中
	PID_ErrorID	WORD	WORD#16#0	PIDAT错误代码
	PID_Error	BOOL	FALSE	PIDAT错误
	MV	REAL	0.0	操作量
	ATDone	BOOL	FALSE	自动调谐正常结束
	TPO_Error	BOOL	FALSE	TimeProportionalOut错误
	PIDbank	ARRAY[0..7] OF sPID_BANK	[8((PB:=10, TI:=T#233s, TD:=T#60s))]	最佳PID参数保存数组
	TPO_Enable	BOOL	FALSE	TimeProportionalOut的动作指令
	MinPlsWidth	REAL	0.0	最小脉冲宽度
	Delay	REAL	0.0	延迟
	ACSP	AC_StepProgram		
	PID	PIDAT		
	TPO	TimeProportionalOut		

外部变量	名称	数据类型	常数	注释
	N1_Ch1_Measured_Value_REAL	REAL	☐	通道的测量值(REAL型)
	N2_Output_Bit_00	BOOL	☐	输出触点

```
TPO_Enable := TRUE;

// 执行自动调谐时，执行AC_StepProgram指令的保持功能
IF StartAT AND PreStartAT=FALSE THEN
  Hold := TRUE;
END_IF;
PreStartAT := StartAT;

// 执行AC_StepProgram指令
IF ACSP_Enable THEN
  ACSP(Enable       :=ACSP_Enable,
       Hold         :=Hold,
       Advance      :=Advance,
       PV           :=N1_Ch1_Measured_Value_REAL,
       IntegrationTime :=PIDbank[StepNo].TI,
       Alpha        :=OprSetParams.Alpha,
       Option       :=Option,
       ProgramPattern :=ProgramPattern,
       Done         =>ACSP_Done,
       Busy         =>ACSP_Busy,
       Error        =>ACSP_Error,
       ErrorID      =>ACSP_ErrorID,
       Wait         =>Wait,
       StepNo       =>StepNo,
       PresentSP    =>PresentSP,
       PredictSP    =>PredictSP,
       TimeInfo     =>TimeInfo);
END_IF;

// 执行PIDAT指令
IF Run THEN
  PID(Run            :=Run,
      ManCtl         :=ManCtl,
      StartAT        :=StartAT,
      PV             :=N1_Ch1_Measured_Value_REAL,
      SP             :=PredictSP,
      OprSetParams   :=OprSetParams,
      InitSetParams  :=InitSetParams,
      ProportionalBand :=PIDbank[StepNo].PB,
      IntegrationTime  :=PIDbank[StepNo].TI,
      DerivativeTime   :=PIDbank[StepNo].TD,
      ManMV          :=ManMV,
      ATDone         =>ATDone,
      ATBusy         =>ATBusy,
      Error          =>PID_Error,
      ErrorID        =>PID_ErrorID,
      MV=>MV);
END_IF;

// 执行TimeProportionalOut指令
TPO(Enable     :=TPO_Enable,
    AIn        :=MV,
    CtlPrd     :=T#2s,
    MinPlsWidth:=MinPlsWidth,
```

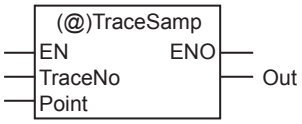
```
Delay    :=Delay,
DOut     =>N2_Output_Bit_00,
Error    =>TPO_Error);
```

系统控制指令

指令	名称	页码
TraceSamp	数据跟踪采样	2–784
TraceTrig	数据跟踪触发条件成立	2–787
GetTraceStatus	数据跟踪状态读取	2–790
SetAlarm	用户异常发生	2–793
ResetAlarm	用户异常解除	2–798
GetAlarm	用户异常状态获取	2–800
ResetPLCError	PLC异常解除	2–802
GetPLCError	PLC异常状态获取	2–805
ResetCJBError	I/O总线异常解除	2–807
GetCJBError	I/O总线异常状态获取	2–809
GetEIPError	EtherNet/IP异常状态获取	2–811
ResetMCError	运动控制异常解除	2–813
GetMCError	运动控制异常状态获取	2–818
ResetECError	EtherCAT异常解除	2–820
GetECError	EtherCAT异常状态获取	2–822
ResetNXBError	NX总线异常解除	2–825
GetNXBError	NX总线异常状态获取	2–825
GetNXUnitError	NX单元异常状态获取	2–829
SetInfo	用户信息生成	2–836
ResetUnit	单元重启	2–838
GetNTPStatus	NTP状态读取	2–842
RestartNXUnit	NX单元重启	2–844
NX_ChangeWriteMode	NX单元写入模式变更	2–849
NX_SaveParam	NX单元参数保存	2–854
NX_ReadTotalPowerOnTime	NX单元累计通电时间读取	2–859
PLC_ReadTotalPowerOnTime	PLC累计通电时间读取	2–866

TraceSamp

执行数据跟踪的采样。

指令	名称	FB/FUN	图形表现	ST表现
TraceSamp	数据跟踪采样	FUN		TraceSamp(TraceNo, Point);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
TraceNo	跟踪编号	输入	跟踪编号	*1	–	0
Point	采样点编号		采样点编号	遵从数据类型		
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

*1 NX701 CPU单元、NJ501 CPU单元、NY系列控制器时为 “0 ~ 3”。
NX1P2 CPU单元、NJ301 CPU单元、NJ101 CPU单元时为 “0 ~ 1”。

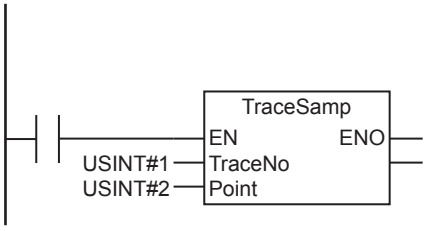
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
TraceNo						○														
Point						○														
Out	○																			

功能

执行数据跟踪的采样。
根据Sysmac Studio所指定的采样设定，读取采样对象的所有变量的当前值，加上跟踪编号 “TraceNo”和采样点编号 “Point”后，保存在跟踪存储器中。
但是，只有当正在执行数据跟踪，且基于Sysmac Studio的跟踪的采样时间的设定为 “采样指令”时，才会执行本指令。

示例如下所示。加上跟踪编号1、采样点编号2的值之后，将采样对象的所有变量的当前值保存在跟踪存储器中。

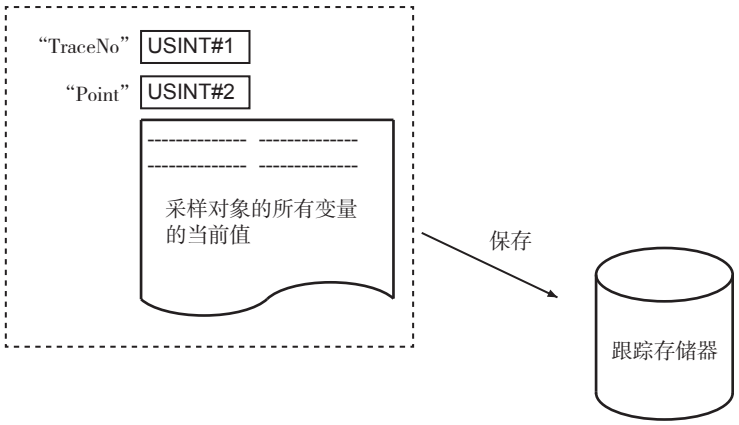
LD



ST

```
TraceSamp(USINT#1, USINT#2);
```

读取采样对象的所有变量的当前值，加上跟踪编号“TraceNo”和采样点编号“Point”后，保存在跟踪存储器中。



相关的系统定义变量

变量名称	名称	数据类型	内容
*1	跟踪信息	*2	跟踪信息*3

*1 NX701 CPU单元、NJ501 CPU单元、NY系列控制器时，变量名称为_PLC_TraceSta[0..3]。
NX1P2 CPU单元、NJ301 CPU单元、NJ101 CPU单元时，变量名称为_PLC_TraceSta[0..1]。

*2 _sTRACE_STA[ ]

*3 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

参考

- 数据跟踪功能的详情请参阅 “NJ/NX 系列 CPU 单元 用户手册 软件篇 (SBCA-359)”或 “NY 系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。
- 跟踪是指按指定的条件时间，对指定的变量的值进行采样的功能。各项条件在 Sysmac Studio 中进行指定。
- 本指令可以在程序中配置多个。可以根据条件写入采样程序。
- 通过适当设定 “Point”，能够从Sysmac Studio的跟踪窗口画面中确定是按哪个TraceSamp指令进行采样的值。省略了 “Point”时值为0。

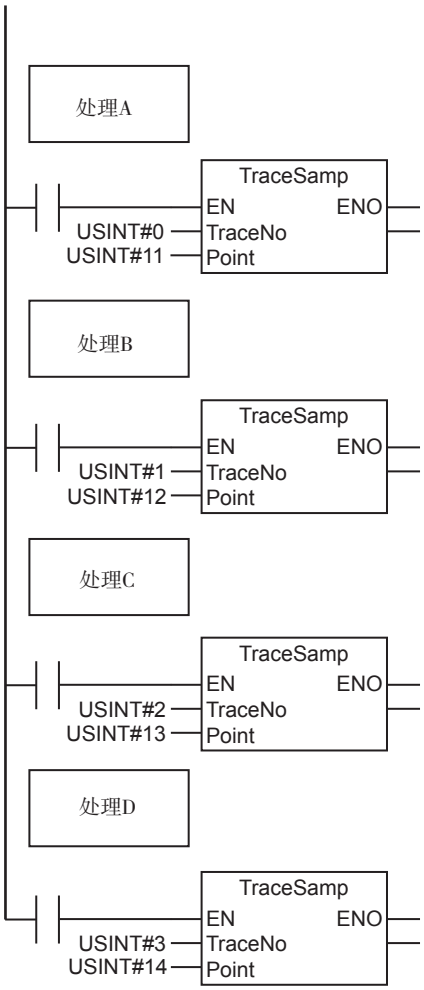
使用注意事项

- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 下列情况下，不执行任何操作而正常结束。
  - 数据跟踪停止时。
  - 跟踪设定的采样时间为 “采样指令”以外时。
  - “TraceNo”的值不是Sysmac Studio中设定的跟踪编号时。
- 以下情况时会发生异常。ENO为FALSE。
  - “TraceNo”的值超过有效范围时。

示例程序

在A ~ D的各项处理的完成时刻执行采样。
记录各时刻下各变量的值。

LD



ST

```
处理A
TraceSamp(USINT#0, USINT#11);
处理B
TraceSamp(USINT#1, USINT#12);
处理C
TraceSamp(USINT#2, USINT#13);
处理D
TraceSamp(USINT#3, USINT#14);
```

TraceTrig

触发数据跟踪。

指令	名称	FB/FUN	图形表现	ST表现
TraceTrig	数据跟踪触发条件成立	FUN	(@)TraceTrig EN ENO TraceNo Out	TraceTrig(TraceNo);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
TraceNo	跟踪编号	输入	跟踪编号	*1	–	0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

*1 NX701 CPU单元、NJ501 CPU单元、NY系列控制器时为 “0 ~ 3”。
NX1P2 CPU单元、NJ301 CPU单元、NJ101 CPU单元时为 “0 ~ 1”。

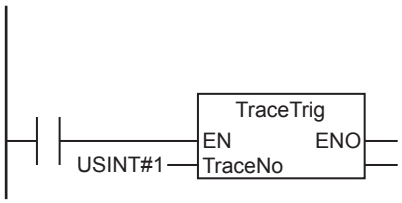
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
TraceNo						○														
Out	○																			

功能

强制触发数据跟踪。与通过Sysmac Studio指定的触发条件是否成立无关。
执行本指令时，如果正在执行跟踪编号 “TraceNo”的数据跟踪，则执行采样。

示例如下所示。触发跟踪编号1的数据跟踪。

LD



ST

```
TraceTrig(USINT#1);
```

触发跟踪编号 “TraceNo”的数据跟踪。

相关的系统定义变量

变量名称	名称	数据类型	内容
*1	跟踪信息	*2	跟踪信息*3

*1 NX701 CPU单元、NJ501 CPU单元、NY系列控制器时，变量名称为_PLC_TraceSta[0..3]。
NX1P2 CPU单元、NJ301 CPU单元、NJ101 CPU单元时，变量名称为_PLC_TraceSta[0..1]。

*2 _sTRACE_STA[ ]

*3 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”。

参考

- 数据跟踪功能的详情请参阅 “NJ/NX 系列 CPU 单元 用户手册 软件篇 (SBCA–359)”或 “NY 系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”。
- 本指令可以在程序中配置多个。可以根据条件写入使触发条件成立的程序。
- 可以写入在变量间的对比等通常的触发条件所无法设定的时间使触发条件成立的程序。

使用注意事项

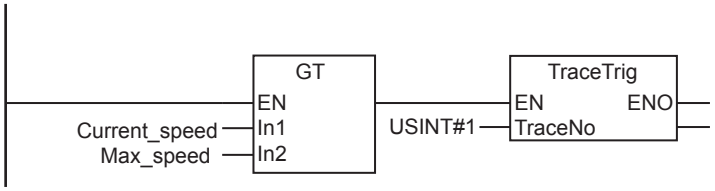
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 下列情况下，不执行任何操作而正常结束。
  - 数据跟踪停止时。
  - 触发条件已经成立时。
  - “TraceNo”的值不是Sysmac Studio中设定的跟踪编号时。
  - “TraceNo”中指定的跟踪编号的跟踪类型为 “连续跟踪”时。
- 以下情况时会发生异常。ENO为FALSE。
  - “TraceNo”的值超过有效范围时。

示例程序

当速度的当前值超过最大限度值时，强制触发数据跟踪，记录各变量的值。
当前速度Current_speed的值超过速度上限值Max_speed的值时，执行TraceTrig指令。

LD

名称	数据类型	初始值	注释
Current_speed	INT	0	当前速度
Max_speed	INT	20	速度上限值



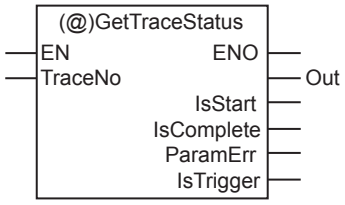
ST

名称	数据类型	初始值	注释
Current_speed	INT	0	当前速度
Max_speed	INT	20	速度上限值

```
IF (Current_speed > Max_speed) THEN
    TraceTrig(USINT#1);
END_IF;
```

GetTraceStatus

读取数据跟踪的执行状态。

指令	名称	FB/FUN	图形表现	ST表现
GetTraceStatus	数据跟踪状态读取	FUN		GetTraceStatus(TraceNo, IsStart, IsComplete, ParamErr, IsTrigger);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
TraceNo	跟踪编号	输入	跟踪编号	*1	–	0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
IsStart	执行中标志		TRUE：跟踪执行中 FALSE：未执行跟踪	遵从数据类型		
IsComplete	完成标志		TRUE：跟踪执行完毕 FALSE：跟踪执行中或未执行			
ParamErr	参数异常标志		TRUE：跟踪设定异常 FALSE：跟踪设定无异常			
IsTrigger	触发标志		TRUE：跟踪触发条件成立 FALSE：跟踪触发条件不成立			

*1 NX701 CPU单元、NJ501 CPU单元、NY系列控制器时为 “0 ~ 3”。
NX1P2 CPU单元、NJ301 CPU单元、NJ101 CPU单元时为 “0 ~ 1”。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
TraceNo						○														
Out	○																			
IsStart	○																			
IsComplete	○																			
ParamErr	○																			
IsTrigger	○																			

功能

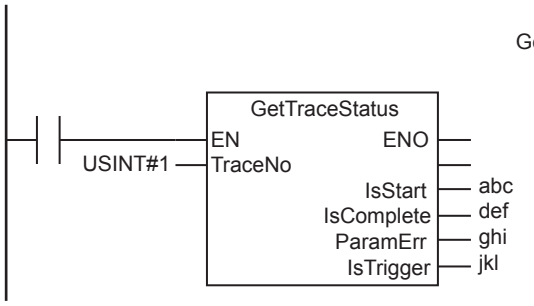
读取跟踪编号 “TraceNo”中指定的数据跟踪的执行状态。读取信息有执行中标志 “IsStart”、完成标志 “IsComplete”、参数异常标志 “ParamErr”、触发标志 “IsTrigger”。

“ParamErr”的值为TRUE说明跟踪设定存在如下异常。

- 触发设定、采样设定中指定的变量不存在。
- 采样时间的设定为 “指定任务周期”，但指定的任务不存在。

示例如下所示。读取跟踪编号1的执行状态。

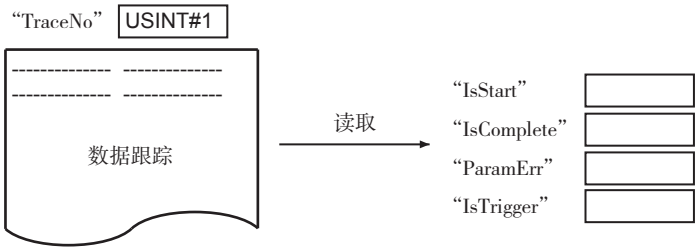
LD



ST

```
GetTraceStatus(USINT#1, abc, def, ghi, jkl);
```

读取跟踪编号“TraceNo”中指定的数据跟踪的执行状态。



相关的系统定义变量

变量名称	名称	数据类型	内容
*1	跟踪信息	*2	跟踪信息*3

*1 NX701 CPU单元、NJ501 CPU单元、NY系列控制器时，变量名称为_PLC_TraceSta[0..3]。
NX1P2 CPU单元、NJ301 CPU单元、NJ101 CPU单元时，变量名称为_PLC_TraceSta[0..1]。

*2 _sTRACE_STA[]

*3 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

参考

数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

使用注意事项

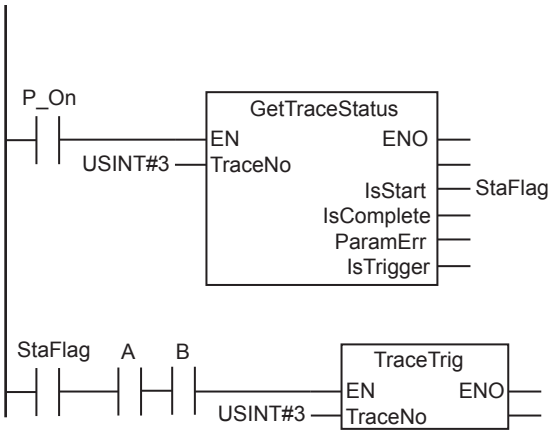
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 通过本指令读取的信息为系统定义变量 _PLC_TraceSta[ ]的内容。该变量无法直接访问。要读取该变量的内容，必须使用本指令。
- 以下情况时会发生异常。ENO为FALSE。
  - “TraceNo”的值超过有效范围时。

示例程序

通过GetTraceStatus指令读取跟踪编号3的执行状态。正在执行数据跟踪，且变量A与B的值均为TRUE时，执行TraceTrig指令，触发数据跟踪。

LD

名称	数据类型	初始值	注释
StaFlag	BOOL	FALSE	跟踪的执行状态
A	BOOL	FALSE	
B	BOOL	FALSE	



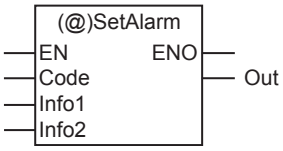
ST

名称	数据类型	初始值	注释
StaFlag	BOOL	FALSE	跟踪的执行状态
A	BOOL	FALSE	
B	BOOL	FALSE	

```
GetTraceStatus(TraceNo:=USINT#3, IsStart=>StaFlag);

IF ( (StaFlag=TRUE) AND (A=TRUE) AND (B=TRUE) ) THEN
    TraceTrig(TraceNo:=USINT#3);
END_IF;
```

SetAlarm

发生用户异常。

指令	名称	FB/FUN	图形表现	ST表现
SetAlarm	用户异常发生	FUN		SetAlarm(Code, Info1, Info2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Code	事件代码	输入	发生的用户异常的事件代码	1 ~ 40000	–	1
Info1	附属信息1		在发生用户异常的同时，事件日志中记录的值	遵从数据类型		(*)
Info2	附属信息2					
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Code							○													
Info1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	
Info2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	
Out	○																			

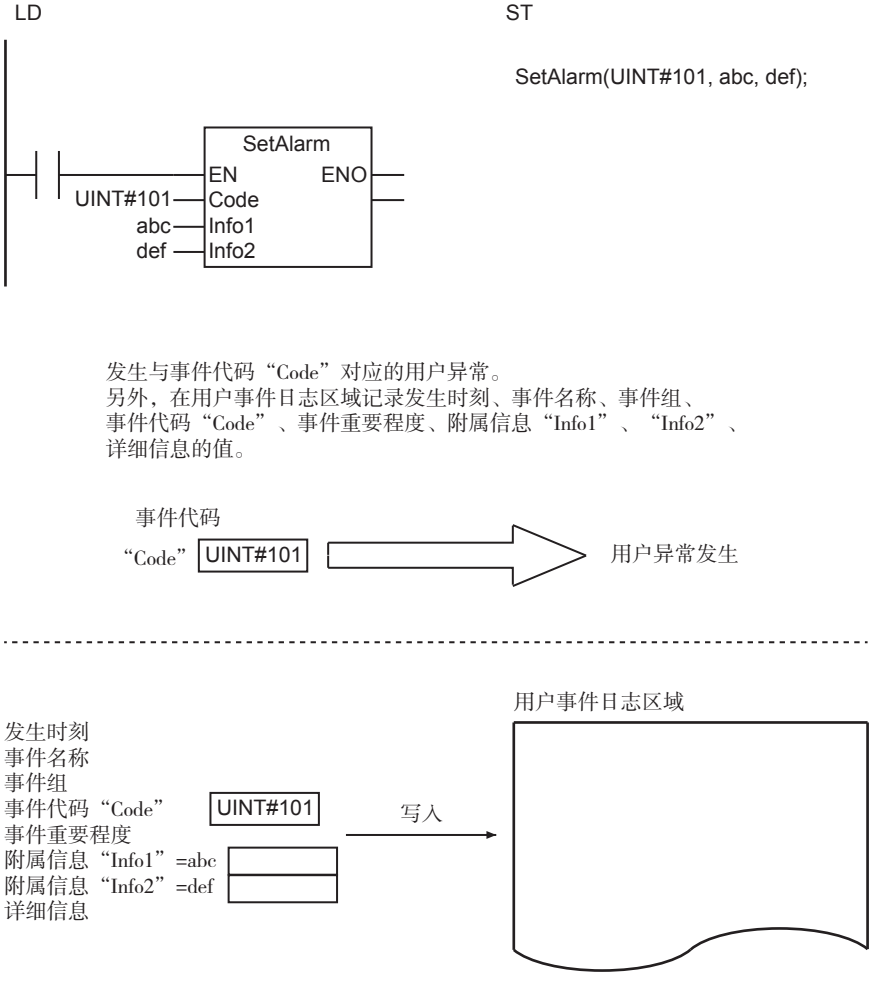
功能

发生与Sysmac Studio的事件设定表中定义的事件代码 “Code”对应的用户异常。
另外，在与事件代码的重要程度相对应的用户事件日志区域记录发生时刻、事件名称、事件组、事件代码 “Code”、事件重要程度、附属信息 “Info1”、“Info2”、详细信息的值。发生时刻记录自动获取的值。事件名称、事件组、详细信息则记录Sysmac Studio中设定的信息。

事件重要程度记录如下所示按事件代码规定的分类。分类编号越小，事件重要程度越高。

事件代码	分类 User fault Level
1 ~ 5000	1
5001 ~ 10000	2
10001 ~ 15000	3
15001 ~ 20000	4
20001 ~ 25000	5
25001 ~ 30000	6
30001 ~ 35000	7
35001 ~ 40000	8

示例如下所示。发生与事件代码101对应的用户异常。作为附属信息，记录变量abc、def的值。



相关的系统定义变量

变量名称	名称	数据类型	内容
_AlarmFlag	用户异常的异常状态	WORD	表示有无用户异常检测的标志。 从位0到位7分别表示用户异常的各种重要程度(1 ~ 8)的发生状况。

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

参考

“Info1”、“Info2”中既可以指定全局变量，也可以指定局部变量。

使用注意事项

- 可同时发生的用户异常为8种事件重要程度每种各32件(最多256件)。
- 已经发生相同事件代码的用户异常时，不在事件日志中进行记录。
- 请务必将传输至 “Info1”、“Info2”的输入参数设为变量。如果是常量，编连时会发生异常。
- “Code”的值中设定了Sysmac Studio中未设定的事件代码时，不会发生异常。此时，用户事件日志区域中不记录事件组和详细信息。另外，事件名称记录 “Code”的值。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE。
  - “Code”的值超过有效范围时。
  - 发生了超过最大件数的用户异常时。

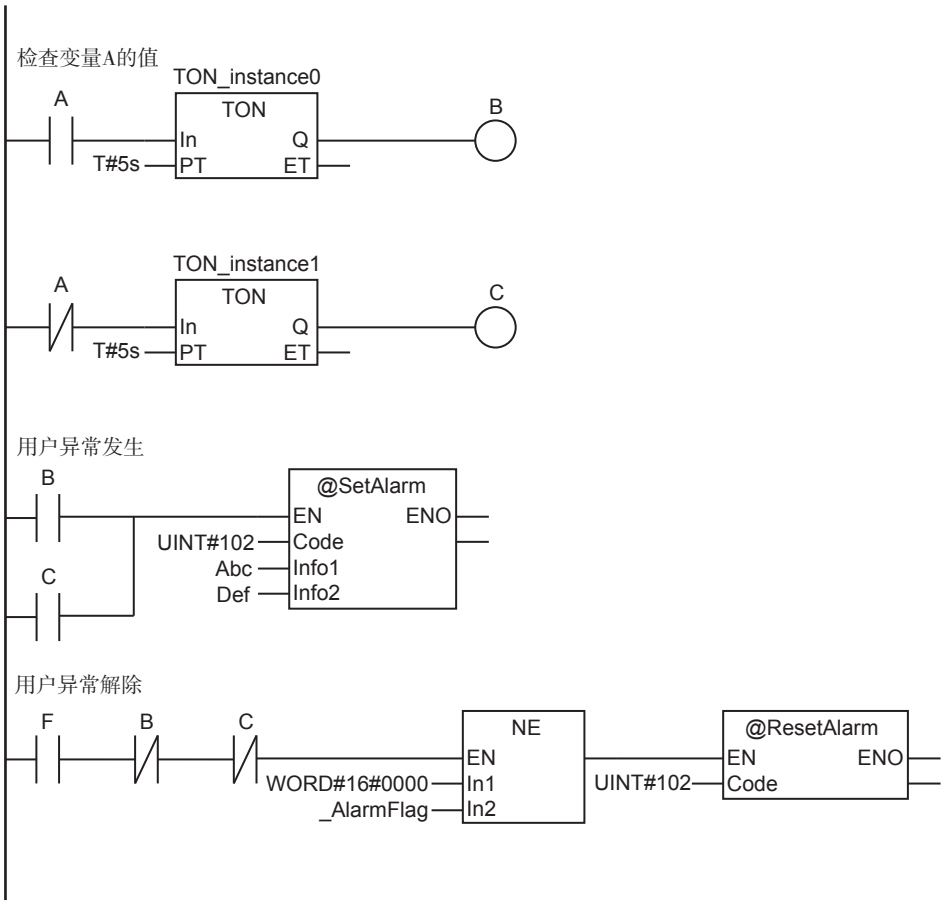
示例程序

变量A的值在正常时以5s间隔重复TRUE和FALSE。
监视A的值，值的变化量未达5s以上时，发生事件代码102的用户异常。附属信息为UINT#123和UINT#456。
变量F的值变为TRUE时，解除用户异常。

LD

内部变量	名称	数据类型	初始值
	A	BOOL	FALSE
	B	BOOL	FALSE
	C	BOOL	FALSE
	F	BOOL	FALSE
	Abc	UINT	123
	Def	UINT	456
	TON_instance0	TON	
	TON_instance1	TON	

外部变量	名称	数据类型	常数	注释
	_AlarmFlag	WORD	☑	用户异常的异常状态


检查变量A的值
TON_instance0
TON
A
B
In
Q
T#5s
PT
ET
TON_instance1
TON
A
C
In
Q
T#5s
PT
ET
用户异常发生
@SetAlarm
B
EN
ENO
UINT#102
Code
C
Abc
Info1
Def
Info2
用户异常解除
NE
@ResetAlarm
F
B
C
EN
EN
ENO
WORD#16#0000
In1
UINT#102
Code
_AlarmFlag
In2

ST

内部变量	名称	数据类型	初始值
	A	BOOL	FALSE
	B	BOOL	FALSE
	C	BOOL	FALSE
	F	BOOL	FALSE
	Abc	UINT	123
	Def	UINT	456
	TON_instance0	TON	
	TON_instance1	TON	

外部变量	名称	数据类型	常数	注释
	_AlarmFlag	WORD	☑	用户异常的异常状态

```
// 检查变量A的值
IF (A=TRUE) THEN
    TON_instance0(In:=TRUE, PT:=T#5s, Q=>B);
ELSE
    TON_instance0(In:=FALSE, Q=>B);
END_IF;

IF (A=FALSE) THEN
    TON_instance1(In:=TRUE, PT:=T#5s, Q=>C);
ELSE
    TON_instance1(In:=FALSE, Q=>C);
END_IF;

// 用户异常发生
IF (B=TRUE) OR (C=TRUE) THEN
    SetAlarm(
        Code:=UINT#102,
        Info1 :=Abc,
        info2 :=Def);
END_IF;

// 用户异常解除
IF (F=TRUE) & (B=FALSE) & (C=FALSE) & (_AlarmFlag<>WORD#16#0000) THEN
    ResetAlarm(Code:=UINT#102);
END_IF;
```

ResetAlarm

解除用户异常。

指令	名称	FB/FUN	图形表现	ST表现
ResetAlarm	用户异常解除	FUN	(@)ResetAlarm EN ENO Code Out	ResetAlarm(Code);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Code	事件代码	输入	解除的用户异常的事件代码。 0：解除所有的用户异常	0 ~ 40000	–	0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Code							○													
Out	○																			

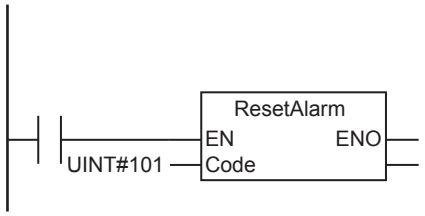
功能

解除事件代码 “Code”中指定的用户异常。
同时在用户事件日志区域中记录表示用户异常个别解除的事件。该事件的事件代码为No.65533，重要程度为"User Information"。

“Code”的值为0时，解除发生的所有用户异常。
此时，在用户事件日志区域中记录表示用户异常全部解除的事件。该事件的事件代码为No.65534，重要程度为"User Information"。

示例如下所示。解除与事件代码101对应的用户异常。

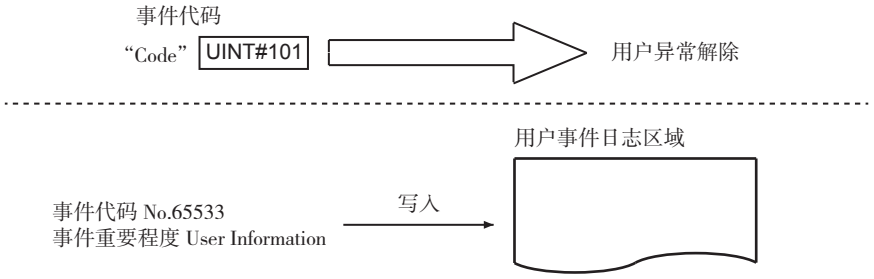
LD



ST

```
ResetAlarm(UINT#101);
```

解除事件代码“Code”中指定的用户异常。
并在用户事件日志区域中记录表示用户异常个别解除的事件。



相关的系统定义变量

变量名称	名称	数据类型	内容
_AlarmFlag	用户异常的异常状态	WORD	表示有无用户异常检测的标志。 从位0到位7分别表示用户异常的各种重要程度(1 ~ 8)的发生状况。

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”。

使用注意事项

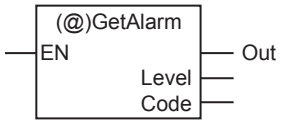
- 未发生 “Code”中指定的用户异常时，不属于异常。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO为FALSE。
  - “Code”的值超过有效范围时。

示例程序

请参阅 “SetAlarm指令(P.2–793)”的示例程序。

GetAlarm

获取发生的用户异常的最重要事件的重要程度(用户异常等级1 ~ 8)和最重要事件的代码。

指令	名称	FB/FUN	图形表现	ST表现
GetAlarm	用户异常状态获取	FUN		Out:=GetAlarm(Level, Code);

变量

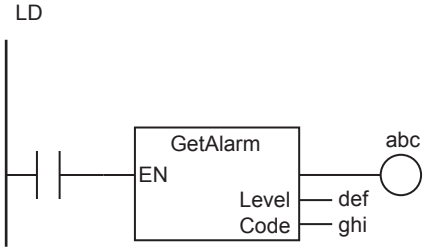
	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生用户异常 FALSE：未发生用户异常	遵从数据类型	–	–
Level	最重要事件重要程度		发生的用户异常中，最重要的事件重要程度 0　：无用户异常 1 ~ 8：事件重要程度	0 ~ 8		
Code	最重要事件代码		发生的用户异常中，最重要的事件代码 0　：无用户异常 1 ~ 40000：事件重要程度	0 ~ 40000		

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
Level							○													
Code							○													

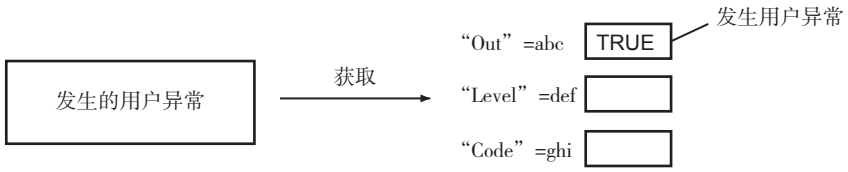
功能

获取发生的用户异常的最重要的事件重要程度 “Level”和最重要的事件代码 “Code”。
未发生用户异常时，有无异常 “Out”的值为FALSE。
发生了多个最重要的事件重要程度的用户异常时，“Code”的值为最早发生的用户异常的事件代码。

示例如下所示。



获取发生的用户异常的最重要的事件重要程度“Level”和最重要的事件代码“Code”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_AlarmFlag	用户异常的异常状态	WORD	表示有无用户异常检测的标志。 从位0到位7分别表示用户异常的各种重要程度(1 ~ 8)的发生状况。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

使用注意事项

通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。

ResetPLCError

解除PLC功能模块中发生的控制器异常。

指令	名称	FB/FUN	图形表现	ST表现
ResetPLCError	PLC异常解除	FB	ResetPLCError_instance ResetPLCError Execute Done Busy Error ErrorID	ResetPLCError(Execute, Done, Busy, Error, ErrorID);

变量

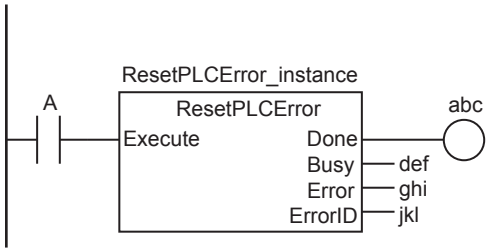
仅共通的变量

功能

解除PLC功能模块中发生的控制器异常。

示例如下所示。

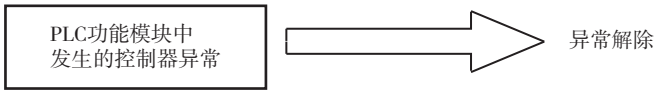
LD



ST

```
ResetPLCError_instance(A, abc, def, ghi, jkl);
```

解除PLC功能模块中发生的控制器异常。



相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_ErrSta	PLC功能模块异常状态	WORD	PLC功能模块中检测到的异常的状态。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)” 或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

使用注意事项

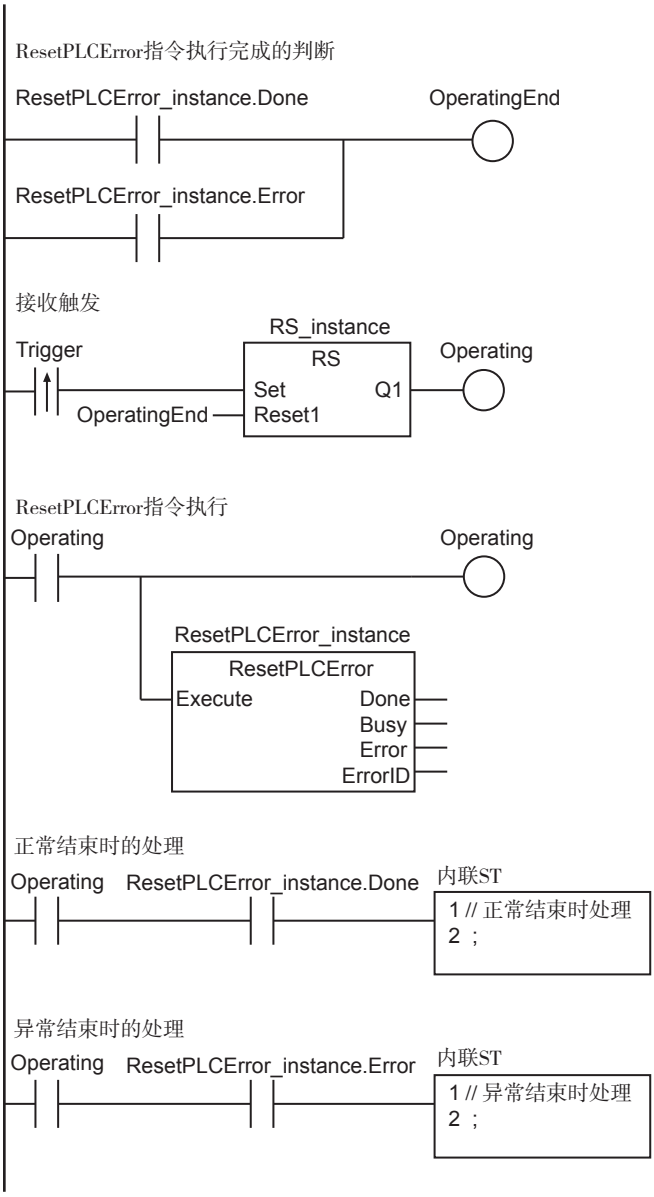
- 执行本指令之后，并不一定能够解除异常。请通过GetPLCError指令确认异常是否解除。

示例程序

如果Trigger的值由FALSE变为TRUE，则执行ResetPLCError指令。如果ResetPLCError指令正常结束(Done的值为TRUE)，则执行正常结束时处理。如果异常结束(Error的值为TRUE)，则执行异常结束时处理。

LD

名称	数据类型	初始值	注释
OperatingEnd	BOOL	FALSE	处理结束
Trigger	BOOL	FALSE	执行条件
Operating	BOOL	FALSE	处理中
RS_instance	RS		
ResetPLCError_instance	ResetPLCError		



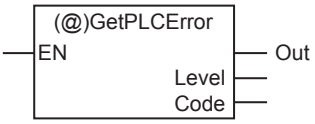
ST

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
OperatingStart	BOOL	FALSE	处理开始
Operating	BOOL	FALSE	处理中
ResetPLCError_instance	ResetPLCError		

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    OperatingStart:=TRUE;
    Operating:=TRUE;
END_IF;
LastTrigger:=Trigger;

// ResetPLCError_instance初始化
IF (OperatingStart=TRUE) THEN
    ResetPLCError_instance(Execute:=FALSE);
    OperatingStart:=FALSE;
END_IF;

// ResetPLCError指令执行
IF (Operating=TRUE) THEN
    ResetPLCError_instance(Execute:=TRUE);

    IF (ResetPLCError_instance.Done=TRUE) THEN
        // 正常结束时处理
        Operating:=FALSE;
    END_IF;

    IF (ResetPLCError_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating:=FALSE;
    END_IF;
END_IF;
```

GetPLCError

获取PLC功能模块中发生的控制器异常的最重要状态(部分停止故障、轻度故障)和最重要的事件代码。

指令	名称	FB/FUN	图形表现	ST表现
GetPLCError	PLC异常状态获取	FUN		Out:=GetPLCError(Level, Code);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生控制器异常 FALSE：未发生控制器异常	遵从数据类型	–	–
Level	最重要状态		PLC功能模块中发生的控制器异常中最重要的状态 0：无控制器异常 2：部分停止故障电平 3：轻度故障电平	0，2，3		
Code	最重要事件代码		PLC功能模块中发生的控制器异常中最重要的事件代码 16#0000_0000：无控制器异常 16#0007_0000～16#FFFF_FFFF：事件代码	16#00000000 16#00070000～16#FFFFFFFF		

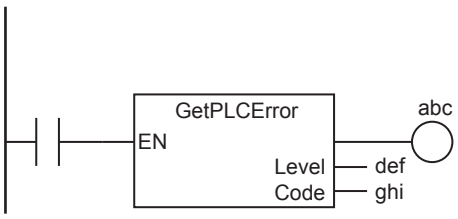
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
Level							○													
Code				○																

功能

获取PLC功能模块中发生的控制器异常的最重要的状态 “Level”和最重要的事件代码 “Code”。
未发生控制器异常时，有无异常 “Out”的值为FALSE。
发生了多个最重要的事件重要程度的控制器异常时，“Code”的值为最早发生的控制器异常的事件代码。

示例如下所示。

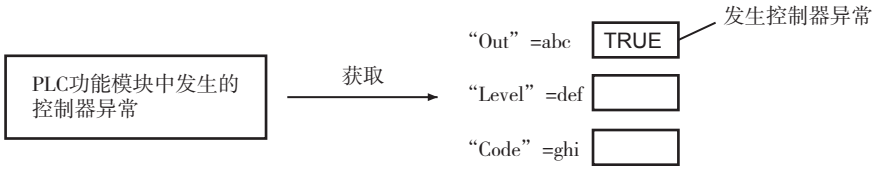
LD



ST

```
abc:=GetPLCError(def, ghi);
```

获取PLC功能模块中发生的控制器异常的最重要的状态“Level”和最重要的事件代码“Code”。



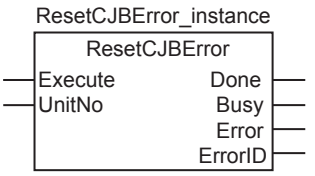
相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_ErrSta	PLC功能模块异常状态	WORD	PLC功能模块中检测到的异常的状态。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA–359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA–436)”。

ResetCJBError

解除I/O总线中发生的控制器异常。

指令	名称	FB/FUN	图形表现	ST表现
ResetCJBError	I/O总线异常解除	FB		ResetCJBError_instance(Execute, UnitNo, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitNo	单元编号	输入	解除异常的单元编号	_CBU_No00 ~ _CBU_No15, _SIO_No00 ~ _SIO_No95 _UNIT_ALL	–	_UNIT_ALL

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitNo	枚举体_eUnitNo　枚举元素参阅功能说明																			

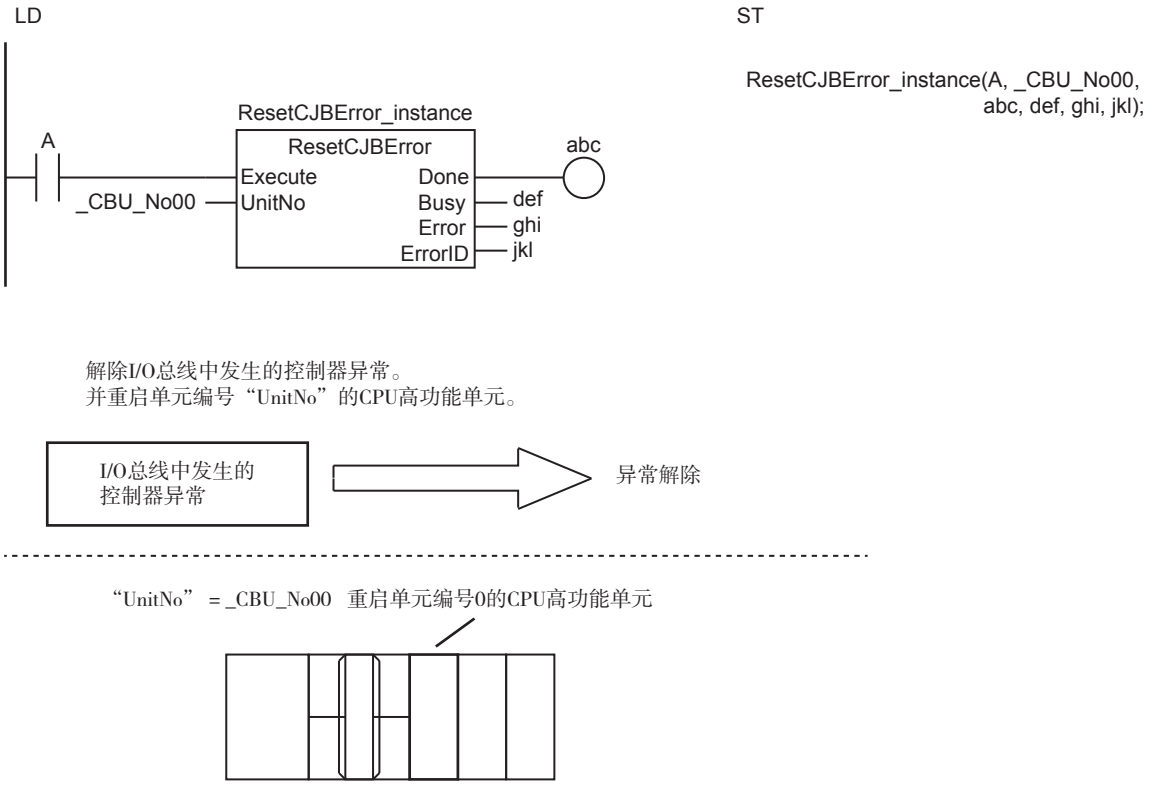
功能

解除I/O总线中发生的控制器异常。
单元编号 “UnitNo”指定的单元为CJ系列高功能单元时，同时重启该单元。

“UnitNo”的数据类型为枚举体_eUnitNo。枚举元素的含义如下所示。

枚举元素	含义
_CBU_No00 ~ _CBU_No15	CPU高功能单元的单元编号00 ~ 15
_SIO_No00 ~ _SIO_No95	高功能I/O单元的单元No.00 ~ 95
_UNIT_ALL	所有单元

“UnitNo”=_CBU_No00时的示例如下所示。在解除I/O总线中发生的控制器异常的同时，重启单元号0的CPU高功能单元。



相关的系统定义变量

变量名称	名称	数据类型	内容
_CJB_ErrSta	I/O总线异常状态	WORD	I/O总线中检测到的异常的状态。*1

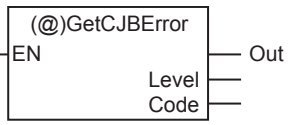
*1 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”。

使用注意事项

- 执行本指令之后，并不一定能够解除异常。请通过GetCJBError指令确认异常是否解除。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “UnitNo”的值超过有效范围时。
  - “UnitNo”中指定的单元不存在时。

GetCJBError

获取NJ系列CPU单元的I/O总线中发生的控制器异常的最重要状态(部分停止故障、轻度故障)和最重要的事件代码。

指令	名称	FB/FUN	图形表现	ST表现
GetCJBError	I/O总线异常状态获取	FUN		Out:=GetCJBError(Level, Code);

使用注意事项

本指令无法在NX系列CPU单元中使用。

变量

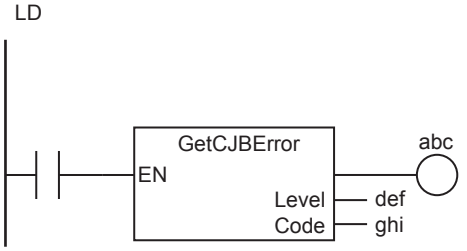
	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生控制器异常 FALSE：未发生控制器异常	遵从数据类型	–	–
Level	最重要状态		I/O总线中发生的控制器异常中最重要的状态 0：无控制器异常 2：部分停止故障电平 3：轻度故障电平	0，2，3		
Code	最重要事件代码		I/O总线中发生的控制器异常中最重要的事件代码 16#0000_0000：无控制器异常 16#0007_0000～16#FFFF_FFFF：事件代码	16#00000000 16#00070000～16#FFFFFFFF		

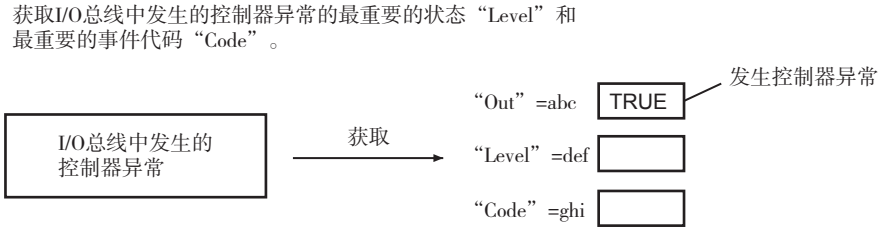
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
Level							○													
Code				○																

功能

获取I/O总线中发生的控制器异常的最重要的状态 “Level”和最重要的事件代码 “Code”。
未发生控制器异常时，有无异常 “Out”的值为FALSE。
发生了多个最重要的事件重要程度的控制器异常时，“Code”的值为最早发生的控制器异常的事件代码。

示例如下所示。





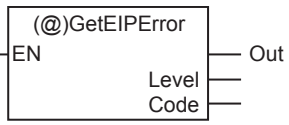
相关的系统定义变量

变量名称	名称	数据类型	内容
_CJB_ErrSta	I/O总线异常状态	WORD	I/O总线中检测到的异常的状态。*1

*1 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”。

GetEIPError

获取EtherNet/IP功能模块中发生的控制器异常的最重要状态(部分停止故障、轻度故障)和最重要的事件代码。

指令	名称	FB/FUN	图形表现	ST表现
GetEIPError	EtherNet/IP异常状态获取	FUN		Out:=GetEIPError(Level, Code);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生控制器异常 FALSE：未发生控制器异常	遵从数据类型	–	–
Level	最重要状态		EtherNet/IP功能模块中发生的控制器异常中最重要的状态 0：无控制器异常 2：部分停止故障电平 3：轻度故障电平	0，2，3		
Code	最重要事件代码		EtherNet/IP功能模块中发生的控制器异常中最重要的事件代码 16#0000_0000：无控制器异常 16#0007_0000 ~ 16#FFFF_FFFF：事件代码	16#00000000 16#00070000 ~ 16#FFFFFFFF		

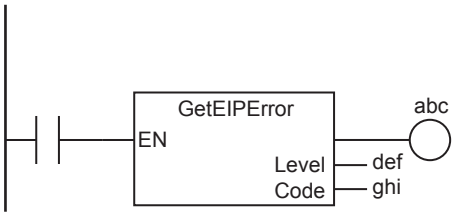
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
Level							○													
Code				○																

功能

获取EtherNet/IP功能模块中发生的控制器异常的最重要的状态 “Level”和最重要的事件代码 “Code”。
未发生控制器异常时，有无异常 “Out”的值为FALSE。
发生了多个最重要的事件重要程度的控制器异常时，“Code”的值为最早发生的控制器异常的事件代码。

示例如下所示。

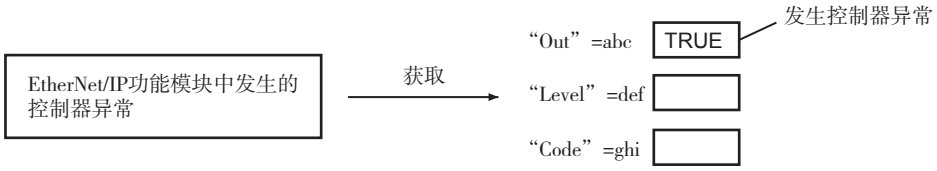
LD



ST

```
abc:=GetEIPError(def, ghi);
```

获取EtherNet/IP功能模块中发生的控制器异常的最重要的状态“Level”和最重要的事件代码“Code”。



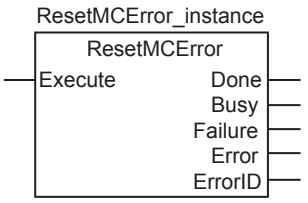
相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_ErrSta	EtherNet/IP功能模块异常状态	WORD	EtherNet/IP功能模块中检测到的异常的状态。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

ResetMCError

解除运动控制功能模块中发生的控制器异常。

指令	名称	FB/FUN	图形表现	ST表现
ResetMCError	运动控制异常解除	FB		ResetMCError_instance(Execute, Done, Busy, Failure Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Failure	非法结束	输出	TRUE：异常解除失败 FALSE：异常解除成功	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Failure	○																			

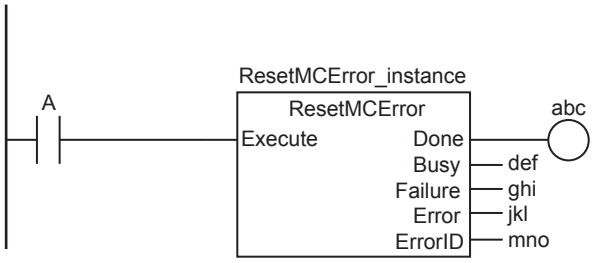
功能

解除运动控制功能模块中发生的控制器异常。
解除失败时，“Failure”的值变为TRUE。

无论在哪个任务中记述执行ResetMCError指令的程序，均以所有轴、所有轴组为对象。

示例如下所示。

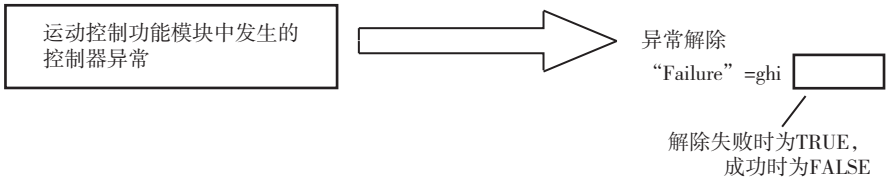
LD



ST

```
ResetMCError_instance(A, abc, def,
                      ghi, jkl, mno);
```

解除运动控制功能模块中发生的控制器异常。
解除失败时，“Failure”的值变为TRUE。



相关的系统定义变量

变量名称	名称	数据类型	内容
_MC_ErrSta	运动控制异常状态	WORD	运动控制功能模块中检测到的异常的状态。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

使用注意事项

- 执行本指令之后，并不一定能够解除异常。请通过GetMCError指令确认异常是否解除。
- 如果在MC试运行中尝试执行本指令，由于 “Busy”的值保持TRUE而无法执行。
- 对欧姆龙制伺服驱动器G5系列使用本指令时，请进行指令的排他性控制，以免同时执行ResetECError指令。如果本指令与ResetECError指令同时执行，G5系列可能会无法接收之后的SDO。

版本相关信息

- 版本为从Ver.1.02到Ver.1.09的CPU 单元最多只可生成100个本指令的实例。
- 通过版本为从Ver.1.02到Ver.1.09的CPU 单元将生成了101个以上本指令的实例的用户程序传送到控制器时，会发生控制器异常。根据用户程序的传送方法，控制器异常会有如下不同。

用户程序的传送方法	控制器异常的事件代码	控制器异常的重要程度
使用 “同步”功能传送项目	10250000Hex	全部停止故障电平
	571D0000Hex	监视信息
在线编辑的传送	571D0000Hex	监视信息

- 通过版本为Ver.1.01以下的CPU 单元将生成了101个以上本指令的实例的用户程序传送到控制器时，不会发生上述控制器异常。但是，如果生成了过多的本指令的实例，会增大用户程序的规模，可能发生全部停止故障电平的控制器异常。

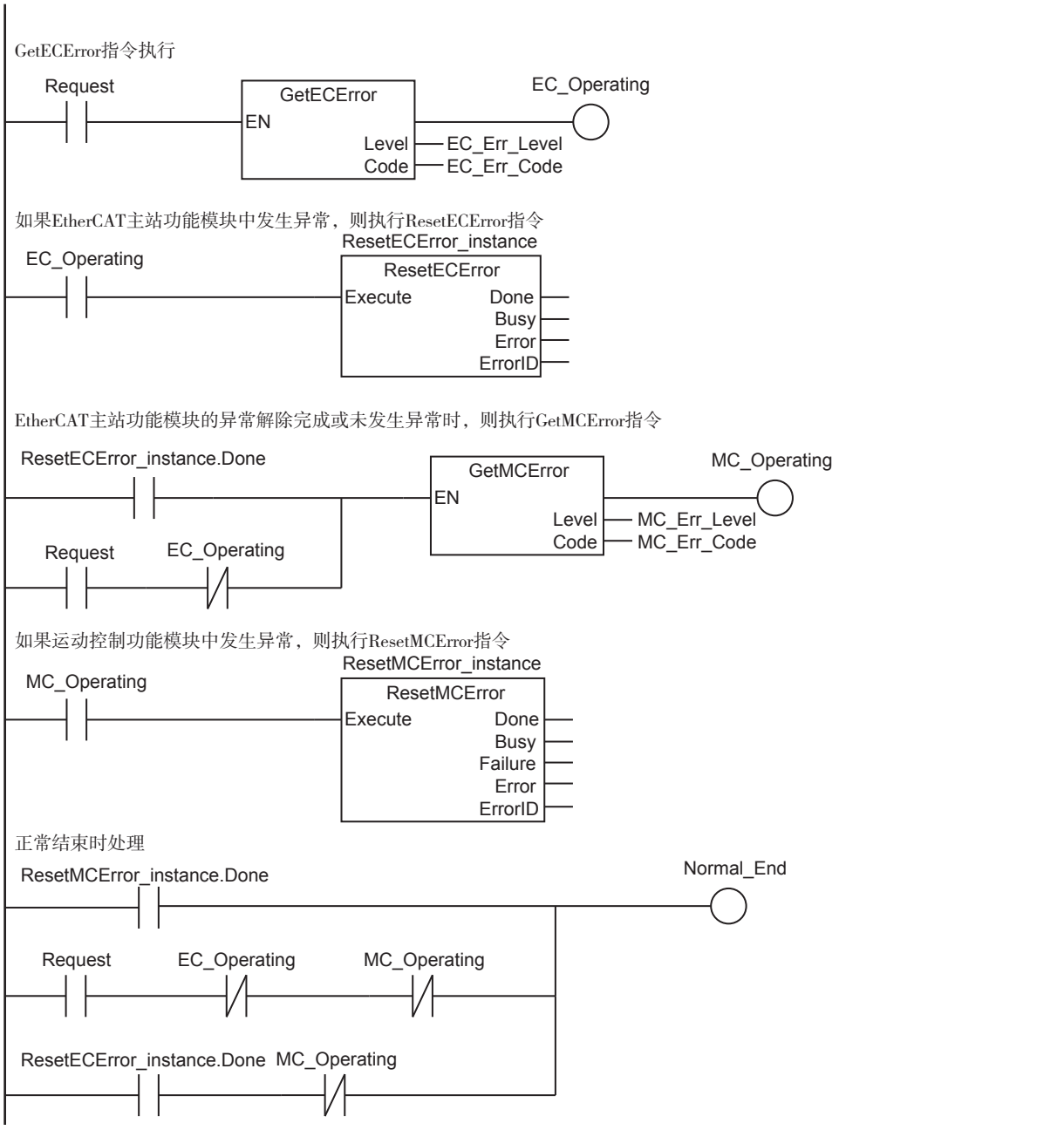
示例程序

检测EtherCAT主站功能模块和运动控制功能模块中发生的控制器异常。如果发生异常，则将其解除。处理步骤如下所示。

1. 通过GetECError指令检测EtherCAT主站功能模块中发生的控制器异常。
2. 如果发生异常，则通过ResetECError指令解除异常。
3. 通过GetMCError指令检测运动控制功能模块中发生的控制器异常。
4. 如果发生异常，则通过ResetMCError指令解除异常。

LD

名称	数据类型	初始值	注释
Request	BOOL	FALSE	异常检测解除要求
EC_Err_Level	UINT	0	EtherCAT主站功能模块 最重要事件重要程度
EC_Err_Code	DWORD	DWORD#16#0	EtherCAT主站功能模块 最重要事件代码
EC_Operating	BOOL	FALSE	EtherCAT主站功能模块异常解除中
MC_Err_Level	UINT	0	运动控制功能模块 最重要事件重要程度
MC_Err_Code	DWORD	DWORD#16#0	运动控制功能模块 最重要事件代码
MC_Operating	BOOL	FALSE	运动控制功能模块异常解除中
Normal_End	BOOL	FALSE	正常结束
ResetECError_instance	ResetECError		
ResetMCError_instance	ResetMCError		



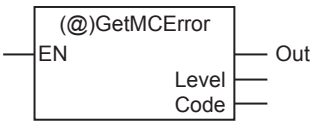
ST

名称	数据类型	初始值	注释
Request	BOOL	FALSE	异常检测解除要求
EC_Error	BOOL	FALSE	有EtherCAT主站功能模块异常
EC_Err_Level	UINT	0	EtherCAT主站功能模块最重要事件重要程度
EC_Err_Code	DWORD	DWORD #16#0	EtherCAT主站功能模块最重要事件代码
EC_Stage	INT	0	EtherCAT主站功能模块异常解除状态
MC_Error	BOOL	FALSE	有运动控制功能模块异常
MC_Err_Level	UINT	0	运动控制功能模块最重要事件重要程度
MC_Err_Code	DWORD	DWORD #16#0	运动控制功能模块最重要事件代码
MC_Stage	INT	0	运动控制功能模块异常解除状态
ResetECError_instance	ResetECError		
ResetMCError_instance	ResetMCError		

```
IF (Request=TRUE) THEN                                    // 判断异常解除要求
    EC_Error:=GetECError(EC_Err_Level, EC_Err_Code);    // EtherCAT功能模块的控制器异常检测
    MC_Error:=GetMCError(MC_Err_Level, MC_Err_Code); // 运动控制功能模块的控制器异常检测

    IF (EC_Error=TRUE) THEN          // 有EtherCAT功能模块的控制器异常
        CASE EC_Stage OF
        0 :                 // 初始化
            ResetECError_instance(Execute:=FALSE);
            EC_Stage:=INT#1;
        1 :                 // EtherCAT功能模块的控制器异常解除中
            ResetECError_instance(Execute:=TRUE);
            IF (ResetECError_instance.Done=TRUE) THEN
                EC_Stage:=INT#99;           // 正常结束
            END_IF;
            IF (ResetECError_instance.Error=TRUE) THEN
                EC_Stage:=INT#98;           // 异常结束
            END_IF;
        99 :                // 正常结束时处理
            EC_Stage:=INT#0;
        98 :                // 异常结束时处理
            EC_Stage:=INT#0;
        END_CASE;
    END_IF;

    IF (MC_Error=TRUE) THEN      // 有运动控制功能模块的控制器异常
        CASE MC_Stage OF
        0 :                 // 初始化
            ResetMCError_instance(Execute:=FALSE);
            MC_Stage:=INT#1;
        1 :                 // 运动控制功能模块的控制器异常解除中
            IF (EC_Error=FALSE) THEN
                ResetMCError_instance(Execute:=TRUE);           // 全部从站复位
                IF (ResetMCError_instance.Done=TRUE) THEN
                    MC_Stage:=INT#99;       // 正常结束
                END_IF;
                IF ( (ResetMCError_instance.Error=TRUE) OR (ResetMCError_instance.Failure=TRUE) ) THEN
                    MC_Stage:=INT#98;       // 异常结束
                END_IF;
            END_IF;
        99 :                // 正常结束时处理
            MC_Stage:=INT#0;
        98 :                // 异常结束时处理
            MC_Stage:=INT#0;
        END_CASE;
    END_IF;
END_IF;
```

GetMCError

获取运动控制功能模块中发生的控制器异常的最重要状态 (部分停止故障、轻度故障) 和最重要的事件代码。

指令	名称	FB/FUN	图形表现	ST表现
GetMCError	运动控制异常状态获取	FUN		Out:=GetMCError(Level, Code);

变量

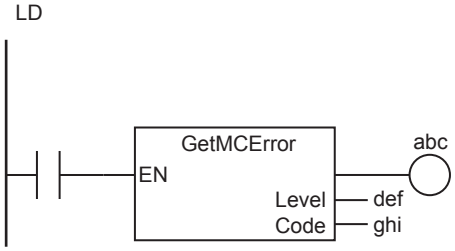
	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生控制器异常 FALSE：未发生控制器异常	遵从数据类型	–	–
Level	最重要状态		运动控制功能模块中发生的控制器异常中最重要的状态 0：无控制器异常 2：部分停止故障电平 3：轻度故障电平	0，2，3		
Code	最重要事件代码		运动控制功能模块中发生的控制器异常中最重要的事件代码 16#0000_0000：无控制器异常 16#0007_0000 ~ 16#FFFF_FFFF：事件代码	16#00000000 16#00070000 ~ 16#FFFFFFFF		

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
Level							○													
Code				○																

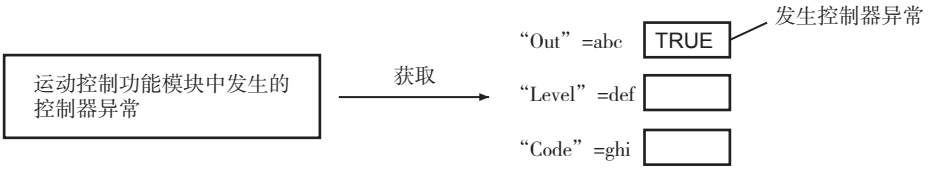
功能

获取运动控制功能模块中发生的控制器异常的最重要的状态 “Level”和最重要的事件代码 “Code”。
未发生控制器异常时，有无异常 “Out”的值为FALSE。
发生了多个最重要的事件重要程度的控制器异常时，“Code”的值为最早发生的控制器异常的事件代码。

示例如下所示。



获取运动控制功能模块中发生的控制器异常的最重要的状态“Level”和最重要的事件代码“Code”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_MC_ErrSta	运动控制异常状态	WORD	运动控制功能模块中检测到的异常的状态。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

示例程序

请参阅 “ResetMCError指令(P.2-813)”的示例程序。

ResetECError

解除EtherCAT功能模块中发生的控制器异常。

指令	名称	FB/FUN	图形表现	ST表现
ResetECError	EtherCAT异常解除	FB	ResetECError_instance ResetECError Execute / Done Busy Error ErrorID	ResetECError_instance(Execute, Done, Busy, Error, ErrorID);

变量

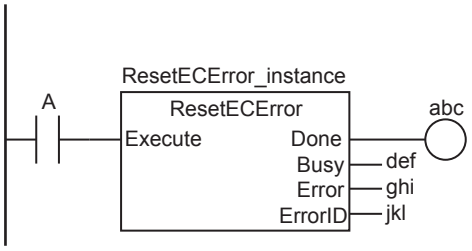
仅共通的变量。

功能

解除EtherCAT功能模块中发生的控制器异常。

示例如下所示。

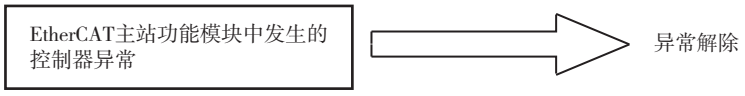
LD



ST

```
ResetECError_instance(A, abc, def, ghi, jkl);
```

解除EtherCAT功能模块中发生的控制器异常。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_ErrSta	内置EtherCAT异常	WORD	EtherCAT主站功能模块的异常的集合。*1

*1 数据跟踪功能的详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)” 或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

使用注意事项

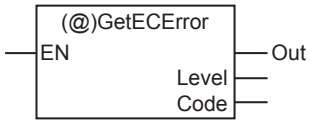
- 执行本指令之后，并不一定能够解除异常。请通过GetECError指令确认异常是否解除。
- 对欧姆龙制伺服驱动器G5系列使用本指令时，请进行指令的排他性控制，以免同时执行ResetMCError指令、MC_Reset指令、MC_GroupReset指令中任意一项指令。如果本指令与上述3项指令中的任意一项同时执行，G5系列可能会无法接收之后的SDO。
- EC_DisconnectSlave 指令、EC_ConnectSlave 指令、EC_ChangeEnableSetting 指令、ResetECError 指令、RestartNXUnit指令、NX_ChangeWriteMode指令正在执行时，无法执行本指令。
- 以下情况时会发生异常。“Error”变为TRUE。
  - EtherCAT主站功能模块中发生的控制器异常解除处理中，还执行了本指令时。
  - EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令中的任一指令正在执行时。

示例程序

请参阅 “ResetMCError指令(P.2–813)”的示例程序。

GetECError

检测EtherCAT主站功能模块中发生的通信端口异常、主站异常、从站异常。

指令	名称	FB/FUN	图形表现	ST表现
GetECError	EtherCAT异常状态获取	FUN		Out:=GetECError(Level, Code);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生通信端口异常、主站异常、从站异常*1 FALSE：未发生通信端口异常、主站异常、从站异常	遵从数据类型	–	–
Level	最重要状态		发生的通信端口异常、主站异常、从站异常中最重要的状态*1 0：无异常 2：部分停止故障电平 3：轻度故障电平	0，2，3		
Code	最重要事件代码		发生的通信端口异常、主站异常中最重要的事件代码*1	16#00000000 16#00070000 ~ 16#FFFFFFFF		

*1 可检测的异常种类因CPU单元和Sysmac Studio的版本而异。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
Level							○													
Code				○																

功能

检测EtherCAT主站功能模块中发生的通信端口异常、主站异常、从站异常。
“Out”的值在有异常时为TRUE，无异常时为FALSE。
“Level”在发生通信端口异常、主站异常、从站异常时显示最重要的状态。
“Code”在发生通信端口异常、主站异常时显示最重要的事件代码。

检测的异常种类和输出变量的值

本指令可检测的异常种类因CPU单元的单元版本而异。各版本检测的异常种类如下所述。

CPU单元的单元版本	检测的异常种类
Ver.1.02以上	通信端口异常、主站异常、从站异常
Ver.1.01以下	通信端口异常、主站异常

CPU单元的单元版本、EtherCAT主站功能模块的状态及各输出变量值之间的关系如下所述。

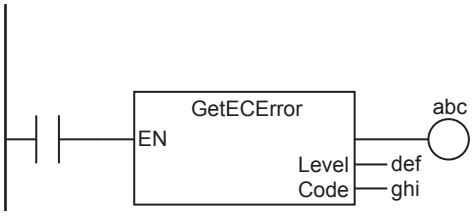
CPU单元的单元版本	EtherCAT主站功能模块的状态	“Out”的值	“Level”的值	“Code”的值
Ver.1.04以上	无异常	FALSE	0	16#0000_0000
	有通信端口异常或主站异常	TRUE	发生的异常中最重要的状态 2：部分停止故障等级 3：轻度故障等级	发生的异常中最重要的事件代码*1 16#0007_0000～16#FFFF_FFFF
	有从站异常			16#0000_0000
Ver.1.02、Ver.1.03	无异常	FALSE	0	16#0000_0000
	有通信端口异常或主站异常	TRUE	发生的异常中最重要的状态 2：部分停止故障等级 3：轻度故障等级	发生的异常中最重要的事件代码*1 16#0007_0000～16#FFFF_FFFF
	有从站异常	FALSE		16#0000_0000
Ver.1.00、Ver.1.01	无异常	FALSE	0	16#0000_0000
	有通信端口异常或主站异常	TRUE	发生的异常中最重要的状态 2：部分停止故障等级 3：轻度故障等级	发生的异常中最重要的事件代码*1 16#0007_0000～16#FFFF_FFFF
	有从站异常	FALSE	0	16#0000_0000

*1 发生了多个最重要的事件重要程度的通信端口异常、主站异常时，为最早发生的异常的事件代码。

记述示例

示例如下所示。

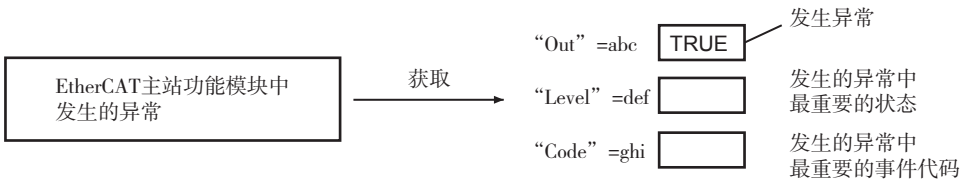
LD



ST

```
abc:=GetECError(def, ghi);
```

检测EtherCAT主站功能模块中发生的通信端口异常、主站异常、从站异常。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_ErrSta	内置EtherCAT异常	WORD	EtherCAT主站功能模块的异常的集合。(*2)
_EC_PortErr(*1)	通信端口异常	WORD	EtherCAT主站用通信端口的异常的集合。(*2)
_EC_MstrErr(*1)	主站异常	WORD	EtherCAT主站的异常与EtherCAT主站检测到的从站异常的集合。(*2)
_EC_SlavErr	从站异常	WORD	EtherCAT从站全体的异常状态的集合。(*2)

*1 本指令所获取的是属于_EC_PortErr(通信端口异常)与_EC_MstrErr(主站异常)的异常。

*2 详情请参阅 “NJ/NX系列 CPU单元内置EtherCAT端口 用户手册(SBCD-358)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

版本相关信息

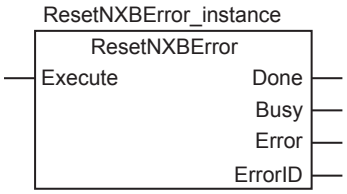
可通过本指令检测从站异常的是版本Ver.1.02以上的CPU单元。

示例程序

请参阅 “ResetMCError指令(P.2-813)”的示例程序。

ResetNXBError

解除NX总线功能模块中发生的控制器异常。

指令	名称	FB/FUN	图形表现	ST表现
ResetNXBError	NX总线异常解除	FB		ResetNXBError_instance(Execute, Done, Busy, Error, ErrorID);

使用注意事项

本指令仅NX1P2 CPU单元可使用。

变量

仅共通的变量

功能

解除NX总线功能模块中发生的控制器异常。

异常解除完成时，输出变量 “Busy”变为FALSE，输出变量 “Done”变为TRUE。
但无法解除安全控制单元的异常。

异常解除处理中，除异常解除处理中的指令外，在其它实例中另行执行本指令时，后执行的指令将发生错误。
后执行的指令的输出变量 “Error”将变为TRUE，输出变量 “ErrorID”将输出错误代码(指令多重启动：041A)。

相关的系统定义变量

变量名称	名称
_NXB_ErrSta	NX总线功能模块 异常状态

参考

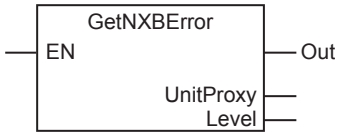
- 使用模拟器运行本指令的情况下，“Execute”从FALSE变为TRUE时，“Done”变为TRUE，“Busy”变为FALSE，“Error”变为FALSE，“ErrorID”变为 “0”。不执行异常解除。
- 解除异常后异常原因仍存在且持续异常状态时，看上去如同无法解除异常。
- 事件日志不会清除。

使用注意事项

事件任务中无法使用。编译时会发生错误。

GetNXBError

获取NX系列CPU单元的NX总线功能模块中发生的控制器异常的最重要的状态。

指令	名称	FB/FUN	图形表现	ST表现
GetNXBError	NX总线异常状态获取	FUN		Out:=GetNXBError(UnitProxy, Level);

使用注意事项

本指令仅NX1P2 CPU单元可使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	有无异常	输出	TRUE：发生异常 FALSE：未发生异常	遵从数据类型	–	–
UnitProxy	低位层NX单元		在发生的异常中，发生 “Level”异常的NX单元	–		
Level	最重要状态		发生的异常中最重要的状态 0：无异常 2：部分停止故障等级 3：轻度故障等级	0,2,3		

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
UnitProxy	结构体_sNXUNIT_ID 详情参阅功能说明																			
Level							○													

功能

获取NX总线功能模块中发生的控制器异常的最重要状态(部分停止故障、轻度故障)。
对象为与NX1P2 CPU单元及CPU单元上的NX总线连接的NX单元。

“Out”的值在有异常时为TRUE，无异常时为FALSE。
“Level”表示发生的异常中最重要的状态。
“UnitProxy”将返回在发生的异常中发生 “Level”异常的NX单元的 “UnitProxy”。

多个单元发生同一等级的异常时，将返回与主站最近、单元编号最小的 “UnitProxy”。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	低位层NX单元	在发生的异常中，发生 “Level”异常的NX单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	NX单元的单元编号	UDINT
Path	路径	至NX单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将通过变量表创建的_sNXUNIT_ID结构体的变量传输给 “UnitProxy”。

相关的系统定义变量

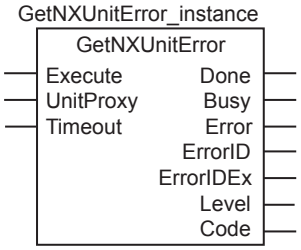
变量名称	名称
_NXB_ErrSta	NX总线功能模块 异常状态
_NXB_MstrErrSta	Nx总线功能模块 主站异常状态
_NXB_UnitErrSta[1] ~ [63]	Nx总线功能模块 单元异常状态

参考

- 使用模拟器运行本指令时，必定返回无错误，“Out”变为 FALSE，“Level”变为 “0”，“UnitProxy”变为不定值。

GetNXUnitError

获取NX系列CPU单元的NX总线功能模块或NX单元中发生的控制器异常的最重要的状态和最重要的事件代码。

指令	名称	FB/FUN	图形表现	ST表现
GetNXUnitError	NX单元异常状态获取	FB		GetNXUnitError_instance(Execute, UnitProxy, TimeOut, Done, Busy, Error, ErrorID, ErrorIDEx, Level, Code);

使用注意事项

本指令仅NX1P2 CPU单元可使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	指定单元	输入	对象NX单元的指定	–	–	*1
TimeOut	超时时间		超时时间 0时为2.0s	0 ~ 60000	ms	2000 (2.0s)
Level	最重要状态	输出	发生的异常中最重要的状态 0：无异常 2：部分停止故障等级 3：轻度故障等级	0,2,3	–	–
Code	最重要事件代码		发生的异常中最重要的事件代码	16#00000000 16#00070000 ~ 16#FFFFFFFF		

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID 详情参阅功能说明																			
TimeOut							○													
Level							○													
Code				○																

功能

输出NX总线功能模块和NX单元中发生的控制器异常的最重要状态(部分停止故障、轻度故障)、最重要的事件代码和低位层的NX单元。
对象为与NX1P2 CPU单元及CPU单元上的NX总线连接的NX单元。

通过 “UnitProxy”指定要获取的单元。
“Done”的值变为TRUE时，获取完成。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	指定单元	要指定的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至要指定单元的设备变量传输至 “UnitProxy”。

超时时间由 “TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。

“Level”表示发生的异常中最重要的状态。
“Code”表示发生的异常中最重要的事件代码。多个单元发生同一等级的异常时，表示最早的事件代码。
此外，无异常时为16#00000000。

指定的NX单元和各变量的值

输出至输入输出变量及输出变量的内容因指定的NX单元而异。
NX单元与各变量值的关系如下所述。

指定的单元	NX1P2 CPU单元	CPU单元上的NX单元
“Level”的值	CPU单元的最重要状态	NX单元的最重要状态
“Code”的值	存在多个异常时：最重要的事件代码 存在多个同一 “Level”的异常时：最早的事件代码 无异常时：16#0000_0000	

与GetNXBError指令组合使用时

用户程序通常使用GetNXBError(NX总线异常状态获取)指令监视NX总线有无错误。

GetNXBError指令的输出变量 “Level”为0以外时，GetNXBError指令的 “UnitProxy”中将保存发生了重要程度最高异常的NX单元的值。

为了获取发生异常的NX单元的 “Level”和 “Code”，执行GetNXUnitError指令。

相关的系统定义变量

变量名称	名称
_NXB_UnitErrFlagTbl	NX单元异常状态

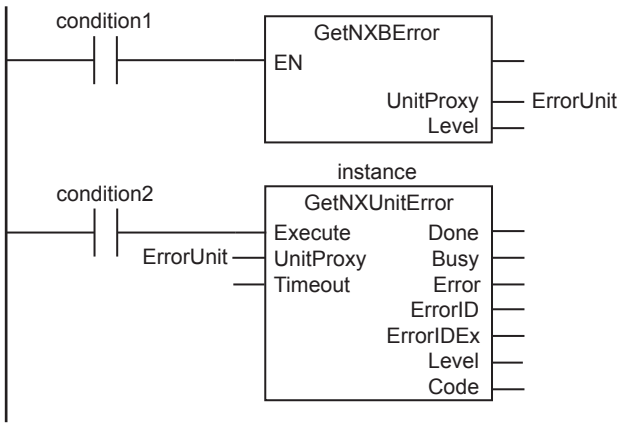
参考

传输至 “UnitProxy”的参数

传输至输入变量 “UnitProxy”的参数如下所述。

● 只使用用户程序传输 “UnitProxy”时

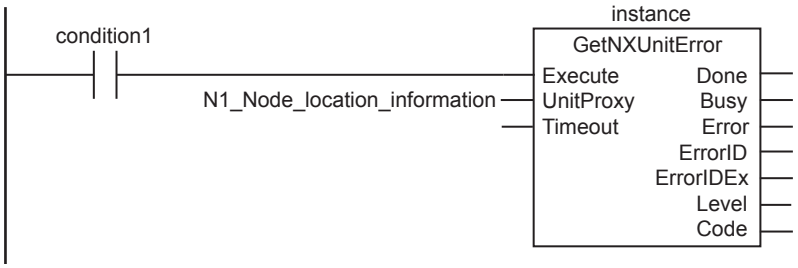
将通过对NX总线功能模块发送的GetNXBError指令而获取的 “UnitProxy”的值 “ErrorUnit”传输至GetNXUnitError指令的 “UnitProxy”。



● 使用设备变量传输 “UnitProxy”时

生成指定NX总线上单元的设备变量，传输至本指令的 “UnitProxy”。

外部变量	名称	数据类型	常数	注释
	N1_Node_location_information	_sNXUNIT_ID	☑	



模拟器的动作

使用模拟器运行本指令的情况下，“Execute”从FALSE变为TRUE时，如下所述。

输出变量	内容
Done	TRUE
Busy	FALSE
Error	FALSE
ErrorID	0
ErrorIDEx	0
Level	0
Code	0

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- 无法同时执行多个本指令。本指令只能同时执行1个。
- 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0400	16#00000000	· “UnitProxy”的值超过有效范围。 · “TimeOut”的值超过有效范围。
16#041A	16#00000000	执行本指令的过程中，再次执行了本指令。
16#2C00	16#00000401	指定的单元非本指令的对象。
	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。
	16#00001101	指定的单元不正确。 请确认单元。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。
	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。
	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000 ~ 0x6FFF或0x7000 ~ 0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。
	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ~ 16#8FFF0000 16#90010000 ~ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。
	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。
	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C02	16#00000000	通信过程中发生了超时。

示例程序

NX总线发生异常时，将发生的单元、等级及代码的值传输至触摸屏显示用变量。

采用以下系统构成。
在NX1P2 CPU单元上连接3个NX单元。

设备变量

NX单元	设备变量
第1个NX单元	N1_Node_location_information
第2个NX单元	N2_Node_location_information
第3个NX单元	N3_Node_location_information

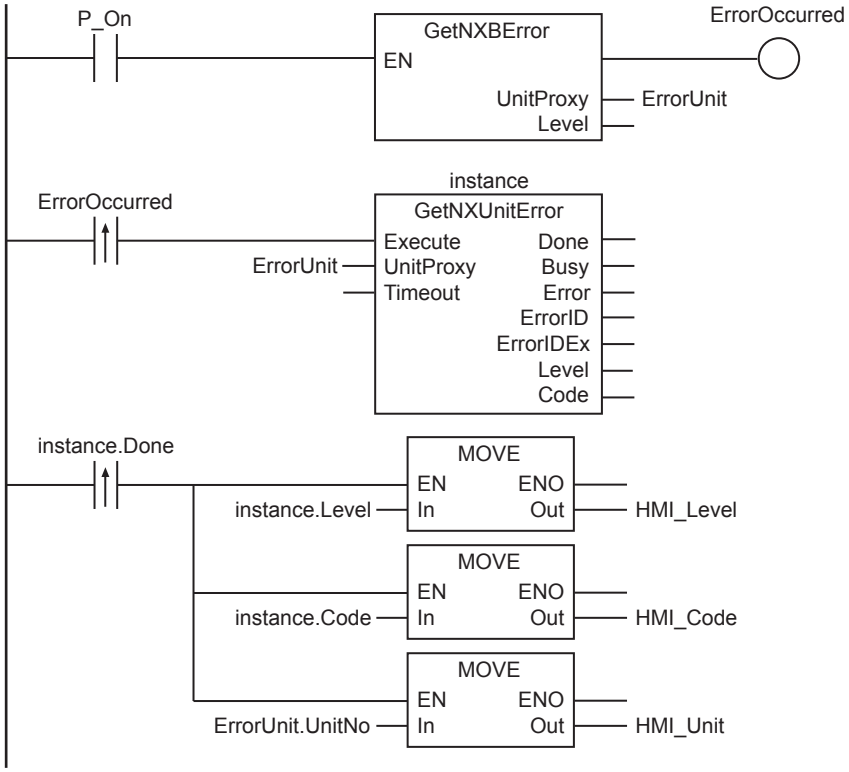
全局变量的定义

全局变量

名称	数据类型	常数	注释
N1_Node_location_information	_sNXUNIT_ID	☑	
N2_Node_location_information	_sNXUNIT_ID	☑	
N3_Node_location_information	_sNXUNIT_ID	☑	
HMI_Level*1	UINT	☐	
HMI_Code*1	DWORD	☐	
HMI_Unit*1	UDINT	☐	

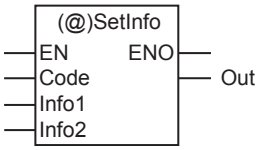
*1 以 “HMI_”开头的名称为触摸屏显示用的变量。

内部变量	名称	数据类型	常数	注释
	instance	GetNXUnitError	⌐¬	
	ErrorOccurred	BOOL	⌐¬	
	ErrorUnit	_sNXUNIT_ID	⌐¬	



SetInfo

生成用户信息。

指令	名称	FB/FUN	图形表现	ST表现
SetInfo	用户信息生成	FUN		SetInfo(Code, Info1, Info2);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Code	事件代码	输入	生成的用户信息的事件代码	40001 ~ 60000	–	40001
Info1	附属信息1		在生成用户信息的同时，事件日志中记录的值	遵从数据类型		(*)
Info2	附属信息2					
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

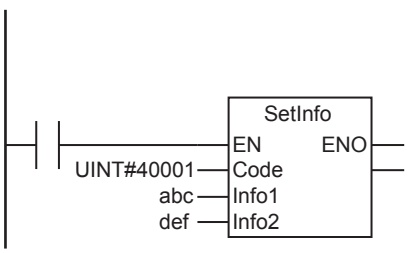
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Code							○													
Info1	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	
Info2	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	
Out	○																			

功能

生成事件代码 “Code”中指定的用户信息。
另外，在与事件代码的重要程度相对应的用户事件日志区域记录发生时刻、事件代码 “Code” 、事件重要程度、附属信息 “Info1”、“Info2”的值。

示例如下所示。生成与事件代码40001对应的用户信息。作为附属信息，记录变量abc、def的值。

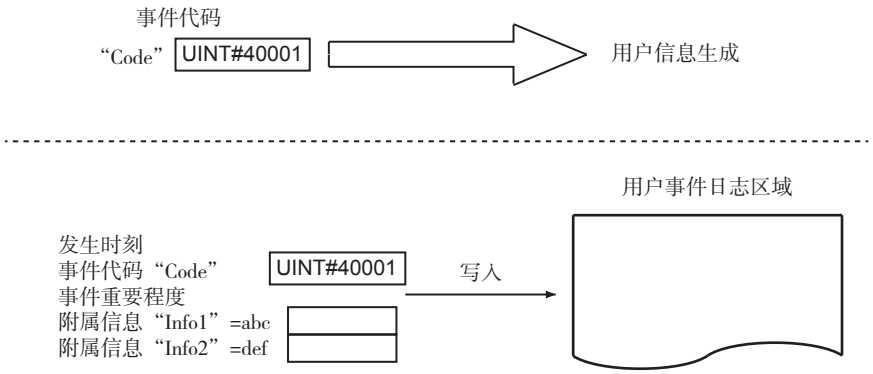
LD



ST

```
SetInfo(UINT#40001, abc, def);
```

生成事件代码“Code”中指定的用户信息。
另外，在与事件代码的重要程度相对应的用户事件日志区域记录发生时刻、事件代码“Code”、事件重要程度、附属信息“Info1”、“Info2”的值。

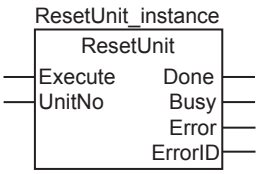


使用注意事项

- 请务必将传输至“Info1”、“Info2”的输入参数指定为变量。即使不使用附属信息，也请指定虚拟变量。如果指定为常量，编连时会发生异常。
- 在ST程序中使用本指令时，不使用返回值“Out”。
- 以下情况时会发生异常。ENO为FALSE。
  - “Code”的值超过有效范围时。

ResetUnit

重启CPU高功能单元或高功能I/O单元。

指令	名称	FB/FUN	图形表现	ST表现
ResetUnit	单元重启	FB		ResetUnit_instance(Execute, UnitNo, Done, Busy, Error, ErrorID);

使用注意事项

本指令无法在NX系列CPU单元中使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitNo	单元编号	输入	重启的单元编号	_CBU_No00 ~ _CBU_No15, _SIO_No00 ~ _SIO_No95	–	_CBU_No00

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitNo	枚举体_eUnitNo 枚举元素参阅功能说明																			

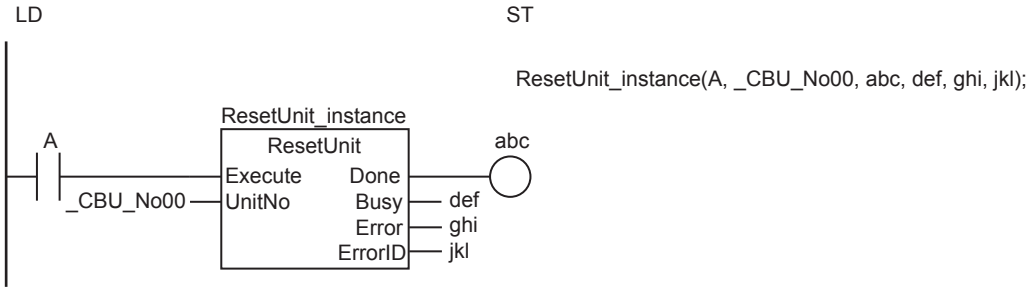
功能

重启单元编号 “UnitNo”的CPU高功能单元或高功能I/O单元。

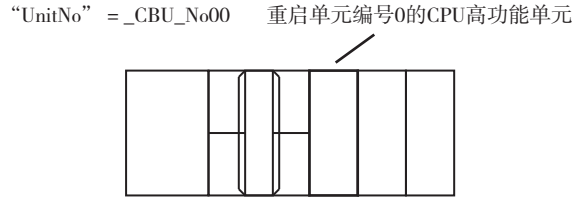
“UnitNo”的数据类型为枚举体_eUnitNo。枚举元素的含义如下所示。

枚举元素	含义
_CBU_No00 ~ _CBUNo15	CPU高功能单元的单元编号00 ~ 15
_SIO_No00 ~ _SIO_No95	高功能I/O单元的单元No.00 ~ 95

“UnitNo”=_CBU_No00时的示例如下所示。重启单元编号0的CPU高功能单元。



重启单元编号“UnitNo”的CPU高功能单元或高功能I/O单元。



使用注意事项

- “UnitNo”中指定的单元在重启中时，本指令不会出现异常。“Busy”的值变为TRUE，重启结束时“Done”的值变为TRUE。重启的请求不进行排队处理。
- 运行开始时 “Execute”的值为TRUE的情况下，重启单元。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “UnitNo”的值超过有效范围时。
  - “UnitNo”中指定的单元不存在时。
  - 重启失败时。

示例程序

如果Trigger的值由FALSE变为TRUE，则在将单元编号0的串行通信单元的串行端口1的传送速度设定为38400bps的基础上，重启单元。

全局变量的定义

全局变量

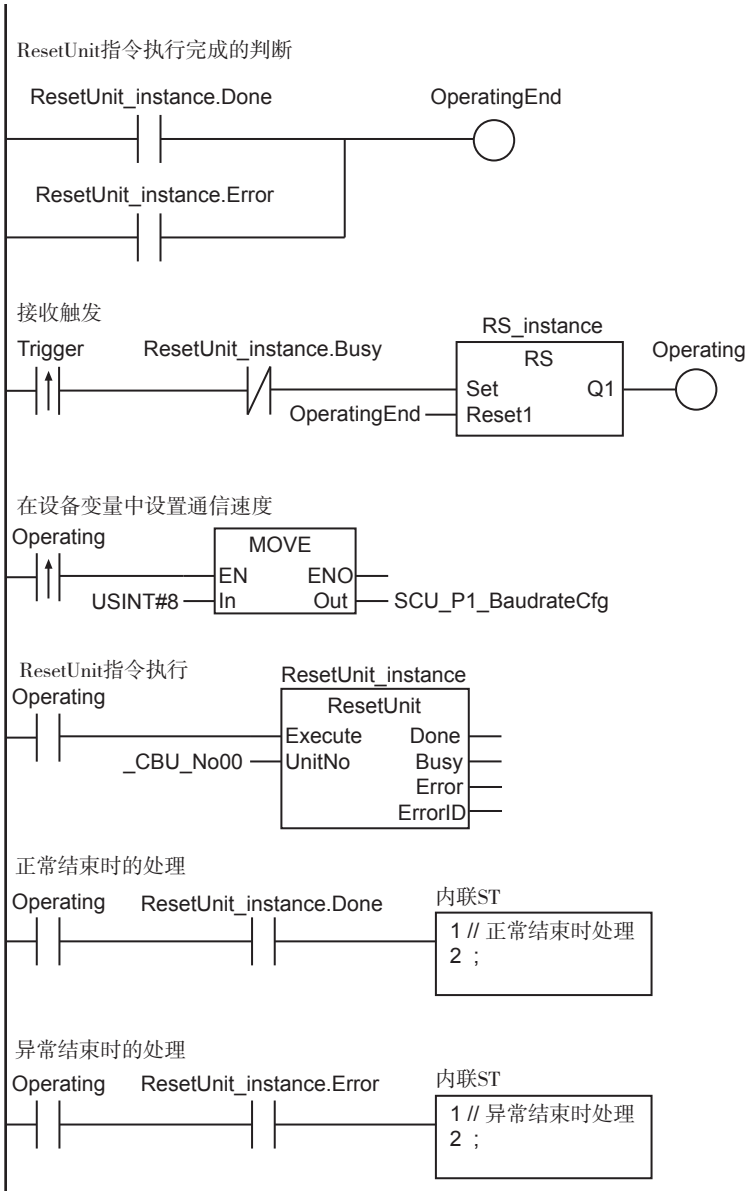
名称	数据类型	初始值	分配对象*1	保持	注释
SCU_P1_BaudrateCfg	USINT	0	IOBus://rack#0/slot#0/P1_BaudrateCfg	☑	传送速度

*1 是将串行通信单元安装于机架编号0的插槽编号0上时的分配对象。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	RS_instance	RS		
	ResetUnit_instance	ResetUnit		

外部变量	名称	数据类型	注释
	SCU_P1_BaudrateCfg	USINT	传送速度



ST

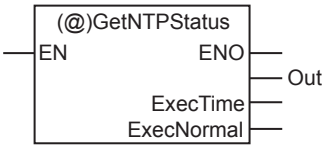
内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	ResetUnit_instance	ResetUnit		

外部变量	名称	数据类型	注释
	SCU_P1_BaudrateCfg	USINT	传送速度

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (ResetUnit_instance.Busy=FALSE) ) THEN
    OperatingStart:=TRUE;
    Operating:=TRUE;
END_IF;
LastTrigger:=Trigger;

// ResetUnit_instance初始化和在设备变量中设置传送速度
IF (OperatingStart=TRUE) THEN
    ResetUnit_instance(Execute:=FALSE);
    SCU_P1_BaudrateCfg:=USINT#8;
    OperatingStart:=FALSE;
END_IF;

// ResetUnit指令执行
IF (Operating=TRUE) THEN
    ResetUnit_instance(
        Execute :=TRUE,          // 启动条件
        UnitNo  :=_CBU_No00);    // 单元编号

    IF (ResetUnit_instance.Done=TRUE) THEN
        // 正常结束时处理
        Operating:=FALSE;
    END_IF;

    IF (ResetUnit_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating:=FALSE;
    END_IF;
END_IF;
```

GetNTPStatus

读取NTP的状态。

指令	名称	FB/FUN	图形表现	ST表现
GetNTPStatus	NTP状态读取	FUN		GetNTPStatus(ExecTime, ExecNormal);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
ExecTime	NTP最终正常动作时刻		NTP最终正常动作时刻	遵从数据类型	年月日时分秒	
ExecNormal	NTP正常结束标志		TRUE：正常结束 FALSE：异常结束		–	

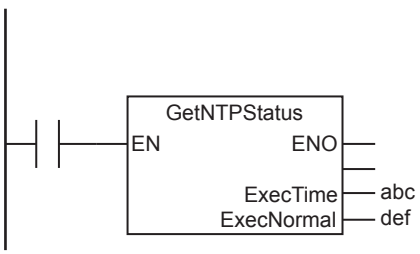
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			
ExecTime																			○	
Exec Normal	○																			

功能

读取NTP的状态。读取信息为NTP最终正常动作时刻 “ExecTime”和正常结束标志 “ExecNormal”。

示例如下所示。

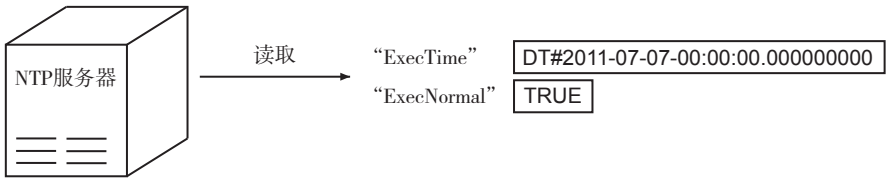
LD



ST

```
GetNTPStatus(abc, def);
```

读取NTP的状态。
NTP最终正常动作时刻为2011年7月7日0时0分0秒时，“ExecTime”与“ExecNormal”的值变为如下。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_NTPResult	NTP状态	*1	NTP状态。*2

*1 _sNTP_RESULT

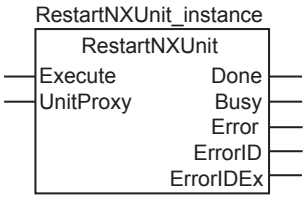
*2 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)”。

使用注意事项

- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 通过本指令读取的信息为系统定义变量 “_EIP_NTPResult”的内容。该变量无法直接访问。要读取该变量的内容，必须使用本指令。

RestartNXUnit

重启EtherCAT耦合器单元或NX单元。

指令	名称	FB/FUN	图形表现	ST表现
RestartNXUnit	NX单元重启	FB		RestartNXUnit_instance(Execute, UnitProxy, Done, Busy, Error, ErrorID、 ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	指定单元	输入	要重启的EtherCAT耦合器单元、NX总线功能模块或NX单元	–	–	(*)

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID 详情参阅功能说明																			

功能

重启EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与NX总线功能模块或CPU单元上的NX总线连接的NX单元。
可单独重启指定的单元。
但无法单独重启EtherCAT耦合器单元或NX总线功能模块。
指定了EtherCAT耦合器单元或NX总线功能模块时，与其连接的NX单元将全部重启。
要重启的单元通过 “UnitProxy”指定。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

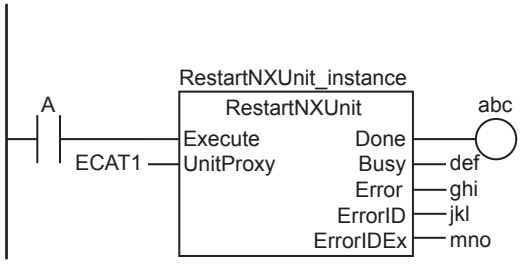
变量	名称	内容	数据类型
UnitProxy	指定单元	要指定的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至指定EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与NX总线功能模块或CPU单元上的NX总线连接的NX单元的设备变量传输至 “UnitProxy”。

记述示例

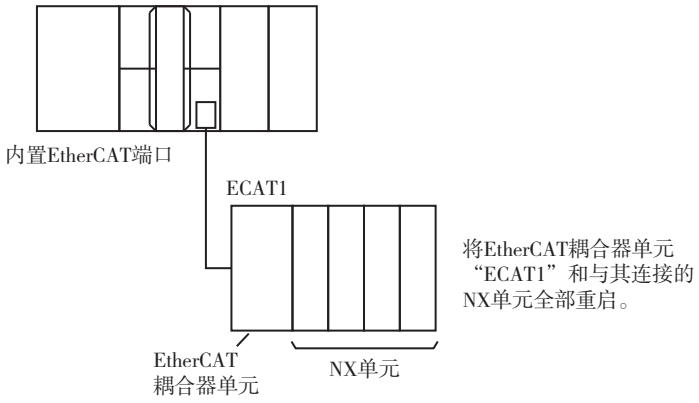
重启所有 EtherCAT 从站终端时的示例如下所示。设 EtherCAT 耦合器单元中分配 _sNXUNIT_ID 型的“ECAT1”变量。

LD



ST

```
RestartNXUnit_instance(A, ECAT1,
                       abc, def, ghi, jkl, mno);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信
_NXB_UnitMsgActiveTbl[i]	NX单元信息可通信状态	BOOL	可进行信息通信的从站一览。 用于确认是否可与相应从站进行通信。

参考

下面介绍将包含以下属性的数据写入EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元时的步骤。

- 带断电保存属性
- 在重启单元时反映数值

1 使用 “NX_ChangeWriteMode指令(P.2-849)”，将对象单元变更成可写入数据的模式。

2 使用 “NX_WriteObj指令(P.2-951)”，将数据写入对象单元。

3 使用 “NX_SaveParam指令(P.2-854)”，保存已写入的数据。

4 使用本指令，重启对象单元。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 在 “UnitProxy”中指定分配至运动控制轴(数据类型_sAXIS_REF)的单元时，运动控制模块会发生控制器异常。请使用 “ResetMCError指令(P.2-813)”解除控制器异常。
- 在 “UnitProxy”中指定通过Sysmac Studio的I/O映射分配至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元、与NX总线功能模块或CPU单元上的NX总线连接的NX单元的设备变量。分配设备变量的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。
- 试图在执行本指令或 “NX_ChangeWriteMode指令(P.2-849)”的过程中执行本指令时，之后试图执行的本指令将进行排队处理。排队处理的最大数量为192。超出192时，会发生异常。此外，排队时间不计入超时时间。
- 本指令排队处理时，“Busy”的值将变为TRUE。
- 本指令是与NX信息通信异常相关的指令。一次性执行多个与NX信息通信异常相关的指令时，可能会发生 NX 信息通信异常。与 NX 信息通信异常相关的指令请参阅 “与 NX 信息通信异常相关的指令(P.A-186)”。
- EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令正在执行时，无法执行本指令。如执行，会发生异常。
- 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0419	16#00000000	“UnitProxy”的数据类型错误。
16#2C00	16#00000401	指定的单元非本指令的对象。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。
	16#00001101	指定的单元不正确。 请确认单元。
	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。
	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。
	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000 ~ 0x6FFF或0x7000 ~ 0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。
	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ~ 16#8FFF0000 16#90010000 ~ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。
	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。
	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C01	16#00000000	本指令或NX_ChangeWriteMode指令的排队数超过了192。
16#2C02	16#00000000	通信过程中发生了超时。
16#2C05	–	EtherCAT网络发生了异常。 请确认 “UnitProxy”的值及EtherCAT网络构成。
16#2C06	16#00000000	指定的单元根据Sysmac Studio的指示已执行重启。因此，无需执行本指令。
16#2C07	16#00000000	指定单元的从站节点地址连接了非本指令对象的从站。请确认“UnitProxy”的值及EtherCAT网络构成。

版本相关信息

本指令可用于CPU单元Ver.1.05以上且Sysmac Studio Ver.1.06以上。
使用部分版本时，以下对象可能不支持 “重启指定NX单元的功能”。

- CPU单元
- Sysmac Studio
- EtherCAT耦合器单元
- NX单元

使用不支持“重启指定NX单元的功能”的版本时，仅EtherCAT耦合器单元可指定为要重启的单元。
支持“重启指定NX单元的功能”的版本请参阅 “NX系列 EtherCAT耦合器单元用户手册(SBCD-361C以上)”。

示例程序

请参阅 “NX_WriteObj指令(P.2-951)”的示例程序。

NX_ChangeWriteMode

将EtherCAT耦合器单元或NX单元变更成可写入数据的模式。

指令	名称	FB/FUN	图形表现	ST表现
NX_ChangeWriteMode	NX单元写入模式变更	FB	NX_ChangeWriteMode_instance NX_ChangeWriteMode Execute / UnitProxy Done / Busy / Error / ErrorID / ErrorIDEx	NX_ChangeWriteMode_instance(Execute, UnitProxy, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	指定单元	输入	要变更模式的单元	–	–	(*)

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID　详情参阅功能说明																			

功能

将EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元变更成可写入数据的模式。

通过 “UnitProxy”指定要变更模式的单元。

“Done”的值变为TRUE时，将进入可写入数据的状态。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	指定单元	要变更写入模式的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至要指定单元的设备变量传输至 “UnitProxy”。

相关指令和执行步骤

本指令用于将包含以下属性的数据写入EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元。

- 带断电保存属性
- 在重启单元时反映数值

相关指令和执行步骤如下所述。

1 使用本指令，将对象单元变更成可写入数据的模式。

2 使用 “NX_WriteObj指令(P.2–951)”，将数据写入对象单元。

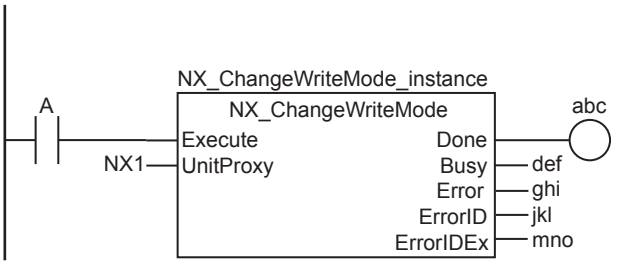
3 使用 “NX_SaveParam指令(P.2–854)”，保存已写入的数据。

4 使用 “RestartNXUnit指令(P.2–844)”，重启对象单元。

记述示例

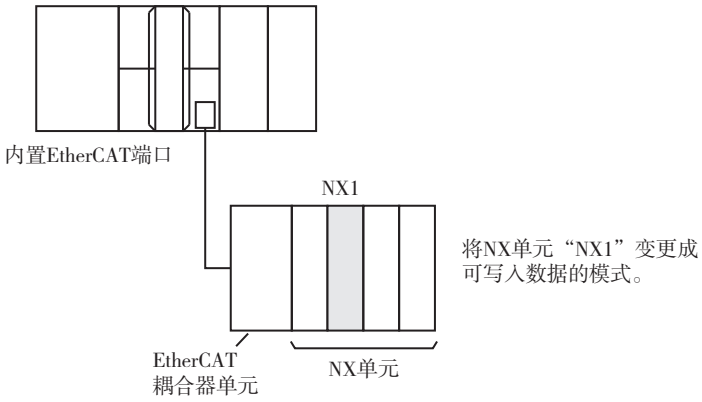
将NX单元 “NX1” 变更成可写入数据的模式时的示例如下所示。设对象NX单元中分配了_sNXUNIT_ID型的 “NX1” 变量。

LD



ST

```
NX_ChangeWriteMode_instance(A, NX1, abc, def,
                            ghi, jkl, mno);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信
_NXB_UnitMsgActiveTbl[i]	NX单元信息可通信状态	BOOL	可进行信息通信的从站一览。 用于确认是否可与相应从站进行通信。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 在 “UnitProxy”中指定分配至运动控制轴(数据类型_sAXIS_REF)的单元时，运动控制模块会发生控制器异常。此时，请使用 “ResetMCError指令(P.2-813)”解除控制器异常。
- 在 “UnitProxy”中指定通过Sysmac Studio的I/O映射分配至EtherCAT耦合器单元、EtherCAT耦合器单元上的 NX 单元、与 CPU 单元上的 NX 总线连接的 NX 单元的设备变量。分配设备变量的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。
- 试图在执行本指令或 “RestartNXUnit指令(P.2-844)”的过程中执行本指令时，之后试图执行的本指令将进行排队处理。排队处理的最大数量为192。超出192时，会发生异常。此外，排队时间不计入超时时间。
- 本指令排队处理时，“Busy”的值将变为TRUE。
- 本指令是与NX信息通信异常相关的指令。一次性执行多个与NX信息通信异常相关的指令时，可能会发生 NX 信息通信异常。与 NX 信息通信异常相关的指令请参阅 “与 NX 信息通信异常相关的指令(P.A-186)”。
- EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令正在执行时，无法执行本指令。如执行，会发生异常。
- 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0419	16#00000000	“UnitProxy”的数据类型错误。
16#2C00	16#00000401	指定的单元非本指令的对象。
	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#00001101	指定的单元不正确。 请确认单元。
	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。
	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。
	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000 ~ 0x6FFF或0x7000 ~ 0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。
	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ~ 16#8FFF0000 16#90010000 ~ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。
	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。
	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C01	16#00000000	本指令或RestartNXUnit指令的排队数超过了192。
16#2C02	16#00000000	通信过程中发生了超时。
16#2C05	–	EtherCAT网络发生了异常。 请确认 “UnitProxy”的值及EtherCAT网络构成。
16#2C07	16#00000000	指定单元的从站节点地址连接了非本指令对象的从站。请确认“UnitProxy”的值及EtherCAT网络构成。

版本相关信息

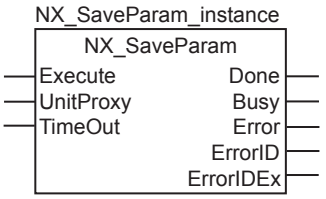
本指令可用于CPU单元Ver.1.05 以上且Sysmac Studio Ver.1.06 以上。

示例程序

请参阅 “NX_WriteObj指令(P.2–951)”的示例程序。

NX_SaveParam

保存写入EtherCAT耦合器单元或NX单元的数据。

指令	名称	FB/FUN	图形表现	ST表现
NX_SaveParam	NX单元参数保存	FB		NX_SaveParam_instance(Execute, UnitProxy, TimeOut, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	指定单元	输入	要保存数据的单元	–	–	(*)
TimeOut	超时时间		超时时间 0时为2.0s	0 ~ 60000	ms	2000 (2.0s)

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID 详情参阅功能说明																			
TimeOut							○													

功能

保存写入至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的数据。

通过 “UnitProxy”指定要保存数据的单元。

“Done”的值变为TRUE时，数据保存完成。

写入数据的指令为 “NX_WriteObj指令(P.2–951)”。

执行本指令后，即使发生断电，带断电保存属性的数据值仍会得到保存。

超时时间由“TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。此时，不会保存单元数据。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	指定单元	要保存数据的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至要指定单元的设备变量传输至 “UnitProxy”。

相关指令和执行步骤

本指令根据写入至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的数据属性，有时需与其它相关指令配合执行。各种情况下的执行步骤如下所述。

● 执行步骤1

下面介绍写入带以下属性的数据时的步骤。

- 带断电保存属性
- 在重启单元时反映数值

1 使用 “NX_ChangeWriteMode指令(P.2–849)”，将对象单元变更成可写入数据的模式。

2 使用 “NX_WriteObj指令(P.2–951)”，将数据写入对象单元。

3 使用本指令，保存已写入的数据。

4 使用 “RestartNXUnit指令(P.2–844)”，重启对象单元。

● 执行步骤2

下面介绍写入带以下属性的数据时的步骤。

- 带断电保存属性
- 写入后即反映数值

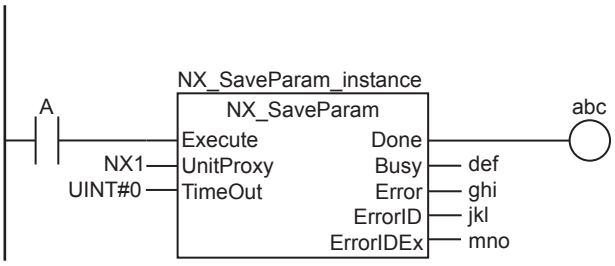
1 使用 “NX_WriteObj指令(P.2–951)”，将数据写入对象单元。

2 使用本指令，保存已写入的数据。

记述示例

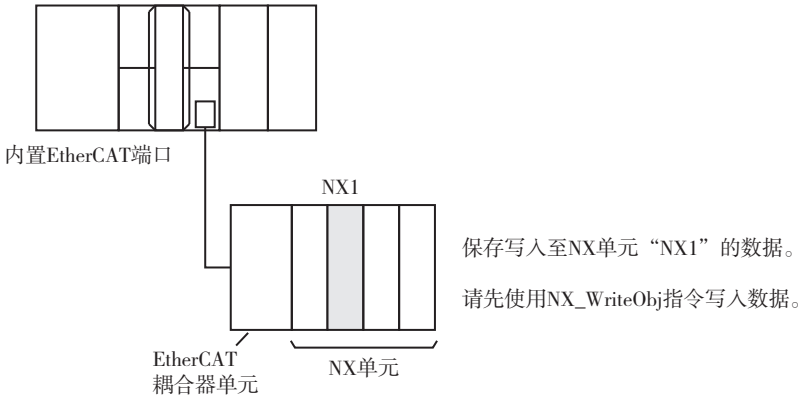
保存写入至NX单元“NX1”的数据时的示例如下所示。设对象NX单元中分配了_sNXUNIT_ID型的“NX1”变量。

LD



ST

```
NX_SaveParam_instance(A, NX1, UINT#0,
                      abc, def, ghi, jkl, mno);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- “UnitProxy”中指定的单元正在保存数据时，本指令不会发生异常。“Busy”的值变为TRUE，数据保存完成时 “Done”的值变为TRUE。数据保存的请求不进行排队处理。
- 即使在未对单元写入数据的状态下执行本指令，也不会发生异常。
- 部分单元对数据的可写入次数存在限制。详情请参阅各单元的手册。
- 在 “UnitProxy”中指定通过Sysmac Studio的I/O映射分配至EtherCAT耦合器单元、EtherCAT耦合器单元上的 NX 单元、与 CPU 单元上的 NX 总线连接的 NX 单元的设备变量。分配设备变量的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。
- 需要写入并保存具有断电保存属性的数据时，请执行 “NX_WriteObj指令(P.2-951)”后再执行本指令。未执行本指令而重启对象单元时，NX对象的数据将恢复为写入前的值。
- 本指令是与NX信息通信异常相关的指令。一次性执行多个与NX信息通信异常相关的指令时，可能会发生 NX 信息通信异常。与 NX 信息通信异常相关的指令请参阅 “与 NX 信息通信异常相关的指令(P.A-186)”。

• 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0400	16#00000000	· “UnitProxy”的值超过有效范围。 · “TimeOut”的值超过有效范围。
16#0419	16#00000000	“UnitProxy”的数据类型错误。
16#2C00	16#00000401	指定的单元非本指令的对象。
	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。
	16#00001101	指定的单元不正确。 请确认单元。
	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。
	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。
	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000 ~ 0x6FFF或0x7000 ~ 0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。
	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ~ 16#8FFF0000 16#90010000 ~ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。
	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。
	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C01	16#00000000	超过了可同时执行的指令数。
16#2C02	16#00000000	通信过程中发生了超时。

版本相关信息

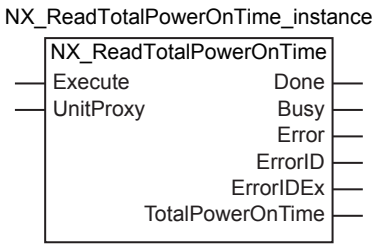
本指令可用于CPU单元Ver.1.05 以上且Sysmac Studio Ver.1.06 以上。

示例程序

请参阅 “NX_WriteObj指令(P.2–951)”的示例程序。

NX_ReadTotalPowerOnTime

读取通信耦合器单元及NX单元保存的累计通电时间。

指令	名称	FB/FUN	图形表现	ST表现
NX_ReadTotalPowerOnTime	NX单元累计通电时间读取	FB		NX_ReadTotalPowerONTime_instance(Execute, UnitProxy, Done, Busy, Error, ErrorID, ErrorIDEx, TotalPowerOnTime);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	单元指定	输入	指定对象NX单元。	–	–	*1
TotalPowerOnTime	累计通电时间	输出	输出读取的累计通电时间。	遵从数据类型		0

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID 详情参阅功能说明																			
TotalPowerOnTime																○				

功能

以通信耦合器单元、通信耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的累计通电时间为大致标准进行读取。
精度为1小时/月。

通过 “UnitProxy”指定要读取累计通电时间的单元。
“Done”的值变为TRUE时，累计通电时间读取完成。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	指定单元	要指定的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定的NX单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至指定通信耦合器单元、通信耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的设备变量传输至 “UnitProxy”。

基于版本的组合

根据与CPU单元连接的通信耦合器单元、通信耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的版本，部分组合可读取累计通电时间。

● EtherCAT从站终端

单元	NX单元的版本	EtherCAT耦合器单元的版本
数字 I/O单元	Ver.1.0以上	Ver.1.2以上
模拟I/O单元		
系统单元		
位置接口单元	Ver.1.1以上	
温度输入单元		

● NX1P2上的NX单元

单元	NX单元的版本
数字 I/O单元	Ver.1.0以上
模拟I/O单元	
系统单元	
位置接口单元	Ver.1.1以上
温度输入单元	

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信
_NXB_UnitMsgActiveTbl[i]	NX单元信息可通信状态	BOOL	可进行信息通信的从站一览。 用于确认是否可与相应从站进行通信。

参考

使用模拟器运行本指令时，“Execute”从False变为True的情况下，仅1个任务周期内“Busy”会变为True。

下一任务周期内，“Busy”变为False，“Done”变为True。
读取的 “TotalPowerOnTime”将变为 “0”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 在 “UnitProxy”中指定通过Sysmac Studio的I/O映射分配至EtherCAT耦合器单元、EtherCAT耦合器单元上的 NX 单元、与 CPU 单元上的 NX 总线连接的 NX 单元的设备变量。分配设备变量的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA–362G以上)”。
  在 “UnitProxy”中指定了NX系列 CPU单元时，将发生异常。
- 台数限制因通信耦合器而异。详情请参阅通信耦合器单元的手册。
- 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0400	16#00000000	“UnitProxy”的值超过有效范围。
16#0419	16#00000000	“UnitProxy”的数据类型错误。
16#2C00	16#00000401	指定的单元非本指令的对象。
	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。
	16#00001101	指定的单元不正确。 请确认单元。
	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。
	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000～0x6FFF或0x7000～0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。
	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。
	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ～ 16#8FFF0000 16#90010000 ～ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。
	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。
16#2C00	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C01	16#00000000	超过了可同时执行的指令数。
16#2C02	16#00000000	通信过程中发生了超时。
16#2C08	16#00000000	未能读取通电时间。

示例程序

在程序中创建维护模式和RUN模式。
其中，在维护模式时按下累计时间读取按钮，将读取事先确定的单元3的累计通电时间。
累计通电时间超过5年时，更换单元通知指示灯将点亮。
更换单元后，按下更换单元结束按钮时，更换单元通知指示灯将熄灭。

采用以下系统构成。

单元	内容
单元1	NX单元(ID)
单元2	NX单元(OD)
单元3	NX单元(读取累计通电时间的对象)

变量的定义

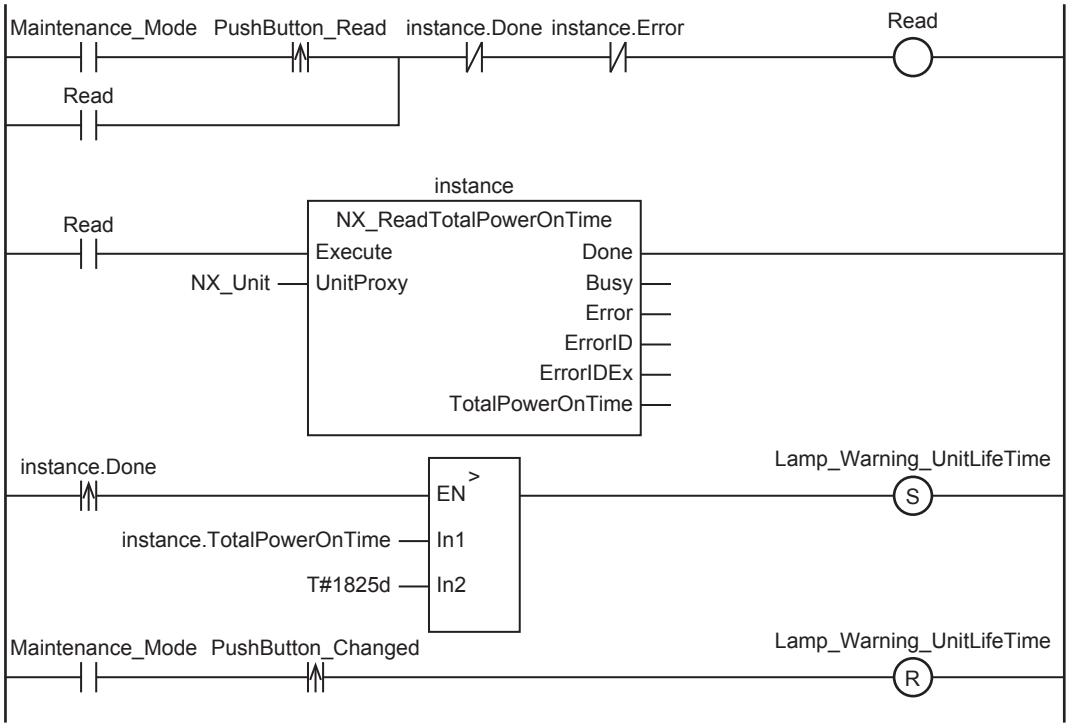
LD

内部变量	名称	数据类型	初始值	注释
	Maintenance_Mode	BOOL	FALSE	维护模式
	Run_Mode	BOOL	FALSE	RUN模式
	PushButton_Read	BOOL	FALSE	累计时间读取
	PushButton_Changed	BOOL	FALSE	更换单元结束
	Lamp_Warning_UnitLifeTime	BOOL	FALSE	更换单元通知
	Read	BOOL	FALSE	
	instance	NX_ReadTotalPowerOnTime		

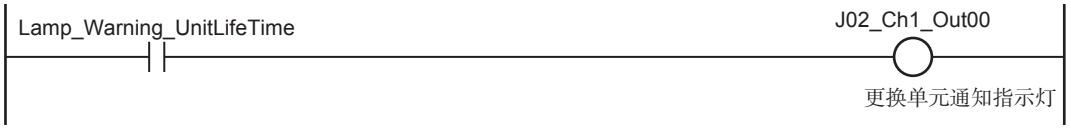
外部变量	名称	数据类型	注释
	NX_Unit	_sNXUNIT_ID	
	J01_Ch1_In00	BOOL	维护模式按钮
	J01_Ch1_In01	BOOL	累计通电时间读取按钮
	J01_Ch1_In02	BOOL	更换单元结束按钮
	J02_Ch1_Out00	BOOL	更换单元通知指示灯

获取按钮的状态

累计通电时间读取



输出至指示灯



ST

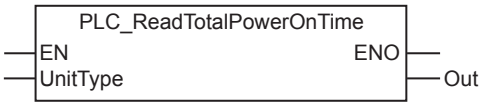
内部变量	名称	数据类型	初始值	注释
	Maintenance_Mode	BOOL	FALSE	维护模式
	Run_Mode	BOOL	FALSE	RUN模式
	PushButton_Read	BOOL	FALSE	累计时间读取
	PushButton_Changed	BOOL	FALSE	更换单元结束
	Lamp_Warning_UnitLifeTime	BOOL	FALSE	更换单元通知
	Read	BOOL	FALSE	
	instance	NX_ReadTotalPowerOnTime		
	RS_instance	RS		
	RS_instance2	RS		
	R_TRIG_instance1	R_TRIG		
	R_TRIG_instance2	R_TRIG		
	R_TRIG_instance3	R_TRIG		
	PushButton_Read_R_TRIG	BOOL		
	instance_Done_R_TRIG	BOOL		
	PushButton_Change_R_TRIG	BOOL		

外部变量	名称	数据类型	注释
	NX_Unit	_sNXUNIT_ID	
	J01_Ch1_In00	BOOL	维护模式按钮
	J01_Ch1_In01	BOOL	累计通电时间读取按钮
	J01_Ch1_In02	BOOL	更换单元结束按钮
	J02_Ch1_Out00	BOOL	更换单元通知指示灯

```
// 获取按钮的状态
Maintenance_Mode   := J01_Ch1_In00;
Run_Mode           := NOT(J01_Ch1_In00);
PushButton_Read    := J01_Ch1_In01;
PushButton_Changed := J01_Ch1_In02;

R_TRIG_instance1(Clk:= PushButton_Read, Q=>PushButton_Read_R_TRIG);

// 累计通电时间读取
Rs_instance( Set:= (Maintenance_Mode & PushButton_Read_R_TRIG),
        Reset1:=((instance.Done) OR (instance.Error)),
        Q1=>Read);
instance(Execute:=Read, UnitProxy:=NX_Unit);

R_TRIG_instance2(Clk:= instance.Done, Q=>instance_Done_R_TRIG);
R_TRIG_instance3(Clk:= PushButton_Changed, Q=>PushButton_Changed_R_TRIG);

RS_instance2(Set:=(instance_Done_R_TRIG & (instance.TotalPowerOnTime>T#1825d)),
        Reset1:=(Maintenance_Mode & PushButton_Changed_R_TRIG),
        Q1=>Lamp_Warning_UnitLifeTime);

// 输出至指示灯
J02_Ch1_Out00  := Lamp_Warning_UnitLifeTime;
```

PLC_ReadTotalPowerOnTime

读取指定CPU单元保存的累计通电时间。

指令	名称	FB/FUN	图形表现	ST表现
PLC_ReadTotalPowerOnTime	PLC累计通电时间读取	FUN		Out:=PLC_ReadTotalPowerOnTime(UnitType, ENO);

使用注意事项

本指令仅NX1P2 CPU单元可使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitType	单元类型	输入	对象单元的指定	_CPU_UNIT	–	_CPU_UNIT
Out	累计通电时间	输出	读取的累计通电时间的输出	遵从数据类型		–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitType	枚举体_ePLC_UNIT_TYPE　枚举元素参阅功能说明																			
Out																○				

功能

以指定单元的累计通电时间为大致标准进行读取。
精度为1小时/月。

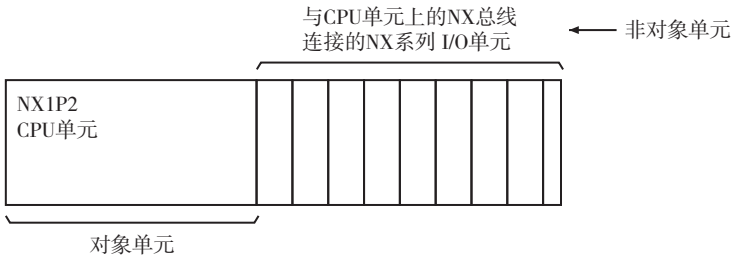
通过 “UnitType”指定要读取累计通电时间的单元。

“UnitType”的数据类型为枚举体_ePLC_UNIT_TYPE。枚举元素的含义如下所示。

枚举元素	含义
_CPU_UNIT	指定CPU单元

对象单元

● NX1P2



读取对象只有NX1P2 CPU单元。
在 “UnitType”中设置枚举体_ePLC_UNIT_TYPE的枚举元素_CPU_UNIT进行指定。
与CPU单元上的NX总线连接的NX系列 I/O单元非读取对象。

参考

读取NX系列I/O单元的累计通电时间时，请使用 “NX_ReadTotalPowerOnTime指令(P.2-859)”。

参考

使用模拟器运行本指令时，“Out”的值始终为 “T#0s”。

使用注意事项

指定非对象单元时，会发生异常。
“ENO”变为FALSE，“Out”的值不变。

示例程序

在程序中创建Maintenance_Mode和Run_Mode。
其中，在Maintenance_Mode时按下累计时间读取按钮，将读取NX1P2 CPU单元的累计通电时间。
累计通电时间超过5年时，更换单元通知指示灯将点亮。
更换单元后，按下更换单元结束按钮时，更换单元通知指示灯将熄灭。

采用以下系统构成。

单元	内容
单元1	与NX系列 CPU单元上的NX总线连接的单元1 NX系列 I/O单元(ID)
单元2	与NX系列 CPU单元上的NX总线连接的单元2 NX系列 I/O单元(IO)
单元3	NX1P2 CPU单元(要读取累计通电时间的对象)

变量的定义

ST

外部变量	名称	数据类型	注释
	J01_Ch1_In00	BOOL	维护模式按钮
	J01_Ch1_In01	BOOL	累计通电时间读取按钮
	J02_Ch1_Out00	BOOL	更换单元通知指示灯
	Maintenance_Mode	BOOL	
	Run_Mode	BOOL	
	PushButton_Read	BOOL	
	Lamp_Warning_UnitLifeTime	BOOL	
	PowerOnTime	TIME	
	R_TRIG_instance1	R_TRIG	
	PushButton_Read_R_TRIG	BOOL	
	RS_instance	RS	

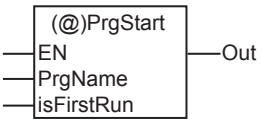
```
// 获取按钮的状态
Maintenance_Mode  := J01_Ch1_In00;
Run_Mode          := NOT(J01_Ch1_In00);
PushButton_Read   := J01_Ch1_In01;

R_TRIG_instance1(clk:=PushButton_Read, Q=>PushButton_Read_R_TRIG);

// 累计运转时间读取
PowerOnTime  := PLC_ReadTotalPowerOnTime(EN:=(Maintenance_Mode & PushButton_Read_R_TRIG),
                                         UnitType:=_CPU_UNIT);

RS_instance( Set:=(PowerOnTime > T#1825d),
        Reset1:=Maintenance_Mode,
        Q1=>Lamp_Warning_UnitLifeTime);

// 输出至指示灯
J02_Ch1_Out00  := Lamp_Warning_UnitLifeTime;
```

程序控制指令

指令	名称	页码
PrgStart	程序执行指令	2–870
PrgStop	程序停止指令	2–878
PrgStatus	程序状态读取	2–896

PrgStart

发出执行指定程序的指令。

指令	名称	FB/FUN	图形表现	ST表现
PrgStart	程序执行指令	FUN		Out:=PrgStart(PrgName, isFirstRun);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
PrgName	程序名	输入	要指定的程序名	最大128字节(127个半角英数字字符+结尾NULL字符)	–	*1
isFirstRun	程序启动时1个周期ON标志有效		指定执行程序时，系统定义变量P_First_Run首个任务周期的动作 TRUE ：设为TRUE FALSE：设为FALSE	遵从数据类型		TRUE
Out	正常结束标志	输出	正常结束标志 TRUE ：正常结束 FALSE：异常结束	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
PrgName																				○
isFirstRun	○																			
Out	○																			

功能

发出指令，执行程序名“PrgName”指定的程序。指定程序在下一次执行程序时执行。指定程序已在执行中时，不会发生异常。

指定程序与执行本指令的任务在同一任务或不同任务内均可。

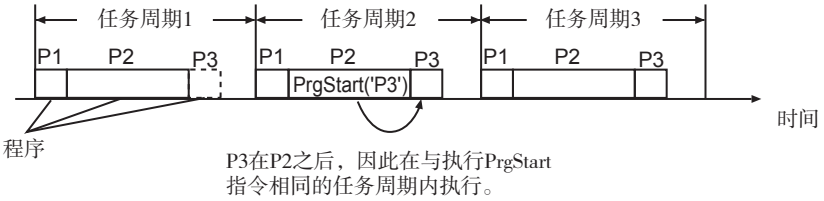
正常结束标志 “Out”的值在本指令正常结束时变为TRUE，异常结束时变为FALSE。

指定我的任务的程序时的动作示例

下面介绍指定与执行本指令的任务在同一任务内的程序时的动作示例。

● 发出PrgStart指令之后的程序的执行指令时

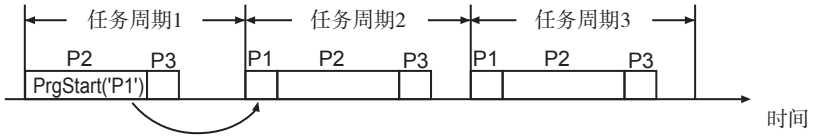
- 同一任务内有P1、P2及P3等三个程序。
- 任务周期1内，P3为停止状态。
- 在任务周期2的P2内，指定P3，执行PrgStart指令。
- P3在P2之后，因此P3在任务周期2内执行。
- 之后，重新指定P3，即使不执行PrgStart指令，仍会执行P3。



P3在P2之后，因此在与执行PrgStart指令相同的任务周期内执行。

● 发出PrgStart指令之前的程序的执行指令时

- 同一任务内有P1、P2及P3等三个程序。
- 任务周期1内，P1为停止状态。
- 在任务周期1的P2内，指定P1，执行PrgStart指令。
- P1在P2之前，因此P1在任务周期2内执行。
- 之后，重新指定P1，即使不执行PrgStart指令，仍会执行P1。



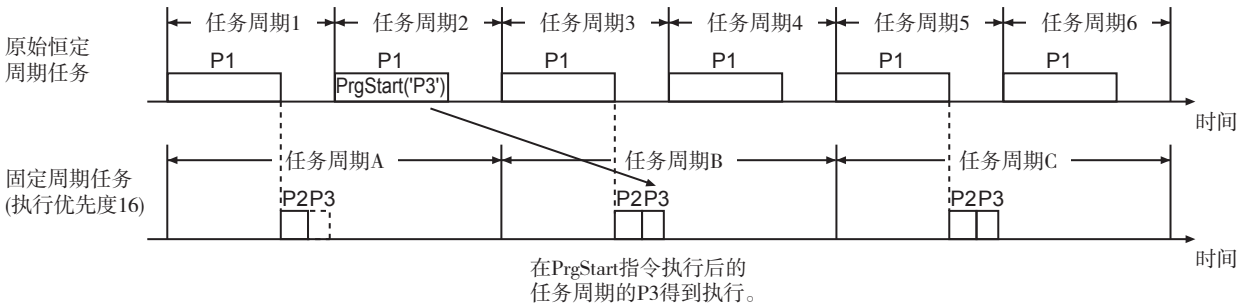
P1在P2之前，因此在执行PrgStart指令的下一任务周期内执行。

指定其它任务的程序时的动作示例

下面介绍指定与执行本指令的任务在不同任务内的程序时的动作示例。

● 发出优先度低于我的任务的其它任务内程序的执行指令时

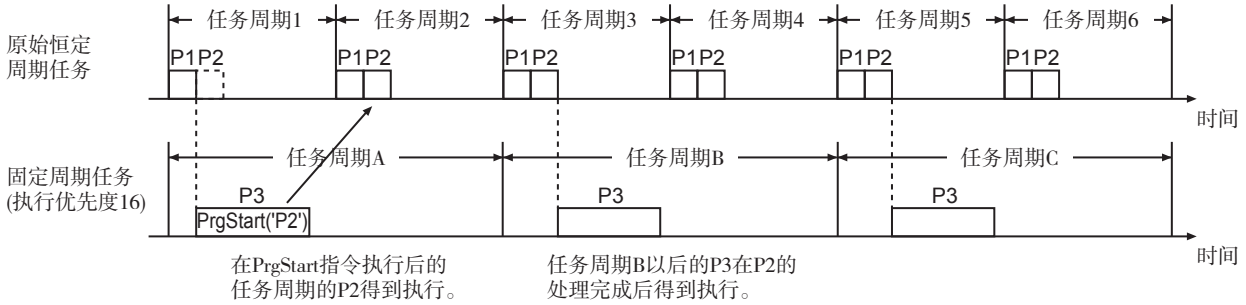
- 原始恒定周期任务中有P1、固定周期任务中有P2、P3共三个程序。
- 固定周期任务的任务周期A中，P3为停止状态。
- 在原始恒定周期任务的任务周期2的P1内，指定P3，执行PrgStart指令。
- 在PrgStart指令执行后执行的固定周期任务中任务周期B的P3将得到执行。
- 之后，重新指定P3，即使不执行PrgStart指令，仍会执行P3。



在PrgStart指令执行后的任务周期的P3得到执行。

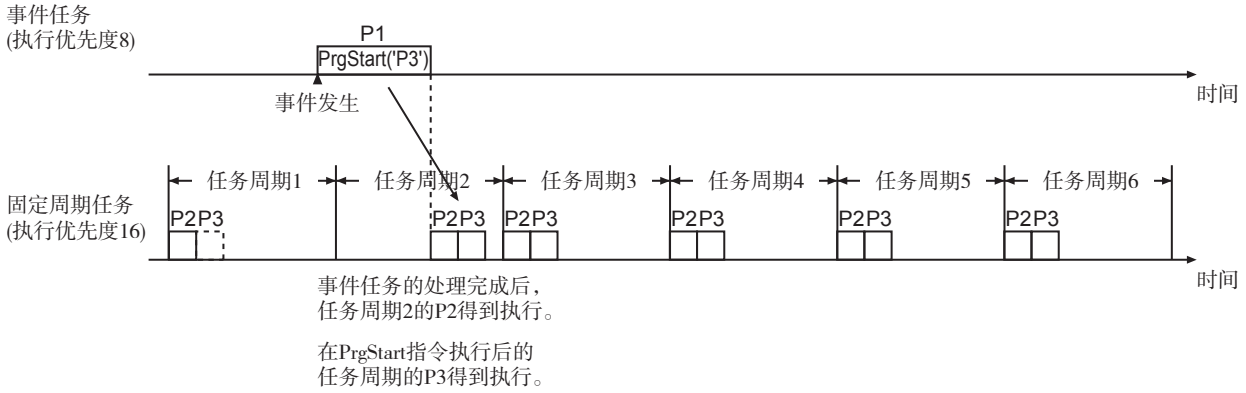
● 发出优先度高于我的任务的其它任务内程序的执行指令时

- 原始恒定周期任务中有P1、P2，固定周期任务中有P3共三个程序。
- 原始恒定周期任务的任务周期1中，P2为停止状态。
- 在固定周期任务的任务周期A的P3内，指定P2，执行PrgStart指令。
- 在PrgStart指令执行后执行的原始恒定周期任务中任务周期2的P2将得到执行。
- 之后，重新指定P2，即使不执行PrgStart指令，仍会执行P2。
- 原始恒定周期任务的执行优先度高于固定周期任务，因此任务周期B以后的P3将在P2处理完成后执行。



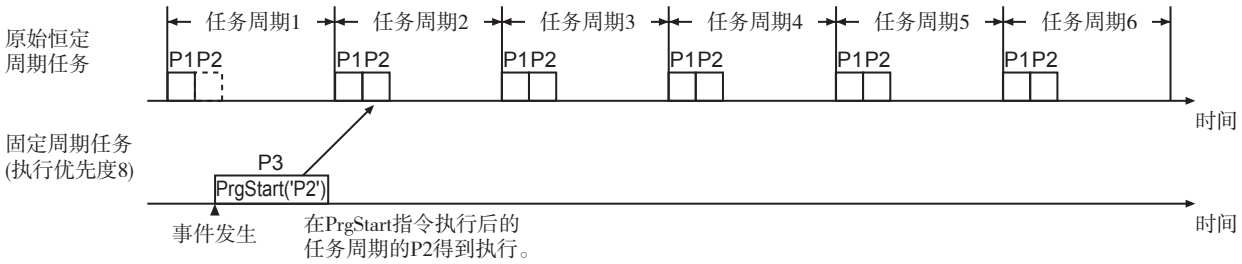
● 从事件任务中发出优先度较低任务内程序的执行指令时

- 事件任务(执行优先度8)中有P1，固定周期任务(执行优先度16)中有P2、P3共三个程序。
- 固定周期任务的任务周期1中，P3为停止状态。
- 在事件任务内，指定P3，执行PrgStart指令。
- 执行事件任务后，固定周期任务的任务周期2的P2、P3将在事件任务处理完成后执行。
- 固定周期任务中任务周期2的P3在PrgStart指令执行之后，因此会得到执行。
- 之后，重新指定P3，即使不执行PrgStart指令，仍会执行P3。



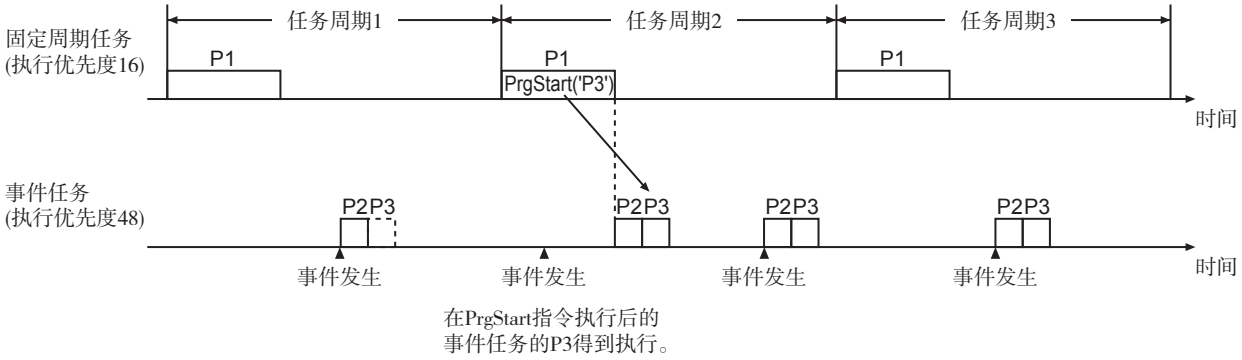
● 从事件任务中发出优先度较高任务内程序的执行指令时

- 原始恒定周期任务中有P1、P2，事件任务中有P3共三个程序。
- 原始恒定周期任务的任务周期1中，P2为停止状态。
- 在事件任务内，指定P2，执行PrgStart指令。
- 在PrgStart指令执行后执行的原始恒定周期任务中任务周期2的P2将得到执行。
- 之后，重新指定P2，即使不执行PrgStart指令，仍会执行P2。



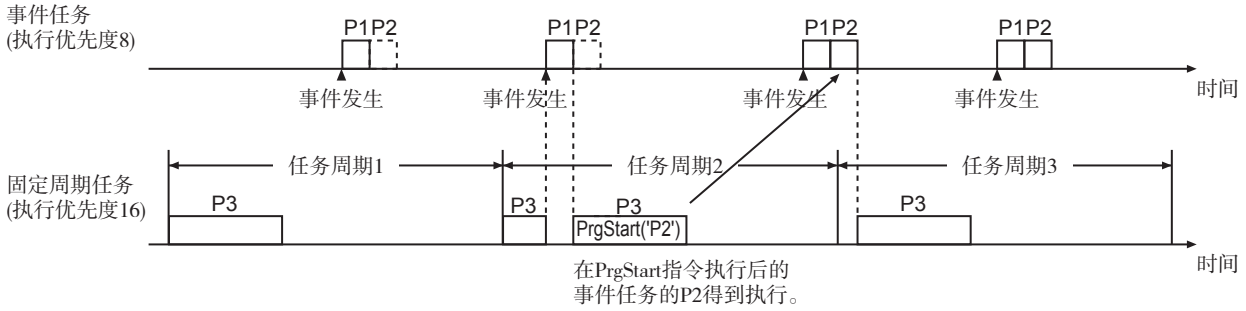
● 从固定周期任务中发出优先度较低事件任务内程序的执行指令时

- 固定周期任务(执行优先度16)中有P1，事件任务(执行优先度48)中有P2、P3共三个程序。
- 事件任务的P3为停止状态。
- 在固定周期任务内，指定P3，执行PrgStart指令。
- 在PrgStart指令执行后执行的事件任务的P3将得到执行。
- 之后，重新指定P3，即使不执行PrgStart指令，仍会执行P3。



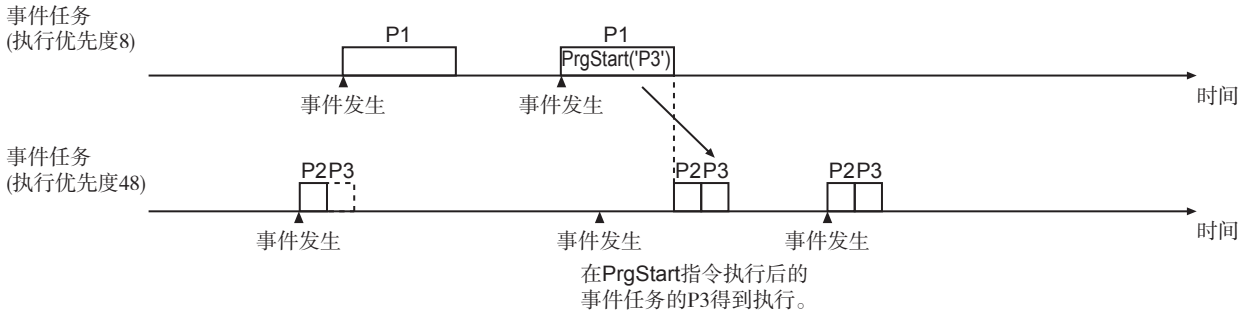
● 从固定周期任务中发出优先度较高事件任务内程序的执行指令时

- 事件任务(执行优先度8)中有P1和P2，固定周期任务(执行优先度16)中有P3共三个程序。
- 事件任务的P2为停止状态。
- 在固定周期任务内，指定P2，执行PrgStart指令。
- 在PrgStart指令执行后执行的事件任务的P2将得到执行。
- 之后，重新指定P2，即使不执行PrgStart指令，仍会执行P2。



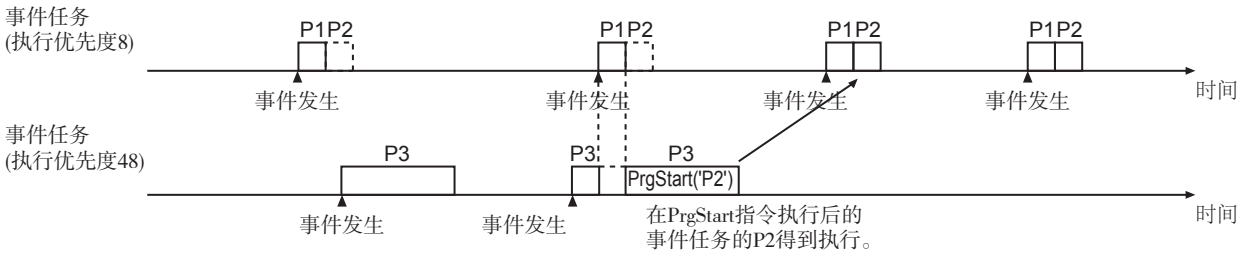
● 从事件任务中发出优先度较低事件任务内程序的执行指令时

- 事件任务(执行优先度8)中有P1，事件任务(执行优先度48)中有P2、P3共三个程序。
- 事件任务(执行优先度48)的P3为停止状态。
- 在事件任务(执行优先度8)内，指定P3，执行PrgStart指令。
- 在PrgStart指令执行后执行的事件任务(执行优先度48)的P3将得到执行。
- 之后，重新指定P3，即使不执行PrgStart指令，仍会执行P3。



● 从事件任务中发出优先度较高事件任务内程序的执行指令时

- 事件任务(执行优先度8)中有P1和P2，事件任务(执行优先度48)中有P3共三个程序。
- 事件任务(执行优先度8)的P2为停止状态。
- 在事件任务(执行优先度48)内，指定P2，执行PrgStart指令。
- 在PrgStart指令执行后执行的事件任务(执行优先度8)的P2将得到执行。
- 之后，重新指定P2，即使不执行PrgStart指令，仍会执行P2。



程序启动时1个周期ON标志有效 “isFirstRun”

程序启动时1个周期ON标志有效 “isFirstRun”是指定是否将程序启动时1个周期ON标志的系统定义变量“P_First_Run”设为有效的变量。将 “isFirstRun”的值设为TRUE后执行本指令时，“P_First_Run”在相应程序启动后，其值只在1个任务周期内为TRUE。“isFirstRun”的值为FALSE时，即使相应程序启动，“P_First_Run”的值仍会保持FALSE不变。

“isFirstRun”用于仅在某条件成立时，需在相应程序启动后执行特定处理的情况。条件成立时，将“isFisrtRun”的值设为TRUE后执行本指令。相应程序采用了在 “P_First_Run”的值为TRUE时执行特定处理的算法。

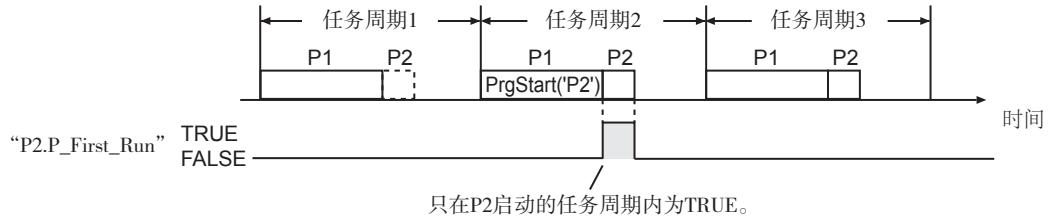
“isFirstRun”与 “P_First_Run”的关系如下所述。“P_First_Run”的动作会根据相应程序为已停止或是已启动而改变。

“isFirstRun”的值	相应程序的状态	“P_First_Run”的值
TRUE	已停止	程序启动后，在1个任务周期内为TRUE。从下一任务周期起变为FALSE。
	已启动	保持FALSE不变。
FALSE	–	保持FALSE不变。

“isFirstRun”与 “P_First_Run”的关系示例如下图所示。

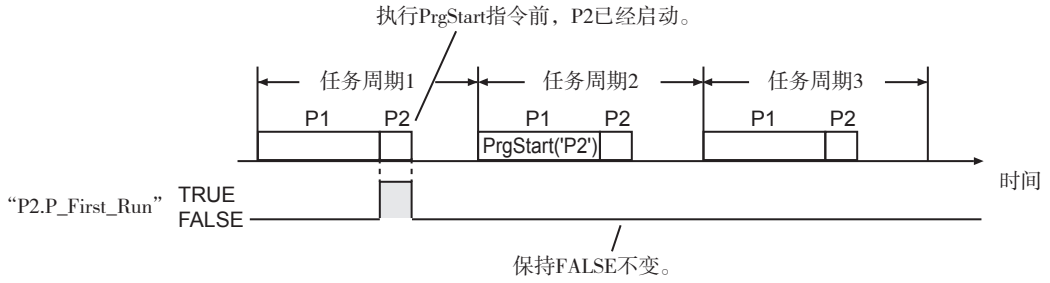
● “isFirstRun”的值为TRUE，相应程序停止时

相应程序启动后，“P_First_Run”的值将在1个任务周期内为TRUE。此后，“P_First_Run”的值将变为FALSE。



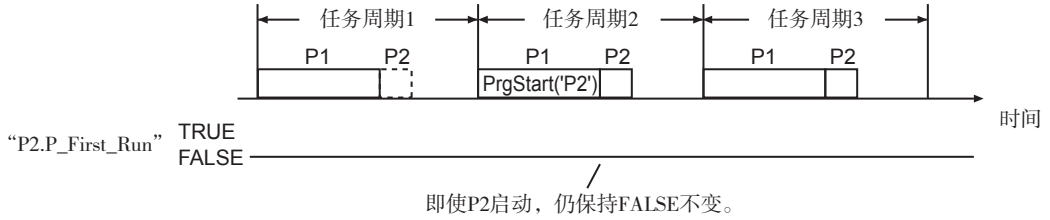
- **“isFirstRun”的值为TRUE，相应程序已启动时**

即使执行PrgStart指令，“P_First_Run”的值仍会保持FALSE不变。



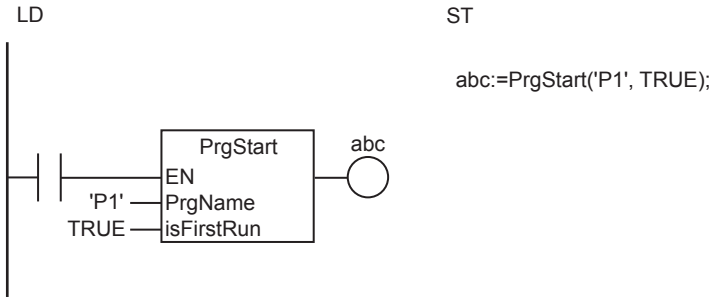
- **“isFirstRun”的值为FALSE时**

即使相应程序启动，“P_First_Run”的值仍会保持FALSE不变。



记述示例

发出程序 “P1”的执行指令时的记述示例如下所示。



相关的系统定义变量

变量名称	名称	数据类型	内容
P_First_Run	程序启动时1个周期ON标志	BOOL	开始执行相应程序后，将在1个任务周期内为TRUE。 其余情况下，均为FALSE。 但将 “isFirstRun”的值设为FALSE后执行了PrgStart指令时，即使开始执行相应程序，仍将保持FALSE。 用于需在每次开始执行相应程序时执行特定处理的情况。
P_First_RunMode	运转开始时1个周期ON标志	BOOL	将控制器的动作模式从程序模式变更为运行模式后，相应程序正在执行中时，仅在1个任务周期内为TRUE。 若相应程序未在执行中，则保持FALSE状态。 用于CPU单元运行开始后进行初始处理等场合。

参考

- 需通过用户程序停止指定程序时，请使用 “PrgStop指令(P.2-878)”。
- 需通过用户程序确认指定程序的状态时，
  请使用 “PrgStatus指令(P.2-896)”。

使用注意事项

- 指定了已处于执行状态的程序后执行本指令时，不会发生异常。
- 指定同一程序后执行了多个本指令时，“isFirstRun”的指定将按照最早执行的指令。
- 执行PrgStart指令，在相应程序执行前执行了PrgStop指令时，相应程序将不会执行。
- 执行PrgStop指令，在相应程序停止前执行了PrgStart指令时，相应程序将不会停止。
- 控制器的动作模式切换为运行模式后，各程序的动作取决于Sysmac Studio各程序的 “初始状态”设定。即切换运行模式前执行的PrgStart指令及PrgStop指令，在切换运行模式后无效。
- 指定了与执行本指令的任务不同的任务内的程序时，指定程序的执行时间因任务双方的执行优先度而异，控制器可能会发生意外动作。通过分配了指定程序的任务的开头程序执行本指令，可在执行了本指令的任务周期中切实执行指定程序。
- 指定程序的内部变量、输入变量、输出变量、输入输出变量将保持该程序上一次执行时的值。需初始化这些变量后再执行程序时，请将 “isFirstRun”的值设为 TRUE 后执行本指令，并在指定程序内的“P_First_Run”的值为TRUE时加入初始化变量的处理。
- 以下情况时会发生异常。“Out”为FALSE。
  - “PrgName”指定的程序不存在时。

版本相关信息

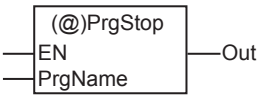
本指令可用于CPU单元Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

请参阅 “PrgStop指令(P.2-878)”的示例程序。

PrgStop

发出指定程序的停止指令。

指令	名称	FB/FUN	图形表现	ST表现
PrgStop	程序停止指令	FUN		Out:=PrgStop(PrgName);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
PrgName	程序名	输入	要指定的程序名	最大128字节(127个半角英数字字符+结尾NULL字符)	–	*1
Out	正常结束标志	输出	正常结束标志 TRUE ：正常结束 FALSE：异常结束	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
PrgName																				○
Out	○																			

功能

发出指令，停止程序名 “PrgName”指定的程序。指定程序在下一次执行程序时停止。指定程序已停止时，不会发生异常。

指定程序与执行本指令的任务在同一任务或不同任务内均可。

也可指定包含本指令的程序。此时，包含本指令的程序将执行至执行本指令的任务周期的末尾，从下一任务周期起停止。

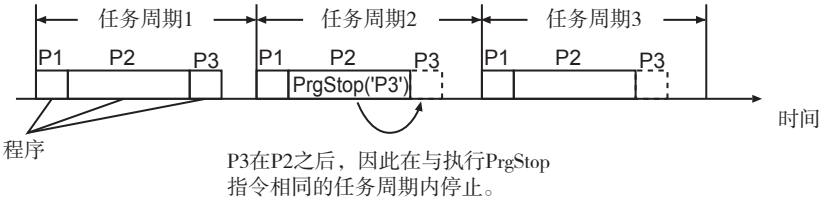
正常结束标志 “Out”的值在本指令正常结束时变为TRUE，异常结束时变为FALSE。

指定我的任务的程序时的动作示例

下面介绍指定与执行本指令的任务在同一任务内的程序时的动作示例。

● 发出PrgStart指令之后的程序的停止指令时

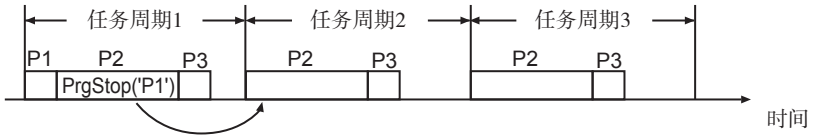
- 同一任务内有P1、P2及P3等三个程序。
- 任务周期1内，P3得到执行。
- 在任务周期2的P2内，指定P3，执行PrgStop指令。
- P3在P2之后，因此P3在任务周期2内停止。
- 之后，重新指定P3，即使不执行PrgStop指令，P3仍会保持停止状态。



P3在P2之后，因此在与执行PrgStop指令相同的任务周期内停止。

● 发出PrgStart指令之前的程序的停止指令时

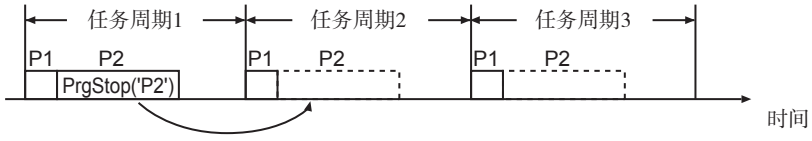
- 同一任务内有P1、P2及P3等三个程序。
- 任务周期1内，P1得到执行。
- 在任务周期1的P2内，指定P1，执行PrgStop指令。
- P1在P2之前，因此P1在任务周期2内停止。
- 之后，重新指定P1，即使不执行PrgStop指令，P1仍会保持停止状态。



P1在P2之前，因此在执行PrgStop指令的下一任务周期内停止。

● 发出包含PrgStop指令的程序的停止指令时

- 同一任务内有P1及P2两个程序。
- 任务周期1内，P2得到执行。
- 在任务周期1的P2内，指定P2，执行PrgStop指令。
- 任务周期1的P2将执行至末尾。
- 从任务周期2起，P2停止。
- 之后，重新指定P2，即使不执行PrgStop指令，P2仍会保持停止状态。



执行了PrgStop指令的任务周期将执行至程序的末尾。

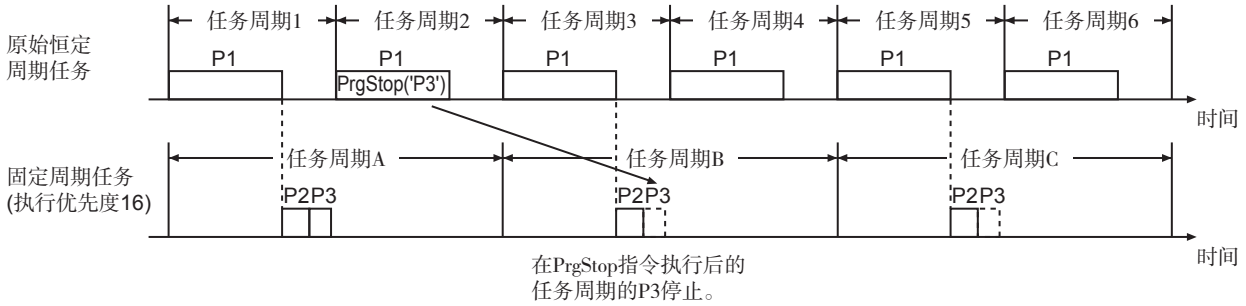
从执行了PrgStop指令的任务周期的下一任务周期起，程序停止。

指定其它任务的程序时的动作示例

下面介绍指定与执行本指令的任务在不同任务内的程序时的动作示例。

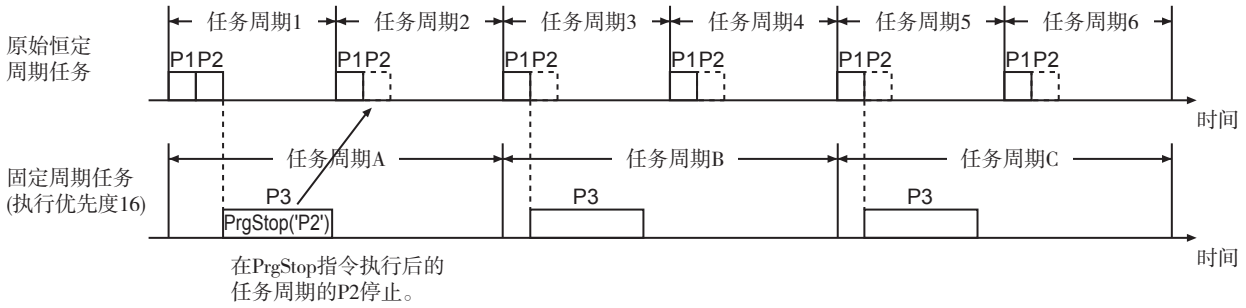
● 发出优先度低于我的任务的其它任务内程序的停止指令时

- 原始恒定周期任务中有P1、固定周期任务中有P2、P3共三个程序。
- 固定周期任务的任务周期A中，P3得到执行。
- 在原始恒定周期任务的任务周期2的P1内，指定P3，执行PrgStop指令。
- 在PrgStop指令执行后执行的固定周期任务中任务周期B的P3停止。
- 之后，重新指定P3，即使不执行PrgStop指令，P3仍会保持停止状态。



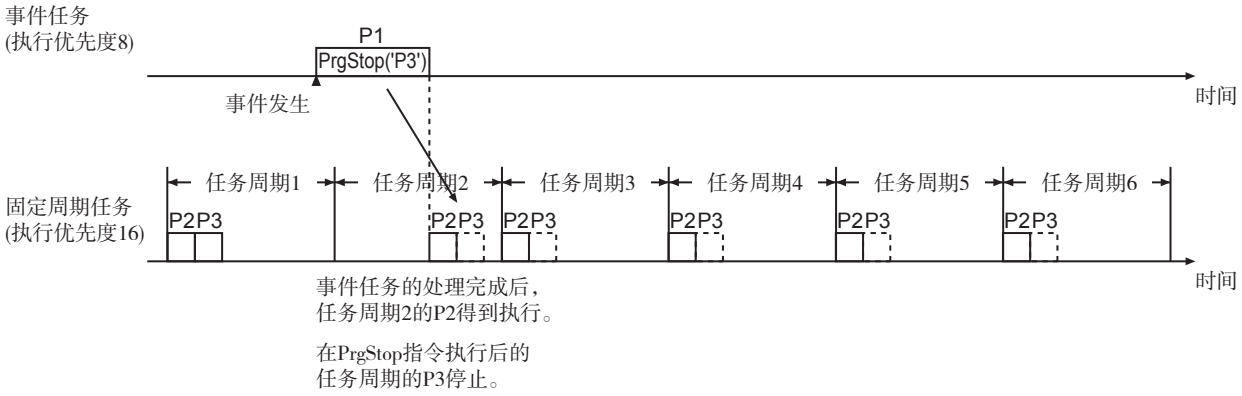
● 发出优先度高于我的任务的其它任务内程序的停止指令时

- 原始恒定周期任务中有P1、P2，固定周期任务中有P3共三个程序。
- 原始恒定周期任务的任务周期1中，P2得到执行。
- 在固定周期任务的任务周期A的P3内，指定P2，执行PrgStop指令。
- 在PrgStop指令执行后执行的原始恒定周期任务中任务周期2的P2停止。
- 之后，重新指定P2，即使不执行PrgStop指令，P2仍会保持停止状态。



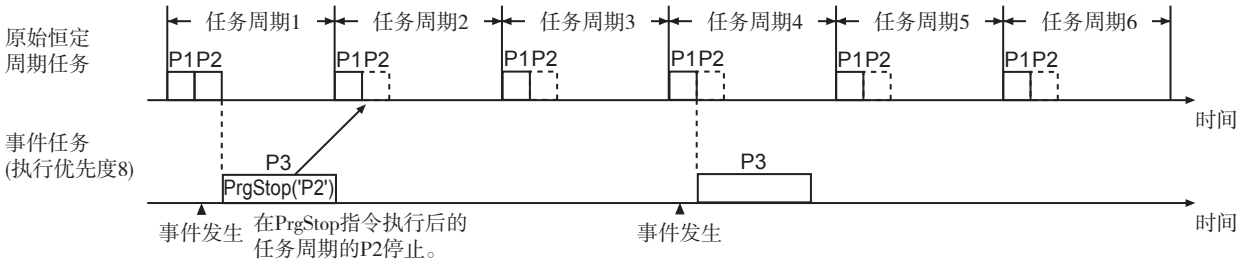
● 从事件任务中发出优先度较低的任务内程序的停止指令时

- 事件任务(执行优先度8)中有P1，固定周期任务(执行优先度16)中有P2、P3共三个程序。
- 固定周期任务的任务周期1中，P3得到执行。
- 在事件任务内，指定P3，执行PrgStop指令。
- 执行事件任务后，固定周期任务的任务周期2的P2、P3将在事件任务处理完成后执行。
- 固定周期任务中任务周期2的P3在PrgStop指令执行之后，因此将停止。
- 之后，重新指定P3，即使不执行PrgStop指令，P3仍会保持停止状态。



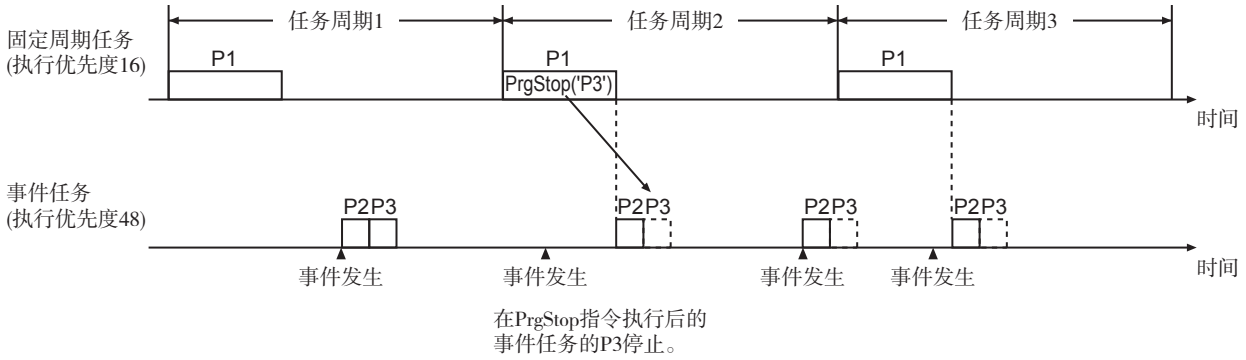
● 从事件任务中发出优先度较高任务内程序的停止指令时

- 原始恒定周期任务中有P1、P2，事件任务中有P3共三个程序。
- 原始恒定周期任务的任务周期1中，P2得到执行。
- 在事件任务内，指定P2，执行PrgStop指令。
- 在PrgStop指令执行后执行的原始恒定周期任务中任务周期2的P2停止。
- 之后，重新指定P2，即使不执行PrgStop指令，P2仍会保持停止状态。



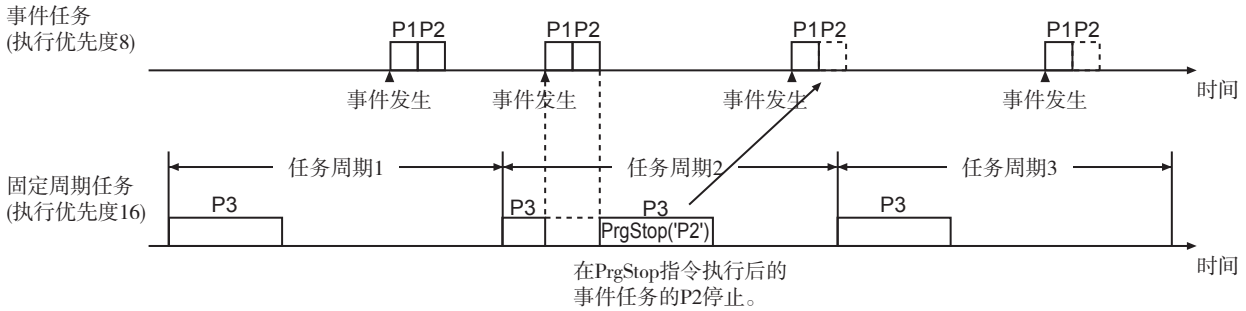
● 从固定周期任务中发出优先度较低事件任务内程序的停止指令时

- 固定周期任务(执行优先度16)中有P1，事件任务(执行优先度48)中有P2、P3共三个程序。
- 事件任务的P3得到执行。
- 在固定周期任务内，指定P3，执行PrgStop指令。
- 在PrgStop指令执行后执行的事件任务的P3停止。
- 之后，重新指定P3，即使不执行PrgStop指令，P3仍会保持停止状态。



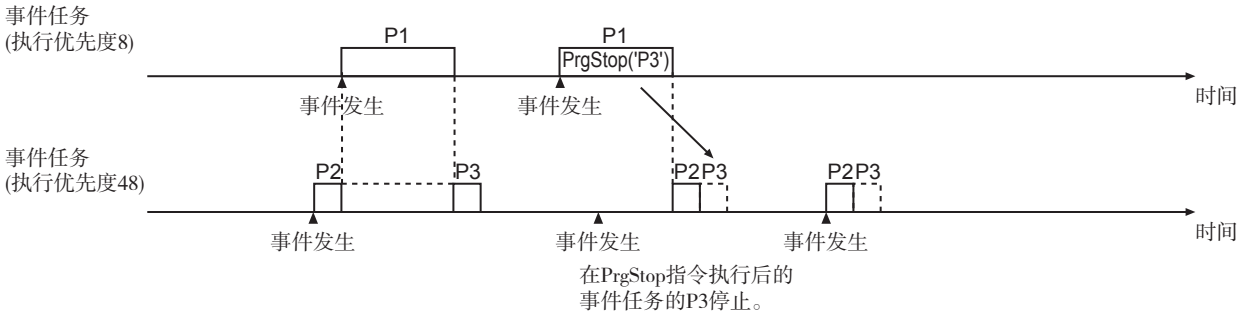
● 从固定周期任务中发出优先度较高事件任务内程序的停止指令时

- 事件任务(执行优先度8)中有P1和P2，固定周期任务(执行优先度16)中有P3共三个程序。
- 事件任务的P2得到执行。
- 在固定周期任务内，指定P2，执行PrgStop指令。
- 在PrgStop指令执行后执行的事件任务的P2停止。
- 之后，重新指定P2，即使不执行PrgStop指令，P2仍会保持停止状态。



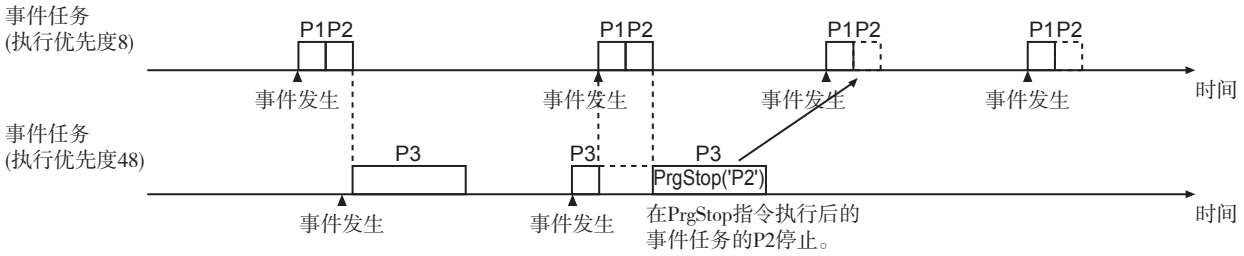
● 从事件任务中发出优先度较低事件任务内程序的停止指令时

- 事件任务(执行优先度8)中有P1，事件任务(执行优先度48)中有P2、P3共三个程序。
- 事件任务(执行优先度48)的P3得到执行。
- 在事件任务(执行优先度8)内，指定P3，执行PrgStop指令。
- 在PrgStop指令执行后执行的事件任务(执行优先度48)的P3停止。
- 之后，重新指定P3，即使不执行PrgStop指令，P3仍会保持停止状态。



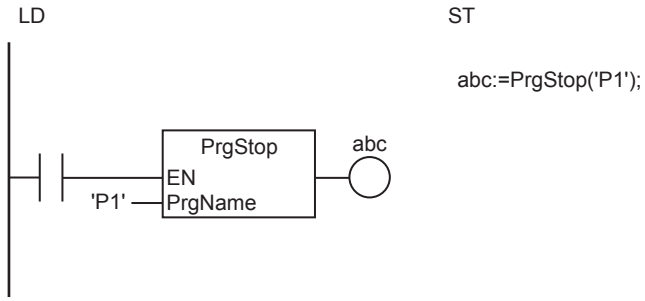
● 从事件任务中发出优先度较高事件任务内程序的停止指令时

- 事件任务(执行优先度8)中有P1和P2，事件任务(执行优先度48)中有P3共三个程序。
- 事件任务(执行优先度8)的P2得到执行。
- 在事件任务(执行优先度48)内，指定P2，执行PrgStop指令。
- 在PrgStop指令执行后执行的事件任务(执行优先度8)的P2停止。
- 之后，重新指定P2，即使不执行PrgStop指令，P2仍会保持停止状态。



记述示例

发出程序 “P1”的停止指令时的记述示例如下所示。



参考

- 需通过用户程序执行指定程序时，请使用 “PrgStart指令(P.2-870)”。
- 需通过用户程序确认指定程序的状态时，
  请使用 “PrgStatus指令(P.2-896)”。

使用注意事项

- 指定了已处于停止状态的程序后执行本指令时，不会发生异常。
- 执行PrgStart指令，在相应程序执行前执行了PrgStop指令时，相应程序将不会执行。
- 执行PrgStop指令，在相应程序停止前执行了PrgStart指令时，相应程序将不会停止。
- 输入变量中的 “Execute”指令，即使执行时间超过任务周期，仍将继续处理直至最后。发出包含此类指令的程序的停止指令时，请事先确认该指令的输出变量 “Busy”的值为 FALSE，该指令处于停止状态。
- NX_DOutTimeStamp指令及NX_AryDOutTimeStamp指令可能会跨越多个任务进行处理。发出包含这些指令的程序的停止指令时，请事先确认这些指令的输入变量 “Enable”的值为FALSE。
- 控制器的动作模式切换为运行模式后，各程序的动作取决于Sysmac Studio各程序的 “初始状态”设定。即切换运行模式前执行的PrgStart指令及PrgStop指令，在切换运行模式后无效。
- 指定了与执行本指令的任务不同的任务内的程序时，指定程序的停止时间因任务双方的执行优先度而异，控制器可能会发生意外动作。通过分配了指定程序的任务的开头程序执行本指令，可在执行了本指令的任务周期中切实停止指定程序。
- 请确认指定程序的以下内容后再执行本指令。
  - 运动控制指令未在执行中。
  - 输入变量中的“Execute”指令，即执行时间超过任务周期，处理也会持续到最后的指令未在执行中。
  - 时间戳指令非等待指定时间的状态。
- 指定程序停止时，程序的输出不会复位。将保持停止前的值。需在程序停止时复位输出时，请在事先指定的程序内执行基于主站控制的复位。
- 本指令指定的程序停止后，程序内带输入变量 “Execute”的FB型指令仍将继续处理直至最后。
- 本指令指定的程序停止后，程序内的运动控制指令仍将继续处理直至最后。
- 以下情况时会发生异常。“Out”为FALSE。
  - “PrgName”指定的程序不存在时。

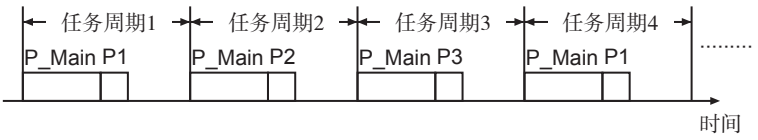
版本相关信息

本指令可用于CPU单元Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

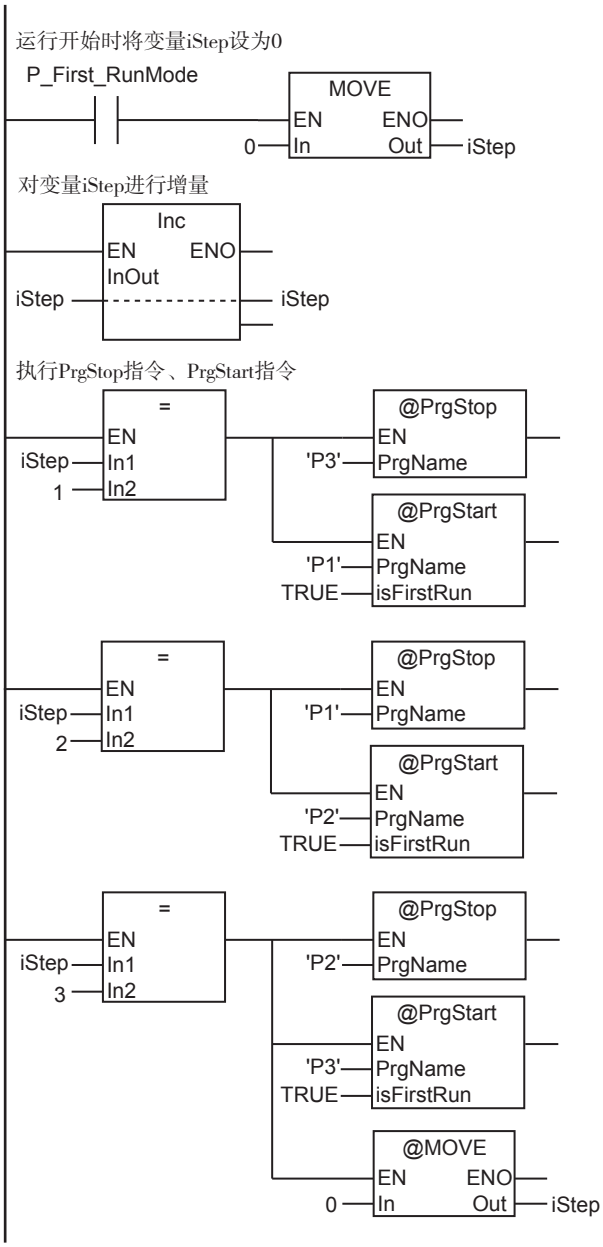
● 按任务周期依次执行三个程序的示例

有P1、P2及P3三个程序。按任务周期依次重复执行这些程序。发出指令执行和停止三个程序的是P_Main程序。



LD

名称	数据类型	初始值	注释
iStep	DINT	0	执行程序的编号



ST

名称	数据类型	初始值	注释
iStep	DINT	0	执行程序的编号

```
// 运行开始时将变量iStep设为0
IF P_First_RunMode THEN
  iStep:=0;
END_IF;

// 对变量iStep进行增量
iStep:=iStep+1;

// 执行PrgStop指令、PrgStart指令
IF iStep = 1 THEN
  PrgStop( ‘P3’ );
  PrgStart( ‘P1’ ,TRUE);
ELSIF iStep = 2 THEN
  PrgStop( ‘P1’ );
  PrgStart( ‘P2’ ,FALSE);
ELSIF iStep = 3 THEN
  PrgStop( ‘P2’ );
  PrgStart( ‘P3’ ,TRUE);
  iStep:=0;
END_IF;
```

- **多个程序中，在下一次运行时只执行指定程序的示例**

  指定需在下一次运行时执行的程序后，切断控制器电源。然后接通电源，只执行预先指定的程序。

程序、模块、模块构成

程序分为Program1至Program8共8种。
各程序与ModuleA至ModuleE的5种模块具有以下从属关系。

模块	包含的程序
ModuleA	Program1
ModuleB	Program2
ModuleC	Program3、Program4
ModuleD	Program5、Program6、Program7
ModuleE	Program8

执行的程序以模块为单位进行指定。此外，模块构成表示执行模块的组合。
例如指定了执行ModuleA和ModuleC的模块构成时，将执行Program1、Program3、Program4三个程序。

执行的模块构成的指定方法

模块构成在设定文件中以文本数据的形式进行记述。设定文件的文件名为Config.txt，事先保存在SD存储卡的根目录中。设定文件可记述多个模块构成。
关闭电源前，使用触摸屏，从设定文件的记述中指定下一次运行时要执行的模块构成编号。

设定文件的格式

设定文件的格式如下所述。

行	记述内容
第1行	模块构成数
第2行以后	模块构成编号、ModuleA的执行标志*1、ModuleB的执行标志、ModuleC的执行标志、ModuleD的执行标志、ModuleE的执行标志

*1　记述执行时为TRUE，不执行时为FALSE。

设定文件的记述示例如下所示。

```
3
Config1, TRUE, TRUE, TRUE, FALSE, FALSE
Config2, TRUE, TRUE, FALSE, TRUE, FALSE
Config3, TRUE, TRUE, TRUE, FALSE, TRUE
```

该设定文件中记述了Config1、Config2及Config3等三个模块构成。其中，Config1的模块构成为执行ModuleA、ModuleB及ModuleC，不执行ModuleD及ModuleE。

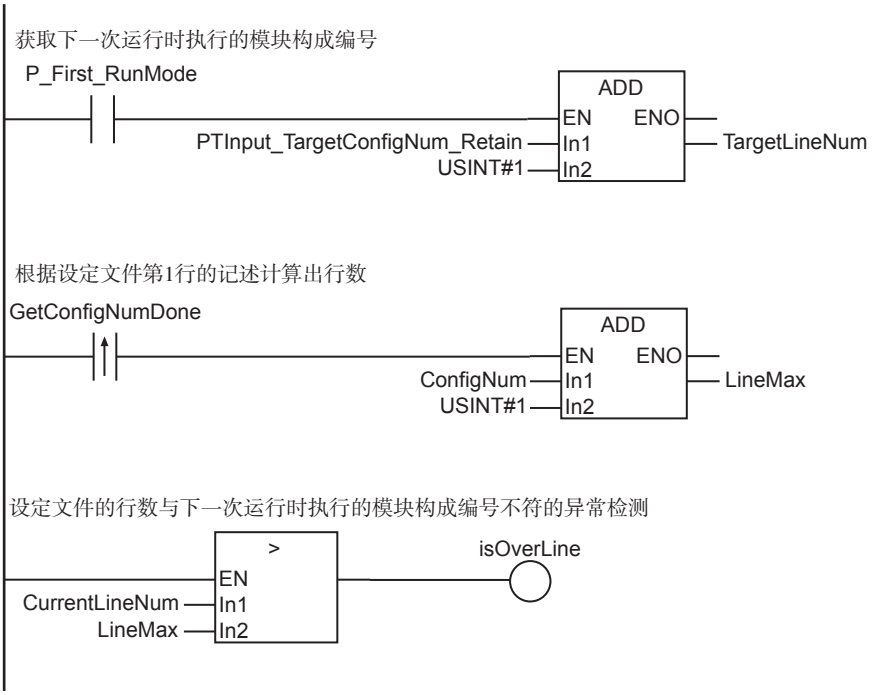
数据类型定义

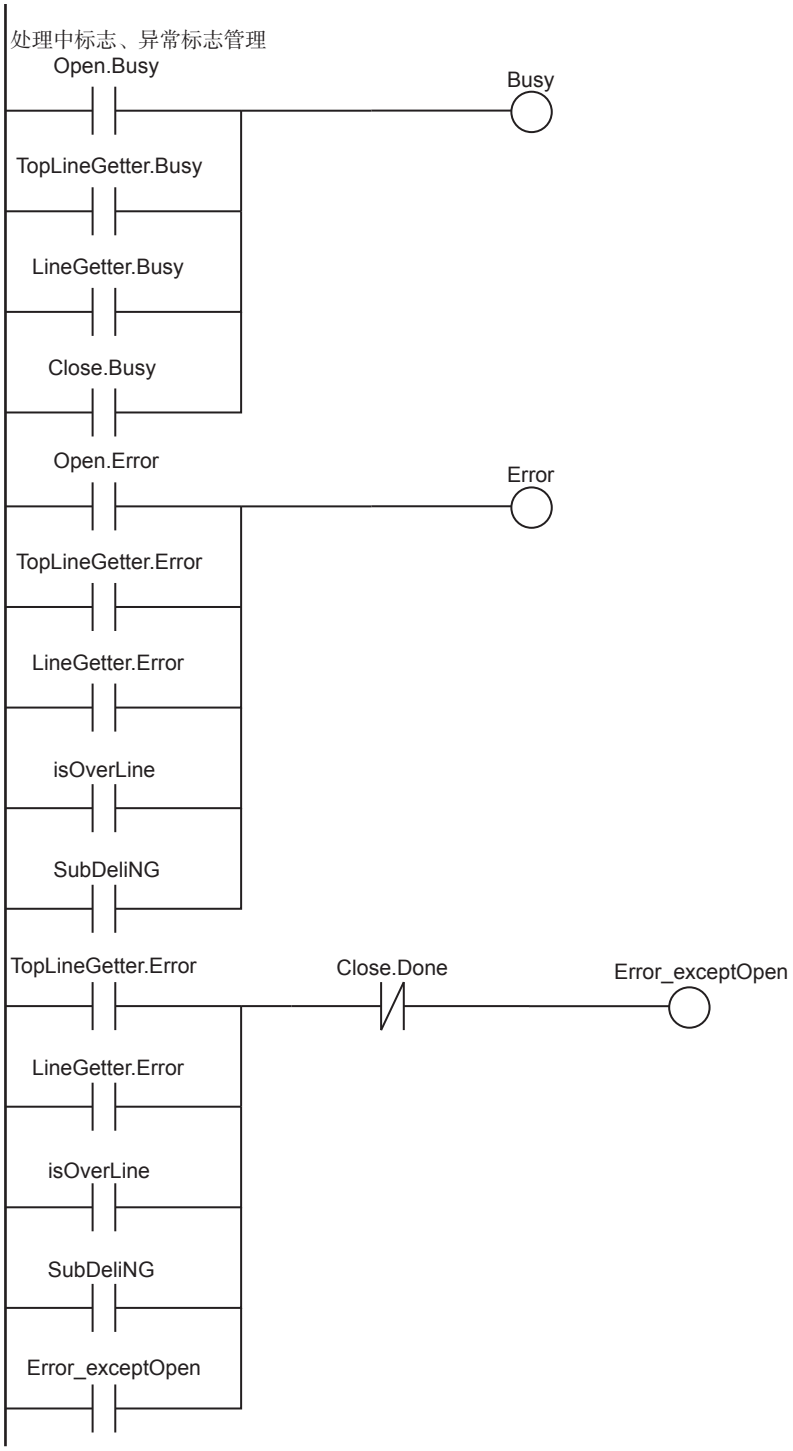
如下所述，定义结构体型myConfig。

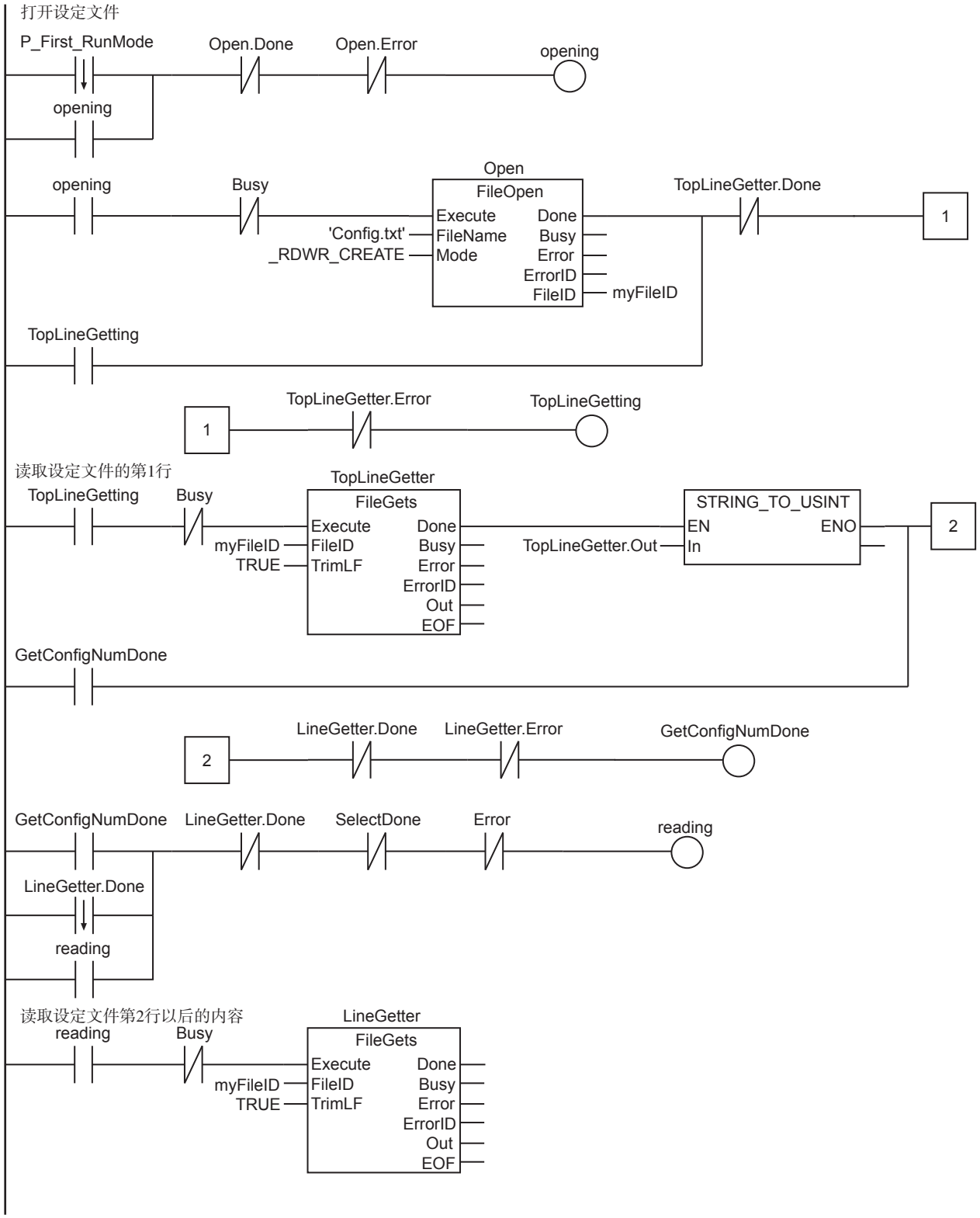
结构体型		名称	数据类型	偏置种类	注释
	▼	myConfig	STRUCT	NJ	模块构成
		configName	STRING[32]		模块构成名称
		moduleA	BOOL		ModuleA的执行标志
		moduleB	BOOL		ModuleB的执行标志
		moduleC	BOOL		ModuleC的执行标志
		moduleD	BOOL		ModuleD的执行标志
		moduleE	BOOL		ModuleE的执行标志

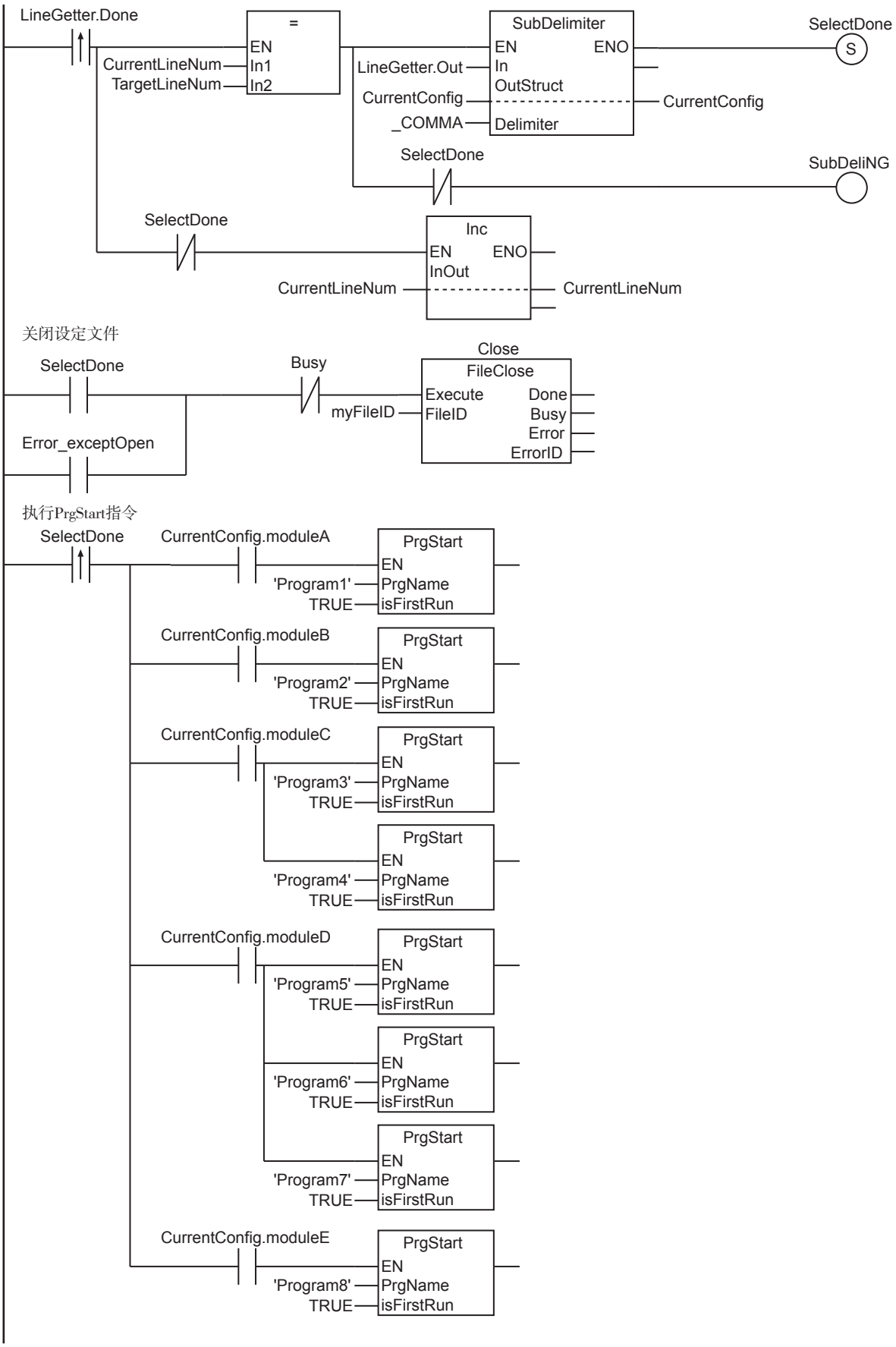
LD

名称	数据类型	初始值	保持	注释
Open	FileOpen		☐	FileOpen指令的实例
TopLineGetter	FileGets		☐	FileGets指令的实例
LineGetter	FileGets		☐	FileGets指令的实例
Close	FileClose		☐	FileClose指令的实例
PTInput_TargetConfigNum_Retain	USINT	0	☑	下一次运行时执行的模块构成编号
CurrentLineNum	USINT	1	☐	设定文件的当前读取中的行
TargetLineNum	USINT	0	☐	“CurrentConfig”的设定文件内的行
ConfigNum	USINT	1	☐	设定文件第1行中记述的数量
LineMax	USINT	3	☐	根据 “ConfigNum”计算出的设定文件的行数
isOverLine	BOOL	FALSE	☐	“LineMax”值小于“PTInput_TargetConfigNum_Retain”值的异常发生标志
Busy	BOOL	FALSE	☐	处理中标志
SubDeliNG	BOOL	FALSE	☐	“CurrentConfig”的读取异常结束标志
Error	BOOL	FALSE	☐	异常标志
opening	BOOL	FALSE	☐	设定文件打开的执行标志
myFileID	DWORD	0	☐	设定文件的文件ID
TopLineGetting	BOOL	FALSE	☐	“ConfigNum”的读取执行标志
GetConfigNumDone	BOOL	FALSE	☐	“ConfigNum”的读取完成标志
SelectDone	BOOL	FALSE	☐	“CurrentConfig”的读取完成标志
reading	BOOL	FALSE	☐	设定文件第2行以后的读取执行标志
CurrentConfig	myConfig	(configName:='', moduleA:=FALSE, moduleB:=FALSE, moduleC:=FALSE, moduleD:=FALSE)	☐	下一次运行时执行的模块构成
Error_exceptOpen	BOOL	FALSE	☐	发生异常时的设定文件关闭执行标志


获取下一次运行时执行的模块构成编号
P_First_RunMode
ADD
EN
ENO
PTInput_TargetConfigNum_Retain
In1
TargetLineNum
USINT#1
In2
根据设定文件第1行的记述计算出行数
GetConfigNumDone
ADD
EN
ENO
ConfigNum
In1
LineMax
USINT#1
In2
设定文件的行数与下一次运行时执行的模块构成编号不符的异常检测
>
isOverLine
EN
CurrentLineNum
In1
LineMax
In2


处理中标志、异常标志管理
Open.Busy
Busy
TopLineGetter.Busy
LineGetter.Busy
Close.Busy
Open.Error
Error
TopLineGetter.Error
LineGetter.Error
isOverLine
SubDeliNG
TopLineGetter.Error
Close.Done
Error_exceptOpen
LineGetter.Error
isOverLine
SubDeliNG
Error_exceptOpen


打开设定文件
P_First_RunMode
Open.Done
Open.Error
opening
opening
opening
Busy
Open
FileOpen
Execute
FileName
Mode
'Config.txt'
_RDWR_CREATE
Done
Busy
Error
ErrorID
FileID
myFileID
TopLineGetter.Done
1
TopLineGetting
1
TopLineGetter.Error
TopLineGetting
读取设定文件的第1行
TopLineGetting
Busy
TopLineGetter
FileGets
Execute
FileID
TrimLF
myFileID
TRUE
Done
Busy
Error
ErrorID
Out
EOF
TopLineGetter.Out
STRING_TO_USINT
EN
In
ENO
2
GetConfigNumDone
2
LineGetter.Done
LineGetter.Error
GetConfigNumDone
GetConfigNumDone
LineGetter.Done
SelectDone
Error
reading
LineGetter.Done
reading
读取设定文件第2行以后的内容
reading
Busy
LineGetter
FileGets
Execute
FileID
TrimLF
myFileID
TRUE
Done
Busy
Error
ErrorID
Out
EOF


LineGetter.Done
=
EN
In1
In2
CurrentLineNum
TargetLineNum
SubDelimiter
EN
ENO
In
OutStruct
Delimiter
LineGetter.Out
CurrentConfig
_COMMA
CurrentConfig
SelectDone
S
SubDeliNG
SelectDone
Inc
EN
ENO
InOut
CurrentLineNum
CurrentLineNum
关闭设定文件
SelectDone
Busy
Close
FileClose
Execute
FileID
Done
Busy
Error
ErrorID
myFileID
Error_exceptOpen
执行PrgStart指令
SelectDone
CurrentConfig.moduleA
PrgStart
EN
PrgName
isFirstRun
'Program1'
TRUE
CurrentConfig.moduleB
'Program2'
CurrentConfig.moduleC
'Program3'
'Program4'
CurrentConfig.moduleD
'Program5'
'Program6'
'Program7'
CurrentConfig.moduleE
'Program8'

ST

名称	数据类型	初始值	保持	注释
Open	FileOpen		☐	FileOpen指令的实例
TopLineGetter	FileGets		☐	FileGets指令的实例
LineGetter	FileGets		☐	FileGets指令的实例
Close	FileClose		☐	FileClose指令的实例
PTInput_TargetConfigNum_Retain	USINT	0	☑	下一次运行时执行的模块构成编号
CurrentLineNum	USINT	1	☐	设定文件的当前读取中的行
TargetLineNum	USINT	0	☐	“CurrentConfig”的设定文件内的行
ConfigNum	USINT	1	☐	设定文件第1行中记述的数量
LineMax	USINT	3	☐	根据 “ConfigNum”计算出的设定文件的行数
isOverLine	BOOL	FALSE	☐	“LineMax”值小于“PTInput_TargetConfigNum_Retain”值的异常发生标志
Busy	BOOL	FALSE	☐	处理中标志
SubDeliNG	BOOL	FALSE	☐	“CurrentConfig”的读取异常结束标志
Error	BOOL	FALSE	☐	异常标志
opening	BOOL	FALSE	☐	设定文件打开的执行标志
myFileID	DWORD	0	☐	设定文件的文件ID
TopLineGetting	BOOL	FALSE	☐	“ConfigNum”的读取执行标志
GetConfigNumDone	BOOL	FALSE	☐	“ConfigNum”的读取完成标志
SelectDone	BOOL	FALSE	☐	“CurrentConfig”的读取完成标志
reading	BOOL	FALSE	☐	设定文件第2行以后的读取执行标志
CurrentConfig	myConfig	(configName:='', moduleA:=FALSE, moduleB:=FALSE, moduleC:=FALSE, moduleD:=FALSE)	☐	下一次运行时执行的模块构成
Error_exceptOpen	BOOL	FALSE	☐	发生异常时的设定文件关闭执行标志
R_GetConfigNumDone	R_TRIG		☐	R_TRIG指令的实例
RS_1	RS		☐	RS指令的实例
RS_2	RS		☐	RS指令的实例
SecondCycle	F_TRIG		☐	F_TRIG指令的实例
RS_3	RS		☐	RS指令的实例
ConvertDone	BOOL	FALSE	☐	设定文件第1行中记述的字符转换成数字的转换完成标志
RS_4	RS		☐	RS指令的实例
F_LineGetterDone	F_TRIG		☐	F_TRIG指令的实例
R_LineGetterDone	R_TRIG		☐	R_TRIG指令的实例
isTargetLine	BOOL	FALSE	☐	表示读取中的行是下一次运行时执行的模块构成行的标志
SubDeliCondition	BOOL	FALSE	☐	从模块构成的记述向 “CurrentConfig”展开的执行标志

名称	数据类型	初始值	保持	注释
RS_5	RS		☐	RS指令的实例
SubDeliDone	BOOL	FALSE	☐	从模块构成的记述向 “CurrentConfig” 展开的完成标志
R_SelectDone	R_TRIG		☐	R_TRIG指令的实例

```
// 获取下一次运行时执行的模块构成编号
IF P_First_RunMode THEN
  TargetLineNum := PTInput_TargetConfigNum_Retain + USINT#1;
END_IF;

// 根据设定文件第1行的记述计算出行数
R_GetConfigNumDone(Clk:=GetConfigNumDone);
IF R_GetConfigNumDone.Q THEN
  LineMax := ConfigNum + USINT#1;
END_IF;

// 设定文件的行数与下一次运行时执行的模块构成编号不符的异常检测
isOverLine := (CurrentLineNum > LineMax);

// 处理中标志、异常标志管理
Busy := Open.Busy OR TopLineGetter.Busy OR LineGetter.Busy OR Close.Busy;

Error := Open.Error OR TopLineGetter.Error OR LineGetter.Error OR isOverLine OR SubDeliNG;

RS_1(Set:= (TopLineGetter.Error OR LineGetter.Error OR isOverLine OR SubDeliNG), reset1 := Close.Done, Q1 =>
Error_exceptOpen);

// 打开设定文件
SecondCycle(Clk:=P_First_RunMode);
RS_2(Set := SecondCycle.Q, reset1:=(Open.Done OR Open.Error), Q1 => opening);
Open(Execute:=(opening & NOT(Busy)), FileName :='Config.txt', FileID => myFileID);
RS_3(Set := Open.Done, Reset1:=(TopLineGetter.Done OR TopLineGetter.Error), Q1=>TopLineGetting);

// 读取设定文件的第1行
TopLineGetter(Execute :=(TopLineGetting & NOT(Busy)), FileID := myFileID, TrimLF := TRUE);
ConfigNum := STRING_TO_USINT(EN:= TopLineGetter.Done, IN:=TopLineGetter.Out, ENO=>ConvertDone);
RS_4(Set := ConvertDone, Reset1:=(LineGetter.Done OR LineGetter.Error), Q1=>GetConfigNumDone);
F_LineGetterDone(Clk:=LineGetter.Done);
RS_5(Set := (GetConfigNumDone OR F_LineGetterDone.Q), Reset1:=(LineGetter.Done OR SelectDone OR Error), Q1=>reading);

// 读取设定文件第2行以后的内容
LineGetter(Execute:=(reading & NOT(Busy)), FileID:=myFileID, TrimLF := TRUE);
R_LineGetterDone(Clk:=LineGetter.Done);
isTargetLine := (CurrentLineNum = TargetLineNum);
SubDeliCondition := (R_LineGetterDone.Q & isTargetLine);
SubDelimiter(EN := SubDeliCondition, In := LineGetter.Out, OutStruct := CurrentConfig, Delimiter := _COMMA, ENO =>
SubDeliDone);
IF SubDeliDone THEN
  SelectDone := TRUE;
END_IF;
SubDeliNG := (SubDeliCondition & NOT(SubDeliDone));
Inc(EN := (R_LineGetterDone.Q & NOT(SelectDone)), InOut:= CurrentLineNum);
```

```
// 关闭设定文件
Close(Execute := ((SelectDone OR Error_exceptOpen) & NOT(Busy)), FileID := myFileID);

// 执行PrgStart指令
R_SelectDone(Clk:=SelectDone);
//moduleA
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleA), PrgName :='Program1', isFirstRun:=TRUE);
//moduleB
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleB), PrgName :='Program2', isFirstRun:=TRUE);
//moduleC
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleC), PrgName :='Program3', isFirstRun:=TRUE);
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleC), PrgName :='Program4', isFirstRun:=TRUE);
//moduleD
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleD), PrgName :='Program5', isFirstRun:=TRUE);
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleD), PrgName :='Program6', isFirstRun:=TRUE);
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleD), PrgName :='Program7', isFirstRun:=TRUE);
//moduleE
PrgStart(EN := (R_SelectDone.Q & CurrentConfig.moduleE), PrgName :='Program8', isFirstRun:=TRUE);
```

PrgStatus

读取指定程序的状态。

指令	名称	FB/FUN	图形表现	ST表现
PrgStatus	程序状态读取	FUN	(@)PrgStatus EN ENO PrgName Out	Out:=PrgStatus(PrgName);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
PrgName	程序名	输入	要指定的程序名	最大128字节(127个半角英数字字符+结尾NULL字符)	–	*1
Out	程序的状态	输出	下一次执行时的程序状态 TRUE ：执行 FALSE：停止	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编译时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
PrgName																				○
Out	○																			

功能

读取程序名 “PrgName”指定的程序在下一次执行时的状态。指定程序在下一次执行的时间内执行时，“Out”的值变为TRUE。指定程序在下一次执行的时间内停止时，“Out”的值变为FALSE。
下一次执行的时间内执行及停止具有以下含义。

程序的状态	含义
下一次执行的时间内执行	• Sysmac Studio相应程序的 “初始状态”中设定了 “启动”。 • 对相应程序执行了PrgStart指令。
下一次执行的时间内停止	• Sysmac Studio相应程序的 “初始状态”中设定了 “停止”。 • 对相应程序执行了PrgStop指令。

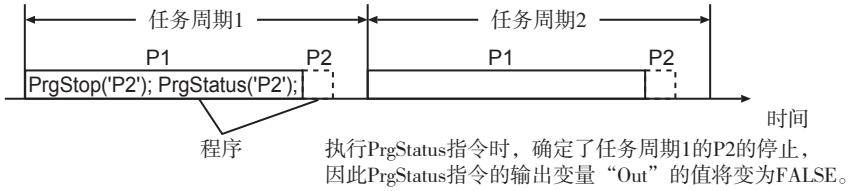
指定程序与执行本指令的任务在同一任务或不同任务内均可。

动作示例

下面介绍本指令在几种情况下的动作示例。

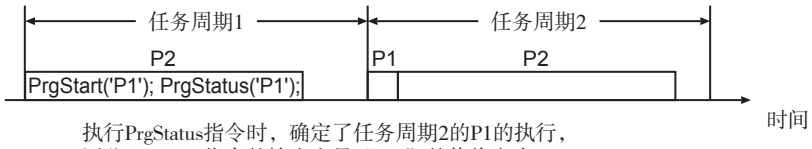
● 读取我的任务中PrgStatus指令之后的程序状态时

- 同一任务内有P1及P2两个程序。
- 在任务周期1的P1内，指定P2，执行PrgStop指令。
- 然后，在任务周期1的P1内，指定P2，执行PrgStatus指令。
- 由于已确定任务周期1的P2的停止，因此PrgStatus指令的输出变量 “Out”的值将变为FALSE。



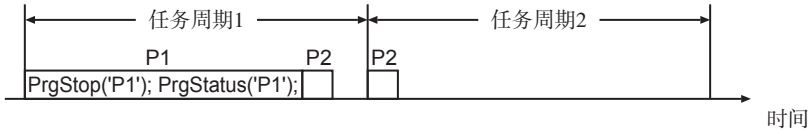
● 读取我的任务中PrgStatus指令之前的程序状态时

- 同一任务内有P1及P2两个程序。
- 在任务周期1的P2内，指定P1，执行PrgStart指令。
- 然后，在任务周期1的P2内，指定P1，执行PrgStatus指令。
- 由于已确定任务周期2的P1的执行，因此PrgStatus指令的输出变量 “Out”的值将变为TRUE。



● 读取包含PrgStatus指令的程序状态时

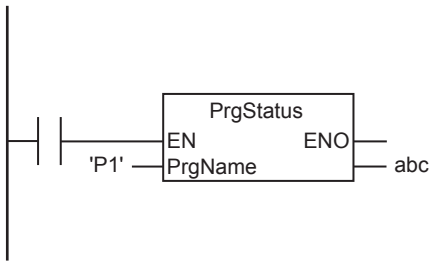
- 在任务周期1的P1内，指定P1，执行PrgStop指令。
- 然后，在任务周期1的P1内，指定P1，执行PrgStatus指令。
- 由于已确定任务周期2的P1的停止，因此PrgStatus指令的输出变量 “Out”的值将变为FALSE。



记述示例

读取程序 “P1”的状态时的记述示例如下所示。

LD



ST

```
abc:=PrgStatus('P1');
```

参考

- 需通过用户程序执行指定程序时，请使用 “PrgStart指令(P.2-870)”。
- 需通过用户程序停止指定程序时，请使用 “PrgStop指令(P.2-878)”。

使用注意事项

- 以下情况时会发生异常。“Out”为FALSE。
  - “PrgName”指定的程序不存在时。

版本相关信息

本指令可用于CPU单元Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

有P1、P2及P3三个程序。根据触摸屏的指示，切换执行的程序。

触摸屏的规格

本程序以控制器连接触摸屏为前提。
触摸屏上有以下指示灯。

指示灯名称	含义
P1执行中指示灯	P1执行中时点亮。
P2执行中指示灯	P2执行中时点亮。
P3执行中指示灯	P3执行中时点亮。

此外，触摸屏上有以下按钮。

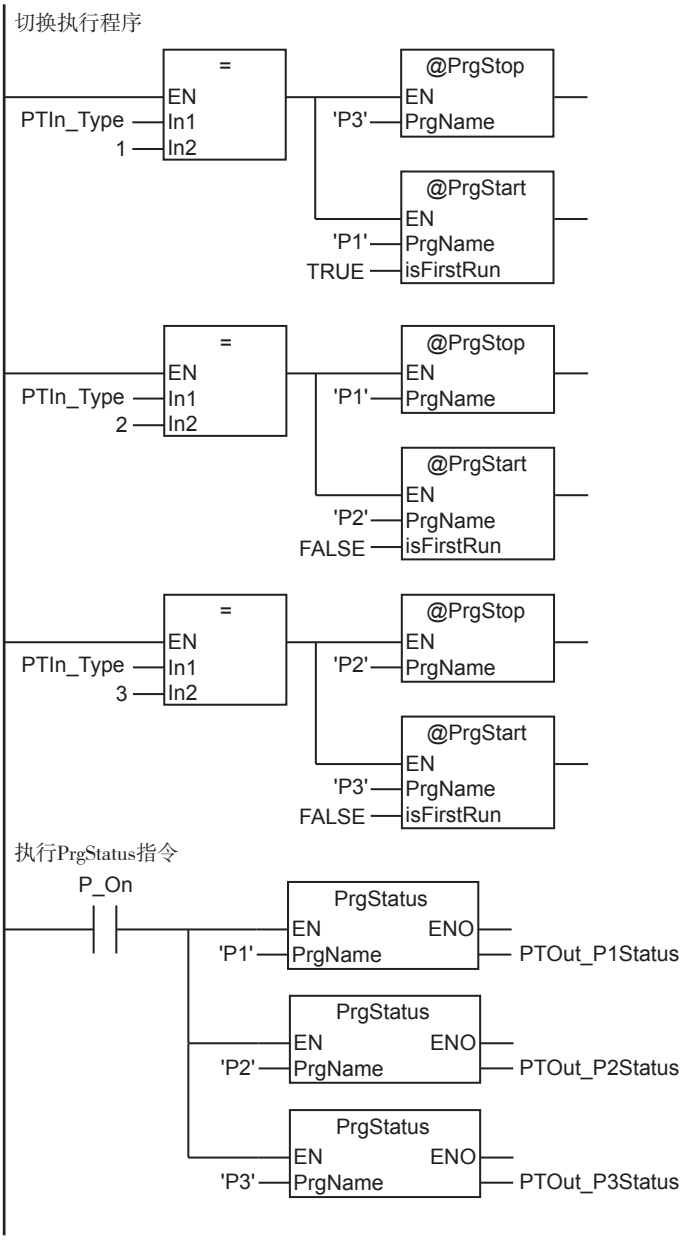
按钮名称	按下时的动作
执行程序切换按钮	每次按下按钮时，执行的程序按照P1→P2→P3→P1··的顺序依次切换。

全局变量

名称	数据类型	初始值	注释
PTIn_Type	INT	0	执行程序切换按钮输入
PTOut_P1Status	BOOL	FALSE	P1执行中指示灯输出
PTOut_P2Status	BOOL	FALSE	P2执行中指示灯输出
PTOut_P3Status	BOOL	FALSE	P3执行中指示灯输出

LD

外部变量	名称	数据类型	注释
	PTIn_Type	INT	执行程序切换按钮输入
	PTOut_P1Status	BOOL	P1执行中指示灯输出
	PTOut_P2Status	BOOL	P2执行中指示灯输出
	PTOut_P3Status	BOOL	P3执行中指示灯输出



ST

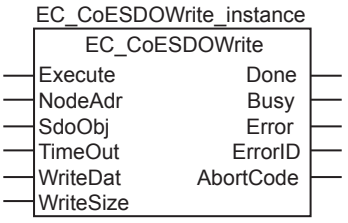
外部变量	名称	数据类型	注释
	PTIn_Type	INT	执行程序切换按钮输入
	PTOut_P1Status	BOOL	P1执行中指示灯输出
	PTOut_P2Status	BOOL	P2执行中指示灯输出
	PTOut_P3Status	BOOL	P3执行中指示灯输出

```
// 切换执行程序
IF PTIn_Type = 1 THEN
  PrgStop('P3');
  PrgStart('P1',TRUE);
ELSIF PTIn_Type = 2 THEN
  PrgStop('P1');
  PrgStart('P2',FALSE);
ELSIF PTIn_Type = 3 THEN
  PrgStop('P2');
  PrgStart('P3',FALSE);
END_IF;

// 执行PrgStatus指令
IF P_On THEN
  PTOut_P1Status:=PrgStatus('P1');
  PTOut_P2Status:=PrgStatus('P2');
  PTOut_P3Status:=PrgStatus('P3');
END_IF;
```

EtherCAT通信指令

EC_CoESDOWrite

将值写入EtherCAT网络上指定从站的CoE (*)对象。

指令	名称	FB/FUN	图形表现	ST表现
EC_CoESDOWrite	CoE SDO写入	FB		EC_CoESDOWrite_instance(Execute, NodeAdr, SdoObj, TimeOut, WriteDat, WriteSize, Done, Busy, Error, ErrorID, AbortCode);

* CAN application protocol over EtherCAT的简称。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
NodeAdr	从站节点地址	输入	待访问的从站的节点地址	1 ~ 512*1	–	–
SdoObj	SDO参数		SDO参数	–		
TimeOut	超时时间		0 ：2.0s 1 ~ 65535：0.1 ~ 6553.5s	遵从数据类型	0.1s	20 (2.0s)
WriteDat	写入数据		写入数据		–	–
WriteSize	写入数据大小		写入数据大小*2	1 ~ 2048	字节	
AbortCode	Abort代码	输出	CoE规定的SDO访问的响应代码 0：正常结束	遵从数据类型	–	–

*1 NJ系列CPU单元时为 “1 ~ 192”。

*2 写入数据为BOOL型或BOOL型数组等时，写入数据大小可能小于1字节。此时，请将 “WriteSize”的值设为1。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
NodeAdr							○													
SdoObj	结构体_sSDO_ACCESS 详情参阅功能说明																			
TimeOut							○													
WriteDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、结构体的1个结构要素、联合体的1个结构要素																			
WriteSize							○													
AbortCode				○																

功能

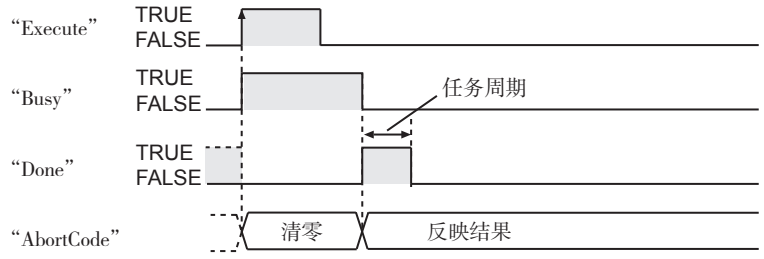
将数据写入具有从站节点地址 “NodeAdr”指定的节点的CoE的对象。
写入数据为 “WriteDat”的内容。写入数据大小由 “WriteSize”指定。
SDO参数由 “SdoObj”指定。

“SdoObj”的数据类型为结构体_sSDO_ACCESS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
SdoObj	SDO参数	SDO参数	_sSDO_ACCESS	–	–	–
Index	索引	CoE定义的对象词典的索引编号	UINT	1 ~ 65535	–	–
Subindex	子索引	CoE定义的对象词典的子索引编号	USINT	遵从数据类型		
IsComplete Access	Complete访问	指定SDO的Complete访问功能 TRUE：访问所有子索引的数据 FALSE：访问指定子索引的数据	BOOL			

写入完成后，仅在超时时间 “TimeOut”指定的时间内等待响应。响应保存至 “AbortCode” 。正常结束时 “AbortCode”的值为0。仅 “ErrorID”的值为16#1804(SDO中止响应)时，将值保存至 “AbortCode”。“AbortCode”的值和含义由从站规定。请参阅从站的手册。

时序图如下所示。指令的处理结束，“Busy”的值为FALSE时，将值保存至 “AbortCode”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信

参考

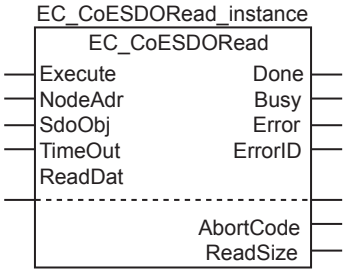
- EtherCAT 通信的详情请参阅 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。
- SDO Abort代码请参阅 “SDO Abort代码一览(P.A-187)”。

使用注意事项

- 请务必将传输至 “WriteDat”的输入参数设为变量。如果传输常数，编连时会发生异常。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - EtherCAT主站未处于可通信的状态时。
  - “NodeAdr”指定的从站不存在时。
  - “NodeAdr”指定的从站未处于可通信的状态时。
  - 从站返回异常响应时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

EC_CoESDORead

从EtherCAT网络上指定从站的CoE(*)对象中读取值。

指令	名称	FB/FUN	图形表现	ST表现
EC_CoESDORead	CoE SDO读取	FB		EC_CoESDORead_instance(Execute, NodeAdr, SdoObj, TimeOut, ReadDat, Done, Busy, Error, ErrorID, AbortCode, ReadSize);

* CAN application protocol over EtherCAT的简称。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
NodeAdr	从站节点地址	输入	待访问的从站的节点地址	1 ~ 512[*1]	–	–
SdoObj	SDO参数		SDO参数	–		
TimeOut	超时时间		0 ：2.0s 1 ~ 65535：0.1 ~ 6553.5s	遵从数据类型	0.1s	0 (2.0s)
AbortCode	Abort代码	输出	CoE规定的SDO访问的响应代码 0：正常结束	遵从数据类型	–	–
ReadSize	读取数据大小		读取后保存至 “ReadDat” 的数据大小[*2]		字节	
ReadDat	读取数据	输入输出	保存读取数据用缓存	遵从数据类型	–	–

*1 NJ系列CPU单元时为 “1 ~ 192”。

*2 读取数据为BOOL型或BOOL型数组等时，读取数据大小可能小于1字节。此时，“ReadSize” 的值为1。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
NodeAdr							○													
SdoObj	结构体_sSDO_ACCESS 详情参阅功能说明																			
TimeOut							○													
AbortCode				○																
ReadSize							○													
ReadDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、结构体的1个结构要素、联合体的1个结构要素																			

功能

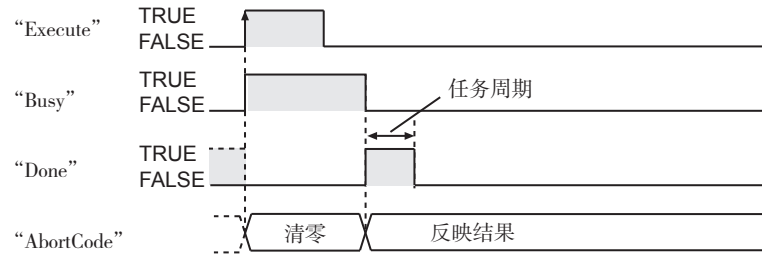
从具有站节点地址 “NodeAdr”指定的节点的CoE的对象读取数据。
读取的数据保存至 “ReadDat”。再将已保存的数据大小保存在 “ReadSize”中。仅保存成功时 “ReadSize”的值有效。
SDO参数由 “SdoObj”指定。

“SdoObj”的数据类型为结构体_sSDO_ACCESS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
SdoObj	SDO参数	SDO参数	_sSDO_ACCESS	–	–	–
Index	索引	CoE定义的对象词典的索引编号	UINT	1 ~ 65535	–	–
Subindex	子索引	CoE定义的对象词典的子索引编号	USINT	遵从数据类型		
IsComplete Access	Complete访问	指定SDO的Complete访问功能 TRUE：访问所有子索引的数据 FALSE：访问指定子索引的数据	BOOL			

读取完成后，仅在超时时间 “TimeOut”指定的时间内等待响应。响应保存至 “AbortCode” 。正常结束时 “AbortCode”的值为0。仅 “ErrorID”的值为16#1804(SDO中止响应)时，将值保存至 “AbortCode”。“AbortCode”的值和含义由从站规定。请参阅从站的手册。

时序图如下所示。指令的处理结束，“Busy”的值为FALSE时，将值保存至 “AbortCode”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信

参考

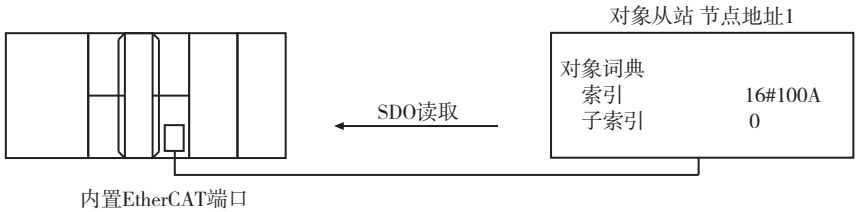
- EtherCAT 通信的详情请参阅 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。
- SDO Abort代码请参阅 “SDO Abort代码一览(P.A-187)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - EtherCAT主站未处于可通信的状态时。
  - “NodeAdr”指定的从站不存在时。
  - “NodeAdr”指定的从站未处于可通信的状态时。
  - 从站返回异常响应时。
  - 读取的数据大小大于 “ReadDat”的大小时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

示例程序

使用EtherCAT的SDO信息，通过欧姆龙制伺服驱动器1S系列读取软件版本。该从站的节点地址为1。
软件版本的对象索引为16#100A，子索引为0。
读取的值保存至STRING型变量VersionInfo。

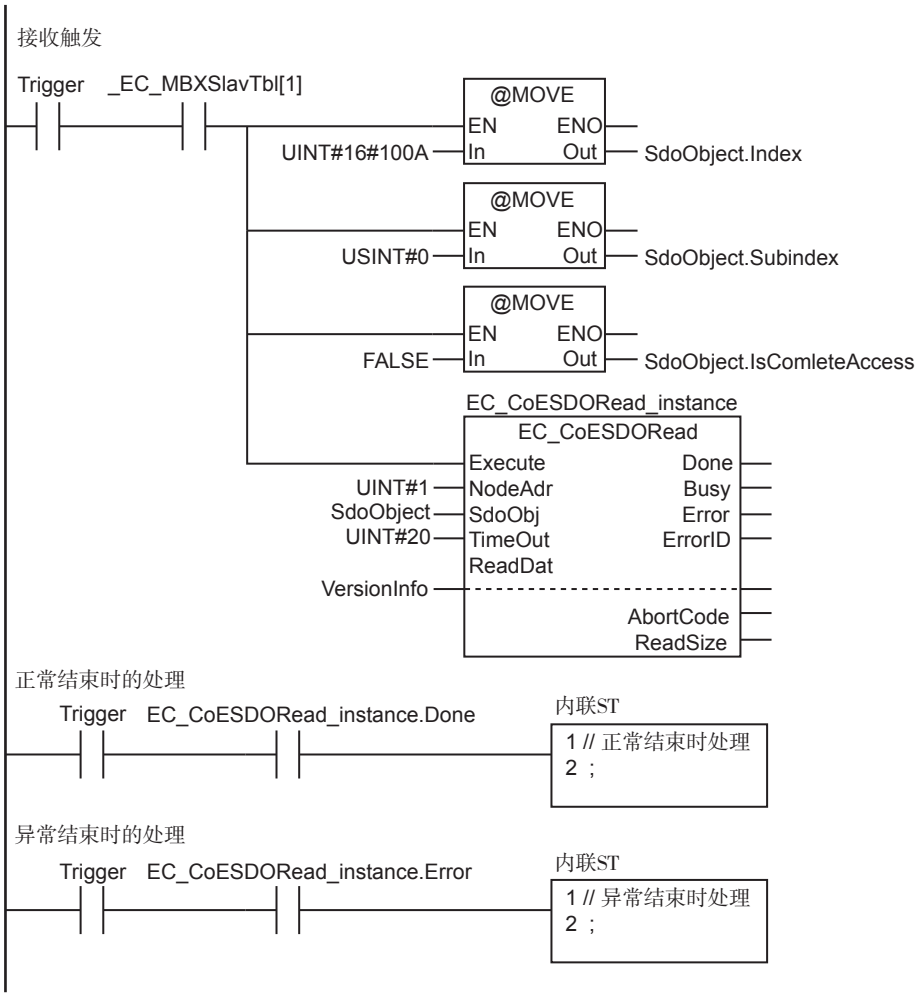


LD

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	SdoObject	_sSDO_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess:=FALSE)	SDO参数
	VersionInfo	STRING[256]	''	读取数据
	EC_CoESDORead _instance	EC_CoESDORead		

外部变量	名称	数据类型	常数	注释
	_EC_MBXSlavTbl	ARRAY[1..512] OF BOOL*1	☑	可进行信息通信的从站表

*1 NJ系列CPU单元时为 “ARRAY [1..192] OF BOOL”。



ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	SdoObject	_sSDO_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess:=FALSE)	SDO参数
	DoSdoRead	BOOL	FALSE	处理中
	VersionInfo	STRING[256]	''	读取数据
	NormalEnd	UINT	0	正常结束
	ErrorEnd	UINT	0	异常结束
	EC_CoESDORead _instance	EC_CoESDORead		

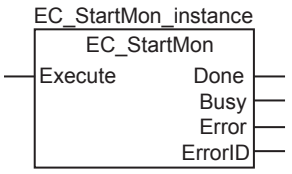
外部变量	名称	数据类型	常数	注释
	_EC_MBXSlavTbl	ARRAY[1..512] OF BOOL*1	☑	可进行信息通信的从站表

*1 NJ系列CPU单元时为 “ARRAY [1..192] OF BOOL”。

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (DoSdoRead=FALSE) AND (_EC_MBXSlavTbl[1]=TRUE) ) THEN
    DoSdoRead                 :=TRUE;
    SdoObject.Index           :=UINT#16#100A;
    SdoObject.Subindex        :=USINT#0;
    SdoObject.IsCompleteAccess :=FALSE;
    EC_CoESDORead_instance(
        Execute  :=FALSE,          // 实例初始化
        ReadDat :=VersionInfo);    // 虚拟
END_IF;

// EC_CoESDORead 执行指令
IF (DoSdoRead=TRUE) THEN
    EC_CoESDORead_instance(
        Execute  :=TRUE,
        NodeAdr:=UINT#1,           // 节点地址1
        SdoObj   :=SdoObject,      // SDO参数
        TimeOut :=UINT#20,         // 超时时间2.0s
        ReadDat :=VersionInfo);    // 读取数据

    IF (EC_CoESDORead_instance.Done=TRUE) THEN
        // 正常结束时处理
        NormalEnd:=NormalEnd+UINT#1;
    ELSIF (EC_CoESDORead_instance.Error=TRUE) THEN
        // 异常结束时处理
        ErrorEnd   :=ErrorEnd+UINT#1;
    END_IF;

END_IF;
```

EC_StartMon

开始执行EtherCAT通信的分组监控功能。

指令	名称	FB/FUN	图形表现	ST表现
EC_StartMon	EtherCAT分组监控功能启动	FB		EC_StartMon_instance(Execute, Done, Busy, Error, ErrorID);

变量

仅共通的变量

功能

开始执行EtherCAT通信的分组监控功能。
分组监控功能是指按固定数量对近期 EtherCAT 通信的分组进行收集的功能。收集数超过固定数量时，从旧的分组起依次删除。
执行本指令时，持续执行分组监控功能直至执行 “EC_StopMon”指令。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_PktMonStop	分组监控停止中	BOOL	表示分组监控功能是否已停止。 TRUE ：已停止 FALSE：未停止
_EC_PktSaving	分组数据文件保存中	BOOL	表示是否正在将分组数据保存至CPU单元主存储器的内部文件中。 TRUE ：保存中 FALSE：非保存中

参考

- 执行本指令时，无法将收集的分组数据保存至CPU单元主存储器的内部文件。
- 将分组数据保存至CPU单元主存储器的内部文件时的操作如下所述。首先，通过EC_StopMon指令停止分组监控功能。然后，通过EC_SaveMon指令执行保存。
- EtherCAT通信的详情请参阅 “NJ/NX系列 CPU单元内置EtherCAT端口 用户手册(SBCD-358)”或 “NY系列工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon 指令、EC_DisconnectSlave 指令、EC_ConnectSlave 指令、EC_ChangeEnableSetting 指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 将分组数据保存至CPU单元主存储器的内部文件时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

版本相关信息

本指令根据CPU单元及Sysmac Studio的版本，存在以下限制。
- NX701 CPU单元及NJ101 CPU单元可用于Sysmac Studio Ver.1.13以上。
- NX1P2 CPU单元可用于Sysmac Studio Ver.1.17以上。
- NJ301 CPU单元可用于CPU单元Ver.1.10以上且Sysmac Studio Ver.1.12以上。
- NY系列控制器可用于Sysmac Studio Ver.1.17以上。

示例程序

发生EtherCAT的从站异常时，将EtherCAT通信的分组保存至SD存储卡。文件名为'PacketFile'。
处理步骤如下所示。

1 监控内置EtherCAT异常的系统定义变量 “_EC_ErrSta”，发生异常时开始处理。

2 使用EC_StopMon指令，停止EtherCAT通信的分组监控功能。

3 使用EC_SaveMon指令，将EtherCAT通信的分组数据保存至CPU单元主存储器的内部文件。

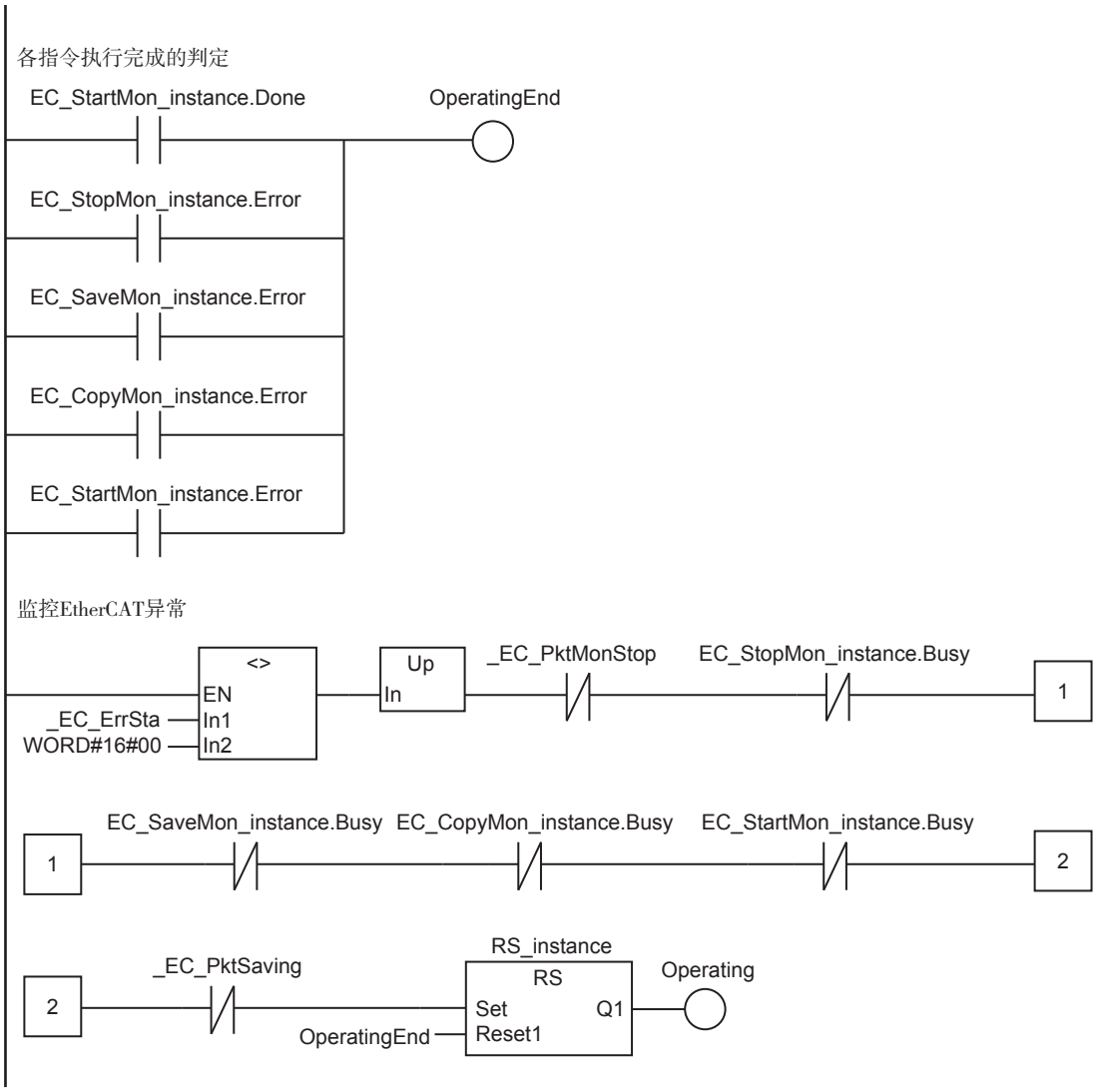
4 使用EC_CopyMon指令，将该文件复制到SD存储卡。

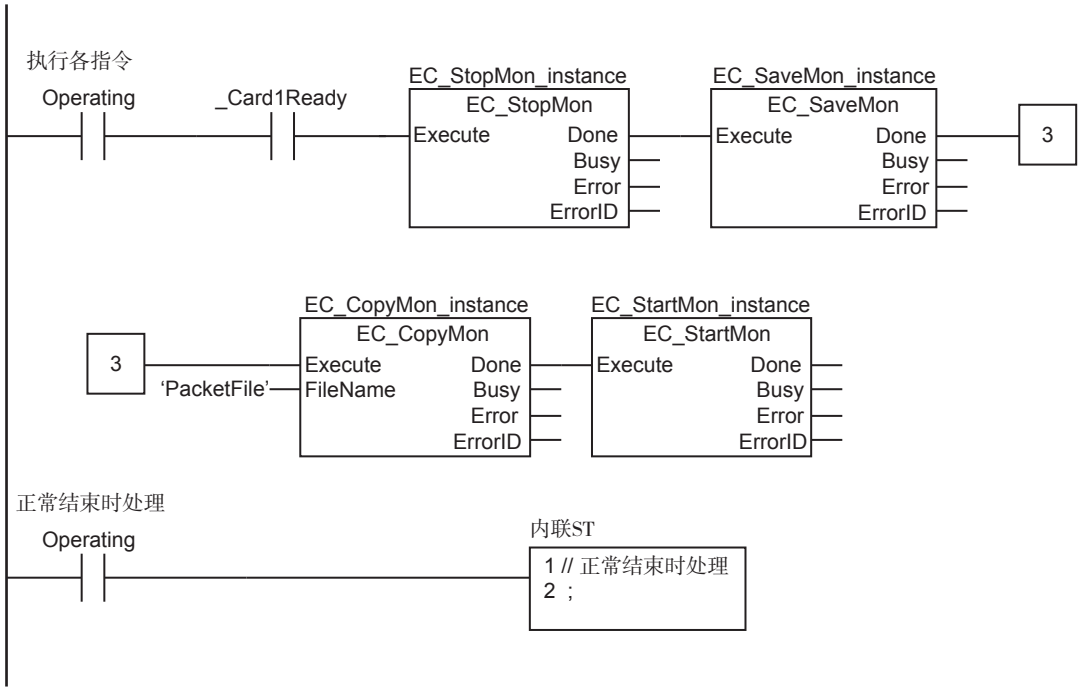
5 使用EC_StartMon指令，重新开始执行EtherCAT通信的分组监控功能。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Operating	BOOL	FALSE	执行条件
	RS_instance	RS		
	EC_StopMon_instance	EC_StopMon		
	EC_SaveMon_instance	EC_SaveMon		
	EC_CopyMon_instance	EC_CopyMon		
	EC_StartMon_instance	EC_StartMon		

外部变量	名称	数据类型	常数	注释
	_EC_ErrSta	WORD	☑	内置EtherCAT异常
	_EC_PktMonStop	BOOL	☑	分组监控停止中
	_EC_PktSaving	BOOL	☑	分组数据文件保存中
	_Card1Ready	BOOL	☑	可使用SD存储卡标志




执行各指令
Operating
_Card1Ready
EC_StopMon_instance
EC_StopMon
Execute
Done
Busy
Error
ErrorID
EC_SaveMon_instance
EC_SaveMon
Execute
Done
Busy
Error
ErrorID
3
3
'PacketFile'
EC_CopyMon_instance
EC_CopyMon
Execute
FileName
Done
Busy
Error
ErrorID
EC_StartMon_instance
EC_StartMon
Execute
Done
Busy
Error
ErrorID
正常结束时处理
Operating
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	EC_Err	BOOL	FALSE	EtherCAT主站功能模块的控制器发生异常
	EC_Err_Trigger	BOOL	FALSE	EC_Err的上升沿
	DoEC_PktSave	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	R_TRIG_instance	R_TRIG		
	EC_StopMon_instance	EC_StopMon		
	EC_SaveMon_instance	EC_SaveMon		
	EC_CopyMon_instance	EC_CopyMon		
	EC_StartMon_instance	EC_StartMon		

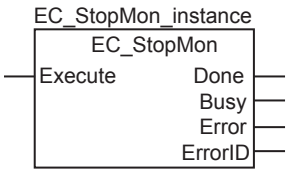
外部变量	名称	数据类型	常数	注释
	_EC_ErrSta	WORD	☑	内置EtherCAT异常
	_EC_PktMonStop	BOOL	☑	分组监控停止中
	_EC_PktSaving	BOOL	☑	分组数据文件保存中
	_Card1Ready	BOOL	☑	可使用SD存储卡标志

```
// 在 _ EC_ErrSta的上升沿启动时序
EC_Err:=(_EC_ErrSta <> WORD#16#00);
R_TRIG_instance(Clk:=EC_Err, Q=>EC_Err_Trigger);

IF ( (EC_Err_Trigger=TRUE) AND (DoEC_PktSave=FALSE) AND (_EC_PktMonStop=FALSE)
    AND (_EC_PktSaving=FALSE) AND (_Card1Ready=TRUE) ) THEN
    DoEC_PktSave :=TRUE;
    Stage            :=INT#1;
    EC_StopMon_instance(Execute:=FALSE);          // 实例初始化
    EC_SaveMon_instance(Execute:=FALSE);
    EC_CopyMon_instance(Execute:=FALSE);
    EC_StartMon_instance(Execute:=FALSE);
END_IF;

// 各指令执行
IF (DoEC_PktSave=TRUE) THEN
    CASE Stage OF
    1 :                                           // EtherCAT分组监控功能停止
        EC_StopMon_instance(
            Execute:=TRUE);

        IF (EC_StopMon_instance.Done=TRUE) THEN
            Stage:=INT#2;                         // 正常结束
        ELSIF (EC_StopMon_instance.Error=TRUE) THEN
            Stage:=INT#10;                        // 异常结束
        END_IF;

    2 :                                           // 将EtherCAT分组数据保存至内部文件中
        EC_SaveMon_instance(
            Execute:=TRUE);

        IF (EC_SaveMon_instance.Done=TRUE) THEN
            Stage:=INT#3;                         // 正常结束
        ELSIF (EC_SaveMon_instance.Error=TRUE) THEN
            Stage:=INT#20;                        // 异常结束
        END_IF;
```

```
    3 :                                          // 将EtherCAT分组数据文件复制至SD存储卡中
        EC_CopyMon_instance(
            Execute   :=TRUE,
            FileName:='PacketFile');

        IF (EC_CopyMon_instance.Done=TRUE) THEN
            Stage:=INT#4;                        // 正常结束
        ELSIF (EC_CopyMon_instance.Error=TRUE) THEN
            Stage:=INT#30;                       // 异常结束
        END_IF;

    4 :                                          // 重启EtherCAT分组监控功能
        EC_StartMon_instance(
            Execute:=TRUE);

        IF (EC_StartMon_instance.Done=TRUE) THEN
            Stage:=INT#0;                        // 正常结束
        ELSIF (EC_StartMon_instance.Error=TRUE) THEN
            Stage:=INT#40;                       // 异常结束
        END_IF;

    0 :                                          // 正常结束时处理
        DoEC_PktSave:=FALSE;

    ELSE                                         // 异常结束时处理
        DoEC_PktSave:=FALSE;
    END_CASE;
END_IF;
```

EC_StopMon

停止EtherCAT通信的分组监控功能。

指令	名称	FB/FUN	图形表现	ST表现
EC_StopMon	EtherCAT分组监控功能停止	FB		EC_StopMon_instance(Execute, Done, Busy, Error, ErrorID);

变量

仅共通的变量

功能

停止EtherCAT通信的分组监控功能。
分组监控功能是指按固定数量对近期EtherCAT通信的分组进行收集的功能。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_PktMonStop	分组监控停止中	BOOL	表示分组监控功能是否已停止。 TRUE ：已停止 FALSE：未停止
_EC_PktSaving	分组数据文件保存中	BOOL	表示是否正在将分组数据保存至CPU单元主存储器的内部文件中。 TRUE ：保存中 FALSE：非保存中

参考

- 将收集的分组数据保存至CPU单元主存储器的内部文件时的操作如下所述。首先，使用本指令停止分组监控功能，然后使用EC_SaveMon指令进行保存。
- EtherCAT 通信的详情请参阅 🕮“NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或 🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

- 本指令一旦执行，即使 “Execute” 的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done” 的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error” 的时序图请参阅 “本章说明(P.2–3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon 指令、EC_DisconnectSlave 指令、EC_ConnectSlave 指令、EC_ChangeEnableSetting 指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error” 变为TRUE。
  - 分组监控功能已停止时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

版本相关信息

本指令根据CPU单元及Sysmac Studio的版本，存在以下限制。

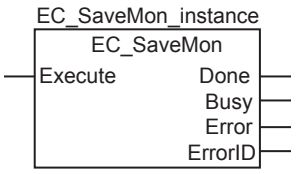
- NX701 CPU单元及NJ101 CPU单元可用于Sysmac Studio Ver.1.13以上。
- NX1P2 CPU单元可用于Sysmac Studio Ver.1.17以上。
- NJ301 CPU单元可用于CPU单元Ver.1.10以上且Sysmac Studio Ver.1.12以上。
- NY系列控制器可用于Sysmac Studio Ver.1.17以上。

示例程序

请参阅 “EC_StartMon指令(P.2–910)” 的示例程序。

EC_SaveMon

将EtherCAT通信的分组数据保存至CPU单元主存储器的内部文件中。

指令	名称	FB/FUN	图形表现	ST表现
EC_SaveMon	EtherCAT分组保存	FB		EC_SaveMon_instance(Execute, Done, Busy, Error, ErrorID);

变量

仅共通的变量

功能

将EtherCAT通信中使用分组监控功能收集到的分组数据，保存至CPU单元主存储器的内部文件中。
分组监控功能是指按固定数量对近期EtherCAT通信的分组进行收集的功能。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_PktMonStop	分组监控停止中	BOOL	表示分组监控功能是否已停止。 TRUE ：已停止 FALSE：未停止
_EC_PktSaving	分组数据文件保存中	BOOL	表示是否正在将分组数据保存至CPU单元主存储器的内部文件中。 TRUE ：保存中 FALSE：非保存中

参考

- 本指令执行时无法执行分组监控功能。
- EtherCAT 通信的详情请参阅 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

- 本指令一旦执行，即使 “Execute” 的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done” 的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error” 的时序图请参阅 “本章说明(P.2–3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- 本指令无法在分组监控功能执行时使用。请先使用EC_StopMon指令，停止分组监控功能。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon 指令、EC_DisconnectSlave 指令、EC_ConnectSlave 指令、EC_ChangeEnableSetting 指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error” 变为TRUE。
  - 分组监控功能正在执行时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

版本相关信息

本指令根据CPU单元及Sysmac Studio的版本，存在以下限制。

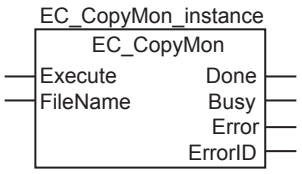
- NX701 CPU单元及NJ101 CPU单元可用于Sysmac Studio Ver.1.13以上。
- NX1P2 CPU单元可用于Sysmac Studio Ver.1.17以上。
- NJ301 CPU单元可用于CPU单元Ver.1.10以上且Sysmac Studio Ver.1.12以上。
- NY系列控制器可用于Sysmac Studio Ver.1.17以上。

示例程序

请参阅 “EC_StartMon指令(P.2–910)” 的示例程序。

EC_CopyMon

将EtherCAT通信的CPU单元主存储器中内部文件内的分组数据传送至SD存储卡。

指令	名称	FB/FUN	图形表现	ST表现
EC_CopyMon	EtherCAT分组传送	FB		EC_CopyMon_instance(Execute, FileName, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileName	文件名	输入	SD存储卡内的文件名	遵从数据类型	-	-

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileName																				○

功能

将EtherCAT通信的CPU单元主存储器中内部文件内的分组数据传送至SD存储卡。
通过 “FileName”指定SD存储卡内的文件名。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_PktSaving	分组数据文件保存中	BOOL	表示是否正在将分组数据保存至CPU单元主存储器的内部文件中。 TRUE ：保存中 FALSE：非保存中

参考

EtherCAT 通信的详情请参阅 🕮 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- 本指令无法在分组监控保存执行时使用。
- 使用本指令前，请先使用EC_SaveMon指令，将分组数据保存至CPU单元主存储器的内部文件中。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon 指令、EC_DisconnectSlave 指令、EC_ConnectSlave 指令、EC_ChangeEnableSetting 指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 分组数据文件正在保存时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

版本相关信息

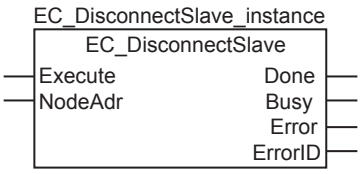
本指令根据CPU单元及Sysmac Studio的版本，存在以下限制。
- NX701 CPU单元及NJ101 CPU单元可用于Sysmac Studio Ver.1.13以上。
- NX1P2 CPU单元可用于Sysmac Studio Ver.1.17以上。
- NJ301 CPU单元可用于CPU单元Ver.1.10以上且Sysmac Studio Ver.1.12以上。
- NY系列控制器可用于Sysmac Studio Ver.1.17以上。

示例程序

请参阅 “EC_StartMon指令(P.2–910)”的示例程序。

EC_DisconnectSlave

将EtherCAT网络上的指定从站从网络中脱离。

指令	名称	FB/FUN	图形表现	ST表现
EC_DisconnectSlave	EtherCAT从站脱离	FB		EC_DisconnectSlave_instance(Execute, NodeAdr, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
NodeAdr	从站节点地址	输入	待脱离从站的节点地址	1 ~ 512*1	–	–

*1 NJ系列CPU单元时为 “1 ~ 192”。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
NodeAdr							○													

功能

将从站节点地址 “NodeAdr”指定的从站从EtherCAT网络中脱离。
脱离网络是指从站虽然存在于网络中但不动作的状态。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_EntrySlavTbl[i] i为节点地址	加入网络的从站表	BOOL[ ]	表示从站是否已加入(存在于)网络。 TRUE ：已加入(存在) FALSE：未加入(存在)
_EC_DisconnSlavTbl[i] i为节点地址	脱离指令中的从站表	BOOL[ ]	表示从站是否正在脱离指令。 TRUE ：脱离指令中 FALSE：未在脱离指令中
_EC_DisableSlavTbl[i] i为节点地址	无效设定从站表	BOOL[ ]	表示从站是否设定为无效。 TRUE ：设定为无效 FALSE：未设定为无效

参考

EtherCAT 通信的详情请参阅 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- 已脱离的从站之后存在菊花链连接(与Out端口连接)的从站时，该从站也将从EtherCAT网络中脱离。
- EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令正在执行时，无法执行本指令。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “NodeAdr”指定的从站未加入(不存在于)EtherCAT网络时。即加入网络的从站表_EC_EntrySlavTbl[i]的值为FALSE时。
  - “NodeAdr”指定的从站设定为无效时。
  - EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令中的任一指令正在执行时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

示例程序

将从站1从EtherCAT网络中脱离，以及将其重新加入。
输入Trigger1后使用EC_DisconnectSlave指令予以脱离，输入Trigger2后使用EC_ConnectSlave指令予以重新加入。

指令的排他性控制

EC_DisconnectSlave指令与EC_ConnectSlave指令无法同时执行。此外，两个指令的执行均跨多个任务周期。因此，需在确认先执行的指令完成后再执行下一指令。为此，使用 “ExclusiveFlg(指令排他性标志)”变量。“ExclusiveFlg”的值为TRUE时，表示有指令正在执行。“ExclusiveFlg”的值为TRUE的过程中，不执行下一指令。

EC_DisconnectSlave指令和EC_ConnectSlave指令无法与其它任务中的这些指令同时执行。本示例程序中，将 “ExclusiveFlg”作为全局变量。由此，本示例程序将与其它任务内的指令进行排他性控制。此时，在其它任务内，也需使用相同的全局变量 “ExclusiveFlg”执行指令排他性控制。
此外，EC_DisconnectSlave指令和EC_ConnectSlave指令无法与EC_ChangeEnableSetting指令同时执行。
“EC_ChangeEnableSetting指令(P.2-932)”的示例程序与本示例程序相同，是使用全局变量“ExclusiveFlg”执行指令排他性控制的示例。

全局变量的定义

全局变量

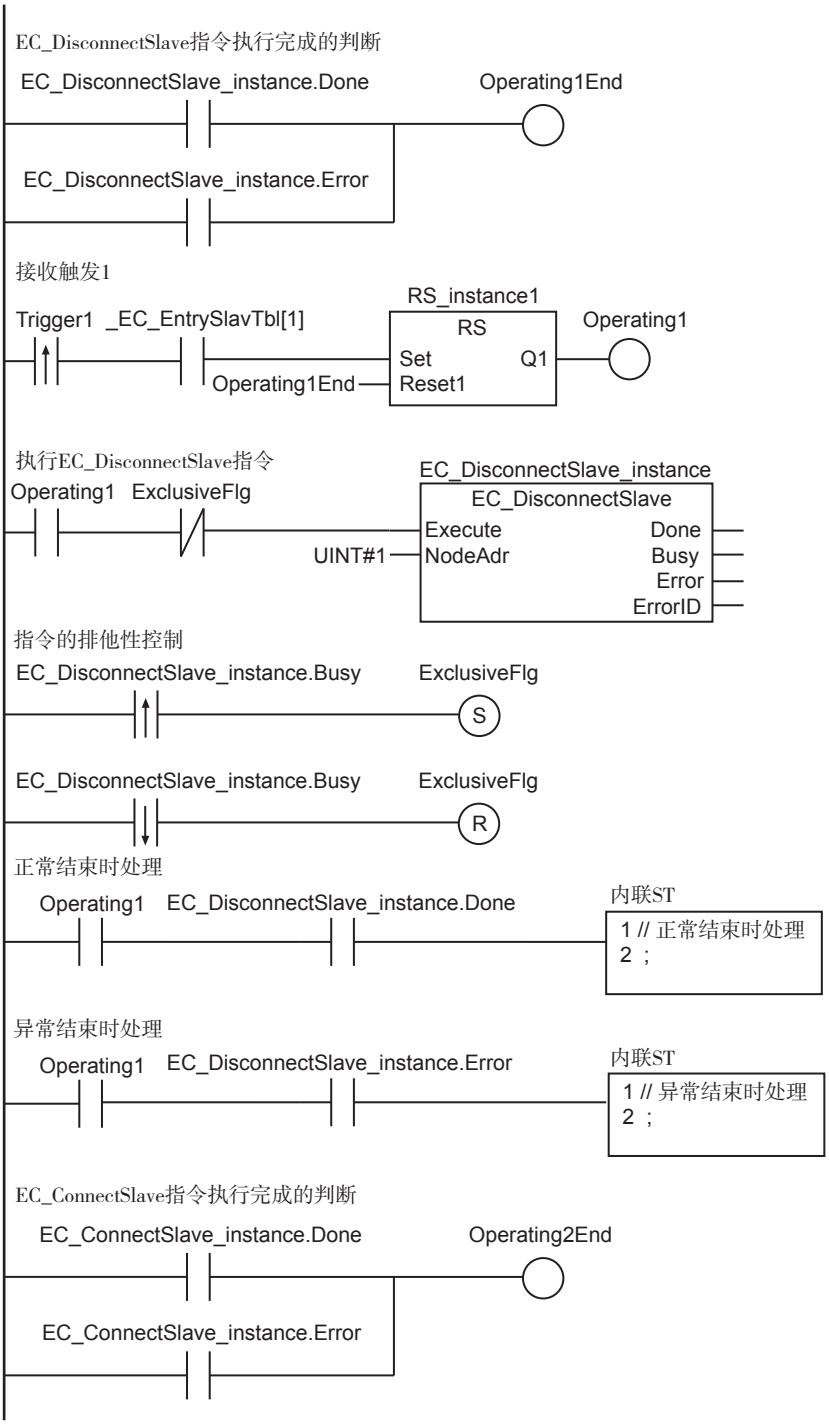
名称	数据类型	初始值	注释
ExclusiveFlg	BOOL	FALSE	指令排他性标志

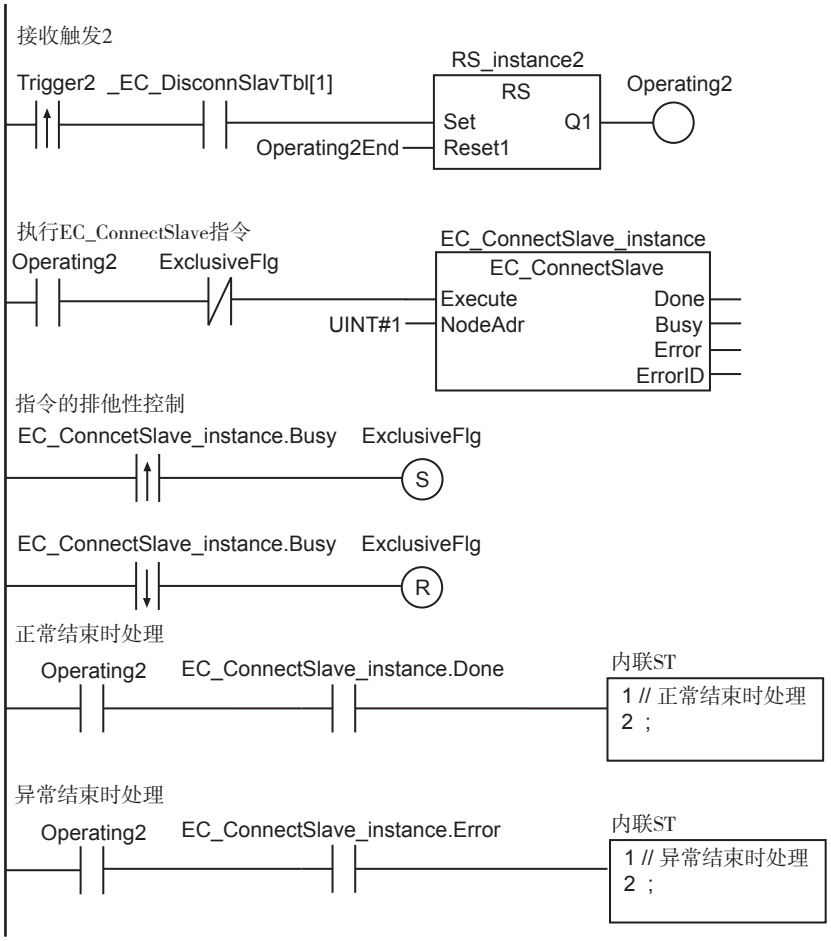
LD

内部变量	名称	数据类型	初始值	注释
	Operating1End	BOOL	FALSE	处理结束1
	Trigger1	BOOL	FALSE	执行条件1
	Operating1	BOOL	FALSE	处理中1
	RS_instance1	RS		
	EC_DisconnectSlave_instance	EC_DisconnectSlave		
	Operating2End	BOOL	FALSE	处理结束2
	Trigger2	BOOL	FALSE	执行条件2
	Operating2	BOOL	FALSE	处理中2
	RS_instance2	RS		
	EC_ConnectSlave_instance	EC_ConnectSlave		

外部变量	名称	数据类型	常数	注释
	_EC_EntrySlavTbl	ARRAY[1..512] OF BOOL*1	☑	加入网络的从站表
	_EC_DisconnSlavTbl	ARRAY[1..512] OF BOOL*1	☑	脱离指令中的从站表
	ExclusiveFlg	BOOL	☐	指令排他性标志

*1 NJ系列CPU单元时为 “ARRAY [1..192] OF BOOL”。


EC_DisconnectSlave指令执行完成的判断
EC_DisconnectSlave_instance.Done
Operating1End
EC_DisconnectSlave_instance.Error
接收触发1
RS_instance1
RS
Trigger1
_EC_EntrySlavTbl[1]
Set
Q1
Operating1
Operating1End
Reset1
执行EC_DisconnectSlave指令
EC_DisconnectSlave_instance
EC_DisconnectSlave
Operating1
ExclusiveFlg
Execute
Done
UINT#1
NodeAdr
Busy
Error
ErrorID
指令的排他性控制
EC_DisconnectSlave_instance.Busy
ExclusiveFlg
S
EC_DisconnectSlave_instance.Busy
ExclusiveFlg
R
正常结束时处理
Operating1
EC_DisconnectSlave_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating1
EC_DisconnectSlave_instance.Error
内联ST
1 // 异常结束时处理
2 ;
EC_ConnectSlave指令执行完成的判断
EC_ConnectSlave_instance.Done
Operating2End
EC_ConnectSlave_instance.Error


接收触发2
Trigger2
_EC_DisconnSlavTbl[1]
RS_instance2
RS
Set
Reset1
Q1
Operating2End
Operating2
执行EC_ConnectSlave指令
Operating2
ExclusiveFlg
EC_ConnectSlave_instance
EC_ConnectSlave
Execute
NodeAdr
UINT#1
Done
Busy
Error
ErrorID
指令的排他性控制
EC_ConncetSlave_instance.Busy
ExclusiveFlg
S
EC_ConnectSlave_instance.Busy
ExclusiveFlg
R
正常结束时处理
Operating2
EC_ConnectSlave_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating2
EC_ConnectSlave_instance.Error
内联ST
1 // 异常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger1	BOOL	FALSE	执行条件1
	LastTrigger1	BOOL	FALSE	上个任务周期的Trigger1的值
	Operating1Start	BOOL	FALSE	处理开始1
	Operating1	BOOL	FALSE	处理中1
	DisconnectSet	BOOL	FALSE	EC_DisconnectSlave指令执行中
	DisconnectReset	BOOL	FALSE	EC_DisconnectSlave指令未执行
	EC_DisconnectSlave _instance	EC_DisconnectSlave		
	Trigger2	BOOL	FALSE	执行条件2
	LastTrigger2	BOOL	FALSE	上个任务周期的Trigger2 的值
	Operating2Start	BOOL	FALSE	处理开始2
	Operating2	BOOL	FALSE	处理中2
	ConnectSet	BOOL	FALSE	EC_ConnectSlave指令执行中
	ConnectReset	BOOL	FALSE	EC_ConnectSlave指令未执行
	EC_ConnectSlave _instance	EC_ConnectSlave		
	R_TRIG_instance1	R_TRIG		
	F_TRIG_instance1	F_TRIG		
	RS_instance1	RS		
	R_TRIG_instance2	R_TRIG		
	F_TRIG_instance2	F_TRIG		
	RS_instance2	RS		

外部变量	名称	数据类型	常数	注释
	_EC_EntrySlavTbl	ARRAY[1..512] OF BOOL*1	☑	加入网络的从站表
	_EC_DisconnSlavTbl	ARRAY[1..512] OF BOOL*1	☑	脱离指令中的从站表
	ExclusiveFlg	BOOL	☐	指令排他性标志

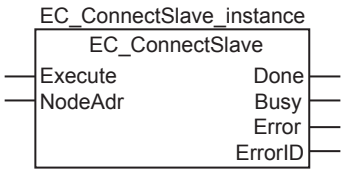
*1 NJ系列CPU单元时为 “ARRAY [1..192] OF BOOL”。

```
// Trigger1上升沿检测
IF ( (Trigger1=TRUE) AND (LastTrigger1=FALSE) AND (_EC_EntrySlavTbl[1]=TRUE) ) THEN
    Operating1Start :=TRUE;
    Operating1      :=TRUE;
END_IF;
LastTrigger1:=Trigger1;

// EC_DisconnectSlave指令初始化
IF (Operating1Start=TRUE) THEN
    EC_DisconnectSlave_instance(Execute:=FALSE);
    Operating1Start:=FALSE;
END_IF;

// EC_DisconnectSlave指令执行
IF (Operating1=TRUE) THEN
    EC_DisconnectSlave_instance(
        Execute  :=NOT(ExclusiveFlg),
        NodeAdr :=UINT#1);

    // 指令的排他性控制
    R_TRIG_instance1(EC_DisconnectSlave_instance.Busy, DisconnectSet);
    F_TRIG_instance1(EC_DisconnectSlave_instance.Busy, DisconnectReset);
    RS_instance1(DisconnectSet, DisconnectReset, ExclusiveFlg);

    IF (EC_DisconnectSlave_instance.Done=TRUE) THEN
        // 正常结束时处理
        Operating1:=FALSE;
    END_IF;

    IF (EC_DisconnectSlave_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating1:=FALSE;
    END_IF;
END_IF;

// Trigger2上升沿检测
IF ( (Trigger2=TRUE) AND (LastTrigger2=FALSE) AND (_EC_DisconnSlavTbl[1]=TRUE) ) THEN
    Operating2Start :=TRUE;
    Operating2      :=TRUE;
END_IF;
LastTrigger2:=Trigger2;

// EC_ConnectSlave指令初始化
IF (Operating2Start=TRUE) THEN
    EC_ConnectSlave_instance(Execute:=FALSE);
    Operating2Start:=FALSE;
END_IF;

// EC_ConnectSlave指令执行
IF (Operating2=TRUE) THEN
    EC_ConnectSlave_instance(
        Execute  :=NOT(ExclusiveFlg),
        NodeAdr :=UINT#1);

    // 指令的排他性控制
    R_TRIG_instance2(EC_ConnectSlave_instance.Busy, ConnectSet);
    F_TRIG_instance2(EC_ConnectSlave_instance.Busy, ConnectReset);
    RS_instance2(ConnectSet, ConnectReset, ExclusiveFlg);
```

```
    IF (EC_ConnectSlave_instance.Done=TRUE) THEN
        // 正常结束时处理
        Operating2:=FALSE;
    END_IF;

    IF (EC_ConnectSlave_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating2:=FALSE;
    END_IF;
END_IF;
```

EC_ConnectSlave

将EtherCAT网络上的指定从站重新加入网络。

指令	名称	FB/FUN	图形表现	ST表现
EC_Connect Slave	重新加入EtherCAT从站	FB		EC_ConnectSlave_instance(Execute, NodeAdr, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
NodeAdr	从站节点地址	输入	待登录从站的节点地址	0*1 ~ 512*2	–	–

*1 “0”表示网络设定中登录的所有从站。
*2 NJ系列CPU单元时为 “0 ~ 192”。

版本相关信息

NJ系列CPU单元时，从站节点地址的有效范围根据版本有以下不同。
- Ver.1.10以上为 “0 ~ 192”
- Ver.1.09以下为 “1 ~ 192”

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
NodeAdr							○													

功能

将从站节点地址 “NodeAdr”指定的从站重新加入EtherCAT网络。
重新加入网络是指从站存在于网络中且处于动作的状态。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_EntrySlavTbl[i] i为节点地址	加入网络的从站表	BOOL[ ]	表示从站是否已加入(存在于)网络。 TRUE ：已加入(存在) FALSE：未加入(存在)
_EC_DisconnSlavTbl[i] i为节点地址	脱离指令中的从站表	BOOL[ ]	表示从站是否正在脱离指令。 TRUE ：脱离指令中 FALSE：未在脱离指令中

参考

EtherCAT 通信的详情请参阅 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。

使用注意事项

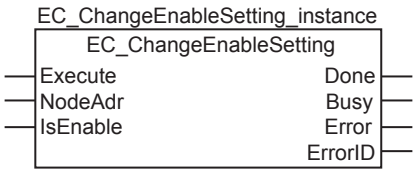
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令正在执行时，无法执行本指令。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “NodeAdr”指定的从站未加入(不存在于)EtherCAT网络时。即加入网络的从站表_EC_EntrySlavTbl[i]的值为FALSE时。
  - EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令中的任一指令正在执行时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

示例程序

请参阅 “EC_DisconnectSlave指令(P.2-922)”的示例程序。

EC_ChangeEnableSetting

切换EtherCAT从站的有效/无效。

指令	名称	FB/FUN	图形表现	ST表现
EC_ChangeEnableSetting	EtherCAT从站有效/无效切换	FB		EC_ChangeEnableSetting_instance(Execute, NodeAdr, IsEnable, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
NodeAdr	从站节点地址	输入	执行有效/无效切换的EtherCAT从站的节点地址	1 ~ 512*1	–	1
IsEnable	有效/无效切换指示		将指定EtherCAT从站设定为有效还是无效的指示 TRUE ：有效 FALSE：无效	遵从数据类型		TRUE

*1 NJ系列CPU单元时为 “1 ~ 192”。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
NodeAdr							○													
IsEnable	○																			

功能

对从站节点地址 “NodeAdr” 指定的EtherCAT从站，进行有效/无效切换。有效/无效切换指示 “IsEnable” 的值为TRUE时则有效，为FALSE时则无效。
在本指令正常结束(“Done” 的值变为TRUE)时，有效/无效切换完成。

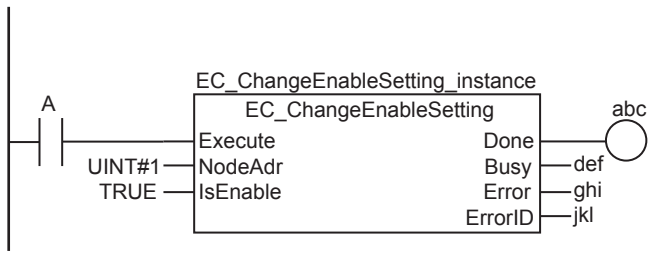
本指令根据指定EtherCAT从站的状态为有效还是无效、加入中还是脱离中、是否存在于EtherCAT网络中，表现为正常结束或异常结束。对应执行本指令前的 EtherCAT 从站的各状态，执行本指令后的状态如下所述。

执行本指令前的状态			“IsEnable”的值	执行本指令后的状态	
有效/无效	加入中/脱离中	存在*1		正常结束/异常结束	有效/无效
有效	加入中	○	TRUE	正常结束	有效
			FALSE		无效
有效	脱离中	○	TRUE	异常结束	有效 (与执行前一致)
			FALSE		
		×	TRUE		
			FALSE		
无效	_ *2	○	TRUE	正常结束	有效
			FALSE		无效
无效	_ *2	×	TRUE	异常结束	无效 (与执行前一致)
			FALSE		

*1 表示指定EtherCAT从站是否以物理方式安装在EtherCAT网络内。○：已安装。×：未安装。
*2 设定成无效的EtherCAT从站无加入中/脱离中的区别。

将节点地址1的EtherCAT从站设为有效时的示例如下所述。“NodeAdr”中指定UINT#1，“IsEnable”中指定TRUE。

LD



ST

```
EC_ChangeEnableSetting_instance(A, UINT#1, TRUE,
                                abc, def, ghi, jkl);
```

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_EntrySlavTbl[i] i为节点地址	加入网络的从站表	BOOL[ ]	表示从站是否已加入(存在于)网络。 TRUE ：已加入(存在) FALSE：未加入(存在)
_EC_DisconnSlavTbl[i] i为节点地址	脱离指令中的从站表	BOOL[ ]	表示从站是否正在脱离指令。 TRUE ：脱离指令中 FALSE：未在脱离指令中
_EC_DisableSlavTbl[i] i为节点地址	无效设定从站表	BOOL[ ]	表示从站是否无效。 TRUE ：无效 FALSE：有效或未加入(存在于)网络中

参考

- EtherCAT 通信的详情请参阅 “NJ/NX 系列 CPU 单元内置 EtherCAT 端口 用户手册 (SBCD-358)”或“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherCAT端口篇(SBCD-368)”。
- 将EtherCAT从站重新加入EtherCAT网络时，请使用 “EC_ConnectSlave(P.2-930)”指令。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 本指令仅适用于NJ/NX系列的EtherCAT端口。
- EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令正在执行时，无法执行本指令。
- 本指令的执行结果未保存至CPU单元的非易失性存储器中。因此，执行本指令后重新接通CPU单元的单元或下载用户程序时，EtherCAT从站的有效/无效设定将恢复成Sysmac Studio设定的内容。
- EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令可同时执行的数量最多为32个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “NodeAdr”指定的从站未加入(不存在于)EtherCAT网络时。即加入网络的从站表_EC_EntrySlavTbl[i]的值为FALSE时。
  - “NodeAdr”的值超过有效范围时。
  - EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、ResetECError指令、RestartNXUnit指令、NX_ChangeWriteMode指令中的任一指令正在执行时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。

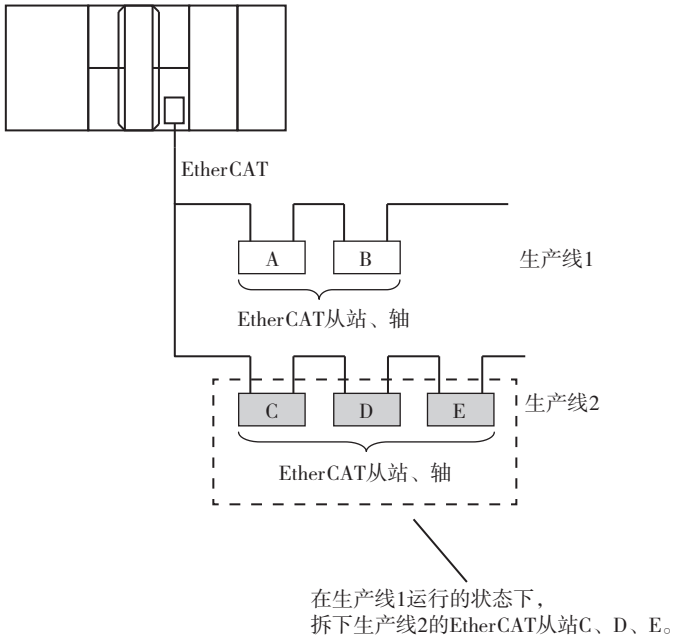
版本相关信息

本指令可用于Ver.1.04以上的CPU单元和Ver.1.05以上的Sysmac Studio。

示例程序

● 将EtherCAT从站从EtherCAT网络中拆除的示例

下图所示系统的生产线1运行的状态下，拆下生产线2的EtherCAT从站C、D、E。EtherCAT从站C、D、E中设有运动控制的轴。因此，在使EtherCAT从站无效的同时，也将进行使轴使用状态变为未使用轴的处理。



操作步骤

本示例程序的操作步骤如下所述。

1 操作人员按下显示器的按钮后，执行条件变为ON。

2 控制器将EtherCAT从站C、D、E设为无效。将各对应的轴使用状态设定为未使用轴。

3 将3个EtherCAT从站设定成无效、未使用轴的处理全部完成后，控制器可拆卸指示灯点亮。

4 操作人员在确认可拆卸指示灯点亮后，将3个EtherCAT从站拆下。

将轴使用状态设为未使用轴的指令

为了将轴使用状态设为未使用轴，而使用MC_ChangeAxisUse指令。MC_ChangeAxisUse指令的规格详情请参阅 “NJ/NX 系列 指令基准手册 运动篇 (SBCE-364)”或“NY 系列 指令基准手册 运动篇 (SBCE-380)”。

指令的排他性控制

EC_ChangeEnableSetting指令无法多个同时执行。此外，EC_ChangeEnableSetting指令的执行跨越多个任务周期。因此，需在确认先执行的EC_ChangeEnableSetting指令完成后，再执行下一EC_ChangeEnableSetting指令。为此，使用 “ExclusiveFlg(指令排他性标志)”变量。“ExclusiveFlg”的值为TRUE时，表示EC_ChangeEnableSetting指令正在执行。“ExclusiveFlg”的值为TRUE的过程中，不执行下一EC_ChangeEnableSetting指令。

EC_ChangeEnableSetting指令与其它任务内的EC_ChangeEnableSetting指令无法同时执行。本示例程序中，将 “ExclusiveFlg”作为全局变量。由此，本示例程序将与其它任务内的EC_ChangeEnableSetting指令进行排他性控制。此时，在其它任务内，也需使用相同的全局变量 “ExclusiveFlg”执行指令排他性控制。此外，EC_ChangeEnableSetting 指令与 EC_DisconnectSlave 指令、EC_ConnectSlave 指令均无法同时执行。

“EC_DisconnectSlave 指令 (P.2–922)”的示例程序与本示例程序相同，是使用全局变量 “ExclusiveFlg”执行指令排他性控制的示例。

各EtherCAT从站的轴变量与节点地址

分配至EtherCAT从站C、D、E轴的轴变量与节点地址如下所示。

EtherCAT从站	轴变量	节点地址
C	MC_Axis000	1
D	MC_Axis001	2
E	MC_Axis002	3

全局变量的定义

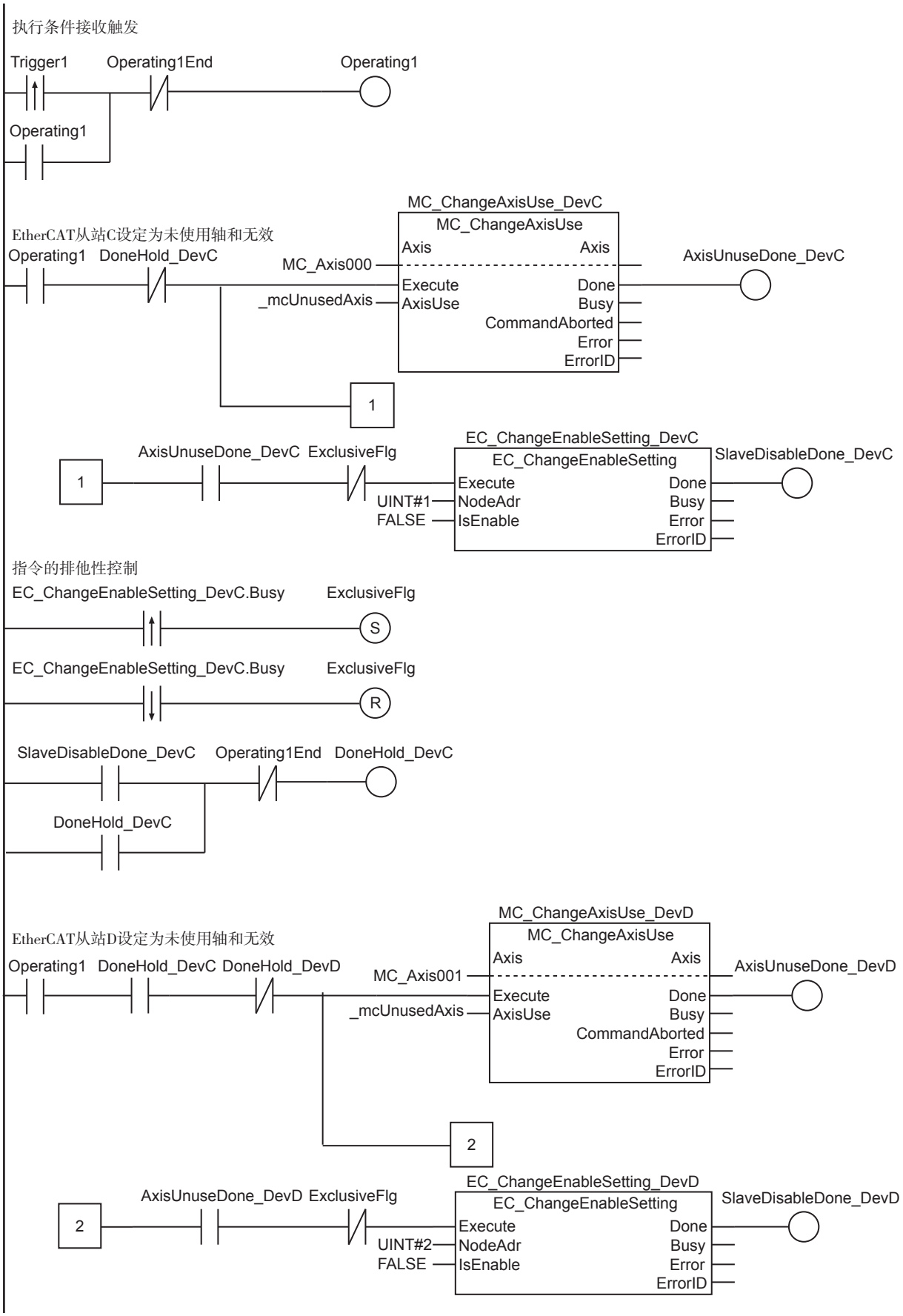
全局变量

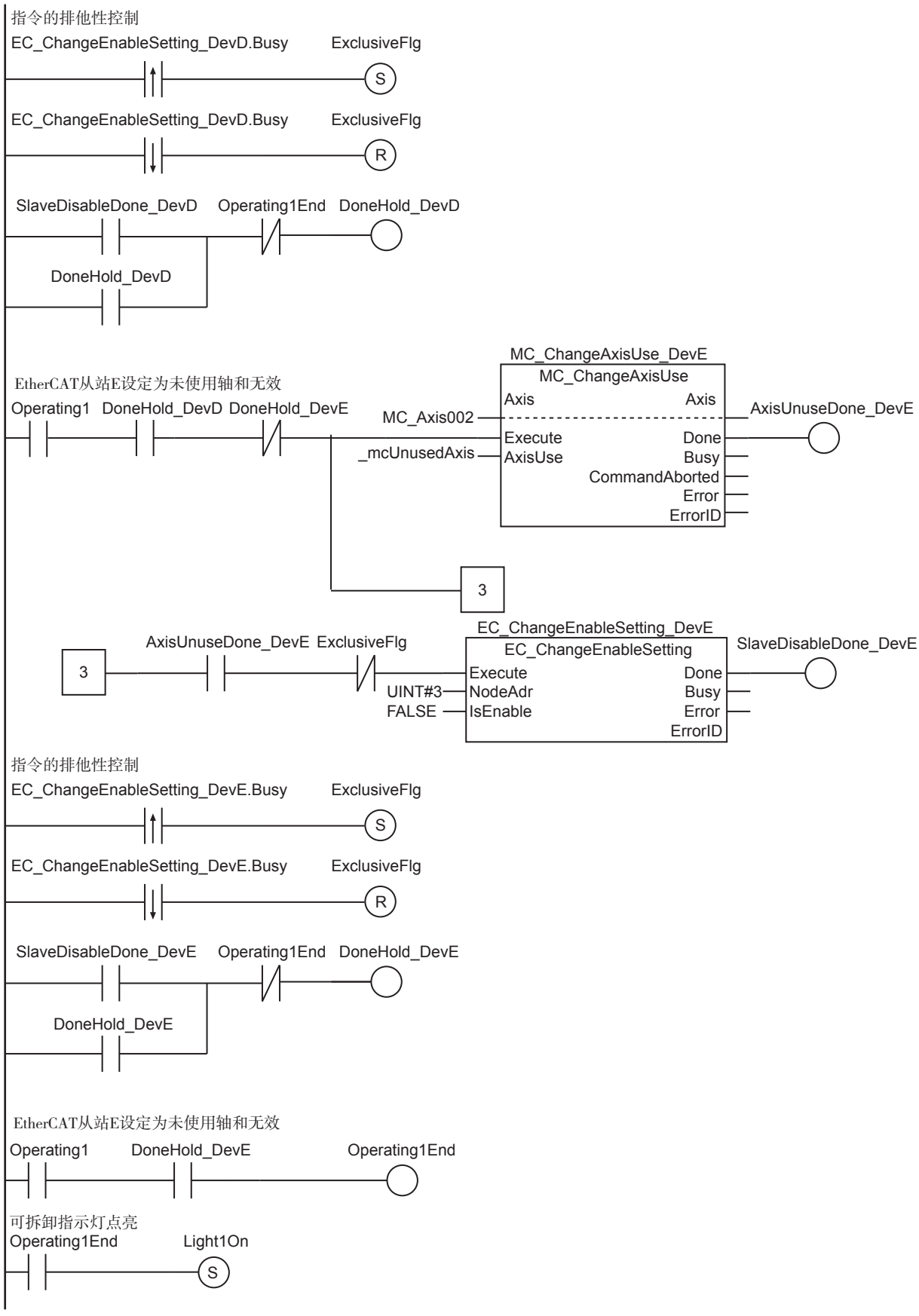
名称	数据类型	初始值	分配对象	常数	注释
MC_Axis000	_sAXIS_REF		MC://_MC_AX[0]	☑	EtherCAT从站C的轴变量
MC_Axis001	_sAXIS_REF		MC://_MC_AX[1]	☑	EtherCAT从站D的轴变量
MC_Axis002	_sAXIS_REF		MC://_MC_AX[2]	☑	EtherCAT从站E的轴变量
ExclusiveFlg	BOOL	FALSE		☐	指令排他性标志

LD

内部变量	名称	数据类型	初始值	注释
	Operating1End	BOOL	FALSE	处理结束
	Trigger1	BOOL	FALSE	执行条件
	Operating1	BOOL	FALSE	处理中
	AxisUnuseDone_DevC	BOOL	FALSE	将EtherCAT从站C的轴使用状态设为未使用轴完成
	SlaveDisableDone_DevC	BOOL	FALSE	EtherCAT从站C的无效化完成
	DoneHold_DevC	BOOL	FALSE	EtherCAT从站C处理完成的保持
	AxisUnuseDone_DevD	BOOL	FALSE	将EtherCAT从站D的轴使用状态设为未使用轴完成
	SlaveDisableDone_DevD	BOOL	FALSE	EtherCAT从站D的无效化完成
	DoneHold_DevD	BOOL	FALSE	EtherCAT从站D处理完成的保持
	AxisUnuseDone_DevE	BOOL	FALSE	将EtherCAT从站E的轴使用状态设为未使用轴完成
	SlaveDisableDone_DevE	BOOL	FALSE	EtherCAT从站E的无效化完成
	DoneHold_DevE	BOOL	FALSE	EtherCAT从站E处理完成的保持
	Light1On	BOOL	FALSE	可拆卸指示灯点亮
	MC_ChangeAxisUse_DevC	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevC	EC_ChangeEnable Setting		
	MC_ChangeAxisUse_DevD	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevD	EC_ChangeEnable Setting		
	MC_ChangeAxisUse_DevE	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevE	EC_ChangeEnable Setting		

外部变量	名称	数据类型	常数	注释
	MC_Axis000	_sAXIS_REF	☑	EtherCAT从站C的轴变量
	MC_Axis001	_sAXIS_REF	☑	EtherCAT从站D的轴变量
	MC_Axis002	_sAXIS_REF	☑	EtherCAT从站E的轴变量
	ExclusiveFlg	BOOL	☐	指令排他性标志


执行条件接收触发
Trigger1
Operating1End
Operating1
Operating1
MC_ChangeAxisUse_DevC
MC_ChangeAxisUse
EtherCAT从站C设定为未使用轴和无效
Operating1
DoneHold_DevC
MC_Axis000
Axis
Axis
AxisUnuseDone_DevC
Execute
Done
_mcUnusedAxis
AxisUse
Busy
CommandAborted
Error
ErrorID
1
EC_ChangeEnableSetting_DevC
EC_ChangeEnableSetting
1
AxisUnuseDone_DevC
ExclusiveFlg
SlaveDisableDone_DevC
Execute
Done
UINT#1
NodeAdr
Busy
FALSE
IsEnable
Error
ErrorID
指令的排他性控制
EC_ChangeEnableSetting_DevC.Busy
ExclusiveFlg
S
EC_ChangeEnableSetting_DevC.Busy
ExclusiveFlg
R
SlaveDisableDone_DevC
Operating1End
DoneHold_DevC
DoneHold_DevC
MC_ChangeAxisUse_DevD
MC_ChangeAxisUse
EtherCAT从站D设定为未使用轴和无效
Operating1
DoneHold_DevC
DoneHold_DevD
MC_Axis001
Axis
Axis
AxisUnuseDone_DevD
Execute
Done
_mcUnusedAxis
AxisUse
Busy
CommandAborted
Error
ErrorID
2
EC_ChangeEnableSetting_DevD
EC_ChangeEnableSetting
2
AxisUnuseDone_DevD
ExclusiveFlg
SlaveDisableDone_DevD
Execute
Done
UINT#2
NodeAdr
Busy
FALSE
IsEnable
Error
ErrorID


指令的排他性控制
EC_ChangeEnableSetting_DevD.Busy
ExclusiveFlg
S
EC_ChangeEnableSetting_DevD.Busy
ExclusiveFlg
R
SlaveDisableDone_DevD
Operating1End
DoneHold_DevD
DoneHold_DevD
EtherCAT从站E设定为未使用轴和无效
Operating1
DoneHold_DevD
DoneHold_DevE
MC_ChangeAxisUse_DevE
MC_ChangeAxisUse
Axis
Axis
MC_Axis002
Execute
Done
_mcUnusedAxis
AxisUse
Busy
CommandAborted
Error
ErrorID
AxisUnuseDone_DevE
3
3
AxisUnuseDone_DevE
ExclusiveFlg
EC_ChangeEnableSetting_DevE
EC_ChangeEnableSetting
Execute
Done
UINT#3
NodeAdr
Busy
FALSE
IsEnable
Error
ErrorID
SlaveDisableDone_DevE
指令的排他性控制
EC_ChangeEnableSetting_DevE.Busy
ExclusiveFlg
S
EC_ChangeEnableSetting_DevE.Busy
ExclusiveFlg
R
SlaveDisableDone_DevE
Operating1End
DoneHold_DevE
DoneHold_DevE
EtherCAT从站E设定为未使用轴和无效
Operating1
DoneHold_DevE
Operating1End
可拆卸指示灯点亮
Operating1End
Light1On
S

ST

内部变量	名称	数据类型	初始值	注释
	Operating1End	BOOL	FALSE	处理结束
	Trigger1	BOOL	FALSE	执行条件
	Operating1	BOOL	FALSE	处理中
	Operating1Set	BOOL	FALSE	处理开始
	Light1On	BOOL	FALSE	可拆卸指示灯点亮
	DoneHold_DevC	BOOL	FALSE	EtherCAT从站C处理完成的保持
	DoneHold_DevD	BOOL	FALSE	EtherCAT从站D处理完成的保持
	DoneHold_DevE	BOOL	FALSE	EtherCAT从站E处理完成的保持
	ExclusiveFlgSet	BOOL	FALSE	指令排他性标志ON
	ExclusiveFlgReset	BOOL	FALSE	指令排他性标志OFF
	R_TRIG_instance1	R_TRIG		
	RS_instance1	RS		
	SR_instance1	SR		
	MC_ChangeAxisUse_DevC	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevC	EC_ChangeEnableSetting		
	R_TRIG_DevC	R_TRIG		
	F_TRIG_DevC	F_TRIG		
	RS_ExFlg_DevC	RS		
	RS_DevC	RS		
	MC_ChangeAxisUse_DevD	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevD	EC_ChangeEnableSetting		
	R_TRIG_DevD	R_TRIG		
	F_TRIG_DevD	F_TRIG		
	RS_ExFlg_DevD	RS		
	RS_DevD	RS		
	MC_ChangeAxisUse_DevE	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevE	EC_ChangeEnableSetting		
	R_TRIG_DevE	R_TRIG		
	F_TRIG_DevE	F_TRIG		
	RS_ExFlg_DevE	RS		
	RS_DevE	RS		

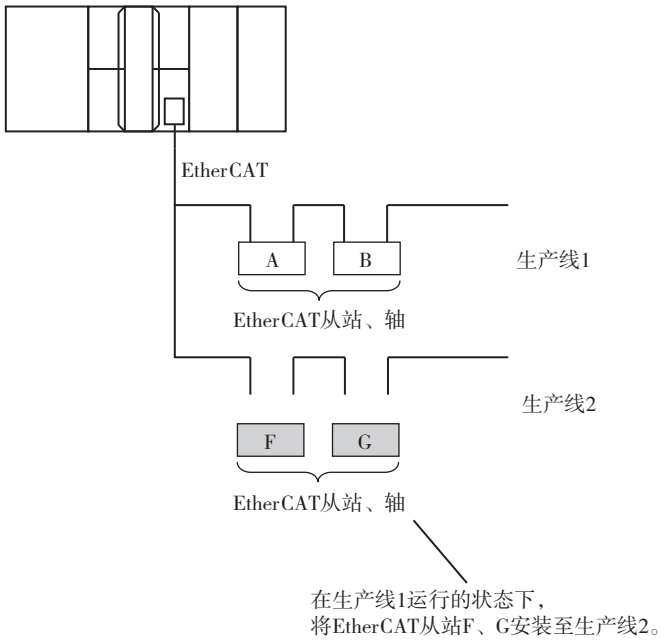
外部变量	名称	数据类型	常数	注释
	MC_Axis000	_sAXIS_REF	☑	EtherCAT从站C的轴变量
	MC_Axis001	_sAXIS_REF	☑	EtherCAT从站D的轴变量
	MC_Axis002	_sAXIS_REF	☑	EtherCAT从站E的轴变量
	ExclusiveFlg	BOOL	☐	指令排他性标志

```
// 执行条件接收触发
R_TRIG_instance1(Trigger1, Operating1Set);
RS_instance1(
    Set    :=Operating1Set,
    Reset1 :=Operating1End,
    Q1     =>Operating1);

// 将EtherCAT从站C的轴使用状态设为未使用轴
MC_ChangeAxisUse_DevC(
    Axis    :=MC_Axis000,
    Execute :=(Operating1 & NOT(DoneHold_DevC)),
    AxisUse :=_mcUnusedAxis);

// EtherCAT从站C的无效化
EC_ChangeEnableSetting_DevC(
    Execute  :=(Operating1 & MC_ChangeAxisUse_DevC.Done & NOT(ExclusiveFlg)),
    NodeAdr :=UINT#1,
    IsEnable :=FALSE);

// 指令的排他性控制
R_TRIG_DevC(EC_ChangeEnableSetting_DevC.Busy, ExclusiveFlgSet);
F_TRIG_DevC(EC_ChangeEnableSetting_DevC.Busy, ExclusiveFlgReset);
RS_ExFlg_DevC(
    Set    :=ExclusiveFlgSet,
    Reset1 :=ExclusiveFlgReset,
    Q1     =>ExclusiveFlg);
RS_DevC(
    Set    :=EC_ChangeEnableSetting_DevC.Done,
    Reset1 :=Operating1End,
    Q1     =>DoneHold_DevC);

// 将EtherCAT从站D的轴使用状态设为未使用轴
MC_ChangeAxisUse_DevD(
    Axis    :=MC_Axis001,
    Execute :=(Operating1 & DoneHold_DevC & NOT(DoneHold_DevD)),
    AxisUse :=_mcUnusedAxis);

// EtherCAT从站D的无效化
EC_ChangeEnableSetting_DevD(
    Execute  :=(Operating1 & DoneHold_DevC & MC_ChangeAxisUse_DevD.Done & NOT(ExclusiveFlg)),
    NodeAdr :=UINT#2,
    IsEnable :=FALSE);

// 指令的排他性控制
R_TRIG_DevD(EC_ChangeEnableSetting_DevD.Busy, ExclusiveFlgSet);
F_TRIG_DevD(EC_ChangeEnableSetting_DevD.Busy, ExclusiveFlgReset);
RS_ExFlg_DevD(
    Set    :=ExclusiveFlgSet,
    Reset1 :=ExclusiveFlgReset,
    Q1     =>ExclusiveFlg);
RS_DevD(
    Set    :=EC_ChangeEnableSetting_DevD.Done,
    Reset1 :=Operating1End,
    Q1     =>DoneHold_DevD);
```

```
// 将EtherCAT从站E的轴使用状态设为未使用轴
MC_ChangeAxisUse_DevE(
    Axis      :=MC_Axis002,
    Execute :=(Operating1 & DoneHold_DevD & NOT(DoneHold_DevE)),
    AxisUse :=_mcUnusedAxis);

// EtherCAT从站E的无效化
EC_ChangeEnableSetting_DevE(
    Execute  :=(Operating1 & DoneHold_DevD & MC_ChangeAxisUse_DevE.Done & NOT(ExclusiveFlg)),
    NodeAdr :=UINT#3,
    IsEnable :=FALSE);

// 指令的排他性控制
R_TRIG_DevE(EC_ChangeEnableSetting_DevE.Busy, ExclusiveFlgSet);
F_TRIG_DevE(EC_ChangeEnableSetting_DevE.Busy, ExclusiveFlgReset);
RS_ExFlg_DevE(
    Set      :=ExclusiveFlgSet,
    Reset1 :=ExclusiveFlgReset,
    Q1     =>ExclusiveFlg);
RS_DevE(
    Set      :=EC_ChangeEnableSetting_DevE.Done,
    Reset1 :=Operating1End,
    Q1     =>DoneHold_DevE);

// 确认EtherCAT从站E已设定成未使用轴和无效
Operating1End:=(Operating1 & DoneHold_DevE);

// 可拆卸指示灯点亮
SR_instance1(
    Set1 :=Operating1End,
    Q1  =>Light1On);
```

● 将EtherCAT从站安装至EtherCAT网络中的示例

与上述示例相同，在系统的生产线1运行的状态下，将EtherCAT从站F、G安装至生产线2。EtherCAT从站F、G中也设有运动控制的轴。因此，在使EtherCAT从站有效的同时，也将进行使轴使用状态变为使用轴的处理。



操作步骤

本示例程序的操作步骤如下所述。

1. 操作人员安装EtherCAT从站F、G。
2. 操作人员按下显示器的按钮后，执行条件变为ON。
3. 控制器将EtherCAT从站F、G设为有效。将各对应的轴使用状态设定为使用轴。
4. 将2个EtherCAT从站设为有效、使用轴的处理全部完成后，控制器安装完成指示灯点亮。

将轴使用状态设为使用轴的指令

为了将轴使用状态设为未使用轴，而使用MC_ChangeAxisUse指令。MC_ChangeAxisUse指令的规格详情请参阅 “NJ/NX 系列 指令基准手册 运动篇 (SBCE-364)”或 “NY 系列 指令基准手册 运动篇 (SBCE-380)”。

指令的排他性控制

EC_ChangeEnableSetting指令无法多个同时执行。此外，EC_ChangeEnableSetting指令的执行跨越多个任务周期。因此，需在确认先执行的EC_ChangeEnableSetting指令完成后，再执行下一EC_ChangeEnableSetting指令。为此，使用 “ExclusiveFlg(指令排他性标志)”变量。“ExclusiveFlg”的值为TRUE时，表示EC_ChangeEnableSetting指令正在执行。“ExclusiveFlg”的值为TRUE的过程中，不执行下一EC_ChangeEnableSetting指令。

EC_ChangeEnableSetting指令与其它任务内的EC_ChangeEnableSetting指令无法同时执行。本示例程序中，将 “ExclusiveFlg”作为全局变量。由此，本示例程序将与其它任务内的EC_ChangeEnableSetting指令进行排他性控制。此时，在其它任务内，也需使用相同的全局变量 “ExclusiveFlg”执行指令排他性控制。此外，EC_ChangeEnableSetting 指令与 EC_DisconnectSlave 指令、EC_ConnectSlave 指令均无法同时执行。

“EC_DisconnectSlave 指令 (P.2–922)”的示例程序与本示例程序相同，是使用全局变量 “ExclusiveFlg”执行指令排他性控制的示例。

各EtherCAT从站的轴变量与节点地址

分配至EtherCAT从站F、G轴的轴变量与节点地址如下所示。

EtherCAT从站	轴变量	节点地址
F	MC_Axis003	4
G	MC_Axis004	5

全局变量的定义

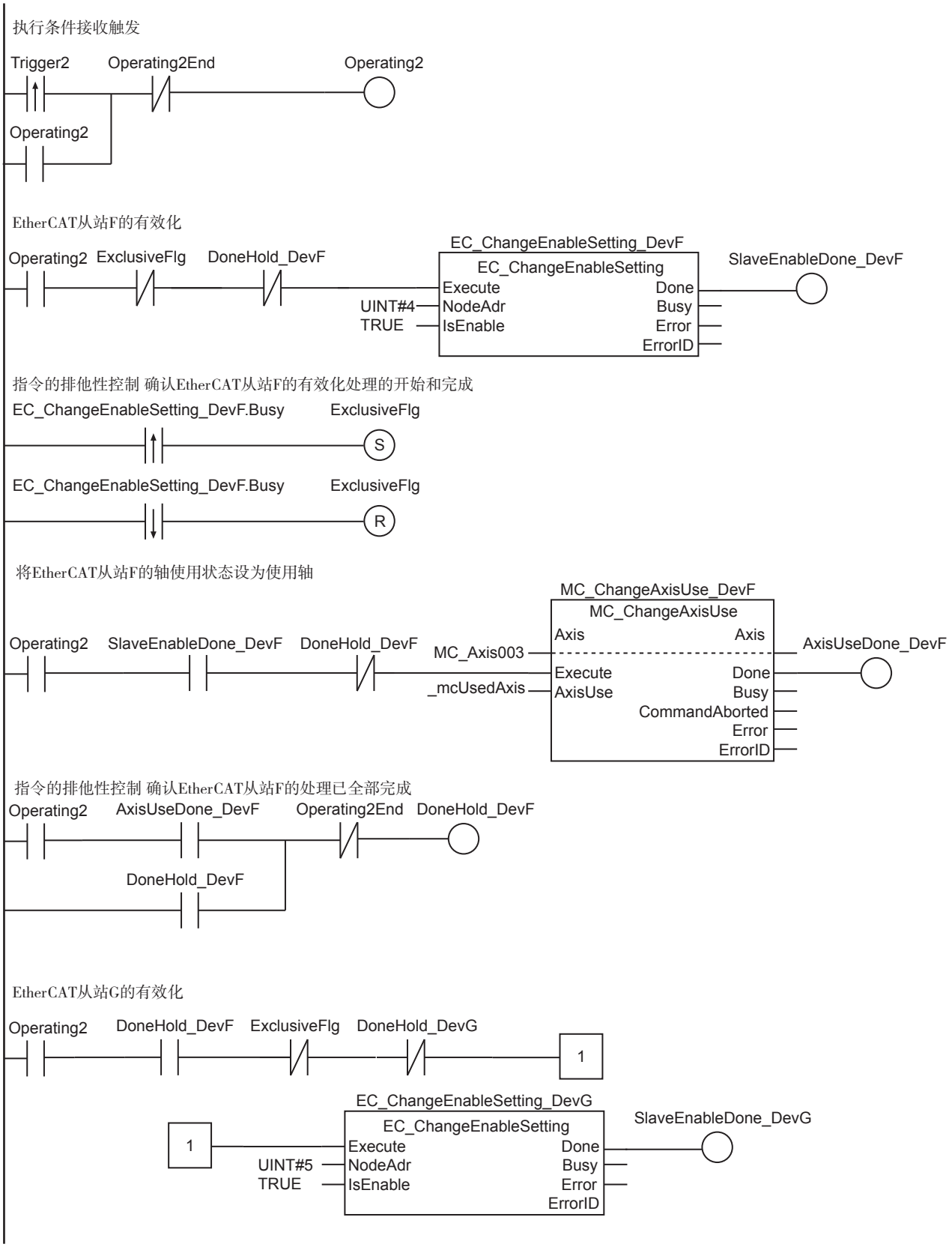
全局变量

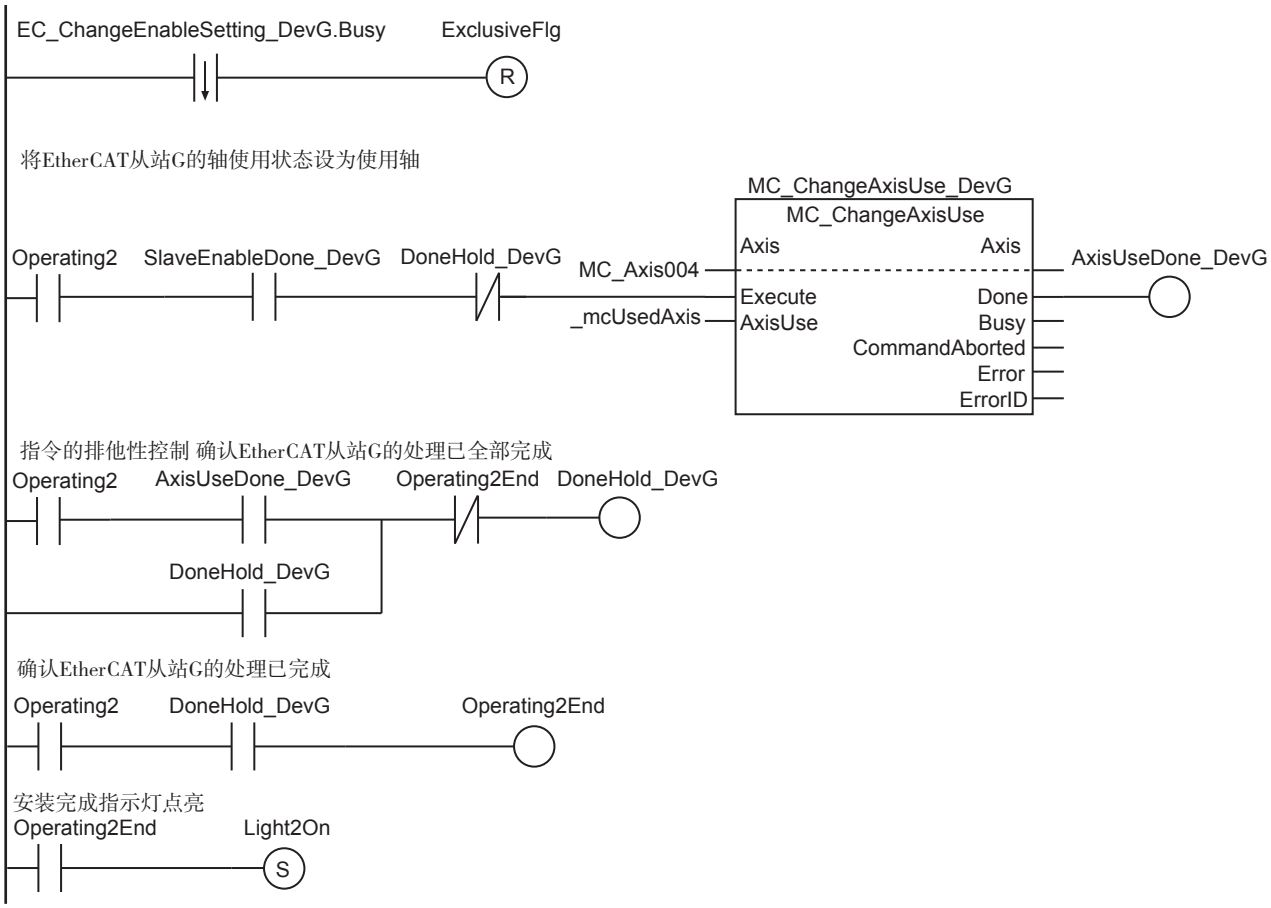
名称	数据类型	初始值	分配对象	常数	注释
MC_Axis003	_sAXIS_REF		MC://_MC_AX[3]	☑	EtherCAT从站F的轴变量
MC_Axis004	_sAXIS_REF		MC://_MC_AX[4]	☑	EtherCAT从站G的轴变量
ExclusiveFlg	BOOL	FALSE		☐	指令排他性标志

LD

内部变量	名称	数据类型	初始值	注释
	Operating2End	BOOL	FALSE	处理结束
	Trigger2	BOOL	FALSE	执行条件
	Operating2	BOOL	FALSE	处理中
	AxisUseDone_DevF	BOOL	FALSE	将EtherCAT从站F的轴使用状态设为使用轴完成
	SlaveEnableDone_DevF	BOOL	FALSE	EtherCAT从站F的有效化完成
	DoneHold_DevF	BOOL	FALSE	EtherCAT从站F处理完成的保持
	AxisUseDone_DevG	BOOL	FALSE	将EtherCAT从站G的轴使用状态设为使用轴完成
	SlaveEnableDone_DevG	BOOL	FALSE	EtherCAT从站G的有效化完成
	DoneHold_DevG	BOOL	FALSE	EtherCAT从站G处理完成的保持
	Light2On	BOOL	FALSE	安装完成指示灯点亮
	MC_ChangeAxisUse_DevF	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevF	EC_ChangeEnableSetting		
	MC_ChangeAxisUse_DevG	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevG	EC_ChangeEnableSetting		

外部变量	名称	数据类型	常数	注释
	MC_Axis003	_sAXIS_REF	☑	EtherCAT从站F的轴变量
	MC_Axis004	_sAXIS_REF	☑	EtherCAT从站G的轴变量
	ExclusiveFlg	BOOL	☐	指令排他性标志


执行条件接收触发
Trigger2
Operating2End
Operating2
Operating2
EtherCAT从站F的有效化
EC_ChangeEnableSetting_DevF
Operating2
ExclusiveFlg
DoneHold_DevF
EC_ChangeEnableSetting
Execute
NodeAdr
IsEnable
UINT#4
TRUE
Done
Busy
Error
ErrorID
SlaveEnableDone_DevF
指令的排他性控制 确认EtherCAT从站F的有效化处理的开始和完成
EC_ChangeEnableSetting_DevF.Busy
ExclusiveFlg
S
EC_ChangeEnableSetting_DevF.Busy
ExclusiveFlg
R
将EtherCAT从站F的轴使用状态设为使用轴
MC_ChangeAxisUse_DevF
MC_ChangeAxisUse
Operating2
SlaveEnableDone_DevF
DoneHold_DevF
MC_Axis003
_mcUsedAxis
Axis
Axis
Execute
AxisUse
Done
Busy
CommandAborted
Error
ErrorID
AxisUseDone_DevF
指令的排他性控制 确认EtherCAT从站F的处理已全部完成
Operating2
AxisUseDone_DevF
Operating2End
DoneHold_DevF
DoneHold_DevF
EtherCAT从站G的有效化
Operating2
DoneHold_DevF
ExclusiveFlg
DoneHold_DevG
1
EC_ChangeEnableSetting_DevG
1
EC_ChangeEnableSetting
Execute
NodeAdr
IsEnable
UINT#5
TRUE
Done
Busy
Error
ErrorID
SlaveEnableDone_DevG


EC_ChangeEnableSetting_DevG.Busy
ExclusiveFlg
R
将EtherCAT从站G的轴使用状态设为使用轴
MC_ChangeAxisUse_DevG
MC_ChangeAxisUse
Operating2
SlaveEnableDone_DevG
DoneHold_DevG
MC_Axis004
_mcUsedAxis
Axis
Axis
Execute
AxisUse
Done
Busy
CommandAborted
Error
ErrorID
AxisUseDone_DevG
指令的排他性控制 确认EtherCAT从站G的处理已全部完成
Operating2
AxisUseDone_DevG
Operating2End
DoneHold_DevG
DoneHold_DevG
确认EtherCAT从站G的处理已完成
Operating2
DoneHold_DevG
Operating2End
安装完成指示灯点亮
Operating2End
Light2On
S

ST

内部变量	名称	数据类型	初始值	注释
	Operating2End	BOOL	FALSE	处理结束
	Trigger2	BOOL	FALSE	执行条件
	Operating2	BOOL	FALSE	处理中
	Operating2Set	BOOL	FALSE	处理开始
	Light2On	BOOL	FALSE	安装完成指示灯点亮
	DoneHold_DevF	BOOL	FALSE	EtherCAT从站F处理完成的保持
	DoneHold_DevG	BOOL	FALSE	EtherCAT从站G处理完成的保持
	ExclusiveFlgSet	BOOL	FALSE	指令排他性标志ON
	ExclusiveFlgReset	BOOL	FALSE	指令排他性标志OFF
	R_TRIG_instance2	R_TRIG		
	RS_instance2	RS		
	SR_instance2	SR		
	MC_ChangeAxisUse_DevF	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevF	EC_ChangeEnableSetting		
	R_TRIG_DevF	R_TRIG		
	F_TRIG_DevF	F_TRIG		
	RS_ExFlg_DevF	RS		
	RS_DevF	RS		
	MC_ChangeAxisUse_DevG	MC_ChangeAxisUse		
	EC_ChangeEnableSetting_DevG	EC_ChangeEnableSetting		
	R_TRIG_DevG	R_TRIG		
	F_TRIG_DevG	F_TRIG		
	RS_ExFlg_DevG	RS		
	RS_DevG	RS		

外部变量	名称	数据类型	常数	注释
	MC_Axis003	_sAXIS_REF	☑	EtherCAT从站F的轴变量
	MC_Axis004	_sAXIS_REF	☑	EtherCAT从站G的轴变量
	ExclusiveFlg	BOOL	☐	指令排他性标志

```
// 执行条件接收触发
R_TRIG_instance2(Trigger2, Operating2Set);
RS_instance2(
    Set      :=Operating2Set,
    Reset1 :=Operating2End,
    Q1       =>Operating2);

// EtherCAT从站F的有效化
EC_ChangeEnableSetting_DevF(
    Execute  :=(Operating2 & NOT(ExclusiveFlg) & NOT(DoneHold_DevF)),
    NodeAdr :=UINT#4,
    IsEnable :=TRUE);

// 指令的排他性控制确认 EtherCAT从站F的有效化处理的开始和完成
R_TRIG_DevF(EC_ChangeEnableSetting_DevF.Busy, ExclusiveFlgSet);
F_TRIG_DevF(EC_ChangeEnableSetting_DevF.Busy, ExclusiveFlgReset);
RS_ExFlg_DevF(
    Set      :=ExclusiveFlgSet,
    Reset1 :=ExclusiveFlgReset,
    Q1       =>ExclusiveFlg);

// 将EtherCAT从站F的轴使用状态设为使用轴
MC_ChangeAxisUse_DevF(
    Axis     :=MC_Axis003,
    Execute :=(Operating2 & EC_ChangeEnableSetting_DevF.Done & NOT(DoneHold_DevF)),
    AxisUse :=_mcUsedAxis);

// 指令的排他性控制 确认EtherCAT从站F的处理已全部完成
RS_DevF(
    Set      :=(Operating2 & MC_ChangeAxisUse_DevF.Done),
    Reset1 :=Operating2End,
    Q1       =>DoneHold_DevF);

// EtherCAT从站G的有效化
EC_ChangeEnableSetting_DevG(
    Execute  :=(Operating2 & DoneHold_DevF & NOT(ExclusiveFlg) & NOT(DoneHold_DevG)),
    NodeAdr :=UINT#5,
    IsEnable :=TRUE);

// 指令的排他性控制 确认EtherCAT从站G的有效化处理的开始和完成
R_TRIG_DevG(EC_ChangeEnableSetting_DevG.Busy, ExclusiveFlgSet);
F_TRIG_DevG(EC_ChangeEnableSetting_DevG.Busy, ExclusiveFlgReset);
RS_ExFlg_DevG(
    Set      :=ExclusiveFlgSet,
    Reset1 :=ExclusiveFlgReset,
    Q1       =>ExclusiveFlg);

// 将EtherCAT从站G的轴使用状态设为使用轴
MC_ChangeAxisUse_DevG(
    Axis     :=MC_Axis004,
    Execute :=(Operating2 & EC_ChangeEnableSetting_DevG.Done & NOT(DoneHold_DevG)),
    AxisUse :=_mcUsedAxis);

// 指令的排他性控制 确认EtherCAT从站G的处理已全部完成
RS_DevG(
    Set      :=(Operating2 & MC_ChangeAxisUse_DevG.Done),
    Reset1 :=Operating2End,
    Q1       =>DoneHold_DevG);
```

```
// 确认EtherCAT从站G的处理已完成
Operating2End:=Operating2 & DoneHold_DevG;

// 安装完成指示灯点亮
SR_instance2(
    Set1 :=Operating2End,
    Q1  =>Light2On);
```

NX_WriteObj

将数据写入EtherCAT耦合器单元或NX单元的NX对象。

指令	名称	FB/FUN	图形表现	ST表现
NX_WriteObj	NX对象写入	FB	NX_WriteObj_instance NX_WriteObj Execute / Done UnitProxy / Busy Obj / Error TimeOut / ErrorID WriteDat / ErrorIDEx	NX_WriteObj_instance(Execute, UnitProxy, Obj, TimeOut, WriteDat, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	指定单元	输入	要写入数据的单元	–	–	(*)
Obj	对象参数		对象参数			–
TimeOut	超时时间		超时时间 0时为2.0s	0 ~ 60000	ms	2000 (2.0s)
WriteDat	写入数据		要写入NX对象的数据	遵从数据类型	–	(*)

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID　详情参阅功能说明																			
Obj	结构体_sNXOBJ_ACCESS　详情参阅功能说明																			
TimeOut							○													
WriteDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定整个数组																			

功能

将 “WriteDat”的内容写入至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的NX对象。
通过 “UnitProxy”指定要写入数据的单元。

超时时间由 “TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。这种情况下，可能不会写入数据。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	指定单元	要指定的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至要指定单元的设备变量传输至 “UnitProxy”。

“Obj”的数据类型为结构体_sNXOBJ_ACCESS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Obj	对象参数	对象参数	_sNXOBJ_ACCESS	–	–	–
Index	索引	索引	UINT	遵从数据类型	–	0
Subindex	子索引	子索引	USINT			
IsComplete Access*1	Complete访问	Complete访问	BOOL	仅FALSE		FALSE

*1 无法利用本指令使用该结构要素。值请务必设定为FALSE。

相关指令和执行步骤

本指令根据写入至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的数据属性，有时需与其它相关指令配合执行。各种情况下的执行步骤如下所述。

● 执行步骤1

下面介绍写入带以下属性的数据时的步骤。

- 带断电保存属性
- 在重启单元时反映数值

1 🕮 使用 “NX_ChangeWriteMode指令(P.2–849)”，将对象单元变更成可写入数据的模式。

2 使用本指令，将数据写入对象单元。

3 🕮 使用 “NX_SaveParam指令(P.2–854)”，保存已写入的数据。

4 🕮 使用 “RestartNXUnit指令(P.2–844)”，重启对象单元。

● 执行步骤2

下面介绍写入带以下属性的数据时的步骤。

- 带断电保存属性
- 写入后即反映数值

1 使用本指令，将数据写入对象单元。

2 🕮 使用 “NX_SaveParam指令(P.2–854)”，保存已写入的数据。

● 执行步骤3

下面介绍写入带以下属性的数据时的步骤。

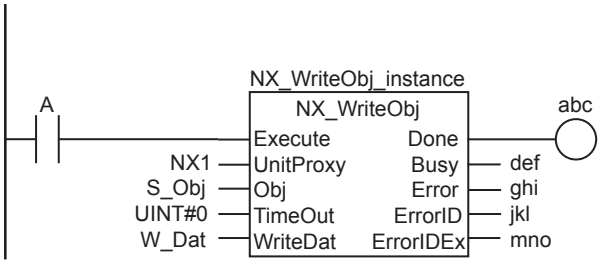
- 无断电保存属性

1 **使用本指令，将数据写入对象单元。**

记述示例

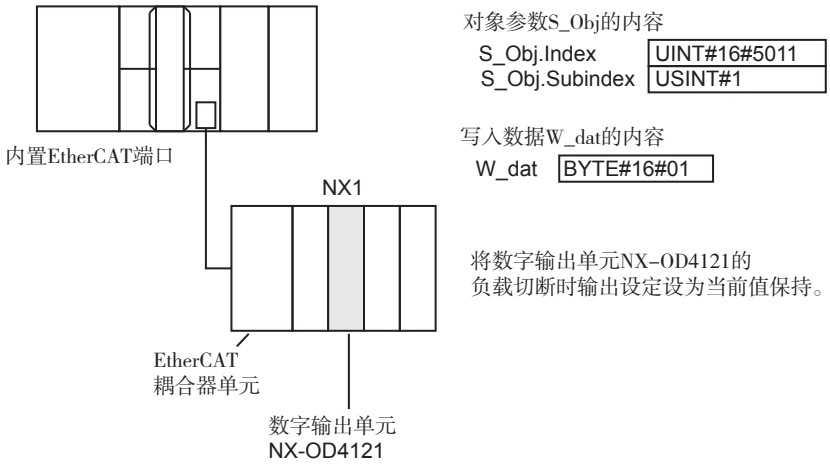
将数字输出单元NX-OD4121的负载切断时输出设定设为当前值保持的示例如下所示。
设要写入数据的单元中分配了_sNXUNIT_ID型的 “NX1”变量。
此外，NX-OD4121的负载切断时输出设定的索引为UINT#16#5011，子索引为USINT#1。
为将负载切断时输出设定设为当前值保持而写入的值为BYTE#16#01。

LD



ST

```
NX_WriteObj_instance(A, NX1, S_Obj, UINT#0,
                     W_Dat, abc, def, ghi, jkl, mno);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信
_NXB_UnitMsgActiveTbl[i]	NX单元信息可通信状态	BOOL	可进行信息通信的从站一览。 用于确认是否可与相应从站进行通信。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- “WriteDat”为整个数组时，请将整个数组的大小设定成与指定单元写入目标的NX对象相同的大小。
- 在 “UnitProxy”中指定通过Sysmac Studio的I/O映射分配至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元、与CPU单元上的NX总线连接的NX单元的设备变量。分配设备变量的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。
- 请务必将传输至 “WriteDat”的参数设为变量。如果传输常数，编连时会发生异常。
- 需要写入并保存具有断电保存属性的数据时，请执行本指令后再执行 “NX_SaveParam指令(P.2-854)”。未执行NX_SaveParam指令而重启对象单元时，NX对象的数据将恢复为写入前的值。
- 本指令是与NX信息通信异常相关的指令。一次性执行多个与NX信息通信异常相关的指令时，可能会发生NX信息通信异常。与NX信息通信异常相关的指令请参阅 “与NX信息通信异常相关的指令(P.A-186)”。
- 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0400	16#00000000	· “UnitProxy”的值超过有效范围。 · “TimeOut”的值超过有效范围。
16#0419	16#00000000	· “UnitProxy”的数据类型错误。 · “WriteDat”的数据类型错误。
16#041B	16#00000000	“WriteDat”中指定了超过2048字节的数据。
16#2C00	16#00000401	指定的单元非本指令的对象。
	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。
	16#00001101	指定的单元不正确。 请确认单元。
	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。
	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000 ~ 0x6FFF或0x7000 ~ 0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。
	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。
	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ~ 16#8FFF0000 16#90010000 ~ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。
	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C01	16#00000000	超过了可同时执行的指令数。
16#2C02	16#00000000	通信过程中发生了超时。
16#2C03	16#00000000	发送信息的大小错误。

版本相关信息

本指令可用于CPU单元Ver.1.05 以上且Sysmac Studio Ver.1.06 以上。

示例程序

- **将带断电保存属性且在单元重启时反映数值的数据写入NX单元的示例**

将与EtherCAT耦合器单元连接的AC输入单元NX-AD2203的对象参数 “Ch1 输入移动平均时间”设定成500μs。
设EtherCAT耦合器单元的节点地址为10。

对象参数 “Ch1 输入移动平均时间”的规格如下所述。

项目	值
索引	16#5004
子索引	16#01
与500μs对应的设定数据	2

对象参数 “Ch1 输入移动平均时间”是带断电保存属性且在单元重启时反映数值的数据。因此，处理步骤如下所述。

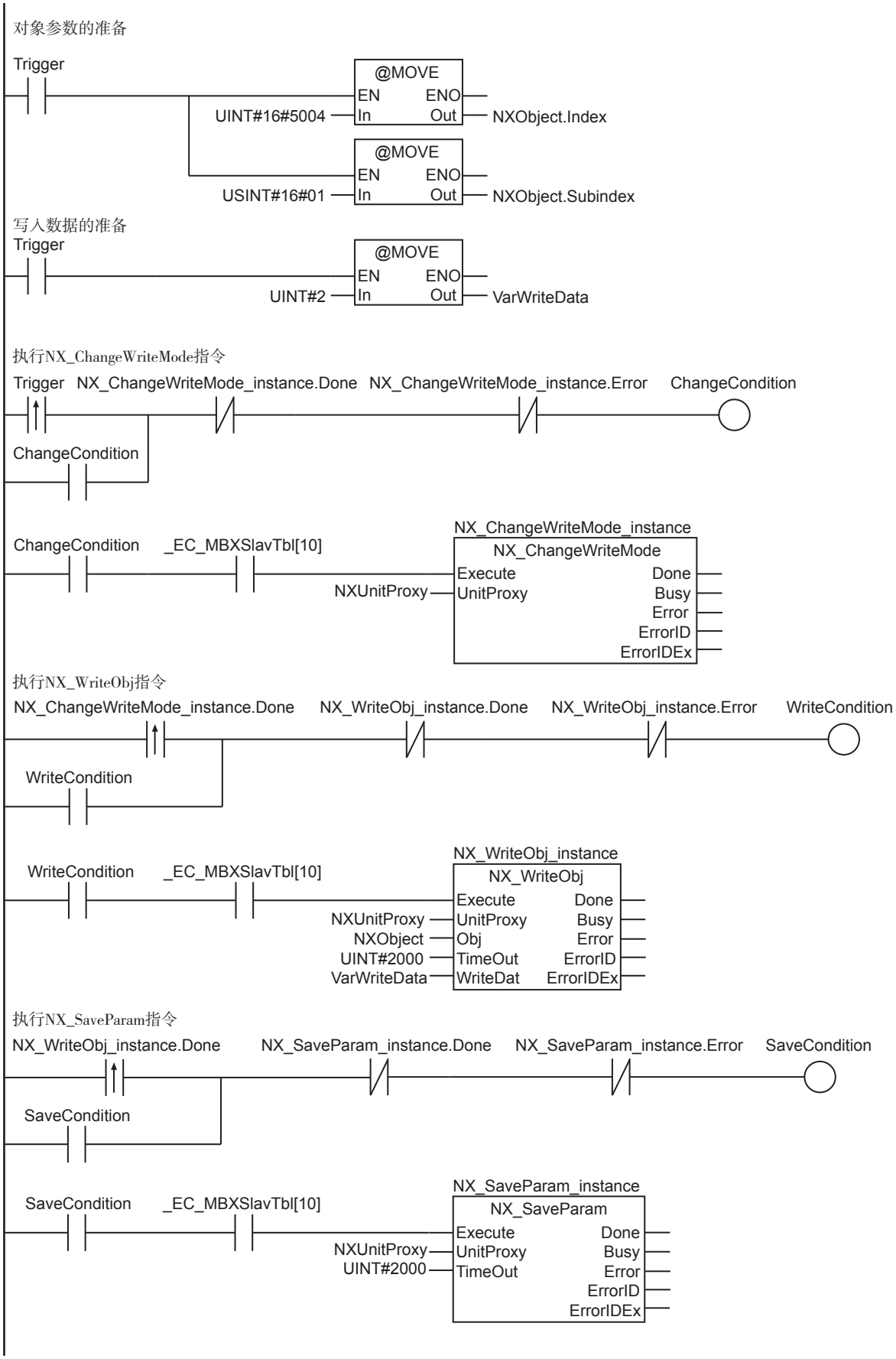
1. **使用NX_ChangeWriteMode指令，将单元变更成可写入数据的模式。**
2. **使用NX_WriteObj指令，将数据写入单元。**
3. **使用NX_SaveParam指令，保存已写入的数据。**
4. **使用RestartNXUnit指令，重启单元。**

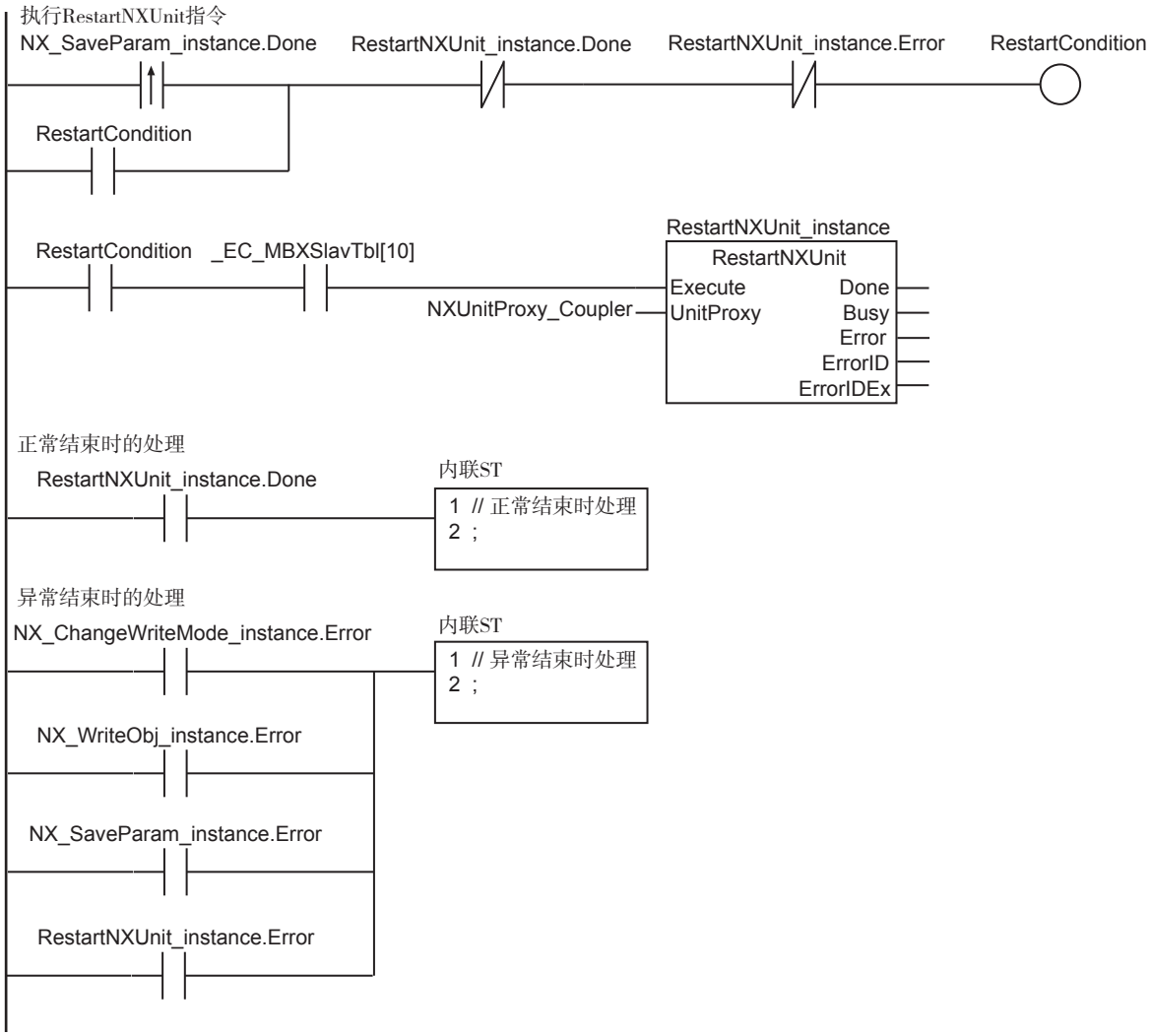
LD

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	ChangeCondition	BOOL	FALSE	写入模式变更执行条件
	WriteCondition	BOOL	FALSE	数据写入执行条件
	SaveCondition	BOOL	FALSE	数据保存执行条件
	RestartCondition	BOOL	FALSE	单元重启执行条件
	NXUnitProxy	_sNXUNIT_ID		单元指定(DC输入单元)
	NXUnitProxy_Coupler	_sNXUNIT_ID		单元指定(EtherCAT耦合器单元)
	NXObject	_sNXOBJ_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess :=FALSE)	对象参数
	VarWriteData	UINT	0	写入数据
	NX_ChangeWriteMode _instance	NX_ChangeWriteMode		
	NX_WriteObj_instance	NX_WriteObj		
	NX_SaveParam_instance	NX_SaveParam		
	RestartNXUnit_instance	RestartNXUnit		

外部变量	名称	常数	数据类型	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL *1	可进行信息通信的从站表

*1 NJ系列 CPU单元时为 “ARRAY [1..192] OF BOOL”。


对象参数的准备
Trigger
@MOVE
EN
ENO
UINT#16#5004
In
Out
NXObject.Index
@MOVE
EN
ENO
USINT#16#01
In
Out
NXObject.Subindex
写入数据的准备
Trigger
@MOVE
EN
ENO
UINT#2
In
Out
VarWriteData
执行NX_ChangeWriteMode指令
Trigger
NX_ChangeWriteMode_instance.Done
NX_ChangeWriteMode_instance.Error
ChangeCondition
ChangeCondition
ChangeCondition
_EC_MBXSlavTbl[10]
NX_ChangeWriteMode_instance
NX_ChangeWriteMode
Execute
Done
NXUnitProxy
UnitProxy
Busy
Error
ErrorID
ErrorIDEx
执行NX_WriteObj指令
NX_ChangeWriteMode_instance.Done
NX_WriteObj_instance.Done
NX_WriteObj_instance.Error
WriteCondition
WriteCondition
WriteCondition
_EC_MBXSlavTbl[10]
NX_WriteObj_instance
NX_WriteObj
Execute
Done
NXUnitProxy
UnitProxy
Busy
NXObject
Obj
Error
UINT#2000
TimeOut
ErrorID
VarWriteData
WriteDat
ErrorIDEx
执行NX_SaveParam指令
NX_WriteObj_instance.Done
NX_SaveParam_instance.Done
NX_SaveParam_instance.Error
SaveCondition
SaveCondition
SaveCondition
_EC_MBXSlavTbl[10]
NX_SaveParam_instance
NX_SaveParam
Execute
Done
NXUnitProxy
UnitProxy
Busy
UINT#2000
TimeOut
Error
ErrorID
ErrorIDEx


执行RestartNXUnit指令
NX_SaveParam_instance.Done
RestartNXUnit_instance.Done
RestartNXUnit_instance.Error
RestartCondition
RestartCondition
RestartCondition
_EC_MBXSlavTbl[10]
RestartNXUnit_instance
RestartNXUnit
Execute
Done
NXUnitProxy_Coupler
UnitProxy
Busy
Error
ErrorID
ErrorIDEx
正常结束时的处理
RestartNXUnit_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时的处理
NX_ChangeWriteMode_instance.Error
内联ST
1 // 异常结束时处理
2 ;
NX_WriteObj_instance.Error
NX_SaveParam_instance.Error
RestartNXUnit_instance.Error

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	ChangeCondition	BOOL	FALSE	写入模式变更执行条件
	ChangeGo	BOOL	FALSE	写入模式变更执行
	WriteCondition	BOOL	FALSE	数据写入执行条件
	WriteGo	BOOL	FALSE	数据写入执行
	SaveCondition	BOOL	FALSE	数据保存执行条件
	SaveGo	BOOL	FALSE	数据保存执行
	RestartCondition	BOOL	FALSE	单元重启执行条件
	RestartGo	BOOL	FALSE	单元重启执行
	NXUnitProxy	_sNXUNIT_ID		单元指定(DC输入单元)
	NXUnitProxy_Coupler	_sNXUNIT_ID		单元指定(EtherCAT耦合器单元)
	NXObject	_sNXOBJ_ACCESS	(Index:=0, Subindex:=0, IsComplete Access:=FALSE)	对象参数
	VarWriteData	UINT	0	写入数据
	NormalEnd	UINT	0	正常结束
	ErrorEnd	UINT	0	异常结束
	NX_ChangeWriteMode _instance	NX_ChangeWriteMode		
	NX_WriteObj_instance	NX_WriteObj		
	NX_SaveParam_instance	NX_SaveParam		
	RestartNXUnit_instance	RestartNXUnit		
	R_Trig_instance	R_TRIG		

外部变量	名称	常数	数据类型	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL *1	可进行信息通信的从站表

*1 NJ系列 CPU单元时为 “ARRAY [1..192] OF BOOL”。

```
// 对象参数和写入数据的准备
R_Trig_instance(Clk := Trigger);
IF (R_Trig_instance.Q=TRUE)THEN
  NXObject.Index    := UINT#16#5004;
  NXObject.Subindex := USINT#1;
  VarWriteData      := UINT#2;
END_IF;

// 执行NX_ChangeWriteMode指令
IF (Trigger = TRUE) THEN
  ChangeCondition := TRUE;
END_IF;

IF ((NX_ChangeWriteMode_instance.Done=TRUE) OR (NX_ChangeWriteMode_instance.Error=TRUE))THEN
  ChangeCondition := FALSE;
END_IF;

ChangeGo := ChangeCondition & _EC_MBXSlavTbl[10];
NX_ChangeWriteMode_instance(
  Execute   := ChangeGo,
  UnitProxy := NXUnitProxy);
```

```
// 执行NX_WriteObj指令
IF (NX_ChangeWriteMode_instance.Done=TRUE) THEN
  WriteCondition := TRUE;
END_IF;

IF ((NX_WriteObj_instance.Done=TRUE) OR (NX_WriteObj_instance.Error=TRUE))THEN
  WriteCondition := FALSE;
END_IF;

WriteGo := WriteCondition & _EC_MBXSlavTbl[10];
NX_WriteObj_instance(
  Execute   := WriteGo,
  UnitProxy := NXUnitProxy,
  Obj       := NXObject,
  TimeOut   := UINT#2000,
  WriteDat  := VarWriteData);

// 执行NX_SaveParam
IF (NX_WriteObj_instance.Done=TRUE) THEN
  SaveCondition := TRUE;
END_IF;

IF ((NX_SaveParam_instance.Done=TRUE) OR (NX_SaveParam_instance.Error=TRUE))THEN
  SaveCondition := FALSE;
END_IF;

SaveGo := SaveCondition & _EC_MBXSlavTbl[10];
NX_SaveParam_instance(
  Execute   := SaveGo,
  UnitProxy := NXUnitProxy,
  TimeOut   := UINT#2000);

// 执行RestartNXUnit
IF (NX_SaveParam_instance.Done=TRUE) THEN
  RestartCondition := TRUE;
END_IF;

IF ((RestartNXUnit_instance.Done=TRUE) OR (RestartNXUnit_instance.Error=TRUE)) THEN
  RestartCondition := FALSE;
END_IF;

RestartGo := RestartCondition & _EC_MBXSlavTbl[10];
RestartNXUnit_instance(
  Execute   := SaveGo,
  UnitProxy := NXUnitProxy_Coupler);

IF (RestartNXUnit_instance.Done=TRUE) THEN
   // 正常结束时处理
   NormalEnd := NormalEnd + UINT#1;
ELSIF ((NX_ChangeWriteMode_instance.Error=TRUE) OR (NX_WriteObj_instance.Error=TRUE)
      OR (NX_SaveParam_instance.Error=TRUE) OR (RestartNXUnit_instance.Error=TRUE)) THEN
  // 异常结束时处理
  ErrorEnd := ErrorEnd + UINT#1;
END_IF;
```

- **将带断电保存属性且在写入后立即反映数值的数据写入NX单元的示例**

将与EtherCAT耦合器单元连接的温度输入单元NX-TS2101的对象参数“Ch1 偏置值(1点补偿)”设定成+0.3℃。设EtherCAT耦合器单元的节点地址为10。

对象参数 “Ch1 偏置值(1点补偿)”的规格如下所述。

项目	值
索引	16#5010
子索引	16#01
要写入的值	0.3

对象参数 “Ch1 偏置值(1点补偿)”是带断电保存属性且在写入后立即反映数值的数据。因此，处理步骤如下所述。

1 **使用NX_WriteObj，将数据写入对象单元。**

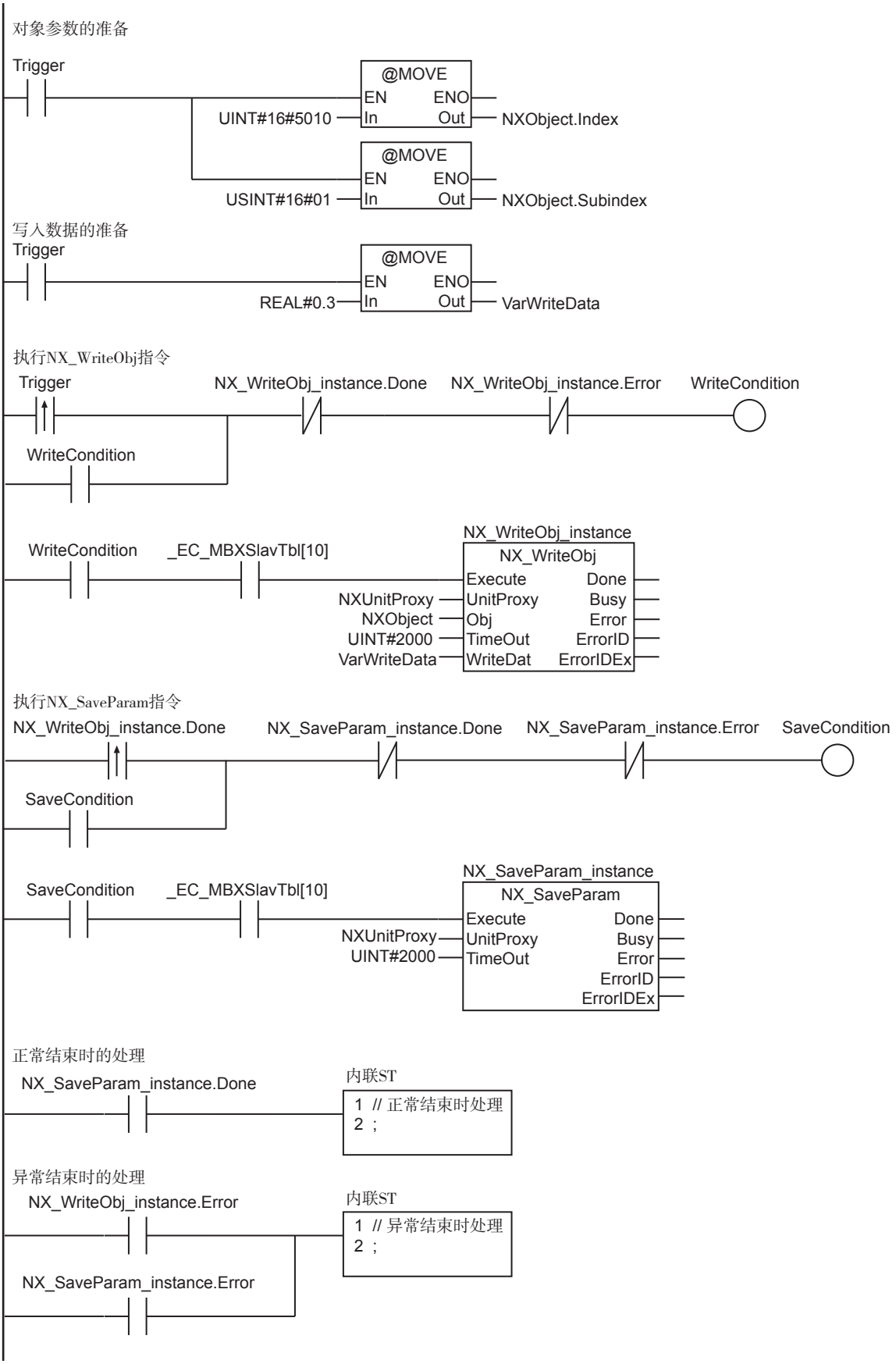
2 **使用NX_SaveParam，保存已写入的数据。**

LD

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	WriteCondition	BOOL	FALSE	数据写入执行条件
	SaveCondition	BOOL	FALSE	数据保存执行条件
	NXUnitProxy	_sNXUNIT_ID		单元指定(AC输入单元)
	NXUnitProxy_Coupler	_sNXUNIT_ID		单元指定(EtherCAT耦合器单元)
	NXObject	_sNXOBJ_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess :=FALSE)	对象参数
	VarWriteData	Real	0.0	写入数据
	NX_WriteObj_instance	NX_WriteObj		
	NX_SaveParam_instance	NX_SaveParam		

外部变量	名称	常数	数据类型	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL *1	可进行信息通信的从站表

*1 NJ系列 CPU单元时为 “ARRAY [1..192] OF BOOL”。


对象参数的准备
Trigger
@MOVE
EN
ENO
UINT#16#5010
In
Out
NXObject.Index
@MOVE
EN
ENO
USINT#16#01
In
Out
NXObject.Subindex
写入数据的准备
Trigger
@MOVE
EN
ENO
REAL#0.3
In
Out
VarWriteData
执行NX_WriteObj指令
Trigger
NX_WriteObj_instance.Done
NX_WriteObj_instance.Error
WriteCondition
WriteCondition
WriteCondition
_EC_MBXSlavTbl[10]
NX_WriteObj_instance
NX_WriteObj
Execute
Done
NXUnitProxy
UnitProxy
Busy
NXObject
Obj
Error
UINT#2000
TimeOut
ErrorID
VarWriteData
WriteDat
ErrorIDEx
执行NX_SaveParam指令
NX_WriteObj_instance.Done
NX_SaveParam_instance.Done
NX_SaveParam_instance.Error
SaveCondition
SaveCondition
SaveCondition
_EC_MBXSlavTbl[10]
NX_SaveParam_instance
NX_SaveParam
Execute
Done
NXUnitProxy
UnitProxy
Busy
UINT#2000
TimeOut
Error
ErrorID
ErrorIDEx
正常结束时的处理
NX_SaveParam_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时的处理
NX_WriteObj_instance.Error
内联ST
1 // 异常结束时处理
2 ;
NX_SaveParam_instance.Error

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	WriteCondition	BOOL	FALSE	数据写入执行条件
	WriteGo	BOOL	FALSE	数据写入执行
	SaveCondition	BOOL	FALSE	数据保存执行条件
	SaveGo	BOOL	FALSE	数据保存执行
	NXUnitProxy	_sNXUNIT_ID		单元指定(温度输入单元)
	NXUnitProxy_Coupler	_sNXUNIT_ID		单元指定(EtherCAT耦合器单元)
	NXObject	_sNXOBJ_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess :=FALSE)	对象参数
	VarWriteData	REAL	0.0	写入数据
	NormalEnd	UINT	0	正常结束
	ErrorEnd	UINT	0	异常结束
	NX_WriteObj_instance	NX_WriteObj		
	NX_SaveParam_instance	NX_SaveParam		
	R_Trig_instance	R_TRIG		

外部变量	名称	常数	数据类型	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL *1	可进行信息通信的从站表

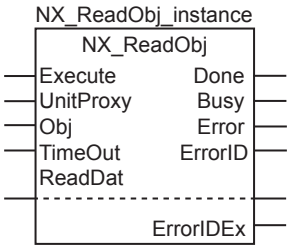
*1 NJ系列 CPU单元时为 “ARRAY [1..192] OF BOOL”。

```
// 对象参数和写入数据的准备
R_Trig_instance(Clk := Trigger);
IF (R_Trig_instance.Q=TRUE)THEN
  NXObject.Index    := UINT#16#5004;
  NXObject.Subindex := USINT#1;
  VarWriteData      := UINT#2;
END_IF;

// 执行NX_WriteObj指令
IF (Trigger=TRUE) THEN
  WriteCondition := TRUE;
END_IF;

IF ((NX_WriteObj_instance.Done=TRUE) OR (NX_WriteObj_instance.Error=TRUE))THEN
  WriteCondition := FALSE;
END_IF;

WriteGo := WriteCondition & _EC_MBXSlavTbl[10];
NX_WriteObj_instance(
  Execute   := WriteGo,
  UnitProxy := NXUnitProxy,
  Obj       := NXObject,
  TimeOut   := UINT#2000,
  WriteDat  := VarWriteData);

// 执行NX_SaveParam
IF (NX_WriteObj_instance.Done=TRUE) THEN
  SaveCondition := TRUE;
END_IF;

IF ((NX_SaveParam_instance.Done=TRUE) OR (NX_SaveParam_instance.Error=TRUE))THEN
  SaveCondition := FALSE;
END_IF;
```

```
SaveGo := SaveCondition & _EC_MBXSlavTbl[10];
NX_SaveParam_instance(
  Execute   := SaveGo,
  UnitProxy := NXUnitProxy,
  TimeOut   := UINT#2000);

IF (NX_SaveParam_instance.Done=TRUE) THEN
   // 正常结束时处理
   NormalEnd := NormalEnd + UINT#1;
ELSIF ((NX_WriteObj_instance.Error=TRUE) OR (NX_SaveParam_instance.Error=TRUE)) THEN
  // 异常结束时处理
  ErrorEnd := ErrorEnd + UINT#1;
END_IF;
```

NX_ReadObj

从EtherCAT耦合器单元或NX单元的NX对象中读取数据。

指令	名称	FB/FUN	图形表现	ST表现
NX_ReadObj	NX对象读取	FB		NX_ReadObj_instance(Execute, UnitProxy, Obj, TimeOut, ReadDat, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitProxy	指定单元	输入	要读取数据的单元	–	–	(*)
Obj	对象参数		对象参数			–
TimeOut	超时时间		超时时间 0时为2.0s	0 ~ 60000	ms	2000 (2.0s)
ReadDat	读取数据	输入输出	已从NX对象中读取的数据	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编译时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitProxy	结构体_sNXUNIT_ID 详情参阅功能说明																			
Obj	结构体_sNXOBJ_ACCESS 详情参阅功能说明																			
TimeOut							○													
ReadDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定整个数组																			

功能

从EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元以及与CPU单元上的NX总线连接的NX单元的NX对象中读取并保存在 “ReadDat”中。
通过 “UnitProxy”指定要读取数据的单元。

超时时间由 “TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。此时，不会读取数据。

“UnitProxy”的数据类型为结构体_sNXUNIT_ID。规格如下所示。

变量	名称	内容	数据类型
UnitProxy	指定单元	要指定的单元	_sNXUNIT_ID
NodeAdr	节点地址	通信耦合器单元的节点地址	UINT
IPAdr	IP地址	通信耦合器单元的IP地址	BYTE[5]
UnitNo	单元编号	要指定单元的单元编号	UDINT
Path	路径	至要指定单元的路径信息	BYTE[64]
PathLength	“Path”的有效数据长度	“Path”的有效数据长度	USINT

请将分配至要指定单元的设备变量传输至 “UnitProxy”。

“Obj”的数据类型为结构体_sNXOBJ_ACCESS。规格如下所示。

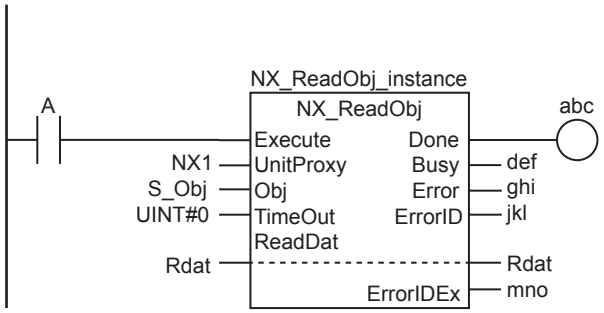
变量	名称	内容	数据类型	有效范围	单位	初始值
Obj	对象参数	对象参数	_sNXOBJ_ACCESS	–	–	–
Index	索引	索引	UINT	遵从数据类型	–	0
Subindex	子索引	子索引	USINT			
IsComplete Access*1	Complete访问	Complete访问	BOOL	仅FALSE		FALSE

*1 无法利用本指令使用该结构要素。值请务必设定为FALSE。

记述示例

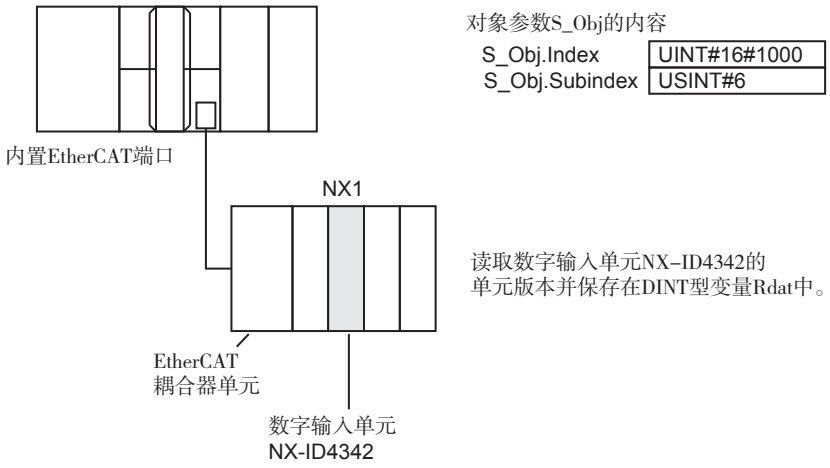
读取数字输入单元NX–ID4342的单元版本时的示例如下所示。
读取的数据保存在UDINT型变量Rdat中。
设要读取数据的单元中分配了_sNXUNIT_ID型的 “NX1”变量。
此外，NX–ID4342的单元版本的索引为UINT#16#1000，子索引为USINT#6。

LD



ST

```
NX_ReadObj_instance(A, NX1, S_Obj, UINT#0,
                    Rdat, abc, def, ghi, jkl, mno);
```



对象参数S_Obj的内容

S_Obj.Index	UINT#16#1000
S_Obj.Subindex	USINT#6

读取数字输入单元NX–ID4342的
单元版本并保存在DINT型变量Rdat中。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl[i] i为节点地址	可进行信息通信的从站表	BOOL	对应的从站表示可否通信。 TRUE ：可通信 FALSE：无法通信
_NXB_UnitMsgActiveTbl[i]	NX单元信息可通信状态	BOOL	可进行信息通信的从站一览。 用于确认是否可与相应从站进行通信。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- “ReadDat”为整个数组时，请将整个数组的大小设定成与指定单元读取源的NX对象相同的大小。
- 在 “UnitProxy”中指定通过Sysmac Studio的I/O映射分配至EtherCAT耦合器单元、EtherCAT耦合器单元上的NX单元、与CPU单元上的NX总线连接的NX单元的设备变量。分配设备变量的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。
- 本指令是与NX信息通信异常相关的指令。一次性执行多个与NX信息通信异常相关的指令时，可能会发生 NX 信息通信异常。与 NX 信息通信异常相关的指令请参阅 “与 NX 信息通信异常相关的指令(P.A-186)”。
- 发生异常时，“Error”将变为TRUE。“ErrorID”、“ErrorIDEx”的值及与其对应的异常内容如下所述。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#0400	16#00000000	· “UnitProxy”的值超过有效范围。 · “TimeOut”的值超过有效范围。
16#0410	16#00000000	“ReadDat”为STRING型，且未以NULL字符结尾。
16#0419	16#00000000	· “UnitProxy”的数据类型错误。 · “ReadDat”的数据类型错误。
16#041C	16#00000000	“ReadDat”的大小与读取源NX对象的大小不符。
16#2C00	16#00000401	指定的单元非本指令的对象。
	16#00001001 16#00001002 16#00170000 16#00200000 16#00210000	输入参数、输出参数、输入输出参数错误。 请确认输入参数、输出参数、输入输出参数是否为预期值。
	16#00001010	指定的NX对象的数据大小与 “WriteDat”的数据大小不符。
	16#00001101	指定的单元不正确。 请确认单元。
	16#0000110B	要读取的数据过大。 请确认要读取数据的指定是否正确。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#00001110	不存在与 “Obj.Index”的值对应的对象。
	16#00001111	不存在与 “Obj.Subindex”的值对应的对象。
	16#00002101	写入目标的NX对象无法写入。
	16#00002110	“WriteDat”的值超过了写入目标NX对象值的范围。
	16#00002210	指定的单元并非可写入数据的模式。
	16#00002213	指定的单元正在执行I/O检查功能，因此无法执行本指令。 请结束I/O检查功能后执行本指令。
	16#00002230	指定单元的状态与读取源或写入目标NX对象的值不符。 “Obj.Index”的值为0x6000 ~ 0x6FFF或0x7000 ~ 0x7FFF时，请采取以下措施。 · 请从I/O分配设定中取消读取源或写入目标NX对象。 · 请解除指定单元的异常。 · 请在可写入数据的模式下解除指定的单元。
	16#00002231	指定的单元正在执行初始处理，因此无法执行本指令。 请等待至动作正常后，再执行本指令。
	16#0000250F	硬件访问失败。 请重新执行本指令。
	16#00002601 16#00002602 16#00100000	指定的单元无法使用本指令。 请确认单元的版本。
	16#00002603	本指令执行失败。 请重新执行本指令。 请确认使用通道选择中是否至少有1个通道设定了 “有效”。
	16#00002621	NX单元未处于接收本指令的状态。 请稍作等待后重新执行本指令。
	16#00010000	指定的单元不存在。 请确认单元的构成是否正确。
	16#00110000	指定的端口No.不存在。 请确认单元的构成是否正确。
	16#00120000 16#00130000 16#00150000 16#00160000	“UnitProxy”的值错误。 请重新设定表示要指定的EtherCAT耦合器单元的变量。
	16#00140000	指定的节点地址错误。 请确认单元的构成是否正确。
	16#00300000 16#80010000	指定的单元为BUSY状态。 请重新执行本指令。
	16#00310000	指定的单元非连接对象。 请确认单元的版本。
	16#80000000 16#80050000 16#81010000 16#81020000 16#82020000 16#82030000 16#82060000 ~ 16#8FFF0000 16#90010000 ~ 16#FFFE0000	通信网络发生异常。 请重新执行本指令。

“ErrorID”的值	“ErrorIDEx”的值	异常内容
16#2C00	16#80020000 16#80030000 16#81030000 16#82000000	通信网络发生异常。 请降低通信负载。
	16#80040000 16#81000000 16#82010000 16#82040000 16#82050000 16#90000000	通信网络发生异常。 请确认单元及电缆的连接。 请确认单元的电源是否接通。
16#2C01	16#00000000	超过了可同时执行的指令数。
16#2C02	16#00000000	通信过程中发生了超时。

版本相关信息

本指令可用于CPU单元Ver.1.05 以上且Sysmac Studio Ver.1.06 以上。

示例程序

读取EtherCAT耦合器单元NX-ECC201的对象参数 “I/O刷新方式1”的值。
设EtherCAT耦合器单元的节点地址为10。

对象参数 “I/O刷新方式1”的索引和子索引的值如下所述。

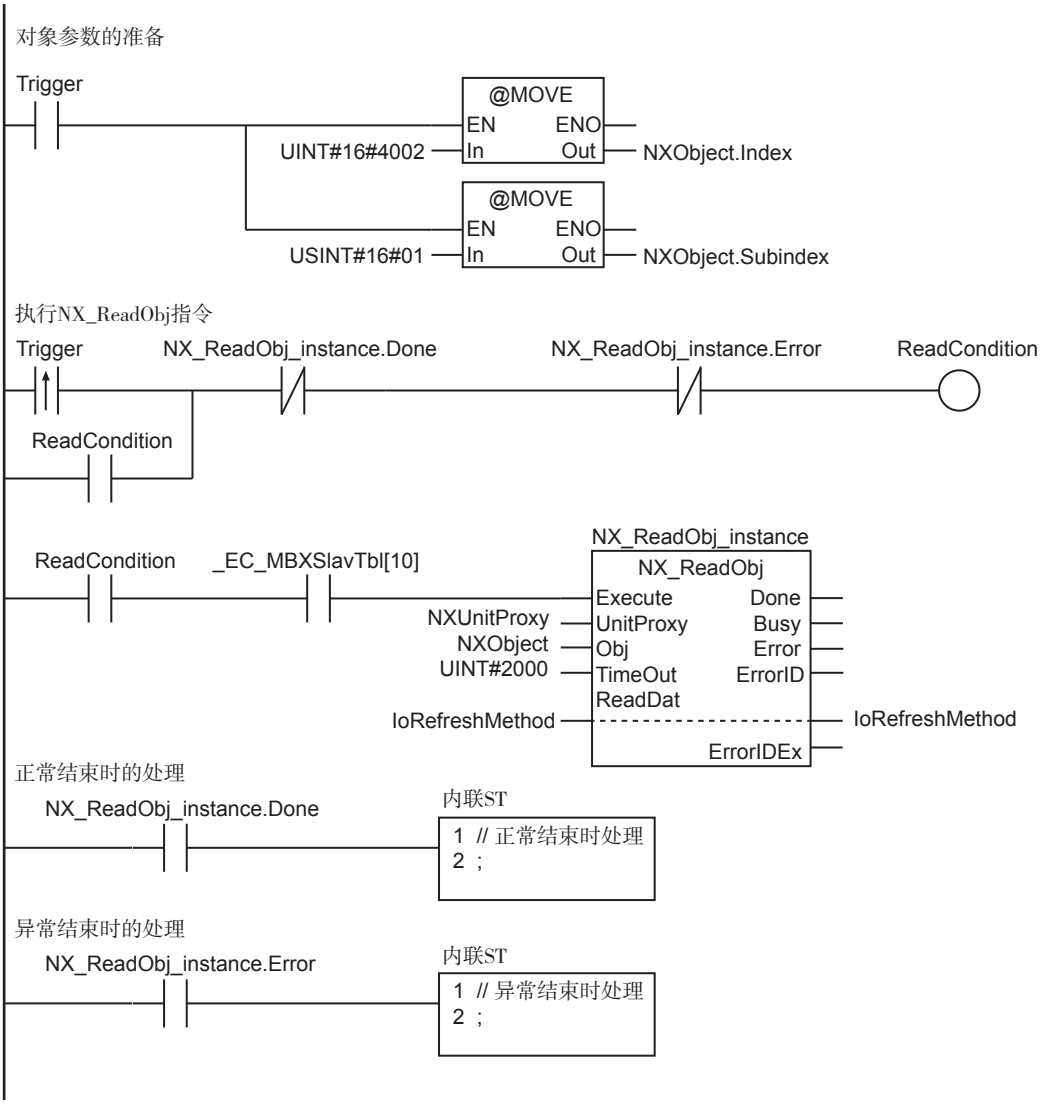
项目	值
索引	16#4002
子索引	16#01

LD

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	ReadCondition	BOOL	FALSE	数据读取执行条件
	NXUnitProxy	_sNXUNIT_ID		单元指定
	NXObject	_sNXOBJ_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess:=FALSE)	对象参数
	IoRefreshMethod	USINT	0	读取数据
	NX_ReadObj_instance	NX_ReadObj		

外部变量	名称	常数	数据类型	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL *1	可进行信息通信的从站表

*1 NJ系列 CPU单元时为 “ARRAY [1..192] OF BOOL”。


对象参数的准备
Trigger
@MOVE
EN
ENO
UINT#16#4002
In
Out
NXObject.Index
@MOVE
EN
ENO
USINT#16#01
In
Out
NXObject.Subindex
执行NX_ReadObj指令
Trigger
NX_ReadObj_instance.Done
NX_ReadObj_instance.Error
ReadCondition
ReadCondition
NX_ReadObj_instance
NX_ReadObj
ReadCondition
_EC_MBXSlavTbl[10]
Execute
Done
NXUnitProxy
UnitProxy
Busy
NXObject
Obj
Error
UINT#2000
TimeOut
ErrorID
ReadDat
IoRefreshMethod
IoRefreshMethod
ErrorIDEx
正常结束时的处理
NX_ReadObj_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时的处理
NX_ReadObj_instance.Error
内联ST
1 // 异常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	ReadCondition	BOOL	FALSE	数据读取执行条件
	ReadGo	BOOL	FALSE	数据读取执行
	NXUnitProxy	_sNXUNIT_ID		单元指定
	NXObject	_sNXOBJ_ACCESS	(Index:=0, Subindex:=0, IsCompleteAccess :=FALSE)	对象参数
	IoRefreshMethod	USINT	0	读取数据
	NormalEnd	UINT	0	正常结束
	ErrorEnd	UINT	0	异常结束
	R_Trig_instance	R_Trig		
	NX_ReadObj_instance	NX_ReadObj		

外部变量	名称	常数	数据类型	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL *1	可进行信息通信的从站表

*1 NJ系列 CPU单元时为 “ARRAY [1..192] OF BOOL”。

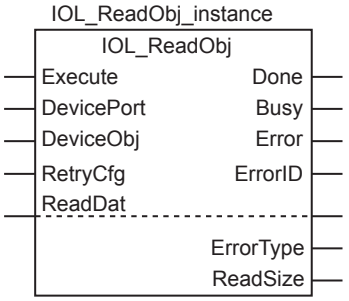
```
// 对象参数的准备
R_Trig_instance(Clk := Trigger);
IF (R_Trig_instance.Q=TRUE)THEN
  NXObject.Index    := UINT#16#4002;
  NXObject.Subindex := USINT#1;
END_IF;

// NX_ReadObj的执行
IF (Trigger=TRUE) THEN
  ReadCondition := TRUE;
END_IF;

IF ( (NX_ReadObj_instance.Done=TRUE) OR (NX_ReadObj_instance.Error=TRUE) ) THEN
  ReadCondition := FALSE;
END_IF;

ReadGo := ReadCondition & _EC_MBXSlavTbl[10];
NX_ReadObj_instance(
  Execute   := ReadGo,
  UnitProxy := NXUnitProxy,
  Obj       := NXObject,
  TimeOut   := UINT#2000,
  ReadDat   := IoRefreshMethod);

// 执行指令后的处理
IF (NX_ReadObj_instance.Done=TRUE) THEN
  // 正常结束时处理
  NormalEnd := NormalEnd + UINT#1;
ELSIF (NX_ReadObj_instance.Error=TRUE) THEN
  // 异常结束时处理
  ErrorEnd := ErrorEnd + UINT#1;
END_IF;
```

IO–Link通信指令

指令	名称	页码
IOL_ReadObj	IO–Link设备对象读取	2–974
IOL_WriteObj	IO–Link设备对象写入	2–982

IOL_ReadObj

从IO-Link设备的对象中读取数据。

指令	名称	FB/FUN	图形表现	ST表现
IOL_ReadObj	IO-Link设备对象读取	FB		IOL_ReadObj_instance(Execute, DevicePort, DeviceObj, RetryCfg, ReadDat, Done, Busy, Error, ErrorID, ErrorType, ReadSize);

版本相关信息

本指令可用于CPU单元Ver.1.12以上且Sysmac Studio Ver.1.16以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
DeviceObj	IO-Link设备的对象参数		IO-Link设备的对象指定	–	–	–
RetryCfg	执行重试设定		指令执行重试设定	–	–	–
ReadDat	读取数据	输入输出	从IO-Link设备中读取的数据	遵从数据类型	–	0
ErrorType	错误类型	输出	“ErrorID”为“4800Hex”时，保存IO-Link设备返回的错误代码	16#0000 ~ 16#FFFF	–	–
ReadSize	读取数据大小		“ReadDat”中保存的数据大小	10#1 ~ 10#232	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
DeviceObj	结构体_sIOLOBJ_ACCESS 详情参阅功能说明																			
RetryCfg	结构体_sIOL_RETRY_CFG 详情参阅功能说明																			
ReadDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定整个数组																			
ErrorType			○																	
ReadSize							○													

功能

读取IO-Link设备的对象数据。

使用输入变量 “DevicePort”指定连接读取对象IO-Link设备的IO-Link主站单元和端口编号。
输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOptionBoard	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOAD_ID	–	–	–
Reserved	保留	保留	–	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2 3：端口3 4：端口4 5：端口5 6：端口6 7：端口7 8：端口8	USINT	遵从数据类型	–	–

使用 “DeviceType”指定设备种类。IO-Link主站单元(NX型)时请设定_DeviceNXUnit，IO-Link主站单元(GX型)时请设定_DeviceEcatSlave。用于设备指定的变量取决于指定的种类。
该指令如下所述。
指定NX型时，使用 “NxUnit”指定设备。这种情况下，不使用 “EcatSlave”。
请将分配至指定设备的设备变量传输至 “NxUnit”。
指定GX型时，使用 “EcatSlave”指定设备。这种情况下，不使用 “NxUnit”。
请将分配至指定设备的设备变量传输至 “EcatSlave”。

使用 “PortNo”指定连接IO-Link设备的端口编号。
端口数因IO-Link主站单元的类型而异。
NX型：1～4
GX型：1～8

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站

使用输入变量 “DeviceObj”指定读取对象IO-Link设备的对象参数。
“DeviceObj”的数据类型为结构体_sIOLOBJ_ACCESS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DeviceObj	IO-Link设备的对象参数	IO-Link设备的对象指定	_sIOLOBJ_ACCESS	-	-	-
Index	索引	索引	UINT	遵从数据类型	-	-
Subindex	子索引	读取所有索引时指定 “0”	USINT	遵从数据类型	-	-

使用输入变量 “RetryCfg”指定指令执行的重试设定。
“RetryCfg”的数据类型为结构体_sIOL_RETRY_CFG。规格如下所示。

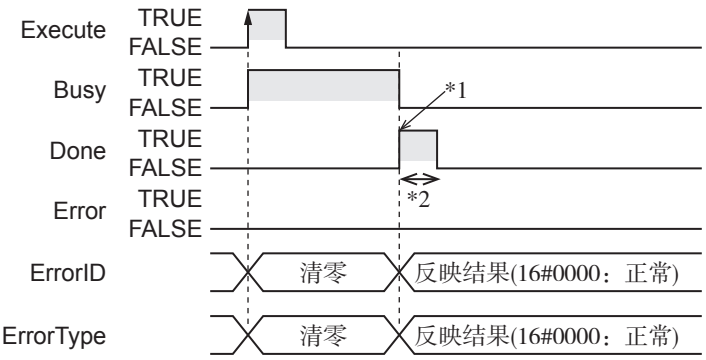
变量	名称	内容	数据类型	有效范围	单位	初始值
RetryCfg	执行重试设定	指令执行重试设定	_sIOL_RETRY_CFG	-	-	-
TimeOut	超时时间	超时时间 “0”时为2.0s	TIME	0 ~ 300s	-	T#2.0s
RetryNum	重试次数	指定发生超时时的重试次数 “0”时为3次	UINT	遵从数据类型	次	3

从IO-Link设备读取的数据保存在输入输出变量 “ReadDat”中。

时序图

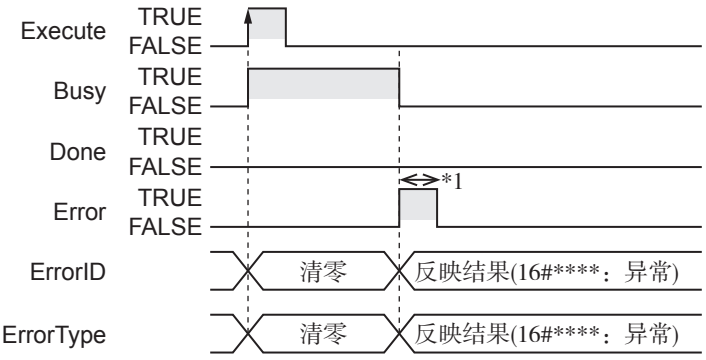
时序图如下所示。

● 正常结束时



*1 读取完成
*2 任务周期

● 异常结束时



*1 任务周期

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl	可进行信息通信的从站表	ARRAY[1..512] OF BOOL*1	可进行信息通信的从站一览。 按从站节点地址顺序在表中进行表示。 TRUE：可通信 FALSE：无法通信

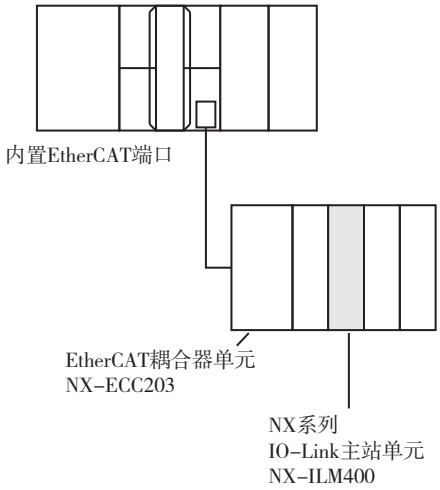
*1 NJ系列 CPU单元时，为ARRAY [1..192] OF BOOL。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在“DevicePort.NxUnit”、“DevicePort.EcatSlave”中指定通过Sysmac Studio的I/O映射分配至IO-Link主站单元的设备变量。分配设备变量的方法请参阅 “Sysmac StudioVersion1 操作手册(SBCA-362G以上)”。
- 请指定 “ReadDat”大于实际读取的对象大小的变量。
- “ReadDat”为STRING型时，请指定实际读取的字符串加上NULL字符大小的变量。
- “ReadDat”为STRING型时，“ReadSize”中将输出不含NULL字符的大小。
- 请务必将传输至 “ReadDat”的参数设为变量。如果传输常数，编连时会发生异常。
- 1个IO-Link主站单元可同时执行的指令数，NX型和GX型均为1个。
- 事件任务中无法使用。编译时会发生错误。
- 本指令在 “Execute”的上升沿动作。“Execute”始终为TRUE时不执行。
- IOL_ReadObj指令及IOL_WriteObj指令可使用的实例数最多为64个。
- 以下情况时会发生异常。
  - “DevicePort.NxUnit”、“DevicePort.EcatSlave”中指定了超出范围的值。
  - 读取源IO-Link设备的对象大小大于 “ReadDat”的大小时。发生本异常时，“ReadDat”中不会保存读取的数据。
  - 接收到IO-Link设备发出的错误响应时。
    高8位为 “ErrorCode”，低8位为 “AdditionalCode”。
    关于“ErrorCode”、“AdditionalCode”，请确认IO-Link Communication Specification的Error type规格。
    关于Error type规格，可从IO-Link Consortium获取。
    http://www.io-link.com/
  - 指定的IO-Link主站单元不存在时。
  - IO-Link 主站的信息处理数超限时。IO-Link 主站正在处理来自其它应用程序的信息，因此无法执行。
  - 指定的IO-Link主站单元不处于接收信息的状态时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。
  - 通信过程中发生了超时时。
  - IO-Link主站单元的指定端口非IO-Link模式时。端口无效或SIO模式时。
  - IO-Link主站单元的指定端口未连接IO-Link设备时。
  - IO-Link主站单元的指定端口未接通IO电源时。
  - IO-Link主站单元的指定端口发生了核查异常或通信异常时。

示例程序

采用EtherCAT耦合器单元NX-ECC203连接IO-Link主站单元NX-ILM400的构成。



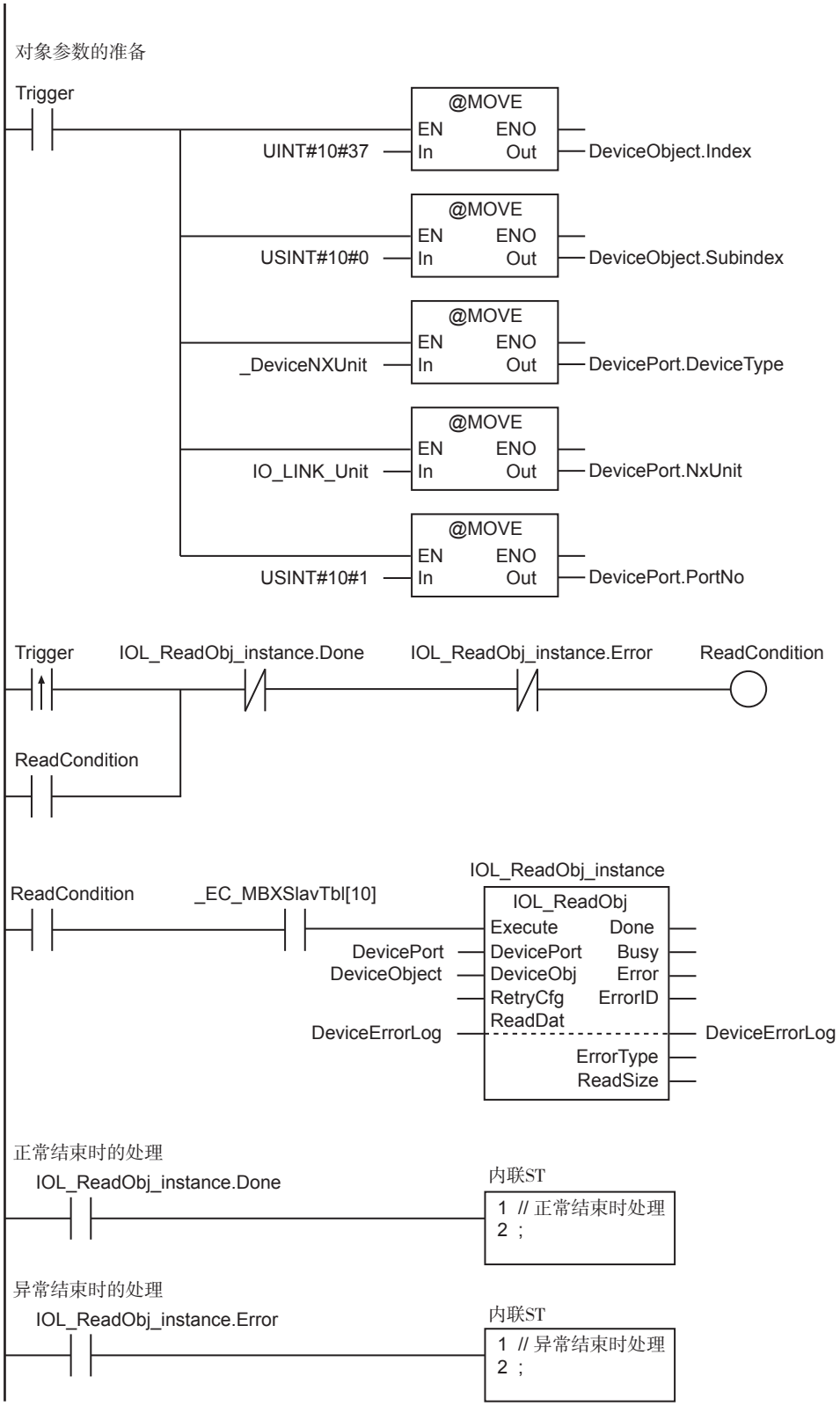
读取NX-ILM400的端口1连接的光电传感器(E3Z)的错误日志(Index:37/Subindex:0) 30byte。读取的数据保存至 “DeviceErrorLog”。
NX-ECC203的节点地址设为10。

LD

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	ReadCondition	BOOL	FALSE	数据读取执行条件
	DevicePort	_sDEVICE_PORT		
	DeviceObject	_sIOLOBJ_ACCESS	(Index:=0, Subindex:=0)	IO-Link设备的对象指定
	DeviceErrorLog	ARRAY[1..30] OF BYTE		读取数据
	IOL_ReadObj_instance	IOL_ReadObj		

外部变量	名称	常数	初始值	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL*1	可进行信息通信的从站表
	IO_LINK_Unit	☑	作为初始值，事先在结构体结构要素 “NxUnit”中设定指定NX-ILM400的设备变量	

*1 NJ系列 CPU单元时，为ARRAY[1..192] OF BOOL。


对象参数的准备
Trigger
@MOVE
EN ENO
UINT#10#37 In Out DeviceObject.Index
@MOVE
EN ENO
USINT#10#0 In Out DeviceObject.Subindex
@MOVE
EN ENO
_DeviceNXUnit In Out DevicePort.DeviceType
@MOVE
EN ENO
IO_LINK_Unit In Out DevicePort.NxUnit
@MOVE
EN ENO
USINT#10#1 In Out DevicePort.PortNo
Trigger
IOL_ReadObj_instance.Done
IOL_ReadObj_instance.Error
ReadCondition
ReadCondition
IOL_ReadObj_instance
ReadCondition
_EC_MBXSlavTbl[10]
IOL_ReadObj
Execute Done
DevicePort DevicePort Busy
DeviceObject DeviceObj Error
RetryCfg ErrorID
ReadDat
DeviceErrorLog DeviceErrorLog
ErrorType
ReadSize
正常结束时的处理
IOL_ReadObj_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时的处理
IOL_ReadObj_instance.Error
内联ST
1 // 异常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	ReadGo	BOOL	FALSE	数据读取执行
	DevicePort	_sDEVICE_PORT		
	DeviceObject	_sIOLOBJ_ACCESS	(Index:=0, Subindex:=0)	IO-Link设备的对象指定
	DeviceErrorLog	ARRAY[1..30] OF BYTE		读取数据
	NormalEnd	UINT	0	正常结束
	ErrorEnd	UINT	0	异常结束
	R_Trig_instance	R_Trig		
	IOL_ReadObj_instance	IOL_ReadObj		

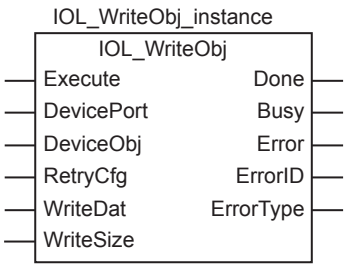
外部变量	名称	常数	初始值	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL*1	可进行信息通信的从站表
	IO_LINK_Unit	☑	作为初始值，事先在结构体结构要素 “NxUnit”中设定指定NX-ILM400的设备变量	

*1 NJ系列 CPU单元时，为ARRAY[1..192] OF BOOL。

```
// 对象参数的准备
R_Trig_instance(Clk := Trigger);
IF (R_Trig_instance.Q=TRUE)THEN
    DeviceObject.Index := UINT#10#37;
    DeviceObject.Subindex := USINT#0;
    DevicePort.DeviceType:= _eDEVICE_TYPE#_DeviceNXUnit;
    DevicePort.NxUnit:= IO_LINK_Unit;
    DevicePort.PortNo:= USINT#10#1;
    IF ( _EC_MBXSlavTbl[10] =TRUE)THEN
        ReadGo := TRUE;
    END_IF;
END_IF;

IF ( (IOL_ReadObj_instance.Done=TRUE) OR (IOL_ReadObj_instance.Error=TRUE) ) THEN
    ReadGo := FALSE;
END_IF;

//执行IOL_ReadObj指令
IOL_ReadObj_instance(
    Execute := ReadGo,
    DevicePort:= DevicePort,
    DeviceObj := DeviceObject,
    ReadDat :=DeviceErrorLog);

// 执行指令后的处理
IF (IOL_ReadObj_instance.Done=TRUE) THEN
    // 正常结束时处理
    NormalEnd := NormalEnd + UINT#1;
ELSIF (IOL_ReadObj_instance.Error=TRUE) THEN
    // 异常结束时处理
    ErrorEnd := ErrorEnd + UINT#1;
END_IF;
```

IOL_WriteObj

在IO-Link设备的对象中写入数据。

指令	名称	FB/FUN	图形表现	ST表现
IOL_WriteObj	IO-Link设备对象写入	FB		IOL_WriteObj_instance(Execute, DevicePort, DeviceObj, RetryCfg, WriteDat, WriteSize, Done, Busy, Error, ErrorID, ErrorType);

版本相关信息

本指令可用于CPU单元Ver.1.12以上且Sysmac Studio Ver.1.16以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
DeviceObj	IO-Link设备的对象参数		IO-Link设备的对象指定	–	–	–
RetryCfg	执行重试设定		指令执行重试设定	–	–	–
WriteDat	写入数据		写入IO-Link设备的数据	遵从数据类型	–	–
WriteSize	写入数据大小		写入数据大小*1	10#1 ~ 10#232	字节	–
ErrorType	错误类型	输出	“ErrorID”为“4800Hex”时，保存IO-Link设备返回的错误代码	16#0000 ~ 16#FFFF	–	–

*1 写入数据为BOOL型时请输入“1”，BOOL型数组时请输入数组的元素数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
DeviceObj	结构体_sIOLOBJ_ACCESS 详情参阅功能说明																			
RetryCfg	结构体_sIOL_RETRY_CFG 详情参阅功能说明																			
WriteDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定整个数组																			
WriteSize							○													
ErrorType			○																	

功能

写入IO-Link设备的对象数据。

使用输入变量 “DevicePort”指定连接写入对象IO-Link设备的IO-Link主站单元和端口编号。
输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOptionBoard	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOAD_ID	–	–	–
Reserved	保留	保留	–	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2 3：端口3 4：端口4 5：端口5 6：端口6 7：端口7 8：端口8	USINT	遵从数据类型	–	–

使用 “DeviceType”指定设备种类。IO-Link主站单元(NX型)时请设定_DeviceNXUnit，IO-Link主站单元(GX型)时请设定_DeviceEcatSlave。用于设备指定的变量取决于指定的种类。
该指令如下所述。
指定NX型时，使用 “NxUnit”指定设备。这种情况下，不使用 “EcatSlave”。
请将分配至指定设备的设备变量传输至 “NxUnit”。
指定GX型时，使用 “EcatSlave”指定设备。这种情况下，不使用 “NxUnit”。
请将分配至指定设备的设备变量传输至 “EcatSlave”。

使用 “PortNo”指定连接IO-Link设备的端口编号。
端口数因IO-Link主站单元的类型而异。
NX型：1～4
GX型：1～8

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站

使用输入变量 “DeviceObj”指定写入对象 IO-Link 设备的对象参数。“DeviceObj”的数据类型为结构体_sIOLOBJ_ACCESS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DeviceObj	IO-Link设备的对象参数	IO-Link设备的对象指定	_sIOLOBJ_ACCESS	-	-	-
Index	索引	索引	UINT	遵从数据类型	-	-
Subindex	子索引	读取所有索引时指定 “0”	USINT	遵从数据类型	-	-

使用输入变量 “RetryCfg”指定指令执行的重试设定。
“RetryCfg”的数据类型为结构体_sIOL_RETRY_CFG。规格如下所示。

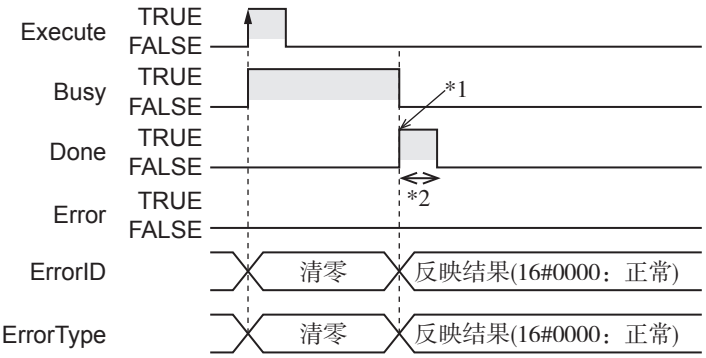
变量	名称	内容	数据类型	有效范围	单位	初始值
RetryCfg	执行重试设定	指令执行重试设定	_sIOL_RETRY_CFG	-	-	-
TimeOut	超时时间	超时时间 “0”时为2.0s	TIME	0 ~ 300s	-	T#2.0s
RetryNum	重试次数	指定发生超时时的重试次数 “0”时为3次	UINT	遵从数据类型	次	3

使用输入变量 “WriteDat”指定写入IO-Link设备的数据。

时序图

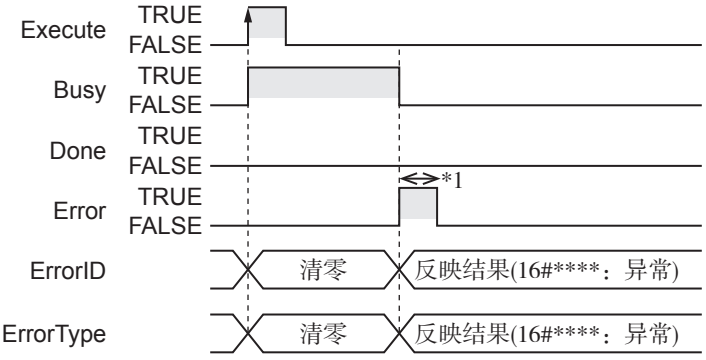
时序图如下所示。

● 正常结束时



*1 写入完成
*2 任务周期

● 异常结束时



*1 任务周期

相关的系统定义变量

变量名称	名称	数据类型	内容
_EC_MBXSlavTbl	可进行信息通信的从站表	ARRAY[1..512] OF BOOL*1	可进行信息通信的从站一览。 按从站节点地址顺序在表中进行表示。 TRUE：可通信 FALSE：无法通信

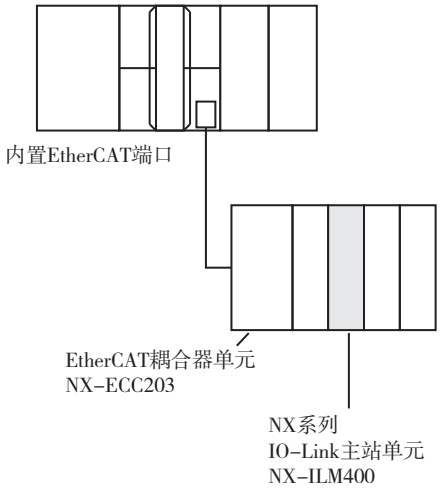
*1 NJ系列 CPU单元时，为ARRAY[1..192] OF BOOL。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在“DevicePort.NxUnit”、“DevicePort.EcatSlave”中指定通过Sysmac Studio的I/O映射分配至IO-Link主站单元的设备变量。分配设备变量的方法请参阅 “Sysmac StudioVersion1 操作手册(SBCA-362G 以上)”。
- 请务必将传输至 “WriteDat”的参数设为变量。如果传输常数，编连时会发生异常。
- 1个IO-Link主站单元可同时执行的指令数，NX型和GX型均为1个。
- 事件任务中无法使用。编译时会发生错误。
- 本指令在 “Execute”的上升沿动作。“Execute”始终为TRUE时不执行。
- IOL_ReadObj指令及IOL_WriteObj指令可使用的实例数最多为64个。
- 以下情况时会发生异常。
  - “DevicePort.NxUnit”、“DevicePort.EcatSlave”中指定了超出范围的值。
  - “TimeOut”超过有效范围时。
  - “DevicePort”的数据类型错误时。
  - “WriteDat”中指定了超过232字节的数据时。
  - 接收到IO-Link设备发出的错误响应时。
    高8位为 “ErrorCode”，低8位为 “AdditionalCode”。
    关于“ErrorCode”、“AdditionalCode”，请确认IO-Link Communication Specification的Error type规格。
    关于Error type规格，可从IO-Link Consortium获取。
    http://www.io-link.com/
  - 指定的IO-Link主站单元不存在时。
  - IO-Link主站的信息处理数超限时。IO-Link主站正在处理来自其它应用程序的信息，因此无法执行。
  - 指定的IO-Link主站单元不处于接收信息的状态时。
  - 同时执行EC_CoESDOWrite指令、EC_CoESDORead指令、EC_StartMon指令、EC_StopMon指令、EC_SaveMon指令、EC_CopyMon指令、EC_DisconnectSlave指令、EC_ConnectSlave指令、EC_ChangeEnableSetting指令、IOL_ReadObj指令、IOL_WriteObj指令超过32个时。
  - 通信过程中发生了超时时。
  - IO-Link主站单元的指定端口非IO-Link模式时。端口无效或SIO模式时。
  - IO-Link主站单元的指定端口未连接IO-Link设备时。
  - IO-Link主站单元的指定端口未接通IO电源时。
  - IO-Link主站单元的指定端口发生了核查异常或通信异常时。

示例程序

采用EtherCAT耦合器单元NX-ECC203连接IO-Link主站单元NX-ILM400的构成。



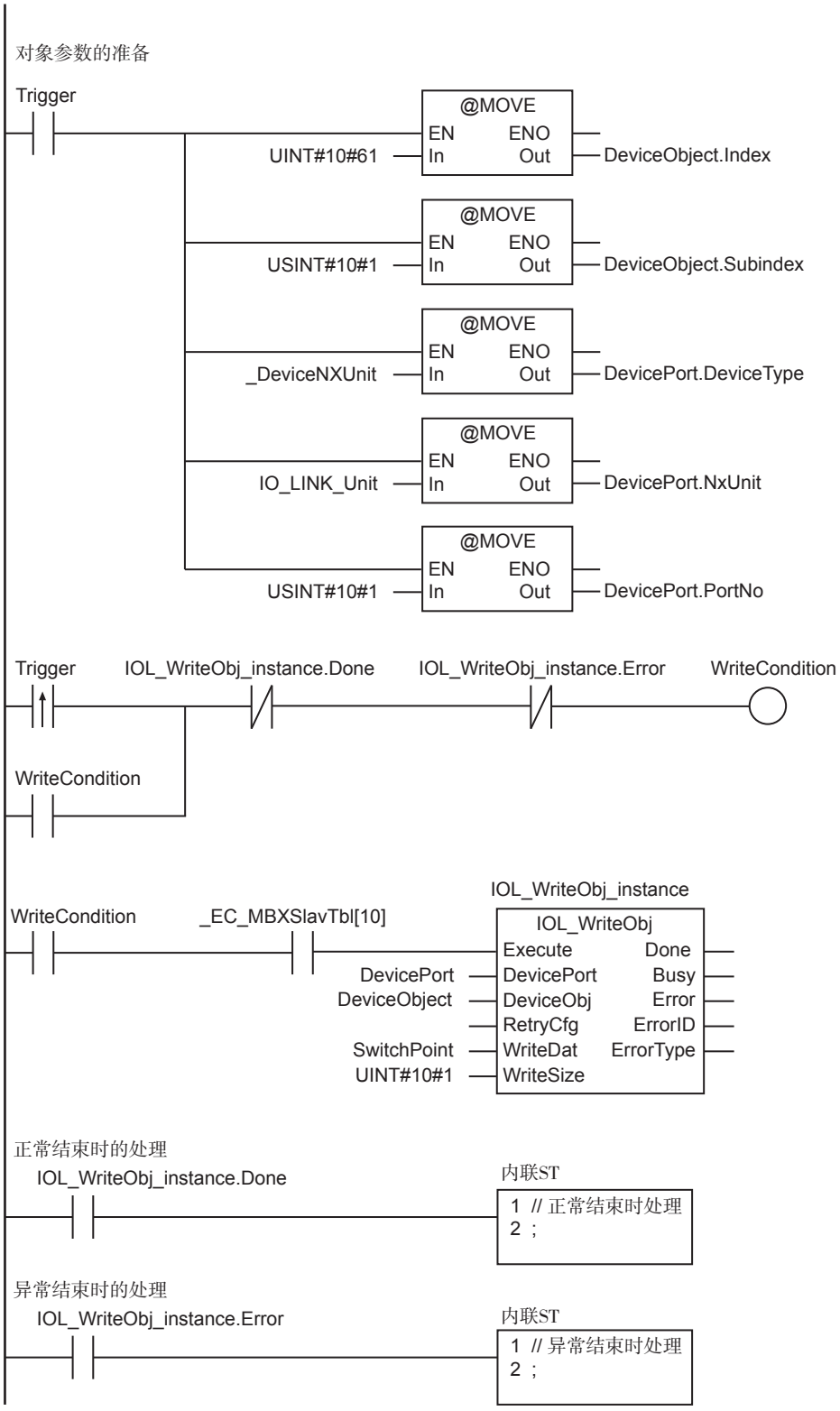
在NX-ILM400的端口1连接的光电传感器(E3Z)的控制输出1动作模式切换(SwitchPoint Logic Output1)(Index:61/Subindex:1)1byte中写入 “01”。写入数据保存在 “SwitchPoint” 中。
NX-ECC203的节点地址设为10。

LD

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	WriteCondition	BOOL	FALSE	数据写入执行条件
	DevicePort	_sDEVICE_PORT		
	DeviceObject	_sIOLOBJ_ACCESS	(Index:=0, Subindex:=0)	IO-Link设备的对象指定
	SwitchPoint	USINT	USINT#01	写入数据
	IOL_WriteObj_instance	IOL_WriteObj		

外部变量	名称	常数	初始值	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL*1	可进行信息通信的从站表
	IO_LINK_Unit	☑	作为初始值，事先在结构体结构要素 “NxUnit” 中设定指定NX-ILM400的设备变量	

*1 NJ系列 CPU单元时，为ARRAY[1..192] OF BOOL。


对象参数的准备
Trigger
@MOVE
EN ENO
UINT#10#61 In Out DeviceObject.Index
@MOVE
EN ENO
USINT#10#1 In Out DeviceObject.Subindex
@MOVE
EN ENO
_DeviceNXUnit In Out DevicePort.DeviceType
@MOVE
EN ENO
IO_LINK_Unit In Out DevicePort.NxUnit
@MOVE
EN ENO
USINT#10#1 In Out DevicePort.PortNo
Trigger
IOL_WriteObj_instance.Done
IOL_WriteObj_instance.Error
WriteCondition
WriteCondition
IOL_WriteObj_instance
WriteCondition
_EC_MBXSlavTbl[10]
IOL_WriteObj
Execute Done
DevicePort DevicePort Busy
DeviceObject DeviceObj Error
RetryCfg ErrorID
SwitchPoint WriteDat ErrorType
UINT#10#1 WriteSize
正常结束时的处理
IOL_WriteObj_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时的处理
IOL_WriteObj_instance.Error
内联ST
1 // 异常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	WriteGo	BOOL	FALSE	数据写入执行
	DevicePort	_sDEVICE_PORT		
	DeviceObject	_sIOLOBJ_ACCESS	(Index:=0, Subindex:=0)	IO-Link设备的对象指定
	SwitchPoint	USINT	USINT#01	写入数据
	NormalEnd	UINT	0	正常结束
	ErrorEnd	UINT	0	异常结束
	R_Trig_instance	R_Trig		
	IOL_WriteObj_instance	IOL_WriteObj		

外部变量	名称	常数	初始值	注释
	_EC_MBXSlavTbl	☑	ARRAY[1..512] OF BOOL*1	可进行信息通信的从站表
	IO_LINK_Unit	☑	作为初始值，事先在结构体结构要素 “NxUnit”中设定指定NX-ILM400的设备变量	

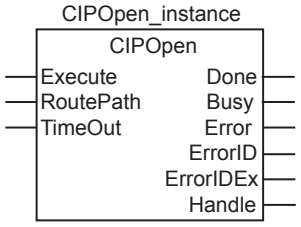
*1 NJ系列 CPU单元时，为ARRAY[1..192] OF BOOL。

```
// 对象参数的准备
R_Trig_instance(Clk := Trigger);
IF (R_Trig_instance.Q=TRUE)THEN
    DeviceObject.Index := UINT#10#61;
    DeviceObject.Subindex := USINT#1;
    DevicePort.DeviceType:= _eDEVICE_TYPE#_DeviceNXUnit;
    DevicePort.NxUnit:= IO_LINK_Unit;
    DevicePort.PortNo:= USINT#10#1;
    IF ( _EC_MBXSlavTbl[10] =TRUE)THEN
        WriteGo := TRUE;
    END_IF;
END_IF;

IF ( (IOL_WriteObj_instance.Done=TRUE) OR (IOL_WriteObj_instance.Error=TRUE) ) THEN
    WriteGo := FALSE;
END_IF;

//执行IOL_WriteObj指令
IOL_WriteObj_instance(
        Execute := WriteGo,
        DevicePort:= DevicePort,
        DeviceObj := DeviceObject,
        WriteDat := SwitchPoint,
        WriteSize := UINT#10#1);

// 执行指令后的处理
IF (IOL_WriteObj_instance.Done=TRUE) THEN
    // 正常结束时处理
    NormalEnd := NormalEnd + UINT#1;
ELSIF (IOL_WriteObj_instance.Error=TRUE) THEN
    // 异常结束时处理
    ErrorEnd := ErrorEnd + UINT#1;
END_IF;
```

EtherNet/IP通信指令

CIPOpen

按1994字节的数据长度，为指定对象节点建立CIP的Class3(Large_Forward_Open)连接。

指令	名称	FB/FUN	图形表现	ST表现
CIPOpen	CIP Class3 (Large _Forward _Open)连接建立	FB		CIPOpen_instance(Execute, RoutePath, TimeOut, Done, Busy, Error, ErrorID, ErrorIDEx, Handle);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
RoutePath	路径	输入	路径	遵从数据类型	–	–
TimeOut	超时时间		连接超时时间	1 ~ 65535	0.1s	20 (2s)
Handle	句柄	输出	句柄	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
RoutePath																				○
TimeOut							○													
Handle	结构体_sCIP_HANDLE　详情参阅功能说明																			

功能

按1994字节的数据长度，对路径 “RoutePath”指定的CIP网络上的对象节点，建立CIP的Class3(Large_Forward_Open) 连接。
连接建立后，输出句柄 “Handle”。

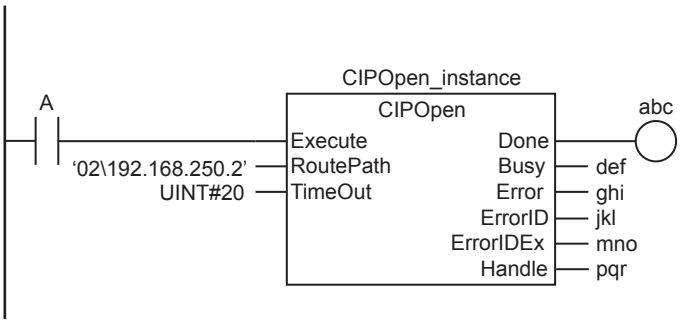
连接超时时间由 “TimeOut”指定。执行CIPSend指令、CIPWrite指令、CIPRead指令中的任一指令后，在连接超时时间内如对象节点未返回响应，则判断为通信失败。执行CIPRead指令、CIPWrite指令、CIPSend指令，连接超时时间将在对象节点返回响应时复位。

“Handle”的数据类型为结构体_sCIP_HANDLE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Handle	句柄	句柄	_sCIP_HANDLE	–	–	–
Handle	句柄	句柄	UDINT	遵从数据类型	–	–

“RoutePath”=‘02\192.168.250.2’、“TimeOut”=UINT#20时的示例如下所示。对IP地址‘192.168.250.2’的对象节点，建立CIP的Class3(Large_Forward_Open) 连接。超时时间为2s。句柄代入变量pqr。

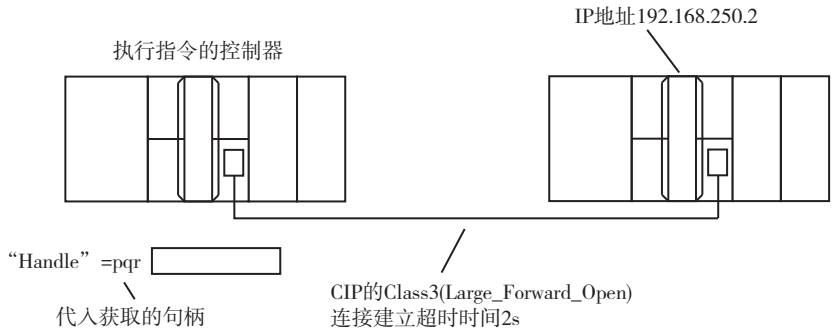
LD



ST

```
CIPOpen_instance(A, '2\192.168.250.2', UINT#20, abc,
def, ghi, jkl, mno, pqr);
```

对“RoutePath”指定的CIP网络上的对象节点，
建立CIP的Class3(Large_Forward_Open)连接。



“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。“ErrorIDEx”的值和含义由对象节点规定。请参阅对象节点的手册。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

- CIP通信的详情请参阅下列手册。
  - 🕮“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
  - 🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
  - 🕮“CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”
- 通过Forward Open建立连接或按任意数据长度建立连接时，请使用 🕮“CIPOpenWithDataSize指令(P.2-1001)”。

版本相关信息

CIPOpenWithDataSize指令可用于CPU单元Ver.1.06以上且Sysmac Studio Ver.1.07以上。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮“本章说明(P.2-3)”。
- 本指令或CIPOpenWithDataSize指令需在CIPRead指令、CIPWrite指令、CIPSend指令前执行。
- 通过本指令建立连接后，首次超时时间在 “TimeOut”的值小于100(10s)时，也为10s。
- 请使用CIPClose指令关闭本指令建立的连接。
- 连接超时时，仍会剩余本指令生成的句柄。因此请务必使用CIPClose指令，关闭连接。
- 本指令生成的句柄在变更程序模式时无效。
- 最多可同时生成32个句柄。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “TimeOut”的值超过有效范围时。
  - “RoutePath”的字符串错误时。
  - 同时执行了超过32个的CIP相关指令时。
  - 试图开通的连接超过CIPClass连接的资源(32个)时。
  - 连接建立未响应时。
  - 建立连接的对象节点不支持Large_Forward_Open时。
  - 发生本机IP地址的设定异常时。
  - 发生了IP重复异常时。
  - TCP连接已全部使用时。
  - 在BOOTP服务器发生异常的状态下，执行了本指令时。

版本相关信息

CPU单元Ver.1.10以上版本时，即使 “Error”变为TRUE，也不会变更 “Handle”的值。Ver.1.09以下版本时，会将 “Handle”的值变更为 “0”。

示例程序

使用CIP的Class3信息，执行变量写入、变量读取、信息发送。
控制器之间通过EtherNet/IP网络连接，对象节点的IP地址为192.168.250.2。
处理步骤如下所述。

1 使用CIPOpen指令，建立Class3(Large_Forward_Open)连接。超时时间为2s。

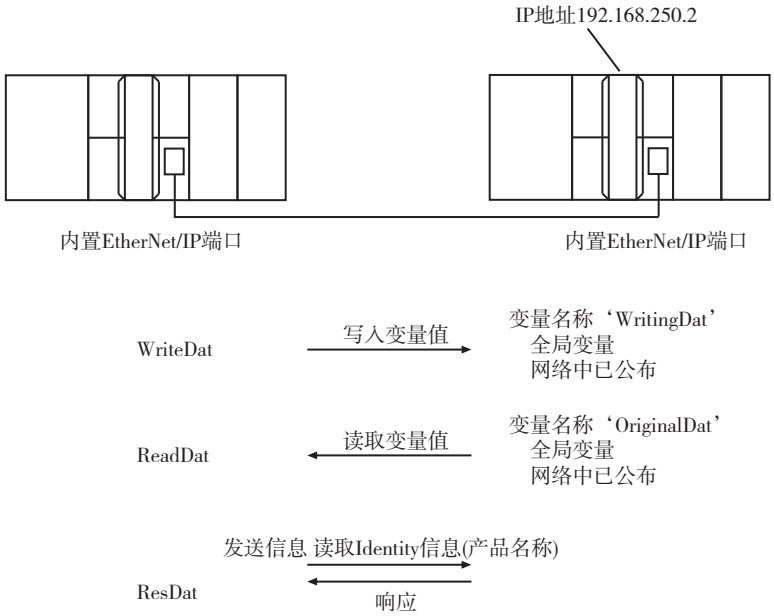
2 使用CIPWrite指令，将值写入对象节点的变量。其它站变量名称为 ‘WritingDat’，写入值为变量WriteDat的内容。‘WritingDat’需由对象节点定义为全局变量，并在网络中公布。

3 使用CIPRead指令读取对象节点的变量值。其它站变量名称为 ‘OriginalDat’，已读取值的保存位置为变量ReadDat。‘OriginalDat’需由对象节点定义为全局变量，并在网络中公布。

4 使用CIPSend指令，将Explicit信息发送至对象节点。信息内容为读取的Identity信息(产品名称)。此时的等级ID、实例ID、属性ID、服务代码如下所示。响应数据保存在变量ResDat中。

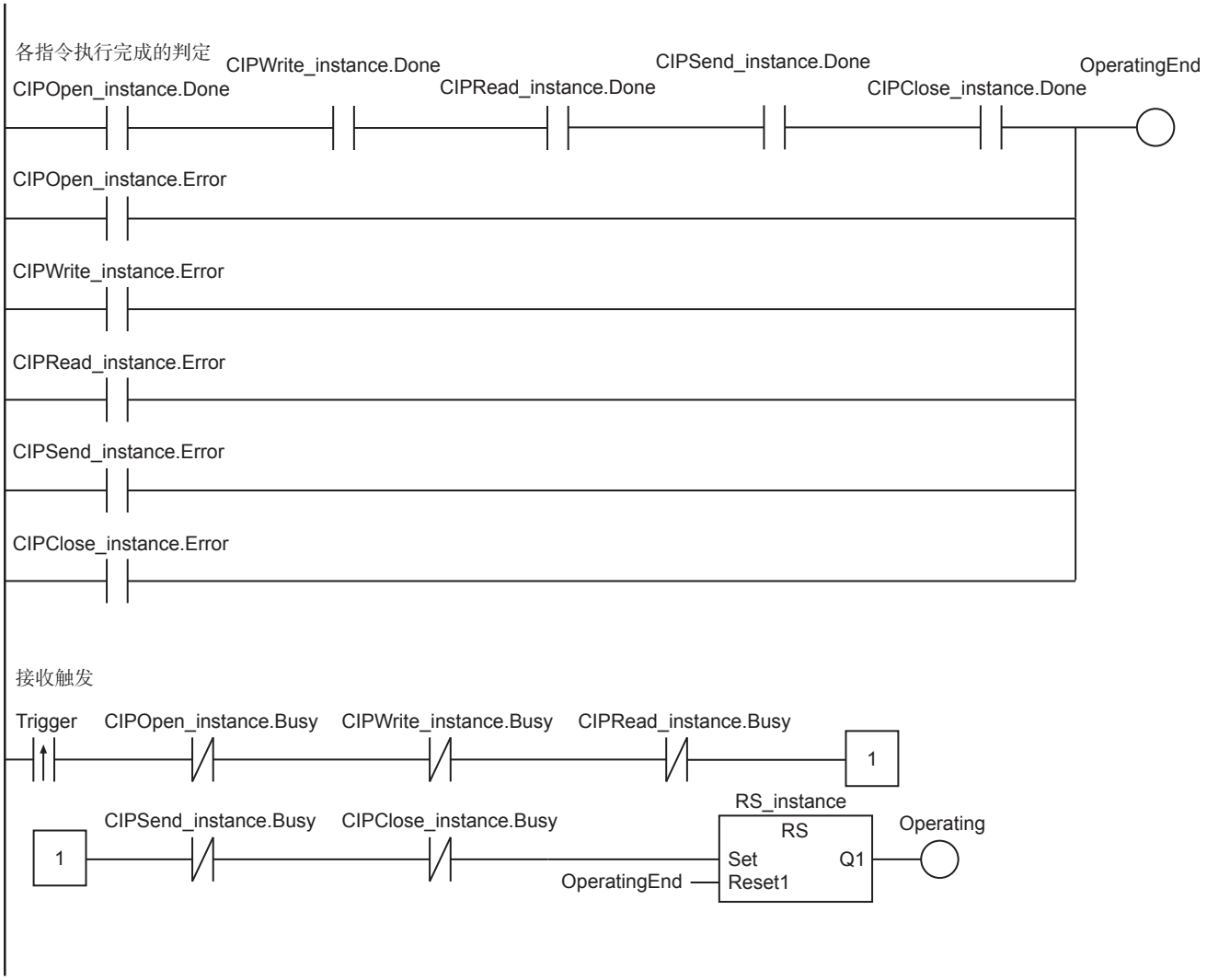
项目	值
等级ID	1
实例ID	1
属性ID	7
服务代码	16#0E

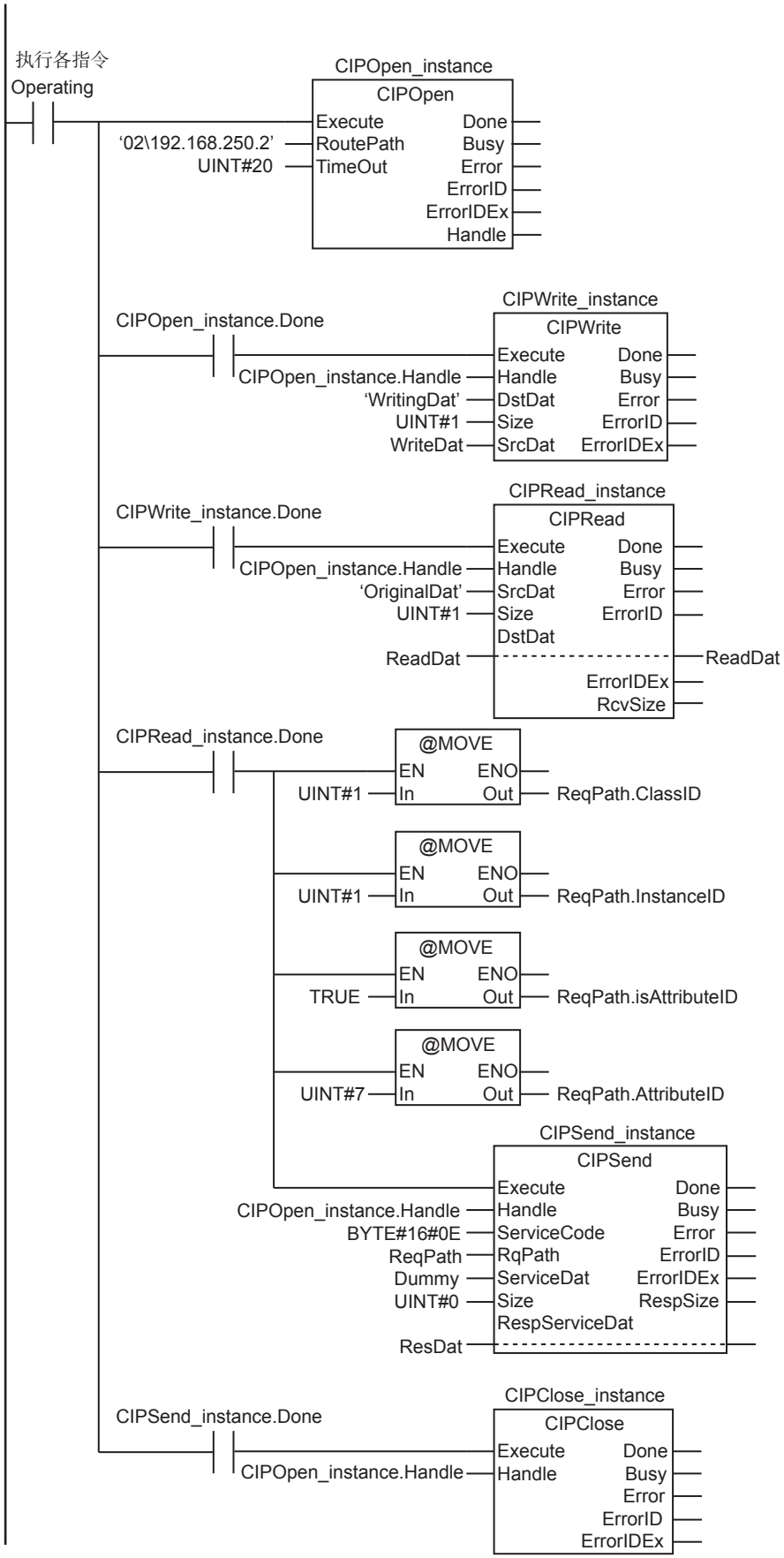
5 使用CIPClose指令，切断Class3连接。

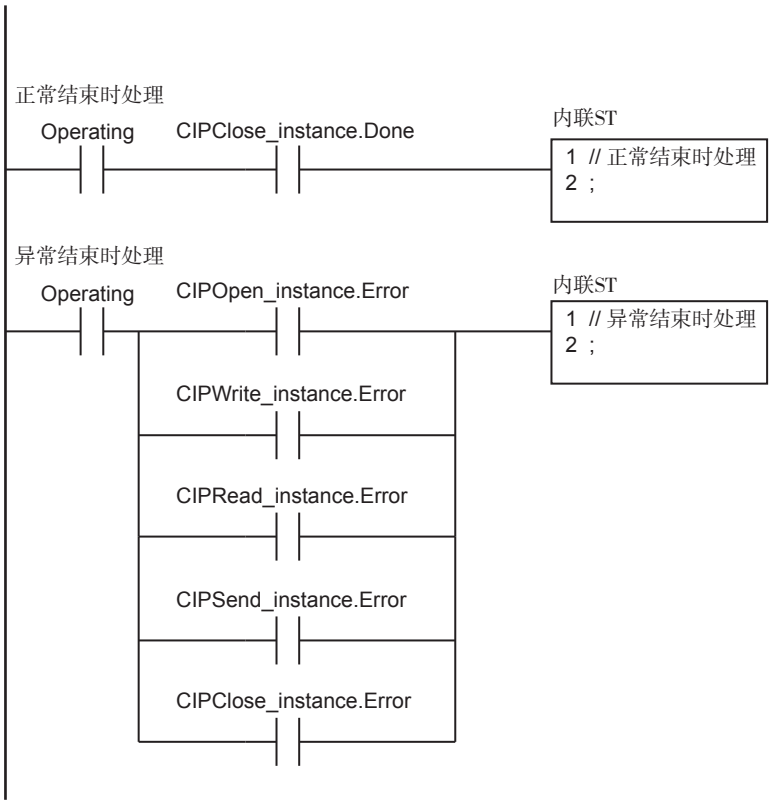


LD

名称	数据类型	初始值	注释
OperatingEnd	BOOL	FALSE	处理结束
Trigger	BOOL	FALSE	执行条件
Operating	BOOL	FALSE	处理中
WriteDat	INT	1234	写入数据
ReadDat	INT	0	读取数据
ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=FALSE, AttributeID:=0)	请求路径
ResDat	ARRAY[0..10] OF BYTE	[11(16#0)]	响应数据
Dummy	BYTE	16#0	虚拟
RS_instance	RS		
CIPOpen_instance	CIPOpen		
CIPWrite_instance	CIPWrite		
CIPRead_instance	CIPRead		
CIPSend_instance	CIPSend		
CIPClose_instance	CIPClose		




执行各指令
Operating
CIPOpen_instance
CIPOpen
Execute
RoutePath
TimeOut
Done
Busy
Error
ErrorID
ErrorIDEx
Handle
'02\192.168.250.2'
UINT#20
CIPWrite_instance
CIPWrite
CIPOpen_instance.Done
CIPOpen_instance.Handle
'WritingDat'
UINT#1
WriteDat
Execute
Handle
DstDat
Size
SrcDat
Done
Busy
Error
ErrorID
ErrorIDEx
CIPRead_instance
CIPRead
CIPWrite_instance.Done
CIPOpen_instance.Handle
'OriginalDat'
UINT#1
ReadDat
Execute
Handle
SrcDat
Size
DstDat
Done
Busy
Error
ErrorID
ReadDat
ErrorIDEx
RcvSize
CIPRead_instance.Done
@MOVE
EN
In
ENO
Out
UINT#1
ReqPath.ClassID
@MOVE
UINT#1
ReqPath.InstanceID
@MOVE
TRUE
ReqPath.isAttributeID
@MOVE
UINT#7
ReqPath.AttributeID
CIPSend_instance
CIPSend
CIPOpen_instance.Handle
BYTE#16#0E
ReqPath
Dummy
UINT#0
ResDat
Execute
Handle
ServiceCode
RqPath
ServiceDat
Size
RespServiceDat
Done
Busy
Error
ErrorID
ErrorIDEx
RespSize
CIPClose_instance
CIPClose
CIPSend_instance.Done
CIPOpen_instance.Handle
Execute
Handle
Done
Busy
Error
ErrorID
ErrorIDEx


正常结束时处理
Operating
CIPClose_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating
CIPOpen_instance.Error
内联ST
1 // 异常结束时处理
2 ;
CIPWrite_instance.Error
CIPRead_instance.Error
CIPSend_instance.Error
CIPClose_instance.Error

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoCIPTrigger	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	WriteDat	INT	0	写入数据
	ReadDat	INT	0	读取数据
	ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=FALSE, AttributeID:=0)	请求路径
	ResDat	ARRAY[0..10] OF BYTE	[11(16#0)]	响应数据
	Dummy	BYTE	16#0	虚拟
	CIPOpen_instance	CIPOpen		
	CIPWrite_instance	CIPWrite		
	CIPRead_instance	CIPRead		
	CIPSend_instance	CIPSend		
	CIPClose_instance	CIPClose		

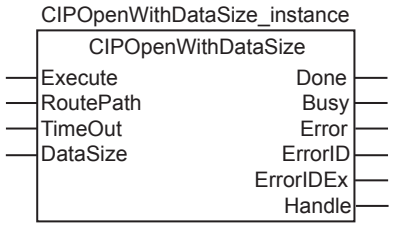
外部变量	名称	常数	数据类型	注释
	_EIP_EtnOnlineSta	☑	BOOL	在线

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoCIPTrigger=FALSE) AND (_Eip_EtnOnlineSta=TRUE) ) THEN
   DoCIPTrigger:=TRUE;
   Stage          :=INT#1;
   CIPOpen_instance(Execute:=FALSE);        // 实例初始化
   CIPWrite_instance(
      Execute     :=FALSE,                  // 实例初始化
      SrcDat      :=WriteDat);              // 虚拟
   CIPRead_instance(                        // 实例初始化
      Execute     :=FALSE,                  // 虚拟
      DstDat      :=ReadDat);               // 虚拟
   CIPSend_instance(
      Execute          :=FALSE,             // 实例初始化
      ServiceDat       := Dummy,            // 虚拟
      RespServiceDat:=ResDat);              // 虚拟
   CIPClose_instance(Execute:=FALSE);       // 实例初始化
END_IF;


IF (DoCIPTrigger=TRUE) THEN
   CASE Stage OF
   1 :                                      // CIP Class3(Large_Forward_Open)连接建立
      CIPOpen_instance(
         Execute  :=TRUE,
         TimeOut  :=UINT#20,                // 超时时间2.0s
         RoutePath:=’02\192.168.250.2’);    // 路径

      IF (CIPOpen_instance.Done=TRUE) THEN
         Stage:=INT#2;                      // 正常结束
      ELSIF (CIPOpen_instance.Error=TRUE) THEN
         Stage:=INT#10;                     // 异常结束
      END_IF;

   2 :                                      // 变量写入请求
      CIPWrite_instance(
         Execute :=TRUE,
         Handle  :=CIPOpen_instance.Handle,     // 句柄
         DstDat  :='WritingDat',                // 其它站变量名称
         Size    :=UINT#1,                      // 写入元素数
         SrcDat  :=WriteDat);                   // 写入数据
```

```
    IF (CIPWrite_instance.Done=TRUE) THEN
      Stage:=INT#3;                    // 正常结束
    ELSIF (CIPWrite_instance.Error=TRUE) THEN
      Stage:=INT#20;                   // 异常结束
    END_IF;

  3 :                                  // 变量读取请求
    CIPRead_instance(
      Execute :=TRUE,
      Handle  :=CIPOpen_instance.Handle,      // 句柄
      SrcDat  :='OriginalDat',                // 其它站变量名称
      Size    :=UINT#1,                       // 读取元素数
      DstDat  :=ReadDat);                     // 读取数据

    IF (CIPRead_instance.Done=TRUE) THEN
      Stage:=INT#4;                    // 正常结束
    ELSIF (CIPRead_instance.Error=TRUE) THEN
      Stage:=INT#30;                   // 异常结束
    END_IF;

  4 :                                  // 发送任意信息
    ReqPath.ClassID         :=UINT#01;
    ReqPath.InstanceID      :=UINT#01;
    ReqPath.isAttributeID :=TRUE;
    ReqPath.AttributeID   :=UINT#07;
    CIPSend_instance(
      Execute           :=TRUE,
      Handle            :=CIPOpen_instance.Handle,   // 句柄
      ServiceCode       :=BYTE#16#0E,                // 服务代码
      RqPath            :=ReqPath,                   // 请求路径
      ServiceDat        :=Dummy,                     // 服务数据
      Size              :=UINT#0,                    // 元素数
      RespServiceDat:=ResDat);                       // 响应数据

    IF (CIPSend_instance.Done=TRUE) THEN
      Stage:=INT#5;                    // 正常结束
    ELSIF (CIPSend_instance.Error=TRUE) THEN
      Stage:=INT#40;                   // 异常结束
    END_IF;

  5 :                                  // 请求切断CIPClass3连接
    CIPClose_instance(
      Execute :=TRUE,
      Handle  :=CIPOpen_instance.Handle);     // 句柄

    IF (CIPClose_instance.Done=TRUE) THEN
      Stage:=INT#0;
    ELSIF (CIPClose_instance.Error=TRUE) THEN
      Stage:=INT#50;
    END_IF;

  0:                                   // 正常结束处理
    DoCIPTrigger:=FALSE;
    Trigger         :=FALSE;

  ELSE                                 // 异常结束处理
    DoCIPTrigger:=FALSE;
    Trigger         :=FALSE;
  END_CASE;
END_IF;
```

CIPOpenWithDataSize

为指定对象节点建立可收发指定数据长度以下的Class3 Explicit信息的CIP Class3连接。

指令	名称	FB/FUN	图形表现	ST表现
CIPOpenWith DataSize	CIP Class3 连接建立(带大小指定)	FB		CIPOpen_instance(Execute, RoutePath, TimeOut, DataSize, Done, Busy, Error, ErrorID, ErrorIDEx, Handle);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
RoutePath	路径	输入	路径	遵从数据类型	–	–
TimeOut	超时时间		超时时间	1 ~ 65535	0.1s	20 (2s)
DataSize	数据长度		数据长度	6 ~ 8192 *1*2	字节	1994
Handle	句柄	输出	句柄	–	–	–

*1 NX1P2 CPU单元及NJ系列 CPU单元时为 “6 ~ 1994”。

*2 CPU单元Ver.1.10以下或Sysmac Studio Ver.1.14以下时，最小值为 “10”。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
RoutePath																				○
TimeOut							○													
DataSize							○													
Handle	结构体_sCIP_HANDLE 详情参阅功能说明																			

功能

对路径 “RoutePath”指定的CIP网络上的对象节点，建立CIP的Class3连接。使用数据长度 “DataSize”，指定要收发的Class3 Explicit信息的数据长度。

如下所述，Class3连接的服务取决于 “DataSize”的值。

“DataSize”的值[字节]	服务
509以下	Forward_Open
510 ~ 8192 *1	Large_Forward_Open

*1 NX1P2 CPU单元及NJ系列 CPU单元时为 “510 ~ 1994”。

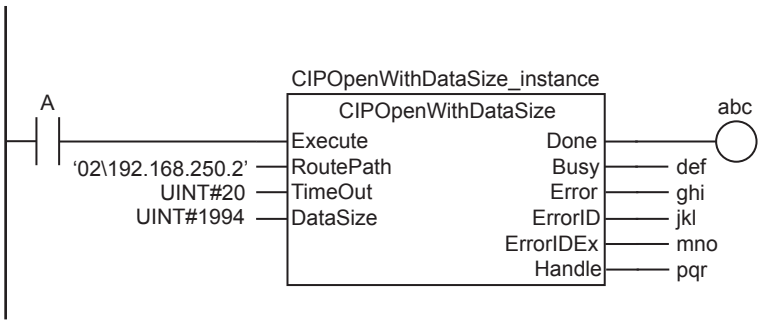
连接建立后，输出句柄 “Handle”。
连接超时时间由 “TimeOut”指定。执行CIPSend指令、CIPWrite指令、CIPRead指令中的任一指令后，在连接超时时间内如对象节点未返回响应，则判断为通信失败。执行CIPRead指令、CIPWrite指令、CIPSend指令，连接超时时间将在对象节点返回响应时复位。

“Handle”的数据类型为结构体_sCIP_HANDLE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Handle	句柄	句柄	_sCIP_HANDLE	–	–	–
Handle	句柄	句柄	UDINT	遵从数据类型	–	–

“RoutePath”=‘02\192.168.250.2’、“TimeOut”=UINT#20时的示例如下所示。对IP地址‘192.168.250.2’的对象节点，建立CIP的Class3连接。数据长度为1994字节，超时时间为2s。句柄代入变量pqr。

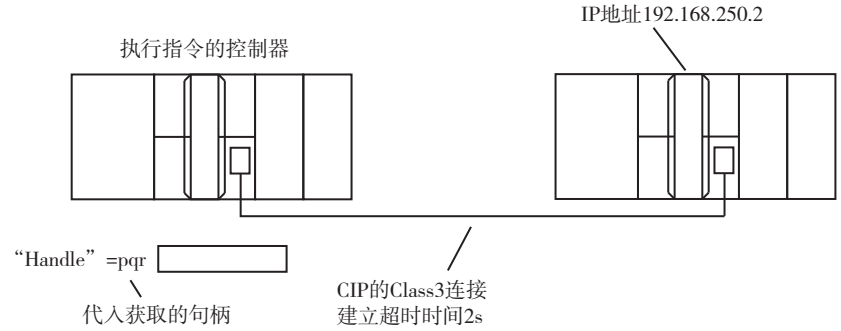
LD



ST

```
CIPOpenWithDataSize_instance(A,
                             '2\192.168.250.2',
                             UINT#20, UINT#1994,
                             abc, def, ghi, jkl, mno,
                             pqr);
```

对“RoutePath”指定的CIP网络上的对象节点，建立CIP的Class3连接。



“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。“ErrorIDEx”的值和含义由对象节点规定。请参阅对象节点的手册。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

- CIP通信的详情请参阅下列手册。
  - 📖“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
  - 📖“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
  - 📖“CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”
- Class3连接的服务为Large_Forward_Open时，📖 也可使用 “CIPOpen指令(P.2-992)”。

使用注意事项

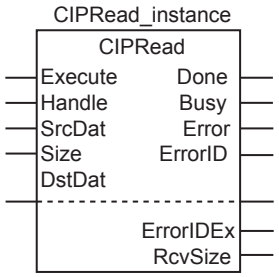
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 📖“本章说明(P.2-3)”。
- 本指令或CIPOpen指令需在CIPRead指令、CIPWrite指令、CIPSend指令前执行。
- 通过本指令建立连接后，首次超时时间在 “TimeOut”的值小于100(10s)时，也为10s。
- 请使用CIPClose指令关闭本指令建立的连接。
- 连接超时时，仍会剩余本指令生成的句柄。因此请务必使用CIPClose指令，关闭连接。
- 本指令生成的句柄在变更程序模式时无效。
- 最多可同时生成32个句柄。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “TimeOut”的值超过有效范围时。
  - “RoutePath”的字符串错误时。
  - 同时执行了超过32个的CIP相关指令时。
  - 试图开通的连接超过CIPClass连接的资源(32个)时。
  - 连接建立未响应时。
  - “DataSize”的值为510～1994，且建立连接的对象节点不支持Large_Forward_Open时。
  - 发生本机IP地址的设定异常时。
  - 发生了IP重复异常时。
  - TCP连接已全部使用时。
  - 在BOOTP服务器发生异常的状态下，执行了本指令时。

版本相关信息

- 本指令可用于CPU单元Ver.1.06以上且Sysmac Studio Ver.1.07以上。
- CPU单元Ver.1.10以上版本时，即使 “Error”变为TRUE，也不会变更 “Handle”的值。Ver.1.09以下版本时，会将 “Handle”的值变更为 “0”。

CIPRead

读取CIP网络上其它站控制器的变量值(Class3 Explicit信息)。

指令	名称	FB/FUN	图形表现	ST表现
CIPRead	变量读取(Class3 Explicit信息)	FB		CIPRead_instance(Execute, Handle, SrcDat, Size, DstDat, Done, Busy, Error, ErrorID, ErrorIDEx, RcvSize);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Handle	句柄	输入	通过CIPOpen或CIPOpenWithDataSize指令获取的句柄	–	–	–
SrcDat	其它站变量名称		待读取的其它站控制器的变量名称	遵从数据类型		"
Size	读取元素数量		待读取的元素数量	0 ~ 8186 *1		1
DstDat	读取数据	输入输出	读取数据的值	遵从数据类型	–	–
RcvSize	读取数据大小	输出	读取数据的大小	0 ~ 8186 *1	字节	–

*1 NX1P2 CPU单元及NJ系列 CPU单元时为 “0 ~ 1988”。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Handle	结构体_sCIP_HANDLE 详情参阅功能说明																			
SrcDat																				○
Size							○													
DstDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、数组、整个结构体、结构体的1个结构要素、联合体的1个结构要素(*)																			
RcvSize							○													

* 无法指定STRING型的数组。

功能

读取 “Handle”指定的CIP网络上其它站控制器内，其它站变量名称 “SrcDat”指定的网络变量值。
读取值保存在读取数据 “DstDat”中。

待读取的元素数由 “Size”指定。“SrcDat”是数组时为待读取的元素数，不是数组时必定指定1。“Size”的值为0时，无论 “SrcDat”是否为数组，均不执行读取。
读取完成后，读取的数据大小以字节为单位，代入读取数据大小 “RcvSize”。
可读取的最大数据会根据执行连接建立的指令及要读取的数据类型，有如下变化。

执行连接建立的指令	要读取的数据类型	可读取的最大数据[字节]
CIPOpen	结构体	1984
	STRING型	1986
	上述以外的数据类型	1988
CIPOpenWithDataSize	结构体	CIPOpenWithDataSize指令的 “DataSize” – 10
	STRING型	CIPOpenWithDataSize指令的 “DataSize” – 8
	上述以外的数据类型	CIPOpenWithDataSize指令的 “DataSize” – 6

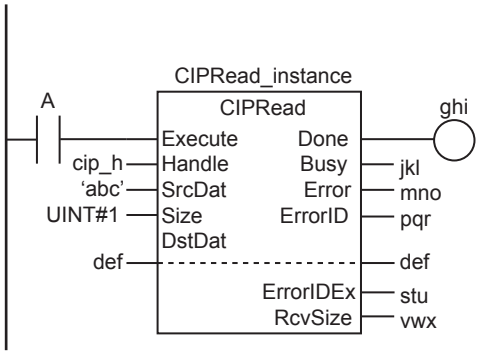
“Handle”的数据类型为结构体_sCIP_HANDLE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Handle	句柄	句柄	_sCIP_HANDLE	–	–	–
Handle	句柄	句柄	UDINT	遵从数据类型	–	–

“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。

读取其它站的变量abc值，并保存至本站的变量def时的示例如下所示。读取元素数 “Size”的值为UINT#1。abc、def的数据类型为SINT。SINT型的数据大小为1字节，因此读取数据大小vwx的值为UINT#1。

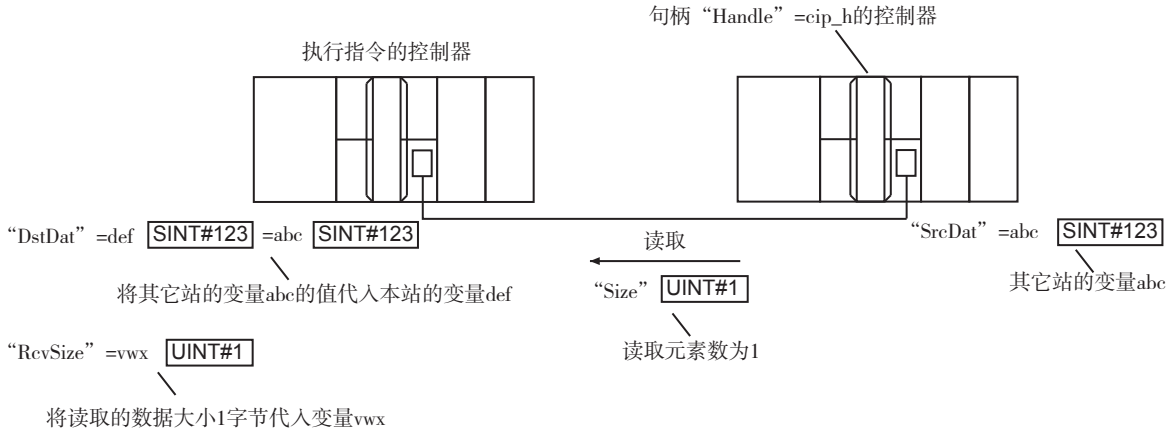
LD



ST

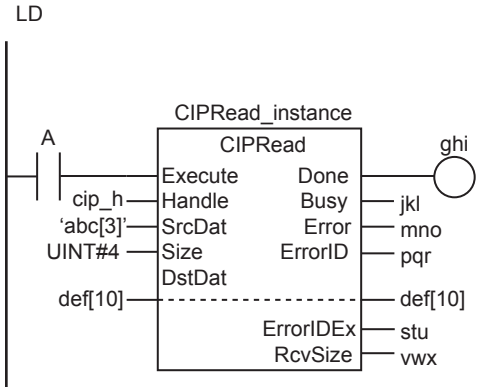
```
CIPRead_instance(A, cip_h, 'abc', UINT#1,
                 def, ghi, jkl, mno, pqr,
                 stu, vwx);
```

将句柄“Handle”指定的CIP网络上其它站的变量“SrcDat”的值代入本站的变量“DstDat”。
读取元素数由“Size”指定。读取的数据大小代入“RcvSize”。



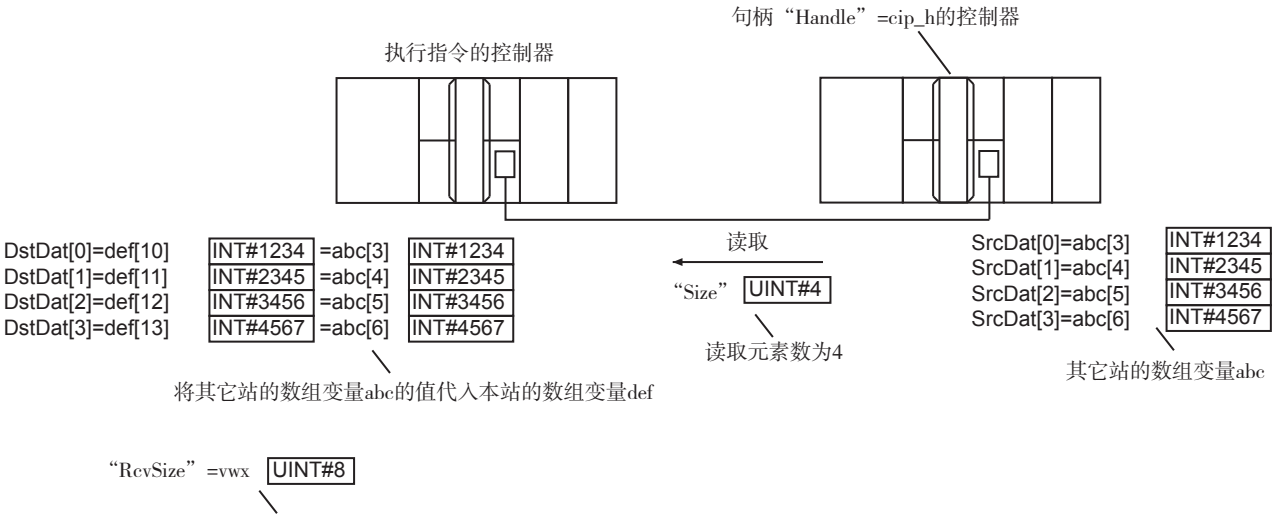
读取数组时

待读取的数据为数组时，必须将带下标的数组元素作为参数传输至“SrcDat”。此外，也必须将带下标的数组元素作为参数传输至“DstDat”。
读取其它站数组变量abc[3]～abc[6]的4个元素，并保存至本站的数组变量def[10]～def[13]时的示例如下所示。abc、def的数据类型为INT。INT型的数据大小为2字节，因此读取数据大小vwx的值为UINT#8。




```
ST

CIPRead_instance(A, cip_h, 'abc[3]', UINT#4,
                 def[10], ghi, jkl, mno, pqr,
                 stu, vwx);
```

将其它站的数组变量abc[3]～abc[6]的值代入本站的数组变量def[10]～def[13]。



“RcvSize”=vwx UINT#8
将读取的数据大小8字节代入变量vwx

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
- 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
- 🕮 “CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮 “本章说明(P.2-3)”。
- 执行本指令前，请务必执行CIPOpen指令或CIPOpenWithDataSize指令获取 “Handle”。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 其它站控制器为欧姆龙制时，可读取的变量仅限于网络中公布的变量。因此，请先在网络中公开指定该变量。
- 无法对CJ单元用存储器的地址进行直接指定并读取数据。读取CJ单元用存储器的特定地址时，请事先通过AT指定将该存储器地址分割为变量。
- 也无法对本站的CJ单元用存储器的地址进行直接指定并保存数据。保存至CJ单元用存储器的特定地址时，请事先通过AT指定将该存储器地址分割为 “DstDat”。
- “SrcDat”可使用的字符的条件如下所述。

项目	规格
最大字节数	127字节
字符代码	UTF-8
可使用的字符	英数字(不区分大小写)、多字节字符、‘_’(下划线)

项目	规格
无法同时使用的字符	· 以ASCII码的数字0 ~ 9(字符代码16#30 ~ 16#39)开头的字符串。 · 仅有ASCII码的 “_”(下划线)的1个字符的字符串。 · 含有2个以上连续的ASCII码的 “_”(下划线)的字符串。 · 首字符为ASCII码的 “_”(下划线)的字符串。 · 尾字符为ASCII码的 “_”(下划线)的字符串。 · 开头2个字符为 “P_”的字符串。

- 以下情况时会发生异常。“Error”变为TRUE。
  - “Size”的值超过有效范围时。
  - “SrcDat”的字符串错误时。
  - 已读取值的数据类型与 “DstDat”的数据类型不一致时。
  - 已读取值的数据大小超过 “DstDat”的范围时。
  - “DstDat”中指定了不支持的数据类型时。
  - 返回了CIP规定的错误响应时。
  - “Handle.Handle”的值超过有效范围时。
  - 同时执行了超过32个的CIP相关指令时。
  - CIPOpen指令或CIPOpenWithDataSize指令生成的连接超时时。
  - “SrcDat”的大小超过了执行连接建立的指令及要读取的数据类型规定的数据大小时。
- 扩展错误代码 “ErrorIDEx”在本指令中具有CIP信息异常代码的含义。相应内容如下所示。

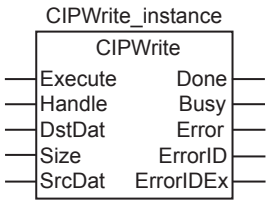
值	异常内容
16#02000000	在通信目标的节点为高负荷的状态下，无法正常通信。
16#04000000	指定的其它站变量为下列数据类型且在其它站中不存在。 · 基本数据类型 · 整个枚举体 · 整个结构体 · 整个联合体 · 整个数组
16#05000000	指定的其它站变量为下列数据类型且在其它站中不存在。 · 枚举体的枚举元素 · 结构体的1个结构要素 · 联合体的1个结构要素 · 数组的元素
16#08000000	不支持所请求的服务。
16#0C008010	指定的其它站变量正在下载中。
16#0C008011	
16#11000000	“Size”的值超过当前可读取的数据大小。
16#1F000102	读取对象为无法读取的变量。
16#1F008007	指定了无法访问的变量。
16#20008017	指定的其它站变量不是数组，但读取元素数不是1。
16#20008018	指定的其它站变量是数组，读取元素数超过该数组的元素数。
16#26000000	指定的其它站变量仅为NULL字符。

示例程序

🕮 请参阅 “CIPOpen指令(P.2–992)”的示例程序。

CIPWrite

将值写入CIP网络上其它站控制器的变量(Class3 Explicit信息)。

指令	名称	FB/FUN	图形表现	ST表现
CIPWrite	变量写入(Class3 Explicit信息)	FB		CIPWrite_instance(Execute, Handle, DstDat, Size, SrcDat, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Handle	句柄	输入	通过CIPOpen指令或CIPOpenWithDataSize指令获取的句柄	-	-	-
DstDat	其它站变量名称		写入的其它站控制器的变量名称	遵从数据类型		''
Size	写入元素数量		待写入的元素数量	0 ~ 8178 *1		1
SrcDat	写入数据		待写入的数据的值	遵从数据类型		*2

*1 NX1P2 CPU单元及NJ系列 CPU单元时为 “0 ~ 1980”。

*2 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Handle	结构体_sCIP_HANDLE 详情参阅功能说明																			
DstDat																				○
Size							○													
SrcDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、数组*1、整个结构体、结构体的1个结构要素、联合体的1个结构要素																			

*1 无法指定STRING型的数组。

功能

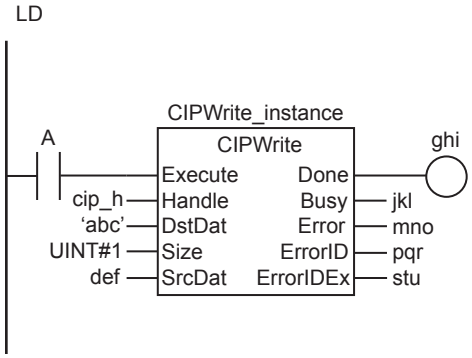
将值写入 “Handle”指定的CIP网络上其它站控制器内其它站变量名称 “DstDat”指定的网络变量。
待写入的值为写入数据 “SrcDat”的内容。
待写入的元素数由 “Size”指定。“DstDat”是数组时为待写入的元素数，不是数组时必定指定1。“Size”的值为0时，无论 “DstDat”是否为数组，均不执行写入。

“Handle”的数据类型为结构体_sCIP_HANDLE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Handle	句柄	句柄	_sCIP_HANDLE	–	–	–
Handle	句柄	句柄	UDINT	遵从数据类型	–	–

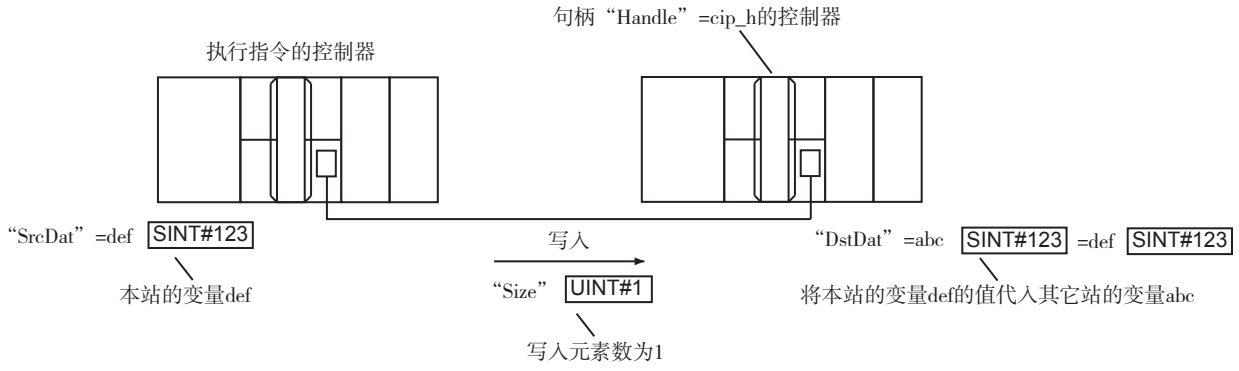
“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。

将本站的变量def的值写入其它站的变量abc时的示例如下所示。写入元素数 “Size”的值为UINT#1。




```
ST
CIPWrite_instance(A, cip_h, 'abc', UINT#1,
                  def, ghi, jkl, mno, pqr, stu);
```

将本站的变量“SrcDat”的值代入句柄“Handle”指定的CIP网络上其它站的变量“DstDat”。写入元素数由“Size”指定。

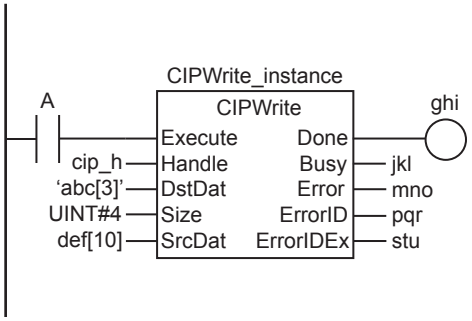


写入数组时

待写入的数据为数组时，必须将带下标的数组元素作为参数传输至 “DstDat”。此外，也必须将带下标的数组元素作为参数传输至 “SrcDat”。

将本站的数组变量def[10] ~ def[13]的值写入其它站数组变量abc[3] ~ abc[6]的4个元素时的示例如下所示。

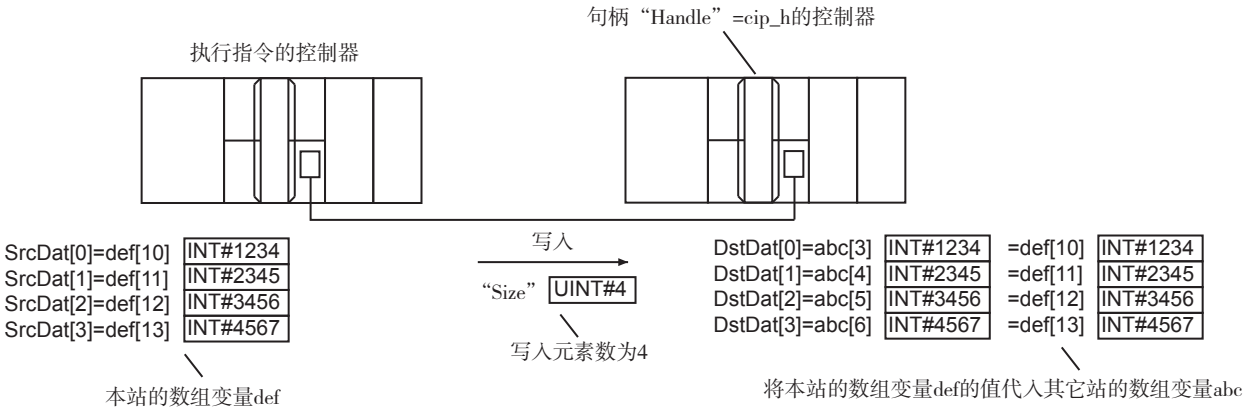
LD



ST

```
CIPWrite_instance(A, cip_h, 'abc[3]', UINT#4, def,
                  ghi[10], jkl, mno, pqr, stu);
```

将本站的数组变量def[10] ~ def[13]的值代入其它站的数组变量abc[3] ~ abc[6]。



可写入的数据大小

如下所述，可写入数据的最大容量因 “DstDat”指定的变量数据类型及变量名称而异。

可写入的数据大小[字节] = 基本大小 – “DstDat”的变量名称大小

上式的项目	含义
基本大小	使用CIPOpen指令建立连接时 · “DstDat”指定变量的数据类型为结构体时，为1984字节。 · “DstDat”指定变量的数据类型为STRING型时，为1986字节。 · 为上述以外的数据类型时，为1988字节。
	使用CIPOpenWithDataSize指令建立连接时 · “DstDat”指定变量的数据类型为结构体时，可按下式计算。 基本大小[字节]= CIPOpenWithDataSize指令的 “DataSize” – 10 · “DstDat”指定变量的数据类型为STRING型时，可按下式计算。 基本大小[字节]= CIPOpenWithDataSize指令的 “DataSize” – 8 · 为上述以外的数据类型时，可按下式计算。 基本大小[字节]= CIPOpenWithDataSize指令的 “DataSize” – 6
“DstDat”的变量名称大小	· 根据变量名称的结构体的各层ASCII码字节数的合计值 + (层数*2)计算。 · 各层字符串的ASCII码字节数为奇数时，在相应值上加1。 · 某层为数组时，再加上(数组的维数*4)。 · 表示结构体层次间隔的 ‘.’、数组的 ‘[‘、’]’、 ‘,’不包含在变量名称大小中。 (例1) “DstDat”的变量名称为 ‘aaa.bbbbb[1,2,3].cc’时 ·第1层aaa的字符串为3字节。为奇数，因此加上1为4字节。 ·第2层bbbbb[1,2,3]的bbbbb字符串为5字节。为奇数，因此加上1为6字节。 ·并且，第2层bbbbb[1,2,3]为3维数组，因此加上3*4=12即18字节。 ·第3层cc的字符串为2字节。为偶数，因此即2字节。 ·第1层4字节、第2层18字节、第3层2字节，加上层数3*2=6， 变量名称大小为30字节。 (例2) “DstDat”的变量名称为 ‘val’时 ·第1层val的字符串为3字节。为奇数，因此加上1为4字节。 ·再加上层数1*2=2，变量名称大小为6字节。 (例3) “DstDat”的变量名称为 ‘array[8]’时 ·第1层array的字符串为5字节。为奇数，因此加上1为6字节。 ·数组的维数为1。因此，加上1*4=4。 ·再加上层数1*2=2，变量名称大小为12字节。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
- 🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
- 🕮“CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮“本章说明(P.2-3)”。
- 执行本指令前，请务必执行CIPOpen指令或CIPOpenWithDataSize指令获取 “Handle”。
- 请务必将传输至 “SrcDat”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “SrcDat”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 其它站控制器为欧姆龙制时，可写入的变量仅限于网络中公布的变量。因此，请先在网络中公开指定该变量。
- 无法对CJ单元用存储器的地址进行直接指定并写入数据。写入CJ单元用存储器的特定地址时，请事先通过AT指定将该存储器地址分割为变量。
- 也无法直接指定本站的CJ单元用存储器的地址。写入CJ单元用存储器的特定地址值时，请事先通过AT指定将该存储器地址分割为 “SrcDat”。
- “DstDat”可使用的字符的条件如下所述。

项目	规格
最大字节数	127字节
字符代码	UTF-8
可使用的字符	英数字(不区分大小写)、多字节字符、‘_’(下划线)
无法同时使用的字符	· 以ASCII码的数字0～9(字符代码16#30～16#39)开头的字符串。 · 仅有ASCII码的 “_”(下划线)的1个字符的字符串。 · 含有2个以上连续的ASCII码的 “_”(下划线)的字符串。 · 首字符为ASCII码的 “_”(下划线)的字符串。 · 尾字符为ASCII码的 “_”(下划线)的字符串。 · 开头2个字符为 “P_”的字符串。

- 以下情况时会发生异常。“Error”变为TRUE。
  - “Size”的值超过有效范围时。
  - “DstDat”的字符串错误时。
  - “Size”的值超过 “SrcDat”的范围时。
  - “SrcDat”中指定了不支持的数据类型时。
  - 返回了CIP规定的错误响应时。
  - “Handle.Handle”的值超过有效范围时。
  - 同时执行了超过32个的CIP相关指令时。
  - CIPOpen指令或CIPOpenWithDataSize指令生成的连接超时时。
  - “DstDat”的大小与“SrcDat”值的总和超过了执行连接建立的指令及要写入的数据类型规定的数据大小时。
- 扩展错误代码 “ErrorIDEx”在本指令中具有CIP信息异常代码的含义。相应内容如下所示。

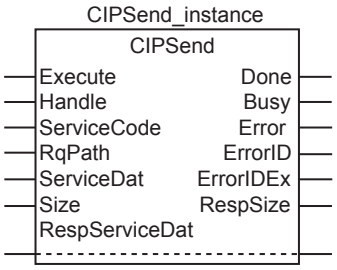
值	异常内容
16#02000000	在通信目标的节点为高负荷的状态下，无法正常通信。
16#04000000	指定的其它站变量为下列数据类型且在其它站中不存在。 · 基本数据类型 · 整个枚举体 · 整个结构体 · 整个联合体 · 整个数组
16#05000000	指定的其它站变量为下列数据类型且在其它站中不存在。 · 枚举体的枚举元素 · 结构体的1个结构要素 · 联合体的1个结构要素 · 数组的元素
16#08000000	不支持所请求的服务。
16#0C008010	指定的其它站变量正在下载中。
16#0C008011	
16#1F000102	· 指定的其它站变量中设定了常数属性，因此无法写入。 · 写入数据与写入元素数不一致。
16#1F008007	指定了无法访问的变量。
16#20008017	指定的其它站变量不是数组，但写入元素数不是1。
16#20008018	指定的其它站变量是数组，写入元素数超过该数组的元素数。
16#20008022	服务器端变量的数据类型与客户端变量的数据类型不一致。
16#20008028	· 指定的其它站变量为枚举体，写入数据为枚举元素中不存在的值。 · 指定的其它站变量具有范围型指定属性，写入数据为超过范围的值。
16#26000000	指定的其它站变量名称仅为NULL字符。

示例程序

请参阅 “CIPOpen指令(P.2–992)”的示例程序。

CIPSend

将Class3的CIP信息发送至CIP网络的指定设备。

指令	名称	FB/FUN	图形表现	ST表现
CIPSend	任意Explicit信息发送(Class3)	FB		CIPSend_instance(Execute, Handle, ServiceCode, RqPath, ServiceDat, Size, RespServiceDat, Done, Busy, Error, ErrorID, ErrorIDEx, RespSize);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Handle	句柄	输入	通过CIPOpen指令或CIPOpenWithDataSize指令获取的句柄	–	–	–
Service Code	服务代码		服务代码	遵从数据类型		
RqPath	请求路径		请求路径	–		
ServiceDat	服务数据		待发送的服务数据	遵从数据类型		(*)
Size	发送元素数		待发送的元素数			1
Resp ServiceDat	响应数据	输入输出	响应数据	遵从数据类型	–	–
RespSize	响应数据大小	输出	响应数据的大小	遵从数据类型	字节	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Handle	结构体_sCIP_HANDLE 详情参阅功能说明																			
Service Code		○																		
RqPath	结构体_sREQUEST_PATH或_sREQUEST_PATH_EX*1 详情参阅功能说明																			
ServiceDat		○	○	○	○	○	○	○	○	○	○	○	○	○	○					
	也可指定数组、结构体的1个结构要素、联合体的1个结构要素																			
Size							○													
Resp ServiceDat		○	○	○	○	○	○	○	○	○	○	○	○	○	○					
	也可指定数组、结构体的1个结构要素、联合体的1个结构要素																			
RespSize							○													

*1 _sREQUEST_PATH_EX型在CPU单元 Ver.1.11以上且Sysmac Studio Ver.1.15以上时可指定。

功能

对应服务代码 “ServiceCode”指定的服务，将服务数据 “ServiceDat”作为Class3 Explicit信息发送。
发送目标由句柄 “Handle”指定。
请求路径由 “RqPath”指定。

待发送的元素数由 “Size”指定。“ServiceDat”为数组时为待发送的元素数，非数组时必定指定1。无需服务数据时，将 “Size”的值设为0。

发送后，响应数据保存至 “RespServiceDat”。响应数据的大小以字节为单位保存至 “RespSize”。

“Handle”的数据类型为结构体_sCIP_HANDLE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Handle	句柄	句柄	_sCIP_HANDLE	–	–	–
Handle	句柄	句柄	UDINT	遵从数据类型	–	–

“RqPath”的数据类型为结构体_sREQUEST_PATH或_sREQUEST_PATH_EX。
通常请使用_sREQUEST_PATH。需设定任意逻辑格式的大小进行使用时，请使用_sREQUEST_PATH_EX。
规格如下所示。

- _sREQUEST_PATH型

变量	名称	内容	数据类型	有效范围	单位	初始值
RqPath	请求路径	请求路径	_sREQUEST_PATH	–	–	–
ClassID	等级ID	等级ID	UINT	遵从数据类型	–	0
InstanceID	实例ID	实例ID	UINT			0
isAttributeID	有无属性	TRUE：使用属性ID FALSE：不使用属性ID	BOOL			FALSE
AttributeID	属性ID	属性ID	UINT			0

(注) _sREQUEST_PATH型时，各ID的逻辑格式大小为16位。

- _sREQUEST_PATH_EX型

变量	名称	内容	数据类型	有效范围	单位	初始值
RqPath	请求路径	请求路径	_sREQUEST_PATH_EX	–	–	–
ClassIDLogicalFormat	等级ID逻辑格式	等级ID的数据大小	_eCIP_LOGICAL_FORMAT	遵从数据类型	–	_8BIT
ClassID	等级ID	等级ID	UDINT			0
InstanceIDLogicalFormat	实例ID逻辑格式	实例ID的数据大小	_eCIP_LOGICAL_FORMAT			_8BIT
InstanceID	实例ID	实例ID	UDINT			0
isAttributeID	有无属性	TRUE：使用属性ID FALSE：不使用属性ID	BOOL			FALSE
AttributeIDLogicalFormat	属性ID逻辑格式	属性ID的数据大小	_eCIP_LOGICAL_FORMAT			_8BIT
AttributeID	属性ID	属性ID	UDINT			0

“ClassIDLogicalFormat”、“InstanceIDLogicalFormat”、“AttributeIDLogicalFormat”的数据类型为枚举体_eCIP_LOGICAL_FORMAT。
枚举体_eCIP_LOGICAL_FORMAT的枚举元素的含义如下所述。

枚举元素	含义
_8BIT	8位
_16BIT	16位
_32BIT	32位

“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。“ErrorIDEx”的值和含义由对象节点规定。请参阅对象节点的手册。

收发数组时

“ServiceDat”或 “RespServiceDat”为数组时，必须将带下标的数组元素作为参数传输至相应变量。

读写的数据大小

可读取的最大数据根据使用CIPOpen指令还是CIPOpenWithDataSize指令建立连接，有如下变化。

执行连接建立的指令	可读取的最大数据
CIPOpen	1990字节
CIPOpenWithDataSize	最多可接收服务器返回的8188 *1字节的响应

*1 NX1P2 CPU单元及NJ系列 CPU单元时上限为 “1990”。

如下所示，可写入的最大数据因请求路径有无属性和执行连接建立的指令而异。

可写入的数据大小[字节] = 基本大小 – 有无属性

上式的项目	含义
基本大小	· 使用CIPOpen指令建立连接时为1992字节 · 使用CIPOpenWithDataSize指令建立连接时为CIPOpenWithDataSize指令的 “DataSize” –2
有无属性*1	· 使用属性ID时为14字节 · 不使用属性ID时为10字节

*1 CPU单元Ver.1.10以下或Sysmac Studio Ver.1.14以下时，如下所述。
使用属性ID时为12字节
不使用属性ID时为8字节

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”
- 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”
- 🕮 “CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD–355)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 执行本指令前，请务必执行CIPOpen指令或CIPOpenWithDataSize指令获取 “Handle”。
- 请务必将传输至 “ServiceDat”的输入参数设为变量。如果传输常数，编连时会发生异常。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 其它站控制器为欧姆龙制时，可写入的变量仅限于网络中公布的变量。因此，请先在网络中公开指定该变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “RqPath.ClassIDLogicalFormat”、“RqPath.AttributeIDLogicalFormat”的值超过有效范围时。
  - “RqPath.ClassIDLogicalFormat”指定的大小与 “RqPath.ClassID”的数据大小、“RqPath.InstanceIDLogicalFormat”指定的大小与 “RqPath.InstanceID”的数据大小或者“RqPath.AttributeIDLogicalFormat”指定的大小与 “RqPath.AttributeID”的数据大小不一致时。
  - “Size”的值超过写入数据范围时。
  - “Size”的值超过 “ServiceDat”的范围时。
  - “RespSize”的值超过 “RespServiceDat”的范围时。
  - “ServiceDat”中指定了不支持的数据类型时。
  - “RespServiceDat”中指定了不支持的数据类型时。
  - “RqPath”中指定了_sREQUEST_PATH或_sREQUEST_PATH_EX以外数据类型的变量时。
  - 返回了CIP规定的错误响应时。
  - “Handle.Handle”的值超过有效范围时。
  - 同时执行了超过32个的CIP相关指令时。
  - CIPOpen指令或CIPOpenWithDataSize指令生成的连接超时时。
  - “RqPath”与 “ServiceDat”的大小总和超过了执行连接建立的指令规定的数据大小时。

示例程序

请参阅 “CIPOpen指令(P.2–992)”的示例程序。

CIPClose

切断指定句柄的CIP Class3的连接。

指令	名称	FB/FUN	图形表现	ST表现
CIPClose	CIP Class3连接的切断	FB	CIPClose_instance CIPClose Execute Done Handle Busy Error ErrorID ErrorIDEx	CIPClose_instance(Execute, Handle, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Handle	句柄	输入	通过CIPOpen指令或CIPOpenWithDataSize指令获取的句柄	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Handle	结构体_sCIP_HANDLE　详情参阅功能说明																			

功能

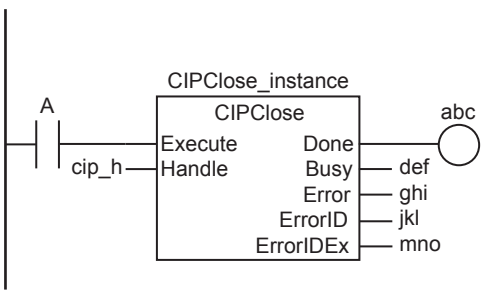
切断句柄 “Handle”指定的CIP Class3连接。

“Handle”的数据类型为结构体_sCIP_HANDLE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Handle	句柄	句柄	_sCIP_HANDLE	–	–	–
Handle	句柄	句柄	UDINT	遵从数据类型	–	–

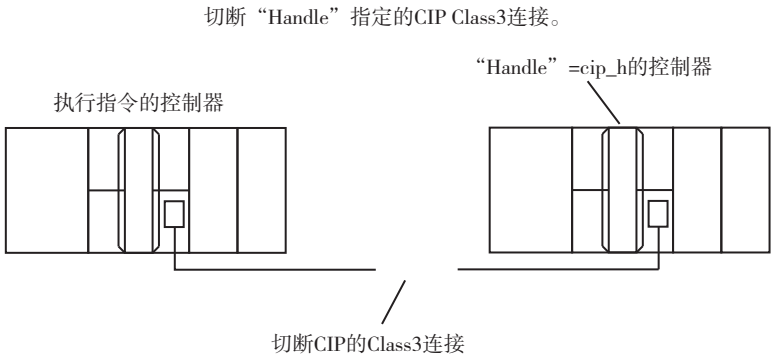
示例如下所示。切断 “Handle” =cip_h指定的CIP Class3连接。

LD



ST

```
CIPClose_instance(A, cip_h, abc, def,
                  ghi, jkl, mno);
```



相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
- 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
- 🕮 “CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”

使用注意事项

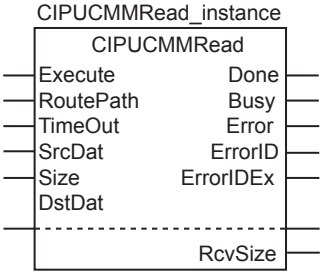
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮 “本章说明(P.2-3)”。
- “Handle”请指定通过CIPOpen指令或CIPOpenWithDataSize指令获取的内容。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 本指令不使用 “ErrorIDEx”。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Handle.Handle”的值超过有效范围时。
  - 同时执行了超过32个的CIP相关指令时。

示例程序

🕮 请参阅 “CIPOpen指令(P.2-992)”的示例程序。

CIPUCMMRead

读取指定CIP网络上其它站控制器的变量值(UCMM Explicit信息)。

指令	名称	FB/FUN	图形表现	ST表现
CIPUCMMRead	变量读取(UCMM Explicit信息)	FB		CIPUCMMRead_instance(Execute, RoutePath, TimeOut, SrcDat, Size, DstDat Done, Busy, Error, ErrorID, ErrorIDEx, RcvSize);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
RoutePath	路径	输入	路径	遵从数据类型	–	–
TimeOut	超时时间		超时时间	1 ~ 65535	0.1s	20 (2s)
SrcDat	其它站变量名称		待读取的其它站控制器的变量名称	遵从数据类型	–	''
Size	读取元素数量		待读取的元素数量	0 ~ 496		1
DstDat	读取数据	输入输出	读取数据的值	遵从数据类型	–	–
RcvSize	读取数据大小	输出	读取数据的大小	0 ~ 496	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
RoutePath																				○
TimeOut							○													
SrcDat																				○
Size							○													
DstDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、数组、整个结构体、结构体的1个结构要素、联合体的1个结构要素(*)																			
RcvSize							○													

* 无法指定STRING型的数组。

功能

读取路径 “RoutePath”指定的CIP网络上其它站控制器内其它站变量名称 “SrcDat”指定的网络变量值。读取值保存在读取数据 “DstDat”中。

待读取的元素数由 “Size”指定。“SrcDat”是数组时为待读取的元素数，不是数组时必定指定1。“Size”的值为0时，无论 “SrcDat”是否为数组，均不执行读取。

读取完成后，读取的数据大小以字节为单位，代入读取数据大小 “RcvSize”。可读取的最大数据会根据变量的数据类型，有如下变化。

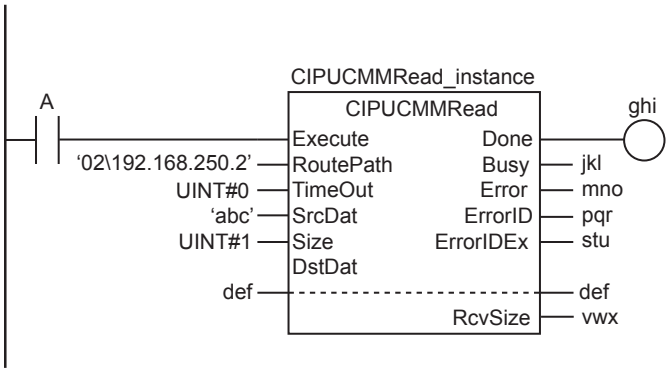
- 结构体时为492字节。
- STRING型时为494字节。
- 为上述以外的数据类型时，为496字节。

超时时间由 “TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。

“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。

读取其它站的变量abc值，并保存至本站的变量def时的示例如下所示。读取元素数“Size”的值为UINT#1。abc、def的数据类型为SINT。SINT型的数据大小为1字节，因此读取数据大小vwx的值为UINT#1。

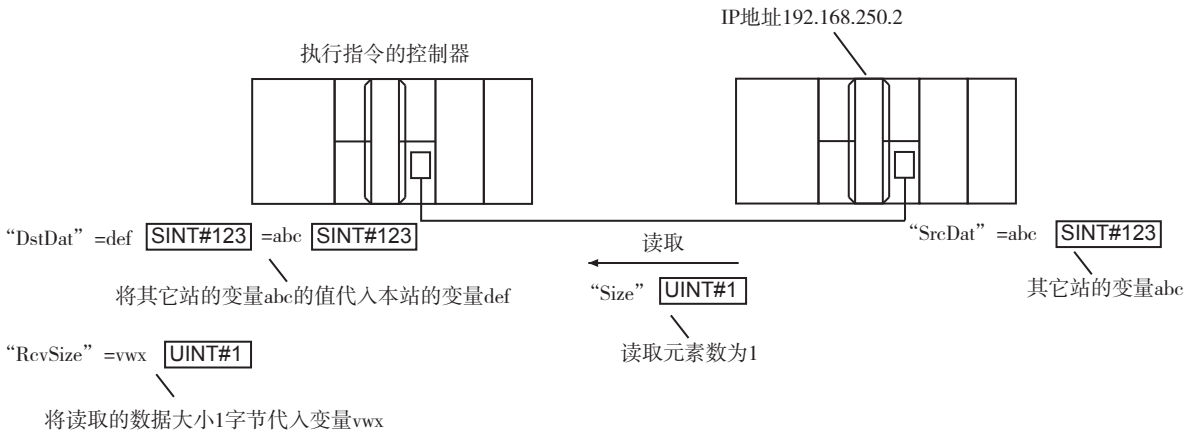
LD



ST

```
CIPUCMMRead_instance(A, '2\192.168.250.2',
                     UINT#0, 'abc', UINT#1,
                     def, ghi, jkl, mno, pqr,
                     stu, vwx);
```

将路径“RoutePath”指定的CIP网络上其它站的变量“SrcDat”的值代入本站的变量“DstDat”。
读取元素数由“Size”指定。读取的数据大小代入“RcvSize”。

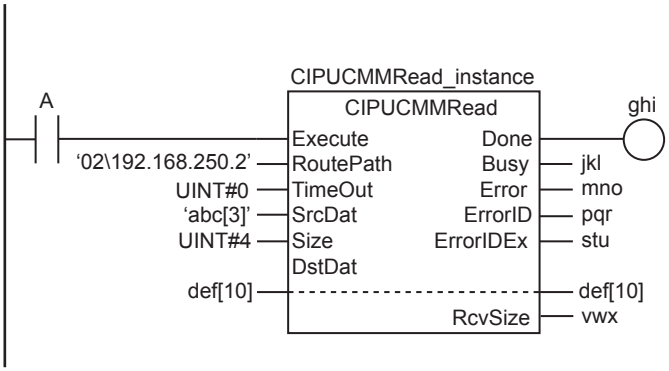


读取数组时

待读取的数据为数组时，必须将带下标的数组元素作为参数传输至 “SrcDat”。此外，也必须将带下标的数组元素作为参数传输至 “DstDat”。

读取其它站数组变量abc[3] ~ abc[6]的4个元素，并保存至本站的数组变量def [10] ~ def [13]时的示例如下所示。abc、def的数据类型为INT。INT型的数据大小为2字节，因此读取数据大小vwx的值为UINT#8。

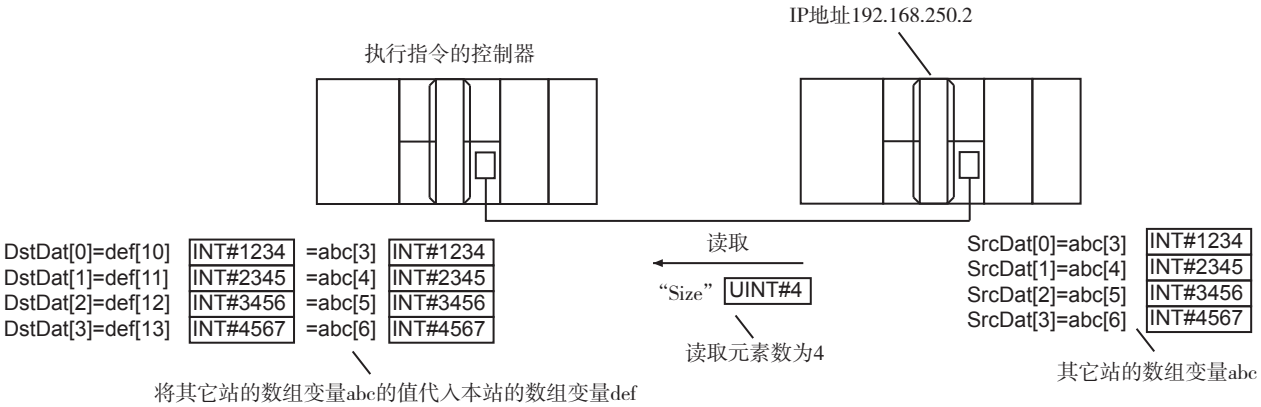
LD

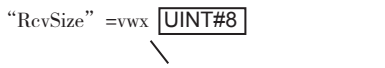


ST

```
CIPUCMMRead_instance(A, '2\192.168.250.2',
                     UINT#0, 'abc[3]', UINT#4,
                     def[10], ghi, jkl, mno, pqr,
                     stu, vwx);
```

将其它站的数组变量abc[3] ~ abc[6]的值代入本站的数组变量def[10] ~ def[13]。





相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
- 🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
- 🕮“CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮“本章说明(P.2-3)”。
- 本指令仅适用于NJ/NX系列 CPU单元及经由NY系列控制器的内置EtherNet/IP端口。
- 其它站控制器为欧姆龙制时，可读取的变量仅限于网络中公布的变量。因此，请先在网络中公开指定该变量。
- 无法对CJ单元用存储器的地址进行直接指定并读取数据。读取CJ单元用存储器的特定地址时，请事先通过AT指定将该存储器地址分割为变量。
- 也无法对本站的CJ单元用存储器的地址进行直接指定并保存数据。保存至CJ单元用存储器的特定地址时，请事先通过AT指定将该存储器地址分割为 “DstDat”。
- “SrcDat”可使用的字符的条件如下所述。

项目	规格
最大字节数	127字节
字符代码	UTF-8
可使用的字符	英数字(不区分大小写)、多字节字符、‘_’(下划线)
无法同时使用的字符	· 以ASCII码的数字0 ~ 9(字符代码16#30 ~ 16#39)开头的字符串。 · 仅有ASCII码的 “_”(下划线)的1个字符的字符串。 · 含有2个以上连续的ASCII码的 “_”(下划线)的字符串。 · 首字符为ASCII码的 “_”(下划线)的字符串。 · 尾字符为ASCII码的 “_”(下划线)的字符串。 · 开头2个字符为 “P_”的字符串。

- 以下情况时会发生异常。“Error”变为TRUE。
  - · “TimeOut”的值超过有效范围时。
  - · “Size”的值超过有效范围时。
  - · “SrcDat”的字符串错误时。
  - · 已读取值的数据类型与 “DstDat”的数据类型不一致时。
  - · 已读取值的数据大小超过 “DstDat”的范围时。
  - · “DstDat”中指定了不支持的数据类型时。
  - · 返回了CIP规定的错误响应时。
  - · “RoutePath”的字符串错误时。
  - · 同时执行了超过32个的CIP相关指令时。
  - · 超过超时时间仍无响应时。
  - · 发生本机IP地址的设定异常时。
  - · 在BOOTP服务器发生异常的状态下，执行了本指令时。

・ 发生了IP重复异常时。

• 扩展错误代码 “ErrorIDEx”在本指令中具有CIP信息异常代码的含义。相应内容如下所示。

值	异常内容
16#02000000	在通信目标的节点为高负荷的状态下，无法正常通信。
16#04000000	指定的其它站变量为下列数据类型且在其它站中不存在。 ・ 基本数据类型 ・ 整个枚举体 ・ 整个结构体 ・ 整个联合体 ・ 整个数组
16#05000000	指定的其它站变量为下列数据类型且在其它站中不存在。 ・ 枚举体的枚举元素 ・ 结构体的1个结构要素 ・ 联合体的1个结构要素 ・ 数组的元素
16#08000000	不支持所请求的服务。
16#0C008010	指定的其它站变量正在下载中。
16#0C008011	
16#11000000	“Size”的值超过当前可读取的数据大小。
16#1F000102	读取对象为无法读取的变量I/O。
16#1F008007	指定了无法访问的变量。
16#20008017	指定的其它站变量不是数组，但读取元素数不是1。
16#20008018	指定的其它站变量是数组，读取元素数超过该数组的元素数。
16#26000000	指定的其它站变量仅为NULL字符。

示例程序

📖 请参阅 “CIPUCMMSend指令(P.2–1034)”的示例程序。

CIPUCMMWrite

将值写入CIP网络上其它站控制器的变量(UCMM Explicit信息)。

指令	名称	FB/FUN	图形表现	ST表现
CIPUCMMWrite	变量写入(UCMM Explicit信息)	FB	CIPUCMMWrite_instance CIPUCMMWrite Execute / Done RoutePath / Busy TimeOut / Error DstDat / ErrorID Size / ErrorIDEx SrcDat	CIPUCMMWrite_instance(Execute, RoutePath, TimeOut, DstDat, Size, SrcDat, Done, Busy, Error, ErrorID, ErrorIDEx);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
RoutePath	路径	输入	路径	遵从数据类型	–	–
TimeOut	超时时间		超时时间	1 ~ 65535	0.1s	20 (2s)
DstDat	其它站变量名称		写入的其它站控制器的变量名称	遵从数据类型	–	''
Size	写入元素数量		待写入的元素数量	0 ~ 488		1
SrcDat	写入数据		待写入的数据的值	遵从数据类型		*1

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
RoutePath																				○
TimeOut							○													
DstDat																				○
Size							○													
SrcDat	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、数组*1、整个结构体、结构体的1个结构要素、联合体的1个结构要素																			

*1 无法指定STRING型的数组。

功能

将值写入路径“RoutePath”指定的CIP网络上其它站控制器内其它站变量名称“DstDat”指定的网络变量。待写入的值为写入数据 “SrcDat”的内容。

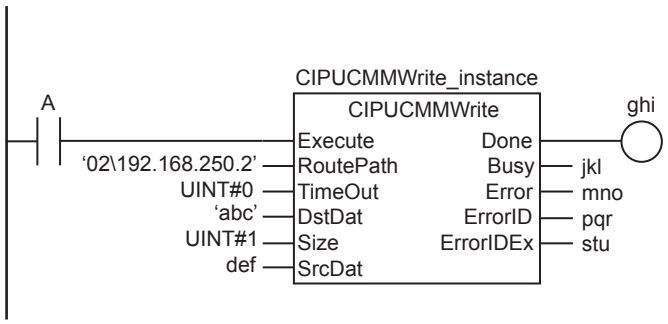
待写入的元素数由 “Size”指定。“DstDat”是数组时为待写入的元素数，不是数组时必定指定1。“Size”的值为0时，无论 “DstDat”是否为数组，均不执行写入。

超时时间由 “TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。

“ErrorID”的值为WORD#16#1C00时，在 “ErrorIDEx”中保存CIP信息异常代码。

将本站的变量def的值写入其它站的变量abc时的示例如下所示。写入元素数 “Size”的值为UINT#1。

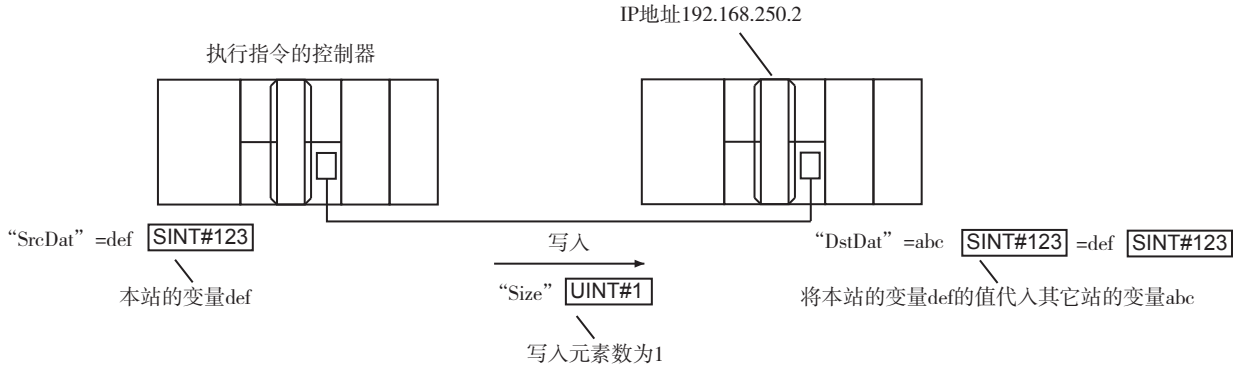
LD



ST

```
CIPUCMMWrite_instance(A, '2\192.168.250.2',
                      UINT#0, 'abc', UINT#1,
                      def, ghi, jkl, mno, pqr,
                      stu);
```

将本站的变量“SrcDat”的值代入路径“RoutePath”指定的CIP网络上其它站的变量“DstDat”。写入元素数由“Size”指定。

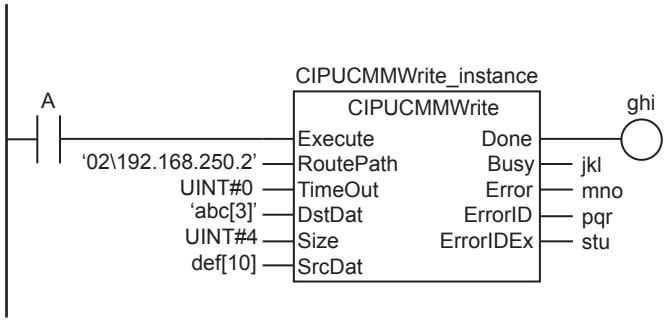


写入数组时

待写入的数据为数组时，必须将带下标的数组元素作为参数传输至 “DstDat”。此外，也必须将带下标的数组元素作为参数传输至 “SrcDat”。

将本站的数组变量def[10] ~ def[13]的值写入其它站数组变量abc[3] ~ abc[6]的4个元素时的示例如下所示。

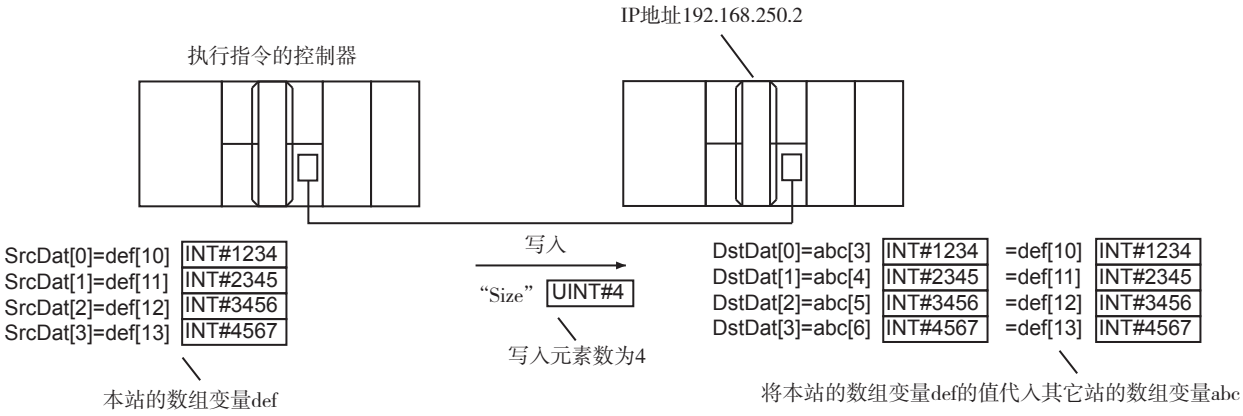
LD



ST

```
CIPUCMMWrite_instance(A, '2\192.168.250.2',
                      UINT#0, 'abc[3]', UINT#4,
                      def[10], ghi, jkl, mno, pqr,
                      stu);
```

将本站的数组变量def[10] ~ def[13]的值代入其它站的数组变量abc[3] ~ abc[6]。



可写入的数据大小

如下所述，可写入的最大数据因 “DstDat”指定的变量数据类型、变量名称及路径而异。

可写入的数据大小[字节] = 基本大小 – “DstDat”的变量名称大小 – 路径信息大小

上式的项目	含义
基本大小	· “DstDat”指定变量的数据类型为结构体时，为492字节。 · “DstDat”指定变量的数据类型为STRING型时，为494字节。 · 为上述以外的数据类型时，为496字节。
“DstDat”的变量名称大小	· 根据变量名称的结构体的各层ASCII码字节数的合计值 + (层数*2)计算。 · 各层字符串的ASCII码字节数为奇数时，在相应值上加1。 · 某层为数组时，再加上(数组的维数*4)。 · 表示结构体层次间隔的 ‘.’、数组的 ‘[‘、’]’、‘,’不包含在变量名称大小中。 (例1) “DstDat”的变量名称为 ‘aaa.bbbbb[1,2,3].cc’时 ・第1层aaa的字符串为3字节。为奇数，因此加上1为4字节。 ・第2层bbbbb[1,2,3]的bbbbb字符串为5字节。为奇数，因此加上1为6字节。 ・并且，第2层bbbbb[1,2,3]为3维数组，因此加上3*4=12即18字节。 ・第3层cc的字符串为2字节。为偶数，因此即2字节。 ・第1层4字节、第2层18字节、第3层2字节，加上层数3*2=6， 变量名称大小为30字节。 (例2) “DstDat”的变量名称为 ‘val’时 ・第1层val的字符串为3字节。为奇数，因此加上1为4字节。 ・再加上层数1*2=2，变量名称大小为6字节。 (例3) “DstDat”的变量名称为 ‘array[8]’时 ・第1层array的字符串为5字节。为奇数，因此加上1为6字节。 ・数组的维数为1。因此，加上1*4=4。 ・再加上层数1*2=2，变量名称大小为12字节。

上式的项目	含义
路径信息大小	· 无跳跃时，路径信息大小为0字节。(*) · 有跳跃时，路径信息大小为路径大小+12字节。 · 路径大小是指路径的字符串的ASCII码的字节数。 · 请注意以下事项。 ・地址部分的开头带 ‘#’时，网络部分和地址部分共计2字节。 ・地址部分的开头无 ‘#’时，网络部分以2字节计算。 ・地址部分的开头无 ‘#’且地址部分的ASCII码的字节数为奇数时，将该值加上1字节。 ・表示路径层次间隔的 ‘\’不包含在路径大小中。 ・开头的跳跃不包含在路径大小中。 (例1) 路径为 ‘01\#11\02\192.168.250.2\01\#01’时 ・开头的跳跃不包含在路径中，因此忽略开头的 ‘01\#11’。 ・之后的网络类型 ‘02’为2字节。 ・之后的地址部分 ‘192.168.250.2’为13字节。为奇数，因此加上1为14字节。 ・之后的 ‘01#01’的地址部分带#，因此网络部分和地址部分共计2字节。 ・合计上述值，路径大小为18字节。 ・路径大小加上12字节，路径信息大小为30字节。 (例2) 路径为 ‘02\192.168.250.2\01\#00’时 ・开头的跳跃不包含在路径中，因此忽略开头的 ‘02\192.168.250.2’。 ・之后的 ‘01#00’的地址部分带#，因此网络部分和地址部分共计2字节。 ・路径大小为2字节。 ・路径大小加上12字节，路径信息大小为14字节。 (例3) 路径为 ‘02\192.168.250.2’时 ・无跳跃，因此路径信息大小为0字节。

*跳跃是指在发送节点和接收节点之间进行路径选择。例如，路径为02\192.168.250.2\01\#00时，一旦路径选择IP地址192.168.250.2的节点，则向00单元发送信息，因此存在1个跳跃。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
- 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
- 🕮 “CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 请务必将传输至 “SrcDat”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “SrcDat”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 其它站控制器为欧姆龙制时，可写入的变量仅限于网络中公布的变量。因此，请先在网络中公开指定该变量。
- 无法对CJ单元用存储器的地址进行直接指定并写入数据。写入CJ单元用存储器的特定地址时，请事先通过AT指定将该存储器地址分割为变量。
- 也无法直接指定本站的CJ单元用存储器的地址。写入CJ单元用存储器的特定地址值时，请事先通过AT指定将该存储器地址分割为 “SrcDat”。
- “DstDat”可使用的字符的条件如下所述。

项目	规格
最大字节数	127字节
字符代码	UTF-8
可使用的字符	英数字(不区分大小写)、多字节字符、‘_’(下划线)
无法同时使用的字符	· 以ASCII码的数字0～9(字符代码16#30～16#39)开头的字符串。 · 仅有ASCII码的 “_”(下划线)的1个字符的字符串。 · 含有2个以上连续的ASCII码的 “_”(下划线)的字符串。 · 首字符为ASCII码的 “_”(下划线)的字符串。 · 尾字符为ASCII码的 “_”(下划线)的字符串。 · 开头2个字符为 “P_”的字符串。

- 以下情况时会发生异常。“Error”变为TRUE。
  - · “TimeOut”的值超过有效范围时。
  - · “Size”的值超过有效范围时。
  - · “DstDat”的字符串错误时。
  - · “Size”的值超过 “SrcDat”的范围时。
  - · “SrcDat”中指定了不支持的数据类型时。
  - · 返回了CIP规定的错误响应时。
  - · “RoutePath”的字符串错误时。
  - · 同时执行了超过32个的CIP相关指令时。
  - · 超过超时时间仍无响应时。
  - · 发生本机IP地址的设定异常时。
  - · 发生了IP重复异常时。

• 扩展错误代码 “ErrorIDEx”在本指令中具有CIP信息异常代码的含义。相应内容如下所示。

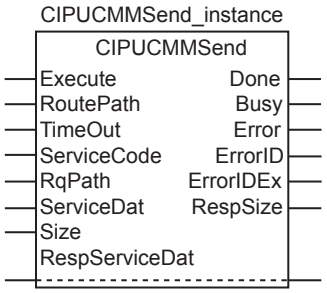
值	异常内容
16#02000000	在通信目标的节点为高负荷的状态下，无法正常通信。
16#04000000	指定的其它站变量为下列数据类型且在其它站中不存在。 · 基本数据类型 · 整个枚举体 · 整个结构体 · 整个联合体 · 整个数组
16#05000000	指定的其它站变量为下列数据类型且在其它站中不存在。 · 枚举体的枚举元素 · 结构体的1个结构要素 · 联合体的1个结构要素 · 数组的元素
16#08000000	不支持所请求的服务。
16#0C008010	指定的其它站变量正在下载中。
16#0C008011	
16#1F000102	指定的其它站变量中设定了常数属性，因此无法写入。
16#1F008007	指定了无法访问的变量。
16#20008017	指定的其它站变量不是数组，但写入元素数不是1。
16#20008018	指定的其它站变量是数组，写入元素数超过该数组的元素数。
16#20008028	· 指定的其它站变量为枚举体，写入数据为枚举元素中不存在的值。 · 指定的其它站变量具有范围指定属性，写入数据为超过范围的值。
16#26000000	指定的其它站变量名称仅为NULL字符。

示例程序

🕮 请参阅 “CIPUCMMSend指令(P.2–1034)”的示例程序。

CIPUCMMSend

将UCMM的CIP信息发送至CIP网络的指定设备。

指令	名称	FB/FUN	图形表现	ST表现
CIPUCMMSend	任意Explicit信息发送(UCMM)	FB		CIPUCMMSend_instance(Execute, RoutePath, TimeOut, ServiceCode, RqPath, ServiceDat, Size, RespServiceDat, Done, Busy, Error, ErrorID, ErrorIDEx, RespSize);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
RoutePath	路径	输入	路径	遵从数据类型	–	–
TimeOut	超时时间		超时时间	1 ~ 65535	0.1s	20 (2.0s)
ServiceCode	服务代码		服务代码	遵从数据类型	–	–
RqPath	请求路径		请求路径	–		
ServiceDat	命令数据		待发送的数据	遵从数据类型		(*)
Size	发送元素数		待发送的元素数			1
Resp ServiceDat	响应数据	输入输出	响应数据	遵从数据类型	–	–
RespSize	响应数据大小	输出	响应数据的大小	遵从数据类型	字节	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
RoutePath																				○
TimeOut							○													
ServiceCode		○																		
RqPath	结构体_sREQUEST_PATH或_sREQUEST_PATH_EX *1 详情参阅功能说明																			
ServiceDat		○	○	○	○	○	○	○	○	○	○	○	○	○	○					
	也可指定数组、结构体的1个结构要素、联合体的1个结构要素																			
Size							○													
Resp ServiceDat		○	○	○	○	○	○	○	○	○	○	○	○	○	○					
	也可指定数组、结构体的1个结构要素、联合体的1个结构要素																			
RespSize							○													

*1 _sREQUEST_PATH_EX型在CPU单元 Ver.1.11以上且Sysmac Studio Ver.1.15以上时可指定。

功能

对应服务代码 “ServiceCode”指定的服务，将指令数据 “ServiceDat”作为UCMM信息发送。
发送目标由路径 “RoutePath”指定。
请求路径由 “RqPath”指定。

待发送的元素数由 “Size”指定。“ServiceDat”为数组时为待发送的元素数，非数组时必定指定1。无需服务数据时，将 “Size”的值设为0。

发送后，响应数据保存至 “RespServiceDat”。响应数据的大小以字节为单位保存至 “RespSize”。

超时时间由 “TimeOut”指定。超时时间以内未返回响应时，判断为通信失败。

“RqPath”的数据类型为结构体_sREQUEST_PATH或_sREQUEST_PATH_EX。
通常请使用_sREQUEST_PATH。需设定任意逻辑格式的大小进行使用时，请使用_sREQUEST_PATH_EX。
规格如下所示。

• _sREQUEST_PATH型

变量	名称	内容	数据类型	有效范围	单位	初始值
RqPath	请求路径	请求路径	_sREQUEST_PATH	–	–	–
ClassID	等级ID	等级ID	UINT	遵从数据类型	–	0
InstanceID	实例ID	实例ID	UINT			0
isAttributeID	有无属性	TRUE：使用属性ID FALSE：不使用属性ID	BOOL			FALSE
AttributeID	属性ID	属性ID	UINT			0

(注) _sREQUEST_PATH型时，各ID的逻辑格式大小为16位。

• _sREQUEST_PATH_EX型

变量	名称	内容	数据类型	有效范围	单位	初始值
RqPath	请求路径	请求路径	_sREQUEST_PATH_EX	–	–	–
ClassIDLogicalFormat	等级ID逻辑格式	等级ID的数据大小	_eCIP_LOGICAL_FORMAT	遵从数据类型	–	_8BIT
ClassID	等级ID	等级ID	UDINT			0
InstanceIDLogicalFormat	实例ID逻辑格式	实例ID的数据大小	_eCIP_LOGICAL_FORMAT			_8BIT
InstanceID	实例ID	实例ID	UDINT			0
isAttributeID	有无属性	TRUE：使用属性ID FALSE：不使用属性ID	BOOL			FALSE
AttributeIDLogicalFormat	属性ID逻辑格式	属性ID的数据大小	_eCIP_LOGICAL_FORMAT			_8BIT
AttributeID	属性ID	属性ID	UDINT			0

“ClassIDLogicalFormat”、“InstanceIDLogicalFormat”、“AttributeIDLogicalFormat”的数据类型为枚举体_eCIP_LOGICAL_FORMAT。
枚举体_eCIP_LOGICAL_FORMAT的枚举元素的含义如下所述。

枚举元素	含义
_8BIT	8位
_16BIT	16位
_32BIT	32位

“ErrorID”的值为WORD#16#1C00时，在“ErrorIDEx”中保存CIP信息异常代码。“ErrorIDEx”的值和含义由对象节点规定。请参阅对象节点的手册。

收发数组时

“ServiceDat”或“RespServiceDat”为数组时，必须将带下标的数组元素作为参数传输至相应变量。

读写的数据大小

可读取的最大数据为492字节。
如下所示，可写入的最大数据因请求路径有无属性及路径而异。

可写入的数据大小[字节] = 基本大小 – 有无属性 – 路径信息大小

上式的项目	含义
基本大小	500字节
有无属性 *1	使用属性ID时为14字节 不使用属性ID时为10字节
路径信息大小	· 无跳跃时，路径信息大小为0字节。*2 · 有跳跃时，路径信息大小为路径大小+12字节。 · 路径大小是指路径的字符串的ASCII码的字节数。 · 请注意以下事项。 ・地址部分的开头带 ‘#’时，网络部分和地址部分共计2字节。 ・地址部分的开头无 ‘#’时，网络部分以2字节计算。 ・地址部分的开头无 ‘#’且地址部分的ASCII码的字节数为奇数时，将该值加上1字节。 ・表示路径层次间隔的 ‘\’不包含在路径大小中。 ・开头的跳跃不包含在路径大小中。 (例1) 路径为 ‘01\#11\02\192.168.250.2\01\#01’时 ・开头的跳跃不包含在路径中，因此忽略开头的 ‘01\#11’。 ・之后的网络类型 ‘02’为2字节。 ・之后的地址部分 ‘192.168.250.2’为13字节。为奇数，因此加上1为14字节。 ・之后的 ‘01#01’的地址部分带#，因此网络部分和地址部分共计2字节。 ・合计上述值，路径大小为18字节。 ・路径大小加上12字节，路径信息大小为30字节。 (例2) 路径为 ‘02\192.168.250.2\01\#00’时 ・开头的跳跃不包含在路径中，因此忽略开头的 ‘02\192.168.250.2’。 ・之后的 ‘01#00’的地址部分带#，因此网络部分和地址部分共计2字节。 ・路径大小为2字节。 ・路径大小加上12字节，路径信息大小为14字节。 (例3) 路径为 ‘02\192.168.250.2’时 ・无跳跃，因此路径信息大小为0字节。

*1 CPU单元Ver.1.10以下或Sysmac Studio Ver.1.14以下时，如下所述。
使用属性ID时为12字节
不使用属性ID时为8字节

*2 跳跃是指在发送节点和接收节点之间进行路径选择。例如，路径为02\192.168.250.2\01\#00时，一旦路径选择IP地址192.168.250.2的节点，则向00单元发送信息，因此存在1个跳跃。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

CIP通信的详情请参阅下列手册。

- 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”
- 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”
- 🕮 “CJ系列 EtherNet/IP单元 用户手册 NJ连接篇(SBCD-355)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮 “本章说明(P.2-3)”。
- 请务必将传输至 “ServiceDat”的输入参数设为变量。如果传输常数，编连时会发生异常。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口，或经由NJ系列 CPU单元连接的EtherNet/IP单元。
- 其它站控制器为欧姆龙制时，可写入的变量仅限于网络中公布的变量。因此，请先在网络中公开指定该变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “RqPath.ClassIDLogicalFormat”、“RqPath.AttributeIDLogicalFormat”的值超过有效范围时。
  - “RqPath.ClassIDLogicalFormat”指定的大小与 “RqPath.ClassID”的数据大小、“RqPath.InstanceIDLogicalFormat”指定的大小与 “RqPath.InstanceID”的数据大小或者“RqPath.AttributeIDLogicalFormat”指定的大小与 “RqPath.AttributeID”的数据大小不一致时。
  - “TimeOut”的值超过有效范围时。
  - “Size”的值超过写入数据范围时。
  - “Size”的值超过 “ServiceDat”的范围时。
  - “RespSize”的值超过 “RespServiceDat”的范围时。
  - “ServiceDat”中指定了不支持的数据类型时。
  - “RespServiceDat”中指定了不支持的数据类型时。
  - “RqPath”中指定了_sREQUEST_PATH或_sREQUEST_PATH_EX以外数据类型的变量时。
  - 发生本机IP地址的设定异常时。
  - 发生了IP重复异常时。
  - 在BOOTP服务器发生异常的状态下，执行了本指令时。
  - 返回了CIP规定的错误响应时。
  - “RoutePath”的字符串错误时。
  - 同时执行了超过32个的CIP相关指令时。
  - 超过超时时间仍无响应时。

示例程序

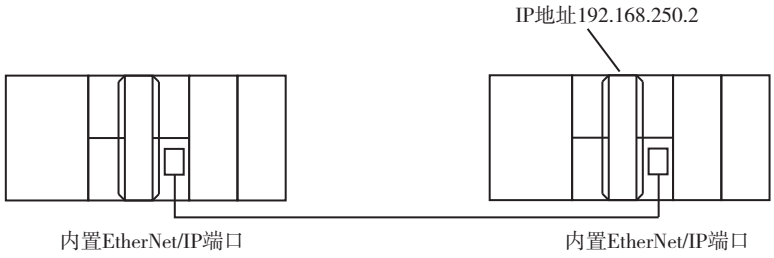
使用CIP的UCMM信息，执行变量写入、变量读取、信息发送。
控制器之间通过EtherNet/IP网络连接，对象节点的IP地址为192.168.250.2。
处理步骤如下所述。

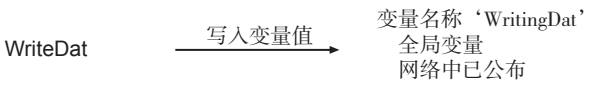
1 使用CIPUCMMWrite指令，将值写入对象节点的变量。其它站变量名称为 ‘WritingDat’，写入值为变量WriteDat的内容。‘WritingDat’需由对象节点定义为全局变量，并在网络中公布。

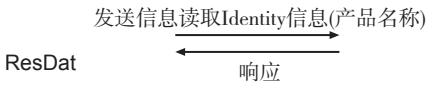
2 使用CIPUCMMRead指令读取对象节点的变量值。其它站变量名称为 ‘OriginalDat’，已读取值的保存位置为变量ReadDat。‘OriginalDat’需由对象节点定义为全局变量，并在网络中公布。

3 使用CIPUCMMSend指令，将Explicit信息发送至对象节点。信息内容为读取的Identity信息(产品名称)。此时的等级ID、实例ID、属性ID、服务代码如下所示。响应数据保存在变量ResDat中。

项目	值
等级ID	1
实例ID	1
属性ID	7
服务代码	16#0E

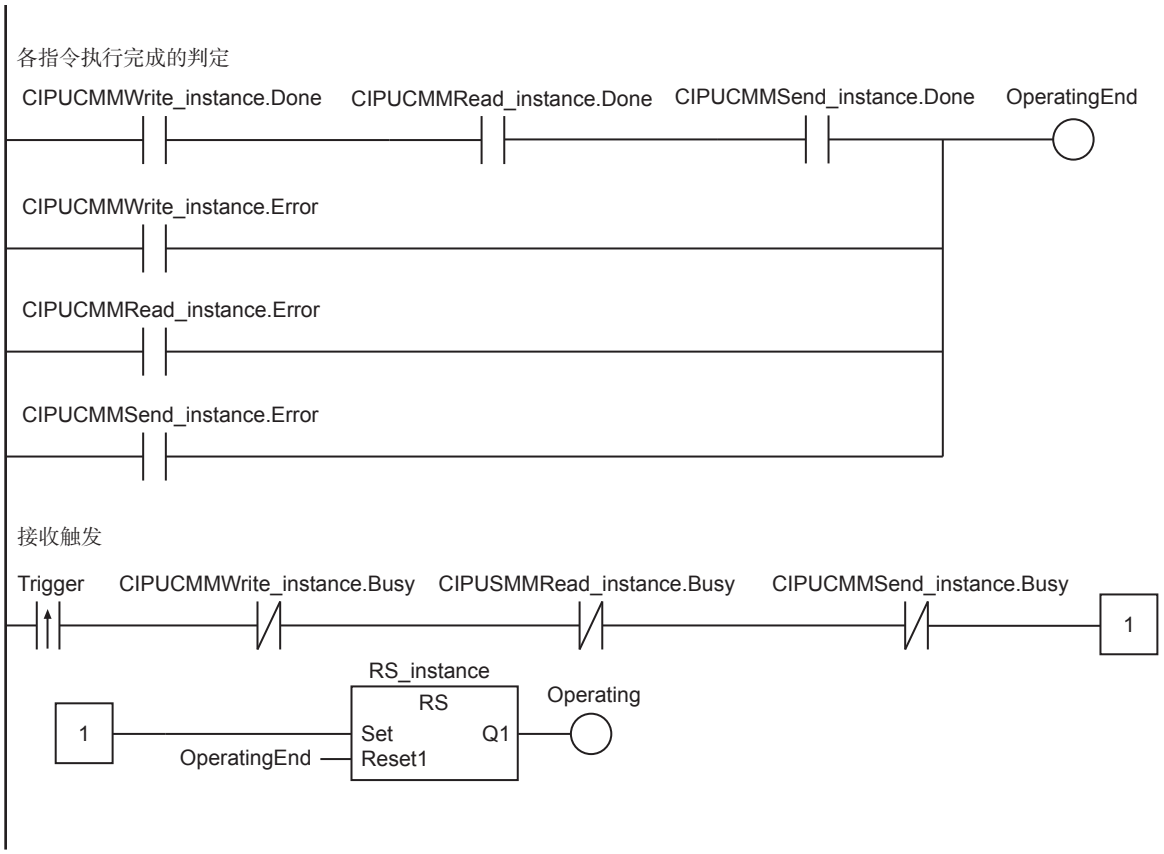


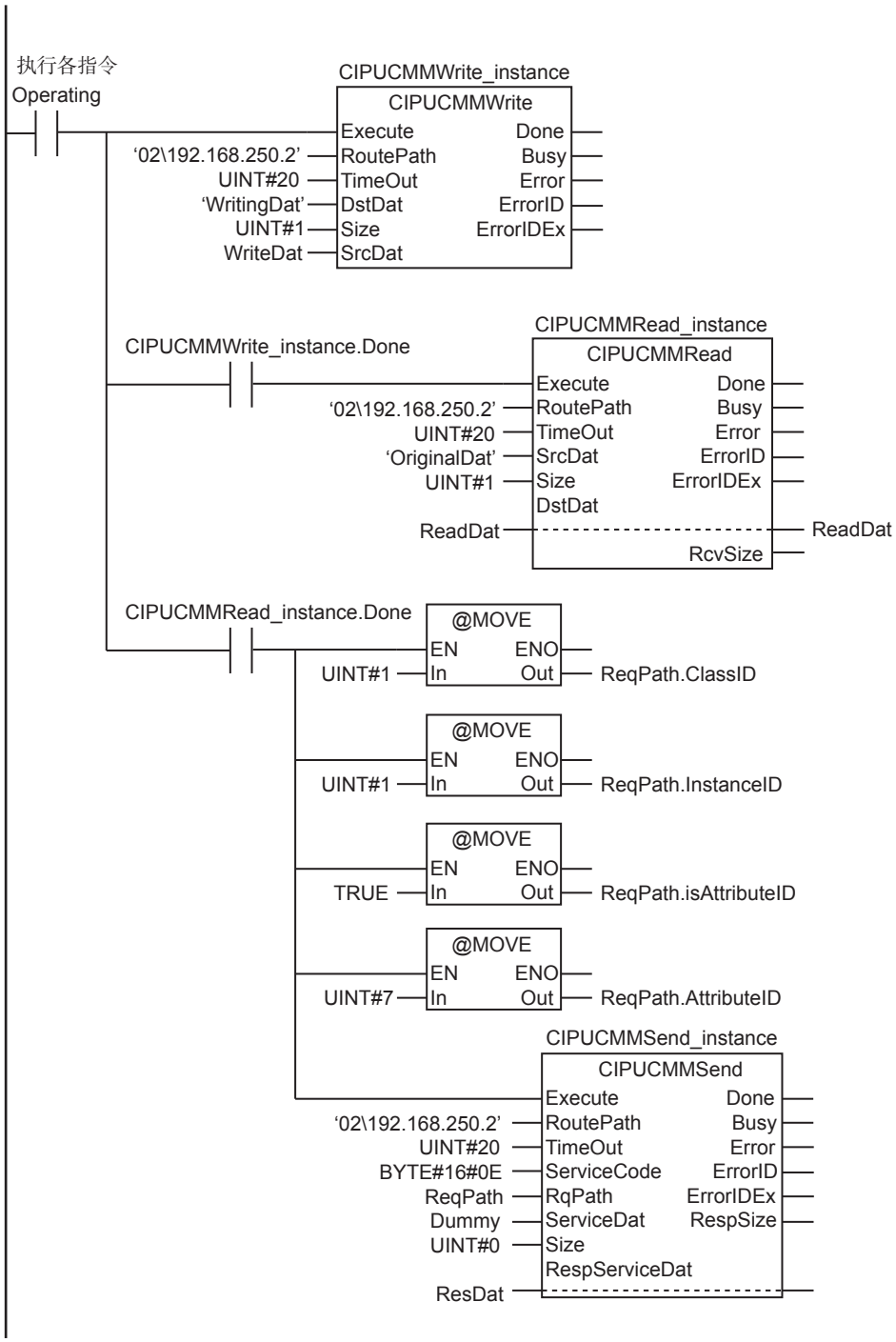


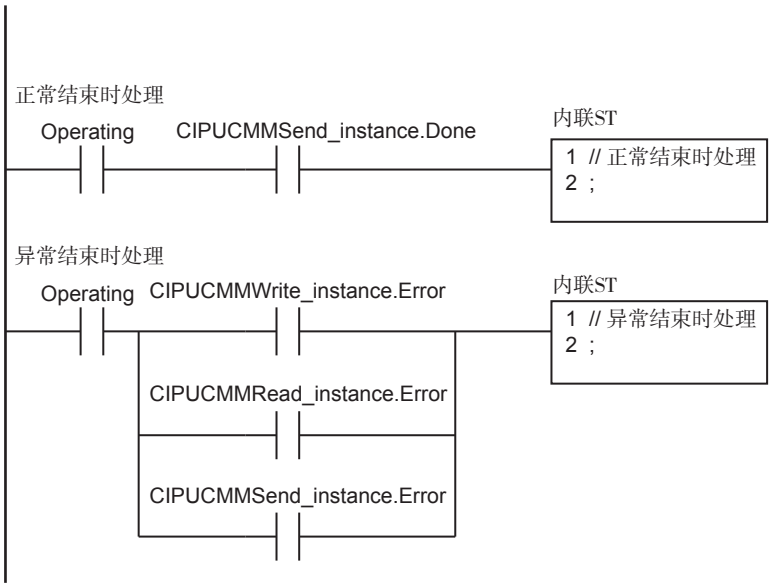


LD

名称	数据类型	初始值	注释
OperatingEnd	BOOL	FALSE	处理结束
Trigger	BOOL	FALSE	执行条件
Operating	BOOL	FALSE	处理中
WriteDat	INT	1234	写入数据
ReadDat	INT	0	读取数据
ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=FALSE, AttributeID:=0)	请求路径
ResDat	ARRAY[0..10] OF BYTE	[11(16#0)]	响应数据
Dummy	BYTE	16#0	虚拟
RS_instance	RS		
CIPUCMMWrite_instance	CIPUCMMWrite		
CIPUCMMRead_instance	CIPUCMMRead		
CIPUCMMSend_instance	CIPUCMMSend		




执行各指令
Operating
CIPUCMMWrite_instance
CIPUCMMWrite
Execute
RoutePath
TimeOut
DstDat
Size
SrcDat
Done
Busy
Error
ErrorID
ErrorIDEx
'02\192.168.250.2'
UINT#20
'WritingDat'
UINT#1
WriteDat
CIPUCMMWrite_instance.Done
CIPUCMMRead_instance
CIPUCMMRead
Execute
RoutePath
TimeOut
SrcDat
Size
DstDat
Done
Busy
Error
ErrorID
ErrorIDEx
RcvSize
'02\192.168.250.2'
UINT#20
'OriginalDat'
UINT#1
ReadDat
ReadDat
CIPUCMMRead_instance.Done
@MOVE
EN
In
ENO
Out
UINT#1
ReqPath.ClassID
@MOVE
UINT#1
ReqPath.InstanceID
@MOVE
TRUE
ReqPath.isAttributeID
@MOVE
UINT#7
ReqPath.AttributeID
CIPUCMMSend_instance
CIPUCMMSend
Execute
RoutePath
TimeOut
ServiceCode
RqPath
ServiceDat
Size
RespServiceDat
Done
Busy
Error
ErrorID
ErrorIDEx
RespSize
'02\192.168.250.2'
UINT#20
BYTE#16#0E
ReqPath
Dummy
UINT#0
ResDat


正常结束时处理
Operating
CIPUCMMSend_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating
CIPUCMMWrite_instance.Error
内联ST
1 // 异常结束时处理
2 ;
CIPUCMMRead_instance.Error
CIPUCMMSend_instance.Error

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoUCMMTrigger	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	WriteDat	INT	0	写入数据
	ReadDat	INT	0	读取数据
	ReqPath	_sREQUEST_PATH	(ClassID:=0, InstanceID:=0, isAttributeID:=FALSE, AttributeID:=0)	请求路径
	ResDat	ARRAY[0..10] OF BYTE	[11(16#0)]	响应数据
	Dummy	BYTE	16#0	虚拟
	CIPUCMMWrite_instance	CIPUCMMWrite		
	CIPUCMMRead_instance	CIPUCMMRead		
	CIPUCMMSend_instance	CIPUCMMSend		

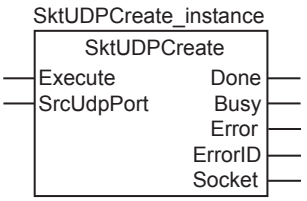
外部变量	名称	常数	数据类型	注释
	_EIP_EtnOnlineSta	☑	BOOL	在线

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoUCMMTrigger=FALSE) AND (_Eip_EtnOnlineSta=TRUE) ) THEN
  DoUCMMTrigger      :=TRUE;
  Stage              :=INT#1;
  CIPUCMMWrite_instance(
    Execute      :=FALSE,          // 实例初始化
    SrcDat       :=WriteDat);      // 虚拟
  CIPUCMMRead_instance(            // 实例初始化
    Execute      :=FALSE,          // 虚拟
    DstDat       :=ReadDat);       // 虚拟
  CIPUCMMSend_instance(
    Execute      :=FALSE,          // 实例初始化
    ServiceDat   := Dummy,         // 虚拟
    RespServiceDat:=ResDat);       // 虚拟
END_IF;

IF (DoUCMMTrigger=TRUE) THEN
  CASE Stage OF
  1 :                              // 变量写入请求
    CIPUCMMWrite_instance(
      Execute  :=TRUE,
      RoutePath:= '02\192.168.250.2',  // 路径
      TimeOut  :=UINT#20,          // 超时时间
      DstDat   := 'WritingDat',    // 其它站变量名称
      Size     :=UINT#1,           // 写入元素数
      SrcDat   :=WriteDat);        // 写入数据

    IF (CIPUCMMWrite_instance.Done=TRUE) THEN
      Stage:=INT#2;                // 正常结束
    ELSIF (CIPUCMMWrite_instance.Error=TRUE) THEN
      Stage:=INT#10;               // 异常结束
    END_IF;
```

```
    2 :                          // 变量读取请求
      CIPUCMMRead_instance(
        Execute    :=TRUE,
        RoutePath := '02\192.168.250.2',        // 路径
        TimeOut   :=UINT#20,                    // 超时时间
        SrcDat     := 'OriginalDat',            // 其它站变量名称
        Size       :=UINT#1,                    // 读取元素数
        DstDat     :=ReadDat);                  // 读取数据

      IF (CIPUCMMRead_instance.Done=TRUE) THEN
        Stage:=INT#3;                          // 正常结束
      ELSIF (CIPUCMMRead_instance.Error=TRUE) THEN
        Stage:=INT#40;                         // 异常结束
      END_IF;

    3 :                          // 发送任意信息
      ReqPath.ClassID        :=UINT#01;
      ReqPath.InstanceID     :=UINT#01;
      ReqPath.isAttributeID :=TRUE;
      ReqPath.AttributeID    :=UINT#07;
      CIPUCMMSend_instance(
        Execute            :=TRUE,
        RoutePath          := '02\192.168.250.2',      // 路径
        TimeOut            :=UINT#20,                  // 超时值
        ServiceCode        :=BYTE#16#0E,               // 服务代码
        RqPath             :=ReqPath,                  // 请求路径
        ServiceDat         := Dummy,                   // 服务数据
        Size               :=UINT#0,                   // 元素数
        RespServiceDat :=ResDat);                      // 响应数据

      IF (CIPUCMMSend_instance.Done=TRUE) THEN
        Stage:=INT#0;                          // 正常结束
      ELSIF (CIPUCMMSend_instance.Error=TRUE) THEN
        Stage:=INT#30;                         // 异常结束
      END_IF;

    0 :                          // 正常结束处理
      DoUCMMTrigger:=FALSE;
      Trigger              :=FALSE;

    ELSE                                       // 异常结束处理
      DoUCMMTrigger:=FALSE;
      Trigger              :=FALSE;
    END_CASE;
  END_IF;
```

SktUDPCreate

执行内置EtherNet/IP的UDP Socket的建立请求(服务器端口打开)。

指令	名称	FB/FUN	图形表现	ST表现
SktUDP Create	UDP Socket建立	FB		SktUDPCreate_instance(Execute, SrcUdpPort, Done, Busy, Error, ErrorID, Socket);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
SrcUdp Port	本机UDP端口编号	输入	本机UDP端口编号	1 ~ 65535	–	1
Socket	Socket	输出	Socket	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
SrcUdp Port							○													
Socket	结构体_sSOCKET　详情参阅功能说明																			

功能

打开本机UDP端口编号 “SrcUdpPort”指定端口(执行Socket函数Socket( )、Bind( ))。
打开Socket的信息保存在 “Socket”中。
在指令正常结束(“Done”的值变为TRUE)时，打开UDP端口。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr*1	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr*1	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr*1	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo*1	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr*1	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

*1 结构要素输出 “0”或NULL。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅🕮 “本章说明(P.2–3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 请使用SktClose指令关闭本指令生成的句柄。
- 本指令生成的句柄在变更程序模式时无效。
- SktUDPCreate 指令、SktUDPRcv 指令、SktUDPSend 指令、SktTCPAccept 指令、SktTCPConnect 指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。

- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “SrcUdpPort”的值超过有效范围时。
  - “SrcUdpPort”指定的端口已打开或者处于关闭处理中时。
  - “SrcUdpPort”指定的端口已在使用时。

版本相关信息

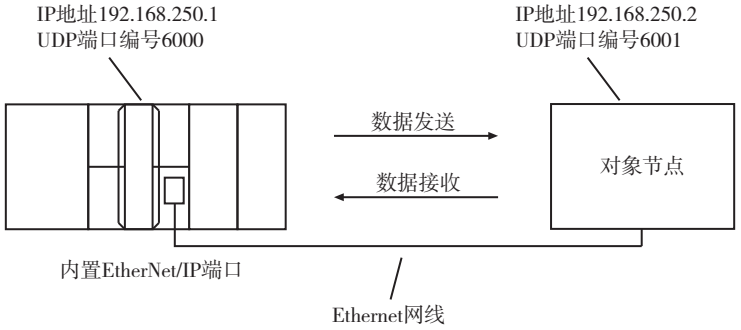
- 根据CPU单元版本的不同，可以同时打开的Socket数有所不同，如下所示。Socket数是UDP Socket和TCP Socket的合计数量。

CPU单元的版本	Socket数
Ver.1.03以上	最大30
Ver.1.02以下	最大16

- CPU单元Ver.1.10以上版本时，即使 “Error”变为TRUE，也不会变更 “Socket”的值。Ver.1.09以下版本时，会将 “Socket”的值变更为 “0”。

示例程序

在NJ/NX系列控制器和对象节点之间，使用Socket服务功能(UDP)收发数据。



处理步骤如下所示。

1. 利用SktUDPCreate指令执行UDP Socket的建立请求。
2. 利用SktUDPSend指令执行发送请求。要发送的数据为SendSocketDat[ ]的内容。
3. 利用SktUDPRcv指令执行接收请求。接收数据保存在RcvSocketDat[ ]中。
4. 利用SktClose指令关闭Socket。

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoSendAndRcv	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	接收数据
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	发送数据
	SktUDPCreate_instance	SktUDPCreate		
	SktUDPSend_instance	SktUDPSend		
	SktUDPRcv_instance	SktUDPRcv		
	SktClose_instance	SktClose		

外部变量	名称	数据类型	常数	注释
	_EIP_EtnOnlineSta	BOOL	☑	在线

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoSendAndRcv=FALSE) AND (_Eip_EtnOnlineSta=TRUE) ) THEN
    DoSendAndRcv :=TRUE;
    Stage              :=INT#1;
    SktUDPCreate_instance(Execute:=FALSE);      // 实例初始化
    SktUDPSend_instance(                        // 实例初始化
        Execute :=FALSE,
        SendDat :=SendSocketDat[0]);            // 虚拟
    SktUDPRcv_instance(                         // 实例初始化
        Execute :=FALSE,
        RcvDat  :=RcvSocketDat[0]);             // 虚拟
    SktClose_instance(Execute:=FALSE);          // 实例初始化
END_IF;

IF (DoSendAndRcv=TRUE) THEN
    CASE Stage OF
    1 :                                         // 建立请求
        SktUDPCreate_instance(
            Execute     :=TRUE,
            SrcUdpPort :=UINT#6000,             // 本机UDP端口编号
            Socket      =>WkSocket);            // Socket

        IF (SktUDPCreate_instance.Done=TRUE) THEN
            Stage:=INT#2;                       // 正常结束
        ELSIF (SktUDPCreate_instance.Error=TRUE) THEN
            Stage:=INT#10;                      // 异常结束
        END_IF;

    2 :                                         // 发送请求
        WkSocket.DstAdr.PortNo:=UINT#6001;
        WkSocket.DstAdr.IpAdr := '192.168.250.2';
        SktUDPSend_instance(
            Execute:=TRUE,
            Socket  :=WkSocket,                 // Socket
```

```
            SendDat:=SendSocketDat[0],          // 发送数据
            Size    :=UINT#2000);               // 发送数据大小

        IF (SktUDPSend_instance.Done=TRUE) THEN
            Stage:=INT#3;                       // 正常结束
        ELSIF (SktUDPSend_instance.Error=TRUE) THEN
            Stage:=INT#20;                      // 异常结束
        END_IF;

    3 :                                         // 接收请求
        SktUDPRcv_instance(
            Execute:=TRUE,
            Socket  :=WkSocket,                 // Socket
            TimeOut:=UINT#0,                    // 超时时间
            Size    :=UINT#2000,                // 接收数据大小
            RcvDat :=RcvSocketDat[0]);          // 接收数据

        IF (SktUDPRcv_instance.Done=TRUE) THEN
            Stage:=INT#4;                       // 正常结束
        ELSIF (SktUDPRcv_instance.Error=TRUE) THEN
            Stage:=INT#30;                      // 异常结束
        END_IF;

    4 :                                         // 关闭请求
        SktClose_instance(
            Execute:=TRUE,
            Socket  :=WkSocket);                // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage:=INT#0;                       // 正常结束
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage:=INT#40;                      // 异常结束
        END_IF;

    0 :                                         // 正常结束
        DoSendAndRcv:=FALSE;
        Trigger      :=FALSE;

    ELSE                                        // 基于异常的中断
        DoSendAndRcv:=FALSE;
        Trigger      :=FALSE;
    END_CASE;

END_IF;
```

● 对象节点的用户程序

在本例中，对象节点也需要用户程序。上述的处理步骤中，发送和接收的顺序相反。

1 利用SktUDPCreate指令执行UDP Socket的建立请求。

2 利用SktUDPRcv指令执行接收请求。接收数据保存在RcvSocketDat[ ]中。

3 利用SktUDPSend指令执行发送请求。要发送的数据为SendSocketDat[ ]的内容。

4 利用SktClose指令关闭Socket。

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoSendAndRcv	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	接收数据
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	发送数据
	SktUDPCreate_instance	SktUDPCreate		
	SktUDPSend_instance	SktUDPSend		
	SktUDPRcv_instance	SktUDPRcv		
	SktClose_instance	SktClose		

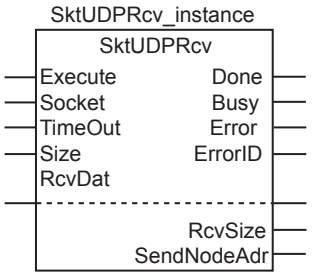
外部变量	名称	数据类型	常数	注释
	_EIP_EtnOnlineSta	BOOL	☑	在线

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoSendAndRcv=FALSE) AND (_Eip_EtnOnlineSta=TRUE) ) THEN
    DoSendAndRcv :=TRUE;
    Stage              :=INT#1;
    SktUDPCreate_instance(Execute:=FALSE);      // 实例初始化
    SktUDPSend_instance(                        // 实例初始化
        Execute :=FALSE,
        SendDat :=SendSocketDat[0]);            // 虚拟
    SktUDPRcv_instance(                         // 实例初始化
        Execute :=FALSE,
        RcvDat  :=RcvSocketDat[0]);             // 虚拟
    SktClose_instance(Execute:=FALSE);          // 实例初始化
END_IF;

IF (DoSendAndRcv=TRUE) THEN
    CASE Stage OF
    1 :                                         // 建立请求
        SktUDPCreate_instance(
            Execute     :=TRUE,
            SrcUdpPort :=UINT#6001,             // 本机UDP端口编号
            Socket      =>WkSocket);            // Socket

        IF (SktUDPCreate_instance.Done=TRUE) THEN
            Stage:=INT#2;                       // 正常结束
        ELSIF (SktUDPCreate_instance.Error=TRUE) THEN
            Stage:=INT#10;                      // 异常结束
        END_IF;
```

```
    2 :                                         // 接收请求
        SktUDPRcv_instance(
            Execute  :=TRUE,
            Socket   :=WkSocket,                // Socket
            TimeOut :=UINT#0,                   // 超时时间
            Size     :=UINT#2000,               // 接收数据大小
            RcvDat   :=RcvSocketDat[0]);        // 接收数据

        IF (SktUDPRcv_instance.Done=TRUE) THEN
            Stage:=INT#3;                       // 正常结束
        ELSIF (SktUDPRcv_instance.Error=TRUE) THEN
            Stage:=INT#20;                      // 异常结束
        END_IF;

    3 :                                         // 发送请求
        WkSocket.DstAdr.PortNo:=UINT#6000;
        WkSocket.DstAdr.IpAdr := '192.168.250.1';
        SktUDPSend_instance(
            Execute  :=TRUE,
            Socket   :=WkSocket,                // Socket
            SendDat :=SendSocketDat[0],         // 发送数据
            Size     :=UINT#2000);              // 发送数据大小

        IF (SktUDPSend_instance.Done=TRUE) THEN
            Stage:=INT#4;                       // 正常结束
        ELSIF (SktUDPSend_instance.Error=TRUE) THEN
            Stage:=INT#30;                      // 异常结束
        END_IF;

      4 :                                       // 关闭请求
        SktClose_instance(
            Execute :=TRUE,
            Socket   :=WkSocket);               // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage:=INT#0;                       // 正常结束
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage:=INT#40;                      // 异常结束
        END_IF;

    0 :                                         // 正常结束
        DoSendAndRcv :=FALSE;
        Trigger          :=FALSE;

    ELSE                                        // 基于异常的中断
        DoSendAndRcv :=FALSE;
        Trigger          :=FALSE;
    END_CASE;

END_IF;
```

SktUDPRcv

读取内置EtherNet/IP的UDP Socket接收缓存中的数据。

指令	名称	FB/FUN	图形表现	ST表现
SktUDPRcv	UDP Socket接收	FB		SktUDPRcv_instance(Execute, Socket, TimeOut, Size, RcvDat, Done, Busy, Error, ErrorID, RcvSize, SendNodeAdr);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–
TimeOut	超时时间		0 ：无超时 1 ~ 65535：0.1 ~ 6553.5s	遵从数据类型	0.1s	0
Size	保存容量		从接收缓存读取的数据大小	0 ~ 2000	字节	1
RcvDat[ ]数组	接收数据	输入输出	接收数据	遵从数据类型	–	–
RcvSize	接收数据大小	输出	实际保存到RcvDat[ ]中的数据大小	0 ~ 2000	字节	–
SendNodeAdr	发送源节点地址		发送源节点地址	–	–	

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET 详情参阅功能说明																			
TimeOut							○													
Size							○													
RcvDat[ ]数组		○																		
RcvSize							○													
SendNodeAdr	结构体_sSOCKET_ADDRESS 详情参阅功能说明																			

功能

将 “Socket”指定的Socket中接收缓存内的数据保存到接收数据RcvDat[ ]中。利用 “Size”指定要保存数据的大小。

然后，在保存数据后，将实际保存的数据大小代入 “RcvSize”。

发送源的节点地址保存在 “SendNodeAdr”中。

如果接收缓存中无数据，则按照超时时间 “TimeOut”中设定的时间等待接收数据。
在指令正常结束(“Done”的值变为TRUE)时，将数据保存到RcvDat[ ]中。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

“SendNodeAdr”的数据类型为结构体_sSOCKET_ADDRESS。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
SendNodeAdr	发送源节点地址	发送源节点地址	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	发送源节点的UDP端口编号	UINT	1 ~ 65535	–	–
IpAdr	IP地址	发送源节点的IP地址	STRING	遵从数据类型		

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

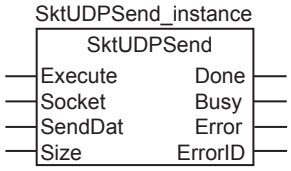
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 一次可接收的数据容量最大为2000字节。
- 已接收指定Socket的数据容量低于 “Size”的值时，仅将已接收数据保存在RcvDat[ ]中。然后，将已保存数据容量代入 “RcvSize”。
- 已接收指定Socket的数据容量高于 “Size”的值时，仅将 “Size”量的数据保存在RcvDat[ ]中。
- “Size”的值为0时，不读取接收数据。
- 接收缓存中无数据时，如果利用SktClose指令关闭Socket，则即使低于超时时间，也会不再等待接收数据而正常结束。此时，“RcvSize”的值变为0。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “Socket”指定的Socket在接收处理中时。
  - “Socket”指定的Socket在未打开时。
  - “Socket.Handle”指定的句柄不存在时。

示例程序

请参阅 “SktUDPCreate指令(P.2-1045)”的示例程序。

SktUDPSend

从内置EtherNet/IP的UDP端口发送数据。

指令	名称	FB/FUN	图形表现	ST表现
SktUDPSend	UDP Socket发送	FB		SktUDPSend_instance(Execute, Socket, SendDat, Size, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–
SendDat[ ]数组	发送数据		发送数据	遵从数据类型		
Size	发送数据大小		发送数据大小	0 ~ 2000	字节	1

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET 详情参阅功能说明																			
SendDat[ ]数组		○																		
Size							○													

功能

从 “Socket”指定的Socket发送发送数据SendDat[ ]。
利用 “Size”指定要发送的数据容量。
利用 “Socket.DstAdr”指定对象节点。
在指令正常结束(“Done”的值变为TRUE)时，将数据SendDat[ ]保存到发送缓存中。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

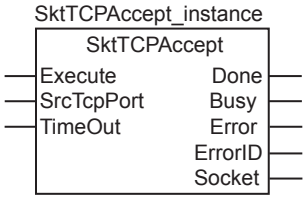
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 一次可发送的数据容量最大为2000字节。SendDat[ ]的数组容量高于2000字节时，最多也只能发送2000字节。还有，指定广播地址时，最多只能发送1472字节。
- “Size”的值为0时，将0字节的发送数据发送到回线。
- SktUDPCreate 指令、SktUDPRcv 指令、SktUDPSend 指令、SktTCPAccept 指令、SktTCPConnect 指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “Socket”的任一结构要素超过有效范围时。
  - “Socket”指定的Socket在发送处理中时。
  - “Socket”指定的Socket在未打开时。
  - “Socket”指定对象节点的域名，地址解析失败时。
  - “Socket.Handle”指定的句柄不存在时。
  - “Size”的值超过SendDat[ ]的元素数时。

示例程序

请参阅 “SktUDPCreate指令(P.2–1045)”的示例程序。

SktTCPAccept

执行内置EtherNet/IP的TCP Socket的接受请求。

指令	名称	FB/FUN	图形表现	ST表现
SktTCP Accept	TCP Socket接受	FB		SktTCPAccept_instance(Execute, SrcTcpPort, TimeOut, Done, Busy, Error, ErrorID, Socket);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
SrcTcpPort	本机TCP端口编号	输入	本机TCP端口编号	1 ~ 65535	–	1
TimeOut	超时时间		0 ：无超时 1 ~ 65535：0.1 ~ 6553.5s	遵从数据类型	0.1s	0
Socket	Socket	输出	Socket	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
SrcTcpPort							○													
TimeOut							○													
Socket	结构体_sSOCKET 详情参阅功能说明																			

功能

执行本机TCP端口编号 “SrcTcpPort”指定端口的接受请求(Socket函数Socket( )、Bind( )、Listen( )、Accept( )的执行)。
按照超时时间 “TimeOut”设定的时间，等待从通信对象建立连接。
在指令正常结束(“Done”的值变为TRUE)时，建立连接。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr*1	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

*1 结构要素输出NULL。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

- Socket服务功能的详情请参阅 🕮“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或 🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。
- 多次执行本指令，能够以一个本机端口编号从多个客户端开通连接。其时，针对各连接返回不同的Socket。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 请使用SktClose指令关闭本指令生成的句柄。
- 本指令生成的句柄在变更程序模式时无效。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “SrcTcpPort”的值超过有效范围时。
  - “SrcTcpPort”指定的TCP端口处于打开处理中时。
  - “SrcTcpPort”指定的TCP端口处于关闭处理中时。
  - 即使等待 “TimeOut”指定时间，仍未建立连接时。

版本相关信息

- 根据CPU单元版本的不同，可以同时打开的Socket数有所不同，如下所示。Socket数是UDP Socket和TCP Socket的合计数量。

CPU单元的版本	Socket数
Ver.1.03以上	最大30
Ver.1.02以下	最大16

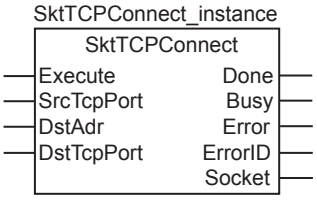
- CPU单元Ver.1.10以上版本时，即使 “Error”变为TRUE，也不会变更 “Socket”的值。Ver.1.09以下版本时，会将 “Socket”的值变更为 “0”。

示例程序

请参阅 “SktTCPConnect指令(P.2-1061)”的示例程序。

SktTCPConnect

从内置EtherNet/IP连接目标TCP端口。

指令	名称	FB/FUN	图形表现	ST表现
SktTCPConnect	TCP Socket连接	FB		SktTCPConnect_instance(Execute, SrcTcpPort, DstAdr, DstTcpPort, Done, Busy, Error, ErrorID, Socket);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
SrcTcpPort	本机TCP端口编号	输入	本机TCP端口编号 为0时，自动分配从Well-Known端口以外的1024开始的空闲TCP端口	遵从数据类型	–	0
DstAdr	对方地址		对方的IP地址或主机名称	最大200字节		–
DstTcpPort	对方TCP端口编号		对方TCP端口编号	1 ~ 65535		1
Socket	Socket	输出	Socket	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
SrcTcpPort							○													
DstAdr																				○
DstTcpPort							○													
Socket	结构体_sSOCKET　详情参阅功能说明																			

功能

执行连接本机TCP端口编号“SrcTcpPort”和对方地址“DstAdr”的对方TCP端口编号 “DstTcpPort”的连接请求(Socket函数Connect( )的执行)。
在指令正常结束(“Done”的值变为TRUE)时，建立连接。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr*1	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 结构要素输出NULL。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket 服务功能的详情请参阅 🕮 “NJ/NX 系列 CPU 单元内置 EtherNet/IP 端口 用户手册 (SBCD–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 请使用SktClose指令关闭本指令生成的句柄。
- 本指令生成的句柄在变更程序模式时无效。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “DstAdr”的值超过有效范围时。
  - “DstTcpPort”的值超过有效范围时。
  - “SrcTcpPort”指定的TCP端口已打开时。
  - “DstAdr”指定的对象节点不存在时。
  - “DstAdr”、“DstTcpPort”指定的对象节点不处于连接等待状态时。
  - “DstAdr”指定主机名称的地址解析失败时。
  - 已建立与同一客户端(IP地址、TCP端口)的连接时。

版本相关信息

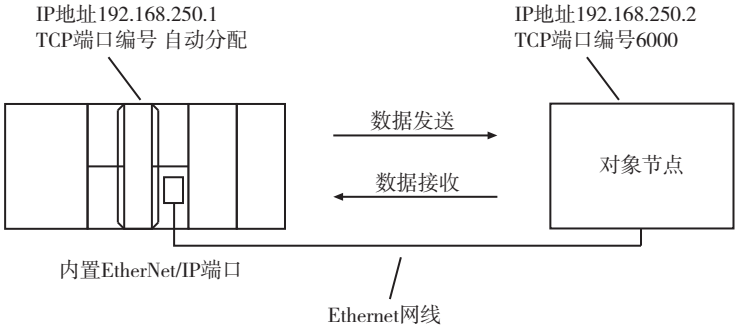
- 根据CPU单元版本的不同，可以同时打开的Socket数有所不同，如下所示。Socket数是UDP Socket和TCP Socket的合计数量。

CPU单元的版本	Socket数
Ver.1.03以上	最大30
Ver.1.02以下	最大16

- CPU单元Ver.1.10以上版本时，即使 “Error”变为TRUE，也不会变更 “Socket”的值。Ver.1.09以下版本时，会将 “Socket”的值变更为 “0”。

示例程序

在NJ/NX系列控制器和对象节点之间，使用Socket服务功能(TCP)收发数据。



处理步骤如下所示。

1. 利用SktTCPConnect指令，对对象节点的TCP端口执行连接请求。
2. 利用SktClearBuf指令，清除TCP Socket的接收缓存。
3. 利用SktGetTCPStatus指令，读取TCP Socket的状态。
4. 利用SktTCPSend指令执行发送请求。要发送的数据为SendSocketDat[ ]的内容。
5. 利用SktTCPRcv指令执行接收请求。接收数据保存在RcvSocketDat[ ]中。
6. 利用SktClose指令关闭Socket。

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoTCP	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	接收数据
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	发送数据
	SktTCPConnect_instance	SktTCPConnect		
	SktClearBuf_instance	SktClearBuf		
	SktGetTCPStatus_instance	SktGetTCPStatus		
	SktTCPSend_instance	SktTCPSend		
	SktTCPRcv_instance	SktTCPRcv		
	SktClose_instance	SktClose		

外部变量	名称	数据类型	常数	注释
	_EIP_EtnOnlineSta	BOOL	☑	在线

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoTCP=FALSE) AND (_Eip_EtnOnlineSta=TRUE) ) THEN
    DoTCP :=TRUE;
    Stage   :=INT#1;
    SktTCPConnect_instance(Execute:=FALSE);          // 实例初始化
    SktClearBuf_instance(Execute:=FALSE);            // 实例初始化
    SktGetTCPStatus_instance(Execute:=FALSE);        // 实例初始化
    SktTCPSend_instance(                             // 实例初始化
        Execute :=FALSE,
        SendDat :=SendSocketDat[0]);                 // 虚拟
    SktTCPRcv_instance(                              // 实例初始化
        Execute :=FALSE,
        RcvDat  :=RcvSocketDat[0]);                  // 虚拟
    SktClose_instance(Execute:=FALSE);               // 实例初始化
END_IF;

IF (DoTCP=TRUE) THEN
    CASE Stage OF
    1 :                                              // 连接请求
        SktTCPConnect_instance(
            Execute     :=TRUE,
            SrcTcpPort :=UINT#0,                      // 本机UDP端口编号   自动分配
            DstAdr      := '192.168.250.2',           // 对方IP地址
            DstTcpPort :=UINT#6000,                   // 对方TCP端口编号
            Socket      =>WkSocket);                  // Socket

        IF (SktTCPConnect_instance.Done=TRUE) THEN
            Stage:=INT#2;                             // 正常结束
        ELSIF (SktTCPConnect_instance.Error=TRUE) THEN
            Stage:=INT#10;                            // 异常结束
        END_IF;

    2 :                                              // 清除接收缓存
        SktClearBuf_instance(
            Execute :=TRUE,
            Socket  :=WkSocket);                      // Socket

        IF (SktClearBuf_instance.Done=TRUE) THEN
            Stage:=INT#3;                             // 正常结束
        ELSIF (SktClearBuf_instance.Error=TRUE) THEN
            Stage:=INT#20;                            // 异常结束
        END_IF;

    3 :                                              // 状态读取请求
        SktGetTCPStatus_instance(
            Execute :=TRUE,
            Socket  :=WkSocket);                      // Socket

        IF (SktGetTCPStatus_instance.Done=TRUE) THEN
            Stage:=INT#4;                             // 正常结束
        ELSIF (SktGetTCPStatus_instance.Error=TRUE) THEN
            Stage:=INT#30;                            // 异常结束
        END_IF;
```

```
    4 :                                          // 发送请求
        SktTCPSend_instance(
            Execute :=TRUE,
            Socket  :=WkSocket,                  // Socket
            SendDat :=SendSocketDat[0],          // 发送数据
            Size    :=UINT#2000);                // 发送数据大小

        IF (SktTCPSend_instance.Done=TRUE) THEN
            Stage:=INT#5;                        // 正常结束
        ELSIF (SktTCPSend_instance.Error=TRUE) THEN
            Stage:=INT#40;                       // 异常结束
        END_IF;

    5 :                                          // 接收请求
        SktTCPRcv_instance(
            Execute  :=TRUE,
            Socket   :=WkSocket,                 // Socket
            TimeOut  :=UINT#0,                   // 超时时间
            Size     :=UINT#2000,                // 接收数据大小
            RcvDat   :=RcvSocketDat[0]);         // 接收数据

        IF (SktTCPRcv_instance.Done=TRUE) THEN
            Stage:=INT#6;                        // 正常结束
        ELSIF (SktTCPRcv_instance.Error=TRUE) THEN
            Stage:=INT#50;                       // 异常结束
        END_IF;

    6 :                                          // 关闭请求
        SktClose_instance(
            Execute :=TRUE,
            Socket  :=WkSocket);                 // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage:=INT#0;                        // 正常结束
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage:=INT#40;                       // 异常结束
        END_IF;

    0 :                                          // 正常结束
        DoTCP   :=FALSE;
        Trigger  :=FALSE;

    ELSE                                         // 基于异常的中断
        DoTCP   :=FALSE;
        Trigger  :=FALSE;
    END_CASE;

END_IF;
```

- **对象节点的用户程序**

  在本例中，对象节点也需要用户程序。上述的处理步骤中，发送和接收的顺序相反。

  1 利用SktTCPAccept指令执行TCP。

  2 利用SktTCPRcv指令执行接收请求。接收数据保存在RcvSocketDat[ ]中。

  3 利用SktTCPSend指令执行发送请求。要发送的数据为SendSocketDat[ ]的内容。

  4 利用SktClose指令关闭Socket。

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoTCP	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	RcvSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	接收数据
	WkSocket	_sSOCKET	(Handle:=0, SrcAdr:=(PortNo:=0, IpAdr:=''), DstAdr:=(PortNo:=0, IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE	[2000(16#0)]	发送数据
	SktTCPAccept_instance	SktTCPAccept		
	SktTCPSend_instance	SktTCPSend		
	SktTCPRcv_instance	SktTCPRcv		
	SktClose_instance	SktClose		

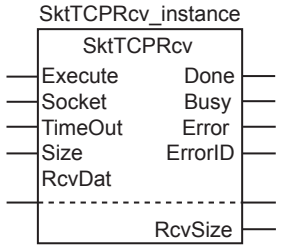
外部变量	名称	数据类型	常数	注释
	_EIP_EtnOnlineSta	BOOL	☐	在线

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoTCP=FALSE) AND (_Eip_EtnOnlineSta=TRUE) ) THEN
    DoTCP :=TRUE;
    Stage  :=INT#1;
    SktTCPAccept_instance(Execute:=FALSE);          // 实例初始化
    SktTCPSend_instance(                            // 实例初始化
        Execute :=FALSE,
        SendDat :=SendSocketDat[0]);                // 虚拟
    SktTCPRcv_instance(                             // 实例初始化
        Execute :=FALSE,
        RcvDat  :=RcvSocketDat[0]);                 // 虚拟
    SktClose_instance(Execute:=FALSE);              // 实例初始化
END_IF;

IF (DoTCP=TRUE) THEN
    CASE Stage OF
    1 :                                             // 接受请求
        SktTCPAccept_instance(
            Execute     :=TRUE,
            SrcTcpPort :=UINT#6000,                 // 本机TCP端口编号
            TimeOut     :=UINT#0,                   // 超时时间
            Socket      =>WkSocket);                // Socket

        IF (SktTCPAccept_instance.Done=TRUE) THEN
            Stage:=INT#2;                           // 正常结束
        ELSIF (SktTCPAccept_instance.Error=TRUE) THEN
            Stage:=INT#10;                          // 异常结束
        END_IF;
```

```
    2 :                                          // 接收请求
        SktTCPRcv_instance(
            Execute  :=TRUE,
            Socket   :=WkSocket,                 // Socket
            TimeOut :=UINT#0,                    // 超时时间
            Size     :=UINT#2000,                // 接收数据大小
            RcvDat   :=RcvSocketDat[0]);         // 接收数据

        IF (SktTCPRcv_instance.Done=TRUE) THEN
            Stage:=INT#3;                        // 正常结束
        ELSIF (SktTCPRcv_instance.Error=TRUE) THEN
            Stage:=INT#20;                       // 异常结束
        END_IF;

    3 :                                          // 发送请求
        SendSocketDat:=RcvSocketDat;
        SktTCPSend_instance(
            Execute  :=TRUE,
            Socket   :=WkSocket,                 // Socket
            SendDat :=SendSocketDat[0],          // 发送数据
            Size     :=UINT#2000);               // 发送数据大小

        IF (SktTCPSend_instance.Done=TRUE) THEN
            Stage:=INT#4;                        // 正常结束
        ELSIF (SktTCPSend_instance.Error=TRUE) THEN
            Stage:=INT#30;                       // 异常结束
        END_IF;

     4 :                                         // 关闭请求
        SktClose_instance(
            Execute :=TRUE,
            Socket   :=WkSocket);                // Socket

        IF (SktClose_instance.Done=TRUE) THEN
            Stage:=INT#0;                        // 正常结束
        ELSIF (SktClose_instance.Error=TRUE) THEN
            Stage:=INT#40;                       // 异常结束
        END_IF;

    0 :                                          // 正常结束
        DoTCP :=FALSE;
        Trigger :=FALSE;

    ELSE                                         // 基于异常的中断
        DoTCP :=FALSE;
        Trigger :=FALSE;
    END_CASE;

END_IF;
```

SktTCPRcv

读取内置EtherNet/IP的指定TCP Socket的接收缓存中的数据。

指令	名称	FB/FUN	图形表现	ST表现
SktTCPRcv	TCP Socket接收	FB		SktTCPRcv_instance(Execute, Socket, TimeOut, Size, RcvDat, Done, Busy, Error, ErrorID, RcvSize);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–
TimeOut	超时时间		0 ：无超时 1 ~ 65535：0.1 ~ 6553.5s	遵从数据类型	0.1s	0
Size	保存容量		从接收缓存读取的数据大小	0 ~ 2000	字节	1
RcvDat[ ]数组	接收数据	输入输出	接收数据	遵从数据类型	–	–
RcvSize	接收数据大小	输出	实际保存到RcvDat[ ]中的数据大小	0 ~ 2000	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET 详情参阅功能说明																			
TimeOut							○													
Size							○													
RcvDat[ ]数组		○																		
RcvSize							○													

功能

将 “Socket”指定的Socket中接收缓存内的数据保存到接收数据RcvDat[ ]中。利用 “Size”指定要保存数据的大小。
然后，在保存数据后，将实际保存的数据大小代入 “RcvSize”。
如果接收缓存中无数据，则按照超时时间 “TimeOut”中设定的时间等待接收数据。
在指令正常结束(“Done”的值变为TRUE)时，将数据保存到RcvDat[ ]中。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 🕮 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

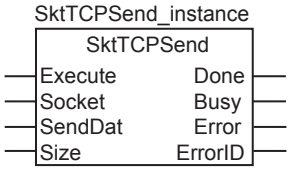
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 一次可读取的数据容量最大为2000字节。RcvDat[ ]的数组容量高于2000字节时，最多也只能读取2000字节。
- 已接收指定Socket的数据容量低于 “Size”的值时，仅将已接收数据保存在RcvDat[ ]中。然后，将已保存数据容量代入 “RcvSize”。
- 已接收指定Socket的数据容量高于 “Size”的值时，仅将 “Size”量的数据保存在RcvDat[ ]中。
- “Size”的值为0时，不读取接收数据。
- 接收缓存中无数据时，如果利用SktClose指令关闭Socket，则即使低于超时时间，也会异常结束。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “Socket”的任一结构要素超过有效范围时。
  - “Socket”指定的Socket在接收处理中时。
  - 未建立 “Socket”指定的Socket的连接时。
  - “Socket.Handle”指定的句柄不存在时。
  - 即使等待 “TimeOut”指定时间，仍未接收数据时。
  - 利用SktClose指令关闭Socket时。

示例程序

请参阅 “SktTCPConnect指令(P.2–1061)”的示例程序。

SktTCPSend

从内置EtherNet/IP的指定TCP端口发送数据。

指令	名称	FB/FUN	图形表现	ST表现
SktTCPSend	TCP Socket发送	FB		SktTCPSend_instance(Execute, Socket, SendDat, Size, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–
SendDat[ ]数组	发送数据		发送数据	遵从数据类型		
Size	发送数据大小		发送数据大小	0 ~ 2000	字节	1

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET 详情参阅功能说明																			
SendDat[ ]数组		○																		
Size							○													

功能

从 “Socket”指定的Socket发送发送数据SendDat[ ]。
利用 “Size”指定要发送的数据容量。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 🕮“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

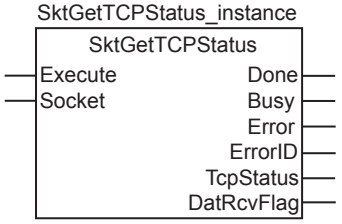
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 一次可发送的数据容量最大为2000字节。SendDat[ ]的数组容量高于2000字节时，最多也只能发送2000字节。
- “Size”的值为0时，不执行数据发送。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “Socket”的任一结构要素超过有效范围时。
  - “Socket”指定的Socket在发送处理中时。
  - 未建立 “Socket”指定的Socket的连接时。
  - “Socket.Handle”指定的句柄不存在时。

示例程序

请参阅 “SktTCPConnect指令(P.2-1061)”的示例程序。

SktGetTCPStatus

读取TCP Socket的状态。

指令	名称	FB/FUN	图形表现	ST表现
SktGetTCP Status	TCP Socket的状态读取	FB		SktGetTCPStatus_instance(Execute, Socket, Done, Busy, Error, ErrorID, TcpStatus, DatRcvFlag);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–
TcpStatus	TCP连接状态	输出	TCP连接状态	(*)	–	–
DatRcv Flag	接收数据有无标志		TRUE ：有接收数据 FALSE：无接收数据	遵从数据类型		

* _CLOSED, _LISTEN, _SYN_SENT, _SYN_RECEIVED, _ESTABLISHED, _CLOSE_WAIT, _FIN_WAIT1, _CLOSING, _LAST_ACK, _FIN_WAIT2, _TIME_WAIT

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET　详情参阅功能说明																			
TcpStatus	枚举体_eCONNECTION_STATE　枚举元素参阅功能说明																			
DatRcv Flag	○																			

功能

获取 “Socket”指定Socket的TCP连接状态 “TcpStatus”。
此外，接收缓存中存在接收数据时，将接收数据有无标志 “DatRcvFlag”的值设为TRUE。
在指令正常结束(“Done”的值变为TRUE)时，将数据保存到 “TcpStatus”和 “DatRcvFlag”中。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	数据收发的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

“TcpStatus”的数据类型为枚举体_eCONNECTION_STATE。
枚举元素显示TCP的各种状态。各含义如下所示。

枚举元素	TCP的状态	含义
_CLOSED	CLOSED	连接已关闭。
_LISTEN	LISTEN	服务器侧被动打开，等待连接建立请求(SYN)。
_SYN_SENT	SYN SENT	客户端侧主动打开，发送连接建立请求(SYN)后，等待相应的确认响应(SYN+ACK)。
_SYN_RECEIVED	SYN RECEIVED	服务器侧对连接建立请求(SYN) 发送确认响应(SYN+ACK)后，等待相应的确认响应(ACK) 。
_ESTABLISHED	ESTABLISHED	连接建立。
_CLOSE_WAIT	CLOSE WAIT	服务器侧对连接切断请求(FIN) 发送确认响应(ACK)后，等待至可切断服务器的应用程序。
_FIN_WAIT1	FIN WAIT1	客户端侧发送连接切断请求(FIN)后，等待相应的确认响应(ACK)。
_CLOSING	CLOSING	客户端侧和服务器侧同时接收连接切断请求(FIN)后，等待相应的确认响应(ACK)。
_LAST_ACK	LAST ACK	服务器侧发送连接切断请求(FIN)后，等待相应的确认响应(ACK)。
_FIN_WAIT2	FIN WAIT2	客户端侧等待连接切断请求(FIN)。
_TIME_WAIT	TIME WAIT	客户端侧接收对连接切断请求(FIN)的确认响应(ACK)后，等待到达服务器后的处理。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”或“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”。

使用注意事项

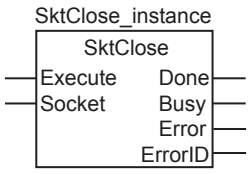
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Socket”的任一结构要素超过有效范围时。
  - “Socket.Handle”指定的句柄不存在时。

示例程序

请参阅 “SktTCPConnect指令(P.2-1061)”的示例程序。

SktClose

闭合内置EtherNet/IP的指定TCP Socket或UDP Socket。

指令	名称	FB/FUN	图形表现	ST表现
SktClose	TCP/UDPSocket闭合	FB		SktClose_instance(Execute, Socket, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET　详情参阅功能说明																			

功能

关闭 “Socket” 指定的Socket。
指定TCP Socket时，执行连接切断后将其关闭。
Socket的句柄 “Soceket.Handle” 的值为0时，利用Socket服务功能关闭正在使用的所有TCP/UDP端口。
在指令正常结束(“Done” 的值变为TRUE)时，关闭TCPUDPSocket。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	要关闭连接的句柄 0：利用Socket服务功能关闭正在使用的所有TCP/UDP连接。	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 🕮“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或🕮“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

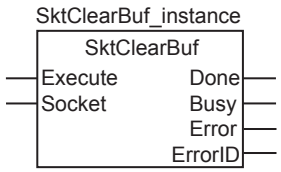
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- 如果在执行SktUDPRcv指令或SktTCPRcv指令，正在通过指定句柄的Socket等待接收时执行了本指令，则取消接收等待。
- 以一个本机端口编号开通多个连接时，仅关闭指定Socket的连接。
- Socket的句柄 “Soceket.Handle”的值为0时，基于SktTCPAccept指令的连接等待也会取消。
- SktUDPCreate 指令、SktUDPRcv 指令、SktUDPSend 指令、SktTCPAccept 指令、SktTCPConnect 指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 发生本机IP地址的设定异常时。
  - “Socket”的任一结构要素超过有效范围时。
  - “Socket.Handle”指定的句柄不存在时。

示例程序

请参阅 “SktUDPCreate指令(P.2-1045)”、或 “SktTCPConnect指令(P.2-1061)”的示例程序。

SktClearBuf

清除内置EtherNet/IP的指定TCP Socket或UDP Socket的接收缓存。

指令	名称	FB/FUN	图形表现	ST表现
SktClearBuf	TCP/UDP Socket 接收缓存清除	FB		SktClearBuf_instance(Execute, Socket, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET　详情参阅功能说明																			

功能

清除 “Socket”指定Socket的接收缓存。
在指令正常结束(“Done”的值变为TRUE)时，完成接收缓存的清除。

“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	要清除接收缓存的Socket的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket服务功能的详情请参阅 “NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD-359)”或“NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD-369)”。

使用注意事项

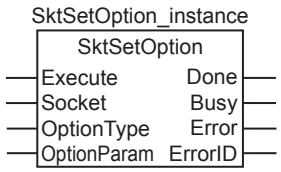
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅“本章说明(P.2-3)”。
- 仅在NJ/NX系列CPU单元的内置EtherNet/IP中可使用本指令。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Socket”的任一结构要素超过有效范围时。
  - “Socket”指定的Socket不存在时。
  - “Socket.Handle”指定的句柄不存在时。

示例程序

请参阅“SktTCPConnect指令(P.2-1061)”的示例程序。

SktSetOption

设定内置EtherNet/IP的指定TCP Socket的Socket选项。

指令	名称	FB/FUN	图形表现	ST表现
SktSetOption	TCP Socket选项设定	FB		SktSetOption_instance(Execute, Socket, OptionType, OptionParam, Done, Busy, Error, ErrorID);

使用注意事项

本指令无法在NX1P2 CPU单元Ver.1.13中使用。

版本相关信息

本指令可用于CPU单元Ver.1.12以上且Sysmac Studio Ver.1.16以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Socket	Socket	输入	Socket	–	–	–
OptionType	选项类型		Socket选项的类型	–	–	–
OptionParam	选项参数		根据指定Socket选项的类型指定参数	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Socket	结构体_sSOCKET　详情参阅功能说明																			
OptionType	枚举体_eSKT_OPTION_TYPE　枚举元素参阅功能说明																			
OptionParam	○*1																			

*1 无法输入常数(文字)。请指定变量。

功能

指定 “Socket”指定的Socket的Socket选项。
指令正常结束后，“Done”变为TRUE。
在指令正常结束时，Socket选项的设定完成。
“Socket”的数据类型为结构体_sSOCKET。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Socket	Socket	Socket	_sSOCKET	–	–	–
Handle	句柄	要清除接收缓存的Socket的句柄	UDINT	遵从数据类型	–	0
SrcAdr(*)	本机地址	本机的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"
DstAdr(*)	对方地址	对方的IP地址、端口编号	_sSOCKET_ADDRESS	–	–	–
PortNo(*)	端口编号	端口编号	UINT	1 ~ 65535	–	0
IpAdr(*)	IP地址	IP地址或主机名称。 但是，采用主机名称时，必须设定DNS或Hosts。	STRING	遵从数据类型		"

* 无法利用本指令使用这些结构要素。

可指定的 “OptionType”的内容及指定 “OptionType”可使用的 “OptionParam”的数据类型如下所述。此外，未使用本指令时的默认动作为以下初始值时的动作。

OptionType		OptionParam		
枚举元素	含义	可使用的数据类型	值的含义	初始值
_TCP_NODELAY	指定TCP_NODELAY选项。 只可用于TCP Socket。	BOOL	TRUE*1 : TCP_NODELAY 选项有效 FALSE: TCP_NODELAY 选项无效	FALSE

*1 设为TRUE时，可将Nagle算法设为无效。数量较少时，也不会集中发送。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

Socket 服务功能的详情请参阅 🕮 “NJ/NX 系列 CPU 单元内置 EtherNet/IP 端口 用户手册 (SBCD–359)”或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP。
- 本指令可在使用SktTCPAccept指令、SktTCPConnect指令打开Socket的句柄之后，在使用SktTCPRcv指令、SktTCPSend指令、SktClearBuf指令开始收发数据前使用。开始收发数据后使用本指令时，会发生异常。
- 需按照 “Socket”指定的句柄设定Socket选项。设定的Socket选项在句柄打开期间有效。执行SktClose指令关闭句柄后，再次执行SktTCPAccept指令、SktTCPConnect指令重新打开句柄时，请执行本指令重新设定Socket选项。
- SktUDPCreate指令、SktUDPRcv指令、SktUDPSend指令、SktTCPAccept指令、SktTCPConnect指令、SktTCPRcv指令、SktTCPSend指令、SktGetTCPStatus指令、SktClose指令、SktClearBuf指令、SktSetOption指令可同时执行的数量最多为32。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Socket”的任一结构要素超过有效范围时。
  - “OptionParam”中指定了 “OptionType”不支持的数据类型时。
  - 指定句柄的Socket已开始收发信息时。
  - 指定了对UDP Socket执行TCP_NODELAY等指定句柄的Socket不支持的Socket类型时。
  - “Socket.Handle”指定的句柄不存在时。

示例程序

ST

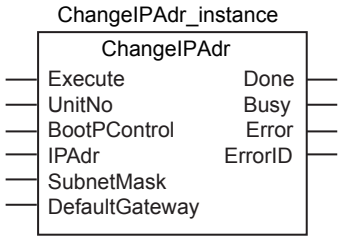
内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	DoTCP	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	WkSocket	_sSOCKET	(Handle:=0,SrcAdr:=(PortNo:=0,IpAdr:=''),DstAdr:=(PortNo:=0,IpAdr:=''))	Socket
	SendSocketDat	ARRAY[0..1999] OF BYTE		发送数据
	Nodelay	BOOL	TRUE	NoDelay设定
	SktTCPConnect_instance	SktTCPConnect		
	SktSetOption_instance	SktSetOption		
	SktTCPSend_instance	SktTCPSend		
	SktClose_instance	SktClose		

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (DoTCP=FALSE) AND (_EIP_EtnOnlineSta=TRUE) ) THEN
    DoTCP:=TRUE;
    Nodelay:=TRUE;
    Stage:=INT#1;
    SktTCPConnect_instance(Execute:=FALSE); // 实例初始化
    SktSetOption_instance( // 实例初始化
        Execute:=FALSE,
        OptionType:=_TCP_NODELAY,
        OptionParam:= Nodelay);
    SktTCPSend_instance( // 实例初始化
        Execute:=FALSE,
        SendDat:=SendSocketDat[0]); // 虚拟
    SktClose_instance(Execute:=FALSE); // 实例初始化
END_IF;

IF (DoTCP=TRUE) THEN
    CASE Stage OF
    1: // 连接请求
        SktTCPConnect_instance(
            Execute:=TRUE,
            SrcTcpPort:=UINT#0, // 本机UDP端口编号　自动分配
            DstAdr:=’192.168.250.2’, // 对方IP地址
            DstTcpPort:=UINT#6000, // 对方TCP端口编号
            Socket =>WkSocket); // Socket
        IF (SktTCPConnect_instance.Done=TRUE) THEN
            Stage:=INT#2; // 正常结束
        ELSIF (SktTCPConnect_instance.Error=TRUE) THEN
            Stage:=INT#10; // 异常结束
        END_IF;

    2: // Socket选项的设定
        SktSetOption_instance(
            Execute:=TRUE,
            Socket:=WkSocket, // Socket
            OptionType:=_TCP_NODELAY, // 选项类型
            OptionParam:= Nodelay); // NODELAY有效
        IF (SktSetOption_instance.Done=TRUE) THEN
            Stage:=INT#3; // 正常结束
        ELSIF (SktSetOption_instance.Error=TRUE) THEN
            Stage:=INT#20; // 异常结束
        END_IF;

    3: // 发送请求
        SktTCPSend_instance(
            Execute:=TRUE,
            Socket:=WkSocket, // Socket
            SendDat:=SendSocketDat[0], // 发送数据
            Size:=UINT#2000); // 发送数据大小
        IF (SktTCPSend_instance.Done=TRUE) THEN
            Stage:=INT#4; // 正常结束
        ELSIF (SktTCPSend_instance.Error=TRUE) THEN
            Stage:=INT#30; // 异常结束
        END_IF;

    4: // 关闭请求
        SktClose_instance(
            Execute:=TRUE,
            Socket:=WkSocket); // Socket
```

```
            IF (SktClose_instance.Done=TRUE) THEN
                Stage:=INT#0; // 正常结束
            ELSIF (SktClose_instance.Error=TRUE) THEN
                Stage:=INT#40; // 异常结束
            END_IF;

        0: // 正常结束
            DoTCP:=FALSE;
            Trigger:=FALSE;

        ELSE // 基于异常的中断
            DoTCP:=FALSE;
            Trigger:=FALSE;
    END_CASE;
END_IF;
```

ChangeIPAdr

变更内置EtherNet/IP端口或EtherNet/IP单元的IP地址。

指令	名称	FB/FUN	图形表现	ST表现
ChangeIPAdr	IP地址变更	FB		ChangeIPAdr_instance(Execute, UnitNo, BootPControl, IPAdr, SubnetMask, DefaultGateway, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitNo	单元编号	输入	要变更IP地址的单元编号	_CBU_CPU*1、_CBU_CPU_Port1*2、_CBU_CPU_Port2*3、_CBU_CPU_InPort1*4、_CBU_No00 ~ _CBU_No15*5	–	_CBU_No00
BootP Control	IP 地址获取方法+设定时间		IP 地址的获取方法和设定时间	0 ~ 3*6		0
IPAdr[ ] 数组*7	IP地址		IP地址	*8		–
Subnet Mask[ ] 数组*7	子网掩码		子网掩码			
Default Gateway[ ] 数组*7	默认网关		默认网关			

*1 使用NJ系列CPU单元时可指定。
*2 使用NX系列CPU单元的端口1或NY系列控制器时可指定。也可指定_CBU_CPU代替_CBU_CPU_Port1。
*3 使用NX系列CPU单元的端口2时可指定。
*4 使用NY系列控制器时可指定。
*5 使用NJ系列CPU单元时可指定。
*6 使用NX系列CPU单元的端口1、NJ系列CPU单元及NY系列控制器时为 “0 ~ 2”。使用NX系列CPU单元的端口2时为 “0 ~ 3”。使用NY系列控制器的内部通信端口时为 “0”。
*7 元素编号为0 ~ 3、元素数为4。
*8 利用单元编号 “UnitNo” 指定内置EtherNet/IP端口，还是指定EtherNet/IP单元，将决定不同的有效范围。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitNo	枚举体_eUnitNo 枚举元素参阅功能说明																			
BootP Control							○													
IPAdr[ ] 数组		○																		
	指定整个数组																			
Subnet Mask[ ] 数组		○																		
	指定整个数组																			
Default Gateway[ ] 数组		○																		
	指定整个数组																			

功能

根据IP地址获取方法+设定时间 “BootPControl”，变更由单元编号 “UnitNo”指定的、内置EtherNet/IP端口或EtherNet/IP单元的IP地址。
利用 “UnitNo”指定内置EtherNet/IP时，如果本指令执行完毕，则重启链接。打开链接时，能够以变更后的IP地址进行通信。
利用 “UnitNo”指定内部通信端口时，如果本指令执行完毕，则系统变量 “_EIPIn1_EtnOnlineSta”将变为FALSE。变为TRUE时，可通过变更后的IP地址进行通信。
利用 “UnitNo”指定EtherNet/IP单元后，如果本指令执行完毕，则EtherNet/IP单元重启。一旦重启完毕，则能够以变更后的IP地址进行通信。

使用本指令，可以通过显示器等变更内置EtherNet/IP端口或EtherNet/IP单元的IP地址。

“UnitNo”的数据类型为枚举体_eUnitNo。枚举元素的含义如下所示。

枚举元素	含义
_CBU_CPU*1	内置EtherNet/IP端口
_CBU_CPU_Port1*2	内置EtherNet/IP的通信端口1
_CBU_CPU_Port2*3	内置EtherNet/IP的通信端口2
_CBU_CPU_InPort1*4	内部通信端口1
_CBU_No00 ~ _CBUNo15*5	EtherNet/IP单元的单元编号00 ~ 15

*1 使用NJ系列CPU单元时可指定。
*2 使用NX系列CPU单元或NY系列控制器时可指定。
也可指定_CBU_CPU代替_CBU_CPU_Port1。
*3 使用NX系列CPU单元时可指定。无通信端口2的CPU单元无法使用。
*4 使用NY系列控制器时可指定。
*5 使用NJ系列CPU单元时可指定。

根据 “BootPControl”的值的不同，新IP地址的获取方法和设定时间也不同，如下所示。
使用NX系列CPU单元的端口1、NJ系列CPU单元及NY系列控制器时，“BootPControl”的值可指定 “0 ~ 2”。使用NX系列CPU单元的端口2时，可指定 “0 ~ 3”。使用NY系列控制器的内部通信端口时，只可指定 “0”。

“BootPControl”的值	IP地址的获取方法	IP地址的设定时间
0	获取IP地址IPAdr[ ]、子网掩码SubnetMask[ ]、默认网关DefaultGateway[ ]的值表示的IP地址	执行本指令时仅设定1次(固定设定)
1	从BOOTP服务器获取IP地址	执行本指令时设定1次，然后每当开启控制器电源时设定
2	从BOOTP服务器获取IP地址	执行本指令时仅设定1次(固定设定)
3	设定成未使用端口。 将IP地址设定成未设定。	执行本指令时仅设定1次(固定设定)

在IPAdr[ ]、SubnetMask[ ]、DefaultGateway[ ]的元素编号0 ~ 3中，从高位依次设定IP地址、子网掩码、默认网关的值。例如，将 IP 地址变更为 130.58.17.32 时，IPAdr[0]=BYTE#16#82、IPAdr[1]=BYTE#16#3A、IPAdr[2]=BYTE#16#11、IPAdr[3]=BYTE#16#20。
利用 “UnitNo”指定内置EtherNet/IP端口，还是指定EtherNet/IP单元，IPAdr[ ]、SubnetMask[ ]、DefaultGateway[ ]的值的有效范围将表现为如下情况。这些值的有效范围仅 “BootPControl”的值为0时生效。

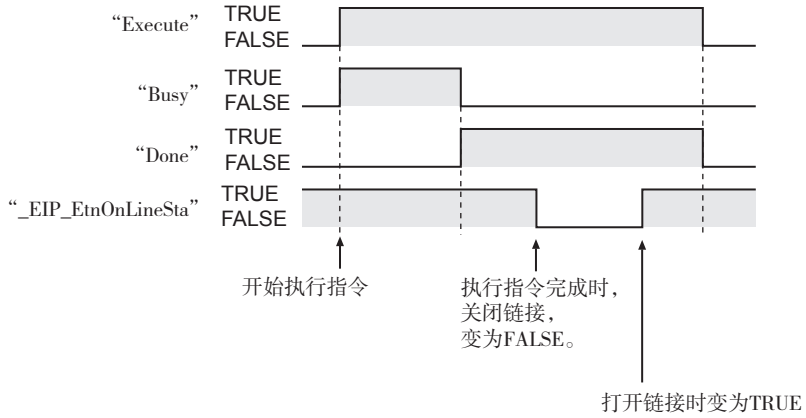
“UnitNo”的指定	输入变量	有效范围
内置EtherNet/IP端口 内部通信端口	IPAdr[ ]数组	非下列IP 地址有效 • 以127、0、255开始的IP 地址 • 主机ID 的所有位为0或者所有位为1的IP 地址 • D级的IP地址(224.0.0.0 ~ 239.255.255.255) • E级的IP地址(240.0.0.0 ~ 255.255.255.255) • 用于AutoIP*1的保留IP地址(169.254.0.0 ~ 169.254.255.255) • USB端口的IP地址(192.168.255.0 ~ 192.168.255.255)*2
	SubnetMask[ ]数组	192.0.0.0 ~ 255.255.255.252
	DefaultGateway[ ]数组	非下列IP 地址有效 • 以127、0、255开始的IP 地址 • 所有位为1的IP地址 • D级的IP地址(224.0.0.0 ~ 239.255.255.255) • E级的IP地址(240.0.0.0 ~ 255.255.255.255) • 用于AutoIP*1的保留IP地址(169.254.0.0 ~ 169.254.255.255) • USB端口的IP地址(192.168.255.0 ~ 192.168.255.255)*2
EtherNet/IP单元	IPAdr[ ]数组	非下列IP 地址有效 • 以127开始的IP地址 • D级的IP地址(224.0.0.0 ~ 239.255.255.255) • E级的IP地址(240.0.0.0 ~ 255.255.255.255)
	SubnetMask[ ]数组	• 0.0.0.0 • 192.0.0.0 ~ 255.255.255.252
	DefaultGateway[ ]数组	非下列IP 地址有效 • 以127开始的IP地址 • D级的IP地址(224.0.0.0 ~ 239.255.255.255) • E级的IP地址(240.0.0.0 ~ 255.255.255.255)

*1 AutoIP是指，Windows 98以上OS配备的IP地址自动分配功能。
*2 NX1P2 CPU单元及NY系列控制器无USB端口。

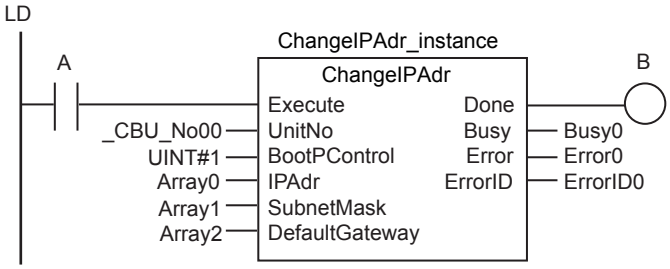
“BootPControl”的值为1或2时，忽略IPAdr[ ]、SubnetMask[ ]、DefaultGateway[ ]的值。IPAdr[ ]、SubnetMask[ ]、DefaultGateway[ ]的值可以超出有效范围。

利用“UnitNo”指定内置 EtherNet/IP 端口后，可以根据系统定义变量“_EIP_EtnOnlineSta”、“_EIP1_EtnOnlineSta”、“_EIP2_EtnOnlineSta”、“_EIPIn1_EtnOnlineSta”，确认链接是否开启。
这里以“_EIP_EtnOnlineSta”为例进行说明，“_EIP1_EtnOnlineSta”、“_EIP2_EtnOnlineSta”、“_EIPIn1_EtnOnlineSta”同理。

“Busy”从 TRUE 变为 FALSE 时，链接关闭，“_EIP_EtnOnlineSta”变为 FALSE。然后，打开链接，则“_EIP_EtnOnlineSta”从FALSE变为TRUE。



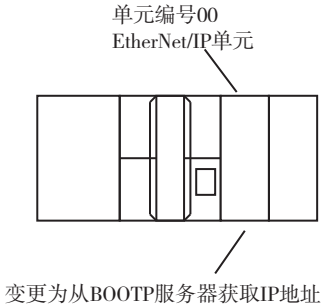
每次从BOOTP服务器获取单元编号00的EtherNet/IP单元的IP地址并变更时的描述示例如下所示。
通过显示器等将“Execute”=A设为ON后，变更为从BOOTP服务器获取的IP地址。之后，每当开启电源，均从BOOTP服务器获取IP地址并变更。



ST

```
ChangeIPAdr_instance(A,_CBU_No00,UINT#1,Array0,Array1,Array2,B,Busy0,Error0,ErrorID0);
```

为单元编号00的EtherNet/IP单元设定从BOOTP服务器获取的IP地址。
之后，每当开启电源，均重新设定从BOOTP服务器获取的IP地址。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	内置EtherNet/IP端口的通信功能状态 TRUE：可使用 FALSE：不可使用
_EIP1_EtnOnlineSta*2 _EIP2_EtnOnlineSta*3			
_EIPIn1_EtnOnlineSta*4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

- 利用 “UnitNo”指定内置EtherNet/IP后，如果执行本指令，则以下项目记录到事件日志中。
  - 链接关闭检测
  - IP地址确定
- 利用 “UnitNo”指定内部通信端口时，如果执行本指令，则以下项目记录到事件日志中。
  - IP地址确定
- CPU单元的写保护有效时，也可利用本指令变更IP地址。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 利用 “UnitNo”指定内置EtherNet/IP 时，如果本指令执行完毕，则内置EtherNet/IP 关闭链接。确认链接关闭后设备无障碍后，请执行本指令。
- 利用 “UnitNo”指定EtherNet/IP单元后，如果本指令执行完毕，则EtherNet/IP单元重启。确认重启后设备无障碍后，请执行本指令。
- 本指令无法在事件任务中使用。编译时会发生错误。
- 以下情况时会发生异常。“Error”变为TRUE。“ErrorID”的值及与其对应的异常内容如下所述。

“ErrorID”的值	异常名称	异常内容
16#0400	超过输入值范围	任一输入变量超过有效范围
16#2400	无执行权限	在无法变更的状态下执行了指令。 · 正在变更设定 · 正在重启内置EtherNet/IP端口 · 正在从NetworkConfigrator下载标签数据链接设定
16#2402	同时执行指令数超限	ChangeIPAdr指令、ChangeFTPAccount指令、ChangeNTPServerAdr指令的同时执行数超限。
16#240D	IP地址设定错误	指定端口与其他端口的网络地址相同。

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

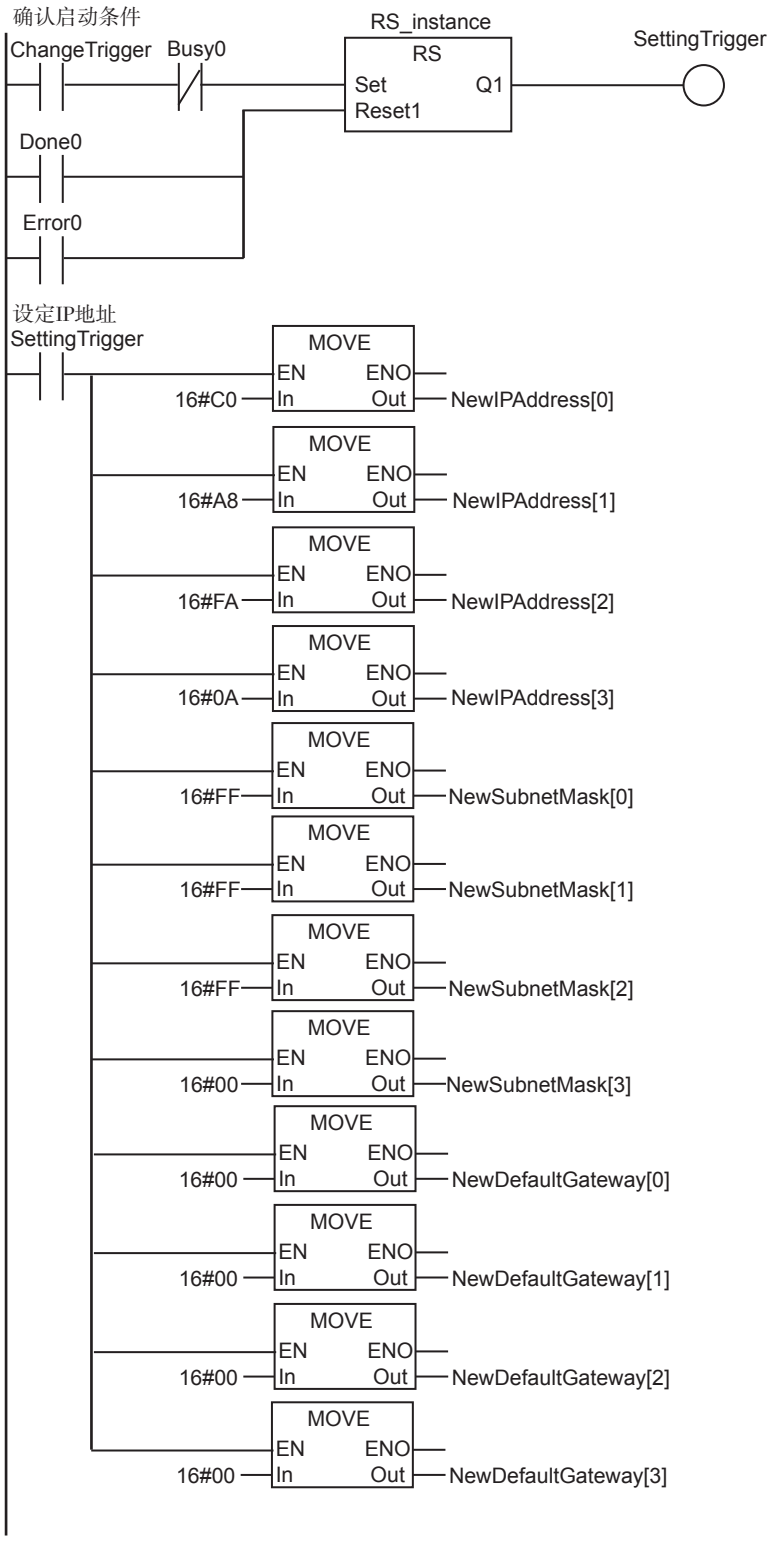
示例程序

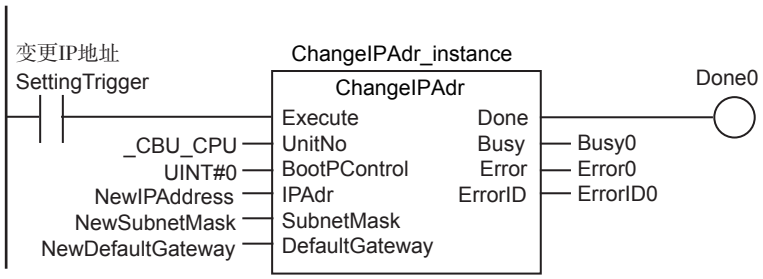
将内置EtherNet/IP端口的IP地址变更为以下IP地址的固定设定。

项目	内容
IP地址	192.168.250.10
子网掩码	255.255.255.0
默认网关	0.0.0.0

LD

名称	数据类型	初始值	注释
ChangeTrigger	BOOL	False	变更标志
SettingTrigger	BOOL	False	IP地址变更中标志
Done0	BOOL	False	IP地址变更完毕
Error0	BOOL	False	IP地址变更异常
Busy0	BOOL	False	IP地址变更中
ErrorID0	WORD	16#0	IP地址变更异常ID
NewIPAddress	ARRAY[0..3] OF BYTE	[4(16#0)]	IP地址
NewSubnetMask	ARRAY[0..3] OF BYTE	[4(16#0)]	子网掩码
NewDefaultGateway	ARRAY[0..3] OF BYTE	[4(16#0)]	默认网关
RS_instance	RS		
ChangeIPAdr_instance	ChangeIPAdr		


确认启动条件
ChangeTrigger
Busy0
RS_instance
RS
Set
Reset1
Q1
SettingTrigger
Done0
Error0
设定IP地址
SettingTrigger
MOVE
EN
ENO
16#C0
In
Out
NewIPAddress[0]
16#A8
NewIPAddress[1]
16#FA
NewIPAddress[2]
16#0A
NewIPAddress[3]
16#FF
NewSubnetMask[0]
NewSubnetMask[1]
NewSubnetMask[2]
16#00
NewSubnetMask[3]
NewDefaultGateway[0]
NewDefaultGateway[1]
NewDefaultGateway[2]
NewDefaultGateway[3]


变更IP地址
SettingTrigger
ChangeIPAdr_instance
ChangeIPAdr
Execute
Done
Done0
_CBU_CPU
UnitNo
Busy
Busy0
UINT#0
BootPControl
Error
Error0
NewIPAddress
IPAdr
ErrorID
ErrorID0
NewSubnetMask
SubnetMask
NewDefaultGateway
DefaultGateway

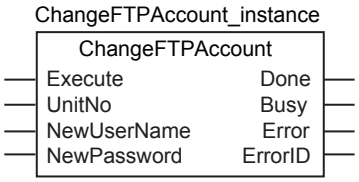
ST

名称	数据类型	初始值	注释
ChangeTrigger	BOOL	False	变更标志
SettingTrigger	BOOL	False	IP地址变更中标志
Done0	BOOL	False	IP地址变更完毕
Error0	BOOL	False	IP地址变更异常
Busy0	BOOL	False	IP地址变更中
ErrorID0	WORD	16#0	IP地址变更异常ID
NewIPAddress	ARRAY[0..3] OF BYTE	[4(16#0)]	IP地址
NewSubnetMask	ARRAY[0..3] OF BYTE	[4(16#0)]	子网掩码
NewDefaultGateway	ARRAY[0..3] OF BYTE	[4(16#0)]	默认网关
RS_instance	RS		
ChangeIPAdr_instance	ChangeIPAdr		

```
//确认启动条件
IF((ChangeTrigger=TRUE)AND(Busy0=FALSE))THEN
    SettingTrigger:= TRUE;
END_IF;

IF((Done0=TRUE)OR(Error0=TRUE))THEN
    SettingTrigger:= FALSE;
END_IF;

//设定IP地址
IF(SettingTrigger=TRUE)THEN
    NewIPAddress[0]      := 16#C0;
    NewIPAddress[1]      := 16#A8;
    NewIPAddress[2]      := 16#FA;
    NewIPAddress[3]      := 16#0A;
    NewSubnetMask[0]     := 16#FF;
    NewSubnetMask[1]     := 16#FF;
    NewSubnetMask[2]     := 16#FF;
    NewSubnetMask[3]     := 16#00;
    NewDefaultGateway[0]:= 16#00;
    NewDefaultGateway[1]:= 16#00;
    NewDefaultGateway[2]:= 16#00;
    NewDefaultGateway[3]:= 16#00;
END_IF;

//变更IP地址
ChangeIPAdr_instance(
            Execute        := SettingTrigger,
            UnitNo         := _CBU_CPU,
            BootPControl   := UINT#0,
            IPAdr          := NewIPAddress,
            SubnetMask     := NewSubnetMask,
            DefaultGateway := NewDefaultGateway,
            Done           =>Done0,
            Busy           =>Busy0,
            Error          =>Error0,
            ErrorID        =>ErrorID0);
```

ChangeFTPAccount

变更内置EtherNet/IP端口或EtherNet/IP单元的FTP登录名和密码。

指令	名称	FB/FUN	图形表现	ST表现
ChangeFTP Account	FTP账号变更	FB		ChangeFTPAccount_instance(Execute, UnitNo, NewUserName, NewPassword, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitNo	单元编号	输入	变更FTP登录名和密码的单元编号	_CBU_CPU、_CBU_No00 ~ _CBU_No15*1	–	_CBU_No00
New UserName	登录名		登录名	1 ~ 12个半角英数字字符(区分大小写)		–
NewPassword	密码		密码	*2		

*1 “_CBU_No00 ~ _CBU_No15”仅NJ系列CPU单元时可设定。

*2 利用单元编号 “UnitNo”指定内置EtherNet/IP端口，还是指定EtherNet/IP单元，将决定不同的有效范围。详情请参阅功能说明。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitNo	枚举体_eUnitNo 枚举元素参阅功能说明																			
New UserName																				○
NewPassword																				○

功能

对利用单元编号 “UnitNo”指定的内置EtherNet/IP端口或EtherNet/IP单元的FTP登录名 “NewUserName”和密码 “NewPassword”进行变更。
“Execute”从FALSE变为TRUE时，将 “NewUserName”、“NewPassword”的值作为内置EtherNet/IP端口的FTP登录名和密码进行写入。
执行过程中 “Busy”变为TRUE，设定变更请求受理完成时 “Done”变为TRUE。
“Done”变为TRUE时，设定尚未反映。
利用 “UnitNo”指定EtherNet/IP单元后，如果本指令执行完毕，则EtherNet/IP单元重启。重启完毕后，可使用变更后的登录名和密码。

使用本指令，可通过显示器等变更内置EtherNet/IP端口或EtherNet/IP单元的FTP登录名和密码。

“UnitNo”的数据类型为枚举体_eUnitNo。枚举元素的含义如下所示。

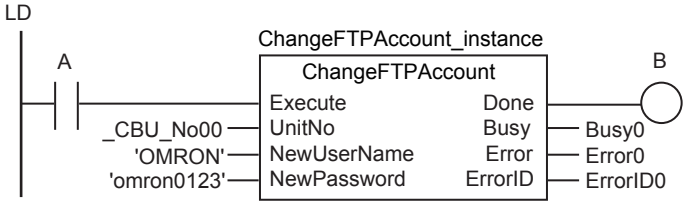
枚举元素	含义
_CBU_CPU	内置EtherNet/IP端口
_CBU_No00 ~ _CBU_No15*1	EtherNet/IP单元的单元编号00 ~ 15

*1 仅NJ系列CPU单元时可设定。

利用“UnitNo”指定内置EtherNet/IP端口，还是指定EtherNet/IP单元，“NewPassword”的值的有效范围将表现为如下情况。

“UnitNo”的指定	有效范围
内置EtherNet/IP端口	8 ~ 32个半角英数字字符(区分大小写)
EtherNet/IP单元	1 ~ 8个半角英数字字符(区分大小写)

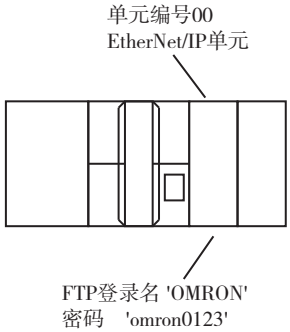
变更单元编号 00 的 EtherNet/IP 单元的 FTP 登录名和密码时的描述示例如下所示。通过显示器等将“Execute”=A设为ON，将FTP登录名变更为'OMRON'、密码变更为'omron0123'。

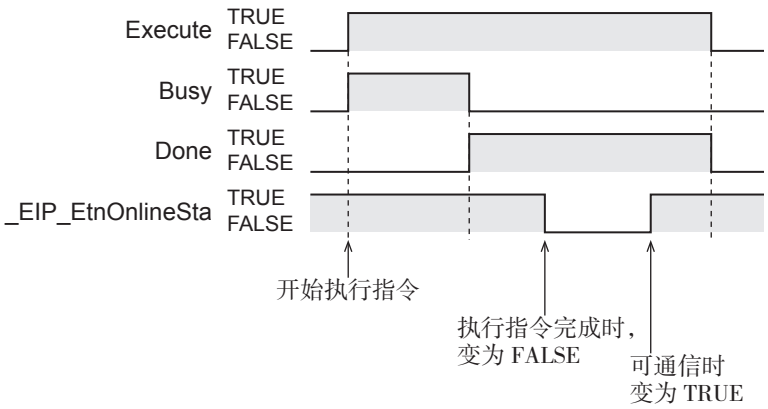


ST

```
ChangeFTPAccount_instance(A,_CBU_No00,'OMRON','omron0123',B,Busy0,Error0,ErrorID0);
```

将单元编号00的EtherNet/IP单元的FTP登录名变更为'OMRON'，
密码变更为'omron0123'。





相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

参考

- 使用本指令，即使CPU单元的写保护有效，也可变更FTP登录名和密码。
- 连接FTP服务器，传送文件时，即使利用本指令变更FTP登录名和密码，仍将继续传送文件。
- 在FTP登录的状态下，使用本指令变更了设定后，已建立的FTP会话仍将继续保持。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 本指令无法在事件任务中使用。编译时会发生错误。
- 以下情况时会发生异常。“Error”变为TRUE。“ErrorID”的值及与其对应的异常内容如下所述。

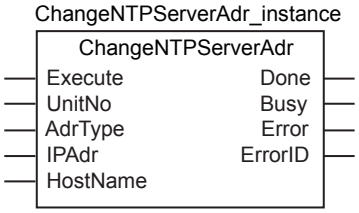
“ErrorID”的值	异常名称	异常内容
16#0400	超过输入值范围	· 任一输入变量超过有效范围 · 任一输入变量错误
16#2400	无执行权限	在无法变更的状态下执行了指令。 · 正在变更设定 · 正在重启内置EtherNet/IP端口 · 正在从NetworkConfigrator下载标签数据链接设定
16#2402	同时执行指令数超限	ChangeIPAdr指令、ChangeFTPAccount指令、ChangeNTPServerAdr指令的同时执行数超限。

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

ChangeNTPServerAdr

变更内置EtherNet/IP端口或EtherNet/IP单元的NTP服务器地址。

指令	名称	FB/FUN	图形表现	ST表现
ChangeNTP ServerAdr	NTP服务器地址变更	FB		ChangeNTPServerAdr_instance(Execute, UnitNo, AdrType, IPAdr, HostName, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
UnitNo	单元编号	输入	要变更NTP服务器地址的单元编号	_CBU_CPU、_CBU_No00 ~ _CBU_No15*1	–	_CBU_No00
AdrType	服务器设定方法		NTP服务器地址的设定方法 TRUE：IP地址 FALSE：主机名称	遵从数据类型		FALSE
IPAdr[ ]数组*2	IP地址		NTP服务器地址的IP地址	*3		–
HostName	主机名称		NTP服务器地址的主机名称	1 ~ 200个半角英数字字符和“–”(连字符)、“.”(点)*4		

*1 “_CBU_No00 ~ _CBU_No15”仅NJ系列CPU单元时可设定。

*2 元素编号为0 ~ 3、元素数为4。

*3 利用单元编号 “UnitNo”指定内置EtherNet/IP端口，还是指定EtherNet/IP单元，将决定不同的有效范围。详情请参阅功能说明。

*4 “.”(点)和 “.”(和)之间为1 ~ 63个半角英数字字符。“HostName”的有效范围仅在服务器设定方法 “AdrType”为FALSE时有效。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
UnitNo	枚举体_eUnitNo 枚举元素参阅功能说明																			
AdrType	○																			
IPAdr[ ]数组		○																		
	指定整个数组																			
HostName																				○

功能

对利用单元编号 “UnitNo”指定的内置EtherNet/IP端口或EtherNet/IP单元的NTP服务器的地址进行变更。服务器设定方法“AdrType”为TRUE时，将NTP服务器的地址变更为IP地址IPAdr[ ]。“AdrType”为FALSE时，将NTP服务器的地址变更为主机名称 “HostName”。
“Execute”从FALSE变为TRUE时，将 “NewUserName”、“NewPassword”的值作为内置EtherNet/IP端口的FTP登录名和密码进行写入。
执行过程中 “Busy”变为TRUE，设定变更请求受理完成时 “Done”变为TRUE。
“Done”变为TRUE时，设定尚未反映。
利用 “UnitNo”指定EtherNet/IP单元后，如果本指令执行完毕，则EtherNet/IP单元重启。重启完毕后，可使用已变更的NTP服务器的地址。

使用本指令，可通过显示器等变更内置EtherNet/IP端口或EtherNet/IP单元的NTP服务器的地址。

“UnitNo”的数据类型为枚举体_eUnitNo。枚举元素的含义如下所示。

枚举元素	含义
_CBU_CPU	内置EtherNet/IP端口
_CBU_No00 ~ _CBU_No15*1	EtherNet/IP单元的单元编号00 ~ 15

*1 仅NJ系列CPU单元时可设定。

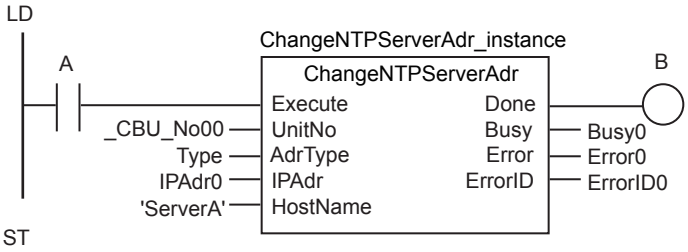
在IPAdr[ ]的元素编号0 ~ 3中，从高位依次设定IP地址的值。例如，将NTP服务器的地址变更为IP地址130.58.17.32时，IPAdr[0]=BYTE#16#82、IPAdr[1]=BYTE#16#3A、IPAdr[2]=BYTE#16#11、IPAdr[3]=BYTE#16#20。

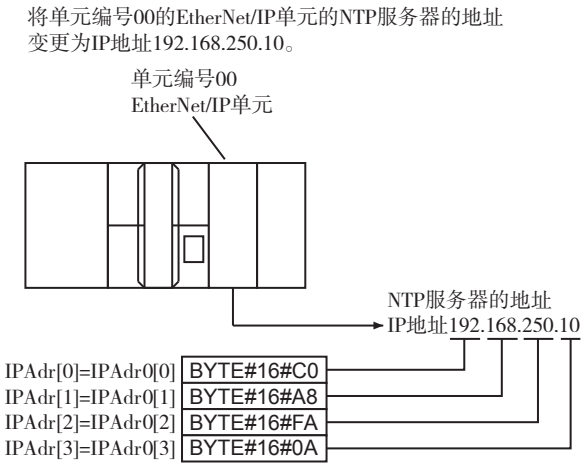
利用“UnitNo”指定内置EtherNet/IP端口，还是指定EtherNet/IP单元，IPAdr[ ]的值的有效范围将表现为如下情况。这些值的有效范围仅 “AdrType”的值为TRUE时生效。

“UnitNo”的指定	有效范围
内置EtherNet/IP端口	非下列IP 地址有效 • 以127、0、255开始的IP 地址 • D级的IP地址(224.0.0.0 ~ 239.255.255.255) • E级的IP地址(240.0.0.0 ~ 255.255.255.255) • 用于AutoIP*1的保留IP地址(169.254.0.0 ~ 169.254.255.255) • USB端口的IP地址(192.168.255.0 ~ 192.168.255.255)
EtherNet/IP单元	非下列IP 地址有效 • 以127开始的IP地址 • D级的IP地址(224.0.0.0 ~ 239.255.255.255) • E级的IP地址(240.0.0.0 ~ 255.255.255.255)

*1 AutoIP是指，Windows 98以上OS配备的IP地址自动分配功能。

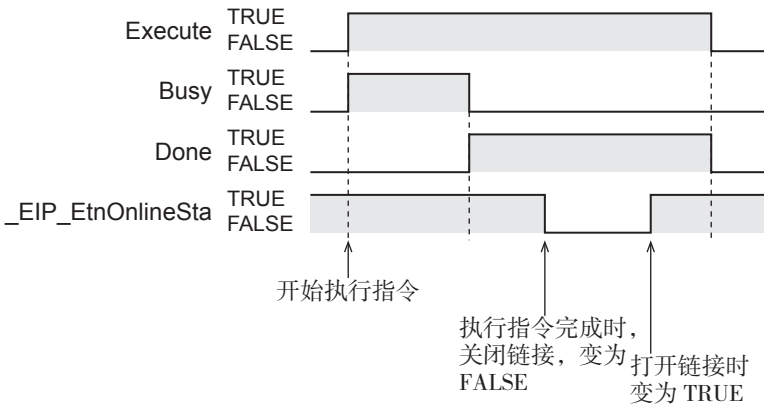
变更单元编号00的EtherNet/IP单元的NTP服务器地址时的描述示例如下所示。通过显示器等将“Execute”=A设为ON，变更NTP服务器地址。
例如，IPAdr0[0]=BYTE#16#C0、IPAdr0[1]=BYTE#16#A8、IPAdr0[2]=BYTE#16#FA、IPAdr0[3]=BYTE#16#0A。“AdrType”=Type为TRUE时，将NTP服务器的地址变更为IP地址192.168.250.10。“AdrType”=Type为FALSE时，将NTP服务器的地址变更为主机名称ServerA。





可根据系统定义变量 “_EIP_EtnOnlineSta”、“_EIP1_EtnOnlineSta”、“_EIP2_EtnOnlineSta”确认链接是否开启。
这里以 “_EIP_EtnOnlineSta”为例进行说明，“_EIP1_EtnOnlineSta”、“_EIP2_EtnOnlineSta”同理。

“Busy”从 TRUE 变为 FALSE 时，链接关闭，“_EIP_EtnOnlineSta”变为 FALSE 。然后，打开链接，则“_EIP_EtnOnlineSta”从FALSE 变为TRUE 。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta*1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta*2			
_EIP2_EtnOnlineSta*3			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。

参考

- 使用本指令，即使CPU单元的写保护有效，也可变更NTP服务器的地址。
- 要变更的NTP服务器的NTP动作时间为指定时间间隔时，从本指令执行完毕之时对指定时间间隔进行计时。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 本指令无法在事件任务中使用。编译时会发生错误。
- 以下情况时会发生异常。“Error”变为TRUE。“ErrorID”的值及与其对应的异常内容如下所述。

“ErrorID”的值	异常名称	异常内容
16#0400	超过输入值范围	“IPAdr[] 数组”或 “HostName”的值错误*1
16#2400	无执行权限	在无法变更的状态下执行了指令。 · 正在变更设定 · 正在重启内置EtherNet/IP端口 · 正在从NetworkConfigrator下载标签数据链接设定
16#2402	同时执行指令数超限	ChangeIPAdr指令、ChangeFTPAccount指令、ChangeNTPServerAdr指令的同时执行数超限。

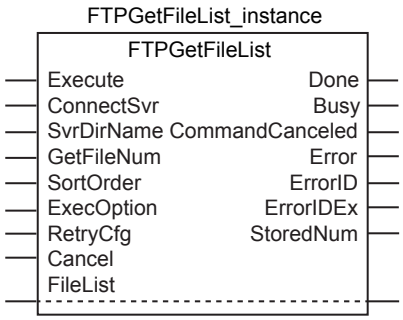
*1 只检查与 “AdrType”的指定相符的设定范围。

版本相关信息

本指令可用于Ver.1.02以上的CPU单元和Ver.1.03以上的Sysmac Studio。

FTPGetFileList

获取FTP服务器内的文件列表。

指令	名称	FB/FUN	图形表现	ST表现
FTPGetFileList	FTP服务器的文件列表获取	FB		FTPGetFileList_instance(Execute, ConnectSvr, SvrDirName, GetFileNum, SortOrder, ExecOption, RetryCfg, Cancel, FileList, Done, Busy, CommandCanceled, Error, ErrorID, ErrorIDEx, StoredNum);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	输入	连接目标FTP服务器的各种设定参数	–	–	*1
SvrDirName	FTP服务器目录名称		获取文件列表的FTP服务器目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)*2		''*3
GetFileNum	获取文件数		待获取列表的文件数	1 ~ 1000		1
SortOrder*4	排列顺序		文件列表的排列顺序	_NAME_ASC, _NAME_DESC, _DATE_ASC, _DATE_DESC		_NAME_ASC
ExecOption	FTP执行选项		FTP执行的相关选项	–		–
RetryCfg	执行重试设定		指令执行重试设定			
Cancel	取消		TRUE：取消指令执行 FALSE：不取消指令执行	遵从数据类型		FALSE
FileList[]数组*5*6*7	文件详情	输入输出	获取文件的详细信息	–	–	*1
Command Canceled	取消完成	输出	TRUE：取消完成 FALSE：取消未完成	遵从数据类型	–	–
StoredNum	已获取文件数		获取详细信息的文件数	0 ~ 1000		

*1 省略输入参数时，初始值不适用。编连时会发生异常。
2 FTP服务器目录名称，不能使用下列字符。“”、“?”、“<”、“>”、“|”、“"”(双引号)
*3 FTP服务器的登录时的主目录。
*4 FTP服务器不支持排序时，按名称的升序排序，与 “SortOrder”的值无关。
*5 数组的最大元素数为1000。
*6 1维数组。指定2维以上数组的情况下，编连时会发生异常。

*7　开头的元素编号为0。指定开头元素编号非0的数组的情况下，编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
ConnectSvr	结构体_sFTP_CONNECT_SVR　详情参阅功能说明																			
SvrDirName																				○
GetFileNum							○													
SortOrder	枚举体_eFILE_SORT_ORDER　枚举元素参阅功能说明																			
ExecOption	结构体_sFTP_EXEC_OPTION　详情参阅功能说明																			
RetryCfg	结构体_sFTP_RETRY_CFG　详情参阅功能说明																			
Cancel	○																			
FileList[]数组	结构体_sFTP_FILE_DETAIL　详情参阅功能说明																			
Command Canceled	○																			
StoredNum							○													

功能

获取各文件的详细信息和连接目标FTP服务器 “ConnectSvr”指定目录 “SvrDirName”中的多个文件的列表。
获取列表的文件数量由获取文件数“GetFileNum”指定。获取的文件信息的排序由排列顺序 “SortOrder”指定。

“ConnectSvr”的数据类型为结构体_sFTP_CONNECT_SVR。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	连接目标FTP服务器的各种设定参数	_sFTP_CONNECT_SVR	–	–	–
Adr	地址	IP地址或主机名称*1	STRING	1 ~ 200字节*2	–	–
PortNo	端口编号	FTP服务器的控制连接的TCP端口编号	UINT	0 ~ 65535*3		
UserName	用户名	FTP服务器的用户名	STRING	最大33字节 *2*4*5		
Password	密码	FTP服务器的密码	STRING	最大33字节 *2*4*5		

*1 指定主机名称时，必须另行设定DNS或Hosts。
*2 可使用的字符为半角的 “A–Z”、“a–z”、“0–9”、“–”(连字符)、“.”(句号)、“_”(下划线)。
*3 为0时，TCP端口编号为21。
*4 字节数含结尾NULL字符。
*5 CPU单元Ver.1.08请指定至少1个字符的字符串。指定仅结尾NULL字符的字符串时，将发生异常。

“SortOrder”的数据类型为枚举体_eFILE_SORT_ORDER。枚举元素的含义如下所示。

枚举元素	含义
_NAME_ASC	名称升序
_NAME_DESC	名称降序
_DATE_ASC	更新日期时间升序
_DATE_DESC	更新日期时间降序

各文件的详细信息将保存在文件详情FileList[]中。获取的文件数将保存在已获取文件数“StoredNum”中。

FileList[]的数据类型为结构体_sFTP_FILE_DETAIL。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
FileList	文件详情	获取文件的详细信息	_sFTP_FILE_DETAIL	–	–	–
Name	文件夹名称、文件名	文件夹名称或文件名	STRING	最大256字节 (255个半角英数字字符+结尾NULL字符)	–	–
ModifiedDate	更新日期时间	文件夹或文件的更新日期时间	DATE_AND_TIME	–		
Size	文件大小	文件的大小*1	ULINT		字节	
ReadOnly	属性–只读	文件夹和文件的只读属性TRUE：只读 FALSE：非只读	BOOL	遵从数据类型	–	
Folder	文件夹	TRUE：文件夹 FALSE：非文件夹	BOOL			

*1 文件夹时保存0。

与FTP服务器的处理相关的选项指定

从FTP服务器获取文件列表时，进行FTP执行选项 “ExecOption”指定的动作。
选项设定内容与 “FTPGetFile指令(P.2–1120)” 的规格相同。请参阅该处。
但，本指令有效的选项仅被动模式指定 “ExecOption.PassiveMode”。

与FTP服务器之间连接处理的重试指定

与FTP服务器之间的连接处理，在指定的超时时间 “RetryCfg.TimeOut”内未成功时，则认定超时而结束。
FTP服务器拒绝连接时，不等到超时时间直接结束。
连接失败后，经过指定的重试间隔 “RetryInterval”秒后，重试连接处理。按指定重试次数 “RetryCfg.RetryNum”进行重试，未能与FTP服务器连接时，指令将发生执行错误。

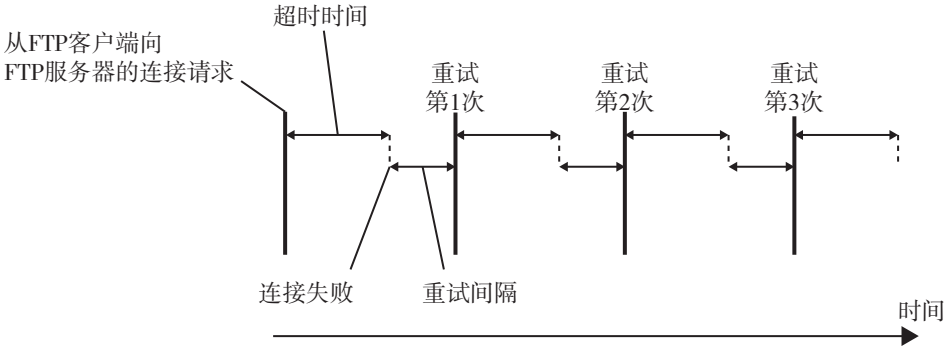
与FTP服务器之间的连接处理成功后，因网络障碍等原因而中断文件传输时，按 “RetryCfg.TimeOut”指定的时间超时处理，不进行重试。

“RetryCfg”的数据类型为结构体_sFTP_RETRY_CFG。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
RetryCfg	执行重试设定	指令执行重试设定	_sFTP_RETRY_CFG	–	–	–
TimeOut	超时时间	与FTP服务器的连接超时时间	UINT	0 ~ 60*1	秒	20
RetryNum	重试次数	连接失败时的重试次数	UINT	0 ~ 3	次	0
RetryInterval	重试间隔	连接失败时的重试间隔	UINT	0 ~ 65535*2	秒	1

*1 设定0时，超时时间为20秒。
*2 设定0时，重试间隔为1秒。

从FTP客户端连接到FTP服务器时的超时时间、重试次数、重试间隔之间的关系如下所示。

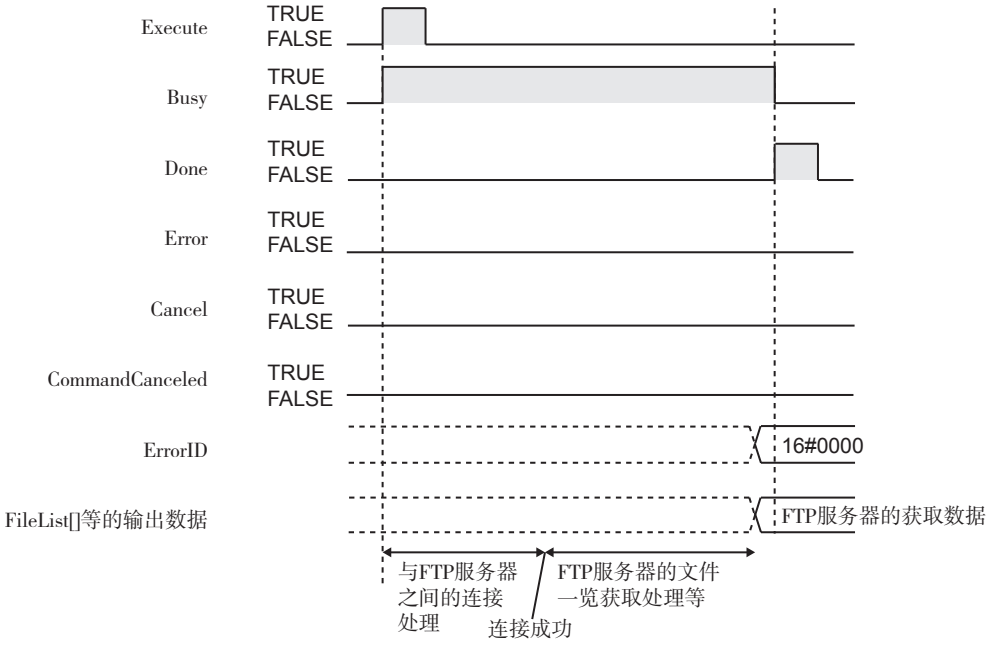


● 与FTP服务器连接成功时

与FTP服务器连接成功时，从FTP服务器获取文件列表后，进行下列处理。

- 16#0000保存在 “ErrorID”中。
- 获取数据保存在FileList[]等的输出数据中。
- 正常结束 “Done”的值变为TRUE。

与FTP服务器连接成功时的时序图的示例如下所示。

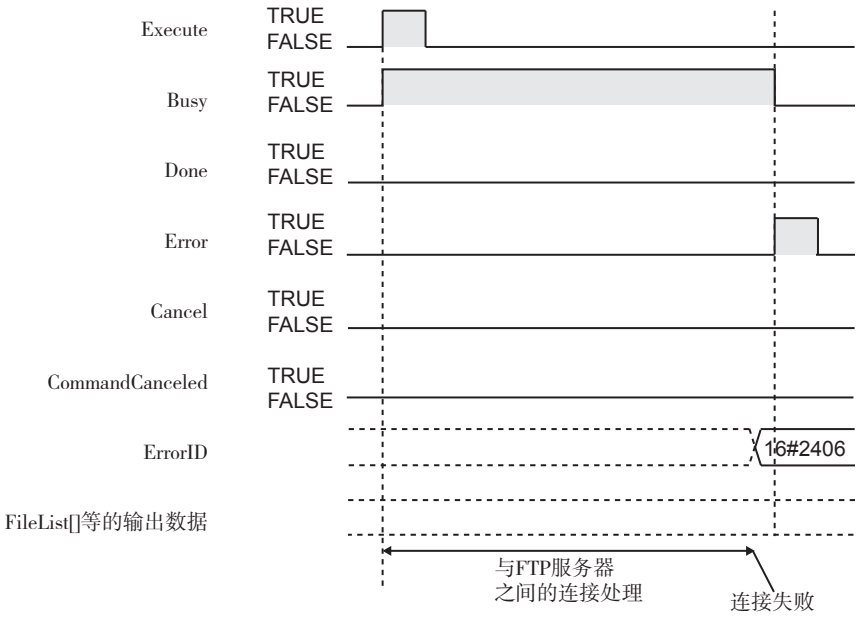


● 与FTP服务器连接失败时

与FTP服务器连接失败时，进行下列处理。

- 错误代码保存在 “ErrorID”中。
- 异常结束 “Error”的值变为TRUE。

与FTP服务器的连接处理异常结束时的时序图的示例如下所示。



取消指令执行

指令执行中，将取消 “Cancel”设为TRUE，强制结束与FTP服务器的处理。在获取文件列表或与FTP服务器连接处理需要时间，要结束处理时使用。

● 执行与FTP服务器的处理的过程中，将取消 “Cancel”设为TRUE时

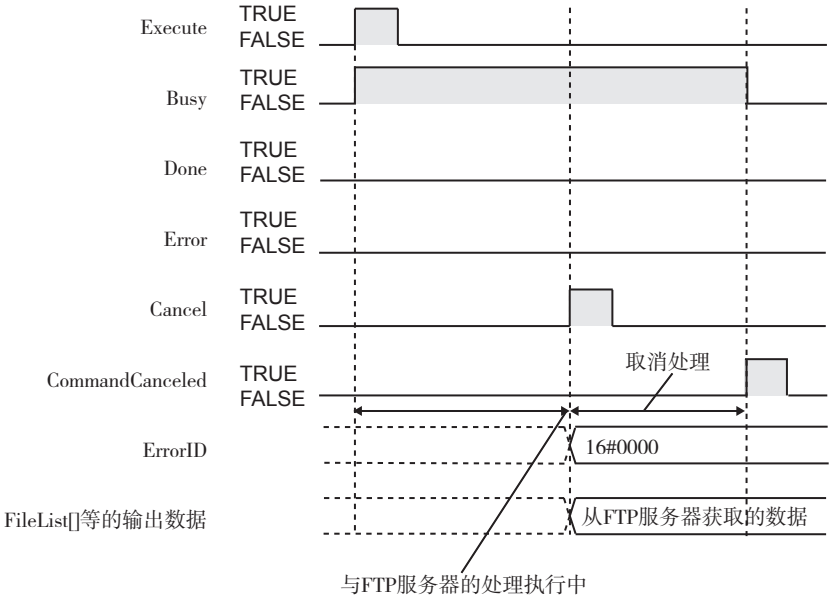
利用FTPGetFileList指令从FTP服务器获取文件列表的过程中，将取消 “Cancel”设为TRUE，将出现下列情况。

在文件详情FileList[]中仅保存已从FTP服务器获取的文件详细信息部分。

在获取文件数 “StoredNum”中保存已正常获取的文件详细信息的文件数量。

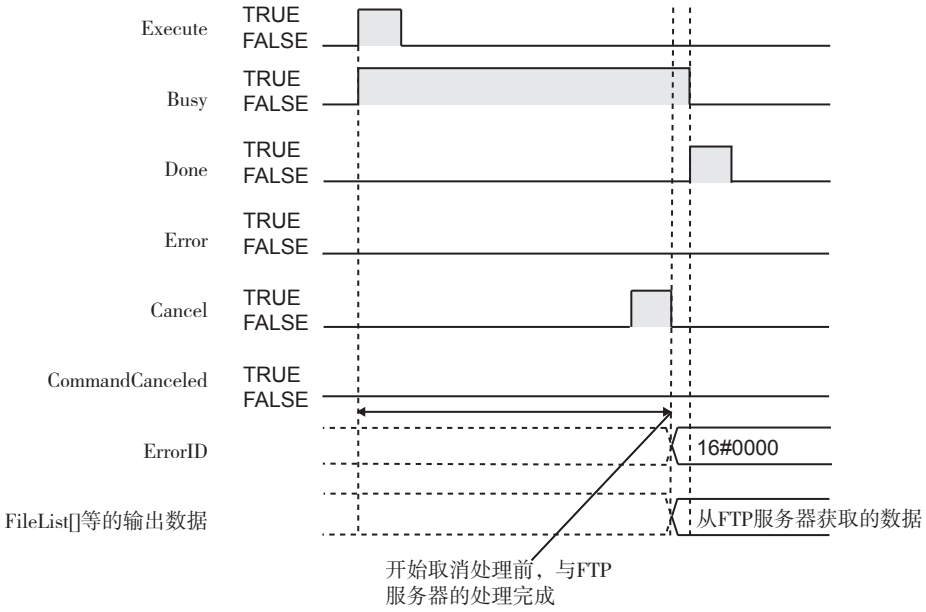
正常结束 “Done”不会变为TRUE。

请通过 “CommandCanceled”的值是否变为TRUE确认取消是否正常结束。



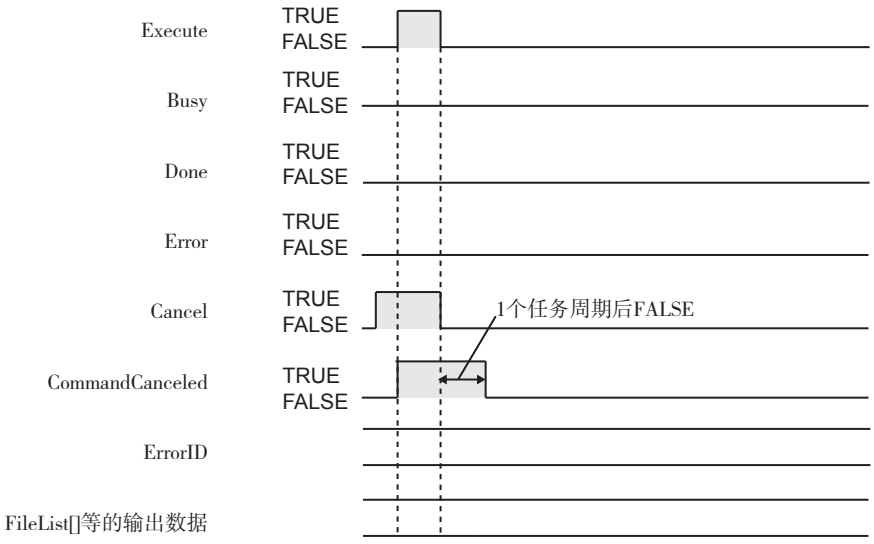
● 开始取消处理前，与FTP服务器的处理完成时

在开始取消处理前，与FTP服务器的处理已完成的情况下，即使取消 “Cancel”变为TRUE，也会通知正常结束 “Done”。且，“CommandCanceled”的值不会变为TRUE。



● 将取消 “Cancel”与启动 “Execute”均设为TRUE时

取消 “Cancel”与启动 “Execute”均为TRUE时，优先取消，不进行与FTP服务器的处理。“CommandCanceled”变为TRUE。



相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

使用注意事项

- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 在输入变量 “SvrDirName”指定的目录中没有文件或子目录时，正常结束，“Done”变为TRUE。“StoredNum”中保存0，FileList[]中不保存任何内容。
- FileList[]的数组元素数比输入变量 “GetFileNum”指定的获取文件数少时，文件信息仅按FileList[]的数组元素数保存，不保存超出部分的信息。此时，“Error”不会变为TRUE。
- 关于超出 255 个字符的文件名，将从开头起的 255 个字符部分保存在 FileList[] 的 “Name”中。此时，“Error”不会变为TRUE。
- 根据FTP服务器规格的不同，可能无法获取指定的部分或全部文件详细信息。对于未能获取详细信息的文件，FileList[]的结构要素的值如下所示。此时，“Error”的值变为FALSE。

结构要素	值
ModifiedDate	DT#1970-01-01-00:00:00.000000000
Size	0
ReadOnly	FALSE
Folder	FALSE

- FTPGetFileList指令、FTPGetFile指令、FTPPutFile指令、FTPRemoveFile指令、FTPRemoveDir指令可同时执行的数量最多为3个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 任一输入参数的值超过有效范围时
  - 在 “SvrDirName”中指定上1层的目录 “..”时
  - 在 “SvrDirName”中指定 “//”等错误路径时
  - FTP服务器中不存在 “SvrDirName”指定的目录时
  - 网络上不存在 “ConnectSvr”指定的FTP服务器，或者服务已停止时
  - 同时执行的 FTPGetFileList 指令、FTPGetFile 指令、FTPPutFile 指令、FTPRemoveFile 指令、FTPRemoveDir指令超过3个时。
  - 与FTP服务器之间连接处理的过程中，因网络障碍等原因而中断文件传输处理时

- 扩展错误代码 “ErrorIDEx” 在本指令中表示FTP服务器回复的FTP响应代码。典型的 “ErrorIDEx” 的值、异常内容和处理方法如下所述。详情请确认FTP服务器的规格。“ErrorIDEx” 在错误代码 “ErrorID” 的值为WORD#16#2407时输出。

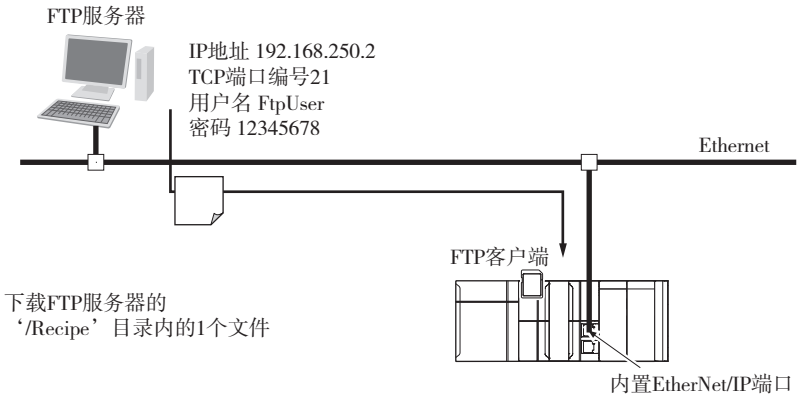
“ErrorIDEx” 的值	异常内容	处理方法
16#000001A9	未能建立数据连接。	与互联网上的FTP服务器进行FTP通信时，请确认FTP的打开模式是否为Active。
16#000001AA	连接已关闭。停止了数据传输。	请确认与FTP服务器的连接。 请确认FTP服务器是否停止。
16#000001C2	未进行请求的文件操作。文件已被打开等情况时，无法使用文件。	请确认对象文件是否已被其他应用程序打开。
16#00000212	用户未能登录。	请确认FTP用户名或密码。
16#00000214	需要文件保存用账号。	请确认FTP用户的访问权。
16#00000226	找不到文件而无法访问等情况时，无法使用文件，因此未执行请求的文件操作。	请确认FTP服务器的目录中是否存在与指定文件名相符的文件。 请确认指定文件的访问权。
16#00000229	文件名不正确，未能执行。	请确认指定目录的访问权。

版本相关信息

本指令可用于CPU单元 Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

下载FTP服务器的 ‘/Recipe’目录内的1个文件，保存在SD存储卡的根目录中。
要下载的文件是，按名称升序排列的FTP服务器的 ‘/Recipe’目录内文件中的最后1个文件。



控制器与FTP服务器之间通过EtherNet/IP网络连接。为与FTP服务器连接而设定的参数值如下所示。

设定参数	值
IP地址	192.168.250.2
TCP端口编号	21
用户名	FtpUser
密码	12345678

处理步骤如下所述。

1　使用FTPGetFileList指令，获取FTP服务器的文件列表。保存FTP服务器的目录名称、获取文件数、排列顺序以及文件详细信息的变量如下所示。

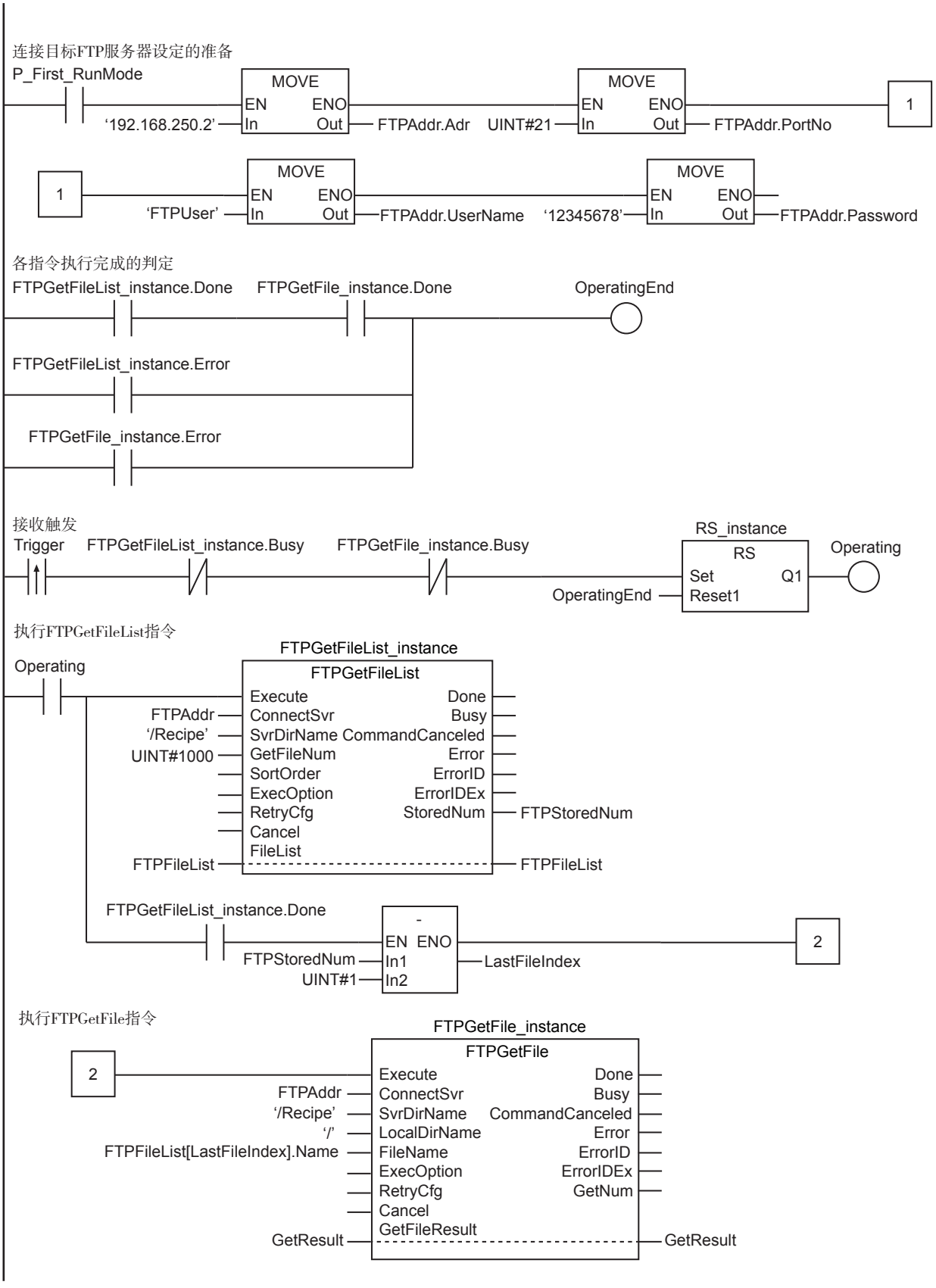
设定项目	规格
FTP服务器的目录名称	‘/Recipe’
获取文件数	1000
排列顺序	名称升序
保存文件详细信息的变量	FTPFileList[]

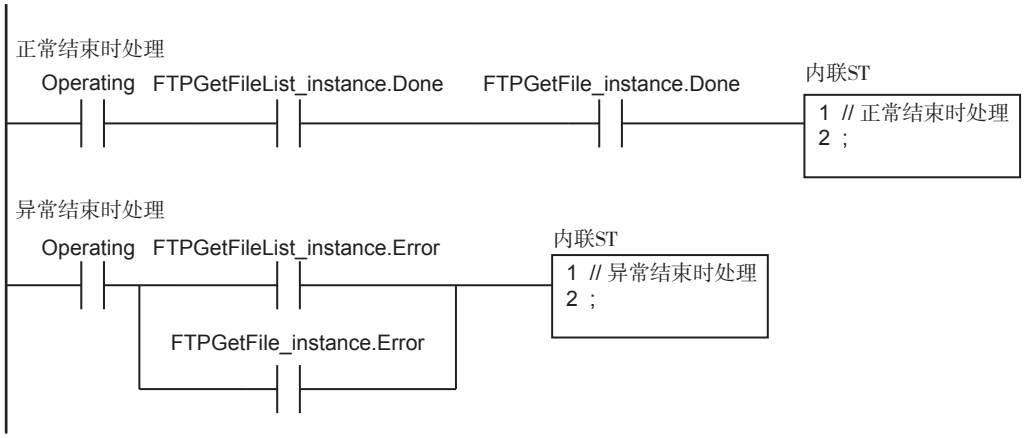
2　使用FTPGetFile指令，从通过步骤1获取的文件列表中，下载按名称升序排列的最后1个文件。文件的保存位置为SD存储卡的根目录。

3　所有处理正常结束后，执行正常结束时处理。发生异常时，执行异常结束时处理。

LD

内部变量	名称	数据类型	初始值	注释
	FTPGetFileList_instance	FTPGetFileList		FTPGetFileList指令的实例
	FTPGetFile_instance	FTPGetFile		FTPGetFile指令的实例
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	连接目标FTP服务器设定
	FTPFileList	ARRAY[0..999] OF _sFTP_FILE_DETAIL	[1000((Name := '', ModifiedDate := DT#1970-01-01-00:00:00, Size := 0, ReadOnly := False, Folder := False))]	文件详细信息
	GetResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	已下载文件结果
	FTPStoredNum	UINT	0	已获取文件列表文件数
	LastFileIndex	UINT	0	按名称升序排列的最后1个文件的索引
	RS_instance	RS		RS指令的实例
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中


连接目标FTP服务器设定的准备
P_First_RunMode
MOVE
EN ENO
'192.168.250.2' In Out FTPAddr.Adr
MOVE
EN ENO
UINT#21 In Out FTPAddr.PortNo
1
1
MOVE
EN ENO
'FTPUser' In Out FTPAddr.UserName
MOVE
EN ENO
'12345678' In Out FTPAddr.Password
各指令执行完成的判定
FTPGetFileList_instance.Done
FTPGetFile_instance.Done
OperatingEnd
FTPGetFileList_instance.Error
FTPGetFile_instance.Error
接收触发
Trigger
FTPGetFileList_instance.Busy
FTPGetFile_instance.Busy
RS_instance
RS
Set Q1
OperatingEnd Reset1
Operating
执行FTPGetFileList指令
FTPGetFileList_instance
Operating
FTPGetFileList
Execute Done
FTPAddr ConnectSvr Busy
'/Recipe' SvrDirName CommandCanceled
UINT#1000 GetFileNum Error
SortOrder ErrorID
ExecOption ErrorIDEx
RetryCfg StoredNum FTPStoredNum
Cancel
FileList
FTPFileList FTPFileList
FTPGetFileList_instance.Done
-
EN ENO
FTPStoredNum In1 LastFileIndex
UINT#1 In2
2
执行FTPGetFile指令
FTPGetFile_instance
FTPGetFile
2
Execute Done
FTPAddr ConnectSvr Busy
'/Recipe' SvrDirName CommandCanceled
'/' LocalDirName Error
FTPFileList[LastFileIndex].Name FileName ErrorID
ExecOption ErrorIDEx
RetryCfg GetNum
Cancel
GetFileResult
GetResult GetResult


正常结束时处理
Operating
FTPGetFileList_instance.Done
FTPGetFile_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating
FTPGetFileList_instance.Error
内联ST
1 // 异常结束时处理
2 ;
FTPGetFile_instance.Error

ST

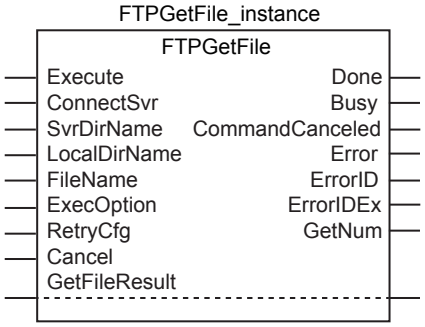
内部变量	名称	数据类型	初始值	注释
	R_TRIG_instance	R_TRIG		R_TRIG指令的实例
	UP_Q	BOOL	FALSE	触发输出
	FTPGetFile_instance	FTPGetFile		FTPGetFile指令的实例
	FTPGetFileList_instance	FTPGetFileList		FTPGetFileList指令的实例
	FTPFileList	ARRAY[0..999] OF _sFTP_FILE_DETAIL	[1000((Name := '', ModifiedDate := DT#1970-01-01-00:00:00, Size := 0, ReadOnly := False, Folder := False))]	文件详细信息
	FTPStoredNum	UINT	0	已获取文件列表文件数
	DoFTPTrigger	BOOL	FALSE	FTPGetFileList、FTPGetFIle执行条件
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	连接目标FTP服务器设定
	GetResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	已下载文件结果
	Stage	UINT	0	指令执行步骤
	Trigger	BOOL	FALSE	执行条件

```
// 连接目标FTP服务器设定的准备
IF P_First_RunMode THEN
  FTPAddr.Adr      := '192.168.250.2'; // 地址
  FTPAddr.PortNo   := UINT#21;      // 端口编号
  FTPAddr.UserName := 'FtpUser';    // 用户名
  FTPAddr.Password := '12345678';   // 密码
END_IF;

// 接收触发
R_TRIG_instance(Trigger, UP_Q);
IF ( (UP_Q = TRUE) AND (FTPGetFileList_instance.Busy = FALSE) AND
    (FTPGetFile_instance.Busy = FALSE) ) THEN
  DoFTPTrigger := TRUE;
  Stage := INT#1;
  FTPGetFileList_instance( // 实例初始化
    Execute     := FALSE,
    ConnectSvr  := FTPAddr,
    SvrDirName  := '/Recipe',
    GetFileNum  := UINT#1000,
    FileList    := FTPFileList,
    StoredNum   => FTPStoredNum) ;
  FTPGetFile_instance( // 实例初始化
    Execute        := FALSE,
    ConnectSvr     := FTPAddr,
    SvrDirName     := '/Recipe',
    LocalDirName   := '/',
    FileName       := '',
    GetFileResult  := GetResult) ;
END_IF;
```

```
IF (DoFTPTrigger = TRUE) THEN
  CASE Stage OF
    1 : // 执行FTPGetFileList指令
      FTPGetFileList_instance(
        Execute      := TRUE,        // 启动
        ConnectSvr  := FTPAddr,      // 连接目标FTP服务器
        SvrDirName := '/Recipe',    // FTP服务器目录名称
        GetFileNum := UINT#1000,    // 获取文件数
        FileList     := FTPFileList,  // 文件详情
        StoredNum  => FTPStoredNum) ;// 已获取文件数
      IF (FTPGetFileList_instance.Done = TRUE) THEN
        Stage := INT#2;           // 至下一个
      ELSIF (FTPGetFileList_instance.Error = TRUE) THEN
        Stage := INT#10;          // 异常结束
      END_IF;
    2 : // 执行FTPGetFile指令
      FTPGetFile_instance(
        Execute          := TRUE,                    // 启动
        ConnectSvr       := FTPAddr,                 // 连接目标FTP服务器
        SvrDirName       := '/Recipe',               // FTP服务器目录名称
        LocalDirName  := '/',                        // 本地目录名称
        FileName         := FTPFileList[FTPStoredNum - 1].Name,// 文件名
        GetFileResult   := GetResult) ;              // 下载文件结果
      IF (FTPGetFile_instance.Done = TRUE) THEN
        Stage := INT#0;                              // 正常结束
      ELSIF (FTPGetFile_instance.Error = TRUE) THEN
        Stage := INT#20;                             // 异常结束
      END_IF;
    0 : // 正常结束处理
      DoFTPTrigger:=FALSE;
      Trigger          :=FALSE;
    ELSE // 异常结束处理
      DoFTPTrigger:=FALSE;
      Trigger          :=FALSE;
  END_CASE;
END_IF;
```

FTPGetFile

从FTP服务器下载文件。

指令	名称	FB/FUN	图形表现	ST表现
FTPGetFile	从FTP服务器下载文件	FB		FTPGetFile_instance(Execute, ConnectSvr, SvrDirName, LocalDirName, FileName, ExecOption, RetryCfg, Cancel, GetFileResult, Done, Busy, CommandCanceled, Error, ErrorID, ErrorIDEx, GetNum) ;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	输入	连接目标FTP服务器的各种设定参数	–	–	*1
SvrDirName	FTP服务器目录名称		文件下载源FTP服务器的目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)*2		''*3
LocalDirName	本地目录名称		保存从FTP服务器下载的文件的目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)		'/'
FileName	文件名		待下载文件名*4	最大256字节(255个半角英数字字符+结尾NULL字符)*5		*1
ExecOption	FTP执行选项		FTP执行的相关选项	–		–
RetryCfg	执行重试设定		指令执行重试设定			
Cancel	取消		TRUE：取消指令执行 FALSE：不取消指令执行	遵从数据类型		FALSE
GetFileResult[]数组*6*7*8	已下载文件结果	输入输出	下载的文件结果	–	–	*1

	名称	输入/输出	内容	有效范围	单位	初始值
Command Canceled	取消完成	输出	TRUE ：取消完成 FALSE：取消未完成	遵从数据类型	–	–
GetNum	下载文件数		下载对象文件数	–		

*1 省略输入参数时，初始值不适用。编连时会发生异常。
2 FTP服务器目录名称，不能使用下列字符。“”、“?”、“<”、“>”、“|”、“"”(双引号)
*3 FTP服务器的登录时的主目录。
*4 文件名可指定通配符。
*5 文件名不能使用下列字符。“|”
*6 数组的最大元素数为1000。
*7 1维数组。指定2维以上数组的情况下，编连时会发生异常。
*8 开头的元素编号为0。指定开头元素编号非0的数组的情况下，编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
ConnectSvr	结构体_sFTP_CONNECT_SVR 详情参阅功能说明																			
SvrDirName																				○
LocalDirName																				○
FileName																				○
ExecOption	结构体_sFTP_EXEC_OPTION 详情参阅功能说明																			
RetryCfg	结构体_sFTP_RETRY_CFG 详情参阅功能说明																			
Cancel	○																			
GetFileResult[]数组	结构体_sFTP_FILE_RESULT 详情参阅功能说明																			
Command Canceled	○																			
GetNum							○													

功能

将连接目标FTP服务器 “ConnectSvr”上的指定目录 “SvrDirName”中的指定文件 “FileName”，下载到SD存储卡内指定目录 “LocalDirName”中。SD存储卡内没有指定目录 “LocalDirName”时，创建目录并下载文件。
“FileName”可指定通配符。因此，可一次下载多个文件。

下载的结果以文件为单位保存在下载文件结果GetFileResult[]中。在下载文件数 “GetNum”中保存下载对象文件数。对 “FileName”指定通配符时，保存与通配符相符的文件数。

传送文件过程中发生异常时，“GetFileResult[].TxError”的值变为TRUE。
下载后，删除传送源文件发生异常时，“GetFileResult[].RemoveError”的值变为TRUE。

NY系列控制器则下载到共享文件夹(虚拟SD存储卡)内。下载需要事先设定虚拟SD存储卡。虚拟SD存储卡的详情，请参阅 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

“ConnectSvr”的数据类型为结构体_sFTP_CONNECT_SVR。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	连接目标FTP服务器的各种设定参数	_sFTP_CONNECT_SVR	–	–	–
Adr	地址	IP地址或主机名称*1	STRING	1 ~ 200字节*2	–	–
PortNo	端口编号	FTP服务器的控制连接的TCP端口编号	UINT	0 ~ 65535*3		
UserName	用户名	FTP服务器的用户名	STRING	最大33字节 *2*4*5		
Password	密码	FTP服务器的密码	STRING	最大33字节 *2*4*5		

*1 指定主机名称时，必须另行设定DNS或Hosts。
*2 可使用的字符为半角的 “A-Z”、“a-z”、“0-9”、“-”(连字符)、“.”(句号)、“_”(下划线)。
*3 为0时，TCP端口编号为21。
*4 字节数含结尾NULL字符。
*5 CPU单元Ver.1.08请指定至少1个字符的字符串。指定仅结尾NULL字符的字符串时，将发生异常。

GetFileResult[]的数据类型为结构体_sFTP_FILE_RESULT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
GetFileResult	已下载文件结果	传送文件的结果	_sFTP_FILE_RESULT	–	–	–
Name	文件名*1	已传送文件名	STRING	最大256字节 (255个半角英数字字符+结尾NULL字符)	–	–
TxError	传送异常	TRUE：传送异常结束 FALSE：传送正常结束	BOOL	遵从数据类型		
RemoveError	删除异常	TRUE：删除异常结束 FALSE：删除正常结束	BOOL			
Reserved	保留	系统保留	ARRAY[0..3] Of Byte	–		0

*1 带后缀。

使用通配符指定文件名

要下载的文件名 “FileName”可指定通配符。
通配符指定字符使用 “*”和 “?”。“*”表示多个字符，“?”表示1个字符。
通配符指定方法的示例如下所示。

设FTP服务器目录的文件构成如下所示。

```
┣DataFiles(对象目录)
┃  ┣ LogA01.log
┃  ┣ LogA02.txt
┃  ┣ LogB.log
┃  ┣ LogC.txt
┃  ┣ ControlDataA1.csv
┃  ┣ ControlDataA10.csv
┃  ┣ ControlDataA100.csv
┃  ┣ ControlDataB10.csv
┃  ┣ ControlDataC100.csv
┃  ┃
┃  ┗ControlSubDataFiles(子目录)
┃       SubData_A001.txt
┃       SubData_A002.txt
┃
```

如下表所示，对象文件由通配符指定方法确定。

通配符指定	对象文件
Log*.log	LogA01.log, LogB.log
Log?.log	LogB.log
Log?.*	LogB.log、LogC.txt
Data	ControlDataA1.csv, ControlDataA10.csv, ControlDataA100.csv, ControlDataB10.csv, ControlDataC100.csv、(ControlSubDataFiles)*1
*	子目录内文件以外的所有文件
.	子目录内文件以外的所有文件
?.?	无对象文件
????.???	LogB.log、LogC.txt

*1 根据FTP服务器规格的不同，子目录也可能包含在对象中。

指定通配符后，待下载文件的最大数量为1000个。
下载的结果，任一文件如 “GetFileResult[].TxError”或 “GetFileResult[].RemoveError”变为TRUE，则 “Error”变为TRUE，保存与最初异常的 “ErrorID”相符的错误代码，将FTP服务器的错误响应保存在 “ErrorIDEx”中。

与FTP服务器的处理相关的选项指定

从FTP服务器下载文件时，进行FTP执行选项 “ExecOption”指定的动作。

“ExecOption”的数据类型为结构体_sFTP_EXEC_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ExecOption	FTP执行选项	FTP执行的相关选项	_sFTP_EXEC_OPTION	–	–	–
PassiveMode	被动模式指定	TRUE：Passive模式 FALSE：Active模式	BOOL	遵从数据类型	–	FALSE
ASCIIMode	ASCII模式指定	TRUE：ASCII模式 FALSE：Binary模式	BOOL			
FileRemove	传送后文件删除指定*1	TRUE：删除传送后文件 FALSE：不删除传送后文件	BOOL			
OverWrite	覆盖指定	TRUE：覆盖传送目标的文件 FALSE：不覆盖传送目标的文件	BOOL			
Reserved	保留	系统保留	ARRAY[0..7] Of Byte	–		0

*1 传送失败时，不删除传送源文件。

对各选项的含义进行说明。

● “PassiveMode”(被动模式指定)

被动模式指定是指，在被动模式下是否执行与FTP服务器之间数据连接的连接请求的指定。不指定被动模式时，在主动模式下执行与FTP服务器之间数据连接的连接请求。

连接请求方法的详情，🕮 请参阅“NJ/NX系列 CPU单元内置EtherNet/IP端口 用户手册(SBCD–359)”或🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 内置EtherNet/IP端口篇(SBCD–369)”。

“PassiveMode”的值及其含义如下所述。

设定值	含义
TRUE	在被动模式下执行与FTP服务器之间数据连接的连接请求。 由FTP客户端侧执行数据连接的连接请求。
FALSE	在主动模式下执行与FTP服务器之间数据连接的连接请求。 由FTP服务器侧执行数据连接的连接请求。

- **“ASCIIMode”(ASCII模式指定)**

ASCII模式指定是指，是否将从传送源系统至传送目标系统的传送模式设为ASCII模式的指定。不指定ASCII模式时，从传送源系统至传送目标系统的传送模式为Binary模式。
“ASCIIMode”的值及其含义如下所述。

设定值	含义
TRUE	将从传送源系统至传送目标系统的传送模式设为ASCII模式。 将文本的换行代码从传送源系统转换为传送目标系统的代码并传送。
FALSE	将从传送源系统至传送目标系统的传送模式设为Binary模式。 不对文本的换行代码进行转换，保持传送源系统的状态直接传送。

- **“FileRemove”(传送后文件删除指定)**

传送后文件删除指定是指，将传送源文件下载至传送目标后，是否删除传送源文件的指定。
“FileRemove”的值及其含义如下所述。

设定值	含义
TRUE	删除传送源文件。
FALSE	不删除传送源文件。

- **“OverWrite”(覆盖指定)**

覆盖指定是指，将文件下载至传送目标时，如传送目标存在同名文件，是否覆盖的指定。不覆盖时，如传送目标存在同名文件，则不传送文件。
文件名无大小写之分。
“OverWrite”的值及其含义如下所述。

设定值	含义
TRUE	覆盖传送目标的文件。
FALSE	不覆盖传送目标的文件。不向传送目标传送文件。

与FTP服务器之间连接处理的重试指定

可进行与FTP服务器的连接重试。
重试设定内容和动作与 “FTPGetFileList指令(P.2–1105)”的规格相同。请参阅该处。

取消指令执行

FTPGetFile指令执行中，可取消指令执行。
取消前，从FTP服务器下载文件的结果，保存在 “GetNum”和已下载文件结果GetFileResult[]中。
取消动作与 “FTPGetFileList指令(P.2–1105)”的规格相同。请参阅该处。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			
_Card1Ready	可使用SD存储卡标志	BOOL	可识别SD存储卡，表示可以使用。 TRUE ：可使用 FALSE：无法使用

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

使用注意事项

- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 已下载文件结果中保存的结果数超过 GetFileResult[] 的数组元素数时，不保存超出部分的结果信息。此时，“Error”不会变为TRUE。
- 关于超出255个字符的文件名，将从开头起的255个字符部分保存在GetFileResult[]的“Name”中。此时，“Error”不会变为TRUE。
- FTPGetFileList指令、FTPGetFile指令、FTPPutFile指令、FTPRemoveFile指令、FTPRemoveDir指令可同时执行的数量最多为3个。
- 使用通配符指定文件名，多个文件发生异常时，GetFileResult[]中保存的文件中，“GetFileResult[].TxError”的值为TRUE的起始文件的结果将保存在 “ErrorID”、“ErrorIDEx”中。
- 文件名无大小写之分。因此，传送目标中存在与传送文件仅文件名大小写不同的文件时，将识别为同名文件。此时的处理如下所示。

“OverWrite”的值	覆盖指定	处理
TRUE	覆盖	覆盖。
FALSE	不覆盖	不覆盖传送目标的文件。不向传送目标传送文件。

- FTP服务器内的指定目录中不存在 “FileName”指定文件时，发生传送异常，“GetFileResult[].TxError”的值变为TRUE。
- “FileName”指定文件如为目录，则发生传送异常，“GetFileResult[].TxError”的值变为TRUE。
- FTP执行选项的传送后文件删除指定 “ExecOption.FileRemove”为TRUE，“FileName”指定文件的属性为只读时，发生删除异常，“GetFileResult[].RemoveError”变为TRUE。

- 以下情况时会发生异常。“Error”变为TRUE。
  - 任一输入参数的值超过有效范围时
  - 在 “SvrDirName”、“LocalDirName”中指定上1层的目录".."时
  - 在 “SvrDirName”、“LocalDirName”中指定"//"等错误路径时
  - FTP服务器中不存在 “SvrDirName”指定的目录时
  - “SvrDirName”指定的FTP服务器的目录中存在1000个以上下载对象文件时
  - FTP服务器下载源目录中不存在 “FileName”指定的文件时
  - “ExecOption.OverWrite”为FALSE，且指定目录 “SvrDirName”中已存在与指定文件名 “FileName”同名的文件时
  - “ExecOption.FileRemove”为TRUE，且与指定文件名 “FileName”一致的文件的属性为只读时
  - 网络上不存在 “ConnectSvr”指定的FTP服务器，或者服务已停止时
  - 因无文件访问权或文件已损坏等理由，导致访问 “FileName”指定文件失败时
  - 同时执行的 FTPGetFileList 指令、FTPGetFile 指令、FTPPutFile 指令、FTPRemoveFile 指令、FTPRemoveDir指令超过3个时。
  - SD存储卡并非可使用状态时
  - SD存储卡处于写保护状态时
  - SD存储卡空间不足时
  - 超出SD存储卡中可创建的文件数量、目录数量时
- 扩展错误代码 “ErrorIDEx”在本指令中表示FTP服务器回复的FTP响应代码。典型的 “ErrorIDEx”的值、异常内容和处理方法如下所述。详情请确认FTP服务器的规格。“ErrorIDEx”在错误代码 “ErrorID”的值为WORD#16#2407时输出。

“ErrorIDEx”的值	异常内容	处理方法
16#000001A9	未能建立数据连接。	与互联网上的FTP服务器进行FTP通信时，请确认FTP的打开模式是否为Active。
16#000001AA	连接已关闭。停止了数据传输。	请确认与FTP服务器的连接。 请确认FTP服务器是否停止。
16#000001C2	未进行请求的文件操作。文件已被打开等情况时，无法使用文件。	请确认对象文件是否已被其他应用程序打开。
16#00000212	用户未能登录。	请确认FTP用户名或密码。
16#00000214	需要文件保存用账号。	请确认FTP用户的访问权。
16#00000226	找不到文件而无法访问等情况时，无法使用文件，因此未执行请求的文件操作。	请确认FTP服务器的目录中是否存在与指定文件名相符的文件。 请确认指定文件的访问权。
16#00000229	文件名不正确，未能执行。	请确认指定目录的访问权。

版本相关信息

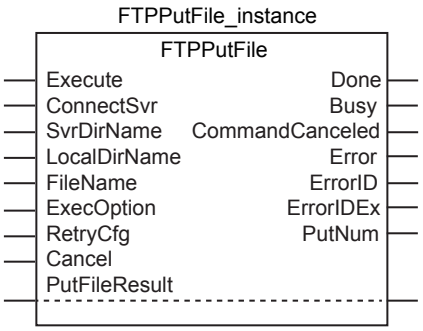
本指令可用于CPU单元Ver.1.08 以上且Sysmac Studio Ver.1.09 以上。

示例程序

请参阅 “FTPGetFileList指令(P.2–1105)”的示例程序。

FTPPutFile

文件上传至FTP服务器。

指令	名称	FB/FUN	图形表现	ST表现
FTPPutFile	文件上传至FTP服务器	FB		FTPPutFile_instance(Execute, ConnectSvr, SvrDirName, LocalDirName, FileName, ExecOption, RetryCfg, Cancel, PutFileResult, Done, Busy, CommandCanceled, Error, ErrorID, ErrorIDEx, PutNum) ;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	输入	连接目标FTP服务器的各种设定参数	–	–	*1
SvrDirName	FTP服务器目录名称		文件上传目标FTP服务器的目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)*2		""*3
LocalDirName	本地目录名称		保存FTP服务器上传文件的目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)		'/'
FileName	文件名		待上传文件名*4			*1
ExecOption	FTP执行选项		FTP执行的相关选项	–		–
RetryCfg	执行重试设定		指令执行重试设定			
Cancel	取消		TRUE：取消指令执行 FALSE：不取消指令执行	遵从数据类型		FALSE
PutFileResult[]数组*5*6*7	已上传文件结果	输入输出	上传文件结果	–	–	*1
CommandCanceled	取消完成	输出	TRUE：取消完成 FALSE：取消未完成	遵从数据类型	–	–
PutNum	上传文件数量		上传对象文件数	–		

*1 省略输入参数时，初始值不适用。编连时会发生异常。
2 FTP服务器目录名称，不能使用下列字符。“”、“?”、“<”、“>”、“|”、“"”(双引号)
*3 FTP服务器的登录时的主目录。
*4 文件名可指定通配符。
*5 数组的最大元素数为1000。
*6 1维数组。指定2维以上数组的情况下，编连时会发生异常。

*7 开头的元素编号为0。指定开头元素编号非0的数组的情况下，编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
ConnectSvr	结构体_sFTP_CONNECT_SVR 详情参阅功能说明																			
SvrDirName																				○
LocalDirName																				○
FileName																				○
ExecOption	结构体_sFTP_EXEC_OPTION 详情参阅功能说明																			
RetryCfg	结构体_sFTP_RETRY_CFG 详情参阅功能说明																			
Cancel	○																			
PutFileResult[]数组	结构体_sFTP_FILE_RESULT 详情参阅功能说明																			
Command Canceled	○																			
PutNum							○													

功能

将SD存储卡内指定目录 “LocalDirName”中的指定文件 “FileName”，上传到连接目标FTP服务器 “ConnectSvr”指定目录 “SvrDirName”中。FTP 服务器中没有指定目录 “SvrDirName”时，创建目录并上传文件。
“FileName”可指定通配符。因此，可一次上传多个文件。

上传的结果以文件为单位保存在上传文件结果PutFileResult[]中。在上传文件数 “PutNum”中保存上传对象文件数。对 “FileName”指定通配符时，保存与通配符相符的文件数。

传送文件过程中发生异常时，“PutFileResult[].TxError”的值变为TRUE。
上传后，删除传送源文件发生异常时，“PutFileResult[].RemoveError”的值变为TRUE。

NY系列控制器则下载到共享文件夹(虚拟SD存储卡)内。下载需要事先设定虚拟SD存储卡。共享文件夹(虚拟SD存储卡)的详情，请参阅 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

“ConnectSvr”的数据类型为结构体_sFTP_CONNECT_SVR。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	连接目标FTP服务器的各种设定参数	_sFTP_CONNECT_SVR	–	–	–
Adr	地址	IP地址或主机名称*1	STRING	1 ~ 200字节*2	–	–
PortNo	端口编号	FTP服务器的控制连接的TCP端口编号	UINT	0 ~ 65535*3		
UserName	用户名	FTP服务器的用户名	STRING	最大33字节 *2*4*5		
Password	密码	FTP服务器的密码	STRING	最大33字节 *2*4*5		

*1 指定主机名称时，必须另行设定DNS或Hosts。
*2 可使用的字符为半角的 “A–Z”、“a–z”、“0–9”、“–”(连字符)、“.”(句号)、“_”(下划线)。
*3 为0时，TCP端口编号为21。
*4 字节数含结尾NULL字符。
*5 CPU单元Ver.1.08请指定至少1个字符的字符串。指定仅结尾NULL字符的字符串时，将发生异常。

PutFileResult[]的数据类型为结构体_sFTP_FILE_RESULT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
PutFileResult	已上传文件结果	传送文件的结果	_sFTP_FILE_RESULT	–	–	–
Name	文件名*1	已传送文件名	STRING	最大256字节 (255个半角英数字字符+结尾NULL字符)	–	–
TxError	传送异常	TRUE：传送异常结束 FALSE：传送正常结束	BOOL	遵从数据类型		
RemoveError	删除异常	TRUE：删除异常结束 FALSE：删除正常结束	BOOL			
Reserved	保留	系统保留	ARRAY[0..3] Of Byte	–		0

*1 带后缀。

使用通配符指定文件名

要上传的文件名可指定通配符。
通配符的指定与 “FTPGetFile指令(P.2–1120)”的规格相同。请参阅该处。

与FTP服务器的处理相关的选项指定

上传时，可指定与FTP服务器的处理选项。
选项设定内容与 “FTPGetFile指令(P.2–1120)”的规格相同。请参阅该处。

与FTP服务器之间连接处理的重试指定

可进行与FTP服务器的连接处理的重试。
重试设定内容和动作与 “FTPGetFileList指令(P.2–1105)”的规格相同。请参阅该处。

取消指令执行

FTPPutFile指令执行中，可取消指令执行。
取消前，从FTP服务器上传文件的结果，保存在 “PutNum”和已上传文件结果PutFileResult[]中。
取消动作与 “FTPGetFileList指令(P.2–1105)” 的规格相同。请参阅该处。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			
_Card1Ready	可使用SD存储卡标志	BOOL	可识别SD存储卡，表示可以使用。 TRUE ：可使用 FALSE：无法使用

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

使用注意事项

- 本指令仅适用于NJ/NX系列 CPU单元及NY系列控制器的内置EtherNet/IP端口。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 已上传文件结果中保存的结果数超过 PutFileResult[] 的数组元素数时，不保存超出部分的结果信息。此时，“Error”不会变为TRUE。
- 关于超出255个字符的文件名，将从开头起的255个字符部分保存在PutFileResult[]的 “Name”中。此时，“Error”不会变为TRUE。
- FTPGetFileList指令、FTPGetFile指令、FTPPutFile指令、FTPRemoveFile指令、FTPRemoveDir指令可同时执行的数量最多为3个。
- 使用通配符指定文件名，多个文件发生异常时，PutFileResult[]中保存的文件中，“PutFileResult[].TxError”的值为TRUE的起始文件的结果将保存在 “ErrorID”、“ErrorIDEx”中。
- 文件名无大小写之分。因此，传送目标中存在与传送文件仅文件名大小写不同的文件时，将识别为同名文件。此时的处理如下所示。

“OverWrite”的值	覆盖指定	处理
TRUE	覆盖	是否覆盖取决于FTP服务器的规格。
FALSE	不覆盖	不覆盖传送目标的文件。不向传送目标传送文件。

- SD 存储卡内的指定目录中不存在 “FileName”指定文件时，发生传送异常，“PutFileResult[].TxError”的值变为TRUE。
- “FileName”指定文件若为目录，则发生传送异常，“PutFileResult[].TxError”的值变为TRUE。
- 传送后文件删除指定 “ExecOption.FileRemove”为TRUE，“FileName”指定文件的属性为只读时，发生删除异常，“PutFileResult[].RemoveError”的值变为TRUE。

- 以下情况时会发生异常。“Error”变为TRUE。
  - 任一输入参数的值超过有效范围时
  - 在 “SvrDirName”、“LocalDirName”中指定上1层的目录".."时
  - 在 “SvrDirName”、“LocalDirName”中指定"//"等错误路径时
  - FTP服务器中不存在 “SvrDirName”指定的目录时
  - FTP客户端不存在 “LocalDirName”指定的目录时
  - “LocalDirName”指定的目录中存在1000个以上上传对象文件时
  - SD存储卡的上传源目录中不存在 “FileName”指定的文件时
  - “ExecOption.OverWrite”为FALSE，且指定目录 “SvrDirName”中已存在与指定文件名 “FileName”同名的文件时
  - “ExecOption.FileRemove”为TRUE，且与指定文件名 “FileName”一致的文件的属性为只读时
  - 网络上不存在 “ConnectSvr”指定的FTP服务器，或者服务已停止时
  - 因无文件访问权或文件已损坏等理由，导致访问 “FileName”指定文件失败时
  - 同时执行的 FTPGetFileList 指令、FTPGetFile 指令、FTPPutFile 指令、FTPRemoveFile 指令、FTPRemoveDir指令超过3个时。
  - SD存储卡并非可使用状态时
- 扩展错误代码 “ErrorIDEx”在本指令中表示FTP服务器回复的FTP响应代码。典型的 “ErrorIDEx”的值、异常内容和处理方法如下所述。详情请确认FTP服务器的规格。“ErrorIDEx”在错误代码 “ErrorID”的值为WORD#16#2407时输出。

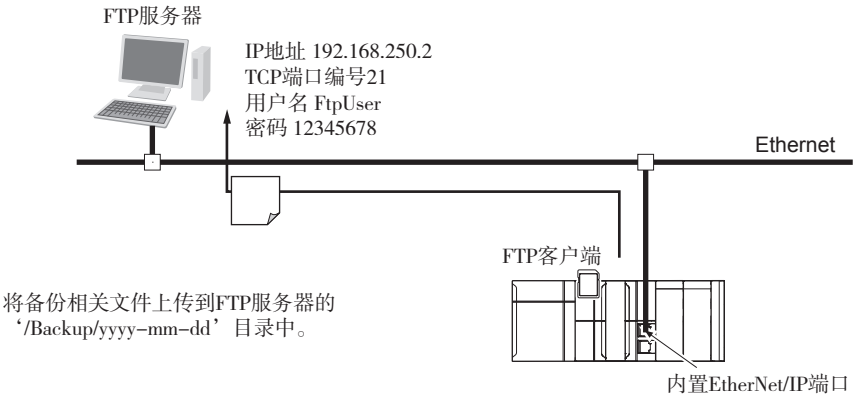
“ErrorIDEx”的值	异常内容	处理方法
16#000001A9	未能建立数据连接。	与互联网上的FTP服务器进行FTP通信时，请确认FTP的打开模式是否为Active。
16#000001AA	连接已关闭。停止了数据传输。	请确认与FTP服务器的连接。 请确认FTP服务器是否停止。
16#000001C2	未进行请求的文件操作。文件已被打开等情况时，无法使用文件。	请确认对象文件是否已被其他应用程序打开。
16#00000212	用户未能登录。	请确认FTP用户名或密码。
16#00000214	需要文件保存用账号。	请确认FTP用户的访问权。
16#00000226	找不到文件而无法访问等情况时，无法使用文件，因此未执行请求的文件操作。	请确认FTP服务器的目录中是否存在与指定文件名相符的文件。 请确认指定文件的访问权。
16#00000229	文件名不正确，未能执行。	请确认指定目录的访问权。

版本相关信息

本指令可用于CPU单元Ver.1.08 以上且Sysmac Studio Ver.1.09 以上。

示例程序

执行SD存储卡备份，将所有备份相关文件上传到FTP服务器的
‘/Backup/yyyy-mm-dd’目录中。



控制器与FTP服务器之间通过EtherNet/IP网络连接。为与FTP服务器连接而设定的参数值如下所示。

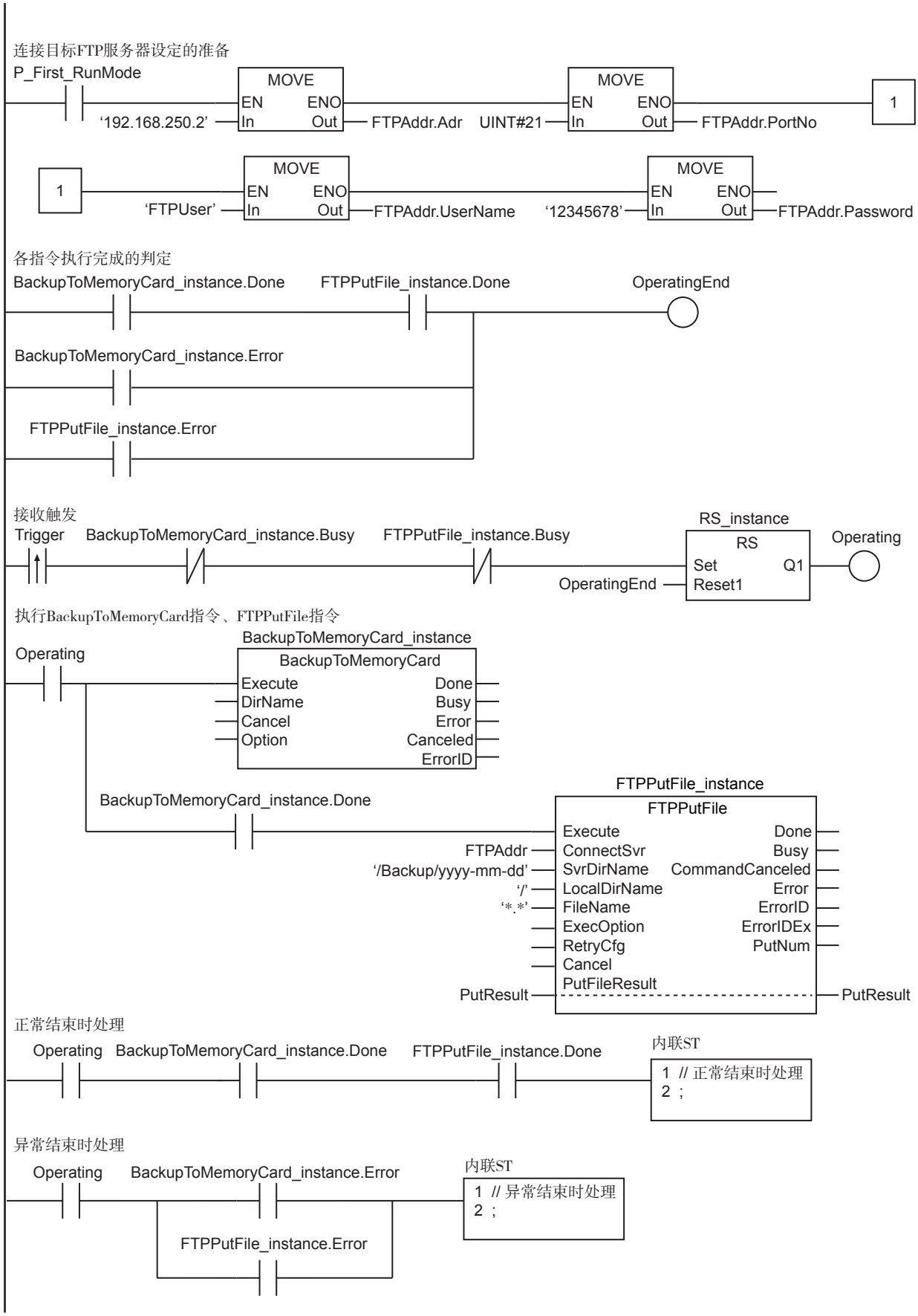
设定参数	值
IP地址	192.168.250.2
TCP端口编号	21
用户名	FtpUser
密码	12345678

处理步骤如下所述。

1. 使用BackupToMemoryCard指令，将NJ/NX系列 控制器的备份相关文件保存到SD存储卡的根目录中。
2. 使用FTPPutFile指令，将备份相关文件上传到FTP服务器的‘/Backup/yyyy-mm-dd’目录中。将传送文件名指定通配符 ‘*.*’。
3. 所有处理正常结束后，执行正常结束时处理。发生异常时，执行异常结束时处理。

LD

内部变量	名称	数据类型	初始值	注释
	FTPPutFile_instance	FTPPutFile		FTPPutFile指令的实例
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	连接目标FTP服务器设定
	PutResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	已上传文件结果
	RS_instance	RS		RS指令的实例
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	BackupToMemoryCard_instance	BackupToMemoryCard		BackupToMemoryCard指令的实例


连接目标FTP服务器设定的准备
P_First_RunMode
MOVE
EN ENO
'192.168.250.2' In Out FTPAddr.Adr
MOVE
EN ENO
UINT#21 In Out FTPAddr.PortNo
1
1
MOVE
EN ENO
'FTPUser' In Out FTPAddr.UserName
MOVE
EN ENO
'12345678' In Out FTPAddr.Password
各指令执行完成的判定
BackupToMemoryCard_instance.Done
FTPPutFile_instance.Done
OperatingEnd
BackupToMemoryCard_instance.Error
FTPPutFile_instance.Error
接收触发
Trigger
BackupToMemoryCard_instance.Busy
FTPPutFile_instance.Busy
RS_instance
RS
Set Q1
OperatingEnd Reset1
Operating
执行BackupToMemoryCard指令、FTPPutFile指令
Operating
BackupToMemoryCard_instance
BackupToMemoryCard
Execute Done
DirName Busy
Cancel Error
Option Canceled
ErrorID
BackupToMemoryCard_instance.Done
FTPPutFile_instance
FTPPutFile
Execute Done
FTPAddr ConnectSvr Busy
'/Backup/yyyy-mm-dd' SvrDirName CommandCanceled
'/' LocalDirName Error
'*.*' FileName ErrorID
ExecOption ErrorIDEx
RetryCfg PutNum
Cancel
PutFileResult
PutResult PutResult
正常结束时处理
Operating
BackupToMemoryCard_instance.Done
FTPPutFile_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating
BackupToMemoryCard_instance.Error
FTPPutFile_instance.Error
内联ST
1 // 异常结束时处理
2 ;

ST

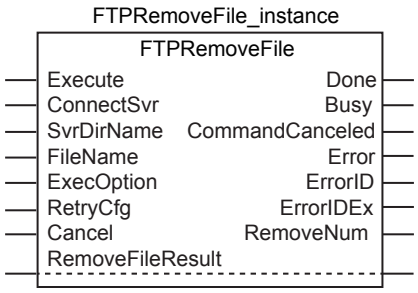
内部变量	名称	数据类型	初始值	注释
	R_TRIG_instance	R_TRIG		R_TRIG指令的实例
	UP_Q	BOOL	FALSE	触发输出
	FTPPutFile _instance	FTPPutFile		FTPPutFile指令的实例
	DoFTPTrigger	BOOL	FALSE	BackupToMemoryCard、FTPPutFIle执行条件
	FTPAddr	_sFTP_CONNECT _SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	连接目标FTP服务器设定
	PutResult	ARRAY[0..0] OF _sFTP_FILE _RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	已上传文件结果
	Stage	UINT	0	指令执行步骤
	Trigger	BOOL	FALSE	执行条件
	BackupToMemory Card_instance	BackupToMemory Card		BackupToMemoryCard指令的实例

```
// 连接目标FTP服务器设定的准备
IF P_First_RunMode THEN
  FTPAddr.Adr      := '192.168.250.2';// 地址
  FTPAddr.PortNo   := UINT#21;      // 端口编号
  FTPAddr.UserName := 'FtpUser';    // 用户名
  FTPAddr.Password := '12345678';   // 密码
END_IF;

// 接收触发
R_TRIG_instance(Trigger, UP_Q);
IF ( (UP_Q = TRUE) AND (BackupToMemoryCard_instance.Busy = FALSE) AND
    (FTPPutFile_instance.Busy = FALSE) ) THEN
  DoFTPTrigger := TRUE;
  Stage := INT#1;
  BackupToMemoryCard_instance( // 实例初始化
    Execute := FALSE) ;
  FTPPutFile_instance(   // 实例初始化
    Execute      := FALSE,
    ConnectSvr   := FTPAddr,
    SvrDirName   := '/Backup/yyyy-mm-dd',
    LocalDirName := '/',
    FileName     := '*.*',
    PutFileResult := PutResult) ;
END_IF;

IF (DoFTPTrigger = TRUE) THEN
  CASE Stage OF
    1 :// 执行BackupToMemoryCard指令
      BackupToMemoryCard_instance(
        Execute := TRUE) ;// 启动
      IF (BackupToMemoryCard_instance.Done = TRUE) THEN
        Stage := INT#2; // 至下一个
      ELSIF (BackupToMemoryCard_instance.Error = TRUE) THEN
        Stage := INT#10; // 异常结束
      END_IF;
```

```
    2 : // 执行FTPPutFile指令
      FTPPutFile_instance(
        Execute          := TRUE,            // 启动
        ConnectSvr       := FTPAddr,         // 连接目标FTP服务器
        SvrDirName       := '/Backup/yyyy-mm-dd',// FTP服务器目录名称
        LocalDirName     := '/',             // 本地目录名称
        FileName         := '*.*',           // 文件名
        PutFileResult    := PutResult) ;     // 上传文件结果
      IF (FTPPutFile_instance.Done = TRUE) THEN
        Stage := INT#0; // 正常结束
      ELSIF (FTPPutFile_instance.Error = TRUE) THEN
        Stage := INT#20; // 异常结束
      END_IF;
    0 : // 正常结束处理
      DoFTPTrigger:=FALSE;
      Trigger        :=FALSE;
    ELSE // 异常结束处理
      DoFTPTrigger:=FALSE;
      Trigger        :=FALSE;
  END_CASE;
END_IF;
```

FTPRemoveFile

删除FTP服务器的文件。

指令	名称	FB/FUN	图形表现	ST表现
FTPRemoveFile	FTP服务器的文件删除	FB		FTPRemoveFile_instance(Execute, ConnectSvr, SvrDirName, FileName, ExecOption, RetryCfg, Cancel, RemoveFileResult, Done, Busy, CommandCanceled, Error, ErrorID, ErrorIDEx, RemoveNum) ;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	输入	连接目标FTP服务器的各种设定参数	–	–	*1
SvrDirName	FTP服务器目录名称		待删除文件所在FTP服务器的目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)*2		""*3
FileName	文件名		待删除的文件名*4	最大256字节(255个半角英数字字符+结尾NULL字符)*5		*1
ExecOption	FTP执行选项		FTP执行的相关选项	–		–
RetryCfg	执行重试设定		指令执行重试设定			
Cancel	取消		TRUE：取消指令执行 FALSE：不取消指令执行	遵从数据类型		
RemoveFileResult[]数组*6*7*8	已删除文件结果	输入输出	删除的文件结果	–	–	*1
CommandCanceled	取消完成	输出	TRUE：取消完成 FALSE：取消未完成	遵从数据类型	–	–
RemoveNum	删除文件数		删除对象文件数	–		

*1 省略输入参数时，初始值不适用。编连时会发生异常。
2 FTP服务器目录名称，不能使用下列字符。“”、“?”、“<”、“>”、“|”、“"”(双引号)
*3 FTP服务器的登录时的主目录。
*4 文件名可指定通配符。
*5 文件名不能使用下列字符。“|”

*6 数组的最大元素数为1000。
*7 1维数组。指定2维以上数组的情况下，编连时会发生异常。
*8 开头的元素编号为0。指定开头元素编号非0的数组的情况下，编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
ConnectSvr	结构体_sFTP_CONNECT_SVR 详情参阅功能说明																			
SvrDirName																				○
FileName																				○
ExecOption	结构体_sFTP_EXEC_OPTION 详情参阅功能说明																			
RetryCfg	结构体_sFTP_RETRY_CFG 详情参阅功能说明																			
Cancel	○																			
Remove FileResult[]数组	结构体_sFTP_FILE_RESULT 详情参阅功能说明																			
Command Canceled	○																			
RemoveNum							○													

功能

删除连接目标FTP服务器 “ConnectSvr”指定目录 “SvrDirName”中的指定文件 “FileName”。
“FileName”可指定通配符。因此，可一次删除多个文件。

删除的结果以文件为单位保存在已删除文件结果RemoveFileResult[]中。在删除文件数 “RemoveNum”中保存删除对象文件数。对 “FileName”指定通配符时，保存与通配符相符的文件数。

删除文件发生异常时，“RemoveFileResult[].RemoveError”的值变为TRUE。

“ConnectSvr”的数据类型为结构体_sFTP_CONNECT_SVR。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	连接目标FTP服务器的各种设定参数	_sFTP_CONNECT_SVR	–	–	–
Adr	地址	IP地址或主机名称*1	STRING	1 ~ 200字节*2	–	–
PortNo	端口编号	FTP服务器的控制连接的TCP端口编号	UINT	0 ~ 65535*3		
UserName	用户名	FTP服务器的用户名	STRING	最大33字节 *2*4*5		
Password	密码	FTP服务器的密码	STRING	最大33字节 *2*4*5		

*1 指定主机名称时，必须另行设定DNS或Hosts。
*2 可使用的字符为半角的 “A–Z”、“a–z”、“0–9”、“–”(连字符)、“.”(句号)、“_”(下划线)。
*3 为0时，TCP端口编号为21。
*4 字节数含结尾NULL字符。
*5 CPU单元Ver.1.08请指定至少1个字符的字符串。指定仅结尾NULL字符的字符串时，将发生异常。

RemoveFileResult[]的数据类型为结构体_sFTP_FILE_RESULT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
RemoveFileResult	已删除文件结果	传送文件的结果	_sFTP_FILE_RESULT	–	–	–
Name	文件名*1	已传送文件名	STRING	最大256字节 (255个半角英数字字符+结尾NULL字符)	–	–
TxError	传送异常	TRUE：传送异常结束 FALSE：传送正常结束	BOOL	遵从数据类型		
RemoveError	删除异常	TRUE：删除异常结束 FALSE：删除正常结束	BOOL			
Reserved	保留	系统保留	ARRAY[0..3] Of Byte	–		0

*1 带后缀。

使用通配符指定文件名

要删除的文件名可指定通配符。
通配符的指定方法与 “FTPGetFile指令(P.2–1120)”的规格相同。请参阅该处。

与FTP服务器的处理相关的选项指定

删除FTP服务器的文件时，进行FTP执行选项 “ExecOption”指定的动作。
选项设定内容与 “FTPGetFile指令(P.2–1120)”的规格相同。请参阅该处。
但，本指令有效的选项仅被动模式指定 “ExecOption.PassiveMode”。

与FTP服务器之间连接处理的重试指定

可进行与FTP服务器的连接处理的重试。
重试设定内容和动作与 “FTPGetFileList指令(P.2–1105)”的规格相同。请参阅该处。

取消指令执行

FTPRemoveFile指令执行中，可取消指令执行。
取消前，从FTP服务器删除文件的结果，保存在 “RemoveNum”和已删除文件结果RemoveFileResult[]中。
取消动作与 “FTPGetFileList指令(P.2–1105)” 的规格相同。请参阅该处。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

使用注意事项

- 本指令仅适用于NJ/NX系列 CPU单元的内置EtherNet/IP端口。
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 已删除文件数超过已删除文件结果RemoveFileResult[]的数组元素数时，不保存超出部分的结果。此时，“Error”不会变为TRUE。
- 关于超出255个字符的文件名，将从开头起的255个字符部分保存在 “RemoveFileResult[].Name”中。此时，“Error”不会变为TRUE。
- FTPGetFileList指令、FTPGetFile指令、FTPPutFile指令、FTPRemoveFile指令、FTPRemoveDir指令可同时执行的数量最多为3个。
- 使用通配符指定文件名，多个文件发生异常时，RemoveFileResult[]中保存的文件中，“RemoveFileResult[].TxError”的值为TRUE的起始文件的结果将保存在 “ErrorID”、“ErrorIDEx”中。
- 下列情况下，将发生删除异常，“RemoveFileResult[].RemoveError”的值变为TRUE。
  - FTP服务器目录中不存在 “FileName”指定的文件时
  - “FileName”指定文件的属性为只读时
  - “FileName”指定的文件为目录时
- 以下情况时会发生异常。“Error”变为TRUE。
  - 任一输入参数的值超过有效范围时
  - 在 “SvrDirName”中指定上1层的目录 “..”时
  - 在 “SvrDirName”中指定 “//”等错误路径时
  - FTP服务器中不存在 “SvrDirName”指定的目录时
  - “SvrDirName”指定的目录中存在1000个以上删除对象文件时
  - FTP服务器目录中不存在与 “FileName”指定的包含通配符的文件名相符的文件时
  - “FileName”指定文件的属性为只读时
  - 网络上不存在 “ConnectSvr”指定的FTP服务器，或者服务已停止时
  - 同时执行的 FTPGetFileList 指令、FTPGetFile 指令、FTPPutFile 指令、FTPRemoveFile 指令、FTPRemoveDir指令超过3个时。

- 扩展错误代码 “ErrorIDEx”在本指令中表示FTP服务器回复的FTP响应代码。典型的 “ErrorIDEx”的值、异常内容和处理方法如下所述。详情请确认FTP服务器的规格。“ErrorIDEx”在错误代码“ErrorID”的值为WORD#16#2407时输出。

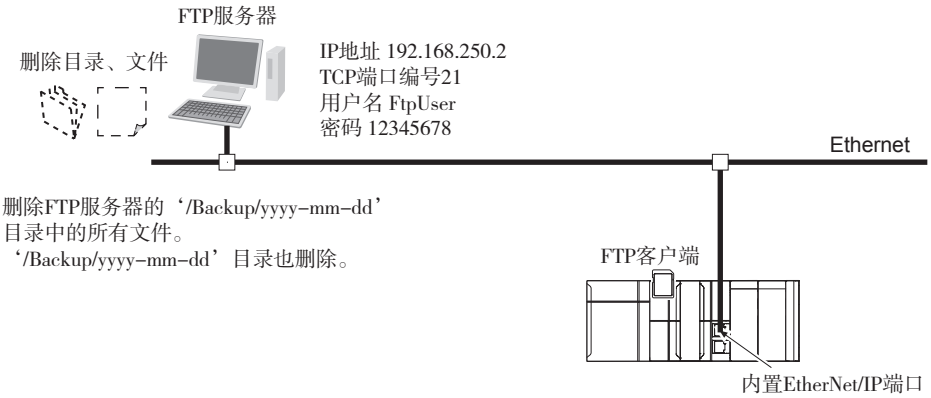
“ErrorIDEx”的值	异常内容	处理方法
16#000001A9	未能建立数据连接。	与互联网上的FTP服务器进行FTP通信时，请确认FTP的打开模式是否为Active。
16#000001AA	连接已关闭。停止了数据传输。	请确认与FTP服务器的连接。 请确认FTP服务器是否停止。
16#000001C2	未进行请求的文件操作。文件已被打开等情况时，无法使用文件。	请确认对象文件是否已被其他应用程序打开。
16#00000212	用户未能登录。	请确认FTP用户名或密码。
16#00000214	需要文件保存用账号。	请确认FTP用户的访问权。
16#00000226	找不到文件而无法访问等情况时，无法使用文件，因此未执行请求的文件操作。	请确认FTP服务器的目录中是否存在与指定文件名相符的文件。 请确认指定文件的访问权。
16#00000229	文件名不正确，未能执行。	请确认指定目录的访问权。

版本相关信息

本指令可用于CPU单元Ver.1.08 以上且Sysmac Studio Ver.1.09 以上。

示例程序

删除FTP服务器的 ‘/Backup/yyyy-mm-dd’目录中的所有文件。之后，‘/Backup/yyyy-mm-dd’目录也删除。



控制器与FTP服务器之间通过EtherNet/IP网络连接。为与FTP服务器连接而设定的参数值如下所示。

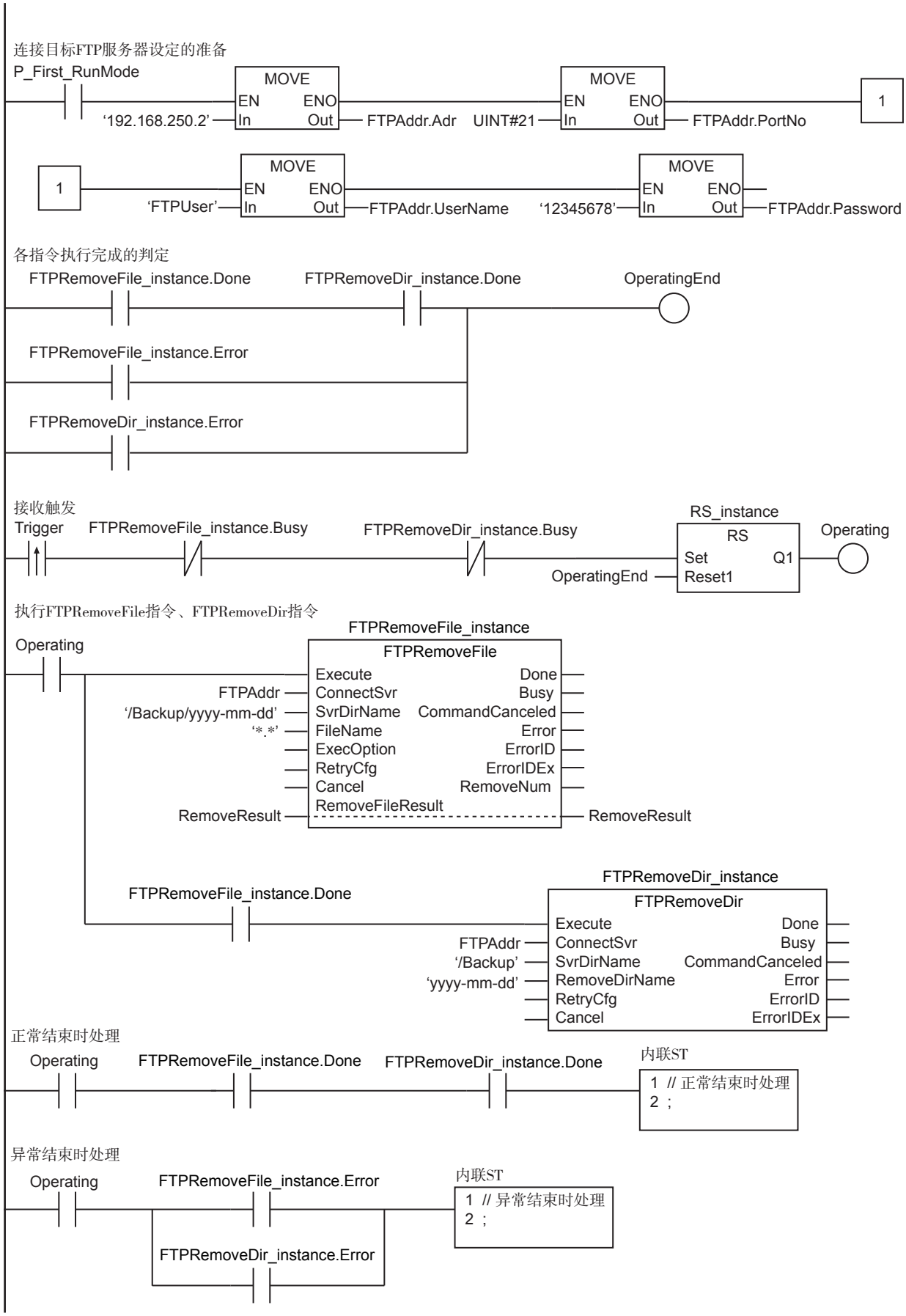
设定参数	值
IP地址	192.168.250.2
TCP端口编号	21
用户名	FtpUser
密码	12345678

处理步骤如下所述。

1. 使用FTPRemoveFile指令，删除FTP服务器的 ‘/Backup/yyyy-mm-dd’目录中的所有文件。将删除文件名指定通配符 ‘*.*’。
2. 使用FTPRemoveDir指令，删除FTP服务器的 ‘/Backup/yyyy-mm-dd’目录。
3. 所有处理正常结束后，执行正常结束时处理。发生异常时，执行异常结束时处理。

LD

内部变量	名称	数据类型	初始值	注释
	FTPRemoveFile_instance	FTPRemoveFile		FTPRemoveFile指令的实例
	FTPRemoveDir_instance	FTPRemoveDir		FTPRemoveDir指令的实例
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	连接目标FTP服务器设定
	RemoveResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	已删除文件结果
	RS_instance	RS		RS指令的实例
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中


连接目标FTP服务器设定的准备
P_First_RunMode
MOVE
EN ENO
'192.168.250.2' In Out FTPAddr.Adr
UINT#21 In Out FTPAddr.PortNo
1
'FTPUser' In Out FTPAddr.UserName
'12345678' In Out FTPAddr.Password
各指令执行完成的判定
FTPRemoveFile_instance.Done
FTPRemoveDir_instance.Done
OperatingEnd
FTPRemoveFile_instance.Error
FTPRemoveDir_instance.Error
接收触发
Trigger
FTPRemoveFile_instance.Busy
FTPRemoveDir_instance.Busy
RS_instance
RS
Set Q1
OperatingEnd Reset1
Operating
执行FTPRemoveFile指令、FTPRemoveDir指令
Operating
FTPRemoveFile_instance
FTPRemoveFile
Execute Done
FTPAddr ConnectSvr Busy
'/Backup/yyyy-mm-dd' SvrDirName CommandCanceled
'*.*' FileName Error
ExecOption ErrorID
RetryCfg ErrorIDEx
Cancel RemoveNum
RemoveResult RemoveFileResult RemoveResult
FTPRemoveFile_instance.Done
FTPRemoveDir_instance
FTPRemoveDir
Execute Done
FTPAddr ConnectSvr Busy
'/Backup' SvrDirName CommandCanceled
'yyyy-mm-dd' RemoveDirName Error
RetryCfg ErrorID
Cancel ErrorIDEx
正常结束时处理
Operating
FTPRemoveFile_instance.Done
FTPRemoveDir_instance.Done
内联ST
1 // 正常结束时处理
2 ;
异常结束时处理
Operating
FTPRemoveFile_instance.Error
FTPRemoveDir_instance.Error
内联ST
1 // 异常结束时处理
2 ;

ST

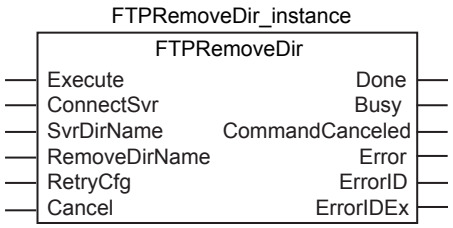
内部变量	名称	数据类型	初始值	注释
	R_TRIG_instance	R_TRIG		R_TRIG指令的实例
	UP_Q	BOOL	FALSE	触发输出
	FTPRemoveFile_instance	FTPRemoveFile		FTPRemoveFile指令的实例
	FTPRemoveDir_instance	FTPRemoveDir		FTPRemoveDir指令的实例
	DoFTPTrigger	BOOL	FALSE	FTPRemoveFile、FTPRemoveDir执行条件
	FTPAddr	_sFTP_CONNECT_SVR	(Adr := '', PortNo := 0, UserName := '', Password := '')	连接目标FTP服务器设定
	RemoveResult	ARRAY[0..0] OF _sFTP_FILE_RESULT	[(Name := '', TxError := False, RemoveError := False, Reserved := [4(16#0)])]	已删除文件结果
	Stage	UINT	0	指令执行步骤
	Trigger	BOOL	FALSE	执行条件

```
// 连接目标FTP服务器设定的准备
IF P_First_RunMode THEN
  FTPAddr.Adr      := '192.168.250.2';// 地址
  FTPAddr.PortNo   := UINT#21;      // 端口编号
  FTPAddr.UserName := 'FtpUser';    // 用户名
  FTPAddr.Password := '12345678';   // 密码
END_IF;

// 接收触发
R_TRIG_instance(Trigger, UP_Q);
IF ( (UP_Q = TRUE) AND (FTPRemoveFile_instance.Busy = FALSE) AND
     (FTPRemoveDir_instance.Busy = FALSE) ) THEN
  DoFTPTrigger := TRUE;
  Stage := INT#1;
  FTPRemoveFile_instance( // 实例初始化
    Execute          := FALSE,
    ConnectSvr       := FTPAddr,
    SvrDirName       := '/Backup/yyyy-mm-dd',
    FileName         := '*.*',
    RemoveFileResult := RemoveResult);
  FTPRemoveDir_instance( // 实例初始化
    Execute          := FALSE,
    ConnectSvr       := FTPAddr,
    SvrDirName       := '/Backup',
    RemoveDirName    := 'yyyy-mm-dd');
END_IF;

IF (DoFTPTrigger = TRUE) THEN
  CASE Stage OF
    1 : // 执行FTPRemoveFile指令
      FTPRemoveFile_instance(
        Execute          := TRUE,          // 启动
        ConnectSvr       := FTPAddr,       // 连接目标FTP服务器
        SvrDirName       := '/Backup/yyyy-mm-dd', // FTP服务器目录名称
        FileName         := '*.*',         // 文件名
        RemoveFileResult := RemoveResult);  // 删除文件结果
```

```
      IF (FTPRemoveFile_instance.Done = TRUE) THEN
        Stage := INT#2; // 至下一个
      ELSIF (FTPRemoveFile_instance.Error = TRUE) THEN
        Stage := INT#10; // 异常结束
      END_IF;
    2 : // 执行FTPRemoveDir指令
      FTPRemoveDir_instance(
        Execute          := TRUE,       // 启动
        ConnectSvr       := FTPAddr,    // 连接目标FTP服务器
        SvrDirName       := '/Backup',  // FTP服务器目录名称
        RemoveDirName := 'yyyy-mm-dd') ;// 删除目录名
      IF (FTPRemoveDir_instance.Done = TRUE) THEN
        Stage:=INT#0; // 正常结束
      ELSIF (FTPRemoveDir_instance.Error = TRUE) THEN
        Stage:=INT#20; // 异常结束
      END_IF;
    0 : // 正常结束处理
      DoFTPTrigger:=FALSE;
      Trigger    :=FALSE;
    ELSE // 异常结束处理
      DoFTPTrigger:=FALSE;
      Trigger    :=FALSE;
  END_CASE;
END_IF;
```

FTPRemoveDir

删除FTP服务器的目录。

指令	名称	FB/FUN	图形表现	ST表现
FTPRemoveDir	FTP服务器的目录删除	FB		FTPRemoveDir_instance(Execute, ConnectSvr, SvrDirName, RemoveDirName, Cancel, RetryCfg, Done, Busy, CommandCanceled, Error, ErrorID, ErrorIDEx) ;

变量

	名称	输入/输出	内容	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	输入	连接目标FTP服务器的各种设定参数	–	–	*1
SvrDirName	FTP服务器目录名称		待删除目录所在FTP服务器的目录名称	最大256字节(255个半角英数字字符+结尾NULL字符)*2		""*3
Remove DirName	删除目录名		待删除的目录名	最大256字节(255个半角英数字字符+结尾NULL字符)		*1
RetryCfg	执行重试设定		指令执行重试设定	–		–
Cancel	取消		TRUE：取消指令执行 FALSE：不取消指令执行	遵从数据类型		FALSE
Command Canceled	取消完成	输出	TRUE：取消完成 FALSE：取消未完成	–	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。
2 FTP服务器目录名称，不能使用下列字符。“”、“?”、“<”、“>”、“|”、“"”(双引号)
*3 FTP服务器的登录时的主目录。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
ConnectSvr	结构体_sFTP_CONNECT_SVR 详情参阅功能说明																			
SvrDirName																				○
RemoveDirName																				○
RetryCfg	结构体_sFTP_RETRY_CFG 详情参阅功能说明																			
Cancel	○																			
CommandCanceled	○																			

功能

删除连接目标 FTP 服务器地址 “ConnectSvr”的待删除目录所在目录 “SvrDirName”中的指定目录 “RemoveDirName”。

在指令正常结束 “Done”的值变为 TRUE 时，指定目录删除完成。目录删除失败时，异常结束 “Error”的值变为TRUE。

“ConnectSvr”的数据类型为结构体_sFTP_CONNECT_SVR。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ConnectSvr	连接目标FTP服务器设定	连接目标FTP服务器的各种设定参数	_sFTP_CONNECT_SVR	–	–	–
Adr	地址	IP地址或主机名称*1	STRING	1 ~ 200字节*2	–	–
PortNo	端口编号	FTP服务器的控制连接的TCP端口编号	UINT	0 ~ 65535*3		
UserName	用户名	FTP服务器的用户名	STRING	最大33字节 *2*4*5		
Password	密码	FTP服务器的密码	STRING	最大33字节 *2*4*5		

*1 指定主机名称时，必须另行设定DNS或Hosts。
*2 可使用的字符为半角的 “A–Z”、“a–z”、“0–9”、“–”(连字符)、“.”(句号)、“_”(下划线)。
*3 为0时，TCP端口编号为21。
*4 字节数含结尾NULL字符。
*5 CPU单元Ver.1.08请指定至少1个字符的字符串。指定仅结尾NULL字符的字符串时，将发生异常。

与FTP服务器之间连接处理的重试指定

可进行与FTP服务器的连接处理的重试。
重试动作与 “FTPGetFileList指令(P.2–1105)”的规格相同。请参阅该处。

取消指令执行

FTPRemoveDir指令执行中，可取消指令执行。
取消动作与 “FTPGetFileList指令(P.2–1105)”的规格相同。请参阅该处。

相关的系统定义变量

变量名称	名称	数据类型	内容
_EIP_EtnOnlineSta *1	在线	BOOL	表示可使用内置EtherNet/IP端口的通信功能。 TRUE ：可通信 FALSE：无法通信
_EIP1_EtnOnlineSta *2 _EIP2_EtnOnlineSta *3			
_EIPIn1_EtnOnlineSta *4			

*1 使用NJ系列CPU单元时的变量名称。
*2 使用NX系列CPU单元的端口1或NY系列控制器时的变量名称。
*3 使用NX系列CPU单元的端口2时的变量名称。
*4 使用NY系列控制器的内部通信端口时的变量名称。

使用注意事项

- 本指令仅适用于NJ/NX系列 CPU单元的内置EtherNet/IP端口。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 利用 “Cancel”取消执行本指令时，可能会因 “Cancel”的时间不同，导致FTP服务器的目录被删除。请确认FTP服务器的目录。
- FTPGetFileList指令、FTPGetFile指令、FTPPutFile指令、FTPRemoveFile指令、FTPRemoveDir指令可同时执行的数量最多为3个。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 任一输入参数的值超过有效范围时
  - FTP服务器中不存在 “SvrDirName”指定的目录时
  - 在 “SvrDirName”、“RemoveDirName”中指定上1层的目录 “..”时
  - 在 “SvrDirName”、“RemoveDirName”中指定 “//”等错误路径时
  - FTP服务器中不存在 “RemoveDirName”指定的目录时
  - “RemoveDirName”指定的目录中存在文件或子目录时
  - “RemoveDirName”指定的目录的属性为只读时
  - 网络上不存在 “ConnectSvr”指定的FTP服务器，或者服务已停止时
  - 同时执行的 FTPGetFileList 指令、FTPGetFile 指令、FTPPutFile 指令、FTPRemoveFile 指令、FTPRemoveDir指令超过3个时。

- 扩展错误代码 “ErrorIDEx”在本指令中表示FTP服务器回复的FTP响应代码。典型的 “ErrorIDEx”的值、异常内容和处理方法如下所述。详情请确认FTP服务器的规格。“ErrorIDEx”在错误代码“ErrorID”的值为WORD#16#2407时输出。

“ErrorIDEx”的值	异常内容	处理方法
16#000001A9	未能建立数据连接。	与互联网上的FTP服务器进行FTP通信时，请确认FTP的打开模式是否为Active。
16#000001AA	连接已关闭。停止了数据传输。	请确认与FTP服务器的连接。 请确认FTP服务器是否停止。
16#000001C2	未进行请求的文件操作。文件已被打开等情况时，无法使用文件。	请确认对象文件是否已被其他应用程序打开。
16#00000212	用户未能登录。	请确认FTP用户名或密码。
16#00000214	需要文件保存用账号。	请确认FTP用户的访问权。
16#00000226	找不到文件而无法访问等情况时，无法使用文件，因此未执行请求的文件操作。	请确认FTP服务器的目录中是否存在与指定文件名相符的文件。 请确认指定文件的访问权。
16#00000229	文件名不正确，未能执行。	请确认指定目录的访问权。

版本相关信息

本指令可用于CPU单元Ver.1.08 以上且Sysmac Studio Ver.1.09 以上。

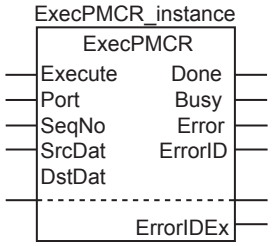
示例程序

请参阅 “FTPRemoveFile指令(P.2–1138)”的示例程序。

串行通信指令

ExecPMCR

请求执行串行通信单元中登录的收发时序(协议数据)。

指令	名称	FB/FUN	图形表现	ST表现
ExecPMCR	协议宏	FB		ExecPMCR_instance(Execute, Port, SeqNo, SrcDat, DstDat, Done, Busy, Error, ErrorID, ErrorIDEx);

使用注意事项

本指令无法在NX系列 CPU单元中使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Port	指定对象端口	输入	指定对象端口	–	–	–
SeqNo	收发时序编号		收发时序编号	0 ~ 999		0
SrcDat[]数组	发送数据数组		发送数据数组	遵从数据类型		(*)
DstDat[]数组	接收数据数组	输入输出	接收数据数组	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Port	结构体_sPORT 详情参阅功能说明																			
SeqNo							○													
SrcDat[]数组			○																	
DstDat[]数组			○																	

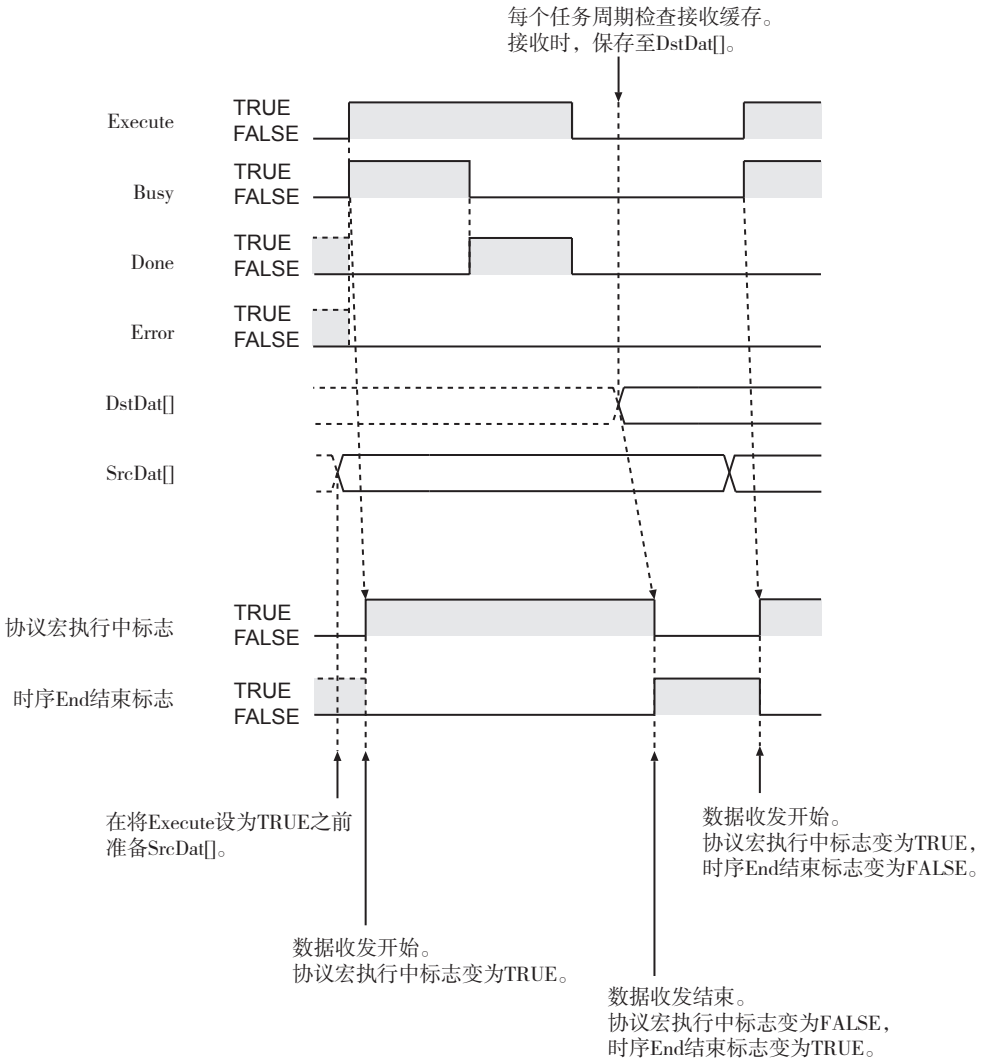
功能

委托指定的对象端口 “Port”执行由收发时序编号 “SeqNo”指定的时序。
发送时，从发送数据数组SrcDat[]的第2个元素(SrcDat[1])起发送。待发送的数组元素数由SrcDat[0]指定。
成功接收数据后，该接收数据保存至接收数据数组DstDat[]的第2个元素(DstDat[1])之后。接收数据元素数保存至DstDat[0]中。
接收失败时，仅事先保存至DstDat[0]中的元素数会保持执行本指令前DstDat[1]以后的值。

对象端口指定 “Port”的数据类型为结构体_sPORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Port	指定对象端口	指定对象端口	_sPORT	–	–	–
UnitNo	单元编号	串行通信单元的单元编号	_eUnitNo	_CBU_No00 ~ _CBU_No15	–	_CBU_No00
PhysicPortNo	串行端口编号	串行通信单元的串行端口编号	USINT	1,2		1

时序图如下所示。“Done”的值即使为TRUE，通信仍将继续直至最后。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Port_numUsingPort	使用端口数	USINT	当前使用中的端口数
_Port_isAvailable	网络通信指令可执行标志	BOOL	TRUE ：有可使用端口 FALSE：无可使用端口
_CJB_SCU##P1ChgSta, _CJB_SCU##P2ChgSta(*)	串行通信单元##单元端口1/2用串行端口设定变更中标志	BOOL	TRUE ：串行端口的设定变更中 FALSE：串行端口的设定未变更

* #中填入串行通信单元的单元编号。

相关的准用户定义变量

变量名称	名称	数据类型	内容
P#_PmrExecSta(*)	协议宏执行中标志	BOOL	TRUE：执行中 FALSE：非执行中或执行失败
P#_PmrSeqEndSta(*)	时序End结束标志	BOOL	TRUE：时序End结束 FALSE：时序End未结束
P#_PmrSeqAbtSta(*)	时序Abort结束标志	BOOL	TRUE：时序Abort结束 FALSE：时序Abort未结束

* #中填入串行通信单元的端口No.。

参考

协议宏功能的详情请参阅 “SYSMAC CX-Protocol 操作手册(SBCA-307)”。

使用注意事项

- 本指令用于执行启动协议宏。协议宏执行的状态请通过协议宏执行中标志的系统定义变量“P#PmrExecSta”进行确认。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- DstDat[]必须在CJ单元存储器的地址中指定AT。
- 直接指定、链接通道指定时，SrcDat[0]及DstDat[0]的值请设为0。设定为其它常数和变量时会发生异常，不执行本指令。
- DstDat[0]的值为0或1时，若接收失败，DstDat[]的元素值会全部变为0。
- 本指令仅可在端口可使用时执行。因此，在本指令的输入条件中，请将网络通信指令可执行标志的系统定义变量 “_Port_isAvailable”作为a触点插入。
- 本指令在 “Busy”的值为TRUE时无法执行。因此，作为本指令的输入条件，请将 “Busy”作为b触点插入。
- 协议宏执行中标志的准用户定义变量 “P#_PmrExecSta”在本指令执行开始时变为 TRUE，在收发时序执行结束且接收数据完全保存至DstDat[]时，变为FALSE。其间，无法对同一串行端口执行本指令。因此，作为本指令的输入条件，请将 “P#_PmrExecSta”作为b触点插入。
- 在ST程序中记述本指令时，执行本指令过程中，请设为每个任务周期都对本指令进行控制。否则，会出现无法正常处理的情况。

- 以下情况时会发生异常。“Error”变为TRUE。
  - 串行通信模式非协议宏模式的情况下，执行本指令时。
  - “_Port_isAvailable”的值为FALSE时。
  - “SeqNo”的值超过有效范围时。
  - “SeqNo”的值未登录至串行通信单元时。
  - “Port.UnitNo”或 “Port.PhysicPortNo”的值超过有效范围时。
  - 指定的单元编号中未安装CJ串行通信单元时。
  - SrcDat[0]的值超过SrcDat[]的大小时。
  - DstDat[0]的值超过DstDat[]的大小时。
  - SrcDat[0]或DstDat[0]的值超过250CH时。
  - 通信失败时。
  - DstDat[]未在CJ单元存储器的地址中指定AT时。
- 扩展错误代码 “ErrorIDEx”在本指令中具有通信响应代码的含义。该值与异常内容如下所述。
  “ErrorIDEx”在错误代码 “ErrorID”的值为WORD#16#0800时输出。

值	异常内容	处理方法
16#00000001	通信服务中断。	· 请确认数据链接启动状态。 · 请确认第三节点的传送目标区域的容量。
16#00000101	本机节点未加入网络。	请将本机节点加入网络。
16#00000102	发生了令牌超时。	请在最大节点地址的范围内设定本机节点。
16#00000103	发生重新发送超限。	执行节点间测试发生异常时，请确认使用环境。
16#00000104	发送许可帧数超限。	请确认网络内执行的事件情况，减少1个任务周期内的事件数。或者，请增加发送许可帧数。
16#00000105	本机节点的节点地址超过设定范围。	请对串行通信单元的旋转开关进行正确设定。
16#00000106	本机节点的节点地址在网络内重复。	请变更重复的节点地址。
16#00000201	对象节点未加入网络。	请将对象节点加入网络。
16#00000202	发送目标中指定的单元地址的单元不存在。	请对发送目标网络地址的单元地址进行正确设定。
16#00000203	第三节点未加入网络。	· 请确认作为第三节点的单元地址。 · 第三节点请只指定1个节点。
16#00000204	对象节点为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在对象节点上。
16#00000205	发生响应超时。	请确认通信参数的设定。
16#00000206	传送通道存在异常。	· 请重试。 · 异常频发时，请确认干扰情况。
16#00000301	通信控制器发生异常。	请在参阅相应单元的用户手册后，采取适当处理。
16#00000302	对象节点的CPU单元异常。	请参阅手册排除对象节点的CPU单元异常。
16#00000303	相应控制器存在异常，未返回响应。	请在确认网络的通信情况后，重启相应控制器。即使这样仍发生异常时，请更换相应控制器。
16#00000304	单元编号设定不正确。	请对串行通信单元的旋转开关进行正确设定。
16#00000401	发送的指令不被支持。	请对指令数组的内容进行正确设定。
16#00000402	单元的机型或版本不被支持。	请确认单元的机型和版本。
16#00000501	对象地址设定异常。	请在路由表中设定目标地址。
16#00000502	未登录路由表。	在发送源节点、目标节点、中继节点设定路由表。
16#00000503	路由表异常。	请正确设定路由表。
16#00000504	发生中继次数超限。	请重组网络或调整路由表，将指令的使用范围设定在3层以内。
16#00001001	指令长度超过最大指令长度。	请对指令数组的内容进行正确设定。
16#00001002	指令长度不足最小指令长度。	请对指令数组的内容进行正确设定。

值	异常内容	处理方法
16#00001003	指令指定的写入元素数与实际写入的数据数不一致。	请使写入元素数与实际的写入数据数一致。
16#00001004	指令格式异常。	请对指令数组的内容进行正确设定。
16#00001005	报头异常。	请正确设定路由表。
16#00001101	无区域种类。	请在参照指令的变量、参数种类代码的基础上，设定相应代码。
16#00001102	访问大小异常。	请正确设定变量、参数的访问大小。
16#00001103	指定了超出范围的地址。	请指定可处理范围内的地址。
16#00001104	超出地址范围。	· 请指定可处理范围内的地址。 · 请正确设定数据链接表。
16#00001106	指定了未登录的收发时序No.。	请修改收发时序No.或通过CX-Protocol增加时序。
16#00001109	发生了相关关系异常。	· 请正确设定指令数据的大小关系。 · 请正确设定数据链接表。
16#0000110A	数据重复。	· 请中断执行中的处理或等待至结束后再执行指令。 · 请正确设定数据链接表。
16#0000110B	响应超过最大响应长度。	请对指令数组内的元素数进行正确设定。
16#0000110C	其他参数异常	请对指令数组的内容进行正确设定。
16#00002002	保护中。	请在解除保护后，重新执行指令。
16#00002003	无登录表。	请正确设定表格。
16#00002004	不存在与检索数据一致的数据。	请正确设定检索数据。
16#00002005	无相应的程序编号。	请设定有效的程序编号。
16#00002006	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002007	发生了核查异常。	· 请确认存储器的内容后，改写成正确的数据。 · 请确认文件的内容。
16#00002101	为只读区域，因此无法存取。	请在解除写保护后，重新执行指令。
16#00002102	保护中或无法写入数据链接表。	· 请在解除写保护后，重新执行指令。 · 请在数据链接表中设定系统设定。
16#00002103	无法登录。	· 请在删除不用的文件后再创建文件或准备新文件用的存储器。 · 请在关闭已打开的文件后，重新执行指令。
16#00002105	无相应的程序编号。	请设定有效的程序编号。
16#00002106	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002107	存在相同的文件名。	请在变更要写入文件的文件名后，重新执行指令。
16#00002108	变更会导致异常，因此无法变更。	请变更设定。
16#00002201	协议宏已在执行中，因此无法动作。	请将协议宏执行中标志设为b触点。
16#00002202	动作模式异常。	请确认动作模式。
16#00002203	指令的动作模式不同(程序模式)。	请确认控制器的动作模式。
16#00002204	指令的动作模式不同(调试模式)。	请确认控制器的动作模式。
16#00002205	指令的动作模式不同(监控模式)。	请确认控制器的动作模式。
16#00002206	指令的动作模式不同(运行模式)。	请确认控制器的动作模式。
16#00002207	指定节点非管理站。	请确认网络的管理站节点。
16#00002208	指令的动作模式不同。	请确认步活性状态。
16#00002211	单元为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在相应单元上。
16#00002301	无文件装置。	请安装介质。或者，请执行EM格式化。
16#00002302	无文件存储器。	请确认文件存储器的安装。
16#00002303	未内置时钟。	请确认机型的规格。
16#00002401	协议宏数据的SUM值异常或正在传送数据。	请通过CX-Protocol重新传送协议宏数据。
16#00002502	处理的对象存储器存在异常。	请将正确数据重新传送至对象存储器。

值	异常内容	处理方法
16#00002503	登录的I/O单元构成与实际的单元构成不同。	请确认I/O单元构成。
16#00002504	登录的I/O点数或远程I/O点数超过最大值。	请对I/O点数、远程I/O点数进行正确设定。
16#00002505	CPU单元与CPU高功能单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002506	机架No.、单元编号、I/O地址中的某一设定重复。	请对重复的内容进行重新设定。
16#00002507	CPU单元与I/O单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002509	SYSMAC BUS/2数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250A	CPU高功能单元的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250D	通道设定重复。	请对I/O通道进行正确设定。
16#0000250F	存储器异常。	· 内部存储器时，请写入正确数据后重新执行指令。 · 存储卡及EM文件存储器时，请执行扩展存储器的格式化指令。 · 进行上述处理后仍无法解除异常时，请更换存储器。
16#00002510	末站的设定异常。	请对末站进行正确设定。
16#00002601	保护解除中。	无需解除保护。
16#00002602	密码不一致。	请指定正确的密码。
16#00002604	保护中。	· 请在解除写保护后，重新执行指令。 · 请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002605	服务执行中。	请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002606	服务停止中。	请根据需要执行相应服务。
16#00002607	无执行权。	· 请通过数据链接参加节点执行。 · 重启后仍发生异常时，请更换控制器。
16#00002608	未设定环境。	请进行必要的设定。
16#00002609	未设定必要项目。	请设定必要项目。
16#0000260A	指定编号已定义。	请变更成未登录的动作、过渡编号后，重新执行指令。
16#0000260B	无法解除异常。	请解除造成异常的原因后，执行异常解除指令。
16#00003001	无访问权。	请等待至访问权释放后，重新执行指令。
16#00004001	服务被中断。	请在解除服务被中断的原因后，重新执行指令。

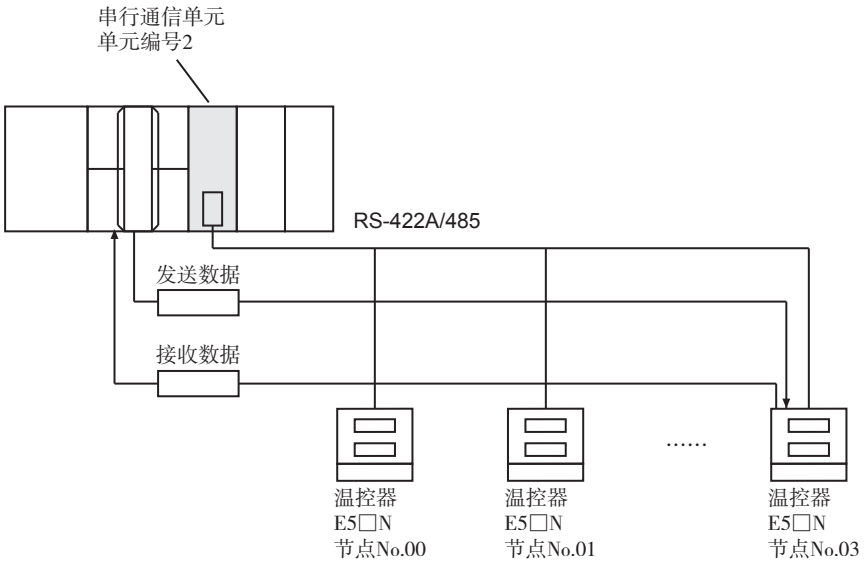
(注) 除上表值以外，终止符的位6、7、15的值可能会变为TRUE。位6、7的值为TRUE时，表示发送目标的CPU单元发生了异常。位15的值为TRUE时，表示网络中继时发生了异常。

示例程序

使用CJ系列的串行通信单元，与欧姆龙制温控器间进行数据收发。收发协议通过CompoWay/F主站的时序No.610(变量区域读取)，读取温控器的当前值。
由控制器发送发送数据数组SendData[]的内容。
从温控器收到的数据保存至接收数据数组RecvData[]。

主要通信规格如下所述。

项目	内容
使用单元	串行通信单元
单元编号	2
端口编号	1(RS422/485)
收发时序编号	610(变量区域读取)
对象节点No.	3
读取内容	当前值



时序No.610(变量区域读取)收发数据的分配内容如下所述。

发送数据　WORD型数组

SendData[0]	发送数据通道数	
SendData[1]	空白	节点No.
SendData[2]	变量种类	
SendData[3]	读取开始地址	
SendData[4]	元素数	

接收数据　WORD型数组

RecvData[0]	接收数据通道数
RecvData[1]	响应代码
RecvData[2]	接收数据
RecvData[3]	

本示例中，发送数据SendData[]和接收数据RecvData[]的内容如下所述。

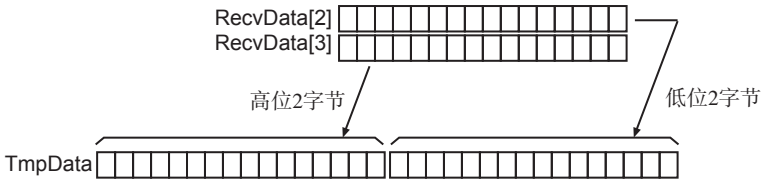
发送数据　WORD型数组

变量	项目	本示例的内容	值
SendData[0]	发送数据通道数	发送SendData[0] ~ SendData[4]的5通道(字)。	WORD#16#0005
SendData[1]	节点No.	作为节点No.3。	WORD#16#0003
SendData[2]	变量种类+读取开始地址的高位1字节	当前值读取的变量种类为BYTE#16#C0，读取开始地址为WORD#16#00。	WORD#16#C000
SendData[3]	读取开始地址的低位1字节+固定值BYTE#16#00		WORD#16#0000
SendData[4]	元素数	元素数为1。	WORD#16#0001

接收数据 WORD型数组

变量	项目	本示例的内容	值
RecvData[0]	接收数据通道数	接收RecvData[0] ~ RecvData[3]的4通道(字)。	WORD#16#0004
RecvData[1]	响应代码	正常结束时恢复为WORD#16#0000。	
RecvData[2]	接收数据	恢复为温控器当前值的低位2字节。	
RecvData[3]		恢复为温控器当前值的高位2字节。	

接收成功时，将温控器当前值的低位2字节(RecvData[2])与高位2字节(RecvData[3])进行组合后，代入TmpData。



全局变量的定义

全局变量

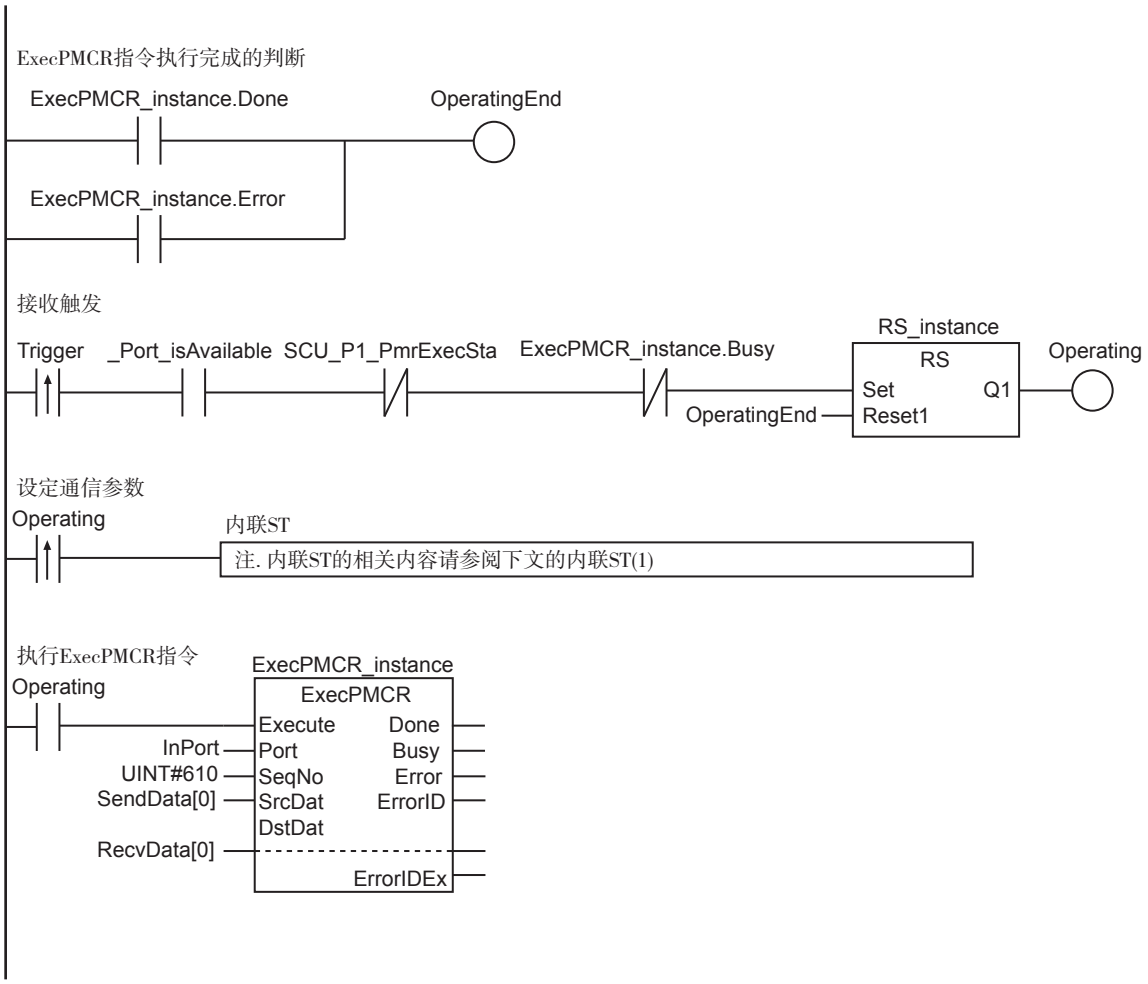
名称	数据类型	分配对象*1	注释
SCU_P1_PmrSeqEndSta	BOOL	IOBus://rack#0/slot#0/P1_PmrSta/P1_PmrSeqEndSta	时序End结束标志
SCU_P1_PmrSeqAbtSta	BOOL	IOBus://rack#0/slot#0/P1_PmrSta/P1_PmrSeqAbtSta	时序Abort结束标志
SCU_P1_PmrExecSta	BOOL	IOBus://rack#0/slot#0/P1_PmrSta/P1_PmrExecSta	协议宏执行中标志

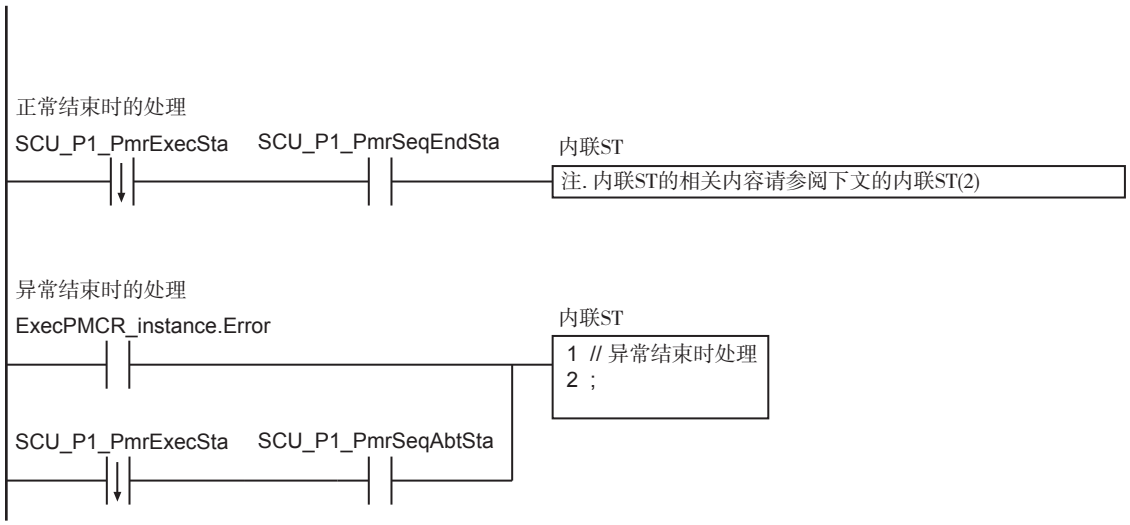
*1 是将串行通信单元安装于机架编号0的插槽编号0上时的分配对象。

LD

内部变量	名称	数据类型	初始值	分配对象	保持	注释
	OperatingEnd	BOOL	FALSE		☐	处理结束
	Trigger	BOOL	FALSE		☐	执行条件
	Operating	BOOL	FALSE		☐	处理中
	InPort	_sPORT	(UnitNo:=_CBU_No00, PhysicPortNo:=0)		☐	指定端口
	SendData	ARRAY[0..4] OF WORD	[5(16#0)]		☐	发送数据
	RecvData	ARRAY[0..3] OF WORD	[4(16#0)]	%D200	☑	接收数据
	TmpData	DINT	0		☐	当前值
	RS_instance	RS			☐	
	ExecPMCR_instance	ExecPMCR			☐	

外部变量	名称	数据类型	注释
	SCU_P1_PmrSeqEndSta	BOOL	时序End结束标志
	SCU_P1_PmrSeqAbtSta	BOOL	时序Abort结束标志
	SCU_P1_PmrExecSta	BOOL	协议宏执行中标志
	_Port_isAvailable	BOOL	网络通信指令可执行标志





- 内联ST(1)的内容

```
InPort.UnitNo          :=_CBU_No02;  // 串行通信单元　单元编号2
InPort.PhysicPortNo    :=USINT#1;    // 端口编号1
SendData[0]            :=WORD#16#0005;
SendData[1]            :=WORD#16#0003;
SendData[2]            :=WORD#16#C000;
SendData[3]            :=WORD#16#0000;
SendData[4]            :=WORD#16#0001;
RecvData[0]            :=WORD#16#0004;
```

- 内联ST(2)的内容

```
// 正常结束时处理
TmpData:=DWORD_TO_DINT(SHL(WORD_TO_DWORD(
      RecvData[3]), 16) OR WORD_TO_DWORD(RecvData[2]) );
```

ST

内部变量	名称	数据类型	初始值	分配对象	保持	注释
	State	INT	0		☐	变化状态
	Trigger	BOOL	FALSE		☐	执行条件
	LastTrigger	BOOL	FALSE		☐	上个任务周期的Trigger的值
	InPort	_sPORT	(UnitNo:=_CBU_No00, PhysicPortNo:=0)		☐	指定端口
	SendData	ARRAY[0..4] OF WORD	[5(16#0)]		☐	发送数据
	RecvData	ARRAY[0..3] OF WORD	[4(16#0)]	%D200	☑	接收数据
	End_ExecPMCR	BOOL	FALSE		☐	ExecPMCR指令执行结束
	TmpData	DINT	0		☐	当前值
	RS_instance	RS			☐	
	ExecPMCR_instance	ExecPMCR			☐	
	F_TRIG_instance	F_TRIG			☐	

外部变量	名称	数据类型	注释
	SCU_P1_PmrSeqEndSta	BOOL	时序End结束标志
	SCU_P1_PmrSeqAbtSta	BOOL	时序Abort结束标志
	SCU_P1_PmrExecSta	BOOL	协议宏执行中标志
	_Port_isAvailable	BOOL	网络通信指令可执行标志

```
// 接收触发
IF (State=INT#0) THEN
  IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Port_isAvailable=TRUE) AND (SCU_P1_PmrExecSta<>TRUE)
      AND (ExecPMCR_instance.Busy<>TRUE) ) THEN
    State:=INT#1;
  END_IF;
END_IF;
LastTrigger:=Trigger;

// 设定通信参数和ExecPMCR指令初始化
IF (State=INT#1) THEN
  InPort.UnitNo          :=_CBU_No02;  // 串行通信单元  单元编号2
  InPort.PhysicPortNo    :=USINT#1;    // 端口编号1
  SendData[0]            :=WORD#16#0005;
  SendData[1]            :=WORD#16#0003;
  SendData[2]            :=WORD#16#C000;
  SendData[3]            :=WORD#16#0000;
  SendData[4]            :=WORD#16#0001;
  RecvData[0]            :=WORD#16#0004;
  ExecPMCR_instance(
    Execute :=FALSE,      // ExecPMCR指令初始化
    SrcDat  :=SendData[0], // 虚拟
    DstDat  :=RecvData[0]);
  State:=INT#2;
END_IF;
```

```
// 执行ExecPMCR指令
IF (State=INT#2) THEN
   ExecPMCR_instance(
   Execute :=TRUE,
   Port      :=InPort,
   SeqNo   :=UINT#610,
   SrcDat  :=SendData[0],
   DstDat  :=RecvData[0]);

   F_TRIG_instance(SCU_P1_PmrExecSta, End_ExecPMCR);

   IF (End_ExecPMCR=TRUE) THEN
      End_ExecPMCR:=FALSE;
      State:=INT#3;
   END_IF;

   IF (ExecPMCR_instance.Error=TRUE) THEN
      State:=INT#5;
   END_IF;
END_IF;

// 确认ExecPMCR指令执行结束
IF (State=INT#3) THEN
   IF (SCU_P1_PmrSeqEndSta=TRUE) THEN
      State:=INT#4;
   END_IF;
   IF (SCU_P1_PmrSeqAbtSta=TRUE) THEN
      State:=INT#5;
   END_IF;
END_IF;

IF (State=INT#4) THEN
   // 正常结束时处理
   TmpData:=DWORD_TO_DINT(SHL(WORD_TO_DWORD(RecvData[3]), 16)
         OR WORD_TO_DWORD(RecvData[2]));
   State:=INT#0;
END_IF;

IF (State=INT#5) THEN
   // 异常结束时处理
   State:=INT#0;
END_IF;
```

SerialSend

无协议从串行通信单元的串行端口发送数据。

指令	名称	FB/FUN	图形表现	ST表现
SerialSend	串行通信单元 串行端口输出	FB	SerialSend_instance SerialSend Execute Done Port Busy SrcDat Error SendSize ErrorID ErrorIDEx	SerialSend_instance(Execute, Port, SrcDat, SendSize, Done, Busy, Error, ErrorID, ErrorIDEx);

使用注意事项

本指令无法在NX系列 CPU单元中使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Port	指定对象端口	输入	指定对象端口	–	–	–
SrcDat[]数组	发送数据数组		发送数据数组	遵从数据类型		(*)
SendSize	发送数据大小		SrcDat[]中待发送数据的大小	0 ~ 256	字节	1

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Port	结构体_sPORT 详情参阅功能说明																			
SrcDat[]数组		○																		
SendSize							○													

功能

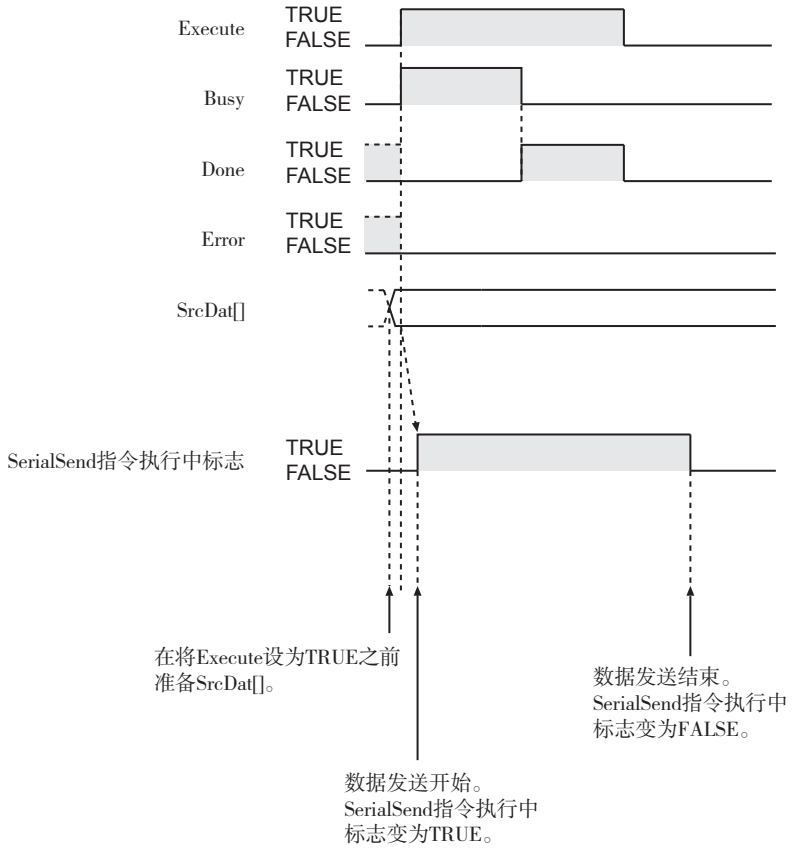
无协议从串行通信单元中 “Port”指定端口发送数据。
待发送的数据为发送数据数组SrcDat[]的内容。待发送数据的大小由发送数据大小 “SendSize”指定。

发送数据中加上起始符和结束符时，通过高功能单元的分配DM设定区域进行设定。
附加起始符、结束符时的可发送字节数最大为259(起始符1字节 + 结束符2字节(指定CR+LF时) + 发送数据256字节)。

对象端口指定 “Port”的数据类型为结构体_sPORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Port	指定对象端口	指定对象端口	_sPORT	–	–	–
UnitNo	单元编号	串行通信单元的单元编号	_eUnitNo	_CBU_No00 ~ _CBU_No15	–	_CBU_No00
PhysicPortNo	串行端口编号	串行通信单元的串行端口编号	USINT	1,2		1

时序图如下所示。“Done”的值即使为TRUE，通信仍将继续直至最后。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Port_numUsingPort	使用端口数	USINT	当前使用中的端口数
_Port_isAvailable	网络通信指令可执行标志	BOOL	TRUE ：有可使用端口 FALSE：无可使用端口

相关的准用户定义变量

变量名称	名称	数据类型	内容
P#_NopSerialSendExecSta(*)	SerialSend指令执行中标志	BOOL	TRUE：执行中 FALSE：非执行中
P#_NopStartCodeYNCfg(*)	有无无协议起始符	BOOL	TRUE：有 FALSE：无
P#_NopEndCodeYNCfg(*)	有无无协议结束符	BOOL	TRUE：有 FALSE：无
P#_NopCRLFCfg(*)	指定无协议CRLF	BOOL	TRUE：指定 FALSE：不指定
P#_NopStartCodeCfg(*)	无协议起始符	USINT	16#00 ~ 16#FF
P#_NopEndCodeCfg(*)	无协议结束符	USINT	16#00 ~ 16#FF

* #中填入串行通信单元的端口No.。

参考

无协议通信的详情请参阅下列手册。

- 🕮“CJ系列 串行通信单元 用户手册 NJ系列连接篇(SBCD-354)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 🕮“本章说明(P.2-3)”。
- 本指令仅可用于设定为无协议模式的串行通信单元的串行端口。
- “SendSize”的值为0时，不发送。此时执行指令时，“Done”的值为TRUE。
- 附加起始符、结束符时，“SendSize”中请设定不含起始符、结束符的值。
- 本指令仅可在端口可使用时执行。因此，在本指令的输入条件中，请将网络通信指令可执行标志的系统定义变量 “_Port_isAvailable”作为a触点插入。
- 本指令在 “Busy”的值为TRUE时无法执行。因此，作为本指令的输入条件，请将 “Busy”作为b触点插入。
- 本指令在SerialSend指令执行中标志的准用户定义变量 “P#NopSerialSendExecSta”的值为TRUE时，无法执行。因此，作为本指令的输入条件，
  请将 “P#NopSerialSendExecSta”作为b触点插入。
- 在ST程序中记述本指令时，执行本指令过程中，请设为每个任务周期都对本指令进行控制。否则，会出现无法正常处理的情况。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 串行通信模式非无协议的情况下，执行本指令时。
  - “_Port_isAvailable”的值为FALSE时。
  - “Port.UnitNo”或 “Port.PhysicPortNo”的值超过有效范围时。
  - 指定的单元编号中未安装CJ串行通信单元时。
  - “SendSize”的值超过有效范围时。
  - “SendSize”的值超过SrcDat[]的大小时。
  - 通信失败时。
  - 单元重启中执行本指令时。

• 扩展错误代码 “ErrorIDEx” 在本指令中具有通信响应代码的含义。该值与异常内容如下所述。“ErrorIDEx” 在错误代码 “ErrorID” 的值为WORD#16#0800时输出。

值	异常内容	处理方法
16#00000001	通信服务中断。	· 请确认数据链接启动状态。 · 请确认第三节点的传送目标区域的容量。
16#00000101	本机节点未加入网络。	请将本机节点加入网络。
16#00000102	发生了令牌超时。	请在最大节点地址的范围内设定本机节点。
16#00000103	发生重新发送超限。	执行节点间测试发生异常时，请确认使用环境。
16#00000104	发送许可帧数超限。	请确认网络内执行的事件情况，减少1个任务周期内的事件数。或者，请增加发送许可帧数。
16#00000105	本机节点的节点地址超过设定范围。	请对串行通信单元的旋转开关进行正确设定。
16#00000106	本机节点的节点地址在网络内重复。	请变更重复的节点地址。
16#00000201	对象节点未加入网络。	请将对象节点加入网络。
16#00000202	发送目标中指定的单元地址的单元不存在。	请对发送目标网络地址的单元地址进行正确设定。
16#00000203	第三节点未加入网络。	· 请确认作为第三节点的单元地址。 · 第三节点请只指定1个节点。
16#00000204	对象节点为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在对象节点上。
16#00000205	发生响应超时。	请确认通信参数的设定。
16#00000206	传送通道存在异常。	· 请重试。 · 异常频发时，请确认干扰情况。
16#00000301	通信控制器发生异常。	请在参阅相应单元的用户手册后，采取适当处理。
16#00000302	对象节点的CPU单元异常。	请参阅手册排除对象节点的CPU单元异常。
16#00000303	相应控制器存在异常，未返回响应。	请在确认网络的通信情况后，重启相应控制器。即使这样仍发生异常时，请更换相应控制器。
16#00000304	单元编号设定不正确。	请对串行通信单元的旋转开关进行正确设定。
16#00000401	发送的指令不被支持。	请对指令数组的内容进行正确设定。
16#00000402	单元的机型或版本不被支持。	请确认单元的机型和版本。
16#00000501	对象地址设定异常。	请在路由表中设定目标地址。
16#00000502	未登录路由表。	在发送源节点、目标节点、中继节点设定路由表。
16#00000503	路由表异常。	请正确设定路由表。
16#00000504	发生中继次数超限。	请重组网络或调整路由表，将指令的使用范围设定在3层以内。
16#00001001	指令长度超过最大指令长度。	请对指令数组的内容进行正确设定。
16#00001002	指令长度不足最小指令长度。	请对指令数组的内容进行正确设定。
16#00001003	指令指定的写入元素数与实际写入的数据数不一致。	请使写入元素数与实际的写入数据数一致。
16#00001004	指令格式异常。	请对指令数组的内容进行正确设定。
16#00001005	报头异常。	请正确设定路由表。
16#00001101	无区域种类。	请在参照指令的变量、参数种类代码的基础上，设定相应代码。
16#00001102	访问大小异常。	请正确设定变量、参数的访问大小。
16#00001103	指定了超出范围的地址。	请指定可处理范围内的地址。
16#00001104	超出地址范围。	· 请指定可处理范围内的地址。 · 请正确设定数据链接表。
16#00001106	指定了未登录的收发时序No.。	请修改收发时序No.或通过CX-Protocol增加时序。
16#00001109	发生了相关关系异常。	· 请正确设定指令数据的大小关系。 · 请正确设定数据链接表。

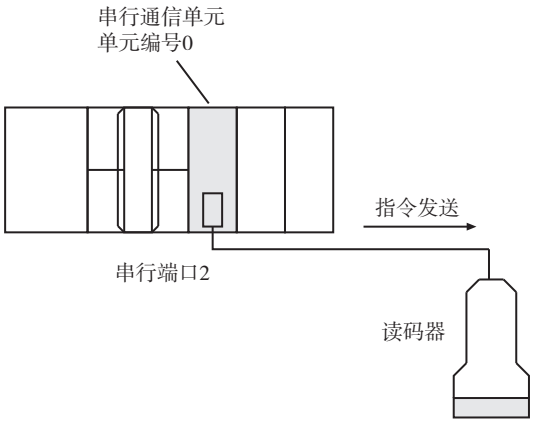
值	异常内容	处理方法
16#0000110A	数据重复。	· 请中断执行中的处理或等待至结束后再执行指令。 · 请正确设定数据链接表。
16#0000110B	响应超过最大响应长度。	请对指令数组内的元素数进行正确设定。
16#0000110C	其他参数异常	请对指令数组的内容进行正确设定。
16#00002002	保护中。	请在解除保护后，重新执行指令。
16#00002003	无登录表。	请正确设定表格。
16#00002004	不存在与检索数据一致的数据。	请正确设定检索数据。
16#00002005	无相应的程序编号。	请设定有效的程序编号。
16#00002006	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002007	发生了核查异常。	· 请确认存储器的内容后，改写成正确的数据。 · 请确认文件的内容。
16#00002101	为只读区域，因此无法存取。	请在解除写保护后，重新执行指令。
16#00002102	保护中或无法写入数据链接表。	· 请在解除写保护后，重新执行指令。 · 请在数据链接表中设定系统设定。
16#00002103	无法登录。	· 请在删除不用的文件后再创建文件或准备新文件用的存储器。 · 请在关闭已打开的文件后，重新执行指令。
16#00002105	无相应的程序编号。	请设定有效的程序编号。
16#00002106	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002107	存在相同的文件名。	请在变更要写入文件的文件名后，重新执行指令。
16#00002108	变更会导致异常，因此无法变更。	请变更设定。
16#00002201	协议宏已在执行中，因此无法动作。	请将协议宏执行中标志设为b触点。
16#00002202	动作模式异常。	请确认动作模式。
16#00002203	指令的动作模式不同(程序模式)。	请确认控制器的动作模式。
16#00002204	指令的动作模式不同(调试模式)。	请确认控制器的动作模式。
16#00002205	指令的动作模式不同(监控模式)。	请确认控制器的动作模式。
16#00002206	指令的动作模式不同(运行模式)。	请确认控制器的动作模式。
16#00002207	指定节点非管理站。	请确认网络的管理站节点。
16#00002208	指令的动作模式不同。	请确认步活性状态。
16#00002211	单元为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在相应单元上。
16#00002301	无文件装置。	请安装介质。或者，请执行EM格式化。
16#00002302	无文件存储器。	请确认文件存储器的安装。
16#00002303	未内置时钟。	请确认机型的规格。
16#00002401	协议宏数据的SUM值异常或正在传送数据。	请通过CX-Protocol重新传送协议宏数据。
16#00002502	处理的对象存储器存在异常。	请将正确数据重新传送至对象存储器。
16#00002503	登录的I/O单元构成与实际的单元构成不同。	请确认I/O单元构成。
16#00002504	登录的I/O点数或远程I/O点数超过最大值。	请对I/O点数、远程I/O点数进行正确设定。
16#00002505	CPU单元与CPU高功能单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002506	机架No.、单元编号、I/O地址中的某一设定重复。	请对重复的内容进行重新设定。
16#00002507	CPU单元与I/O单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002509	SYSMAC BUS/2数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250A	CPU高功能单元的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250D	通道设定重复。	请对I/O通道进行正确设定。

值	异常内容	处理方法
16#0000250F	存储器异常。	· 内部存储器时，请写入正确数据后重新执行指令。 · 存储卡及EM文件存储器时，请执行扩展存储器的格式化指令。 · 进行上述处理后仍无法解除异常时，请更换存储器。
16#00002510	末站的设定异常。	请对末站进行正确设定。
16#00002601	保护解除中。	无需解除保护。
16#00002602	密码不一致。	请指定正确的密码。
16#00002604	保护中。	· 请在解除写保护后，重新执行指令。 · 请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002605	服务执行中。	请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002606	服务停止中。	请根据需要执行相应服务。
16#00002607	无执行权。	· 请通过数据链接参加节点执行。 · 重启后仍发生异常时，请更换控制器。
16#00002608	未设定环境。	请进行必要的设定。
16#00002609	未设定必要项目。	请设定必要项目。
16#0000260A	指定编号已定义。	请变更成未登录的动作、过渡编号后，重新执行指令。
16#0000260B	无法解除异常。	请解除造成异常的原因后，执行异常解除指令。
16#00003001	无访问权。	请等待至访问权释放后，重新执行指令。
16#00004001	服务被中断。	请在解除服务被中断的原因后，重新执行指令。

(注) 除上表值以外，终止符的位6、7、15的值可能会变为TRUE。位6、7的值为TRUE时，表示发送目标的CPU单元发生了异常。位15的值为TRUE时，表示网络中继时发生了异常。

示例程序

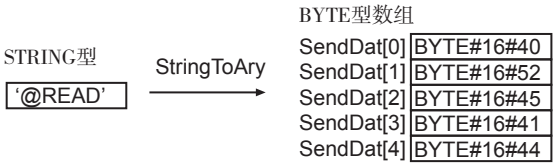
对与CJ系列串行通信单元(单元编号0、设备名称 ‘Barcode’)的串行端口2连接的读码器，发送无协议指令。发送指令为获得场景编号的指令 ‘@READ’。
发送数据为数组变量SendDat[]的内容。无起始符，结束符为16#OD(CR)。



串行通信单元的设定如下所述。

项目名	设定值
端口2：有无任意设定	任意设定
端口2：串行通信模式	无协议
端口2：数据长度	8位
端口2：停止位	1 位
端口2：奇偶校验	无
端口2：传送速度	38400bps
端口2：无协议结束符	D
端口2：有无无协议起始符	无
端口2：有无无协议结束符	有(任意设定结束符)

SendDat[] 中，对字符串 ‘@READ’逐个字符进行分解，并将该文字代码保存至各数组元素中。因此，SendDat[0]中保存BYTE#16#40(@)，SendData[1]中保存BYTE#16#52(R)。该处理使用StringToAry指令。



全局变量的定义

全局变量

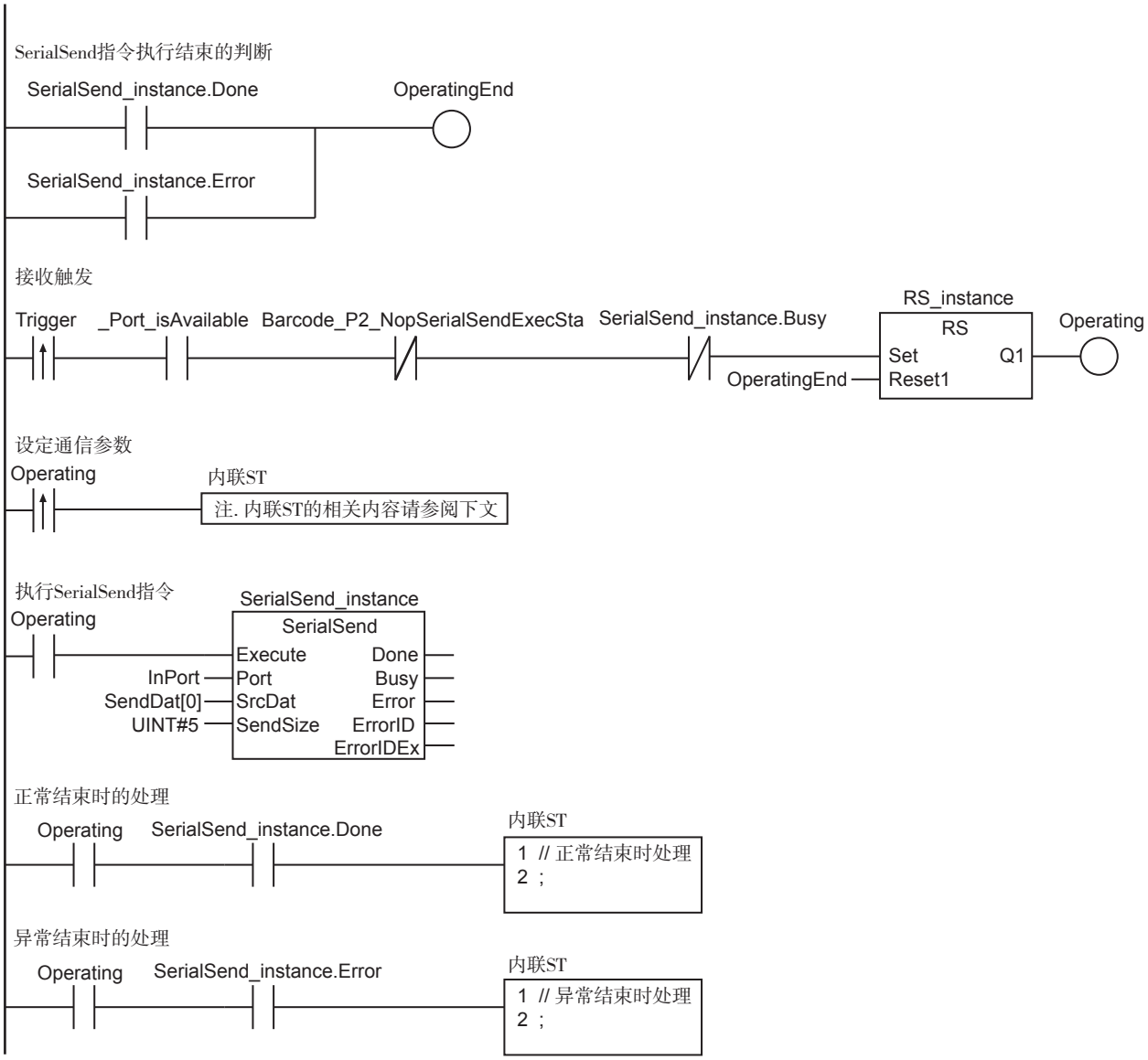
名称	数据类型	分配对象*1	注释
Barcode_P2_NopSerialSendExecSta	BOOL	IOBus://rack#0/slot#0/P2_NopSta/ P2_NopSerialSendExecSta	SerialSend指令 执行中标志

*1　是将串行通信单元安装于机架编号0的插槽编号0上时的分配对象。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	InPort	_sPORT	(UnitNo:=_CBU_No00, PhysicPortNo:=0)	指定端口
	SendDat	ARRAY[0..4] OF BYTE	[5(16#0)]	发送数据
	RS_instance	RS		
	SerialSend_instance	SerialSend		

外部变量	名称	数据类型	注释
	_Port_isAvailable	BOOL	网络通信指令可执行标志
	Barcode_P2_NopSerialSendExecSta	BOOL	SerialSend指令执行中标志



- **内联ST的内容**

```
StringToAry(In:= '@READ', AryOut:=SendDat[0]);  // 准备SendDat[]
InPort.UnitNo       :=_CBU_No00;          // 串行通信单元  单元编号0
InPort.PhysicPortNo:=USINT#2;             // 串行端口2
```

ST

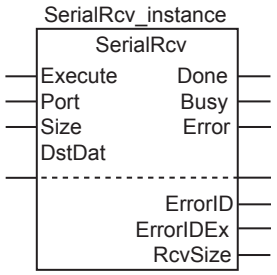
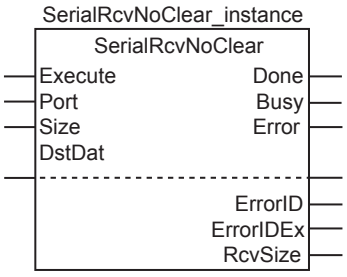
内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	InPort	_sPORT	(UnitNo:=_CBU_No00, PhysicPortNo:=0)	指定端口
	SendDat	ARRAY[0..4] OF BYTE	[5(16#0)]	发送数据
	SerialSend_instance	SerialSend		

外部变量	名称	数据类型	注释
	_Port_isAvailable	BOOL	网络通信指令可执行标志
	Barcode_P2_NopSerialSendExecSta	BOOL	SerialSend指令执行中标志

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Port_isAvailable=TRUE)
     AND (Barcode_P2_NopSerialSendExecSta=FALSE) AND (SerialSend_instance.Busy=FALSE) ) THEN
   OperatingStart:=TRUE;
   Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 设定通信参数和SerialSend指令初始化
IF (OperatingStart=TRUE) THEN
   SerialSend_instance(
      Execute:=FALSE,
      SrcDat :=SendDat[0]);
   StringToAry(In:= '@READ', AryOut:=SendDat[0]);
   InPort.UnitNo        :=_CBU_No00;       // 串行通信单元      单元编号0
   InPort.PhysicPortNo:=USINT#2;          // 串行端口2
   OperatingStart       :=FALSE;
END_IF;

// 执行SerialSend指令
IF (Operating=TRUE) THEN
   SerialSend_instance(
      Execute :=TRUE,
      Port     :=InPort,     // 指定端口
      SrcDat   :=SendDat[0], // 发送数据
      SendSize:=UINT#5);    // 发送数据大小

   IF (SerialSend_instance.Done=TRUE) THEN
      // 正常结束时处理
      Operating:=FALSE;
   END_IF;

   IF (SerialSend_instance.Error=TRUE) THEN
      // 异常结束时处理
      Operating:=FALSE;
  END_IF;
END_IF;
```

SerialRcv/SerialRcvNoClear

从串行通信单元的串行端口读取无协议的接收数据。
SerialRcv　　：读取后，接收缓存全部清除。
SerialRcvNoClear：读取后，接收缓存不会全部清除。

指令	名称	FB/FUN	图形表现	ST表现
SerialRcv	串行通信单元串行端口输入	FB		SerialRcv_instance(Execute, Port, Size, DstDat, Done, Busy, Error, ErrorID, ErrorIDEx, RcvSize);
SerialRcvNoClear	串行通信单元不删除串行端口输入接收缓存	FB		SerialRcvNoClear_instance(Execute, Port, Size, DstDat, Done, Busy, Error, ErrorID, ErrorIDEx, RcvSize);

使用注意事项

本指令无法在NX系列 CPU单元中使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Port	指定对象端口	输入	指定对象端口	–	–	–
Size	接收数据大小		保存至DstDat[]的接收数据大小	0 ~ 256	字节	1
DstDat[]数组	接收数据数组	输入输出	接收数据数组	遵从数据类型	–	–
RcvSize	接收数据保存大小	输出	实际保存至DstDat[]的接收数据大小	0 ~ 256	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Port	结构体_sPORT 详情参阅功能说明。																			
Size							○													
DstDat[]数组		○																		
RcvSize							○													

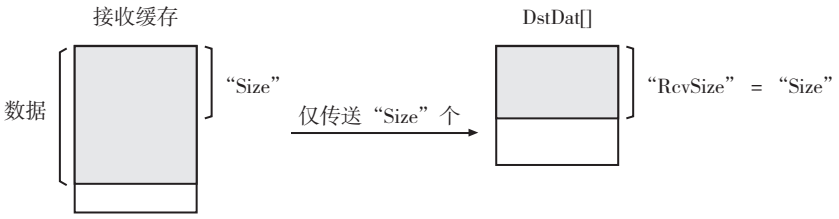
功能

从指定端口“Port”无协议接收的数据先保存至串行通信单元的接收缓存中。对于该接收缓存的数据，本指令仅将接收数据大小“Size”传送至接收数据数组DstDat[]。

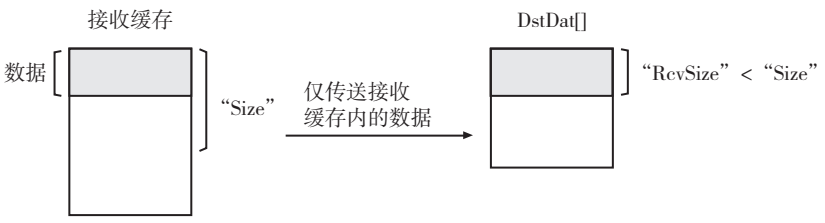
传送后，将实际保存至DstDat[]的数组元素数代入接收数据保存大小“RcvSize”。

接收缓存内的数据量比“Size”少时，仅将接收缓存中存在的数据传送至 DstDat[]。此时，仍将实际保存在DstDat[]中的数据大小代入“RcvSize”。

接收缓存内的数据大于“Size”时



接收缓存内的数据小于“Size”时



接收数据的起始符、结束符

为了在用户程序中识别接收数据的起始符、结束符，使用设备变量。DstDat[]中保存接收数据删除起始符、结束符后的内容。

附加内容	设备变量(端口1)	值
对起始符附加任意值	P1_NopStartCodeYNCfg	TRUE
	P1_NopStartCodeCfg	起始符(16#00 ~ 16#FF)
对结束符附加任意值	P1_NopEndCodeYNCfg	TRUE
	P1_NopCRLFCfg	FALSE
	P1_NopEndCodeCfg	结束符(16#00 ~ 16#FF)
对结束符附加CR+LF	P1_NopEndCodeYNCfg	TRUE
	P1_NopCRLFCfg	TRUE

附加起始符、结束符时的可接收字节数最大为259(起始符1字节 + 结束符2字节(指定CR+LF时) + 发送数据256字节)。

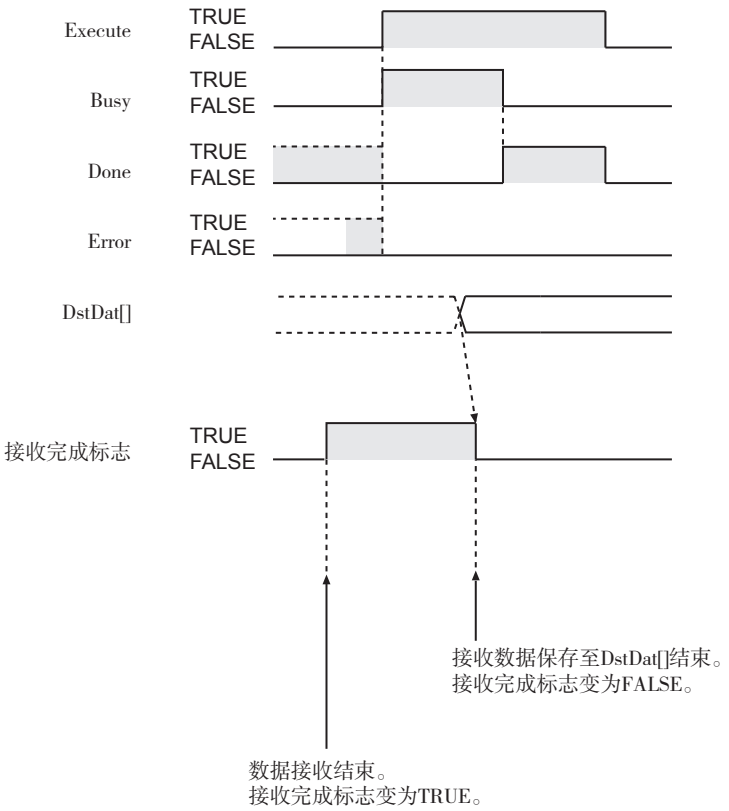
对象端口指定 “Port”的数据类型

对象端口指定 “Port”的数据类型为结构体_sPORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Port	指定对象端口	指定对象端口	_sPORT	–	–	–
UnitNo	单元编号	串行通信单元的单元编号	_eUnitNo	_CBU_No00 ~ _CBU_No15	–	_CBU_No00
PhysicPortNo	串行端口编号	串行通信单元的串行端口编号	USINT	1,2		1

时序图

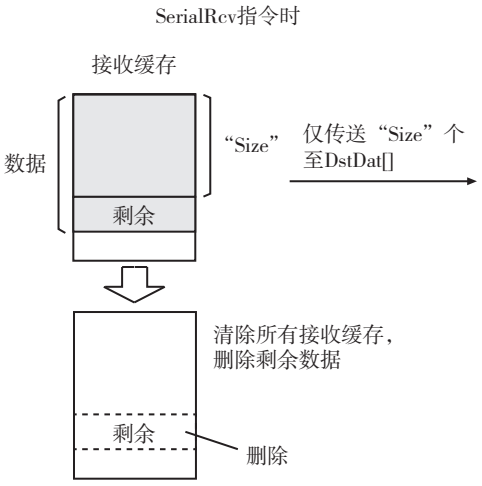
时序图如下所示。



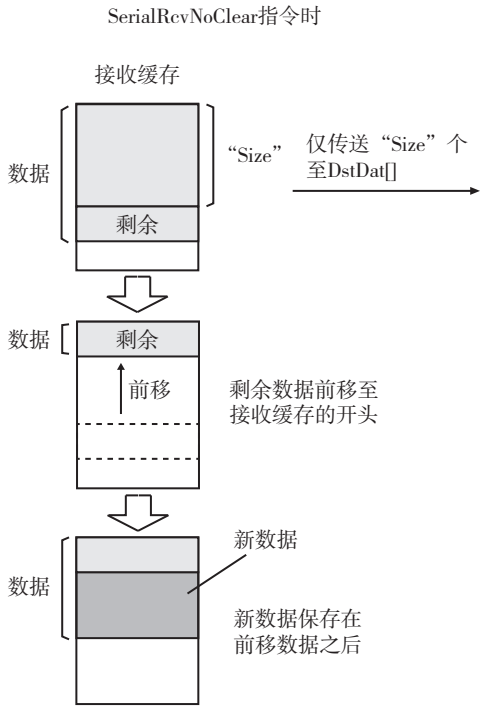
SerialRcv指令与SerialRcvNoClear指令的区别

SerialRcv指令与SerialRcvNoClear指令的区别在于，数据从接收缓存传送至DstDat[]后，接收缓存是否会被自动清除。

SerialRcv指令在数据传送后，接收缓存会被全部清除。因此，接受缓存内的数据比“Size”多时，传送后剩余的数据将被删除。



SerialRcvNoClear 指令在数据传送后仅清除已传送的数据，接收缓存中剩余的数据将前移至接收缓存的开头。之后，新存入接收缓存的数据将保存在已前移的数据的后面。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Port_numUsingPort	使用端口数	USINT	当前使用中的端口数
_Port_isAvailable	网络通信指令可执行标志	BOOL	TRUE ：有可使用端口 FALSE：无可使用端口

相关的准用户定义变量

变量名称	名称	数据类型	内容
P#_NopRcvOvfSta(*)	接收超限标志	BOOL	TRUE：单元接收的数据超过指定接收数据数 (接收完成标志TRUE后仍有接收) FALSE：单元接收的数据未超过指定接收数据数
P#_NopRcvCompleteSta(*)	接收完成标志	BOOL	TRUE：接收完成 FALSE：未接收或接收中
P#_NopRcvCntSta(*)	接收计数	UINT	16#0000 ~ 16#0100：接收数据的字节数
P#_NopStartCodeYNCfg(*)	有无无协议起始符	BOOL	TRUE：有 FALSE：无
P#_NopEndCodeYNCfg(*)	有无无协议结束符	BOOL	TRUE：有 FALSE：无
P#_NopCRLFCfg(*)	指定无协议CRLF	BOOL	TRUE：指定 FALSE：不指定
P#_NopRcvDatSzCfg(*)	无协议接收数据数	USINT	16#01 ~ 16#FF：1 ~ 255字节 16#00：256字节
P#_NopStartCodeCfg(*)	无协议起始符	USINT	16#00 ~ 16#FF
P#_NopEndCodeCfg(*)	无协议结束符	USINT	16#00 ~ 16#FF
P#_TransErr(*)	传送异常	BOOL	TRUE：有异常发生 FALSE：无异常发生
P#_OverRunErr(*)	超程异常	BOOL	TRUE：有异常发生 FALSE：无异常发生

* #中填入串行通信单元的端口No.。

参考

- 下列场合下，接收完成标志 “P#_NopRcvCompleteSta”的值会变为TRUE。
  - 接收了无协议接收数据数 “P#NopRcvDatSzCfg”设定值量的数据。
  - 接收了指定的结束符。
  - 接收了256字节的数据。
- 下列场合下，接收超限标志 “P#_NopRcvOvfSta”的值会变为TRUE。
  - 接收完成标志 “P#_NopRcvCompleteSta”的值为TRUE的状态下，不执行本指令而继续接收数据。
  - 接收的数据超过了无协议接收数据数 “P#NopRcvDatSzCfg”的设定值。
- 无协议通信的详情请参阅下列手册。
  - 🕮 “CJ系列 串行通信单元 用户手册 NJ系列连接篇(SBCD-354)”

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 本指令请在接收完成标志 “P#_NopRcvCompleteSta”的值为TRUE时执行。
- 接收数据时，请务必在执行本指令后，再将接收缓存的数据传送至DstDat[]。否则，下一数据将无法接收。
- 接收259字节的数据时，将自动停止接收。之后，不执行本指令而接收数据，超程异常“P#_OverRunErr”的值将变为TRUE。
- 附加起始符、结束符时，“Size”中请设定不含起始符、结束符的值。
- 本指令仅可用于设定为无协议模式的串行通信单元的串行端口。
- “Size”的值设为0时，不会将接收缓存的数据传送至DstDat[]。此时，接收完成标志“P#_NopRcvCompleteSta”与接收超限标志 “P#_NopRcvOvfSta”的值变为FALSE。此外，接收计数器“P#_NopRcvCntSta”的值变为0。
- 本指令仅可在端口可使用时执行。因此，在本指令的输入条件中，请将网络通信指令可执行标志的系统定义变量 “_Port_isAvailable”作为a触点插入。
- 本指令请在接收完成标志 “P#_NopRcvCompleteSta”的值为TRUE时执行。
- 本指令在 “Busy”的值为TRUE时无法执行。因此，作为本指令的输入条件，请将 “Busy”作为b触点插入。
- 在ST程序中记述本指令时，执行本指令过程中，请设为每个任务周期都对本指令进行控制。否则，会出现无法正常处理的情况。
- 执行SerialRcv指令时，串行通信单元的接收缓存会被全部清除。因此，无法将接收缓存的数据分割传送至DstDat[]。
- SerialRcv指令时，超过 “Size”指定大小的接收数据在下次执行SerialRcv指令时清除。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 串行通信模式非无协议的情况下，执行本指令时。
  - “_Port_isAvailable”的值为FALSE时。
  - “Port.UnitNo”或 “Port.PhysicPortNo”的值超过有效范围时。
  - 指定的单元编号中未安装CJ串行通信单元时。
  - “Size”的值超过有效范围时。
  - “Size”的值超过DstDat[]的大小时。
  - 通信失败时。
  - 单元重启中执行本指令时。
- 扩展错误代码 “ErrorIDEx”在本指令中具有通信响应代码的含义。该值与异常内容如下所述。“ErrorIDEx”在错误代码 “ErrorID”的值为WORD#16#0800时输出。

值	异常内容	处理方法
16#00000001	通信服务中断。	· 请确认数据链接启动状态。 · 请确认第三节点的传送目标区域的容量。
16#00000101	本机节点未加入网络。	请将本机节点加入网络。
16#00000102	发生了令牌超时。	请在最大节点地址的范围内设定本机节点。
16#00000103	发生重新发送超限。	执行节点间测试发生异常时，请确认使用环境。
16#00000104	发送许可帧数超限。	请确认网络内执行的事件情况，减少1个任务周期内的事件数。或者，请增加发送许可帧数。
16#00000105	本机节点的节点地址超过设定范围。	请对串行通信单元的旋转开关进行正确设定。
16#00000106	本机节点的节点地址在网络内重复。	请变更重复的节点地址。
16#00000201	对象节点未加入网络。	请将对象节点加入网络。

值	异常内容	处理方法
16#00000202	发送目标中指定的单元地址的单元不存在。	请对发送目标网络地址的单元地址进行正确设定。
16#00000203	第三节点未加入网络。	· 请确认作为第三节点的单元地址。 · 第三节点请只指定1个节点。
16#00000204	对象节点为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在对象节点上。
16#00000205	发生响应超时。	请确认通信参数的设定。
16#00000206	传送通道存在异常。	· 请重试。 · 异常频发时，请确认干扰情况。
16#00000301	通信控制器发生异常。	请在参阅相应单元的用户手册后，采取适当处理。
16#00000302	对象节点的CPU单元异常。	请参阅手册排除对象节点的CPU单元异常。
16#00000303	相应控制器存在异常，未返回响应。	请在确认网络的通信情况后，重启相应控制器。即使这样仍发生异常时，请更换相应控制器。
16#00000304	单元编号设定不正确。	请对串行通信单元的旋转开关进行正确设定。
16#00000401	发送的指令不被支持。	请对指令数组的内容进行正确设定。
16#00000402	单元的机型或版本不被支持。	请确认单元的机型和版本。
16#00000501	对象地址设定异常。	请在路由表中设定目标地址。
16#00000502	未登录路由表。	在发送源节点、目标节点、中继节点设定路由表。
16#00000503	路由表异常。	请正确设定路由表。
16#00000504	发生中继次数超限。	请重组网络或调整路由表，将指令的使用范围设定在3层以内。
16#00001001	指令长度超过最大指令长度。	请对指令数组的内容进行正确设定。
16#00001002	指令长度不足最小指令长度。	请对指令数组的内容进行正确设定。
16#00001003	指令指定的写入元素数与实际写入的数据数不一致。	请使写入元素数与实际的写入数据数一致。
16#00001004	指令格式异常。	请对指令数组的内容进行正确设定。
16#00001005	报头异常。	请正确设定路由表。
16#00001101	无区域种类。	请在参照指令的变量、参数种类代码的基础上，设定相应代码。
16#00001102	访问大小异常。	请正确设定变量、参数的访问大小。
16#00001103	指定了超出范围的地址。	请指定可处理范围内的地址。
16#00001104	超出地址范围。	· 请指定可处理范围内的地址。 · 请正确设定数据链接表。
16#00001106	指定了未登录的收发时序No.。	请修改收发时序No.或通过CX-Protocol增加时序。
16#00001109	发生了相关关系异常。	· 请正确设定指令数据的大小关系。 · 请正确设定数据链接表。
16#0000110A	数据重复。	· 请中断执行中的处理或等待至结束后再执行指令。 · 请正确设定数据链接表。
16#0000110B	响应超过最大响应长度。	请对指令数组内的元素数进行正确设定。
16#0000110C	其他参数异常	请对指令数组的内容进行正确设定。
16#00002002	保护中。	请在解除保护后，重新执行指令。
16#00002003	无登录表。	请正确设定表格。
16#00002004	不存在与检索数据一致的数据。	请正确设定检索数据。
16#00002005	无相应的程序编号。	请设定有效的程序编号。
16#00002006	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002007	发生了核查异常。	· 请确认存储器的内容后，改写成正确的数据。 · 请确认文件的内容。
16#00002101	为只读区域，因此无法存取。	请在解除写保护后，重新执行指令。

值	异常内容	处理方法
16#00002102	保护中或无法写入数据链接表。	· 请在解除写保护后，重新执行指令。 · 请在数据链接表中设定系统设定。
16#00002103	无法登录。	· 请在删除不用的文件后再创建文件或准备新文件用的存储器。 · 请在关闭已打开的文件后，重新执行指令。
16#00002105	无相应的程序编号。	请设定有效的程序编号。
16#00002106	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002107	存在相同的文件名。	请在变更要写入文件的文件名后，重新执行指令。
16#00002108	变更会导致异常，因此无法变更。	请变更设定。
16#00002201	协议宏已在执行中，因此无法动作。	请将协议宏执行中标志设为b触点。
16#00002202	动作模式异常。	请确认动作模式。
16#00002203	指令的动作模式不同(程序模式)。	请确认控制器的动作模式。
16#00002204	指令的动作模式不同(调试模式)。	请确认控制器的动作模式。
16#00002205	指令的动作模式不同(监控模式)。	请确认控制器的动作模式。
16#00002206	指令的动作模式不同(运行模式)。	请确认控制器的动作模式。
16#00002207	指定节点非管理站。	请确认网络的管理站节点。
16#00002208	指令的动作模式不同。	请确认步活性状态。
16#00002211	单元为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在相应单元上。
16#00002301	无文件装置。	请安装介质。或者，请执行EM格式化。
16#00002302	无文件存储器。	请确认文件存储器的安装。
16#00002303	未内置时钟。	请确认机型的规格。
16#00002401	协议宏数据的SUM值异常或正在传送数据。	请通过CX-Protocol重新传送协议宏数据。
16#00002502	处理的对象存储器存在异常。	请将正确数据重新传送至对象存储器。
16#00002503	登录的I/O单元构成与实际的单元构成不同。	请确认I/O单元构成。
16#00002504	登录的I/O点数或远程I/O点数超过最大值。	请对I/O点数、远程I/O点数进行正确设定。
16#00002505	CPU单元与CPU高功能单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002506	机架No.、单元编号、I/O地址中的某一设定重复。	请对重复的内容进行重新设定。
16#00002507	CPU单元与I/O单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002509	SYSMAC BUS/2数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250A	CPU高功能单元的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250D	通道设定重复。	请对I/O通道进行正确设定。
16#0000250F	存储器异常。	· 内部存储器时，请写入正确数据后重新执行指令。 · 存储卡及EM文件存储器时，请执行扩展存储器的格式化指令。 · 进行上述处理后仍无法解除异常时，请更换存储器。
16#00002510	末站的设定异常。	请对末站进行正确设定。
16#00002601	保护解除中。	无需解除保护。
16#00002602	密码不一致。	请指定正确的密码。
16#00002604	保护中。	· 请在解除写保护后，重新执行指令。 · 请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002605	服务执行中。	请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002606	服务停止中。	请根据需要执行相应服务。

值	异常内容	处理方法
16#00002607	无执行权。	· 请通过数据链接参加节点执行。 · 重启后仍发生异常时，请更换控制器。
16#00002608	未设定环境。	请进行必要的设定。
16#00002609	未设定必要项目。	请设定必要项目。
16#0000260A	指定编号已定义。	请变更成未登录的动作、过渡编号后，重新执行指令。
16#0000260B	无法解除异常。	请解除造成异常的原因后，执行异常解除指令。
16#00003001	无访问权。	请等待至访问权释放后，重新执行指令。
16#00004001	服务被中断。	请在解除服务被中断的原因后，重新执行指令。

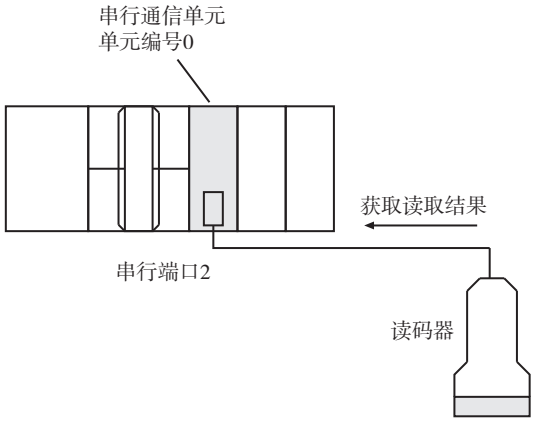
(注) 除上表值以外，终止符的位6、7、15的值可能会变为TRUE。位6、7的值为TRUE时，表示发送目标的CPU单元发生了异常。位15的值为TRUE时，表示网络中继时发生了异常。

版本相关信息

SerialRcvNoClear 指令可用于 Ver.1.03 以上版本的 CPU 单元、Ver.1.04 以上版本的 Sysmac Studio、Ver.2.1以上版本的串行通信单元的组合。

示例程序

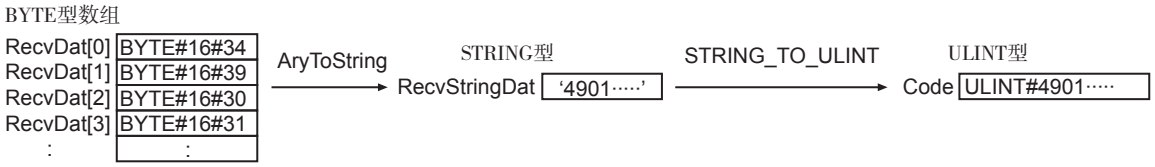
获取与CJ系列串行通信单元(单元编号0、设备名称 ‘Barcode’)的串行端口2连接的读码器的读取结果。接收数据保存至数组变量RecvDat[]。无起始符，结束符为16#OD(CR)。



串行通信单元的设定如下所述。

项目名	设定值
端口2：有无任意设定	任意设定
端口2：串行通信模式	无协议
端口2：数据长度	8位
端口2：停止位	1 位
端口2：奇偶校验	无
端口2：传送速度	38400bps
端口2：无协议结束符	D
端口2：有无无协议起始符	无
端口2：有无无协议结束符	有(任意设定结束符)

保存至RecvDat[]的数据是将条形码数值逐个字符通过文字代码表示的位串。RecvDat[]数组的1个元素与条形码的 1 个字符一一对应。首先，使用 AryToString 指令，将其转换为字符串 RecvStringDat。然后，使用STRING_TO_ULINT指令，将其转换为ULINT型整数Code。



全局变量的定义

全局变量

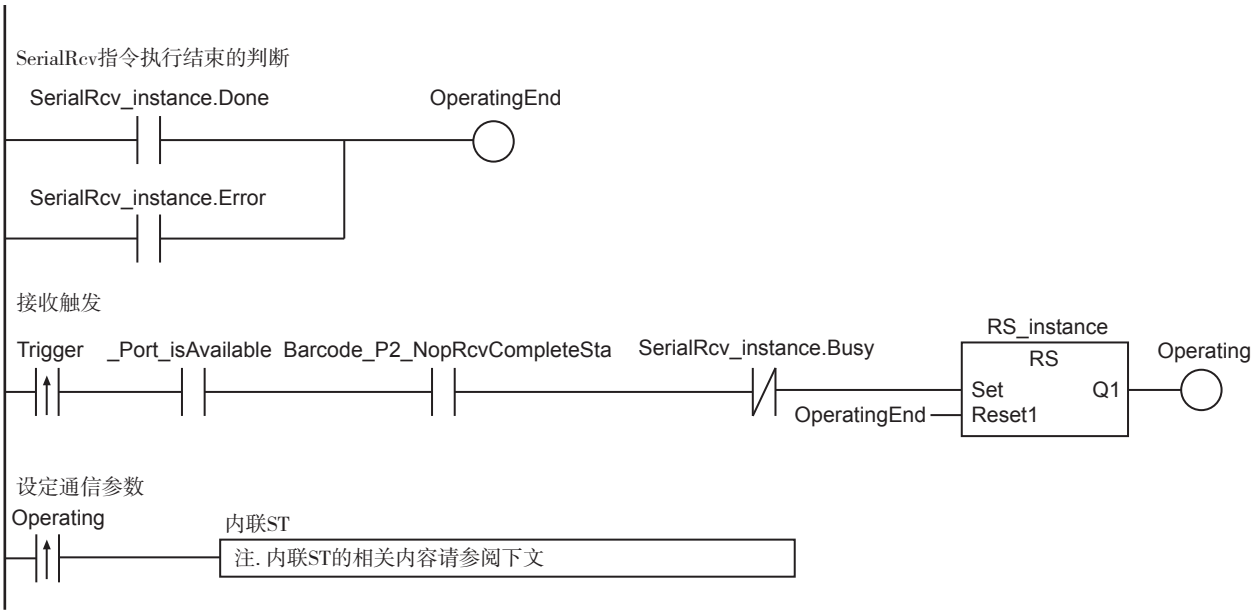
名称	数据类型	分配对象*1	注释
Barcode_P2_NopRcvCompleteSta	BOOL	IOBus://rack#0/slot#0/P2_NopSta/P2_NopRcvCompleteSta	接收完成标志

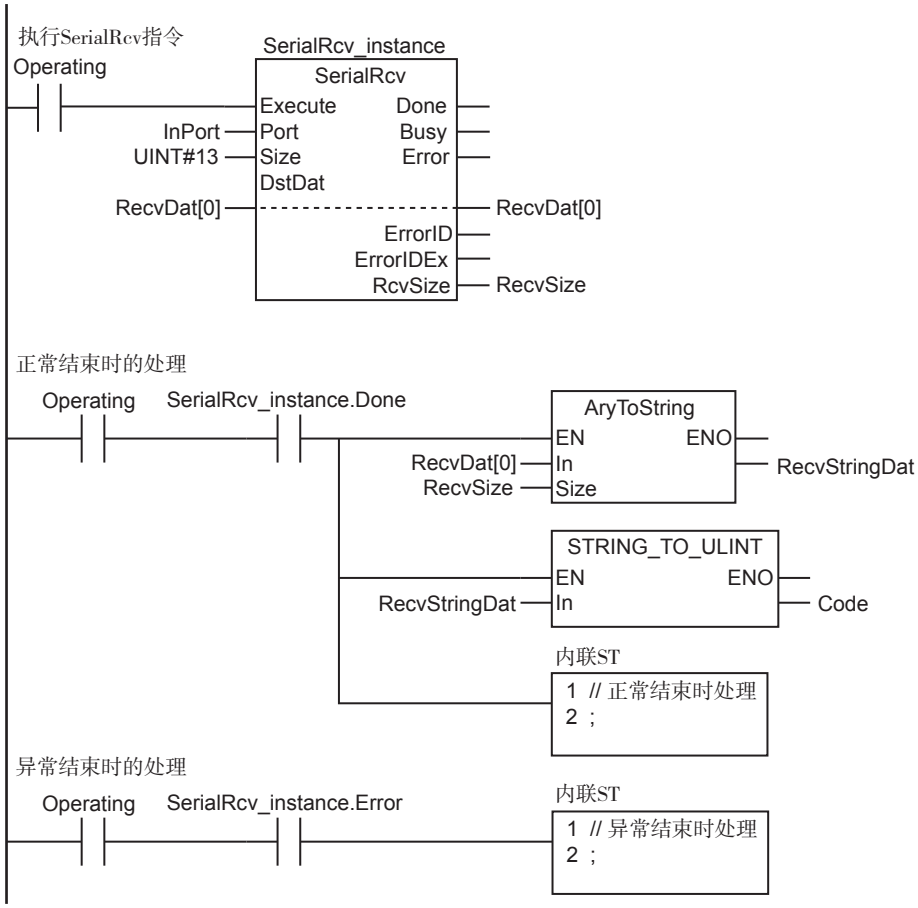
*1 是将串行通信单元安装于机架编号0的插槽编号0上时的分配对象。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	InPort	_sPORT	(UnitNo:=_CBU_No00, PhysicPortNo:=0)	指定端口
	RecvDat	ARRAY[0..12] OF BYTE	[13(16#0)]	接收数据
	RecvSize	UINT	0	接收数据大小
	RecvStringDat	STRING[255]	''	条形码(字符串)
	Code	ULINT	0	条形码(整数)
	RS_instance	RS		
	SerialRcv_instance	SerialRcv		

外部变量	名称	数据类型	注释
	_Port_isAvailable	BOOL	网络通信指令可执行标志
	Barcode_P2_NopRcvCompleteSta	BOOL	接收完成标志




执行SerialRcv指令
Operating
SerialRcv_instance
SerialRcv
Execute
Done
InPort
Port
Busy
UINT#13
Size
Error
DstDat
RecvDat[0]
RecvDat[0]
ErrorID
ErrorIDEx
RcvSize
RecvSize
正常结束时的处理
Operating
SerialRcv_instance.Done
AryToString
EN
ENO
RecvDat[0]
In
RecvStringDat
RecvSize
Size
STRING_TO_ULINT
EN
ENO
RecvStringDat
In
Code
内联ST
1 // 正常结束时处理
2 ;
异常结束时的处理
Operating
SerialRcv_instance.Error
内联ST
1 // 异常结束时处理
2 ;

● 内联ST的内容

```
InPort.UnitNo        :=_CBU_No00;  // 串行通信单元  单元编号0
InPort.PhysicPortNo:=USINT#2;    // 串行端口2
```

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	InPort	_sPORT	(UnitNo:=_CBU_No00, PhysicPortNo:=0)	指定端口
	RecvDat	ARRAY[0..12] OF BYTE	[13(16#0)]	接收数据
	RecvSize	UINT	0	接收数据大小
	RecvStringDat	STRING[255]	''	条形码(字符串)
	Code	ULINT	0	条形码(整数)
	SerialRcv_instance	SerialRcv		

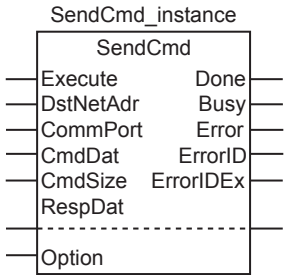
外部变量	名称	数据类型	注释
	_Port_isAvailable	BOOL	网络通信指令可执行标志
	Barcode_P2_NopRcvCompleteSta	BOOL	接收完成标志

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Port_isAvailable=TRUE)
     AND (Barcode_P2_NopRcvCompleteSta=TRUE) AND (SerialRcv_instance.Busy=FALSE) ) THEN
   OperatingStart:=TRUE;
   Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 设定通信参数和SerialRcv指令初始化
IF (OperatingStart=TRUE) THEN
   SerialRcv_instance(
      Execute:=FALSE,     // 实例初始化
      Port     :=InPort,    // 指定端口
      Size     :=UINT#13,   // 接收数据大小
      DstDat  :=RecvDat[0], // 接收数据
      RcvSize =>RecvSize); // 实际接收的数据大小
   InPort.UnitNo        :=_CBU_No00; // 串行通信单元  单元编号0
   InPort.PhysicPortNo:=USINT#2;  // 串行端口2
   OperatingStart       :=FALSE;
END_IF;

// 执行SerialRcv指令
IF (Operating=TRUE) THEN
   SerialRcv_instance(
      Execute:=TRUE,
      Port     :=InPort,
      Size     :=UINT#13,
      DstDat :=RecvDat[0],
      RcvSize =>RecvSize);

   IF (SerialRcv_instance.Done=TRUE) THEN
      // 正常结束时处理
      RecvStringDat:=AryToString(In:=RecvDat[0], Size:=RecvSize);  // 将文字代码转换为字符串
      Code             :=STRING_TO_ULINT(RecvStringDat);           // 将字符串转换为整数
      Operating      :=FALSE;
   END_IF;
```

```
    IF (SerialRcv_instance.Error=TRUE) THEN
      // 异常结束时处理
      Operating:=FALSE;
    END_IF;
  END_IF;
```

SendCmd

通过串行网关功能向串行通信单元发送指令。此外，向DeviceNet单元和CompoNet主站单元发送Explicit指令。

指令	名称	FB/FUN	图形表现	ST表现
SendCmd	指令发送	FB		SendCmd_instance(Execute, DstNetAdr, CommPort, CmdDat, CmdSize, RespDat, Option, Done, Busy, Error, ErrorID, ErrorIDEx);

使用注意事项

本指令无法在NX系列 CPU单元中使用。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DstNetAdr	发送目标网址	输入	发送目标网址	–	–	–
CommPort	发送目标串行端口指定		发送目标串行端口的指定	仅_NONE		_NONE
CmdDat[]数组	指令数组		待发送指令	遵从数据类型		(*)
CmdSize	指令数据大小		指令数据大小	2～最大数据长度(遵从网络类型)	字节	2
Option	响应		响应监视及重新发送指定	–	–	–
RespDat[]数组	响应保存数组	输入输出	保存响应的数组	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DstNetAdr	结构体_sDNET_ADR 详情参阅功能说明																			
CommPort	枚举体_ePORT 枚举元素参阅功能说明																			
CmdDat[]数组		○																		
CmdSize							○													
Option	结构体_sRESPONSE 详情参阅功能说明																			
RespDat[]数组		○																		

功能

将指令数组CmdDat[]的内容发送至发送目标网址 “DstNetAdr”、发送目标串行端口指定 “CommPort”指定的发送目标。
CmdDat[]第几个元素为止视为指令由指令数据大小 “CmdSize”指定。
之后，返回的响应保存至响应保存数组RespDat[]。

“DstNetAdr”的数据类型为结构体_sDNET_ADR。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DstNetAdr	发送目标网址	发送目标网址	_sDNET_ADR	–	–	–
NetNo	网址	网址	USINT	仅0	–	0
NodeNo	节点地址	节点地址	USINT			
UnitNo	单元地址	单元地址	BYTE	遵从数据类型		16#00

“CommPort”的数据类型为枚举体_ePORT。
枚举体_ePORT的枚举元素的含义如下所述。

枚举元素	含义
_NONE	发送目标非串行端口(高位链接模式)

“Option”的数据类型为结构体_sRESPONSE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	响应	响应监视及重新发送指定	_sRESPONSE	–	–	–
isNonResp	无需响应	TRUE ：无需响应 FALSE：需响应	BOOL	遵从数据类型	–	FALSE
TimeOut	超时时间	超时时间 “0”时为2.0s	UINT		0.1s	20 (2.0s)
Retry	重新发送次数	重新发送次数	USINT	0 ~ 15	次	0

无需响应标志 “Option.isNonResp”的值为FALSE时，超时时间
“Option.TimeOut”的值以内无响应时，将重新发送指令直至有响应。重新发送次数由 “Option.Retry”指定。
超时时间为 “Option.TimeOut” × 0.1s。“Option.TimeOut”的值为0时 ，为2.0s。“Option.TimeOut”的初始值也为2.0s。

相关的系统定义变量

变量名称	名称	数据类型	内容
_Port_numUsingPort	使用端口数	USINT	当前使用中的端口数
_Port_isAvailable	网络通信指令可执行标志	BOOL	TRUE ：有可使用端口 FALSE：无可使用端口

参考

- 受干扰等影响，通信途中指令及响应有时会消失。因此，将 “Option.Retry”的值设为0以外，在未返回响应时进行重新发送处理将增加可靠信。
- 通过串行网关功能指定串行端口时，在 “DstNetAdr.UnitNo”中设定串行端口的单元地址。串行通信单元的端口的单元地址如下所述。
  - 使用端口1时

    单元地址=BYTE#16#80 + BYTE#16#04 × 单元编号(16进制)

    (例) 单元编号1时

    BYTE#16#80+BYTE#16#04 × 1 = BYTE#16#84
  - 使用端口2时

    单元地址=BYTE#16#81 + BYTE#16#04 × 单元编号(16进制)

    (例) 单元编号2时

    BYTE#16#81+BYTE#16#04 × 2 = BYTE#16#89

使用注意事项

- 本指令仅可在端口可使用时执行。因此，在本指令的输入条件中，请将网络通信指令可执行标志的系统定义变量 “_Port_isAvailable”作为a触点插入。
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在 ST 程序中记述本指令时，执行本指令过程中，请设为每个任务周期都对本指令进行控制。否则，会出现无法正常处理的情况。
- “CmdSize”的值为0时，不发送指令。此时执行指令时，“Done”的值为TRUE。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “CommPort”的值超过有效范围时。
  - “DstNetAdr”的任一结构要素超过有效范围时。
  - “CmdSize”的值超过有效范围时。
  - “Option”的任一结构要素超过有效范围时。
  - “CmdSize”的值超过CmdDat[]的大小时。
  - 响应的大小超过RespDat[]的大小时。
  - “_Port_isAvailable”的值为FALSE时。
  - 通信失败时。

• 扩展错误代码 “ErrorIDEx” 在本指令中具有通信响应代码的含义。该值与异常内容如下所述。“ErrorIDEx” 在错误代码 “ErrorID” 的值为WORD#16#0800时输出。

值	异常内容	处理方法
16#00000001	通信服务中断。	· 请确认数据链接启动状态。 · 请确认第三节点的传送目标区域的容量。
16#00000101	本机节点未加入网络。	请将本机节点加入网络。
16#00000102	发生了令牌超时。	请在最大节点地址的范围内设定本机节点。
16#00000103	发生重新发送超限。	执行节点间测试发生异常时，请确认使用环境。
16#00000104	发送许可帧数超限。	请确认网络内执行的事件情况，减少1个任务周期内的事件数。或者，请增加发送许可帧数。
16#00000105	本机节点的节点地址超过设定范围。	请对串行通信单元的旋转开关进行正确设定。
16#00000106	本机节点的节点地址在网络内重复。	请变更重复的节点地址。
16#00000201	对象节点未加入网络。	请将对象节点加入网络。
16#00000202	发送目标中指定的单元地址的单元不存在。	请对发送目标网络地址的单元地址进行正确设定。
16#00000203	第三节点未加入网络。	· 请确认作为第三节点的单元地址。 · 第三节点请只指定1个节点。
16#00000204	对象节点为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在对象节点上。
16#00000205	发生响应超时。	请确认通信参数的设定。
16#00000206	传送通道存在异常。	· 请重试。 · 异常频发时，请确认干扰情况。
16#00000301	通信控制器发生异常。	请在参阅相应单元的用户手册后，采取适当处理。
16#00000302	对象节点的CPU单元异常。	请参阅手册排除对象节点的CPU单元异常。
16#00000303	相应控制器存在异常，未返回响应。	请在确认网络的通信情况后，重启相应控制器。即使这样仍发生异常时，请更换相应控制器。
16#00000304	单元编号设定不正确。	请对串行通信单元的旋转开关进行正确设定。
16#00000401	发送的指令不被支持。	请对指令数组的内容进行正确设定。
16#00000402	单元的机型或版本不被支持。	请确认单元的机型和版本。
16#00000501	对象地址设定异常。	请在路由表中设定目标地址。
16#00000502	未登录路由表。	在发送源节点、目标节点、中继节点设定路由表。
16#00000503	路由表异常。	请正确设定路由表。
16#00000504	发生中继次数超限。	请重组网络或调整路由表，将指令的使用范围设定在3层以内。
16#00001001	指令长度超过最大指令长度。	请对指令数组的内容进行正确设定。
16#00001002	指令长度不足最小指令长度。	请对指令数组的内容进行正确设定。
16#00001003	指令指定的写入元素数与实际写入的数据数不一致。	请使写入元素数与实际的写入数据数一致。
16#00001004	指令格式异常。	请对指令数组的内容进行正确设定。
16#00001005	报头异常。	请正确设定路由表。
16#00001101	无区域种类。	请在参照指令的变量、参数种类代码的基础上，设定相应代码。
16#00001102	访问大小异常。	请正确设定变量、参数的访问大小。
16#00001103	指定了超出范围的地址。	请指定可处理范围内的地址。
16#00001104	超出地址范围。	· 请指定可处理范围内的地址。 · 请正确设定数据链接表。
16#00001106	指定了未登录的收发时序No.。	请修改收发时序No.或通过CX-Protocol增加时序。
16#00001109	发生了相关关系异常。	· 请正确设定指令数据的大小关系。 · 请正确设定数据链接表。

值	异常内容	处理方法
16#0000110A	数据重复。	· 请中断执行中的处理或等待至结束后再执行指令。 · 请正确设定数据链接表。
16#0000110B	响应超过最大响应长度。	请对指令数组内的元素数进行正确设定。
16#0000110C	其他参数异常	请对指令数组的内容进行正确设定。
16#00002002	保护中。	请在解除保护后，重新执行指令。
16#00002003	无登录表。	请正确设定表格。
16#00002004	不存在与检索数据一致的数据。	请正确设定检索数据。
16#00002005	无相应的程序编号。	请设定有效的程序编号。
16#00002006	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002007	发生了核查异常。	· 请确认存储器的内容后，改写成正确的数据。 · 请确认文件的内容。
16#00002101	为只读区域，因此无法存取。	请在解除写保护后，重新执行指令。
16#00002102	保护中或无法写入数据链接表。	· 请在解除写保护后，重新执行指令。 · 请在数据链接表中设定系统设定。
16#00002103	无法登录。	· 请在删除不用的文件后再创建文件或准备新文件用的存储器。 · 请在关闭已打开的文件后，重新执行指令。
16#00002105	无相应的程序编号。	请设定有效的程序编号。
16#00002106	无相应文件。	含子目录名称在内，请正确设定文件名。
16#00002107	存在相同的文件名。	请在变更要写入文件的文件名后，重新执行指令。
16#00002108	变更会导致异常，因此无法变更。	请变更设定。
16#00002201	协议宏已在执行中，因此无法动作。	请将协议宏执行中标志设为b触点。
16#00002202	动作模式异常。	请确认动作模式。
16#00002203	指令的动作模式不同(程序模式)。	请确认控制器的动作模式。
16#00002204	指令的动作模式不同(调试模式)。	请确认控制器的动作模式。
16#00002205	指令的动作模式不同(监控模式)。	请确认控制器的动作模式。
16#00002206	指令的动作模式不同(运行模式)。	请确认控制器的动作模式。
16#00002207	指定节点非管理站。	请确认网络的管理站节点。
16#00002208	指令的动作模式不同。	请确认步活性状态。
16#00002211	单元为BUSY状态。	请增加重新发送次数的设定，或调整系统避免通信集中在相应单元上。
16#00002301	无文件装置。	请安装介质。或者，请执行EM格式化。
16#00002302	无文件存储器。	请确认文件存储器的安装。
16#00002303	未内置时钟。	请确认机型的规格。
16#00002401	协议宏数据的SUM值异常或正在传送数据。	请通过CX-Protocol重新传送协议宏数据。
16#00002502	处理的对象存储器存在异常。	请将正确数据重新传送至对象存储器。
16#00002503	登录的I/O单元构成与实际的单元构成不同。	请确认I/O单元构成。
16#00002504	登录的I/O点数或远程I/O点数超过最大值。	请对I/O点数、远程I/O点数进行正确设定。
16#00002505	CPU单元与CPU高功能单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002506	机架No.、单元编号、I/O地址中的某一设定重复。	请对重复的内容进行重新设定。
16#00002507	CPU单元与I/O单元间的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#00002509	SYSMAC BUS/2数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250A	CPU高功能单元的数据传送存在异常。	请确认单元及连接电缆。在解除异常后，请执行异常解除指令。
16#0000250D	通道设定重复。	请对I/O通道进行正确设定。

值	异常内容	处理方法
16#0000250F	存储器异常。	· 内部存储器时，请写入正确数据后重新执行指令。 · 存储卡及EM文件存储器时，请执行扩展存储器的格式化指令。 · 进行上述处理后仍无法解除异常时，请更换存储器。
16#00002510	末站的设定异常。	请对末站进行正确设定。
16#00002601	保护解除中。	无需解除保护。
16#00002602	密码不一致。	请指定正确的密码。
16#00002604	保护中。	· 请在解除写保护后，重新执行指令。 · 请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002605	服务执行中。	请等待至执行中的服务结束或停止服务后重新执行指令。
16#00002606	服务停止中。	请根据需要执行相应服务。
16#00002607	无执行权。	· 请通过数据链接参加节点执行。 · 重启后仍发生异常时，请更换控制器。
16#00002608	未设定环境。	请进行必要的设定。
16#00002609	未设定必要项目。	请设定必要项目。
16#0000260A	指定编号已定义。	请变更成未登录的动作、过渡编号后，重新执行指令。
16#0000260B	无法解除异常。	请解除造成异常的原因后，执行异常解除指令。
16#00003001	无访问权。	请等待至访问权释放后，重新执行指令。
16#00004001	服务被中断。	请在解除服务被中断的原因后，重新执行指令。

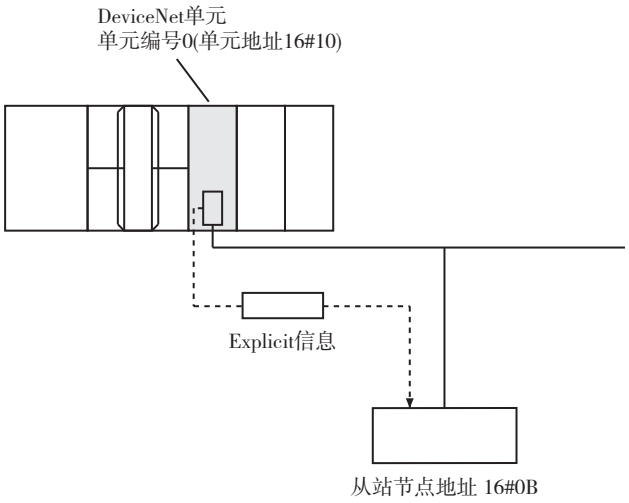
(注) 除上表值以外，终止符的位6、7、15的值可能会变为TRUE。位6、7的值为TRUE时，表示发送目标的CPU单元发生了异常。位15的值为TRUE时，表示网络中继时发生了异常。

示例程序

经由DeviceNet单元发送Explicit信息。经由单元地址16#10的DeviceNet，从节点地址16#0B的从站读取供应商ID。

主要通信规格如下所述。

项目	内容
DeviceNet单元的单元地址	16#10
从站节点地址	16#0B
服务代码	16#0E
等级ID	1
实例ID	1
属性ID	1
超时时间	2.0s
重新发送次数	2



指令数组SendDat[]及响应保存数组RecvDat[]的内容如下所述。

指令数组　BYTE型数组

数组元素	项目	本示例的内容	值
SendDat[0]	命令代码	发送Explicit信息的命令代码为16#2801。	BYTE#16#28
SendDat[1]			BYTE#16#01
SendDat[2]	从站节点地址	节点地址为16#0B。	BYTE#16#0B
SendDat[3]	服务代码	读取指定属性值(Get Arrtibute Single)的服务代码为16#0E。	BYTE#16#0E
SendDat[4]	等级ID	Identity对象的等级ID为16#0001。	BYTE#16#00
SendDat[5]			BYTE#16#01
SendDat[6]	实例ID	–	BYTE#16#00
SendDat[7]			BYTE#16#01
SendDat[8]	属性ID	供应商ID的属性ID(Vendor ID)为16#01。	BYTE#16#01

响应保存数组 BYTE型数组

数组元素	项目	本示例的内容
RecvDat[0]	命令代码	发送Explicit信息的命令代码为16#2801。
RecvDat[1]		
RecvDat[2]	终止符	正常结束时为16#0000。
RecvDat[3]		
RecvDat[4]	从站节点地址之后的接收字节数	4字节。
RecvDat[5]		
RecvDat[6]	从站节点地址	正常结束时，节点地址为16#0B。
RecvDat[7]	服务代码	正常结束的服务代码为16#8E。
RecvDat[8]	供应商ID	从站的供应商ID。
RecvDat[9]		

全局变量的定义

全局变量

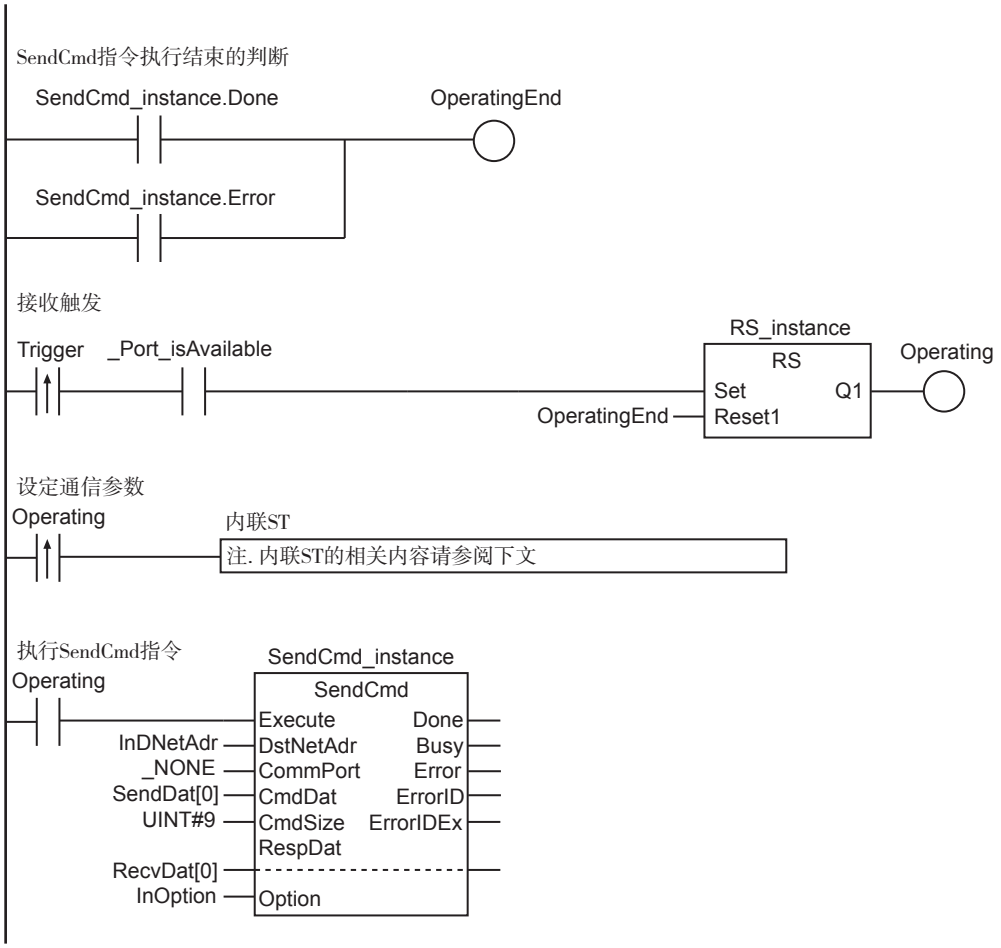
名称	数据类型	分配对象*1	注释
DeviceNet_OnlineSta	BOOL	IOBus://rack#0/slot#0/Unit2Sta/OnlineSta	在线

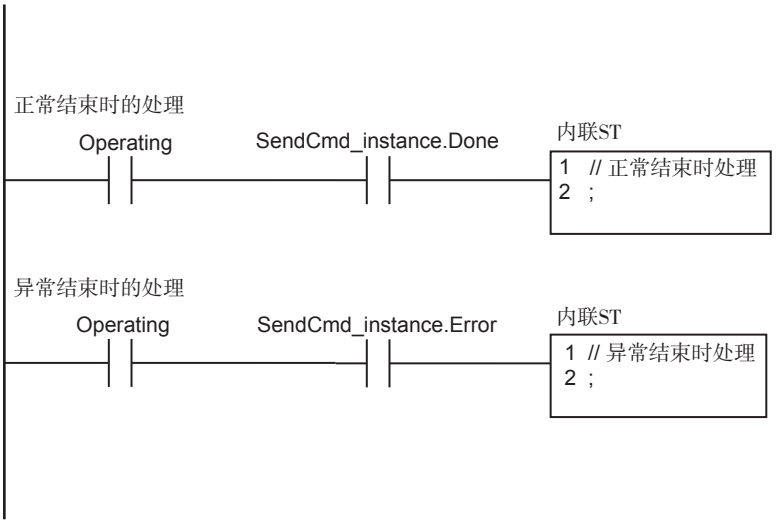
*1 是将串行通信单元安装于机架编号0的插槽编号0上时的分配对象。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	InDNetAdr	_sDNET_ADR	(NetNo:=0, NodeNo:=0, UnitNo:=16#0)	发送目标网址
	InOption	_sRESPONSE	(isNonResp:=FALSE, TimeOut:=0, Retry:=0)	响应
	SendDat	ARRAY[0..8] OF BYTE	[9(16#0)]	发送数据
	RecvDat	ARRAY[0..9] OF BYTE	[10(16#0)]	接收数据
	RS_instance	RS		
	SendCmd_instance	SendCmd		

外部变量	名称	数据类型	注释
	_Port_isAvailable	BOOL	网络通信指令可执行标志





- 内联ST的内容

```
InDNetAdr.NetNo     :=USINT#0;    // 设定网址
InDNetAdr.NodeNo    :=USINT#0;
InDNetAdr.UnitNo    :=BYTE#16#10;
InOption.isNonResp  :=FALSE;      // 设定响应
InOption.TimeOut    :=UINT#20;
InOption.Retry      :=USINT#2;
SendDat[0]          :=BYTE#16#28; // 设定指令数组
SendDat[1]          :=BYTE#16#01;
SendDat[2]          :=BYTE#16#0B;
SendDat[3]          :=BYTE#16#0E;
SendDat[4]          :=BYTE#16#00;
SendDat[5]          :=BYTE#16#01;
SendDat[6]          :=BYTE#16#00;
SendDat[7]          :=BYTE#16#01;
SendDat[8]          :=BYTE#16#01;
```

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	InDNetAdr	_sDNET_ADR	(NetNo:=0, NodeNo:=0, UnitNo:=16#0)	发送目标网址
	InOption	_sRESPONSE	(isNonResp:=FALSE, TimeOut:=0, Retry:=0)	响应
	SendDat	ARRAY[0..8] OF BYTE	[9(16#0)]	发送数据
	RecvDat	ARRAY[0..9] OF BYTE	[10(16#0)]	接收数据
	SendCmd_instance	SendCmd		

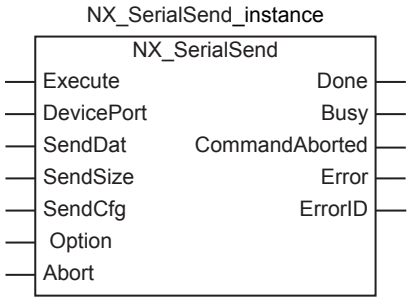
外部变量	名称	数据类型	注释
	DeviceNet_OnlineSta	BOOL	在线
	_Port_isAvailable	BOOL	网络通信指令可执行标志

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Port_isAvailable=TRUE)
     AND (DeviceNet_OnlineSta=TRUE) ) THEN
   OperatingStart:=TRUE;
   Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 设定通信参数和SendCmd指令初始化
IF (OperatingStart=TRUE) THEN
   SendCmd_instance(
      Execute    :=FALSE,
      DstNetAdr :=InDNetAdr,
      CommPort :=_NONE,
      CmdDat     :=SendDat[0],
      CmdSize    :=UINT#9,
      RespDat    :=RecvDat[0],
      Option     :=InOption);
   InDNetAdr.NetNo      :=USINT#0;     // 设定网址
   InDNetAdr.NodeNo     :=USINT#0;
   InDNetAdr.UnitNo     :=BYTE#16#10;
   InOption.isNonResp   :=FALSE;       // 设定响应
   InOption.TimeOut     :=UINT#20;
   InOption.Retry       :=USINT#2;
   SendDat[0]           :=BYTE#16#28;  // 设定指令数组
   SendDat[1]           :=BYTE#16#01;
   SendDat[2]           :=BYTE#16#0B;
   SendDat[3]           :=BYTE#16#0E;
   SendDat[4]           :=BYTE#16#00;
   SendDat[5]           :=BYTE#16#01;
   SendDat[6]           :=BYTE#16#00;
   SendDat[7]           :=BYTE#16#01;
   SendDat[8]           :=BYTE#16#01;
   OperatingStart       :=FALSE;
END_IF;
```

```
// 执行SendCmd指令
IF (Operating=TRUE) THEN
  SendCmd_instance(
    Execute    :=TRUE,
    DstNetAdr :=InDNetAdr,
    CommPort :=_NONE,
    CmdDat     :=SendDat[0],
    CmdSize    :=UINT#9,
    RespDat    :=RecvDat[0],
    Option     :=InOption);

  IF (SendCmd_instance.Done=TRUE) THEN
    // 正常结束时处理
    Operating:=FALSE;
  END_IF;

  IF (SendCmd_instance.Error=TRUE) THEN
    // 异常结束时处理
    Operating:=FALSE;
  END_IF;
END_IF;
```

NX_SerialSend

无协议从NX系列通信接口单元及扩展板的串行端口发送数据。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialSend	无协议数据的发送	FB		NX_SerialSend_instance(Execute, DevicePort, SendDat, SendSize, SendCfg, Option, Abort, Done, Busy, CommandAborted, Error, ErrorID);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
SendDat[]数组	发送数据数组		发送数据数组	遵从数据类型	–	*1
SendSize	发送数据大小		发送数据大小	0～4096	字节	0
SendCfg	发送数据的附加条件		发送数据的附加条件	–	–	–
Option	选项		选项	–	–	–
Abort	中断		中断指令执行	遵从数据类型	–	FALSE
Command Aborted	中断完成	输出	中断完成	遵从数据类型	–	–

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
SendDat[] 数组		○																		
SendSize							○													
SendCfg	结构体_sSERIAL_CFG 详情参阅功能说明																			
Option	结构体_ s SERIAL_SEND_OPTION 详情参阅功能说明																			
Abort	○																			
Command Aborted	○																			

功能

无协议从NX系列通信接口单元及扩展板的指定端口发送数据。

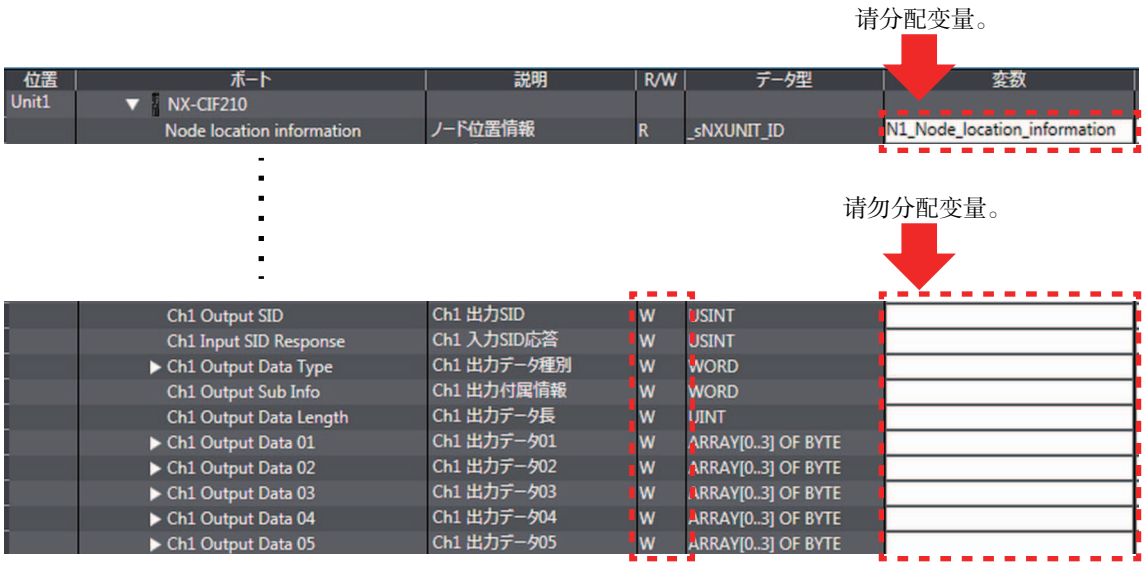
输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType”指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit”指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。
指定扩展板时，使用 “OptBoard”指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为“W”的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

在输入变量 “SendDat”指定的数据中，发送输入变量 “SendSize”指定的大小。
此时，如“SendSize”的值为0则不发送。执行指令时，“Busy”不会变为TRUE，“Done”的值变为TRUE。

发送数据中加上起始符和结束符时，使用输入变量 “SendCfg”进行设定。
输入变量 “SendCfg”的数据类型为结构体_sSERIAL_CFG。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
SendCfg	发送数据的附加条件	发送数据的附加条件	_sSERIAL_CFG	–	–	–
StartTrig	有无起始符	有无起始符	_eSERIAL_START	_SERIAL_START_NONE _SERIAL_START_STARTCODE1 _SERIAL_START_STARTCODE2	–	_SERIAL_START_NONE
StartCode	起始符	起始符	BYTE[2]	遵从数据类型	–	[2(16#0)]
EndTrig	有无结束符	有无结束符	_eSERIAL_END	_SERIAL_END_NONE _SERIAL_END_ENDCODE1 _SERIAL_END_ENDCODE2 _SERIAL_END_TERMINATION_CHAR _SERIAL_END_RCV_SIZE	–	_SERIAL_END_NONE
EndCode	结束符	结束符	BYTE[2]	遵从数据类型	–	[2(16#0)]
RcvSizeCfg	接收大小	本指令不使用	UINT	0 ~ 4096	字节	0

“StartTrig”的数据类型为枚举体_eSERIAL_START。
枚举体_eSERIAL_START的枚举元素的含义如下所述。

枚举元素	含义
_SERIAL_START_NONE	无
_SERIAL_START_STARTCODE1	1字节代码
_SERIAL_START_STARTCODE2	2字节代码

“EndTrig”的数据类型为枚举体_eSERIAL_END。
枚举体_eSERIAL_END的枚举元素的含义如下所述。

枚举元素	含义
_SERIAL_END_NONE	无
_SERIAL_END_ENDCODE1	1字节代码
_SERIAL_END_ENDCODE2	2字节代码
_SERIAL_END_TERMINATION_CHAR	停止条件
_SERIAL_END_RCV_SIZE	接收大小

起始符和结束符的动作详情请参阅 “起始符和结束符的动作(P.2–1219)”。

需延迟控制器对NX系列通信接口单元的发送时，使用输入变量 “Option.SendDelay”，以0.01s为单位指定延迟时间。
输入变量 “Option”的数据类型为结构体_sSERIAL_SEND_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	选项	选项	_sSERIAL_SEND_OPTION	–	–	–
SendDelay	发送延迟时间	发送延迟时间的指定	UINT	遵从数据类型	0.01s	0

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

起始符和结束符的动作

使用 “SendCfg.StartTrig”和 “SendCfg.EndTrig”，指定附加在发送数据中的起始符和结束符的条件。
在发送数据中附加起始符、结束符时，请在输入变量 “SendSize”中设定不含起始符、结束符的值。
“StartTrig”和 “EndTrig”的动作如下所示。

“StartTrig”的值	动作
_SERIAL_START_NONE	–
_SERIAL_START_STARTCODE1	在SendDat的开头附加起始符进行发送 例：STX
_SERIAL_START_STARTCODE2	

“EndTrig”的值	动作
_SERIAL_END_NONE	–
_SERIAL_END_ENDCODE1	在SendDat的结尾附加结束符进行发送 例：ETX
_SERIAL_END_ENDCODE2	
_SERIAL_END_TERMINATION_CHAR	异常
_SERIAL_END_RCV_SIZE	异常

指令执行的中断

在指令执行过程中将 “Abort”设为TRUE时，将中断指令的执行。
中断指令执行时，“CommandAborted”将变为TRUE。即使在发送中，也将中断指令。
指令执行未及时中断时，“Done”将变为TRUE，指令正常结束。
将 “Abort”、“Execute”均设为TRUE时，“CommandAborted”将变为TRUE。

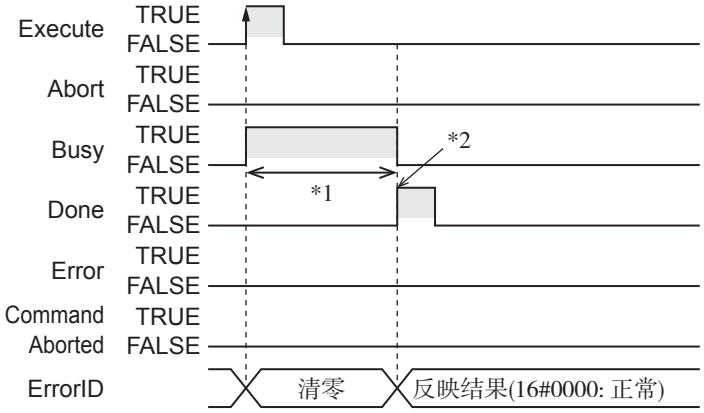
中断动作只结束 “Busy”处理，不会清除发送缓存。需清除缓存时，请使用NX_SerialBufClear指令。

时序图

时序图如下所示。

● 正常结束时(“SendDelay”为 “0”(0s)时)

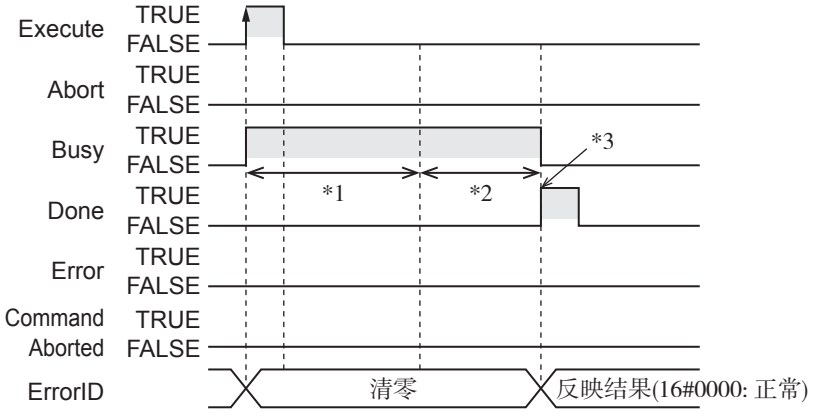
“SendDelay”为 “0”(0s)时，动作如下。



*1 发送处理
*2 发送完成

● 正常结束时(“SendDelay”为 “100”(1s)时)

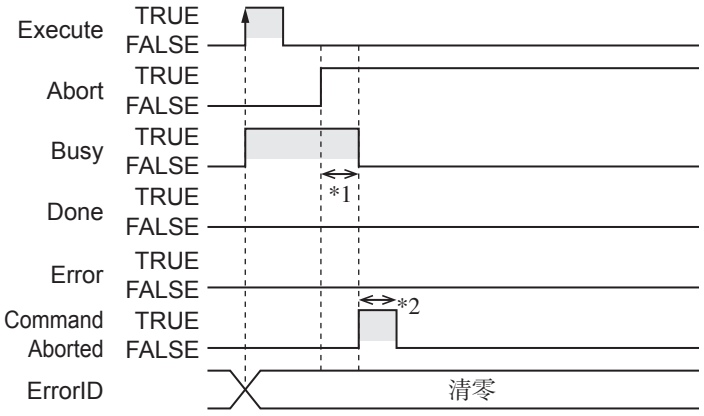
“SendDelay”为 “100”(1s)时，动作如下。



*1 发送延迟时间1s
*2 发送处理
*3 发送完成

● 中断执行时(“Busy”为TRUE时)

“Busy”TRUE的状态下，将 “Abort”设为TRUE时，动作如下。

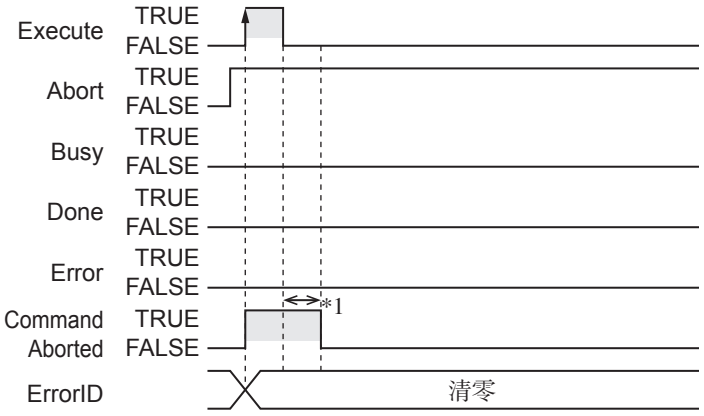


*1 中断处理

*2 1个任务周期后FALSE

● 中断执行时(“Execute”为TRUE时)

将 “Abort”和 “Execute”均设为TRUE时，动作如下。



*1 1个任务周期后FALSE

相关的系统定义变量

自动生成设备名称为 “E001”的EtherCAT耦合器单元的设备变量名称时。

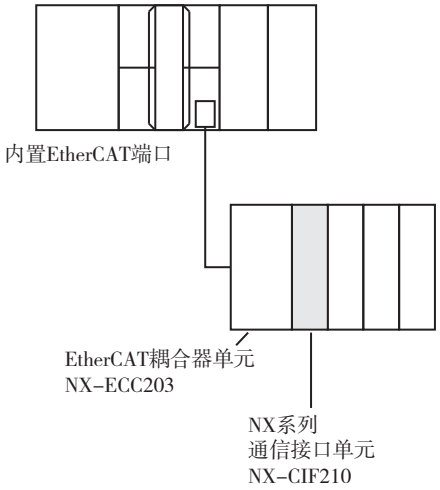
变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

使用注意事项

- “Abort”为FALSE的状态下，即使 “Execute”的值为FALSE或执行时间超过任务周期，本指令仍将继续处理直至最后。
  请通过 “Done”的值是否变为 TRUE 确认处理是否正常结束。指令执行中 “Abort”变为 TRUE 时，“CommandAborted”或 “Done”将变为TRUE。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “SendSize” 、“SendCfg.StartTrig” 、“SendCfg.EndTrig” 、“DevicePort.DevicePortType” 、“DevicePort.PortNo”的值超过有效范围时。
  - “SendDat”指定的数组变量小于 “SendSize”指定的大小时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - “DevicePort”的数据类型错误时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSend指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusWrite指令。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。
  - 指定扩展板的串行通信模式为 “无协议”以外时。

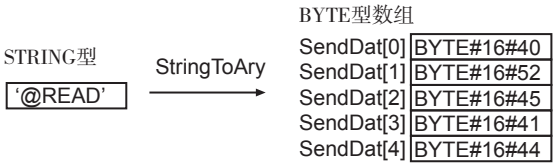
示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。



对与NX-CIF210的串行端口2连接的读码器发送无协议指令。发送指令为获得场景编号的指令‘@READ’。

发送指令使用StringToAry指令，将字符串 ‘@READ’逐个字符转换为字符代码，保存在SendDat[]的各数组元素中。



无起始符，结束符为16#OD(CR)。

NX-CIF210的设定如下所示。

项目名	设定值
端口2：传送速度	38400bps
端口2：数据长度	8位
端口2：奇偶校验	无
端口2：停止位	1 位
端口2：流程控制	无

全局变量的定义

全局变量

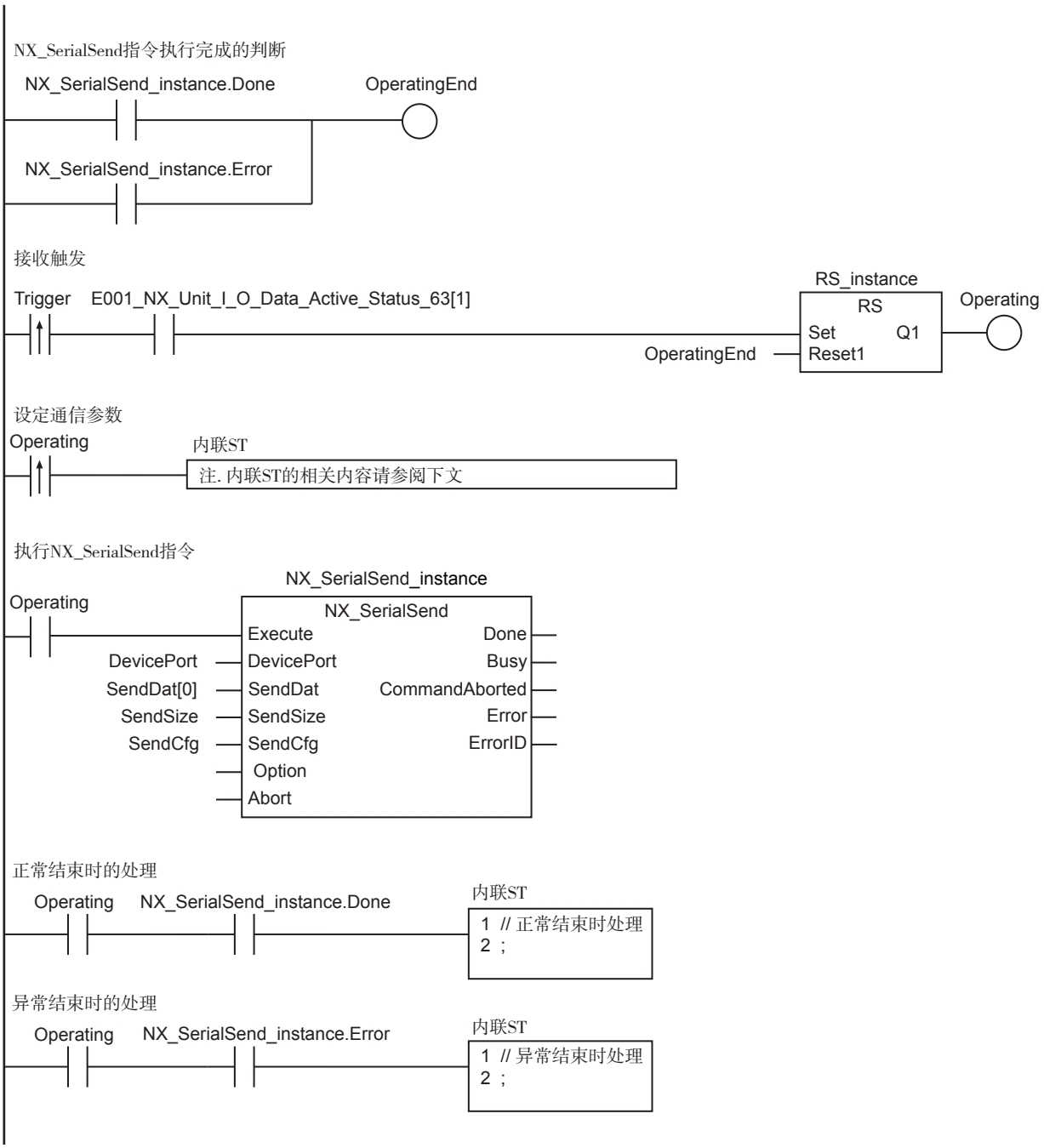
名称	数据类型	分配对象	注释
E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	ECAT://node#1/NX Unit I/O Data Active Status 125	可否使用63台NX单元的I/O数据
N1_Node_location_information	_sNXUNIT_ID	–	NX-CIF210指定的设备变量*1

*1 使用Sysmac Studio右击NX系列从站终端的单元，选择 “显示节点位置端口”，设定设备变量。详情请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

LD

内部变量	名称	数据类型	初始值	注释
	OperationEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	DevicePort	_sDEVICE_PORT		指定端口
	SendDat	ARRAY [0..5] OF BYTE	[6(16#0)]	发送数据
	SendSize	UINT	0	发送数据大小
	RS_instance	RS		
	NX_SerialSend_instance	NX_SerialSend		
	SendCfg	_sSERIAL_SEND_CFG		
	StartTrig	_eSERIAL_START	_SERIAL_START_NONE	无起始符
	StartCode	BYTE[2]	[2(16#0)]	
	EndTrig	_eSERIAL_END	_SERIAL_END_ENDCODE1	有结束符
	EndCode	BYTE[2]	[16#0D,16#00]	16#0D(CR)

外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1” 时，使用E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量



● 内联ST的内容

```
DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
DevicePort.NxUnit:=N1_Node_location_information;
DevicePort.PortNo:=2;
StringToAry(In:= ‘@READ’, AryOut:=SendDat[0]);
SendSize := UINT#10#5;
```

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	DevicePort	_sDEVICE_PORT		指定端口
	SendDat	ARRAY [0..5] of BYTE	[6(16#0)]	发送数据
	SendSize	UINT	0	发送数据大小
	NX_SerialSend_instance	NX_SerialSend		
	SendCfg	_sSERIAL_CFG		
	StartTrig	_eSERIAL_START	_SERIAL_START_NONE	无起始符
	StartCode	BYTE[2]	[2(16#0)]	
	EndTrig	_eSERIAL_END	_SERIAL_END_ENDCODE1	有结束符
	EndCode	BYTE[2]	[16#0D,16#00]	16#0D(CR)

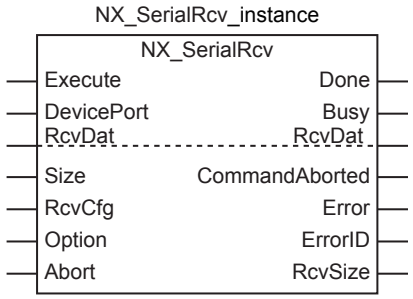
外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1”时，使用E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE)
   AND (E001_NX_Unit_I_O_Data_Active_Status_63[1]) AND (NX_SerialSend_instance.Busy=FALSE) ) THEN
      OperatingStart:=TRUE;
      Operating:=TRUE;
      DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
      DevicePort.NxUnit:=N1_Node_location_information;
      DevicePort.PortNo:=2;
END_IF;
LastTrigger:=Trigger;

// 设定通信参数和NX_SerialSend指令初始化
IF (OperatingStart=TRUE) THEN
      NX_SerialSend_instance(
                  Execute:=FALSE,
                  DevicePort:=DevicePort;
                  SendDat:=SendDat[0],
                  SendSize:=UINT#1,
                  SendCfg:=SendCfg);
      StringToAry(In:= ‘@READ’, AryOut:=SendDat[0]);
      SendSize:=UINT#10#5;
      OperatingStart:=FALSE;
END_IF;

// 执行NX_SerialSend指令
IF (Operating=TRUE) THEN
      NX_SerialSend_instance(
                  Execute:=TRUE,
                  DevicePort:=DevicePort, // 指定端口
                  SendDat:=SendDat[0],   // 发送数据
                  SendSize:=SendSize,    // 发送数据大小
                  SendCfg:=SendCfg);     // 指定结束符
```

```
        IF (NX_SerialSend_instance.Done=TRUE) THEN
            // 正常结束时处理

            Operating:=FALSE;
        END_IF;

        IF (NX_SerialSend_instance.Error=TRUE) THEN
            // 异常结束时处理

            Operating:=FALSE;
        END_IF;
    END_IF;
```

NX_SerialRcv

无协议从NX系列通信接口单元及扩展板的串行端口读取数据。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialRcv	无协议数据的接收	FB		NX_SerialRcv_instance(Execute, DevicePort, RcvDat, Size, RcvCfg, Option, Abort, Done, Busy, CommandAborted, Error, ErrorID, RcvSize);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
Size	保存容量		按字节数指定 “RcvDat” 的大小	1 ~ 4096	字节	1
RcvCfg	接收完成设定		接收完成设定	–	–	–
Option	选项		选项	–	–	–
Abort	中断		指令执行的中断	–	–	FALSE
RcvDat[] 数组	接收数据	输入输出	保存从接收缓存所接收数据的变量	遵从数据类型	–	–
Command Aborted	中断完成	输出	中断完成	遵从数据类型	–	–
RcvSize	接收大小		实际从接收缓存接收的数据大小	0 ~ 4096	字节	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
Size							○													
RcvCfg	结构体_sSERIAL_CFG 详情参阅功能说明																			
Option	结构体_sSERIAL_RCV_OPTION 详情参阅功能说明																			
Abort	○																			
RcvDat[] 数组		○																		
Command Aborted	○																			
RcvSize							○													

功能

无协议从NX系列通信接口单元及扩展板的指定端口读取数据。

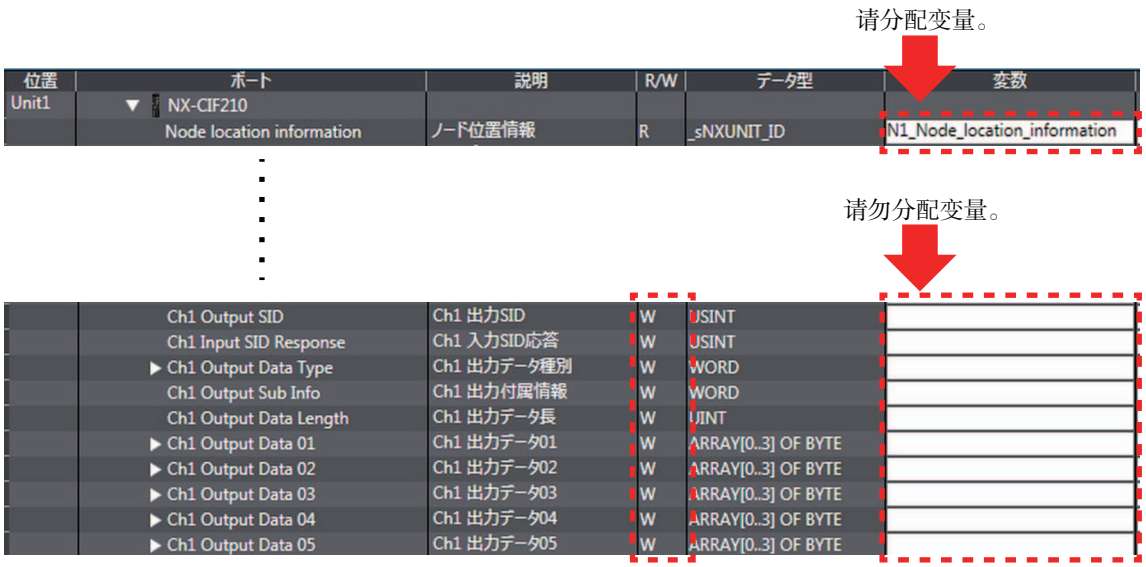
输入变量 “DevicePort” 的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType” 指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit” 指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。
指定扩展板时，使用 “OptBoard” 指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为“W”的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

单元接收的数据首先保存在接收缓存中。
使用输入输出变量 “RcvDat”指定保存从接收缓存接收的数据的变量。
使用输入变量 “Size”按字节数指定 “RcvDat”的大小。
实际从接收缓存接收的数据大小由输出变量 “RcvSize”表示。

接收数据中包含起始符和结束符时，使用输入变量 “RcvCfg”进行设定。
输入变量 “RcvCfg”的数据类型为结构体_sSERIAL_CFG。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
RcvCfg	接收完成设定	接收完成设定	_sSERIAL_CFG	–	–	–
StartTrig	有无起始符	有无起始符	_eSERIAL_START	_SERIAL_START_NONE _SERIAL_START_STARTCODE1 _SERIAL_START_STARTCODE2	–	_SERIAL_START_NONE
StartCode	起始符	起始符	BYTE[2]	遵从数据类型	–	[2(16#0)]
EndTrig	有无结束符	有无结束符	_eSERIAL_END	_SERIAL_END_NONE _SERIAL_END_ENDCODE1 _SERIAL_END_ENDCODE2 _SERIAL_END_TERMINATION_CHAR _SERIAL_END_RCV_SIZE	–	_SERIAL_END_NONE
EndCode	结束符	结束符	BYTE[2]	遵从数据类型	–	[2(16#0)]
RcvSizeCfg	接收大小	结束符为_SERIAL_END_RCV_SIZE时，指定接收大小	UINT	0 ~ 4096	字节	0

“StartTrig”的数据类型为枚举体_eSERIAL_START。
枚举体_eSERIAL_START的枚举元素的含义如下所述。

枚举元素	含义
_SERIAL_START_NONE	无
_SERIAL_START_STARTCODE1	1字节代码
_SERIAL_START_STARTCODE2	2字节代码

“EndTrig”的数据类型为枚举体_eSERIAL_END。
枚举体_eSERIAL_END的枚举元素的含义如下所述。

枚举元素	含义
_SERIAL_END_NONE	无
_SERIAL_END_ENDCODE1	1字节代码
_SERIAL_END_ENDCODE2	2字节代码
_SERIAL_END_TERMINATION_CHAR	停止条件
_SERIAL_END_RCV_SIZE	接收大小

起始符和结束符的动作详情请参阅 “起始符和结束符的动作(P.2–1219)”。

指定选项时，使用输入变量 “Option”进行指定。
输入变量 “Option”的数据类型为结构体_sSERIAL_RCV_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	选项	选项	_sSERIAL_RCV_OPTION	–	–	–
TimeOut*1	超时时间	超时时间	UINT	遵从数据类型	0.1s	20
LastDatRcv (Reserved)	最终数据接收	最终数据接收	BOOL	FALSE*2	–	FALSE
ClearBuf	接收缓存清除条件	接收缓存清除条件	BOOL	遵从数据类型	–	FALSE

*1 指定时间内未正常结束时，会发生异常。
此外，将 “TimeOut”指定成 “0”时，将持续等待直至处理完成。
*2 请始终设为FALSE。

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

起始符和结束符的动作

使用输入变量 “RcvCfg.StartTrig”在接收数据中指定起始符的条件，使用输入变量 “RcvCfg.EndTrig”在接收数据中指定结束符的条件。
基于 “StartTrig”与 “EndTrig”组合的动作如下所述。

StartTrig	EndTrig	动作
_SERIAL_START_NONE	_SERIAL_END_NONE	接收接收缓存中的数据。接收缓存内无接收数据时，将输出变量 “RcvSize”作为0字节输出后，接收指令将正常结束。 指定该条件时，将按照保存大小读取接收缓存中的数据。
	_SERIAL_END_ENDCODE1 _SERIAL_END_ENDCODE2	对接收缓存内的数据，执行从开头至结束符的接收处理。 例：ETX
	_SERIAL_END_TERMINATION_CHAR	对接收缓存内的数据，执行从开头至检测到结尾数据的接收处理。*1
	_SERIAL_END_RCV_SIZE	对接收缓存内的数据，从开头起按 “RcvSizeCfg”指定的接收大小进行接收处理。 等待至缓存中保存了接收大小的数据。
_SERIAL_START_STARTCODE1 _SERIAL_START_STARTCODE2	_SERIAL_END_NONE	从起始符起，接收接收缓存中的所有数据。
	_SERIAL_END_ENDCODE1 _SERIAL_END_ENDCODE2	执行从起始符至结束符的接收处理。 例：ETX
	_SERIAL_END_TERMINATION_CHAR	执行从起始符至检测到结尾数据的接收处理。*1
	_SERIAL_END_RCV_SIZE	从起始符起按 “RcvSize”指定的接收大小进行数据接收处理。 等待至缓存中保存了接收大小的数据。

*1 将通信接口单元检测到结尾的字符数设为 “0(不检测结尾)”时，持续接收数据，直至达到输入变量 “Size”指定的保存大小为止。

使用注意事项

指定扩展板时若选择 “_SERIAL_END_TERMINATION_CHAR”，则会发生异常。

接收数据保存位置空间不足时的动作

如下所示，输入变量“Size”指定的接收数据保存位置的空间大小小于接收数据时，动作因起始符、结束符的组合而异。

StartTrig	EndTrig	动作
_SERIAL_START_NONE	_SERIAL_END_NONE	正常完成
	_SERIAL_END_ENDCODE1	接收异常完成 例：ETX
	_SERIAL_END_ENDCODE2	
	_SERIAL_END_TERMINATION_CHAR	接收异常完成 *1
	_SERIAL_END_RCV_SIZE	· 输入值检查错误导致异常完成 · 无法接收数据
_SERIAL_START_STARTCODE1 _SERIAL_START_STARTCODE2	_SERIAL_END_NONE	接收异常完成
	_SERIAL_END_ENDCODE1	接收异常完成 例：ETX
	_SERIAL_END_ENDCODE2	
	_SERIAL_END_TERMINATION_CHAR	接收异常完成 *1
	_SERIAL_END_RCV_SIZE	· 输入值检查错误导致异常完成 · 无法接收数据

*1 指定扩展板时，会发生异常。

按照接收数据保存位置 “RcvDat”的大小进行接收处理，无法保存的部分保持在接收缓存中。保持的接收数据可通过下一次的SerialRcv指令执行接收处理。
例如，接收缓存有10字节，接收数据保存位置 “RcvDat”为5字节时，对5字节进行接收处理，剩余5字节则保持在接收缓存中。输出变量 “RcvSize”将输出已保存的5字节数据。

接收缓存		接收数据保存位置 RcvDat[ ]
第1字节	执行接收处理。	1
第2字节		2
第3字节		3
第4字节		4
第5字节		5
第6字节	“RcvDat”中无法保存，因此保持在接收缓存中。 通过下一次的NX_SerialRcv指令执行接收处理。	–
第7字节		
第8字节		
第9字节		
第10字节		

指令执行的中断

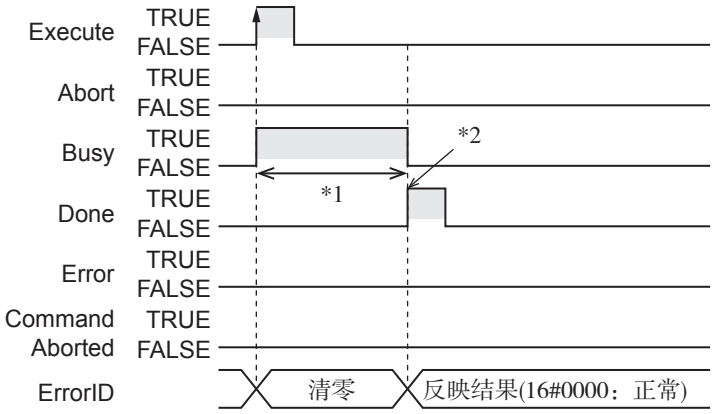
在指令执行过程中将 “Abort”设为TRUE时，将中断指令的执行。
中断指令执行时，“CommandAborted”将变为TRUE。
指令执行未及时中断时，“Done”将变为TRUE，指令正常结束。
将 “Abort”、“Execute”均设为TRUE时，“CommandAborted”将变为TRUE。

中断动作只结束 “Busy”处理，不会清除接收缓存。需清除缓存时，请使用NX_SerialBufClear指令。

时序图

时序图如下所示。

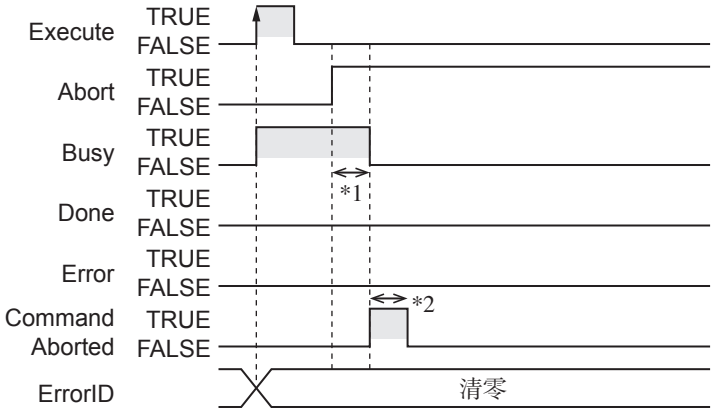
● 正常结束时



*1 接收处理
*2 无协议接收数据

● 中断执行时(“Busy”为TRUE时)

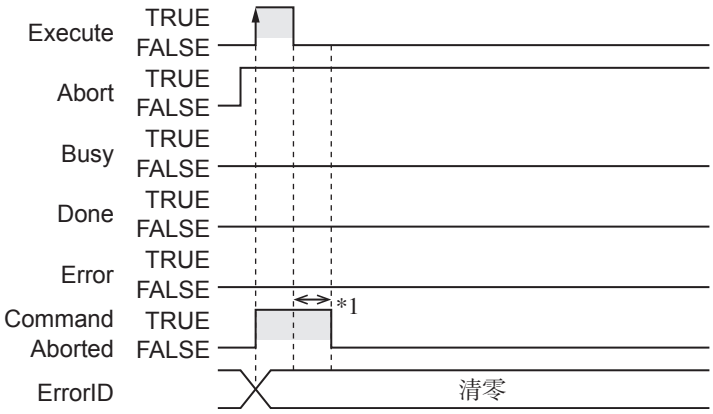
“Busy”TRUE的状态下，将 “Abort”设为TRUE时，动作如下。



*1 中断处理
*2 1个任务周期后FALSE

● 中断执行时(“Execute” 为TRUE时)

将 “Abort” 和 “Execute” 均设为TRUE时，动作如下。



*1 1个任务周期后FALSE

相关的系统定义变量

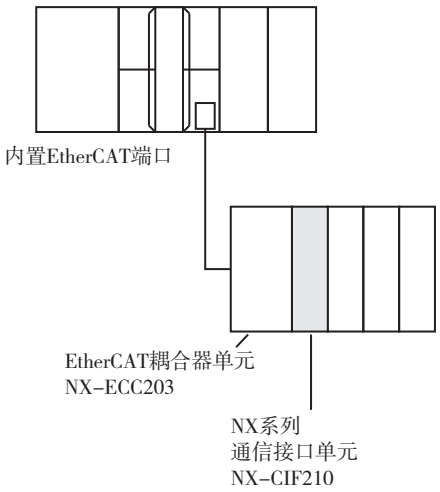
变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

使用注意事项

- “Abort”为FALSE的状态下，即使 “Execute”的值为FALSE或执行时间超过任务周期，本指令仍将继续处理直至最后。
  请通过 “Done”的值是否变为 TRUE 确认处理是否正常结束。指令执行中 “Abort”变为 TRUE 时，“CommandAborted”或 “Done”将变为TRUE。
- “RcvCfg.EndTrig”为_SERIAL_END_RCV_SIZE，输入变量 “RcvCfg.RcvSizeCfg”的值为 “0”时，不会接收数据。此时执行指令时，“Done”的值为TRUE。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX系列通信接口单元重启时，可能会发生 “CIF单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - 使用 “RcvCfg.EndTrig”指定了_SERIAL_END_RCV_SIZE的情况下，“RcvCfg.RcvSizeCfg”超过范围时。
  - “Size”、“DevicePort.DevicePortType”、“DevicePort.PortNo”的值超过有效范围时。
  - “Option.LastDatRcv”为TRUE时。
  - 输入输出变量 “RcvDat”指定的数组变量小于输入变量 “Size”指定的大小时。
  - “Size”指定的 “RcvDat”的接收数据保存大小比接收数据小时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - “DevicePort”的数据类型错误时。
  - “DevicePort”指定了扩展板的情况下，RcvCfg.EndTrig选择了_SERIAL_END_TERMINATION_CHAR时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 接收缓存已满时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。对象指令为NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  - 接收数据发生了奇偶校验错误时。
  - 接收数据发生了结构错误时。
  - 接收数据发生了超程错误时。
  - 超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。
  - 指定扩展板的串行通信模式为 “无协议”以外时。

示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。



获取与NX-CIF210的串行端口2连接的读码器的读取结果。
接收数据保存在输入输出变量 “RecvDat”中。无起始符，结束符为16#0D(CR)。

NX-CIF210的设定如下所示。

项目名	设定值
端口2：传送速度	38400bps
端口2：数据长度	8位
端口2：奇偶校验	无
端口2：停止位	1 位
端口2：流程控制	无

全局变量的定义

全局变量

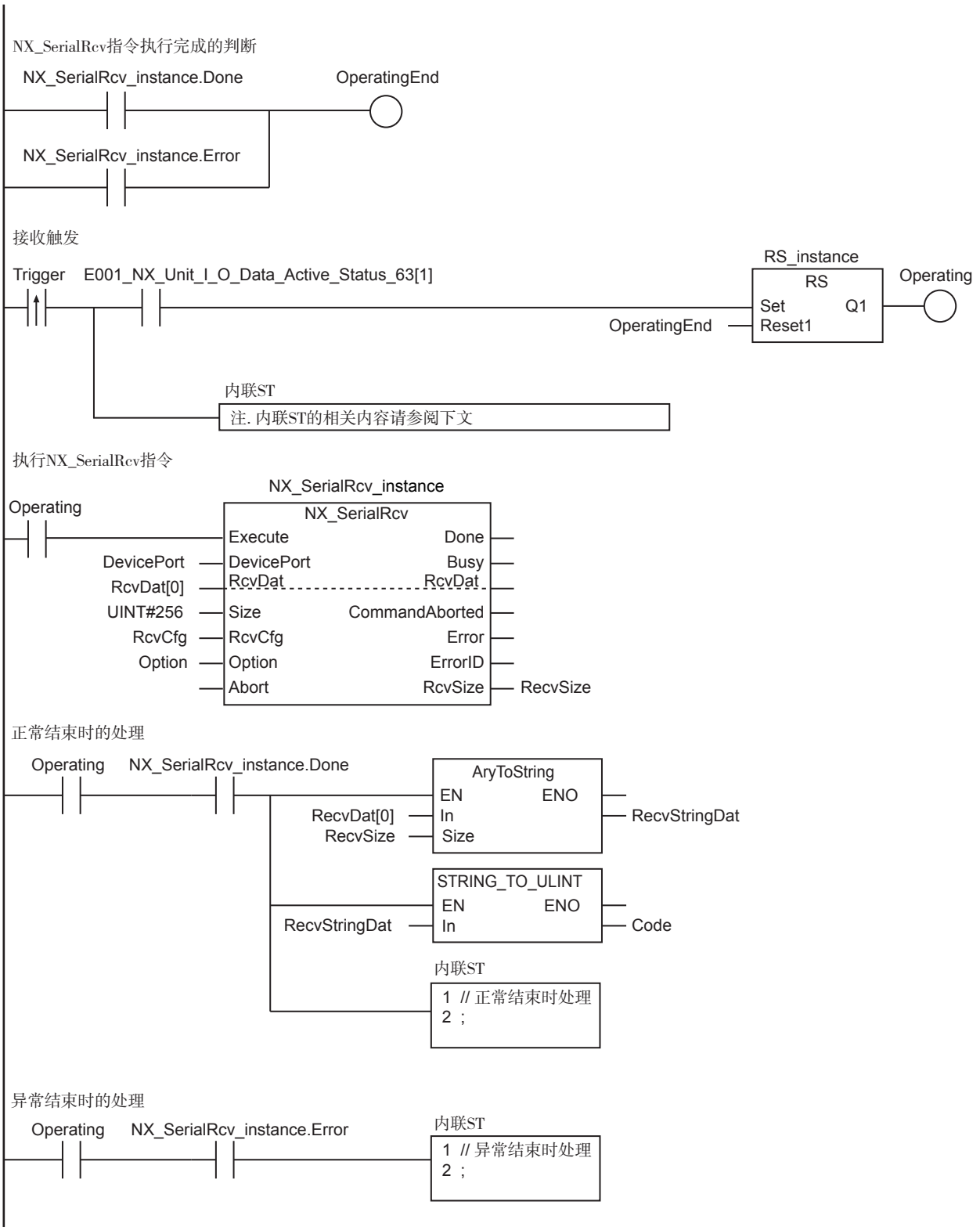
名称	数据类型	分配对象	注释
E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	ECAT://node#1/NX Unit I/O Data Active Status 125	可否使用63台NX单元的I/O数据
N1_Node_location_information	_sNXUNIT_ID	–	NX-CIF210指定的设备变量*1

*1 使用Sysmac Studio右击NX系列从站终端的单元，选择 “显示节点位置端口”，设定设备变量。
详情请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

LD

内部变量	名称	数据类型	初始值	注释
	OperationEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	DevicePort	_sDEVICE_PORT		指定端口
	RecvDat	ARRAY [0..255] of BYTE	[256(16#0)]	接收数据
	RecvSize	UINT	0	接收数据大小
	RecvStringDat	STRING[257]	''	
	Code	ULINT	0	条形码(整数)
	RS_instance	RS		
	NX_SerialRcv _instance	NX_SerialRcv		
	RcvCfg	_sSERIAL_CFG		接收完成设定
	StartTrig	_eSERIAL_START	_SERIAL_START_NONE	无起始符
	StartCode	BYTE[2]	[2(16#0)]	
	EndTrig	_eSERIAL_END	_SERIAL_END_ENDCODE1	有结束符
	EndCode	BYTE[2]	[16#0D,16#00]	16#0D(CR)
	RcvSizeCfg	UINT	0	
	Option	_sSERIAL_RCV _OPTION		选项
	TimeOut	TIME	TIME#0s	
	LastDatRcv	BOOL	FALSE	

外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active _Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1”时，使用 E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量



● 内联ST的内容

```
DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
DevicePort.NxUnit:=N1_Node_location_information;
DevicePort.PortNo:=2;
```

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	DevicePort	_sDEVICE_PORT		指定端口
	RecvDat	ARRAY [0..255] of BYTE	[256(16#0)]	接收数据
	RecvSize	UINT	0	接收数据大小
	RecvStringDat	STRING[257]	''	
	Code	ULINT	0	条形码(整数)
	NX_SerialRcv_instance	NX_SerialRcv		
	RcvCfg	_sSERIAL_CFG		接收完成设定
	StartTrig	_eSERIAL_START	_SERIAL_START_NONE	无起始符
	StartCode	BYTE[2]	[2(16#0)]	
	EndTrig	_eSERIAL_END	_SERIAL_END_ENDCODE1	有结束符
	EndCode	BYTE[2]	[16#0D,16#00]	16#0D(CR)
	RcvSizeCfg	UINT	0	
	Option	_sSERIAL_RCV_OPTION		选项
	TimeOut	TIME	TIME#0s	
	LastDatRcv	BOOL	FALSE	

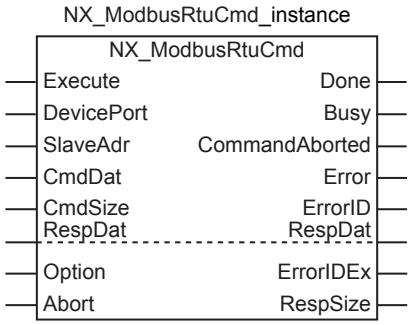
外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1” 时，使用E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE)
   AND(E001_NX_Unit_I_O_Data_Active_Status_63[1]) AND (SerialRcv_instance.Busy=FALSE) ) THEN
      OperatingStart:=TRUE;
      Operating:=TRUE;
      DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
      DevicePort.NxUnit:=N1_Node_location_information;
      DevicePort.Port.PortNo:=2;
END_IF;
LastTrigger:=Trigger;

// 设定通信参数和SerialRcv指令初始化
IF (OperatingStart=TRUE) THEN
      NX_SerialRcv_instance(
                  Execute:=FALSE,       // 实例初始化
                  DevicePort:=DevicePort, // 指定端口
                  Size:=UINT#256,,      // 接收数据大小
                  RcvDat:=RecvDat,      // 接收数据
                  RcvSize=>RecvSize);   // 实际接收的数据大小
      OperatingStart:=FALSE;
END_IF;
// 执行NX_SerialRcv指令
IF (Operating=TRUE) THEN
```

```
    NX_SerialRcv_instance(
                Execute:=TRUE,
                DevicePort:=DevicePort,
                Size:=UINT#256,
                RcvDat:=RecvDat,
                RcvSize=>RecvSize);
    IF (NX_SerialRcv_instance.Done=TRUE) THEN
      // 正常结束时处理
      RecvStringDat:=AryToString(In:=RecvDat[0],Size:=RecvSize); // 将文字代码转换为字符串
      Code:=STRING_TO_ULINT(RecvDat); // 将字符串转换为整数

      Operating:=FALSE;
    END_IF;
    IF (NX_SerialRcv_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating:=FALSE;
    END_IF;
END_IF;
```

NX_ModbusRtuCmd

使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送通用指令。

指令	名称	FB/FUN	图形表现	ST表现
NX_ModbusRtuCmd	Modbus RTU通用指令发送	FB		NX_ModbusRtuCmd_instance(Execute, DevicePort, SlaveAdr, CmdDat, CmdSize, RespDat, Option, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx, RespSize);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
SlaveAdr	从站地址		Modbus-RTU从站的地址*1	0 ~ 247	–	1
CmdDat[]数组	命令数据		命令数据	遵从数据类型	–	*2
CmdSize	指令数据大小		指令数据大小	1 ~ 253	字节	*2*3
Option	选项		选项	–	–	–
Abort	中断		指令执行的中断	遵从数据类型	–	FALSE
RespDat[]数组	读取数据	输入输出	保存读取数据的变量	遵从数据类型	–	–
Command Aborted	中断完成	输出	中断完成	遵从数据类型	–	–
RespSize	接收大小		接收数据大小	1 ~ 253	字节	*4

*1 指定了 “0”时，可对Modbus-RTU从站广播发送指令。
*2 省略输入参数时，初始值不适用。编连时会发生异常。
*3 指定功能代码和指令数据的总字节数。功能代码为1字节。
*4 保存功能代码和读取数据的总字节数。功能代码为1字节。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
SlaveAdr							○													
CmdDat[]数组		○																		
CmdSize							○													
Option	结构体_sSERIAL_MODBUSRTU_OPTION 详情参阅功能说明																			
Abort	○																			
RespDat[]数组		○																		
Command Aborted	○																			
RespSize							○													

功能

使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送通用指令。
接收到对已发送指令的正常响应时，将正常结束。
广播发送时，发送完成后，不等待各从站的响应，本指令即会正常结束。

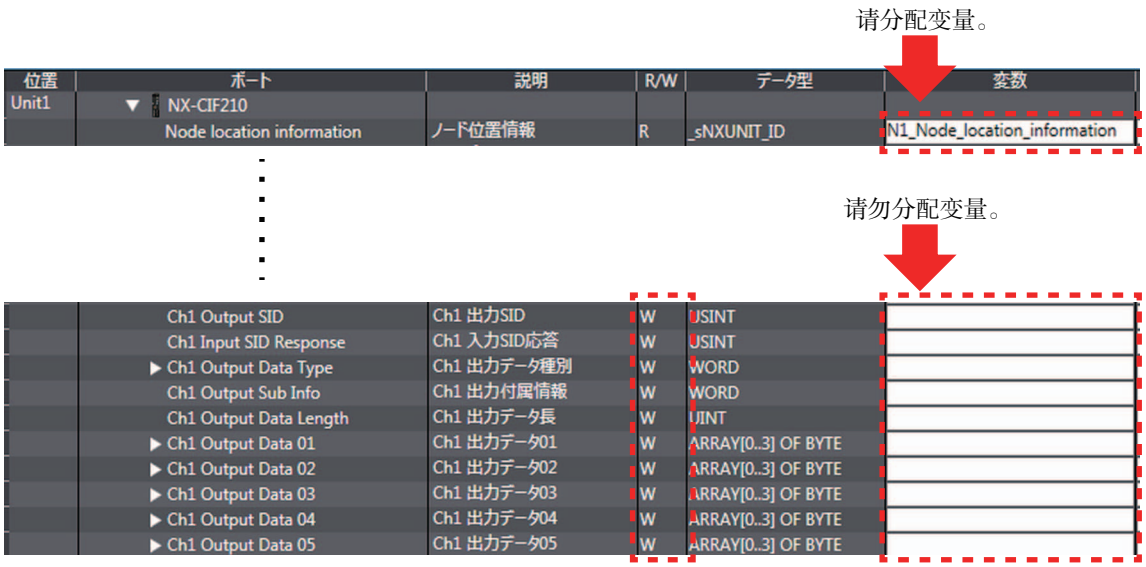
输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType”指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。
用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit”指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。
指定扩展板时，使用 “OptBoard”指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为“W”的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 🕮 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

Modbus-RTU从站的地址使用输入变量 “SlaveAdr”进行指定。
输入变量 “SlaveAdr”中设定 “0”时，对Modbus-RTU从站广播发送指令。

指令数据由输入变量 “CmdDat”指定，指令数据的大小由输入变量 “CmdSize”指定。
CRC为指令附加。
保存读取数据的变量通过输入输出变量 “RespDat”指定。
输出变量 “RespSize”表示接收数据的大小。

指定选项时，使用输入变量 “Option”进行指定。
输入变量 “Option”的数据类型为结构体_sSERIAL_MODBUSRTU_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	选项	选项	_sSERIAL_MODBUSRTU_OPTION	–	–	–
SendDelay	发送延迟时间	以0.01s为单位指定发送延迟时间	UINT	遵从数据类型	0.01s	0
TimeOut	超时时间	超时时间 “0”时为2.0s	UINT	遵从数据类型	0.1s	20
NoResponse	无响应	· 发送指令无响应时，指定TRUE · 指定了TRUE时，发送指令后，不等到超时时间本指令即会正常结束	BOOL	遵从数据类型	–	FALSE
Retry	重新发送次数	指定重新发送次数	USINT	0 ~ 15	–	0

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

指令执行的中断

在指令执行过程中将 “Abort”设为TRUE时，将中断指令的执行。
中断指令执行时，“CommandAborted”将变为TRUE。
指令执行未及时中断时，“Done”将变为TRUE，指令正常结束。
将 “Abort”、“Execute”均设为TRUE时，“CommandAborted”将变为TRUE。

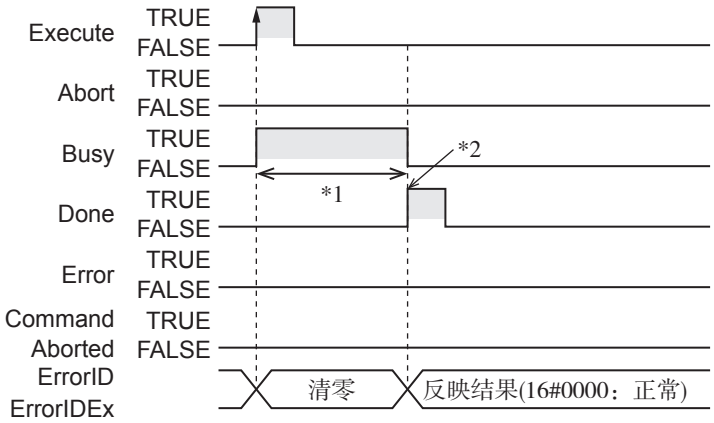
中断动作只结束 “Busy”处理，不会清除收发缓存。需清除缓存时，请使用NX_SerialBufClear指令。

时序图

时序图如下所示。

● 正常结束时(“SendDelay”为 “0”(0s)时)

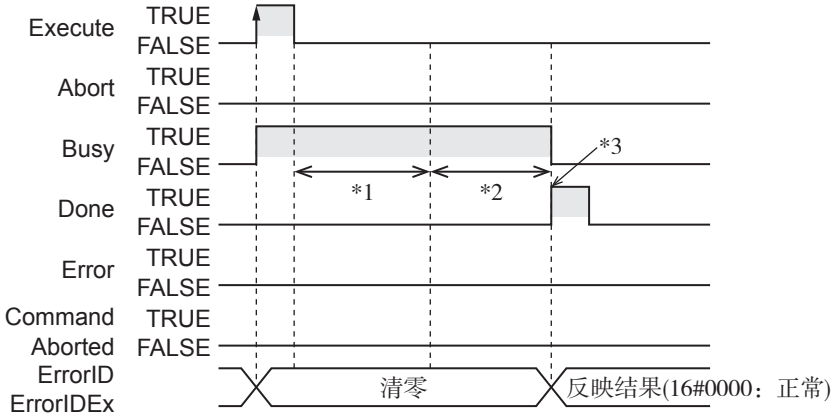
“SendDelay”为 “0”(0s)时，动作如下。



*1 与Modbus-RTU从站之间的处理
*2 接收对指令的响应

● 正常结束时(“SendDelay”为 “100”(1s)时)

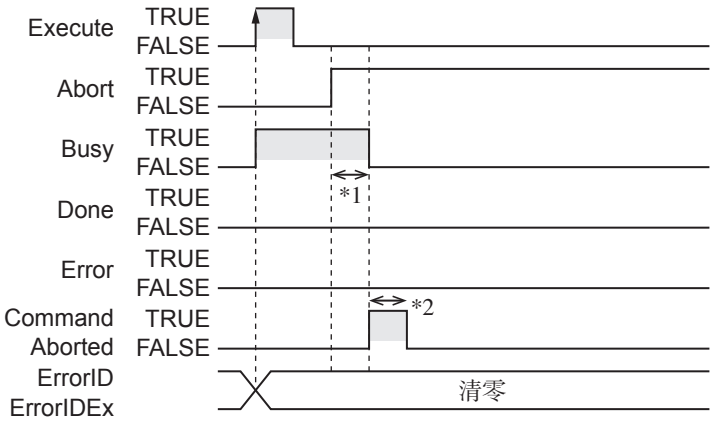
“SendDelay”为 “100”(1s)时，动作如下。



*1 发送延迟时间1s
*2 对Modbus-RTU从站发送指令、从Modbus-RTU从站接收响应
*3 接收对指令的响应

● 中断执行时(“Busy”为TRUE时)

“Busy”TRUE的状态下，将 “Abort”设为TRUE时，动作如下。

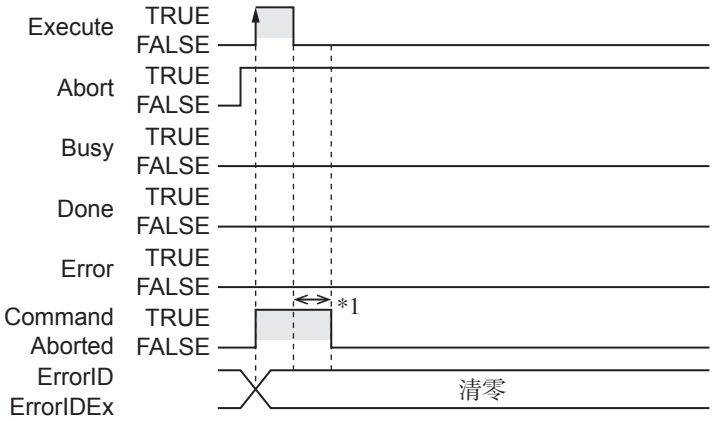


*1 中断处理

*2 1个任务周期后FALSE

● 中断执行时(“Execute”为TRUE时)

将 “Abort”和 “Execute”均设为TRUE时，动作如下。



*1 1个任务周期后FALSE

相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

参考

Modbus-RTU模式的帧格式如下所示。

Slave Address	Function Code	Data	CRC
1字节	1字节	0～字节	2字节*

*CRC代码按从Low字节到High字节的顺序。

关于MODBUS通信协议的规格，请参阅MODBUS Application Protocol Specification。
MODBUS Application Protocol Specification可从MODBUS Organization, Inc.获取。
http://www.modbus.org/

使用注意事项

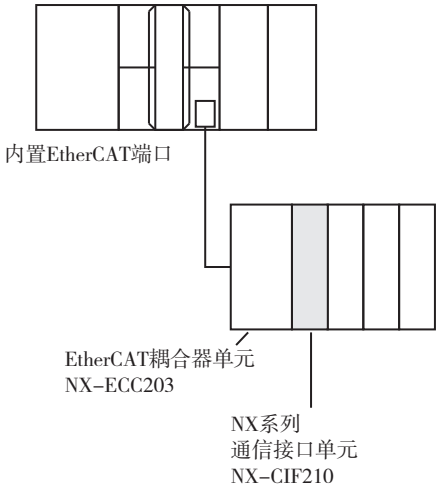
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为 TRUE 确认处理是否正常结束。指令执行中 “Abort”变为 TRUE 时，“CommandAborted”或 “Done”将变为TRUE。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 下述情况下，对象设备端口的缓存中可能会残留数据。需清除缓存时，在执行指令前请执行NX_SerialBufClear指令。
  这里的指令是指NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  - 运行开始后或变更成运行模式时
  - 在执行上一次指令时，设定了重试(Option.Retry≠0)时
  - 上一次的指令执行中断(输出变量 CommandAborted=TRUE)时
  - 上一次的指令执行发生错误(Error=TRUE)时
- 以下情况时会发生异常。“Error”变为TRUE。
  - “CmdSize”、“Option.Retry”、“DevicePort.DevicePortType”、“DevicePort.PortNo” “SlaveAdr”中指定了超出范围的值时。
  - “CmdDat”指定的变量小于 “CmdSize”指定的大小时。
  - 接收数据的大小大于 “RespDat”指定的变量大小时。
  - “DevicePort”指定了不存在的单元或端口时。
  - “DevicePort”的数据类型错误时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  - 接收数据发生了奇偶校验错误时。
  - 接收数据发生了结构错误时。
  - 接收数据发生了超程错误时。
  - 接收数据为CRC不一致时。
  - 超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。

- 从Modbus-RTU从站接收了Exception Response时。Exception Code通过输出变量 “ErrorIDEx”进行确认。
- 来自Modbus-RTU从站的响应数据的功能代码及接收大小错误时。
- 指定扩展板的串行通信模式为 “Modbus-RTU主站”以外时。

• 扩展错误代码 “ErrorIDEx”在本指令检测到 Modbus-RTU 从站异常时显示。“ErrorIDEx”在错误代码 “ErrorID”的值为WORD#16#0C10时输出，显示“ErrorIDEx=000000XX”。关于“XX”，请确认MODBUS通信协议的Exception Code规格。
关于MODBUS通信协议的Exception Code规格，请参阅MODBUS Application Protocol Specification。
MODBUS Application Protocol Specification可从MODBUS Organization, Inc.获取。
http://www.modbus.org/

示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。
将NX-CIF210的单元动作设定的[Ch2 结尾检测字符数]设为 “35”。结尾检测字符数的单位为0.1字符数，因此按3.5字符进行动作。



从NX-CIF210的串行端口2发送Modbus-RTU指令。
“ModbusCmdRequestTrigger”变为ON时，从地址1从站的32号(BYTE#16#0020)读取1个保持寄存器。
使用通用指令收发，执行变量的读取。

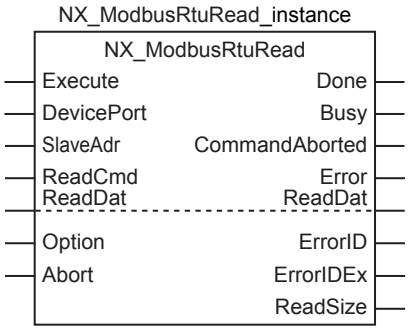
内部变量	名称	数据类型	初始值	注释
	NX_ModbusRtuCmd _instance	NX_ModbusRtuCmd		
	ModbusCmdDat	ARRAY[0..19] OF BYTE	[6(16#0)]	MODBUS指令数据
	ModbusDatSize	UINT	UINT#0	MODBUS指令数据总大小(字节)
	Done	BOOL	BOOL#FALSE	
	Busy	BOOL	BOOL#FALSE	
	ModbusStage	INT	INT#0	MODBUS的通信状态
	RspSize	UINT	UINT#0	实际接收的大小(字节)
	RespDat	ARRAY[0..275] OF BYTE	[6(16#0)]	接收数据保存区域
	Error	BOOL		
	ErrorID	WORD		
	ModbusCmdRequest Trigger	BOOL	FALSE	执行条件
	SlaveAdr	UINT	UINT#0	从站地址
	DevicePort	_sDEVICE_PORT		指定端口

```
IF ( (ModbusCmdRequestTrigger=TRUE)
    AND (ModbusStage=INT#0) ) THEN
    SlaveAdr := 1;              //从站地址
    ModbusCmdDat[1]:=BYTE#16#03; //功能代码(变量读取)
    ModbusCmdDat[2]:=BYTE#16#00; //读取开始地址(H)
    ModbusCmdDat[3]:=BYTE#16#20; //读取开始地址(L)
    ModbusCmdDat[4]:=BYTE#16#00; //数据数(H)
    ModbusCmdDat[5]:=BYTE#16#01; //数据数(L)
    ModbusDatSize:=5;
    ModbusStage:=1;
    DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
    DevicePort.NxUnit:=N1_Node_location_information;
    DevicePort.PortNo:=2;
END_IF

//通用指令发送/接收
NX_ModbusRtuCmd_instance( Execute:=ModbusCmdRequestTrigger,
                DevicePort:=DevicePort,
                SlaveAdr:=SlaveAdr,
                CmdDat:=ModbusCmdDat[1],
                CmdSize:=ModbusDatSize,
                RespDat:=RspDat[0],
                Done=>Done,
                Busy=>Busy,
                Error=>Error,
                ErrorID=>ErrorID,
                RespSize=>RspSize);

IF ModbusStage=1 THEN
  ModbusCmdRequestTrigger:=FALSE;
  // 发生错误
  IF (Error<>FALSE) THEN
     ModbusStage:=0;
  ELSE
     ModbusStage:=2;
  END_IF;
ELSIF (ModbusStage=2) THEN
  // 接收完成
```

```
    IF (Error=FALSE) AND (Done=TRUE) THEN
        // 接收指令解析处理

        ModbusStage:=0;

    // 接收处理中
    ELSIF (Busy=TRUE) THEN

    // 接收失败
    ELSE

        ModbusStage:=0;
    END_IF;
END_IF;
```

NX_ModbusRtuRead

使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送读取指令。

指令	名称	FB/FUN	图形表现	ST表现
NX_ModbusRtuRead	Modbus RTU Read指令发送	FB		NX_ModbusRtuRead_instance(Execute, DevicePort, SlaveAdr, ReadCmd, ReadDat, Option, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx, ReadSize);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
SlaveAdr	从站地址		Modbus-RTU从站的地址 *1	1 ~ 247	–	1
ReadCmd	读取指令		读取指令	–	–	*2
Option	选项		选项	–	–	–
Abort	中断		指令执行的中断	遵从数据类型	–	FALSE
ReadDat[]数组	读取数据	输入输出	保存读取数据的变量	遵从数据类型	–	–
CommandAborted	中断完成	输出	中断完成	遵从数据类型	–	–
ReadSize	接收大小		接收数据大小	1 ~ 2000 *3	– *4	–

*1 设定 “0” 时，会发生异常。
*2 省略输入参数时，初始值不适用。编连时会发生异常。
*3 接收数据为WORD数据时，上限值为 “125”。
*4 与 “ReadCmd.Fun” 指定的读取数据的单位相同。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
SlaveAdr							○													
ReadCmd	结构体_sSERIAL_MODBUSRTU_READ 详情参阅功能说明																			
Option	结构体_sSERIAL_MODBUSRTU_OPTION 详情参阅功能说明																			
Abort	○																			
ReadDat[]数组	○		○																	
	也可指定整个数组																			
Command Aborted	○																			
ReadSize							○													

功能

使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送读取指令，从Modbus-RTU从站读取请求的数据。
接收到对已发送指令的正常响应时，将正常结束。

输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用“DeviceType”指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit”指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。

指定扩展板时，使用 “OptBoard”指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为“W”的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。

将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

Modbus-RTU从站的地址使用输入变量 “SlaveAdr”进行指定。
输入变量 “SlaveAdr”中指定了 “0”时会发生异常，无法对Modbus-RTU从站广播发送指令。

读取指令通过输入变量 “ReadCmd”指定。
CRC为指令附加。
输入变量 “ReadCmd”的数据类型为结构体_sSERIAL_MODBUSRTU_READ。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
ReadCmd	读取指令	读取指令	_sSERIAL_MODBUSRTU_READ	–	–	–
Fun	功能代码	功能代码	_eMDB_FUN	_MDB_READ_COILS _MDB_READ_DISCRETE_INPUTS _MDB_READ_HOLDING_REGISTERS _MDB__READ_INPUT_REGISTERS	–	_MDB_READ_COILS
ReadAdr	读取地址	读取开始地址	UINT	遵从数据类型	–	0
ReadSize	读取大小	读取大小	UINT	遵从功能代码	– *1	_MDB_READ_COILS

*1 与 “ReadCmd.Fun”指定的读取数据的单位相同。

“Fun”的数据类型为枚举体_eMDB_FUN。
枚举体_eMDB_FUN的枚举元素的含义如下所述。

枚举元素	含义
_MDB_READ_COILS	线圈读取(位)
_MDB_READ_DISCRETE_INPUTS	输入读取(位)
_MDB_READ_HOLDING_REGISTERS	保持寄存器读取(字)
_MDB_READ_INPUT_REGISTERS	输入寄存器读取(字)

“ReadSize”可指定的有效范围因功能代码而异。
各数值取决于读取对象的数据大小和最大指令长度。
规格如下所示。

功能代码	ReadSize
_MDB_READ_COILS	1 ~ 2000(位)
_MDB_READ_DISCRETE_INPUTS	1 ~ 2000(位)
_MDB_READ_HOLDING_REGISTERS	1 ~ 125(字)
_MDB_READ_INPUT_REGISTERS	1 ~ 125(字)

保存读取数据的变量通过输入输出变量 “ReadDat”指定。
“ReadDat”可使用的数据类型因功能代码而异。
规格如下所示。

功能代码	数据类型
_MDB_READ_COILS	BOOL BOOL[]
_MDB_READ_DISCRETE_INPUTS	BOOL BOOL[]
_MDB_READ_HOLDING_REGISTERS	WORD WORD[]
_MDB_READ_INPUT_REGISTERS	WORD WORD[]

输出变量 “ReadSize”表示读取数据的大小。

指定选项时，使用输入变量 “Option”进行指定。
输入变量 “Option”的数据类型为结构体_sSERIAL_MODBUSRTU_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	选项	选项	_sSERIAL_MODBUSRTU_OPTION	–	–	–
SendDelay	发送延迟时间	发送延迟时间	UINT	遵从数据类型	0.01s	0
TimeOut	超时时间	超时时间 “0”时为2.0s	UINT	遵从数据类型	0.1s	20
NoResponse	无响应	本指令不使用	BOOL	遵从数据类型	–	FALSE
Retry	重新发送次数	重新发送次数	USINT	0 ~ 15	–	0

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

指令执行的中断

在指令执行过程中将 “Abort”设为TRUE时，将中断指令的执行。
中断指令执行时，“CommandAborted”将变为TRUE。
指令执行未及时中断时，“Done”将变为TRUE，指令正常结束。
将 “Abort”、“Execute”均设为TRUE时，“CommandAborted”将变为TRUE。

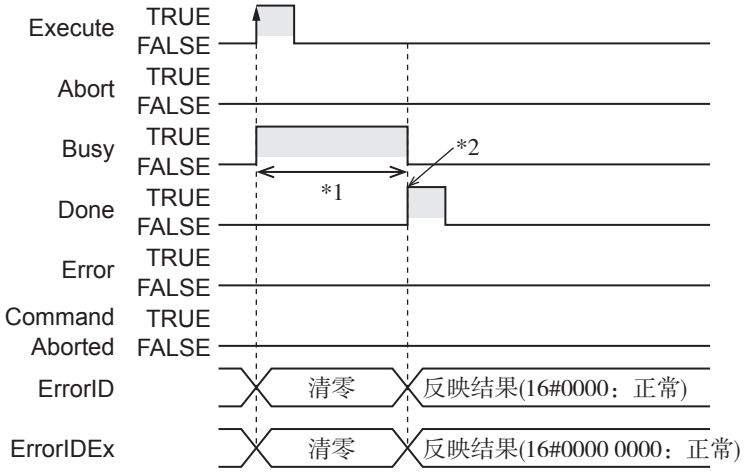
中断动作只结束指令的“Busy”处理，不会清除收发缓存。需清除缓存时，请使用NX_SerialBufClear指令。

时序图

时序图如下所示。

● 正常结束时(“SendDelay”为 “0”(0s)时)

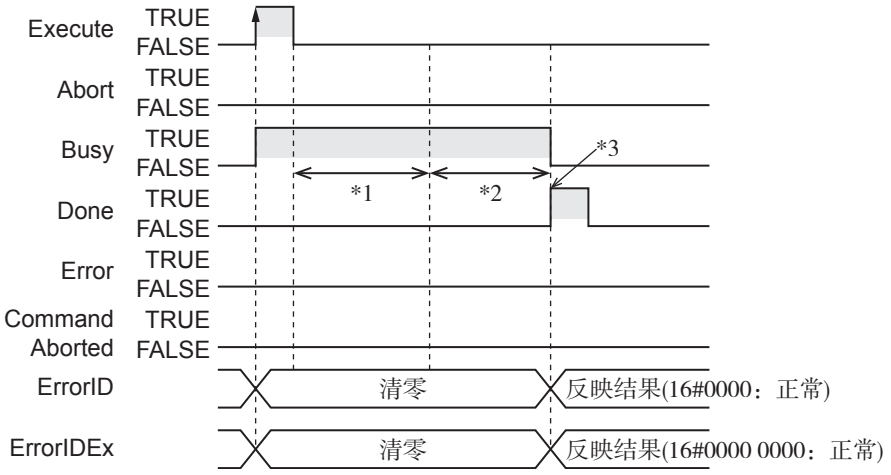
“SendDelay”为 “0”(0s)时，动作如下。



*1 与Modbus-RTU从站之间的处理
*2 接收对指令的响应

● 正常结束时(“SendDelay”为 “100”(1s)时)

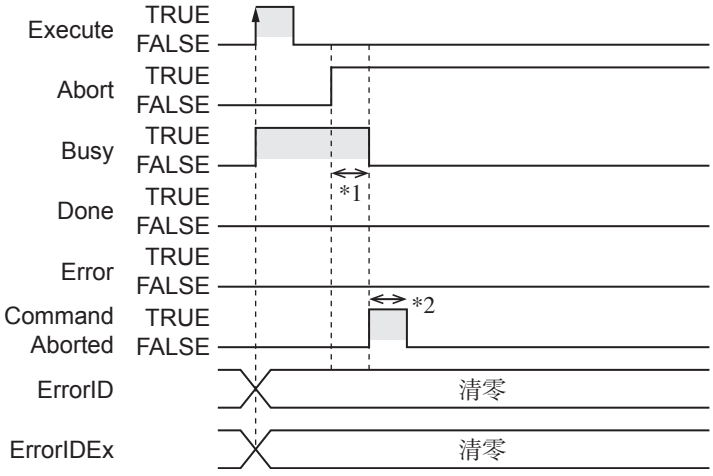
“SendDelay”为 “100”(1s)时，动作如下。



*1 发送延迟时间1s
*2 对Modbus-RTU从站发送读取指令、从Modbus-RTU从站接收响应
*3 接收对指令的响应

● 中断执行时(“Busy”为TRUE时)

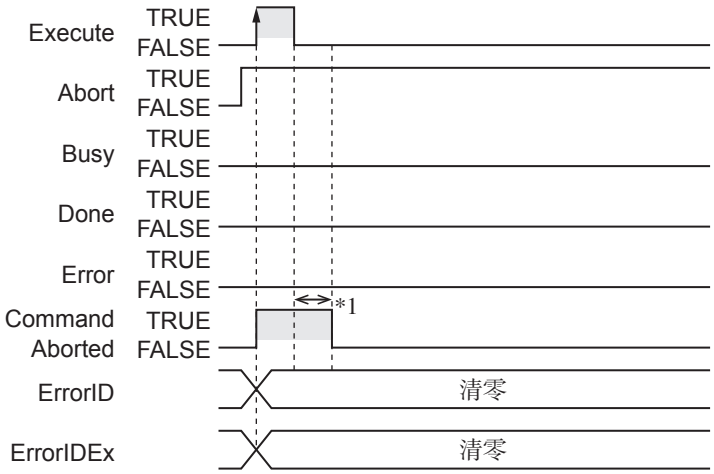
“Busy”TRUE的状态下，将 “Abort”设为TRUE时，动作如下。



*1 中断处理
*2 1个任务周期后FALSE

● 中断执行时(“Execute”为TRUE时)

将 “Abort”和 “Execute”均设为TRUE时，动作如下。



*1 1个任务周期后FALSE

相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

参考

关于MODBUS通信协议的规格，请参阅MODBUS Application Protocol Specification。
MODBUS Application Protocol Specification可从MODBUS Organization, Inc.获取。
http://www.modbus.org/

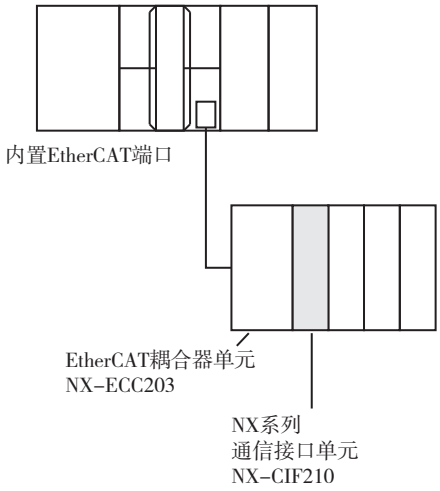
使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为 TRUE 确认处理是否正常结束。指令执行中 “Abort”变为 TRUE 时，“CommandAborted”或 “Done”将变为TRUE。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。

- 下述情况下，对象设备端口的缓存中可能会残留数据。需清除缓存时，在执行指令前请执行NX_SerialBufClear指令。
  这里的指令是指NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  - 运行开始后或变更成运行模式时
  - 在执行上一次指令时，设定了重试(Option.Retry≠0)时
  - 上一次的指令执行中断(输出变量 CommandAborted=TRUE)时
  - 上一次的指令执行发生错误(Error=TRUE)时
- 以下情况时会发生异常。“Error”变为TRUE。
  - “SlaveAdr”、“ReadCmd.ReadSize”、“ReadCmd.Fun”、“Option.Retry”、“DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “ReadDat”指定的变量小于 “ReadCmd.ReadSize”指定的大小时。
  - “DevicePort”指定了不存在的单元或端口时。
  - “DevicePort”、“RespDat”的数据类型错误时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  - 接收数据发生了奇偶校验错误时。
  - 接收数据发生了结构错误时。
  - 接收数据发生了超程错误时。
  - 接收数据为CRC不一致时。
  - 超过超时时间时。(重试设定时为重试次数 × 超时时间)
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。
  - 从Modbus-RTU从站接收了Exception Response时。Exception Code通过输出变量 “ErrorIDEx”进行确认。
  - 来自Modbus-RTU从站的响应数据的功能代码及接收大小错误时。
  - 指定扩展板的串行通信模式为 “Modbus-RTU主站”以外时。
- 扩展错误代码 “ErrorIDEx”在本指令检测到Modbus-RTU从站异常时显示。“ErrorIDEx”在错误代码“ErrorID”的值为WORD#16#0C10时输出，显示 “ErrorIDEx=000000XX”。关于 “XX”，请确认MODBUS通信协议的Exception Code规格。
  关于MODBUS通信协议的Exception Code规格，请参阅MODBUS Application Protocol Specification。
  MODBUS Application Protocol Specification可从MODBUS Organization, Inc.获取。
  http://www.modbus.org/

示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。
将NX-CIF210的单元动作设定的[Ch2 结尾检测字符数]设为 “35”。结尾检测字符数的单位为0.1字符数，因此按3.5字符进行动作。



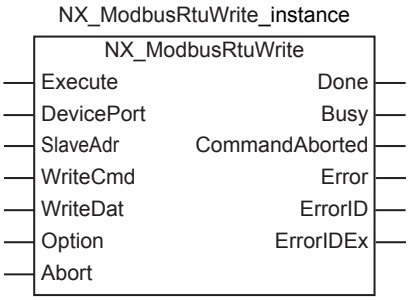
从NX-CIF210的串行端口2发送Modbus-RTU指令。
“ModbusCmdRequestTrigger”变为ON时，从地址1从站的线圈19号读取1个线圈的状态。
使用读取指令发送，执行变量的读取。

内部变量	名称	数据类型	初始值	注释
	NX_ModbusRtuRead _instance	NX_ModbusRtuRead		
	Done	BOOL		
	Busy	BOOL		
	ModbusStage	INT	INT#0	MODBUS的通信状态
	Error	BOOL		
	ErrorID	WORD		
	ModbusCmdRequestTrigger	BOOL	FALSE	执行条件
	SlaveAdr	UINT	UINT#0	从站地址
	DevicePort	_sDEVICE_PORT		指定端口
	ReadCmd	_sSERIAL_MODBUSRTU _READ		
	ReadDat	BOOL	[6(16#0)]	
	ReadSize	UINT	UINT#0	

```
IF ( (ModbusCmdRequestTrigger=TRUE)
     AND (ModbusStage=INT#0) ) THEN
     SlaveAdr:=1;
     ReadCmd.FUN:=_MDB_READ_COILS;
     ReadCmd.ReadAdr:=19;
     ReadCmd.ReadSize:=1;
     ModbusStage=1;
     DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
     DevicePort.NxUnit:=N1_Node_location_information;
     DevicePort.PortNo:=2;
END_IF;
// 指令收发
NX_ModbusRtuRead_instance (Execute:=ModbusCmdRequestTrigger,
                    DevicePort:=DevicePort,
                    SlaveAdr:=SlaveAdr,
                    ReadCmd:=ReadCmd,
                    ReadDat:=ReadDat,
                    Done=>Done,
                    Busy=>Busy,
                    Error=>Error,
                    ReadSize=>ReadSize);

IF (ModbusStage=1) THEN
   // 发送成功
   IF (Error=FALSE) THEN
       ModbusStage:=2;
   // 发生错误
   ELSE
      ModbusStage:=0;
      ModbusCmdRequestTrigger:=FALSE;
   END_IF;
ELSIF (ModbusStage=2) THEN
   // 接收完成
   IF (Error=FALSE) AND (Done=TRUE) THEN
      ModbusStage:=0;
      ModbusCmdRequestTrigger:=FALSE;
   // 接收处理中
   ELSIF (Busy=TRUE) THEN

   // 接收失败
   ELSE

      ModbusStage:=0;
      ModbusCmdRequestTrigger:=FALSE;
   END_IF;
END_IF;
```

NX_ModbusRtuWrite

使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送写入指令。

指令	名称	FB/FUN	图形表现	ST表现
NX_ModbusRtuWrite	Modbus RTU Write指令发送	FB		NX_ModbusRtuWrite_instance(Execute, DevicePort, SlaveAdr, WriteCmd, WriteDat, Option, Abort, Done, Busy, CommandAborted, Error, ErrorID, ErrorIDEx);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
SlaveAdr	从站地址		Modbus-RTU从站的地址*1	0 ~ 247	–	1
WriteCmd	写入指令		写入指令	–	–	*2
WriteDat[]数组	写入数据		写入数据	遵从数据类型	–	*2
Option	选项		选项	–	–	–
Abort	中断		指令执行的中断	遵从数据类型	–	FALSE
Command Aborted	中断完成	输出	中断完成	遵从数据类型	–	–

*1 设定了 “0”时，可对Modbus-RTU从站广播发送指令。
*2 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
SlaveAdr							○													
WriteCmd	结构体_sSERIAL_MODBUSRTU_WRITE 详情参阅功能说明																			
WriteDat[]数组	○		○																	
	也可指定整个数组																			
Option	结构体_sSERIAL_MODBUSRTU_OPTION 详情参阅功能说明																			
Abort	○																			
CommandAborted	○																			

功能

使用Modbus-RTU协议，从NX系列通信接口单元及扩展板的串行端口向Modbus-RTU从站发送写入指令。
接收到对已发送指令的正常响应时，将正常结束。
广播发送时，发送完成后，不等待各从站的响应，本指令即会正常结束。

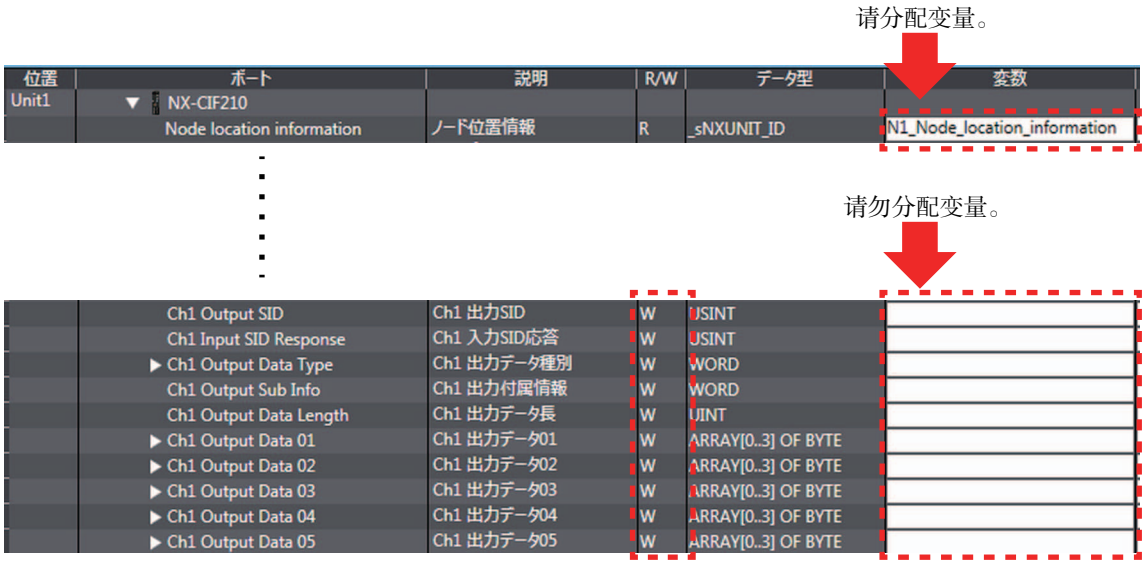
输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	-	-	-
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	-	-
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	-	-	-
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	-	-	-
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	-	-	-
Reserved	保留	保留	保留	-	-	-
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	-	-

使用 “DeviceType”指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。
用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit”指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。
指定扩展板时，使用 “OptBoard”指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。
使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为 “W”的对象，请勿分配设备变量。

例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

Modbus-RTU从站的地址使用输入变量 “SlaveAdr”进行指定。
输入变量 “SlaveAdr”中设定 “0”时，对Modbus-RTU从站广播发送指令。

写入指令通过输入变量 “WriteCmd”指定。
CRC为指令附加。
输入变量 “WriteCmd”的数据类型为结构体_sSERIAL_MODBUSRTU_WRITE。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
WriteCmd	写入指令	写入指令	_sSERIAL_MODBUSRTU_WRITE	–	–	–
Fun	功能代码	功能代码	_eMDB_FUN	_MDB_WRITE_SINGLE_COIL _MDB_WRITE_SINGLE_REGISTER _MDB_WRITE_MULTIPLE_COILS _MDB_WRITE_MULTIPLE_REGISTERS	–	_eMDB_WRITE_SINGLE_COIL
WriteAdr	写入地址	写入开始地址	UINT	遵从数据类型	–	0
WriteSize	写入大小	写入大小	UINT	遵从功能代码	–	_MDB_WRITE_SINGLE_COIL

“Fun”的数据类型为枚举体_eMDB_FUN。
枚举体_eMDB_FUN的枚举元素的含义如下所述。

枚举元素	含义
_MDB_WRITE_SINGLE_COIL	线圈1点写入(位)
_MDB_WRITE_SINGLE_REGISTER	保持寄存器1点写入(字)
_MDB_WRITE_MULTIPLE_COILS	多个线圈写入(位)
_MDB_WRITE_MULTIPLE_REGISTERS	多个保持寄存器写入(字)

“WriteSize”可指定的有效范围因功能代码而异。
各数值取决于写入对象的数据大小和最大指令长度。
规格如下所示。

功能代码	WriteSize
_MDB_WRITE_SINGLE_COIL	1(位)
_MDB_WRITE_SINGLE_REGISTER	1(字)
_MDB_WRITE_MULTIPLE_COILS	1 ~ 1968(位)
_MDB_WRITE_MULTIPLE_REGISTERS	1 ~ 123(字)

写入数据通过输入变量 “WriteDat”指定。
“WriteDat”可使用的数据类型因功能代码而异。
规格如下所示。

功能代码	数据类型
_MDB_WRITE_SINGLE_COIL	BOOL BOOL[]
_MDB_WRITE_SINGLE_REGISTER	WORD WORD[]
_MDB_WRITE_MULTIPLE_COILS	BOOL BOOL[]
_MDB_WRITE_MULTIPLE_REGISTERS	WORD WORD[]

指定选项时，使用输入变量 “Option”进行指定。规格如下所示。
输入变量 “Option”的数据类型为结构体_sSERIAL_MODBUS_OPTION。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
Option	选项	选项	_sSERIAL_MODBUSRTU_OPTION	–	–	–
SendDelay	发送延迟时间	发送延迟时间	UINT	遵从数据类型	0.01s	0
TimeOut	超时时间	超时时间 “0”时为2.0s	UINT	遵从数据类型	0.1s	20
NoResponse	无响应	本指令不使用	BOOL	遵从数据类型	–	FALSE
Retry	重新发送次数	重新发送次数	USINT	0 ~ 15	–	0

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

指令执行的中断

在指令执行过程中将 “Abort”设为TRUE时，将中断指令的执行。
中断指令执行时，“CommandAborted”将变为TRUE。
指令执行未及时中断时，“Done”将变为TRUE，指令正常结束。
将 “Abort”、“Execute”均设为TRUE时，“CommandAborted”将变为TRUE。

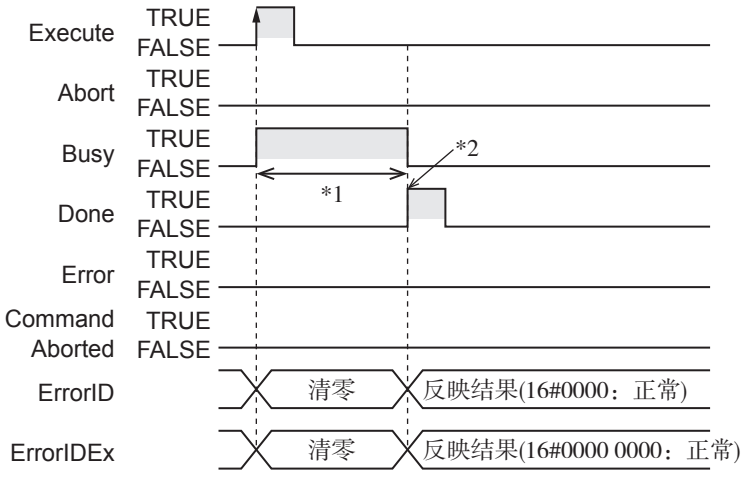
中断动作只结束指令的“Busy”处理，不会清除收发缓存。需清除缓存时，请使用NX_SerialBufClear指令。

时序图

时序图如下所示。

● 正常结束时(“SendDelay”为 “0”(0s)时)

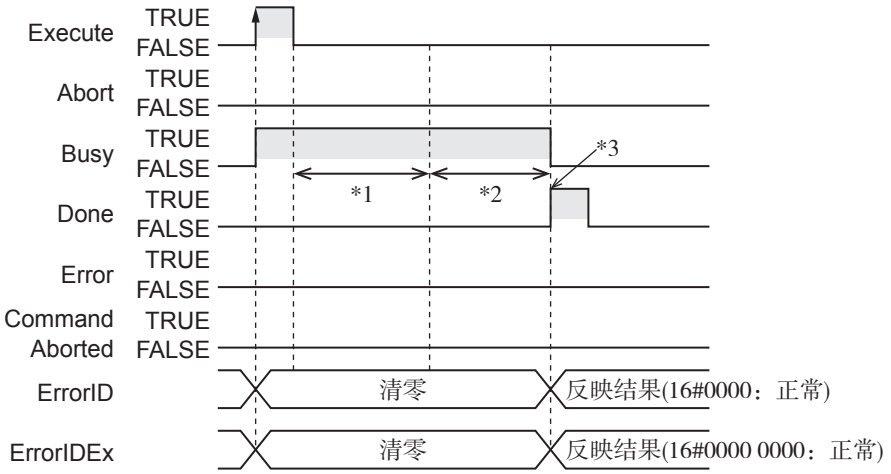
“SendDelay”为 “0”(0s)时，动作如下。



*1 与Modbus-RTU从站之间的处理
*2 接收对指令的响应

● 正常结束时(“SendDelay”为 “100”(1s)时)

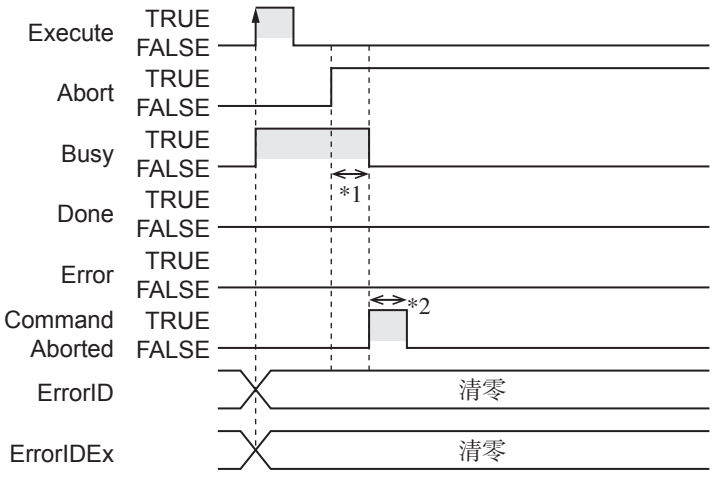
“SendDelay”为 “100”(1s)时，动作如下。



*1 发送延迟时间1s
*2 对Modbus-RTU从站发送写入指令、从Modbus-RTU从站接收响应
*3 接收对指令的响应

● 中断执行时(“Busy”为TRUE时)

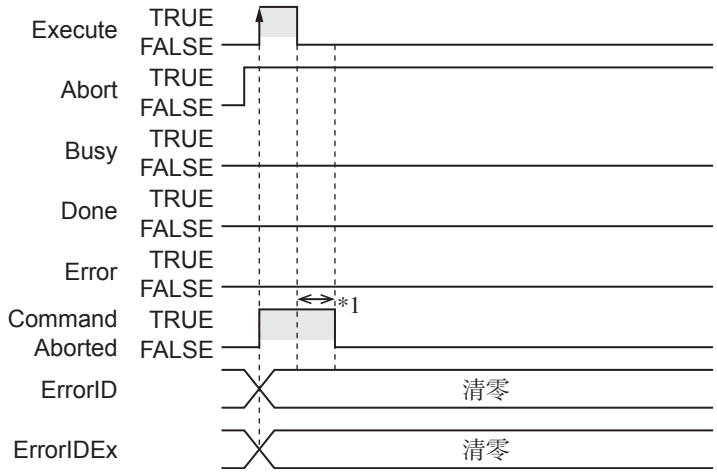
“Busy”TRUE的状态下，将 “Abort”设为TRUE时，动作如下。



*1 中断处理
*2 1个任务周期后FALSE

● 中断执行时(“Execute”为TRUE时)

将 “Abort”和 “Execute”均设为TRUE时，动作如下。



*1 1个任务周期后FALSE

相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

参考

关于MODBUS通信协议的规格，请参阅MODBUS Application Protocol Specification。
MODBUS Application Protocol Specification可从MODBUS Organization, Inc.获取。
http://www.modbus.org/

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为 TRUE 确认处理是否正常结束。指令执行中 “Abort”变为 TRUE 时，“CommandAborted”或 “Done”将变为TRUE。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 下述情况下，对象设备端口的缓存中可能会残留数据。需清除缓存时，在执行指令前请执行NX_SerialBufClear指令。
  这里的指令是指NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  · 运行开始后或变更成运行模式时
  · 在执行上一次指令时，设定了重试(Option.Retry≠0)时
  · 上一次的指令执行中断(输出变量 CommandAborted=TRUE)时

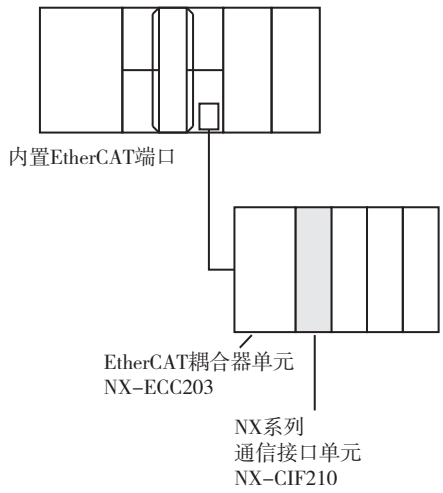
- 上一次的指令执行发生错误(Error=TRUE)时

- 以下情况时会发生异常。“Error”变为TRUE。
  - “SlaveAdr”、“WriteCmd.Fun”、“WriteCmd.WriteSize”、“Option.Retry”、“DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “WriteDat”指定的变量小于 “WriteCmd.WriteSize”指定的大小时。
  - “DevicePort”指定了不存在的单元或端口时。
  - “DevicePort”、“WriteDat”的数据类型错误时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令。
  - 接收数据发生了奇偶校验错误时。
  - 接收数据发生了结构错误时。
  - 接收数据发生了超程错误时。
  - 接收数据为CRC不一致时。
  - 超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。
  - 从Modbus-RTU从站接收了Exception Response时。Exception Code通过输出变量 “ErrorIDEx”进行确认。
  - 来自Modbus-RTU从站的响应数据的功能代码及接收大小错误时。
  - 指定扩展板的串行通信模式为 “Modbus-RTU主站”以外时。

- 扩展错误代码 “ErrorIDEx”在本指令检测到Modbus-RTU从站异常时显示。“ErrorIDEx”在错误代码“ErrorID”的值为WORD#16#0C10时输出，显示 “ErrorIDEx=000000XX”。关于 “XX”，请确认MODBUS通信协议的Exception Code规格。
  关于MODBUS通信协议的Exception Code规格，请参阅MODBUS Application Protocol Specification。
  MODBUS Application Protocol Specification可从MODBUS Organization, Inc.获取。
  http://www.modbus.org/

示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。
将NX-CIF210的单元动作设定的[Ch2 结尾检测字符数]设为 “35”。结尾检测字符数的单位为0.1字符数，因此按3.5字符进行动作。



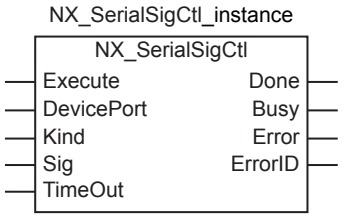
从NX-CIF210的串行端口2发送Modbus-RTU指令。
“ModbusCmdRequestTrigger”变为ON时，从地址1从站的149号将1个线圈设为ON。
使用写入指令收发，执行变量的写入。

内部变量	名称	数据类型	初始值	注释
	NX_ModbusRtuWrite _instance	NX_ModbusRtuWrite		
	Done	BOOL	BOOL#FALSE	
	Busy	BOOL	BOOL#FALSE	
	ModbusStage	INT	INT#0	MODBUS的通信状态
	Error	BOOL		
	ErrorID	WORD		
	ModbusCmdRequestTrigger	BOOL	FALSE	执行条件
	SlaveAdr	UINT	UINT#0	从站地址
	DevicePort	_sDEVICE_PORT		指定端口
	WriteCmd	_sSERIAL_MODBUSRTU _WRITE		
	WriteDat	BOOL	[6(16#0)]	

```
IF ( (ModbusCmdRequestTrigger=TRUE)
     AND (ModbusStage=INT#0) ) THEN

     WriteCmd.FUN:=WRITE_SINGLE_COIL;
     WriteCmd.WriteAdr:=149;
     WriteCmd.WriteSize:=1;

     ModbusCmdRequestTrigger:= FALSE;
     ModbusStage:=1;
     DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
     DevicePort.NxUnit:=N1_Node_location_information;
     DevicePort.PortNo:=2;
END_IF;

// 指令收发
NX_ModbusRtuWrite_instance (Execute:=ModbusCmdRequestTrigger,
                    DevicePort:=DevicePort,
                    SlaveAdr:=SlaveAdr,
                    WriteCmd:=WriteCmd,
                    WriteDat:=WriteDat,
                    Error=>Error,
                    ErrorID=>ErrorID);

IF (ModbusStage=1) THEN
   // 发送成功
   IF (Error=FALSE) THEN
      ModbusStage:=2;
   // 发生错误
   ELSE
      ModbusStage:=0;
   END_IF;
ELSIF (ModbusStage=2) THEN
   // 接收完成
   IF (Error=FALSE) AND (Done=TRUE) THEN
      ModbusStage:=0;
   // 接收处理中
   ELSIF (Busy=TRUE) THEN

   // 接收失败
   ELSE

      ModbusStage:=0;
   END_IF;
END_IF;
```

NX_SerialSigCtl

使NX系列通信接口单元及扩展板串行端口的ER信号或RS信号ON或OFF。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialSigCtl	串行控制信号的ON/OFF切换	FB		NX_SerialSigCtl_instance(Execute, DevicePort, Kind, Sig, TimeOut, Done, Busy, Error, ErrorID);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
Kind	信号指令		信号指令	_RS_SIG _ER_SIG*1	–	*2
Sig	ON/OFF指令		ON/OFF指令	遵从数据类型	–	*2
TimeOut	超时时间		超时时间 “0”时为2.0s	遵从数据类型	0.1s	0

*1　无法使用_CS_SIG和_DR_SIG。如指定，执行指令时会发生异常。
*2　省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT　详情参阅功能说明																			
Kind	枚举体_eSERIAL_SIG　枚举元素参阅功能说明																			
Sig	○																			
TimeOut							○													

功能

使NX系列通信接口单元及扩展板串行端口的ER信号或RS信号ON或OFF。

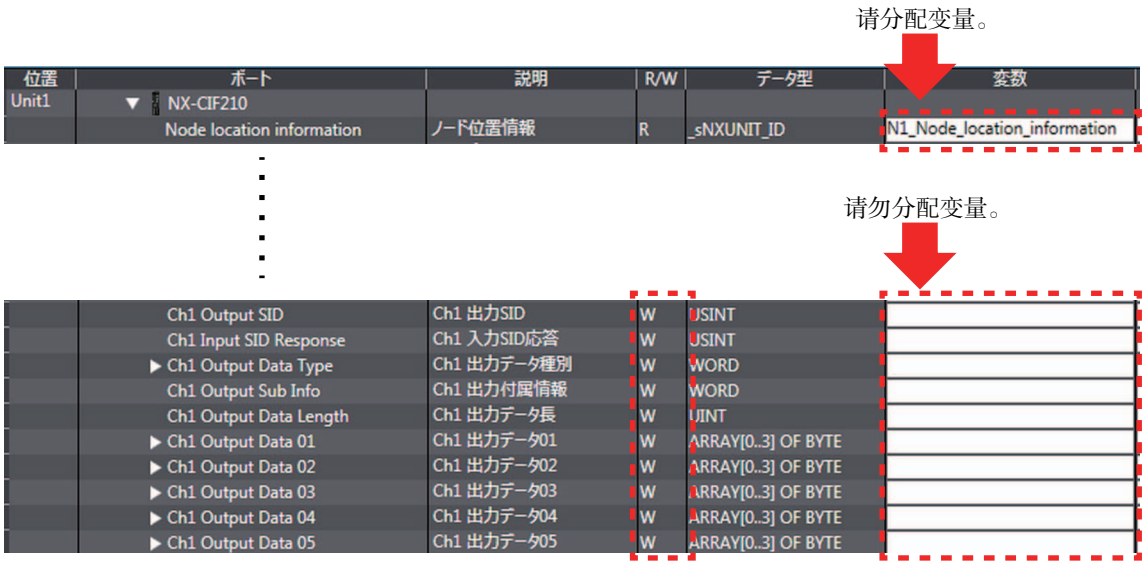
输入变量 “DevicePort” 的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType” 指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit” 指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。
指定扩展板时，使用 “OptBoard” 指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为“W”的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

ER信号或RS信号通过输入变量 “Kind”选择。
输入变量 “Sig”为TRUE时，ER信号或RS信号会变为ON。
输入变量 “Sig”为FALSE时，ER信号或RS信号会变为OFF。

输入变量 “Kind”的数据类型为枚举体_eSERIAL_SIG。
枚举体_eSERIAL_SIG的枚举元素的含义如下所述。

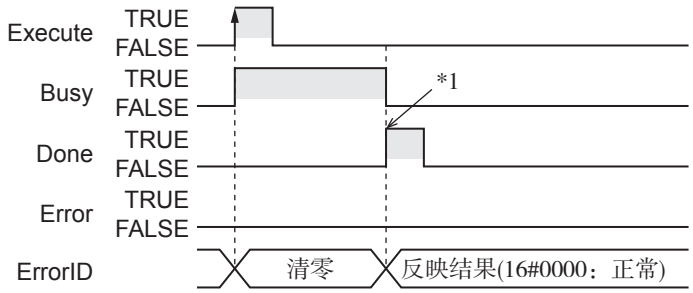
枚举元素	含义
_RS_SIG	RS信号
_ER_SIG	ER信号
_CS_SIG	CS信号
_DR_SIG	DR信号

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

时序图

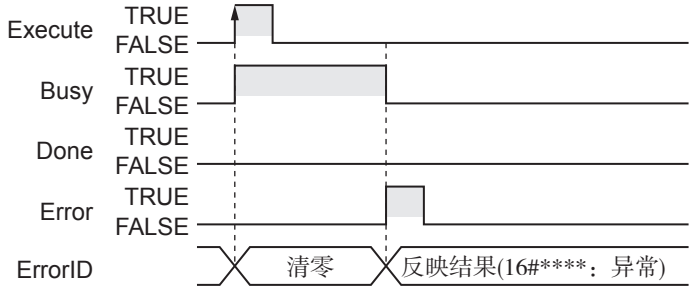
时序图如下所示。

● 正常结束时



*1 信号的ON/OFF控制完成

● 异常结束时



相关的系统定义变量

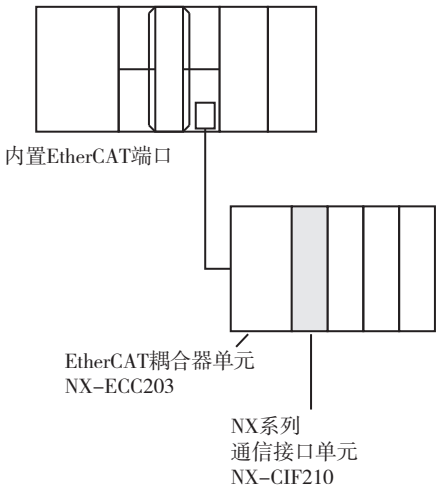
变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- 本指令不执行通信协议及接线状态的检查。请在确认接线状态及通信协议后使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Kind”、“DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - “DevicePort”指定了RS-422A/485的串行端口时。
  - NX 系列通信接口单元的流程控制方法的设定为 “RS/CS 流程控制”时，使用本指令发送了 “RS 信号ON”或 “RS信号OFF”。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为 NX_SerialSigRead 指令、NX_SerialStatusRead 指令、NX_SerialSigCtl 指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令。
  - 串行通信超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。
  - 指定扩展板的串行通信模式为 “无协议”或 “Modbus-RTU主站”以外时。

示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。



对于与NX-CIF210的串行端口2连接的无协议的通信对象，将SetER信号设为ON时则将ER信号设为ON，将ResetER信号设为ON时则将ER信号设为OFF。

全局变量的定义

全局变量

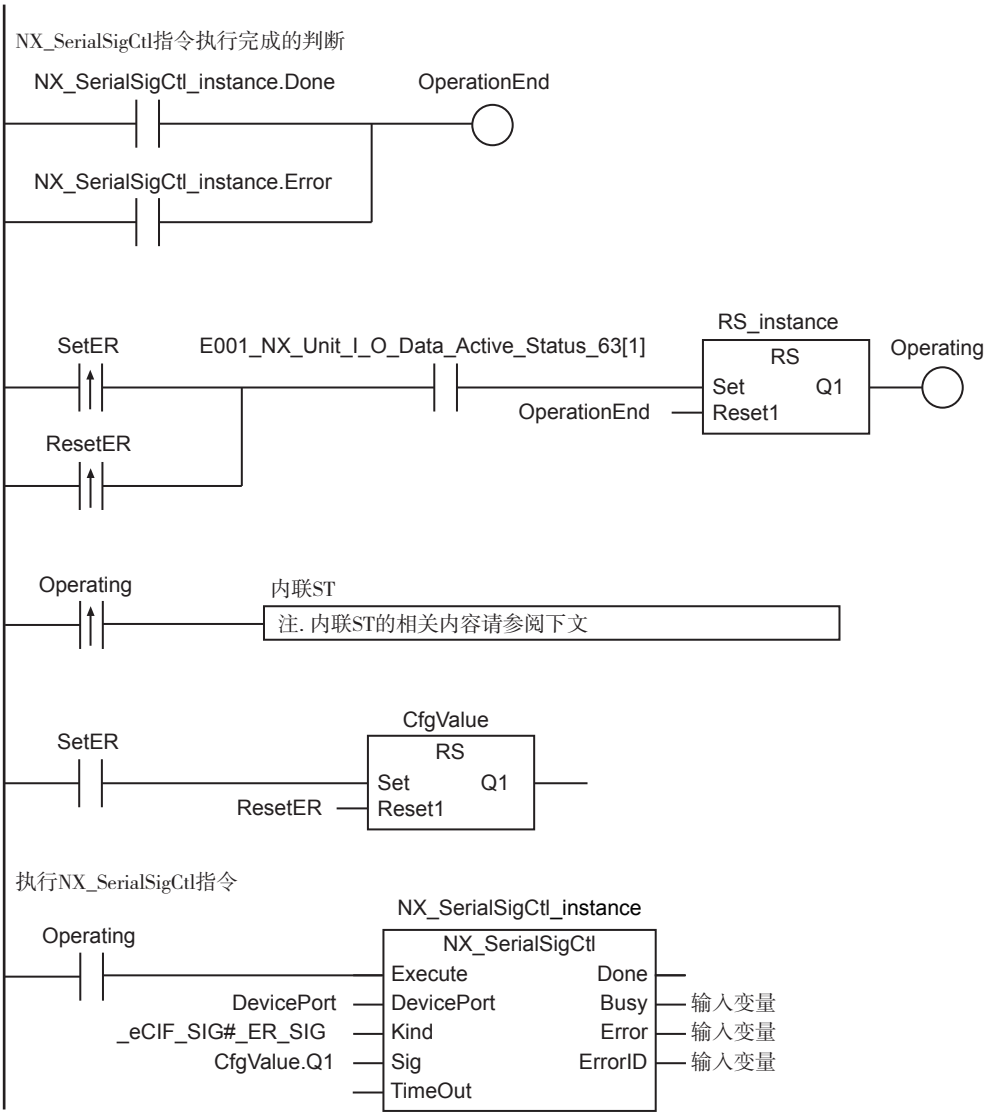
名称	数据类型	分配对象	注释
E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	ECAT://node#1/NX Unit I/O Data Active Status 125	可否使用63台NX单元的I/O数据
N1_Node_location_information	_sNXUNIT_ID	-	NX-CIF210指定的设备变量*1

*1 使用 Sysmac Studio 右击 NX 系列从站终端的单元，选择 “显示节点位置端口” ，设定设备变量。详情请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

LD

内部变量	名称	数据类型	初始值	注释
	OperationEnd	BOOL	FALSE	处理结束
	SetER	BOOL	FALSE	ER 信号ON执行条件
	ResetER	BOOL	FALSE	ER 信号OFF执行条件
	Operating	BOOL	FALSE	处理中
	DevicePort	_sDEVICE_PORT		指定端口
	RS_instance	RS	-	Operating的保持
	CfgValue	RS	-	根据SetER/ResetER确定数值
	NX_SerialSigCtl_instance	NX_SerialSigCtl	-	

外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1”时，使用E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量



● 内联ST的内容

```
DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
DevicePort.NxUnit:=N1_Node_location_information;
DevicePort.PortNo:=2;
```

ST

内部变量	名称	数据类型	初始值	注释
	OperatingStart	BOOL	FALSE	处理开始
	SetER	BOOL	FALSE	ER 信号ON执行条件
	ResetER	BOOL	FALSE	ER 信号OFF执行条件
	DevicePort	_sDEVICE_PORT		指定端口
	CfgValue	RS	–	根据SetER/ResetER确定数值
	NX_SerialSigCtl _instance	NX_SerialSigCtl	–	

外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active _Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1” 时，使用 E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量

```
// SetER或ResetER的检测
IF (NX_SerialSigCtl_instance.Done OR NX_SerialSigCtl_instance.Error) THEN
    OperatingStart:=FALSE;
ELSE_IF
    OperatingStart:=(SetER OR ResetER)
                    AND E001_NX_Unit_I_O_Data_Active_Status_63[1]
                    AND NOT(P_FirstRun);
    DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
    DevicePort.NxUnit:=N1_Node_location_information;
    DevicePort.PortNo:=2;
END_IF;

// 确定ER信号的值
CfgValue(Set:=SetER, Reset1:=ResetER);

// NX_SerialSigCtl指令的执行
NX_SerialSigCtl_instance(Execute:=OperatingStart,
                DevicePort:=DevicePort,
                Kind:=_eSERIAL_SIG#_SIG_ER,
                Sig:=CfgValue.Q1);
```

NX_SerialSigRead

读取扩展板串行端口的CS信号或DR信号。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialSigRead	串行控制信号的读取	FB	NX_SerialSigRead_instance NX_SerialSigRead Execute / DevicePort / Kind / TimeOut Done / Busy / Error / ErrorID / Sig	NX_SerialSigRead_instance(Execute, DevicePort, Kind, TimeOut, Done, Busy, Error, ErrorID, Sig);

使用注意事项

本指令只可用于NX1P2 CPU单元用的扩展板。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
Kind	信号指令		信号指令	_CS_SIG _DR_SIG *1	–	*2
TimeOut	超时时间		超时时间 “0”时为2.0s	遵从数据类型	0.1s	0
Sig	信号	输出	读取信号的输出	遵从数据类型	–	–

*1　无法使用_RS_SIG和_ER_SIG。如指定，执行指令时会发生异常。
*2　省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT　详情参阅功能说明																			
Kind	枚举体_eSERIAL_SIG　枚举元素参阅功能说明																			
TimeOut							○													
Sig	○																			

功能

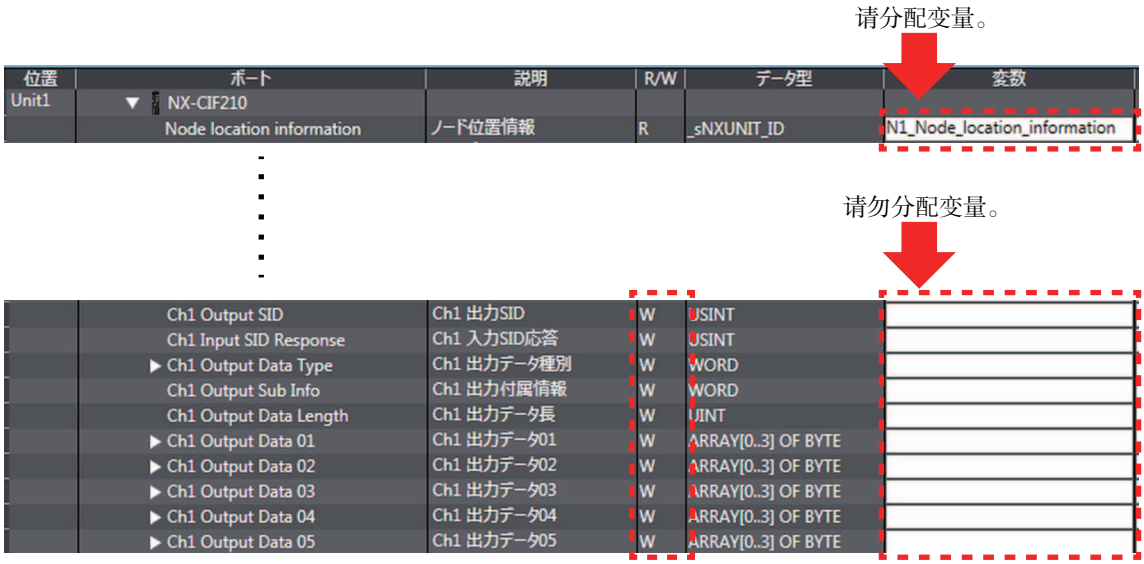
读取扩展板串行端口的CS信号或DR信号。
读取信号为ON时输出变量 “Sig” 为TRUE，信号OFF时 “Sig” 为FALSE。

输入变量 “DevicePort” 的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType” 指定设备种类。扩展板时请设定 _DeviceOptionBoard。用于设备指定的变量取决于指定的种类。
指定扩展板时，使用 “OptBoard” 指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为 “W” 的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 🕮 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceOptionBoard。

CS信号或DR信号通过输入变量 “Kind”选择。

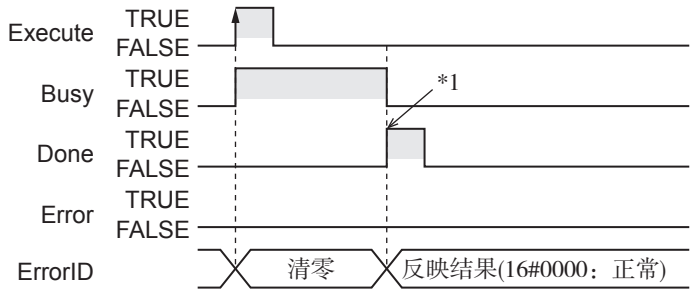
输入变量 “Kind”的数据类型为枚举体_eSERIAL_SIG。
枚举体_eSERIAL_SIG的枚举元素的含义如下所述。

枚举元素	含义
_RS_SIG	RS信号
_ER_SIG	ER信号
_CS_SIG	CS信号
_DR_SIG	DR信号

时序图

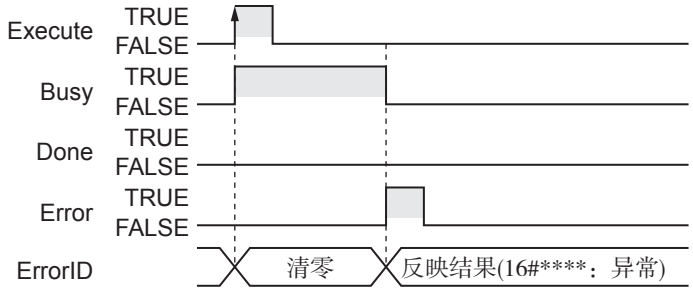
时序图如下所示。

● 正常结束时



*1 信号的读取完成

● 异常结束时



相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- 本指令不执行通信协议及接线状态的检查。请在确认接线状态及通信协议后使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Kind”、“DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - “DevicePort”指定了RS-422A/485的串行端口时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为 NX_SerialSigCtl 指令、NX_SerialSigRead 指令、NX_SerialStatusRead 指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令。
  - 串行通信超过超时时间时。
  - 对扩展板以外执行了本指令时。
  - 指定扩展板的串行通信模式为 “无协议”或 “Modbus-RTU主站”以外时。

示例程序

🕮 请参阅 “NX_SerialSigCtl指令(P.2-1259)”的示例程序。

NX_SerialStatusRead

读取扩展板串行端口的状态。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialStatusRead	串行端口状态的读取	FB	NX_SerialStatusRead_instance NX_SerialStatusRead Execute / DevicePort / TimeOut Done / Busy / Error / ErrorID / PortStatus	NX_SerialStatusRead_instance(Execute, DevicePort, TimeOut, Done, Busy, Error, ErrorID, PortStatus);

使用注意事项

本指令只可用于NX1P2 CPU单元用的扩展板。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
TimeOut	超时时间		超时时间 “0”时为2.0s	遵从数据类型	0.1s	0
PortStatus	端口状态	输出	读取的端口状态的输出	–	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
TimeOut							○													
PortStatus	结构体_sSERIAL_PORT_STATUS 详情参阅功能说明																			

功能

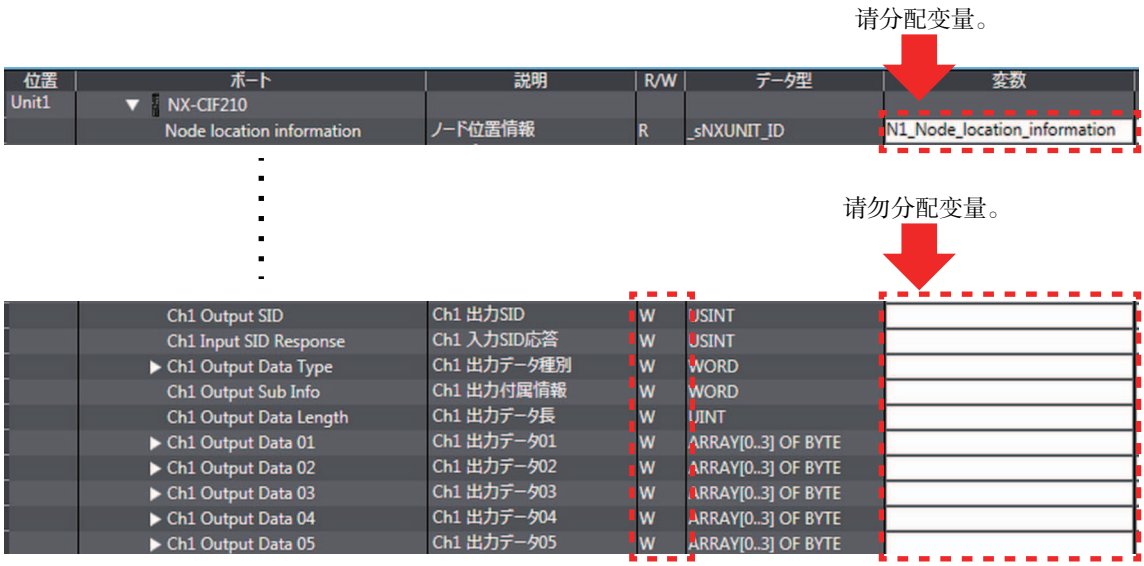
读取扩展板串行端口的状态。

输入变量 “DevicePort” 的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType” 指定设备种类。扩展板时请设定 _DeviceOptionBoard。用于设备指定的变量取决于指定的种类。
指定扩展板时，使用 “OptBoard” 指定设备。这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为 “W” 的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。
使用 “PortNo” 指定端口编号。
1：端口1
2：端口2
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceOptionBoard。

输出变量 “PortStatus” 的数据类型为结构体_sSERIAL_PORT_STATUS。规格如下所示。

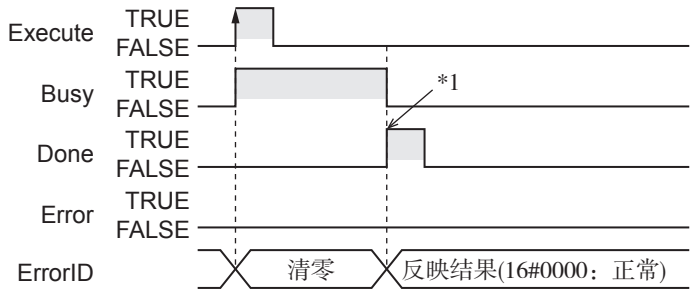
变量	名称	内容	数据类型	有效范围	单位	初始值
PortStatus	端口状态	读取的端口状态的输出	_sSERIAL_PORT_STATUS	–	–	–
FullRcvBuf	基于接收缓存已满的数据删除	TRUE：发生了数据删除*1 FALSE：未发生数据删除	BOOL	遵从数据类型	–	–
Reserved	保留	保留	保留	–	–	–

*1 接收缓存中的数据可能不完整。

时序图

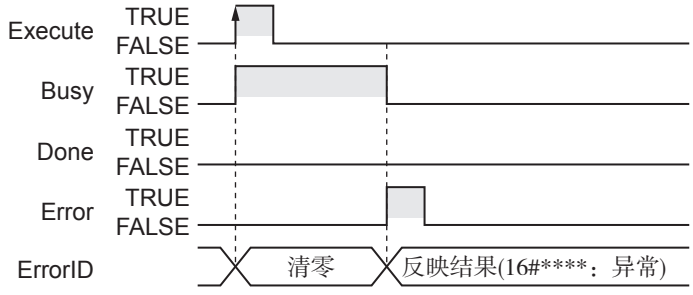
时序图如下所示。

● 正常结束时



*1 端口状态的读取完成

● 异常结束时



相关的系统定义变量

变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- 本指令不执行通信协议及接线状态的检查。请在确认接线状态及通信协议后使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Kind”、“DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令。
  - 串行通信超过超时时间时。
  - 对扩展板以外执行了本指令时。
  - 指定扩展板的串行通信模式为 “无协议”或 “Modbus-RTU主站”以外时。

示例程序

🕮 请参阅 “NX_SerialSigCtl指令(P.2-1259)”的示例程序。

NX_SerialBufClear

清除收发缓存。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialBufClear	缓存清除	FB	NX_SerialBufClear_instance NX_SerialBufClear Execute / Done DevicePort / Busy BufKind / Error TimeOut / ErrorID	NX_SerialBufClear_instance(Execute DevicePort, BufKind, TimeOut, Done, Busy, Error, ErrorID);

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
BufKind	缓存的种类		发送或接收缓存的种类指定	_BUF_SENDRCV _BUF_SEND _BUF_RCV	–	_BUFSENDRCV
TimeOut	超时时间		超时时间 “0”时为2.0s	遵从数据类型	0.1s	0

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT 详情参阅功能说明																			
BufKind	枚举体_eSERIAL_BUF_KIND 枚举元素参阅功能说明																			
TimeOut							○													

功能

指定端口和缓存的种类后，清除该缓存的数据。清除完成时，正常结束。

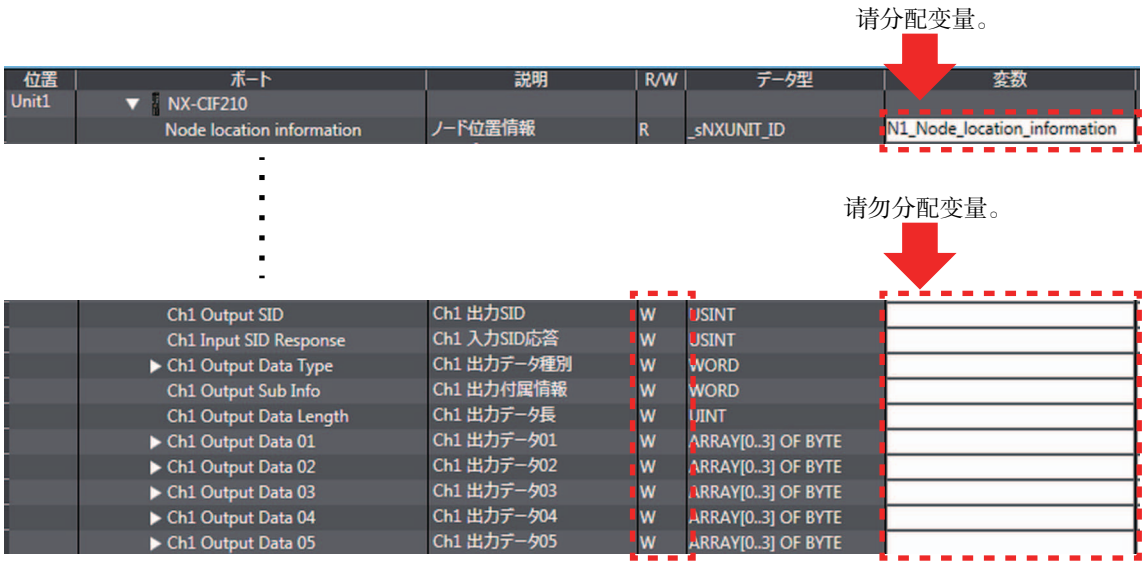
输入变量 “DevicePort” 的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType” 指定设备种类。NX单元时请设定_DeviceNXUnit，扩展板时请设定_DeviceOptionBoard。用于设备指定的变量取决于指定的种类。

指定NX单元时，使用 “NxUnit” 指定设备。
这种情况下，不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。
指定扩展板时，使用 “OptBoard” 指定设备。
这种情况下，不使用 “NxUnit”、“EcatSlave”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “OptBoard”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为“W”的对象，请勿分配设备变量。
例如，NX-CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2
NX单元时，指定端口1或端口2。
扩展板时，指定端口1。

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit或_DeviceOptionBoard。

端口通过 “Port”指定，要清除的缓存通过 “BufKind”指定。

清除接收缓存后，NX系列通信接口单元从外部设备接收的数据不会清除。

输入变量 “BufKind”的数据类型为枚举体_eSERIAL_BUF_KIND。
枚举体_eSERIAL_BUF_KIND的枚举元素的含义如下所述。

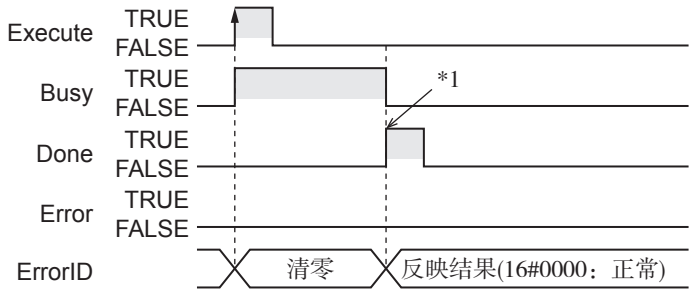
枚举元素	含义
_BUF_SENDRCV	发送缓存和接收缓存
_BUF_SEND	发送缓存
_BUF_RCV	接收缓存

对NX系列通信接口单元及扩展板以外的单元执行本指令时，会发生异常。

时序图

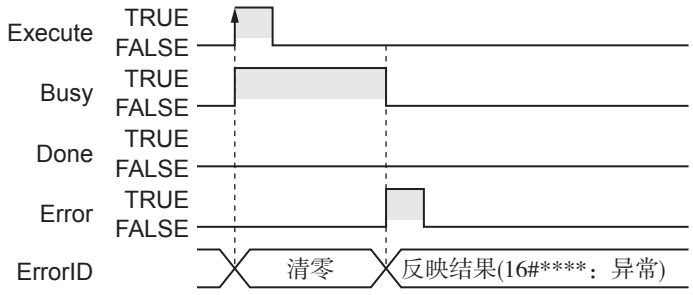
时序图如下所示。

● 正常结束时



*1 缓存清除完成

● 异常结束时



相关的系统定义变量

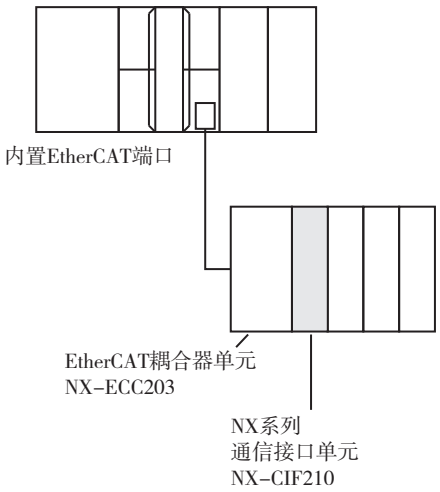
变量名称	名称	数据类型	内容
_PLC_OptBoardSta	扩展板状态	ARRAY[1..2] of _sOPTBOARD_STA	· 保存扩展板的相关状态。
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- 本指令不执行通信协议及接线状态的检查。请在确认接线状态及通信协议后使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “BufKind”、“DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令。
  - 串行通信超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。
  - 指定扩展板的串行通信模式为 “无协议”或 “Modbus-RTU主站”以外时。

示例程序

采用EtherCAT耦合器单元NX-ECC203连接NX系列通信接口单元NX-CIF210的构成。
NX-CIF210的单元编号为 “1”。



清除NX-CIF210的串行端口2的接收缓存。清除完成时，无起始符，结束符为等待接收 “CR”。

全局变量的定义

全局变量

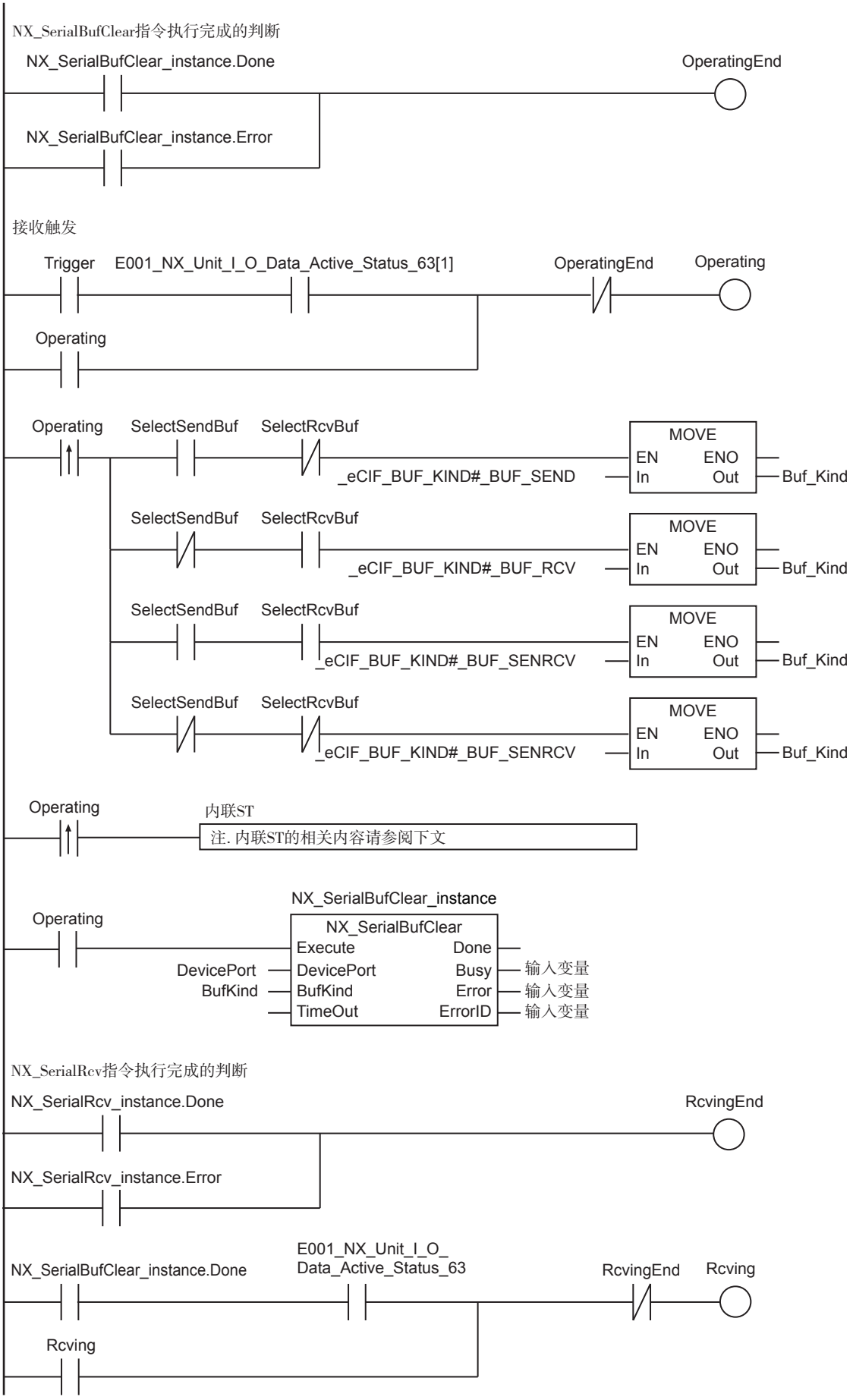
名称	数据类型	分配对象	注释
E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	ECAT://node#1/NX Unit I/O Data Active Status 125	可否使用63台NX单元的I/O数据
N1_Node_location_information	_sNXUNIT_ID	–	NX-CIF210指定的设备变量*1

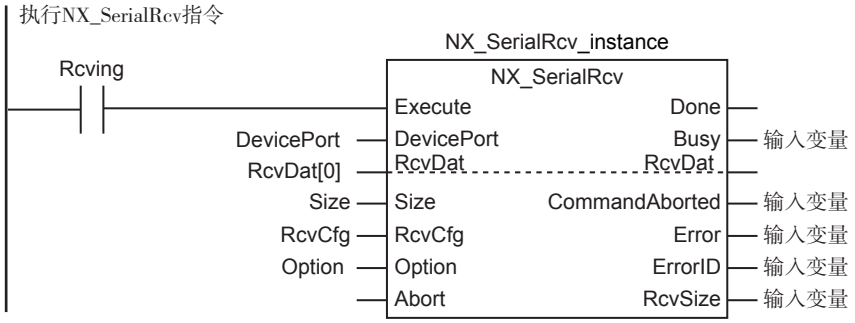
*1 使用 Sysmac Studio 右击 NX 系列从站终端的单元，选择 “显示节点位置端口” ，设定设备变量。详情请参阅 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	缓存清除处理结束
	Trigger	BOOL	FALSE	缓存清除执行条件
	Operating	BOOL	FALSE	缓存清除处理中
	SelectSendBuf	BOOL	FALSE	发送缓存选择
	SelectRcvBuf	BOOL	FALSE	接收缓存选择
	BufKind	_eSERIAL_BUF_KIND	_BUF_SENDRCV	缓存指定
	DevicePort	_sDEVICE_PORT		指定端口
	NX_SerialBufClear_instance	NX_SerialBufClear	–	
	RcvingEnd	BOOL		接收处理完成
	Rcving	BOOL		接收处理中
	RcvCfg	_sSERIAL_CFG		接收完成设定
	StartTrig	_eSERIAL_START	_SERIAL_START_NONE	
	StartCode	ARRAY[0..1] OF BYTE	[2(16#0)]	
	EndTrig	_eSERIAL_END	_SERIAL_END_CODE1	
	EndCode	ARRAY[0..1] OF BYTE	[16#0D,16#00]	结束符：CR
	RcvSizeCfg	UINT	0	
	Option	_sSERIAL_RCV_OPTION		
	TimeOut	TIME	TIME#0s	
	LastDatRcv	BOOL	FALSE	
	ClearBuf	BOOL	FALSE	

外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1”时，使用E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量


NX_SerialBufClear指令执行完成的判断
NX_SerialBufClear_instance.Done
OperatingEnd
NX_SerialBufClear_instance.Error
接收触发
Trigger
E001_NX_Unit_I_O_Data_Active_Status_63[1]
OperatingEnd
Operating
Operating
Operating
SelectSendBuf
SelectRcvBuf
MOVE
EN
ENO
In
Out
_eCIF_BUF_KIND#_BUF_SEND
Buf_Kind
SelectSendBuf
SelectRcvBuf
_eCIF_BUF_KIND#_BUF_RCV
Buf_Kind
SelectSendBuf
SelectRcvBuf
_eCIF_BUF_KIND#_BUF_SENRCV
Buf_Kind
SelectSendBuf
SelectRcvBuf
_eCIF_BUF_KIND#_BUF_SENRCV
Buf_Kind
Operating
内联ST
注. 内联ST的相关内容请参阅下文
NX_SerialBufClear_instance
Operating
NX_SerialBufClear
Execute
Done
DevicePort
DevicePort
Busy
输入变量
BufKind
BufKind
Error
输入变量
TimeOut
ErrorID
输入变量
NX_SerialRcv指令执行完成的判断
NX_SerialRcv_instance.Done
RcvingEnd
NX_SerialRcv_instance.Error
NX_SerialBufClear_instance.Done
E001_NX_Unit_I_O_
Data_Active_Status_63
RcvingEnd
Rcving
Rcving



● 内联ST的内容

```
DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
DevicePort.NxUnit:=N1_Node_location_information;
DevicePort.PortNo:=2;
```

ST

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	缓存清除处理结束
	Trigger	BOOL	FALSE	缓存清除执行条件
	Operating	BOOL	FALSE	缓存清除处理中
	SelectSendBuf	BOOL	FALSE	发送缓存选择
	SelectRcvBuf	BOOL	FALSE	接收缓存选择
	BufKind	_eSERIAL_BUF_KIND	_BUF_SENDRCV	缓存指定
	DevicePort	_sDEVICE_PORT		指定端口
	NX_SerialBufClear_instance	NX_SerialBufClear	–	
	RcvingEnd	BOOL		接收处理完成
	Rcving	BOOL		接收处理中
	RcvCfg	_sSERIAL_CFG		接收完成设定
	StartTrig	_eSERIAL_START	_SERIAL_START_NONE	
	StartCode	ARRAY[0..1] OF BYTE	[2(16#0)]	
	EndTrig	_eSERIAL_END	_SERIAL_END_CODE1	
	EndCode	ARRAY[0..1] OF BYTE	[16#0D,16#00]	结束符：CR
	RcvSizeCfg	UINT	0	
	Option	_sSERIAL_RCV_OPTION		
	TimeOut	TIME	TIME#0s	
	LastDatRcv	BOOL	FALSE	
	ClearBuf	BOOL	FALSE	

外部变量	名称	数据类型	注释
	E001_NX_Unit_I_O_Data_Active_Status_63	ARRAY[0..63] OF BOOL	· 可否使用63台NX单元的I/O数据 · 相应的单元编号为 “1”时，使用E001_NX_Unit_I_O_Data_Active_Status_63[1]
	N1_Node_location_information	_sNXUNIT_ID	NX-CIF210指定的设备变量

```
// 条件设定
RS_instance1(Set:=Trigger AND E001_NX_Unit_I_O_Data_Active_Status_63[1]
         Reset1:=OperatingEnd,
         Q1=>Operating);
R_Trigger_instance(Clk:=Operating);
IF ( (R_Trigger_instance.Q=TRUE) ) THEN
        DevicePort.DeviceType:=_eDEVICE_TYPE#_DeviceNXUnit;
        DevicePort.NxUnit:=N1_Node_location_information;
        DevicePort.PortNo:=2;
        IF( (SelectSendBuf=TRUE) THEN
                IF(SelectRcvBuf=TRUE) THEN
                        BufKind:=_eSERIAL_BUF_KIND#_BUF_SENDRCV;
                ELSE
                        BufKind:=_eSERIAL_BUF_KIND#_BUF_SEND;
                END_IF;
        ELSE
                IF (SelectRcvBuf=TRUE) THEN
                        BufKind:=_eSERIAL_BUF_KIND#_BUF_RCV;
                ELSE
                        BufKind:=_eSERIAL_BUF_KIND#_BUF_SENDRCV;
                END_IF
        END_IF;
END_IF;

// 执行缓存清除
NX_SerialBufClear_instance(Execute:=Operating,
                    DevicePort:=DevicePort,
                    BufKind:=BufKind);

//
RS_instane2(Set:=NX_SerialBufClear.Done AND E001_NX_Unit_I_O_Data_Active_Status_63[1],
         Reset1:=NX_SerialRcv_instance.Done OR NX_SerialRcv_instance.Error,
         Q1=>Rcving);

//
NX_SerialRcv_instance(Execute:=Rcving,
                DevicePort:=DevicePort,
                RcvDat:=RcvDat[0],
                Size:=Size,
                RcvCfg:=RcvCfg,
                Option:=Option);
```

NX_SerialStartMon

开始NX系列通信接口单元的串行线路监控。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialStartMon	串行线路监控的开始	FB	NX_SerialStartMon_instance NX_SerialStartMon Execute / Done DevicePort / Busy Continuous / Error TimeOut / ErrorID	NX_SerialStartMon_instance(Execute, DevicePort, Continuous, TimeOut, Done, Busy, Error, ErrorID);

使用注意事项

本指令无法在NX1P2 CPU单元用的扩展板中使用。

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
Continuous	连续监控		串行线路监控的动作方法的指定 TRUE:连续 FALSE:单次	遵从数据类型	–	FALSE
TimeOut	超时时间		超时时间 “0”时为2.0s	遵从数据类型	0.1s	0

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT　详情参阅功能说明																			
Continuous	○																			
TimeOut							○													

功能

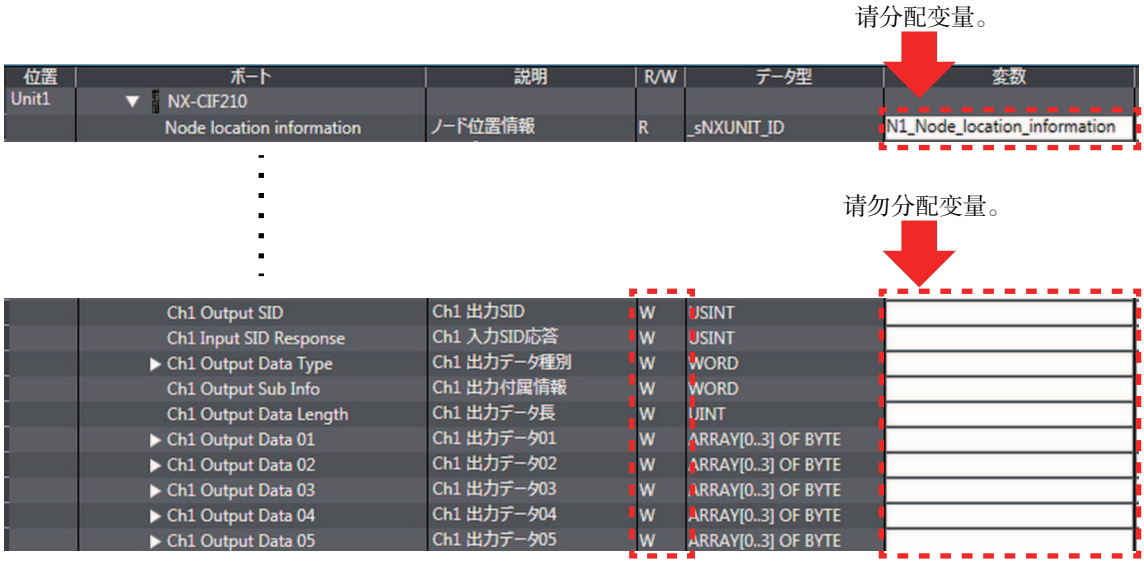
开始NX系列通信接口单元的串行线路监控。
串行线路监控启动时，正常结束。

输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType”指定设备种类。NX单元时请设定_DeviceNXUnit。用于设备指定的变量取决于指定的种类。
该指令使用 “NxUnit”指定设备。不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为 “W”的对象，请勿分配设备变量。
例如，NX–CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 🕮 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

该指令可指定_DeviceNXUnit。

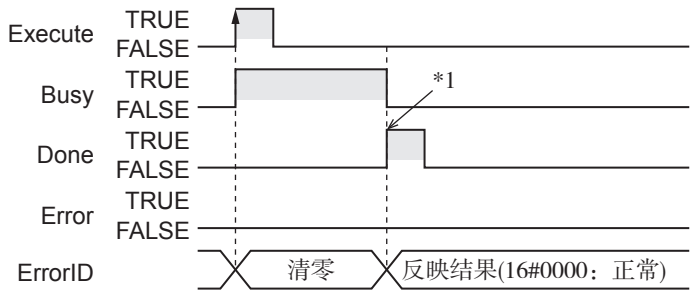
输入变量 “Continuous”为TRUE时为连续监控，将持续监控直至执行NX_SerialStopMon指令。
输入变量 “Continuous”为 FALSE 时为单次监控，将持续执行串行线路监控直至缓存已满或执行NX_SerialStopMon指令。

对NX系列通信接口单元以外的单元执行本指令时，会发生异常。

时序图

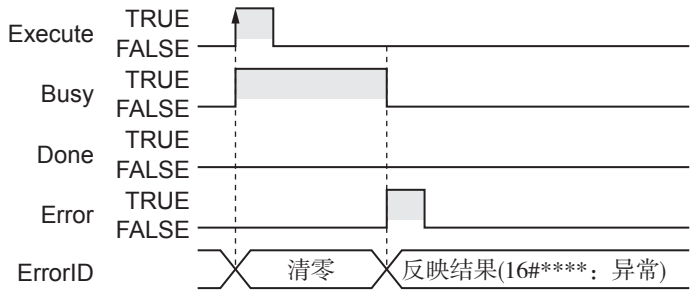
时序图如下所示。

● 正常结束时



*1 开始串行线路监控

● 异常结束时



相关的系统定义变量

变量名称	名称	数据类型	内容
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令。
  - 串行通信超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。

NX_SerialStopMon

停止NX系列通信接口单元的串行线路监控。

指令	名称	FB/FUN	图形表现	ST表现
NX_SerialStopMon	串行线路监控的停止	FB	NX_SerialStopMon_instance NX_SerialStopMon Execute / DevicePort / TimeOut Done / Busy / Error / ErrorID	NX_SerialStopMon_instance(Execute, DevicePort, TimeOut, Done, Busy, Error, ErrorID);

使用注意事项

本指令无法在NX1P2 CPU单元用的扩展板中使用。

版本相关信息

本指令可用于CPU单元Ver.1.11以上且Sysmac Studio Ver.1.15以上。

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DevicePort	设备端口	输入	表示设备端口的对象	–	–	–
TimeOut	超时时间		超时时间 “0”时为2.0s	遵从数据类型	0.1s	0

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DevicePort	结构体_sDEVICE_PORT　详情参阅功能说明																			
TimeOut							○													

功能

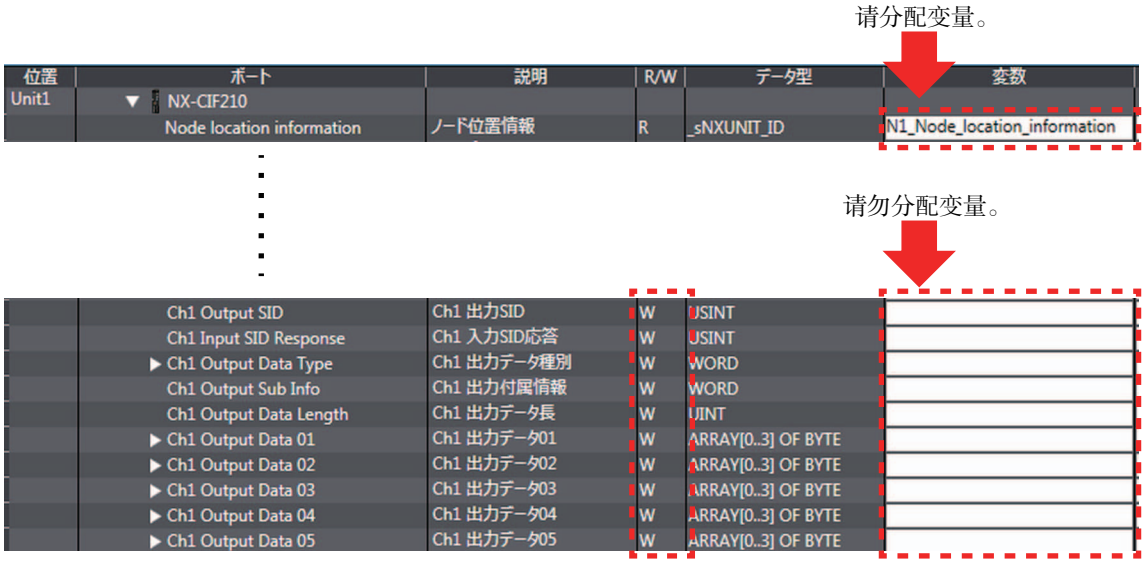
停止NX系列通信接口单元的串行线路监控。
串行线路监控停止时，正常结束。

输入变量 “DevicePort”的数据类型为结构体_sDEVICE_PORT。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
DevicePort	设备端口	表示设备端口的对象	_sDEVICE_PORT	–	–	–
DeviceType	设备类型	指定设备种类	_eDEVICE_TYPE	_DeviceNXUnit _DeviceEcatSlave _DeviceOption Board	–	–
NxUnit	指定单元	指定控制对象的NX单元	_sNXUNIT_ID	–	–	–
EcatSlave	指定从站	指定控制对象的EtherCAT从站	_sECAT_ID	–	–	–
OptBoard	指定扩展板	指定控制对象的扩展板	_sOPTBOARD_ID	–	–	–
Reserved	保留	保留	保留	–	–	–
PortNo	端口编号	端口编号的指定 1：端口1 2：端口2	USINT	遵从数据类型	–	–

使用 “DeviceType”指定设备种类。NX单元时请设定_DeviceNXUnit。用于设备指定的变量取决于指定的种类。
该指令使用 “NxUnit”指定设备。不使用 “EcatSlave”、“OptBoard”。
请将分配至指定设备I/O映射上节点位置信息的设备变量传输至 “NxUnit”。

使用该指令时，请务必将设备变量分配至节点位置信息，对于节点位置信息以后的 I/O 端口且 R/W 栏为 “W”的对象，请勿分配设备变量。
例如，NX–CIF210的端口1使用该指令时，如下所述。



将设备变量分配至节点位置信息的方法请参阅 🕮 “Sysmac Studio Version1 操作手册(SBCA-362G以上)”。

使用 “PortNo”指定端口编号。
1：端口1
2：端口2

“DeviceType”的数据类型为枚举体_eDEVICE_TYPE。
枚举体_eDEVICE_TYPE的枚举元素的含义如下所述。

枚举元素	含义
_DeviceNXUnit	指定NX单元
_DeviceEcatSlave	指定EtherCAT从站
_DeviceOptionBoard	指定扩展板

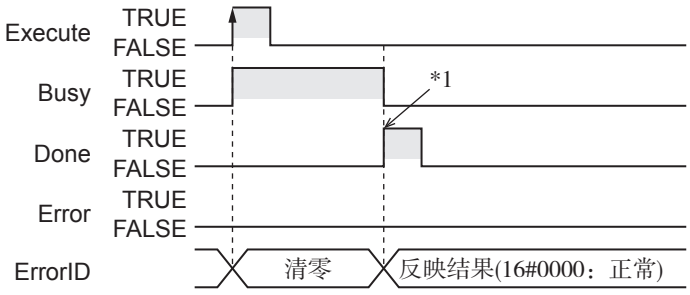
该指令可指定_DeviceNXUnit。

对NX系列通信接口单元以外的单元执行本指令时，会发生异常。

时序图

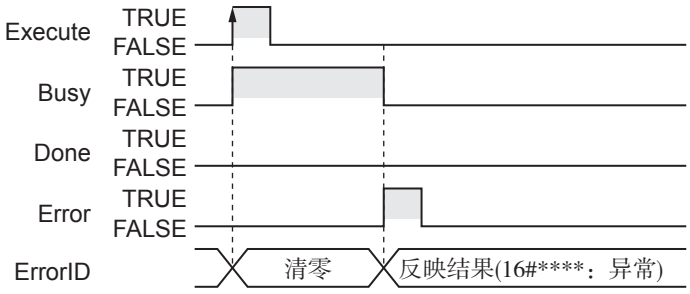
时序图如下所示。

● 正常结束时



*1 停止串行线路监控

● 异常结束时



相关的系统定义变量

变量名称	名称	数据类型	内容
_NXB_UnitIOActiveTbl	NX单元I/O数据通信中状态	ARRAY[0..8] OF BOOL	· 表示可否与NX单元进行I/O数据通信。 · 数组的下标与NX单元编号相对应。下标0表示NX总线主站。

使用注意事项

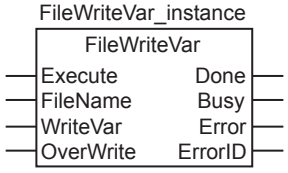
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- 在事件任务中使用了本指令时，将发生编译错误。因此，无法在事件任务中使用。
- NX 系列通信接口单元重启时，可能会发生 “CIF 单元初始化”。请根据需要重新执行数据的发送或接收。
- 使用本指令时，对于作为对象的NX系列通信接口单元，请勿对Sysmac Studio的I/O映射画面中R/W栏为“W”的I/O端口分配设备变量。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “DevicePort.DevicePortType”、“DevicePort.PortNo”中指定了超出范围的值时。
  - “DevicePort”指定了不存在的单元、扩展板或端口时。
  - 同时执行NX_SerialSend指令、NX_SerialRcv指令、NX_ModbusRtuCmd指令、NX_ModbusRtuRead指令、NX_ModbusRtuWrite指令、NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令超过32个时。
  - 指定了与执行中的以下指令指定的设备端口变量相同的设备端口变量，执行本指令时。
    对象指令为NX_SerialSigCtl指令、NX_SerialSigRead指令、NX_SerialStatusRead指令、NX_SerialBufClear指令、NX_SerialStartMon指令、NX_SerialStopMon指令。
  - 串行通信超过超时时间时。
  - 对NX系列通信接口单元或扩展板以外的单元执行了本指令时。

SD存储卡指令

指令	名称	页码
FileWriteVar	变量文件写入	2–1294
FileReadVar	变量文件读取	2–1299
FileOpen	文件打开	2–1304
FileClose	文件关闭	2–1308
FileSeek	文件查找	2–1311
FileRead	文件读取	2–1314
FileWrite	文件写入	2–1322
FileGets	字符串读取	2–1330
FilePuts	字符串写入	2–1338
FileCopy	文件复制	2–1346
FileRemove	文件删除	2–1355
FileRename	文件名变更	2–1360
DirCreate	目录创建	2–1366
DirRemove	目录删除	2–1369
BackupToMemoryCard	SD存储卡备份	2–1373

FileWriteVar

以二进制格式将1个变量值写入SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FileWriteVar	变量文件写入	FB		FileWriteVar_instance(Execute, FileName, WriteVar, OverWrite, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileName	指定文件名	输入	写入文件名	最大66字节(65个半角英数字字符+结尾NULL字符)	–	''
WriteVar	指定变量		写入变量	遵从数据类型		*1
OverWrite	允许覆盖		TRUE ：允许覆盖 FALSE：禁止覆盖			FALSE

*1 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileName																				○
WriteVar	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素																			
OverWrite	○																			

功能

以二进制格式将1个变量 “WriteVar”值写入SD存储卡内 “FileName”指定的文件中。
也可给 “WriteVar”指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素。

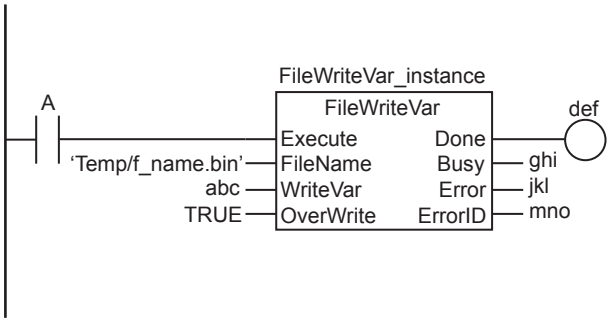
SD存储卡内不存在与 “FileName”同名的文件时，新建文件。
“FileName”包含目录且SD存储卡内不存在该目录时，按不同的目录分别新建。注：仅在指定目录的最底层不存在时，按不同的目录分别新建。

SD存储卡内已存在与 “FileName”同名的文件时，根据允许覆盖 “OverWrite”的值，执行下述处理。

“OverWrite”的值	处理
TRUE(允许覆盖)	覆盖至该文件。
FALSE(禁止覆盖)	不覆盖至该文件时会发生异常。

示例如下所示。将整个数组变量abc写入名为'Temp/f_name.bin'的文件。将abc设定为元素数量3的INT型数组变量。

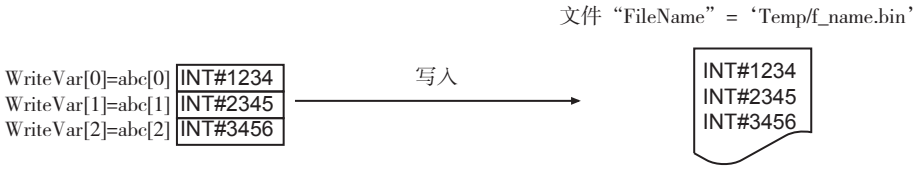
LD



ST

```
FileWriteVar_instance(A, 'Temp/f_name.bin', abc,
                      TRUE, def, ghi, jkl, mno);
```

以二进制格式将变量 “WriteVar”的值写入
SD存储卡内 “FileName”指定的文件中。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹(虚拟SD存储卡)。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 请务必将传输至 “WriteVar”的输入参数设为变量。如果传输常数，编连时会发生异常。
- “WriteVar”为枚举体时，无法直接传输枚举元素。如果直接传输枚举元素，编连时会发生异常。
- 指定文件的容量大于 “WriteVar”的容量时不会发生异常，仅写入对应 “WriteVar”容量的数据。而且，执行本指令后，指定文件会根据 “WriteVar”的容量而减小。
- 以字节为单位，按照低位字节→高位字节的顺序(低字节序)排列写入的数据。
- “WriteVar”为整个结构体时，根据构成的不同，可能会插入各结构要素间的调整用区域。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - SD存储卡空间不足时。
  - “FileName”的值并非正确的文件名时。
  - 超出可创建的文件数量、目录数量时。
  - 已存在与 “FileName”同名的文件且正在访问时。
  - 已存在与 “FileName”同名的文件且 “OverWrite”的值为FALSE时。

- 已存在与 “FileName”同名的文件且该文件禁止写入时。
- 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
- “FileName”的值超出了可作为文件名使用的字节数。
- SD存储卡访问过程中发生某种异常导致无法访问时。

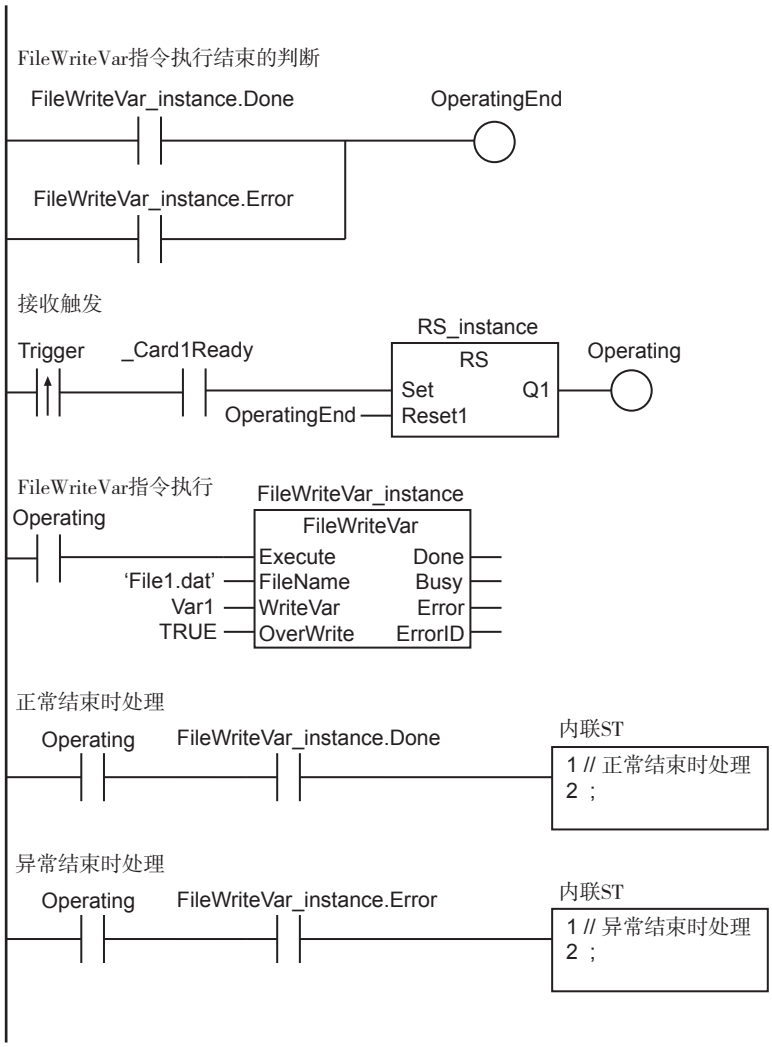
示例程序

将整个数组变量Var1[ ]写入文件 ‘File1.dat’。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	Var1	ARRAY[0..999] OF INT	[1000(0)]	写入数据
	RS_instance	RS		
	FileWriteVar_instance	FileWriteVar		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志



ST

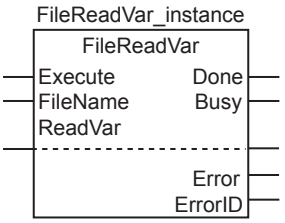
内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	Var1	ARRAY[0..999] OF INT	[1000(0)]	指定变量
	FileWriteVar_instance	FileWriteVar		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// FileWriteVar指令初始化
IF (OperatingStart=TRUE) THEN
    FileWriteVar_instance(
        Execute   :=FALSE,
        WriteVar :=Var1);
    OperatingStart:=FALSE;
END_IF;

// FileWriteVar指令执行
IF (Operating=TRUE) THEN
    FileWriteVar_instance(
        Execute    :=TRUE,
        FileName   := 'File1.dat',// 指定文件名
        WriteVar   :=Var1,          // 指定变量
        OverWrite  :=TRUE);         // 允许覆盖

    IF (FileWriteVar_instance.Done=TRUE) THEN
        // 正常结束时处理
        Operating:=FALSE;
    END_IF;

    IF (FileWriteVar_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating:=FALSE;
    END_IF;
END_IF;
```

FileReadVar

以二进制格式读取SD存储卡内指定文件的值，并写入变量。

指令	名称	FB/FUN	图形表现	ST表现
FileReadVar	变量文件读取	FB		FileReadVar_instance(Execute, FileName, ReadVar, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileName	指定文件名	输入	读取文件名	最大66字节(65个半角英数字字符+结尾NULL字符)	–	''
ReadVar	读取对象变量	输入输出	读取值的保存对象变量	遵从数据类型	–	–

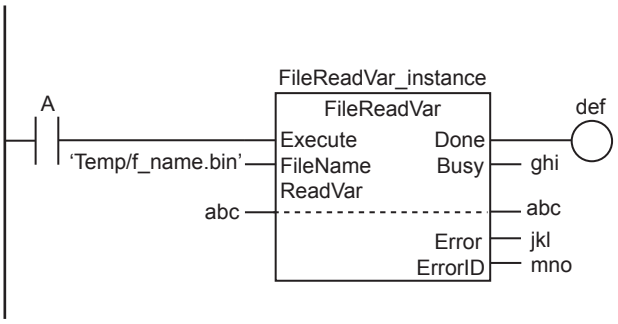
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileName																				○
ReadVar	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素																			

功能

以二进制格式读取SD存储卡内 “FileName”指定的文件内部值。将读取的值代入读取对象变量“ReadVar”。
也可给 “ReadVar”指定枚举体、整个数组、数组的1个元素、整个结构体、结构体的1个结构要素。

示例如下所示。读取名为‘Temp/f_name.bin’的文件内容，并写入数组变量abc[ ]。将abc设定为元素数量3的INT型数组变量。

LD



ST

```
FileReadVar_instance(A, 'Temp/f_name.bin', abc,
                     def, ghi, jkl, mno);
```

以二进制格式读取SD存储卡内“FileName”指定的文件内部值，
并代入变量“ReadVar”。

相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 指定文件的容量大于 “ReadVar”的容量时不会发生异常，仅读取对应 “ReadVar”容量的数据。
- 指定文件的容量小于 “ReadVar”的容量时不会发生异常，仅读取对应指定文件容量的数据。而且，“ReadVar”的剩余区域可保持执行本指令前的值。
- 以字节为单位，按照低位字节→高位字节的顺序(低字节序)排列读取的数据。
- “ReadVar”为整个结构体时，根据构成的不同，可能会插入各结构要素间的调整用区域。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- “ReadVar”无法指定设备变量。如指定，读取的值不会代入 “ReadVar”。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - “FileName”指定的文件不存在时。
  - “FileName”的值并非正确的文件名时。
  - “FileName”指定的文件正在访问时。
  - 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
  - “FileName”的值超出了可作为文件名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

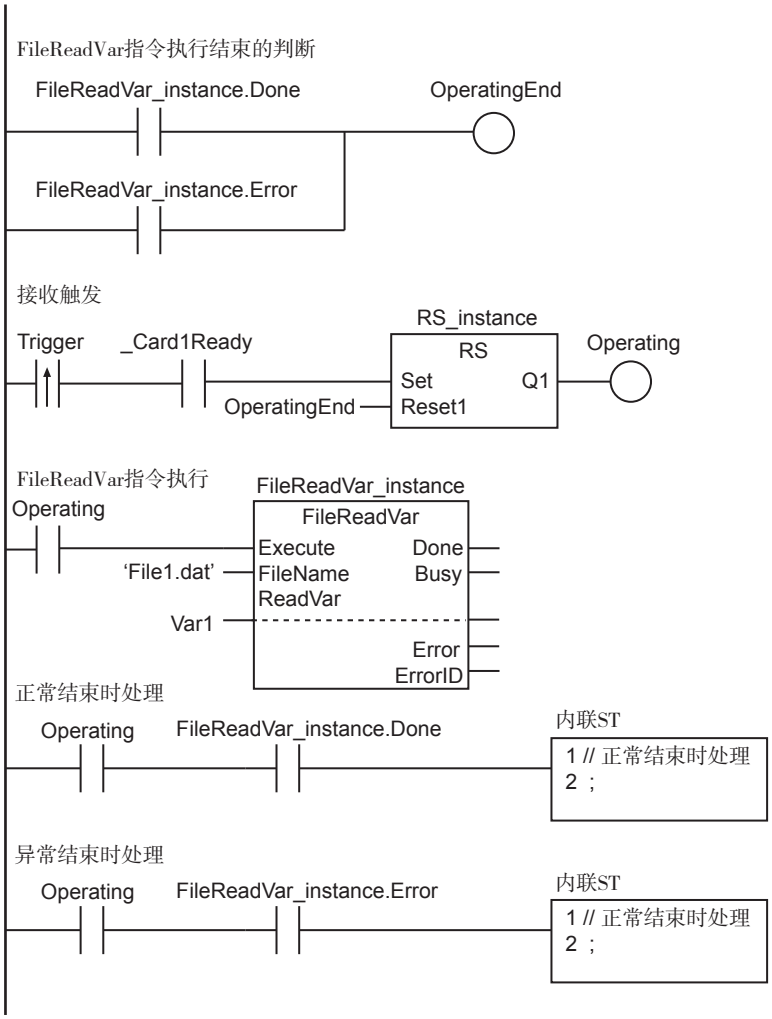
示例程序

读入文件 ‘File1.dat’的内容，并保存至数组变量Var1。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	Var1	ARRAY[0..999] OF INT	[1000(0)]	读取数据
	RS_instance	RS		
	FileReadVar_instance	FileReadVar		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志



ST

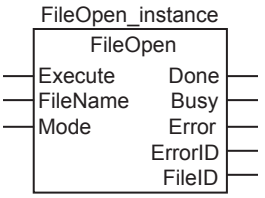
内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	Var1	ARRAY[0..999] OF INT	[1000(0)]	读取对象变量
	FileReadVar_instance	FileReadVar		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// Trigger上升沿检测
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// FileReadVar指令初始化
IF (OperatingStart=TRUE) THEN
    FileReadVar_instance(
        Execute   :=FALSE,
        ReadVar   :=Var1);
    OperatingStart:=FALSE;
END_IF;

// FileReadVar指令执行
IF (Operating=TRUE) THEN
    FileReadVar_instance(
        Execute   :=TRUE,
        FileName :=‘File1.dat’,// 指定文件名
        ReadVar   :=Var1);          // 读取对象变量

    IF (FileReadVar_instance.Done=TRUE) THEN
        // 正常结束时处理
        Operating:=FALSE;
    END_IF;

    IF (FileReadVar_instance.Error=TRUE) THEN
        // 异常结束时处理
        Operating:=FALSE;
    END_IF;
END_IF;
```

FileOpen

打开SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FileOpen	文件打开	FB		FileOpen_instance(Execute, FileName, Mode, Done, Busy, Error, ErrorID, FileID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileName	指定文件名	输入	待打开的文件名	最大66字节(65个半角英数字字符+结尾NULL字符)	–	''
Mode	打开模式		文件打开模式	(*)		_READ _EXIST
FileID	文件ID	输出	打开文件的ID	遵从数据类型	–	–

* _READ_EXIST，_RDWR_EXIST，_WRITE_CREATE，_RDWR_CREATE，_WRITE_APPEND，_RDWR_APPEND

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileName																				○
Mode	枚举体_eFOPEN_MODE　枚举元素参阅功能说明																			
FileID				○																

功能

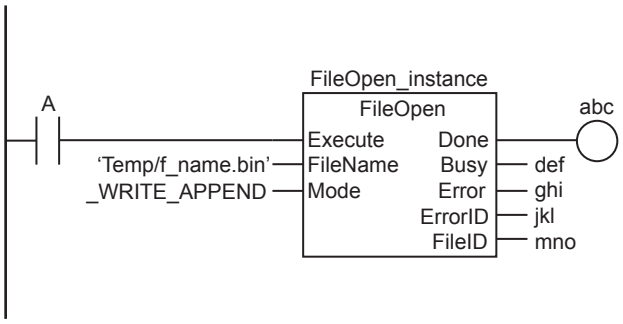
在 “Mode”指定的模式下打开SD存储卡内 “FileName”指定的文件。
打开文件后，输出文件ID “FileID”。通过FileRead指令、FileWrite指令等指定文件时使用 “FileID”。

“Mode”的数据类型为枚举体_eFOPEN_MODE。枚举元素的含义如下所示。

枚举元素	含义
_READ_EXIST	为了读取文本文件而打开。从文件的起始部分开始读取。
_RDWR_EXIST	打开文件，用于读取及写入。从文件的起始部分开始读取及写入。
_WRITE_CREATE	打开文件，用于写入。已存在文件时，删除其内容，将文件容量设定为0。不存在文件时，新建文件。从文件的起始部分开始写入。 注：已存在文件且在该文件中设定了禁止写入时发生异常，无法打开文件。
_RDWR_CREATE	打开文件，用于读取及写入。已存在文件时，删除其内容，将文件容量设定为0。不存在文件时，新建文件。从文件的起始部分开始读取及写入。
_WRITE_APPEND	打开文件，用于追加写入。不存在文件时，新建文件。从文件的末尾开始追加写入。 注：已存在文件且在该文件中设定了禁止写入时发生异常，无法打开文件。
_RDWR_APPEND	打开文件，用于读取及追加写入。不存在文件时，新建文件。从文件的起始部分开始读取。从文件的末尾开始追加写入。

示例如下所示。打开名为 ‘Temp/f_name.bin’的文件，用于追加写入。将文件ID代入变量mno。

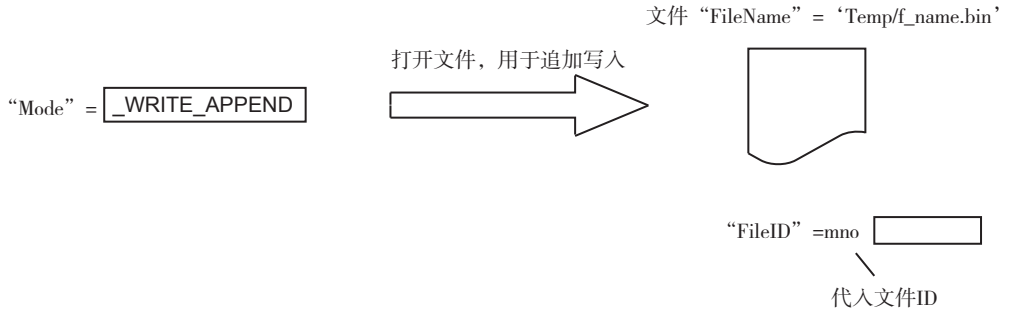
LD



ST

```
FileOpen_instance(A, 'Temp/f_name.bin',
                  _WRITE_APPEND, abc,
                  def, ghi, jkl, mno);
```

打开SD存储卡内“FileName”指定的文件，用于追加写入。
将文件ID代入变量“FileID”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 必须在FileSeek指令、FileRead指令、FileWrite指令、FileGets指令、FilePuts指令之前事先执行本指令。
- 请务必在使用通过本指令执行文件打开操作的文件后执行FileClose指令，并关闭文件。
- 本指令结束时，将数值保存至 “FileID”。即 “Done”的值从FALSE变为TRUE的时间点。
- 在文件打开的状态下将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX 系列 CPU 单元在打开文件的状态下，通过操作 SD 存储卡供电停止按钮停止供电时，文件不会损坏。然而，由于文件打开状态持续，因此请执行FileClose，并关闭文件。
- NJ/NX系列CPU单元在打开文件的状态下，不操作SD存储卡供电停止按钮即拔下SD存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下，即使不操作SD存储卡供电停止按钮即拔下SD存储卡时，文件打开状态仍将持续。请执行FileClose指令，并关闭文件。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡时，文件打开状态虽将持续，但即使重新安装SD存储卡，也无法读写文件。为了执行文件的读写，请暂时关闭文件，然后重新打开文件。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。

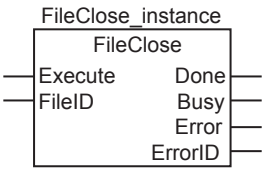
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - “Mode”的值为_READ_EXIST或_RDWR_EXIST且不存在 “FileName”指定的文件时。
  - “FileName”的值并非正确的文件名时。
  - 超出可创建的文件数量、目录数量时。
  - “FileName”指定的文件正在访问时。
  - “FileName”指定的文件禁止写入时。
  - 希望同时打开5个以上文件时。
  - “FileName”的值超出了可作为文件名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。
  - “Mode”的值超过有效范围时。
  - CPU单元Ver.1.10以上版本打开已打开的文件时，会发生 “文件访问中”错误，输出变量 “FileID”中将保存已打开的文件ID。发生其他错误时，输出变量 “FileID”不变。
    CPU单元Ver.1.09以下版本发生错误时，输出变量 “FileID”中将保存 “0”。

示例程序

请参阅 “FileRead指令(P.2–1314)”、“FileWrite指令(P.2–1322)”、“FileGets指令(P.2–1330)”、“FilePuts指令(P.2–1338)”的示例程序。

FileClose

关闭SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FileClose	文件关闭	FB		FileClose_instance(Execute, FileID, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileID	文件ID	输入	待关闭文件的ID	遵从数据类型	–	0

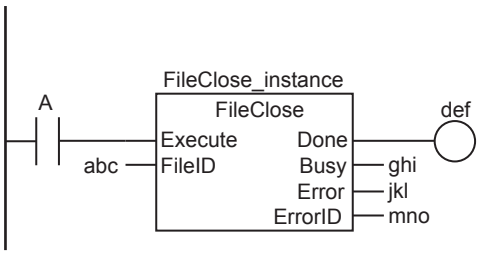
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileID				○																

功能

关闭SD存储卡内 “FileID”指定的文件。

示例如下所示。关闭将变量abc的值设定为文件ID的文件。

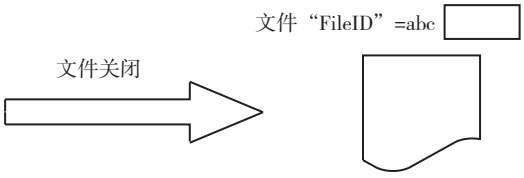
LD



ST

```
FileClose_instance(A, abc, def, ghi, jkl, mno);
```

关闭SD存储卡内“FileID”指定的文件。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

必须事先通过FileOpen指令打开文件的指令为FileSeek指令、FileRead指令、FileWrite指令、FileGets指令、FilePuts指令。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 必须事先执行FileOpen指令，并获取 “FileID”。
- 请务必在使用通过FileOpen指令执行文件打开操作的文件后执行本指令，并关闭文件。
- 在文件打开的状态下将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX系列CPU单元在打开文件的状态下通过供电停止开关操作停止供电时，文件不会损坏。然而，由于文件打开状态持续，因此请执行FileClose，并关闭文件。
- NJ/NX 系列 CPU 单元在打开文件的状态下不操作供电停止开关即拔下 SD 存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下，即使不操作供电停止开关即拔下SD存储卡时，文件打开状态仍将持续。请执行FileClose指令，并关闭文件。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡时，文件打开状态虽将持续，但即使重新安装SD存储卡，也无法读写文件。为了执行文件的读写，请暂时关闭文件，然后重新打开文件。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。

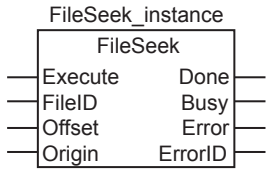
- 以下情况时会发生异常。“Error”变为TRUE。
  - “FileID”指定的文件不存在时。
  - “FileID”指定的文件已关闭时。
  - “FileID”指定的文件正在访问时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。
  - SD存储卡并非可使用状态时。

示例程序

请参阅 “FileRead指令(P.2–1314)”、“FileWrite指令(P.2–1322)”、“FileGets指令(P.2–1330)”、“FilePuts指令(P.2–1338)”的示例程序。

FileSeek

为SD存储卡内的指定文件设定文件位置指示器。

指令	名称	FB/FUN	图形表现	ST表现
FileSeek	文件查找	FB		FileSeek_instance(Execute, FileID, Offset, Origin, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileID	文件ID	输入	设定文件位置指示器的文件ID	遵从数据类型	–	0
Offset	偏置		从 “Origin”起的偏置位置		字节	
Origin	基准位置		文件位置指示器的基准位置	_SEEK_SET, _SEEK_CUR, _SEEK_END	–	_SEEK_SET

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileID				○																
Offset												○								
Origin	枚举体_eFSEEK_ORIGIN　枚举元素参阅功能说明																			

功能

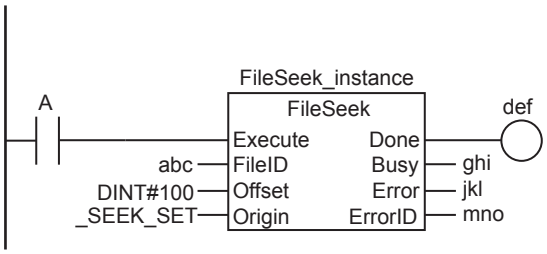
为SD存储卡内文件ID “FileID”指定的文件设定文件位置指示器。所谓文件位置指示器，是指执行FileRead指令及FileWrite指令等之后，开始读取、写入的文件内的位置。例如，希望从文件的起始部分执行读取时，通过FileSeek指令将文件位置指示器设定为文件的起始部分，然后执行FileRead指令。
以基准位置 “Origin”中添加偏置 “Offset”的位置为文件位置指示器设定位置。

“Origin”的数据类型为枚举体_eFSEEK_ORIGIN。枚举元素的含义如下所示。

枚举元素	含义
_SEEK_SET	文件的起始部分
_SEEK_CUR	当前文件位置指示器的位置
_SEEK_END	文件的末尾

示例如下所示。将文件位置指示器设定至从文件起始部分起100字节的位置。

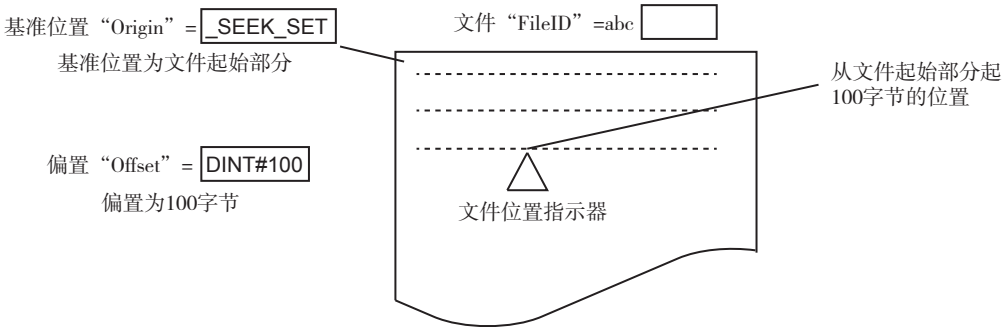
LD



ST

```
FileSeek_instance(A, abc, DINT#100,
                  _SEEK_SET, def,
                  ghi, jkl, mno);
```

为SD存储卡内“FileID”指定的文件设定文件位置指示器。
从文件起始部分起添加“Offset”的位置为文件位置指示器的位置。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

使用注意事项

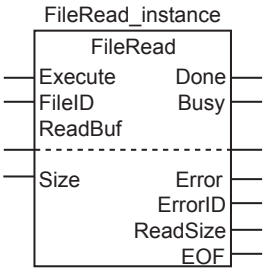
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 执行本指令前，必须事先执行FileOpen指令，并获取 “FileID”。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - “Origin”的值超过有效范围时。
  - SD存储卡并非可使用状态时。
  - “FileID”指定的文件不存在时。
  - “FileID”指定的文件正在访问时。
  - “Origin”与 “Offset”指定的位置超出文件容量时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

示例程序

请参阅 “FileRead指令(P.2–1314)”、“FileWrite指令(P.2–1322)”的示例程序。

FileRead

读取SD存储卡内指定文件的数据。

指令	名称	FB/FUN	图形表现	ST表现
FileRead	文件读取	FB		FileRead_instance(Execute, FileID, ReadBuf, Size, Done, Busy, Error, ErrorID, ReadSize, EOF);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileID	文件ID	输入	待读取文件的ID	遵从数据类型	–	0
Size	读取元素数量		待读取的元素数量			1
ReadBuf[ ]数组	读取缓存	输入输出	读取数据的写入对象	遵从数据类型	–	–
ReadSize	实际读取的元素数量	输出	实际读取的元素数量	遵从数据类型	–	–
EOF	文件结尾		判定是否到达文件结尾 TRUE ：到达 FALSE：未到达			

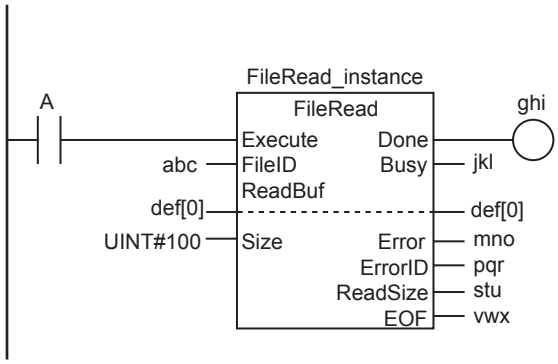
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileID				○																
Size							○													
ReadBuf[ ]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
ReadSize							○													
EOF	○																			

功能

从SD存储卡内文件ID “FileID”指定的文件、文件位置指示器的某个位置起读取数据，并保存至读取缓存ReadBuf[ ]。
事先通过FileSeek指令，将文件位置指示器设定至任意位置。
待读取的数据量为(ReadBuf[ ]数据类型的容量)× “Size”。即ReadBuf[ ] “Size”个的元素量。
ReadBuf[ ]允许为将枚举体设定为元素的数组、将结构体设定为元素的数组。
将实际读取的元素数量保存至 “ReadSize” 。通常情况下，“Size”的值与 “ReadSize”的保持一致。如果从文件位置指示器的某个位置至文件结尾的数据量小于 “Size”，则不会发生异常，将至文件结尾的数据保存至ReadBuf[ ]。此时，“ReadSize”的值小于 “Size”的值。
此外，读取至文件结尾时，文件结尾 “EOF”的值变为TRUE。否则，“EOF”的值变为FALSE。

示例如下所示。如果读取缓存def[ ]为BYTE型数组，则从文件中读取100字节的数据。

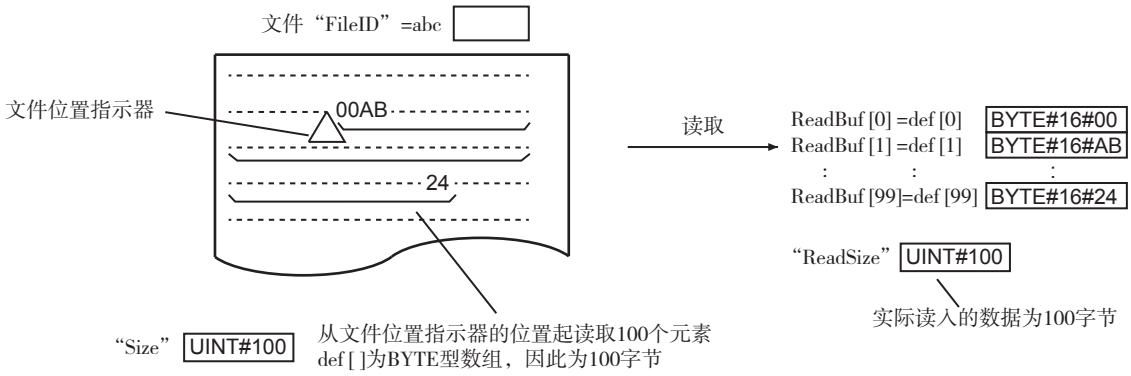
LD



ST

```
FileRead_instance(A, abc, def[0], UINT#100,
                  ghi, jkl, mno, pqr, stu, vwx);
```

从SD存储卡内“FileID”指定的带文件位置指示器的位置起仅读取“Size”元素，并保存至读取缓存ReadBuf[ ]。
然后，将实际读取的数据容量输出至“ReadSize”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 读取至文件结尾的数据且其容量用ReadBuf[ ]数据类型的容量除不尽时，删除不符合ReadBuf[ ]数据容量的数据。文件位置指示器进入文件结尾，“EOF”的值变为TRUE。
- ReadBuf[ ]第 “Size”个之后的元素(不通过读取进行覆盖的元素)保持执行本指令前的值。
- 执行本指令前，必须事先执行FileOpen指令，并获取 “FileID”。
- 以字节为单位，按照低位字节→高位字节的顺序(低字节序)排列读取的数据。
- 本指令结束时，将数值保存至 “EOF”。即 “Done”的值从FALSE变为TRUE的时间点。
- ReadBuf[ ]为将结构体设定为元素的数组时，根据构成的不同，可能会插入各结构要素间的调整用区域。
- 在执行本指令过程中将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - ReadBuf[ ]的数组元素数量小于 “Size”的值时。
  - SD存储卡并非可使用状态时。
  - “FileID”指定的文件不存在时。
  - “FileID”指定的文件正在访问时。
  - 未通过可读取的模式打开 “FileID”指定的文件时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

示例程序

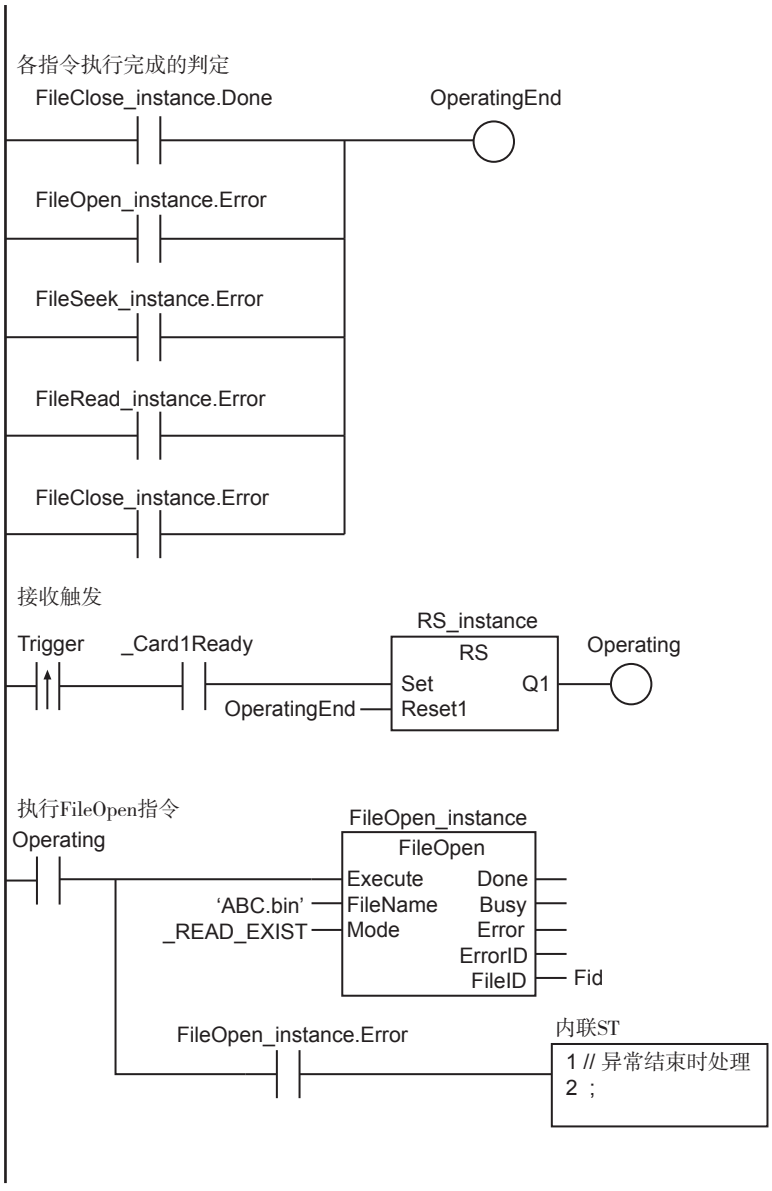
从文件 ‘ABC.bin’起始部分的第2字节位置起读取4字节的数据，并保存至BYTE型数组变量InDat[ ]。
处理步骤如下所示。

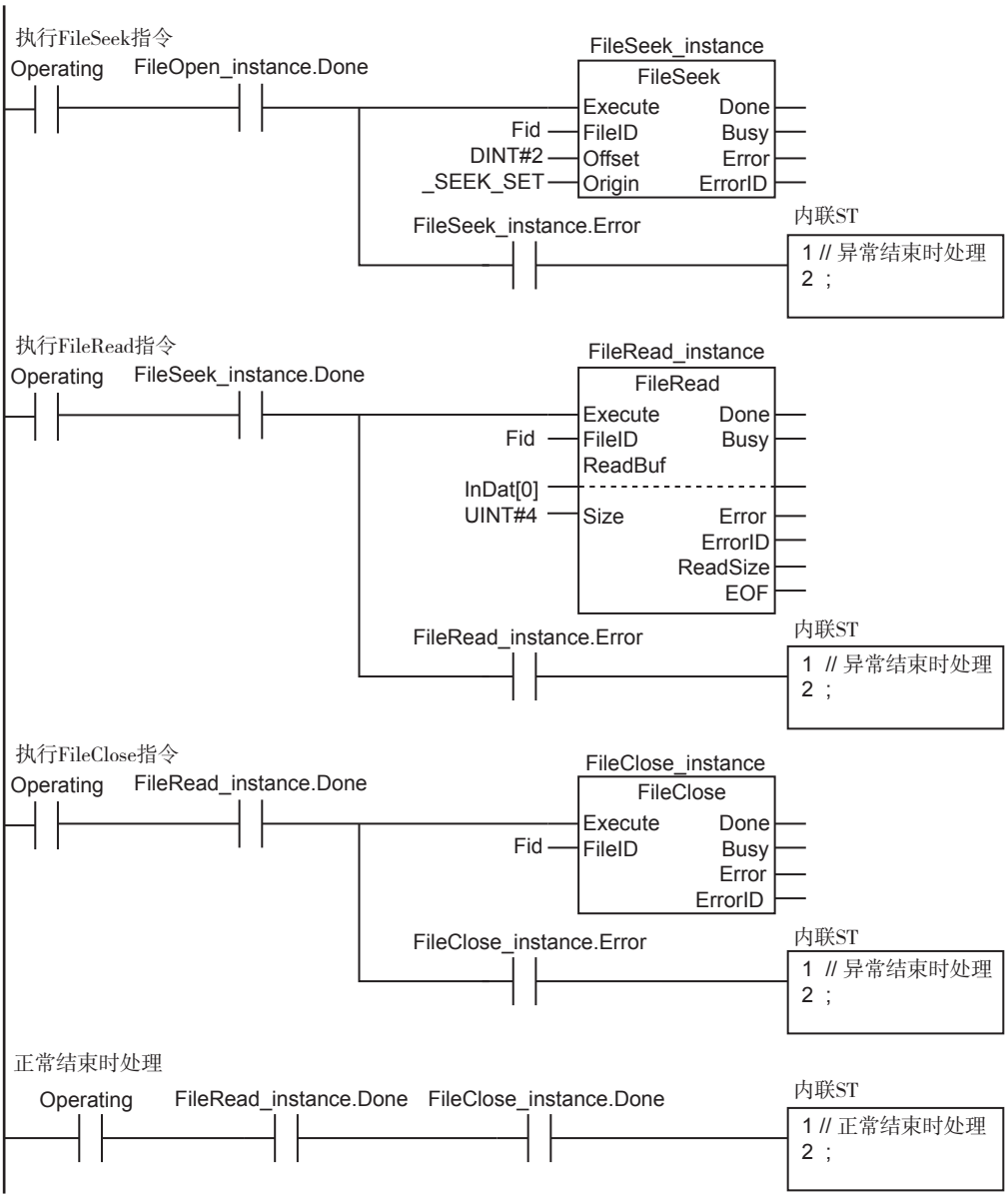
1. 使用FileOpen指令，打开文件'ABC.bin'。
2. 使用FileSeek指令，将文件位置指示器设定至从文件起始部分起的第2字节。
3. 使用FileRead指令，从文件位置指示器的位置起读取4字节的数据，并保存至数组变量InDat[ ]。
4. 使用FileClose指令，关闭'ABC.bin'。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	Fid	DWORD	16#0	文件ID
	InDat	ARRAY[0..999] OF BYTE	[1000(16#0)]	读取数据
	RS_instance	RS		
	FileOpen_instance	FileOpen		
	FileSeek_instance	FileSeek		
	FileRead_instance	FileRead		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志




执行FileSeek指令
Operating
FileOpen_instance.Done
FileSeek_instance
FileSeek
Execute
Done
Fid
FileID
Busy
DINT#2
Offset
Error
_SEEK_SET
Origin
ErrorID
FileSeek_instance.Error
内联ST
1 // 异常结束时处理
2 ;
执行FileRead指令
Operating
FileSeek_instance.Done
FileRead_instance
FileRead
Execute
Done
Fid
FileID
Busy
ReadBuf
InDat[0]
UINT#4
Size
Error
ErrorID
ReadSize
EOF
FileRead_instance.Error
内联ST
1 // 异常结束时处理
2 ;
执行FileClose指令
Operating
FileRead_instance.Done
FileClose_instance
FileClose
Execute
Done
Fid
FileID
Busy
Error
ErrorID
FileClose_instance.Error
内联ST
1 // 异常结束时处理
2 ;
正常结束时处理
Operating
FileRead_instance.Done
FileClose_instance.Done
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	InDat	ARRAY[0..999] OF BYTE	[1000(16#0)]	读取数据
	Stage	INT	0	状态变化
	Fid	DWORD	16#0	文件ID
	FileOpen_instance	FileOpen		
	FileSeek_instance	FileSeek		
	FileRead_instance	FileRead		
	FileClose_instance	FileClose		

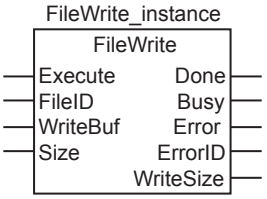
外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileOpen_instance(Execute :=FALSE);      // 实例初始化
    FileSeek_instance(Execute  :=FALSE);     // 实例初始化
    FileRead_instance(
        Execute  :=FALSE,                    // 实例初始化
        ReadBuf :=InDat[0]);                 // 虚拟
    FileClose_instance(Execute:=FALSE);      // 实例初始化
    Stage            :=INT#1;
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                                      // 文件打开
        FileOpen_instance(
            Execute   :=TRUE,
            FileName := 'ABC.bin',           // 文件名
            Mode      :=_READ_EXIST,         // 文件读取
            FileID    =>Fid);                // 文件ID

    IF (FileOpen_instance.Done=TRUE) THEN
        Stage:=INT#2;                        // 正常结束
    END_IF;

    IF (FileOpen_instance.Error=TRUE) THEN
        Stage:=INT#99;                       // 异常结束
    END_IF;
```

```
    2 :                                     // 文件查找
        FileSeek_instance(
            Execute :=TRUE,
            FileID  :=Fid,                  // 文件ID
            Offset  :=DINT#2,               // 文件位置指示器 起始部分起第2字节
            Origin  :=_SEEK_SET);           //

    IF (FileSeek_instance.Done=TRUE) THEN
        Stage:=INT#3;                       // 正常结束
    END_IF;

    IF (FileSeek_instance.Error=TRUE) THEN
        Stage:=INT#99;                      // 异常结束
    END_IF;

    3 :                                     // 文件读取
        FileRead_instance(
            Execute  :=TRUE,
            FileID   :=Fid,                 // 文件ID
            ReadBuf :=InDat[0],             // 读取缓存
            Size     :=UINT#4);             // 读取元素数量 4字节

    IF (FileRead_instance.Done=TRUE) THEN
        Stage:=INT#4;                       // 正常结束
    END_IF;

    IF (FileRead_instance.Error=TRUE) THEN
        Stage:=INT#99;                      // 异常结束
    END_IF;

    4 :                                     // 文件关闭
        FileClose_instance(
            Execute :=TRUE,
            FileID  :=Fid);                  // 文件ID

        IF (FileClose_instance.Done=TRUE) THEN
            Operating:=FALSE;               // 正常结束
        END_IF;

        IF (FileClose_instance.Error=TRUE) THEN
            Stage:=INT#99;                  // 异常结束
        END_IF;

    99 :
        Operating:=FALSE;                   // 异常结束处理
    END_CASE;
END_IF;
```

FileWrite

将数据写入SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FileWrite	文件写入	FB		FileWrite_instance(Execute, FileID, WriteBuf, Size, Done, Busy, Error, ErrorID, WriteSize);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileID	文件ID	输入	待写入文件的ID	遵从数据类型	–	0
WriteBuf[ ]数组	写入缓存		待写入的数据			(*)
Size	写入元素数量		待写入的元素数量			1
WriteSize	实际写入的元素数量	输出	实际写入的元素数量	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

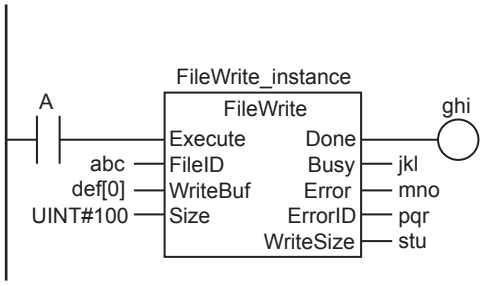
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileID				○																
WriteBuf[ ]数组	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○	○
	也可指定以枚举体为元素的数组、以结构体为元素的数组																			
Size							○													
WriteSize							○													

功能

将数据写入SD存储卡内文件ID “FileID”指定的文件、文件位置指示器的某个位置。
事先通过FileSeek指令，将文件位置指示器设定至任意位置。
待写入的数据为写入缓存WriteBuf[ ]的内容。
待写入的数据量为(WriteBuf[ ]数据类型的容量) “Size”。即WriteBuf[ ] “Size”个的元素量。
WriteBuf[ ]允许为将枚举体设定为元素的数组、将结构体设定为元素的数组。
将实际写入的数据容量输出至 “WriteSize”。

示例如下所示。如果写入缓存def[ ]为BYTE型，则将100字节的数据写入文件。

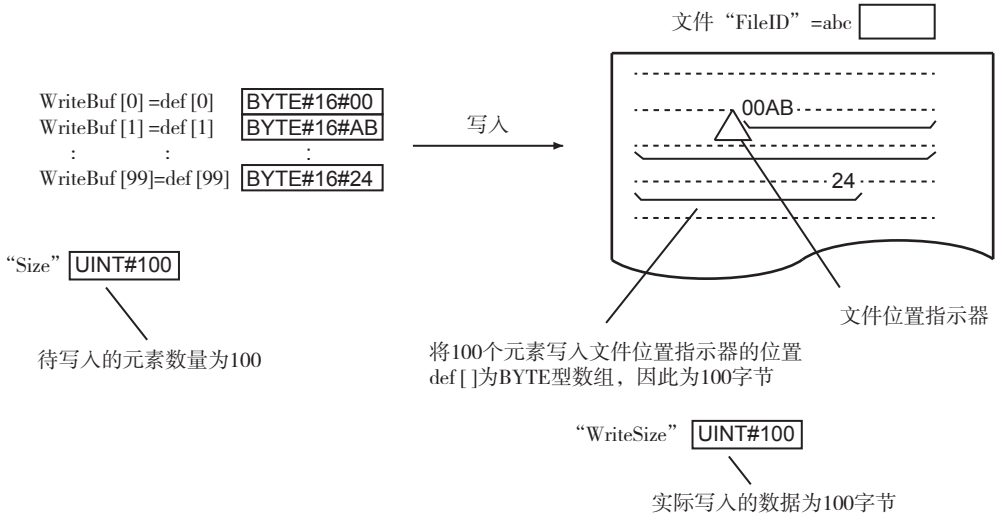
LD



ST

```
FileWrite_instance(A, abc, def[0], UINT#100, ghi,
                   jkl, mno, pqr, stu);
```

仅以“Size”元素为单位，将写入缓存WriteBuf[ ]的内容写入SD存储卡内“FileID”指定的文件、文件位置指示器的某个位置。
然后，将实际写入的数据容量输出至“WriteSize”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 执行本指令前，必须事先执行FileOpen指令，并获取 “FileID”。
- 以字节为单位，按照低位字节→高位字节的顺序(低字节序)排列写入的数据。
- WriteBuf[ ]为将结构体设定为元素的数组时，根据构成的不同，可能会插入各结构要素间的调整用区域。
- 在执行本指令过程中将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - WriteBuf[ ]的数组元素数量小于 “Size”的值时。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - SD存储卡空间不足时。
  - “FileID”指定的文件不存在时。
  - “FileID”指定的文件正在访问时。
  - 未通过可写入的模式打开 “FileID”指定的文件时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

示例程序

将4字节的数据写入从文件名 ‘ABC.bin’起始部分起的第2字节位置。待写入的数据为BYTE型数组变量OutDat[ ]的内容。
处理步骤如下所示。

1 使用FileOpen指令，打开文件'ABC.bin'。

2 使用FileSeek指令，将文件位置指示器设定至从文件起始部分起的第2字节。

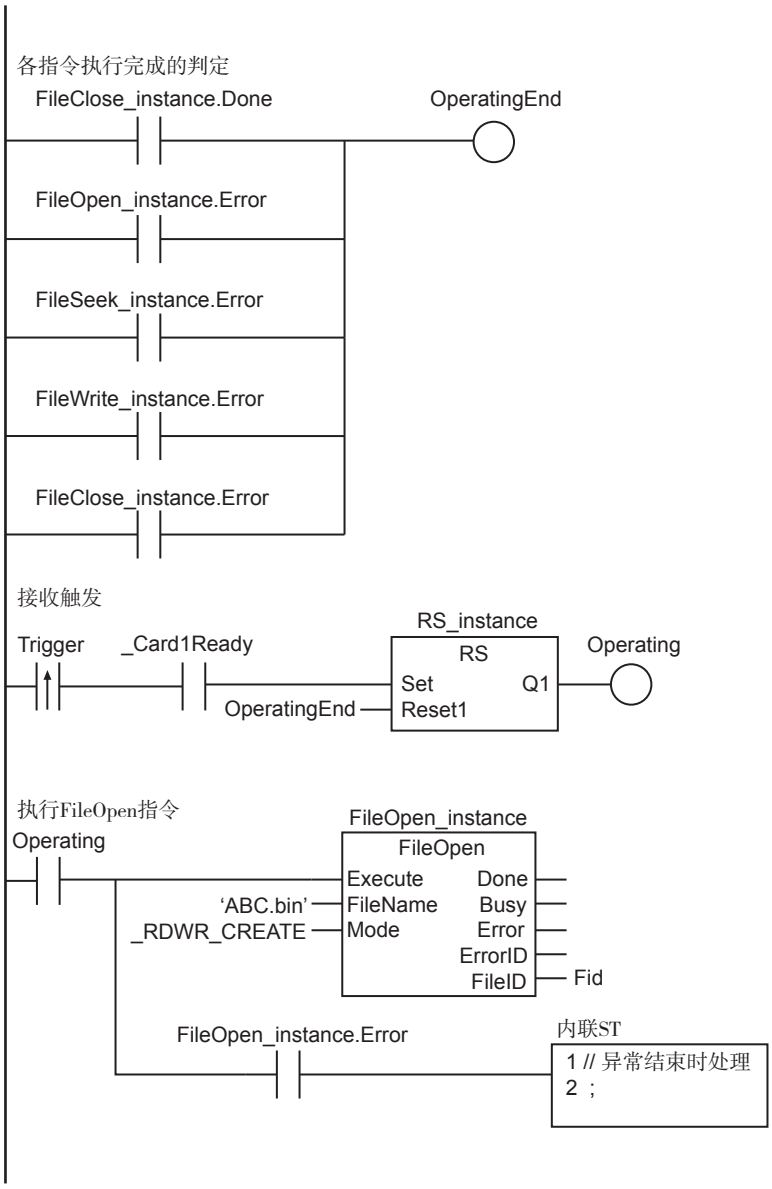
3 使用FileWrite指令，将4字节数组变量OutDat[ ]的值写入文件位置指示器的位置。

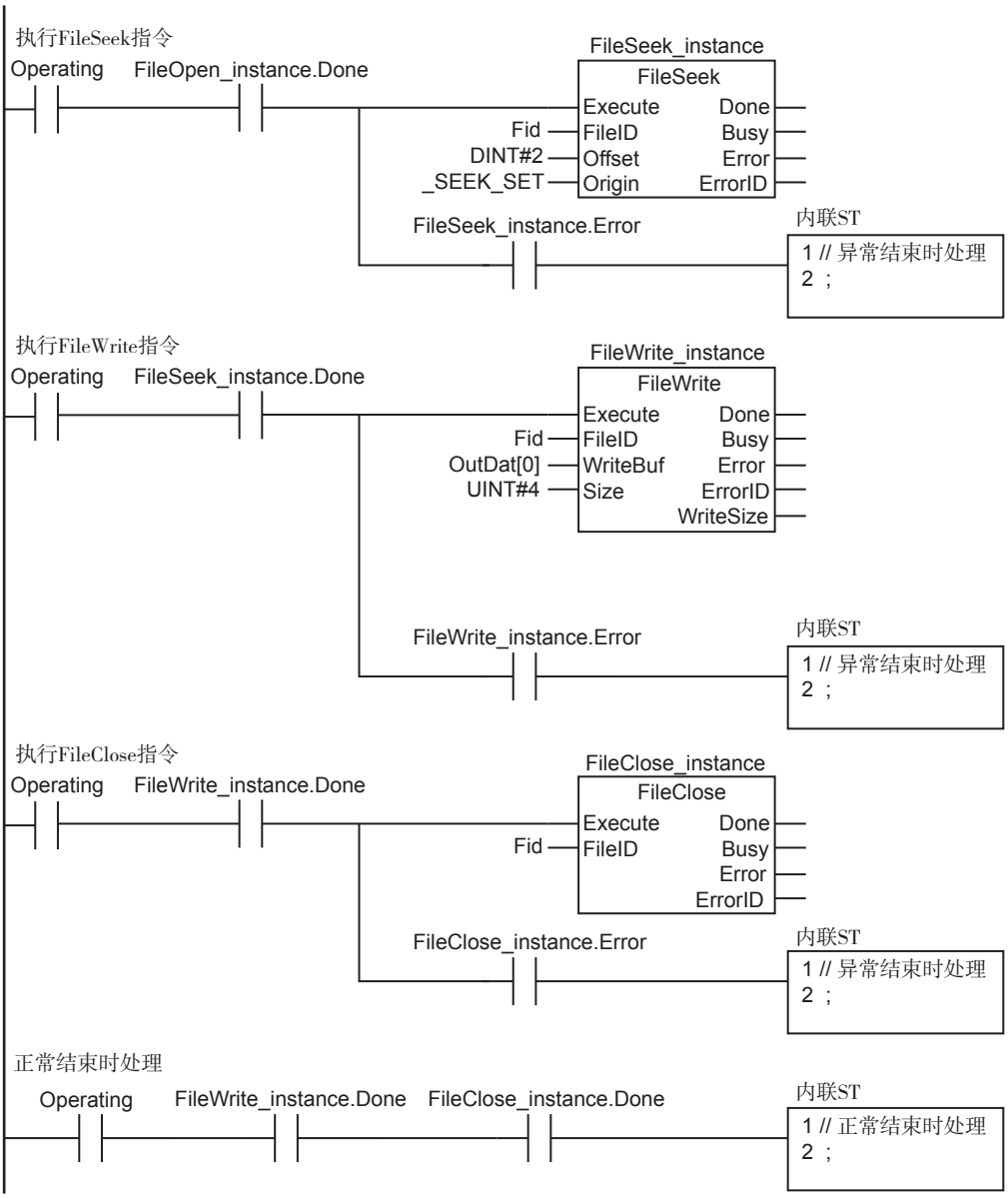
4 使用FileClose指令，关闭'ABC.bin'。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	Fid	DWORD	16#0	文件ID
	OutDat	ARRAY[0..999] OF BYTE	[1000(16#0)]	写入数据
	RS_instance	RS		
	FileOpen_instance	FileOpen		
	FileSeek_instance	FileSeek		
	FileWrite_instance	FileWrite		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志




执行FileSeek指令
Operating
FileOpen_instance.Done
FileSeek_instance
FileSeek
Execute
Done
Fid
FileID
Busy
DINT#2
Offset
Error
_SEEK_SET
Origin
ErrorID
FileSeek_instance.Error
内联ST
1 // 异常结束时处理
2 ;
执行FileWrite指令
Operating
FileSeek_instance.Done
FileWrite_instance
FileWrite
Execute
Done
Fid
FileID
Busy
OutDat[0]
WriteBuf
Error
UINT#4
Size
ErrorID
WriteSize
FileWrite_instance.Error
内联ST
1 // 异常结束时处理
2 ;
执行FileClose指令
Operating
FileWrite_instance.Done
FileClose_instance
FileClose
Execute
Done
Fid
FileID
Busy
Error
ErrorID
FileClose_instance.Error
内联ST
1 // 异常结束时处理
2 ;
正常结束时处理
Operating
FileWrite_instance.Done
FileClose_instance.Done
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	OutDat	ARRAY[0..999] OF BYTE	[1000(16#0)]	写入数据
	Stage	INT	0	状态变化
	Fid	DWORD	16#0	文件ID
	FileOpen_instance	FileOpen		
	FileSeek_instance	FileSeek		
	FileWrite_instance	FileWrite		
	FileClose_instance	FileClose		

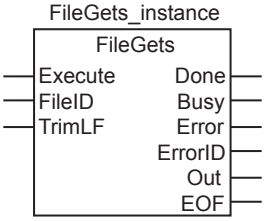
外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileOpen_instance(Execute :=FALSE);
    FileSeek_instance(Execute  :=FALSE);
    FileWrite_instance(
        Execute   :=FALSE,
        WriteBuf :=OutDat[0]);
    FileClose_instance(Execute:=FALSE);
    Stage             :=INT#1;
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                          // 文件打开
        FileOpen_instance(
            Execute   :=TRUE,
            FileName := 'ABC.bin',          // 文件名
            Mode      :=_RDWR_CREATE,       // 文件读取及写入
            FileID    =>Fid);               // 文件ID

        IF (FileOpen_instance.Done=TRUE) THEN
            Stage:=INT#2;            // 正常结束
        END_IF;

        IF (FileOpen_instance.Error=TRUE) THEN
            Stage:=INT#99;           // 异常结束
        END_IF;
```

```
    2 :                         // 文件查找
        FileSeek_instance(
            Execute :=TRUE,
            FileID  :=Fid,                      // 文件ID
            Offset  :=DINT#2,                   // 文件位置指示器 起始部分起第2字节
            Origin  :=_SEEK_SET);               //

        IF (FileSeek_instance.Done=TRUE) THEN
            Stage:=INT#3;           // 正常结束
        END_IF;

        IF (FileSeek_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    3 :                         // 文件写入
        FileWrite_instance(
            Execute   :=TRUE,
            FileID    :=Fid,                    // 文件ID
            WriteBuf  :=OutDat[0],              // 写入缓存
            Size      :=UINT#4);                // 写入元素数量 4字节

        IF (FileWrite_instance.Done=TRUE) THEN
            Stage:=INT#4;           // 正常结束
        END_IF;

        IF (FileWrite_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    4 :                         // 文件关闭
        FileClose_instance(
            Execute :=TRUE,
            FileID  :=Fid);                     // 文件ID

        IF (FileClose_instance.Done=TRUE) THEN
            Operating:=FALSE;       // 正常结束
        END_IF;

        IF (FileClose_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    99 :
        Operating:=FALSE;           // 异常结束处理
    END_CASE;
END_IF;
```

FileGets

从SD存储卡内的指定文件读取1行字符串。

指令	名称	FB/FUN	图形表现	ST表现
FileGets	字符串读取	FB		FileGets_instance(Execute, FileID, TrimLF, Done, Busy, Error, ErrorID, Out, EOF);

变量

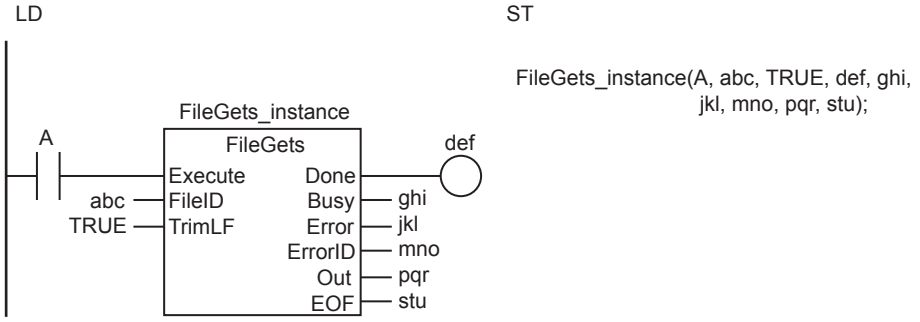
	名称	输入/输出	内容	有效范围	单位	初始值
FileID	文件ID	输入	待读取文件的ID	遵从数据类型	–	0
TrimLF	换行代码删除指定		待读取字符串的换行代码删除指定 TRUE ：删除 FALSE：不删除			FALSE
Out	读取字符串	输出	读取的字符串	遵从数据类型	–	–
EOF	文件结尾		判定是否到达文件结尾 TRUE ：到达 FALSE：未到达			

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileID				○																
TrimLF	○																			
Out																				○
EOF	○																			

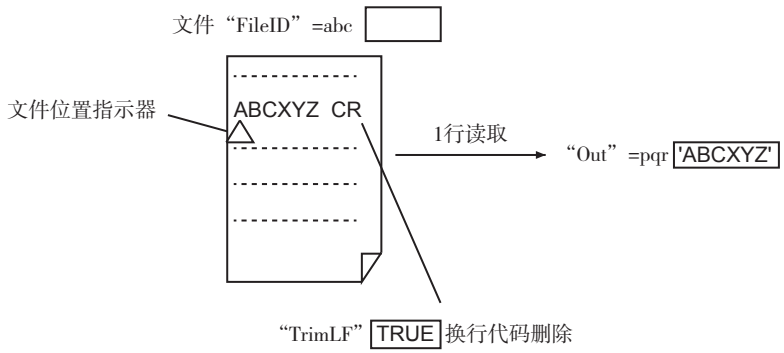
功能

从SD存储卡内文件ID “FileID”指定的文件、文件位置指示器的某个位置起读取1行字符串。
事先通过FileSeek指令，将文件位置指示器设定至任意位置。
通过换行代码识别行与行的间隔。将读取的字符串写入读取字符串 “Out”。
从CR、LF、CR+LF的3种类型中自动判别换行代码。换行代码删除指定 “TrimLF”的值变为TRUE时，删除字符串中的换行代码，然后写入 “Out”。
此外，读取至文件结尾时，文件结尾 “EOF”的值变为TRUE。否则，“EOF”的值变为FALSE。

示例如下所示。从文件中读取1行字符串，然后删除换行代码，并写入pqr。



从SD存储卡内“FileID”指定的文件、文件位置指示器的某个位置读取1行字符串，
并保存至读取字符串“Out”。然后，删除换行代码。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

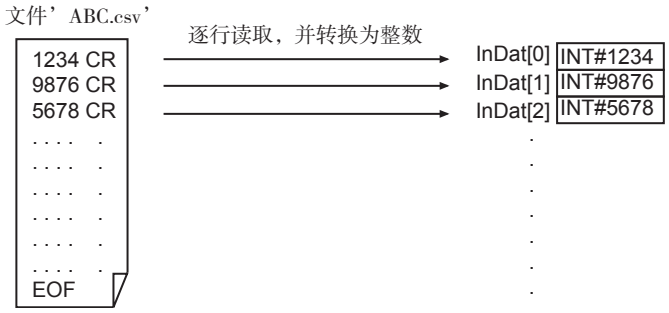
*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 1行字符串的长度超出1986字节(字符代码为UTF-8，含终端的NULL字符)时，将终端的NULL字符添加至字符串(至第1985字节)，并保存至 “Out”。
- 执行本指令前，必须事先执行FileOpen指令，并获取 “FileID”。
- 在执行本指令过程中将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - “FileID”指定的文件不存在时。
  - “FileID”指定的文件正在访问时。
  - 未通过可读取的模式打开 “FileID”指定的文件时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

示例程序

在文件名为 ‘ABC.csv’的文件中多行保存由换行代码CR分隔的字符串。它们均为数字的字符串。
逐行读取该文件，然后将字符串转换为整数，并保存至INT型数组变量InDat[ ]。
读取文件至最后(读取直至文件结尾EOF)时，结束处理。



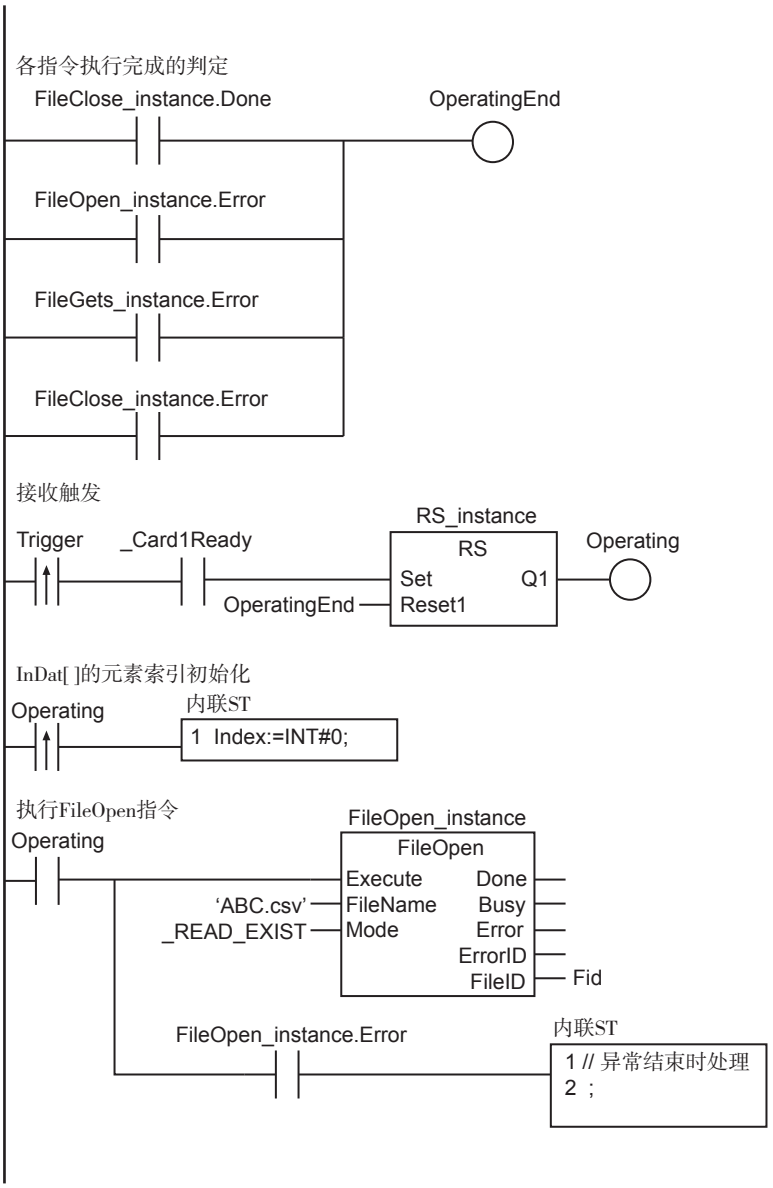
处理步骤如下所示。

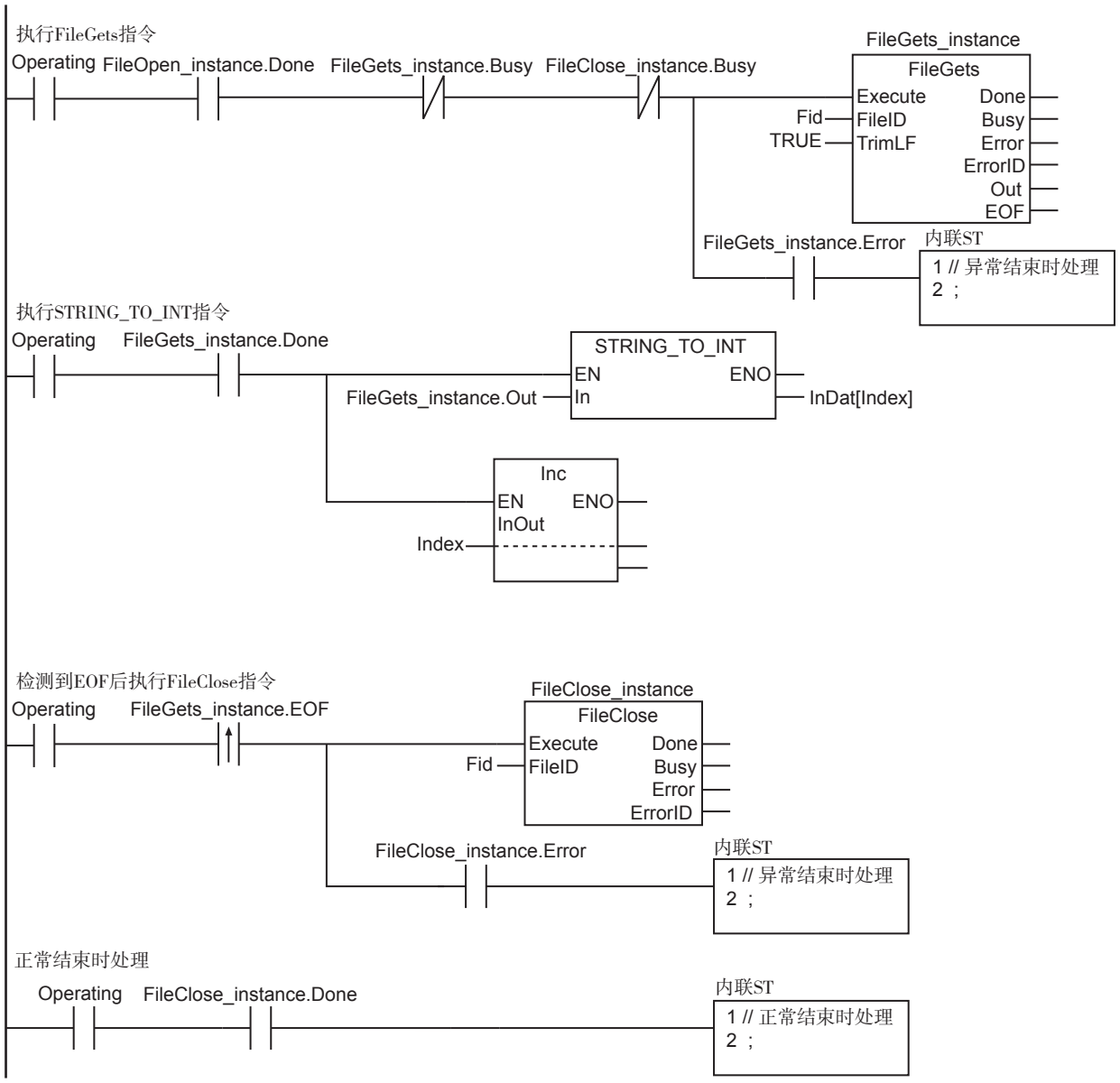
1. 使用FileOpen指令，打开'ABC.csv'文件。
2. 使用FileGets指令，读取1行文件内容。
3. 使用STRING_TO_INT指令，将读取的字符串转换为整数，并保存至InDat[ ]。
4. 继续执行上述2 ~ 3，直至读取文件结尾EOF。
5. 使用FileClose指令，关闭文件。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	Index	INT	0	InDat[ ]元素索引
	Fid	DWORD	16#0	文件ID
	InDat	ARRAY[0..999] OF INT	[1000(0)]	整数数据
	RS_instance	RS		
	FileOpen_instance	FileOpen		
	FileGets_instance	FileGets		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志




执行FileGets指令
Operating
FileOpen_instance.Done
FileGets_instance.Busy
FileClose_instance.Busy
FileGets_instance
FileGets
Execute
Done
Fid
FileID
Busy
TRUE
TrimLF
Error
ErrorID
Out
EOF
FileGets_instance.Error
内联ST
1 // 异常结束时处理
2 ;
执行STRING_TO_INT指令
Operating
FileGets_instance.Done
STRING_TO_INT
EN
ENO
FileGets_instance.Out
In
InDat[Index]
Inc
EN
ENO
InOut
Index
检测到EOF后执行FileClose指令
Operating
FileGets_instance.EOF
FileClose_instance
FileClose
Execute
Done
Fid
FileID
Busy
Error
ErrorID
FileClose_instance.Error
内联ST
1 // 异常结束时处理
2 ;
正常结束时处理
Operating
FileClose_instance.Done
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	InDat	ARRAY[0..999] OF INT	[1000(0)]	整数数据
	Stage	INT	0	状态变化
	Index	INT	0	InDat[ ]元素索引
	Fid	DWORD	16#0	文件ID
	FileOpen_instance	FileOpen		
	FileGets_instance	FileGets		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileOpen_instance(Execute:=FALSE);
    FileGets_instance(Execute:=FALSE);
    FileClose_instance(Execute:=FALSE);
    Stage          :=INT#1;
    Index          :=INT#0;
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                         // 文件打开
        FileOpen_instance(
            Execute   :=TRUE,
            FileName := 'ABC.csv',          // 文件名
            Mode      :=_READ_EXIST,        // 文件读取
            FileID    =>Fid);               // 文件ID

        IF (FileOpen_instance.Done=TRUE) THEN
            Stage:=INT#2;                   // 正常结束
        END_IF;

        IF (FileOpen_instance.Error=TRUE) THEN
            Stage:=INT#99;                  // 异常结束
        END_IF;
```

```
    2 :                         // 字符串读取
        FileGets_instance(
            Execute :=TRUE,
            FileID   :=Fid,
            TrimLF  :=TRUE);

        IF (FileGets_instance.Done=TRUE) THEN
            // 将读取的字符串转换为整数
            InDat[Index]:=STRING_TO_INT(FileGets_instance.Out);
            Index:=Index+INT#1;

            // 到达文件结尾
            IF (FileGets_instance.EOF=TRUE) THEN
                Stage:=INT#3;       // 正常结束
            ELSE
                FileGets_instance(Execute:=FALSE);
            END_IF;
        END_IF;

        IF (FileGets_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    3 :                         // 文件关闭
        FileClose_instance(
            Execute :=TRUE,
            FileID   :=Fid);        // 文件ID

        IF (FileClose_instance.Done=TRUE) THEN
            Operating:=FALSE;     // 正常结束
        END_IF;

        IF (FileClose_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    99 :                        // 异常结束处理
        Operating:=FALSE;
    END_CASE;
END_IF;
```

FilePuts

将字符串写入SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FilePuts	字符串写入	FB	FilePuts_instance FilePuts Execute Done FileID Busy In Error ErrorID	FilePuts_instance(Execute, FileID, In, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileID	文件ID	输入	待写入文件的ID	遵从数据类型	-	0
In	写入字符串		待写入的字符串			''

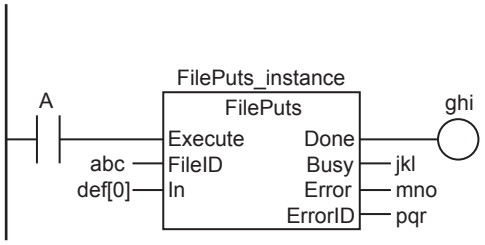
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileID				○																
In																				○

功能

将字符串写入SD存储卡内文件ID “FileID”指定的文件、文件位置指示器的某个位置。
事先通过FileSeek指令，将文件位置指示器设定至任意位置。
待写入的字符串为写入字符串 “In”的内容。

示例如下所示。将数组的1个元素def[0]的内容写入文件。

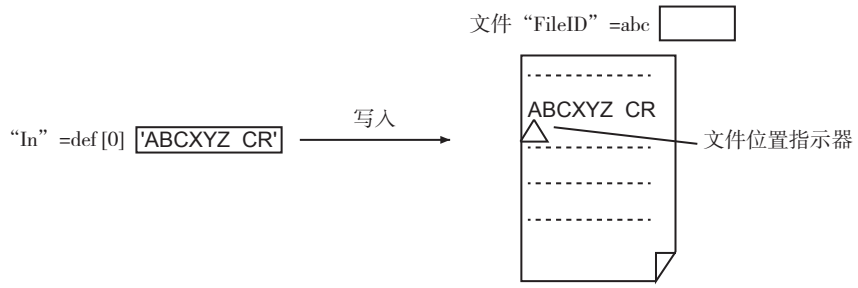
LD



ST

```
FilePuts_instance(A, abc, def[0], ghi, jkl, mno, pqr);
```

将写入字符串“In”的内容写入SD存储卡内“FileID”指定的文件、
文件位置指示器的某个位置。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

希望在写入字符串后换行时，请在 “In”的末尾附加换行代码。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 执行本指令前，必须事先执行FileOpen指令，并获取 “FileID”。
- 在执行本指令过程中将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - SD存储卡空间不足时。
  - “FileID”指定的文件不存在时。
  - “FileID”指定的文件正在访问时。
  - 未通过可写入的模式打开 “FileID”指定的文件时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

示例程序

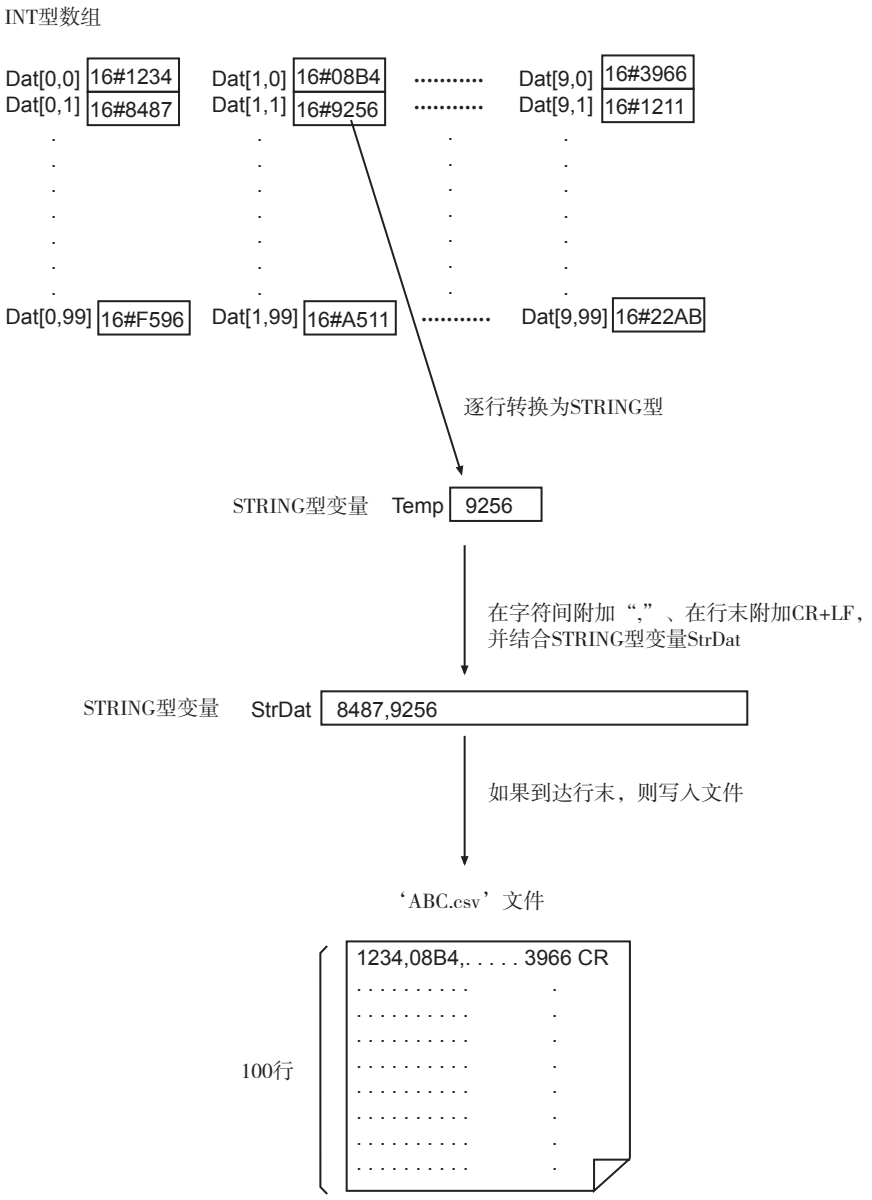
以100行CSV文件格式，将INT型数组变量Dat[0..9,0..99]的内容保存至文件 ‘ABC.csv’。
数值的字符串在1行中排成10个，并在各字符串之间插入 “,”。行末添加换行代码CR + LF。
步骤如下所示。

1 将Dat[ ]的元素转换为1个字符串，并保存至STRING型的变量Temp。

2 如果不是行末，则在Temp之后附加“,”；如果是行末，则在Temp之后附加CR+LF，并与STRING型变量StrDat相结合。

3 如果到达行末，则将StrDat写入文件。

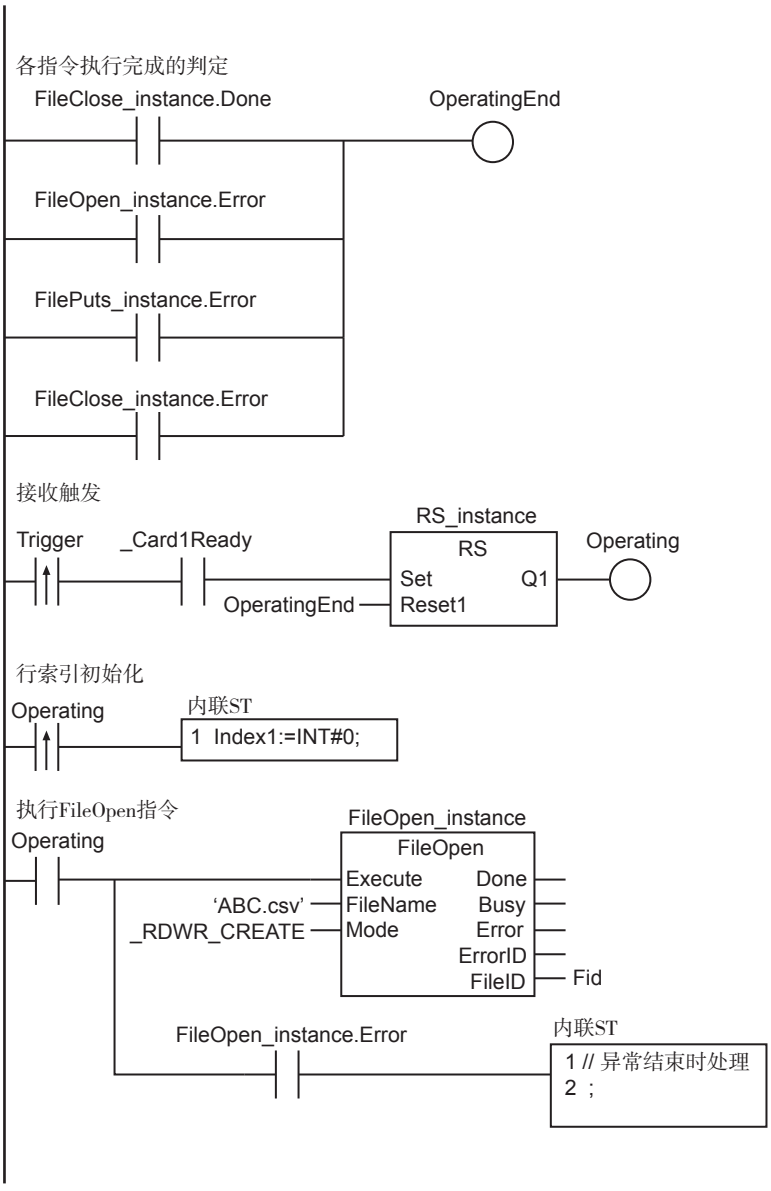
4 以100行为单位，重复执行1 ~ 3。

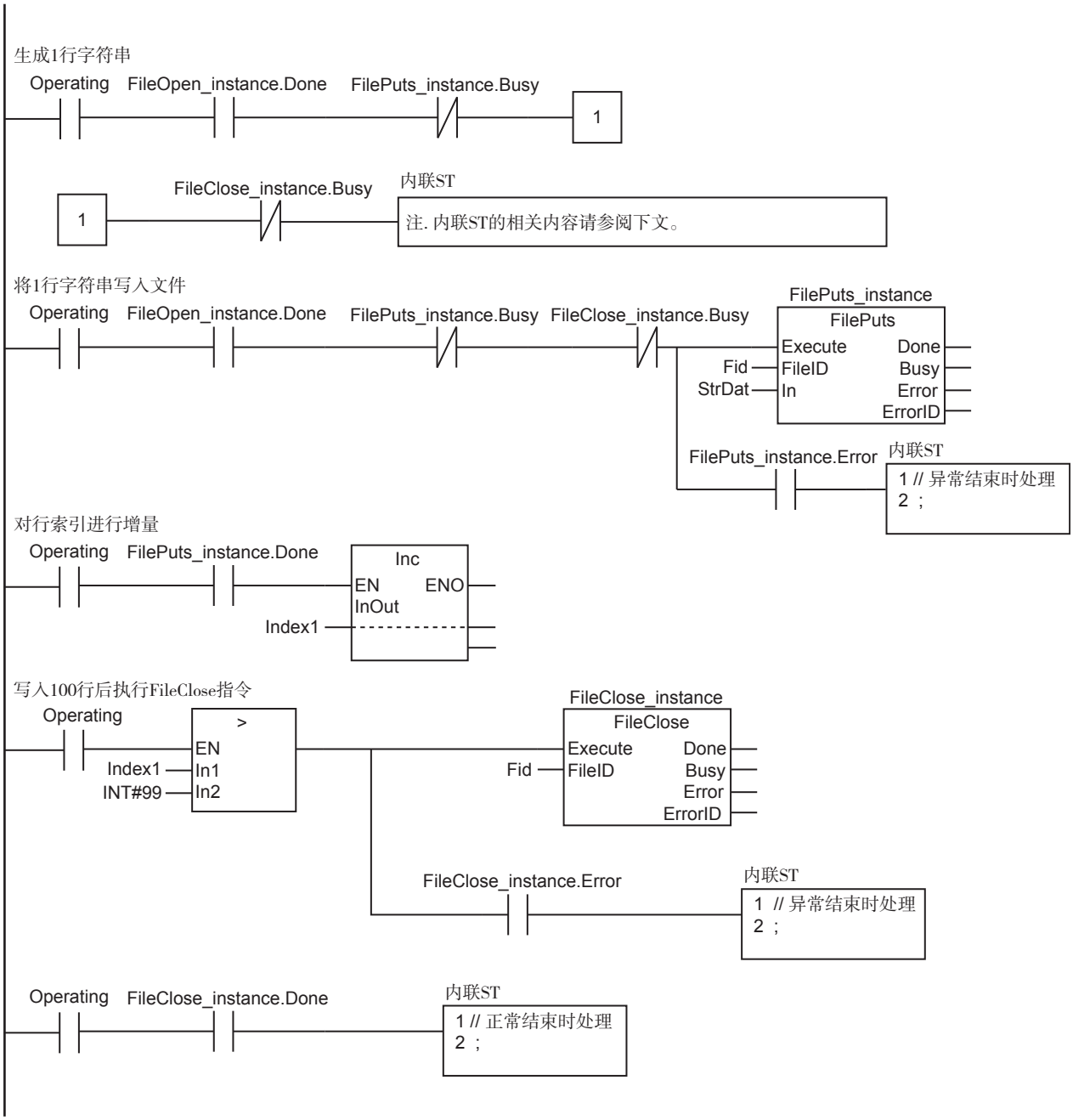

INT型数组
Dat[0,0] 16#1234
Dat[0,1] 16#8487
Dat[1,0] 16#08B4
Dat[1,1] 16#9256
Dat[9,0] 16#3966
Dat[9,1] 16#1211
Dat[0,99] 16#F596
Dat[1,99] 16#A511
Dat[9,99] 16#22AB
逐行转换为STRING型
STRING型变量 Temp 9256
在字符间附加“,”、在行末附加CR+LF，
并结合STRING型变量StrDat
STRING型变量 StrDat 8487,9256
如果到达行末，则写入文件
‘ABC.csv’文件
1234,08B4,. . . . . 3966 CR
100行

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	Index0	INT	0	列索引
	Index1	INT	0	行索引
	Fid	DWORD	16#0	文件ID
	StrDat	STRING[255]	''	字符串数据
	Dat	ARRAY[0..99,0..9] OF INT	[1000(0)]	数值数据
	Temp	STRING[255]	''	临时
	RS_instance	RS		
	FileOpen_instance	FileOpen		
	FilePuts_instance	FilePuts		
	FileClose_instance	FileClose		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志





● 内联ST的内容

```
StrDat:= '';

// 连接第0 ~ 8个字符串

FOR Index0 :=INT#0 TO INT#8 BY INT#1 DO
    Temp  :=INT_TO_STRING(Dat[Index1, Index0]);
    Temp  :=CONCAT(In1:=Temp, In2:= ',');
    StrDat :=CONCAT(IN1:=StrDat, In2:=Temp);
END_FOR;

// 连接第9个字符串，并附加CR+LF
Temp  :=INT_TO_STRING(Dat[Index1, Index0]);
Temp  :=CONCAT(In1:=Temp, In2:= '$r$l');
StrDat :=CONCAT(In1:=StrDat, In2:=Temp);
```

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	Index0	INT	0	列索引
	Index1	INT	0	行索引
	Fid	DWORD	16#0	文件ID
	StrDat	STRING[255]	''	字符串数据
	Dat	ARRAY[0..99,0..9] OF INT	[1000(0)]	数值数据
	Temp	STRING[255]	''	临时
	FileOpen_instance	FileOpen		
	FilePuts_instance	FilePuts		
	FileClose_instance	FileClose		

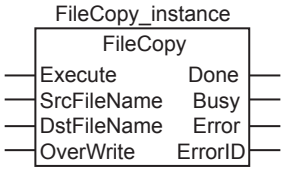
外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileOpen_instance(Execute:=FALSE);
    FilePuts_instance(Execute:=FALSE);
    FileClose_instance(Execute:=FALSE);
    Stage          :=INT#1;
    Index1         :=INT#0;          // 行索引初始化
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                              // 文件打开
        FileOpen_instance(
            Execute  :=TRUE,
            FileName := 'ABC.csv',   // 文件名
            Mode     :=_RDWR_CREATE, // 文件读取
            FileID   =>Fid);         // 文件ID

        IF (FileOpen_instance.Done=TRUE) THEN
            Stage:=INT#2;            // 正常结束
        END_IF;

        IF (FileOpen_instance.Error=TRUE) THEN
            Stage:=INT#99;           // 异常结束
        END_IF;
```

```
    2 :                         // 创建1行字符串
        StrDat:= '' ;

        // 连接第0 ~ 8个字符串
        FOR Index0 :=INT#0 TO INT#8 BY INT#1 DO
            Temp  :=INT_TO_STRING(Dat[Index1, Index0]);
            Temp  :=CONCAT(In1:=Temp, In2:= ',' );
            StrDat :=CONCAT(IN1:=StrDat, In2:=Temp);
        END_FOR;

        // 连接第9个字符串，并附加CR+LF
        Temp  :=INT_TO_STRING(Dat[Index1, Index0]);
        Temp  :=CONCAT(In1:=Temp, In2:= '$r$l' );
        StrDat :=CONCAT(In1:=StrDat, In2:=Temp);

        Stage:=INT#3;

    3 :                         // 字符串写入
        FilePuts_instance(
            Execute :=TRUE,
            FileID   :=Fid,
            In       :=StrDat);

        IF (FilePuts_instance.Done=TRUE) THEN
            Index1:=Index1+INT#1;

            IF (Index1>INT#99) THEN   // 写入100行
                Stage:=INT#4;
            ELSE
                FilePuts_instance(Execute:=FALSE);
                Stage:=INT#2;
            END_IF;
        END_IF;

        IF (FilePuts_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    4 :                         // 文件关闭
        FileClose_instance(
            Execute :=TRUE,
            FileID   :=Fid);                    // 文件ID

        IF (FileClose_instance.Done=TRUE) THEN
            Operating:=FALSE;     // 正常结束
        END_IF;

        IF (FileClose_instance.Error=TRUE) THEN
            Stage:=INT#99;          // 异常结束
        END_IF;

    99 :                        // 异常结束处理
        Operating:=FALSE;
    END_CASE;
END_IF;
```

FileCopy

复制SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FileCopy	文件复制	FB		FileCopy_instance(Execute, SrcFileName, DstFileName, OverWrite, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
SrcFileName	复制源文件名	输入	复制源文件名	最大66字节(65个半角英数字字符+结尾NULL字符)	–	''
DstFileName	复制对象文件名		复制对象文件名			
OverWrite	允许覆盖		TRUE：允许覆盖 FALSE：禁止覆盖	遵从数据类型		FALSE

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
SrcFileName																				○
DstFileName																				○
OverWrite	○																			

功能

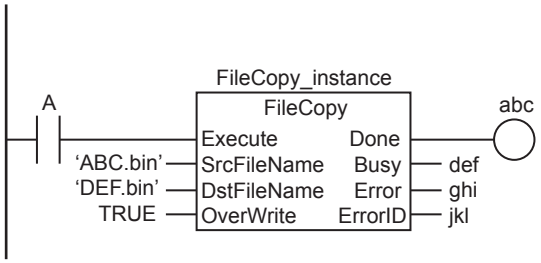
将SD存储卡内复制源文件名 “SrcFileName”指定的文件复制至复制对象文件名 “DstFileName”。

SD存储卡内已存在与 “DstFileName”同名的文件时，根据允许覆盖 “OverWrite”的值，执行下述处理。

“OverWrite”的值	处理
TRUE(允许覆盖)	覆盖至该文件。
FALSE(禁止覆盖)	不覆盖至该文件时会发生异常。

示例如下所示。将文件 ‘ABC.bin’覆盖至文件 ‘DEF.bin’。

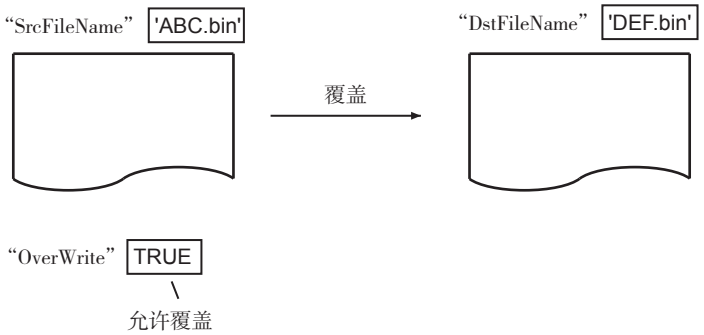
LD



ST

```
FileCopy_instance(A, 'ABC.bin', 'DEF.bin',
                  TRUE, abc, def, ghi, jkl);
```

将SD存储卡内复制源文件名“SrcFileName”指定的文件覆盖至复制对象文件名“DstFileName”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 复制失败时，“DstFileName”指定的文件可能以不完整状态残留于SD存储卡内。
- 在文件打开的状态下将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX系列CPU单元在打开文件的状态下通过供电停止开关操作停止供电时，文件不会损坏。
- NJ/NX 系列 CPU 单元在打开文件的状态下不操作供电停止开关即拔下 SD 存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡后，即使重新安装SD存储卡，也无法读写文件。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - SD存储卡空间不足时。
  - “SrcFileName”指定的文件不存在时。
  - “SrcFileName”的值并非正确的文件名时。
  - “DstFileName”的值并非正确的文件名时。
  - 超出可创建的文件数量、目录数量时。
  - “SrcFileName”或 “DstFileName”指定的文件正在访问时。
  - 已存在与 “DstFileName”同名的文件且 “OverWrite”的值为FALSE时。
  - 已存在与 “DstFileName”同名的文件且该文件禁止写入时。
  - 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
  - “DstFileName”的值超出了可作为文件名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

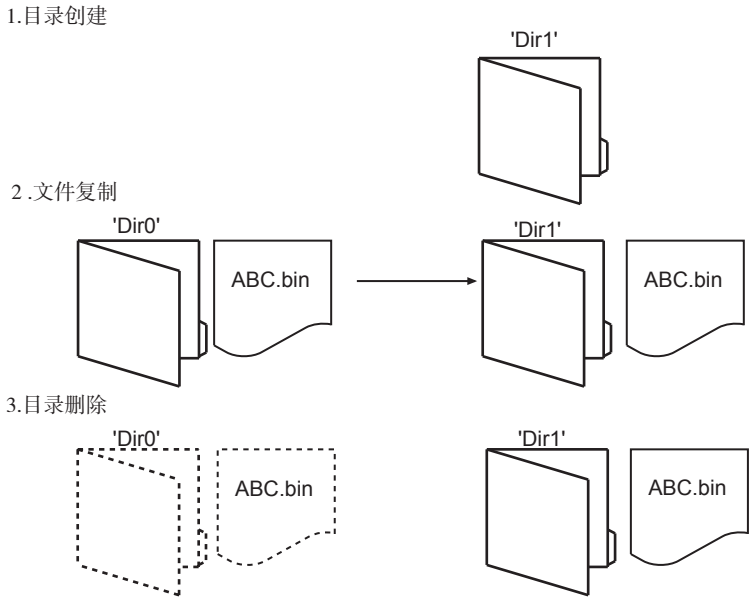
示例程序

按照下述步骤，执行文件的移动。

1 使用DirCreate指令，在SD存储卡内新建目录'Dir1'。

2 使用FileCopy指令，将现有目录'Dir0'内的文件'ABC.bin'复制至目录'Dir1'。

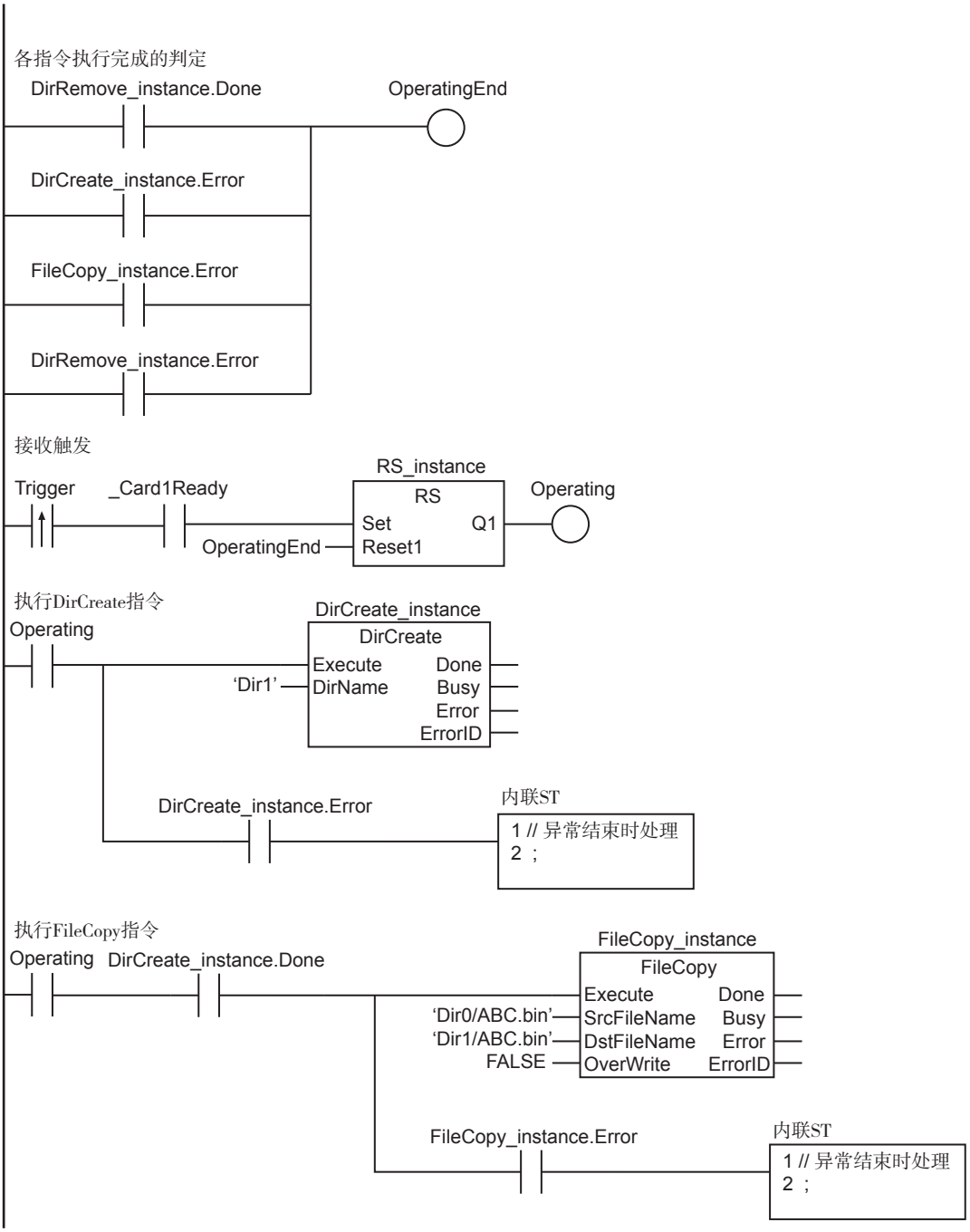
3 使用DirRemove指令，删除复制源目录'Dir0'。

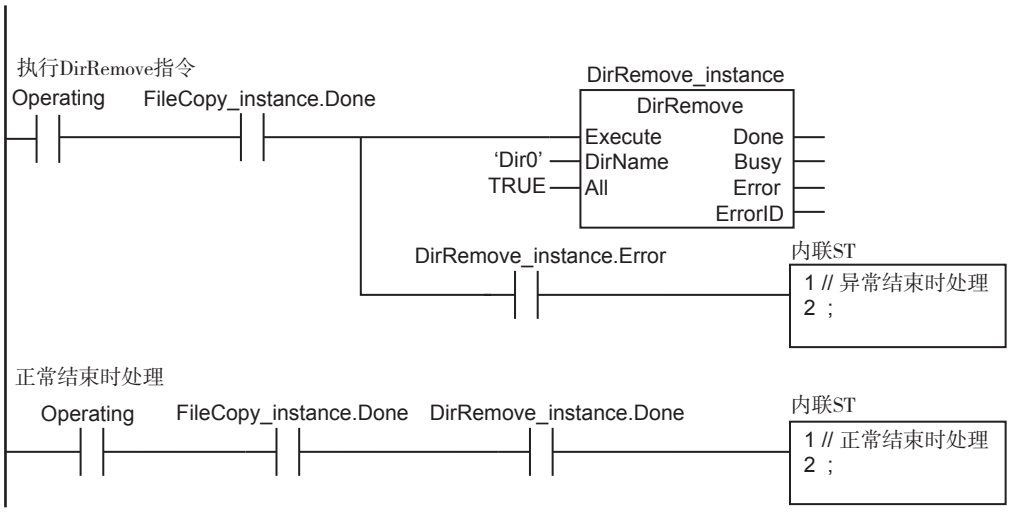


LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	RS_instance	RS		
	DirCreate_instance	DirCreate		
	FileCopy_instance	FileCopy		
	DirRemove_instance	DirRemove		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志




执行DirRemove指令
Operating
FileCopy_instance.Done
DirRemove_instance
DirRemove
Execute
Done
'Dir0'
DirName
Busy
TRUE
All
Error
ErrorID
DirRemove_instance.Error
内联ST
1 // 异常结束时处理
2 ;
正常结束时处理
Operating
FileCopy_instance.Done
DirRemove_instance.Done
内联ST
1 // 正常结束时处理
2 ;

ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	Stage	INT	0	状态变化
	DirCreate_instance	DirCreate		
	FileCopy_instance	FileCopy		
	DirRemove_instance	DirRemove		

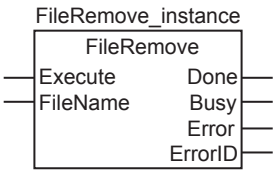
外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    DirCreate_instance(Execute:=FALSE);
    FileCopy_instance(Execute:=FALSE);
    DirRemove_instance(Execute:=FALSE);
    Stage          :=INT#1;
    OperatingStart :=FALSE;
END_IF;

// 各指令执行
IF (Operating=TRUE) THEN
    CASE Stage OF
    1 :                                 // 目录创建
        DirCreate_instance(
            Execute  :=TRUE,
            DirName := 'Dir1');         // 目录名

        IF (DirCreate_instance.Done=TRUE) THEN
            Stage:=INT#2;               // 正常结束
        END_IF;

        IF (DirCreate_instance.Error=TRUE) THEN
            Stage:=INT#99;              // 异常结束
        END_IF;
```

```
    2 :                                 // 文件复制
        FileCopy_instance(
            Execute     :=TRUE,
            SrcFileName := ‘Dir0/ABC.bin’,// 复制源文件名
            DstFileName := ‘Dir1/ABC.bin’,// 复制对象文件名
            OverWrite   :=FALSE);           // 禁止覆盖

        IF (FileCopy_instance.Done=TRUE) THEN
            Stage:=INT#3;
        END_IF;

        IF (FileCopy_instance.Error=TRUE) THEN
            Stage:=INT#99;
        END_IF;

    3 :                                 // 目录删除
        DirRemove_instance(
            Execute  :=TRUE,
            DirName := ‘Dir0’,          // 目录名
            All      :=TRUE);           // 连同文件及子目录一起削除

        IF (DirRemove_instance.Done=TRUE) THEN
            Operating:=FALSE;           // 正常结束
        END_IF;

        IF (DirRemove_instance.Error=TRUE) THEN
            Stage:=INT#99;              // 异常结束
        END_IF;

    99 :                                // 异常结束处理
        Operating:=FALSE;
    END_CASE;
END_IF;
```

FileRemove

删除SD存储卡内的指定文件。

指令	名称	FB/FUN	图形表现	ST表现
FileRemove	文件删除	FB		FileRemove_instance(Execute, FileName, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileName	指定文件名	输入	待删除的文件名	最大66字节(65个半角英数字字符+结尾NULL字符)	-	''

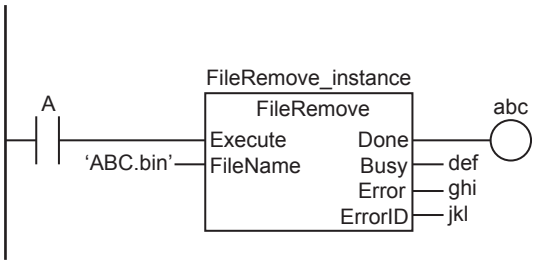
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileName																				○

功能

删除SD存储卡内指定文件名 “FileName”指定的文件。

示例如下所示。删除文件 ‘ABC.bin’。

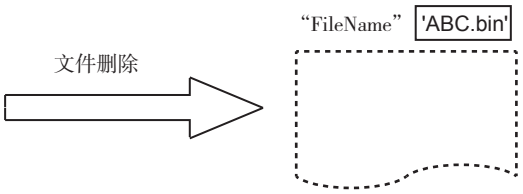
LD



ST

```
FileRemove_instance(A, ‘ABC.bin’, abc,
                    def, ghi, jkl);
```

删除SD存储卡内“FileName”指定的文件。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- 在文件打开的状态下将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX系列CPU单元在打开文件的状态下通过供电停止开关操作停止供电时，文件不会损坏。
- NJ/NX 系列 CPU 单元在打开文件的状态下不操作供电停止开关即拔下 SD 存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡后，即使重新安装SD存储卡，也无法读写文件。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - “FileName”指定的文件不存在时。
  - “FileName”指定的文件正在访问时。
  - 已存在与 “FileName”同名的文件且该文件禁止写入时。
  - 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
  - “FileName”的值超出了可作为文件名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

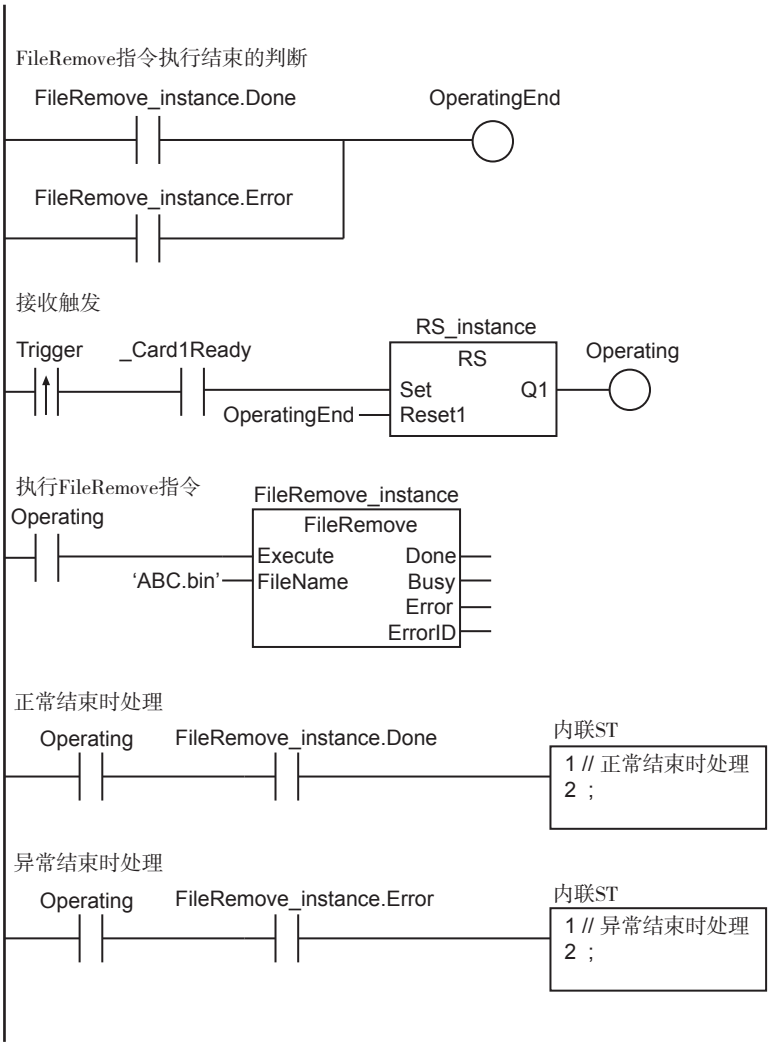
示例程序

删除SD存储卡内的文件 ‘ABC.bin’。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	RS_instance	RS		
	FileRemove_instance	FileRemove		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志



ST

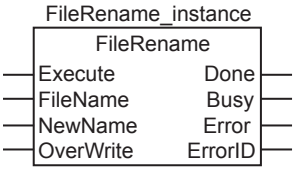
内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	FileRemove_instance	FileRemove		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileRemove_instance(Execute:=FALSE);
    OperatingStart:=FALSE;
END_IF;

// FileRemove指令执行
IF (Operating=TRUE) THEN
    FileRemove_instance(
        Execute   :=TRUE,
        FileName := 'ABC.bin');     // 文件名

    IF (FileRemove_instance.Done=TRUE) THEN
        Operating:=FALSE;               // 正常结束
    END_IF;

    IF (FileRemove_instance.Error=TRUE) THEN
        Operating:=FALSE;               // 异常结束
    END_IF;
END_IF;
```

FileRename

变更SD存储卡内指定文件和目录的名称。

指令	名称	FB/FUN	图形表现	ST表现
FileRename	文件名变更	FB		FileRename_instance(Execute, FileName, NewName, OverWrite, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
FileName	原文件名	输入	原文件名	最大66字节(65个半角英数字字符+结尾NULL字符)	-	''
NewName	变更后的文件名		变更后的文件名			
OverWrite	允许覆盖		TRUE：允许覆盖 FALSE：禁止覆盖	遵从数据类型		FALSE

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
FileName																				○
NewName																				○
OverWrite	○																			

功能

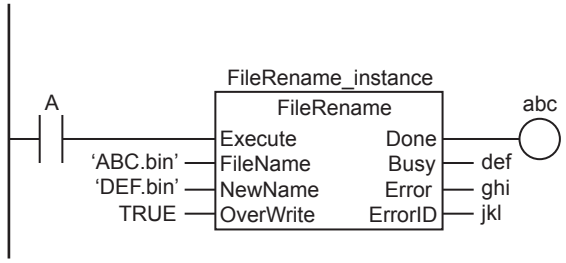
将SD存储卡内原文件名 “FileName”指定的文件及目录的名称变更为变更后的文件名 “NewName”。

SD存储卡内已存在与 “NewName”同名的文件及目录时，根据允许覆盖 “OverWrite”的值，执行下述处理。

“OverWrite”的值	处理
TRUE(允许覆盖)	覆盖至该文件及目录。
FALSE(禁止覆盖)	不覆盖至该文件及目录，从而导致异常。

示例如下所示。将文件 ‘ABC.bin’的名称变更为 ‘DEF.bin’。

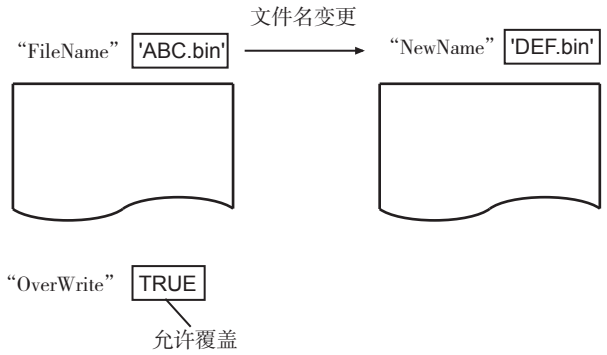
LD



ST

```
FileRename_instance(A, 'ABC.bin', 'DEF.bin',
                    TRUE, abc, def, ghi, jkl);
```

以覆盖的形式，将SD存储卡内原文件名“FileName”指定的文件名称变更为变更后的文件名“NewName”。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- “FileName”与 “NewName”的目录不同时，将文件移动至 “NewName”指定的目录。
- 在文件打开的状态下将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX系列CPU单元在打开文件的状态下通过供电停止开关操作停止供电时，文件不会损坏。
- NJ/NX 系列 CPU 单元在打开文件的状态下不操作供电停止开关即拔下 SD 存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡后，即使重新安装SD存储卡，也无法读写文件。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - “FileName”指定的文件目录不存在时。
  - “FileName”或 “NewName”的值并非正确的文件名及目录名时。
  - “FileName”指定的文件正在访问时。
  - 指定为 “FileName”的目录内存在子目录且 “OverWrite”的值为TRUE时。
  - 已存在与 “NewName”同名的文件且 “OverWrite”的值为FALSE时。
  - 已存在与 “NewName”同名的文件，该文件禁止写入且 “OverWrite”的值为TRUE时。
  - 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
  - “NewName”的值超出了可作为文件名及目录名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。
  - 超出可创建的目录数量时。

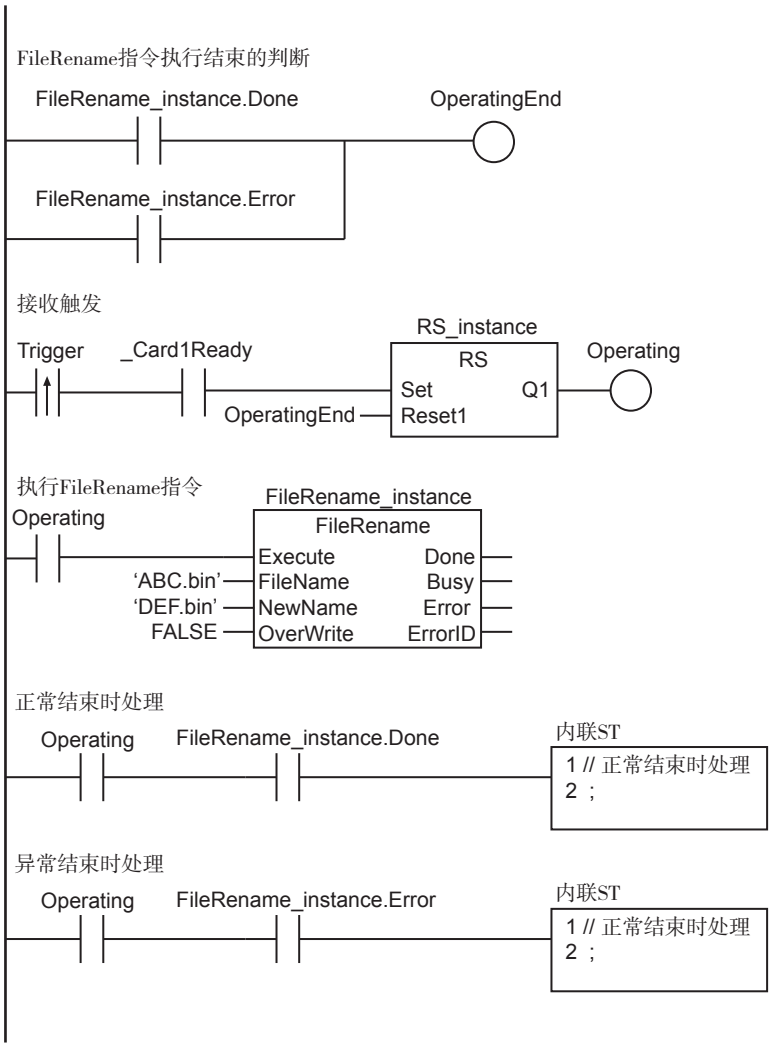
示例程序

将SD存储卡内文件 ‘ABC.bin’的名称变更为'DEF.bin'。

LD

内部变量	名称	数据类型	初始值	注释
	OperatingEnd	BOOL	FALSE	处理结束
	Trigger	BOOL	FALSE	执行条件
	Operating	BOOL	FALSE	处理中
	RS_instance	RS		
	FileRename_instance	FileRename		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志



ST

内部变量	名称	数据类型	初始值	注释
	Trigger	BOOL	FALSE	执行条件
	LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
	OperatingStart	BOOL	FALSE	处理开始
	Operating	BOOL	FALSE	处理中
	FileRename_instance	FileRename		

外部变量	名称	数据类型	注释
	_Card1Ready	BOOL	可使用SD存储卡标志

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) AND (_Card1Ready=TRUE) ) THEN
    OperatingStart :=TRUE;
    Operating      :=TRUE;
END_IF;
LastTrigger:=Trigger;

// 实例初始化
IF (OperatingStart=TRUE) THEN
    FileRename_instance(Execute:=FALSE);
    OperatingStart:=FALSE;
END_IF;

// FileRename指令执行
IF (Operating=TRUE) THEN
    FileRename_instance(
        Execute   :=TRUE,
        FileName  := ‘ABC.bin’,        // 原文件名
        NewName   := ‘DEF.bin’,        // 变更后的文件名
        OverWrite :=FALSE);            // 禁止覆盖

    IF (FileRename_instance.Done=TRUE) THEN
        Operating:=FALSE;              // 正常结束
    END_IF;

    IF (FileRename_instance.Error=TRUE) THEN
        Operating:=FALSE;              // 异常结束
    END_IF;
END_IF;
```

DirCreate

在SD存储卡内创建指定名称的目录。

指令	名称	FB/FUN	图形表现	ST表现
DirCreate	目录创建	FB	DirCreate_instance DirCreate Execute Done DirName Busy Error ErrorID	DirCreate_instance(Execute, DirName, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DirName	创建目录名	输入	待创建的目录名	最大66字节(65个半角英数字字符+结尾NULL字符)	-	''

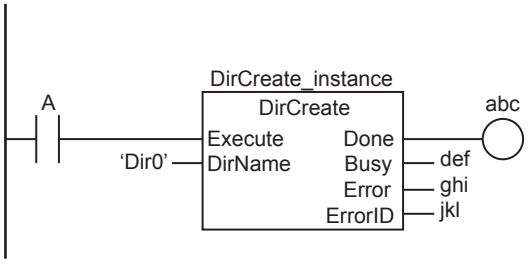
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DirName																				○

功能

在SD存储卡内创建名称由创建目录名 “DirName”指定的目录。

示例如下所示。创建目录 ‘Dir0’。

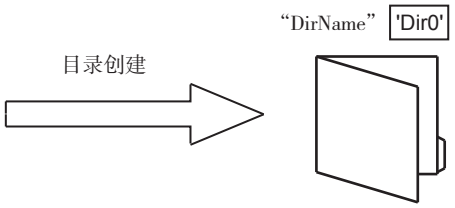
LD



ST

```
DirCreate_instance(A, 'Dir0', abc,
                   def, ghi, jkl);
```

在SD存储卡内创建名称由“DirName”指定的目录



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

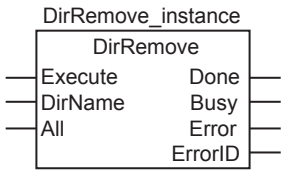
- 本指令一旦执行，即使 “Execute”的值为 FALSE 或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 在文件打开的状态下将 CPU 单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX系列CPU单元在打开文件的状态下通过供电停止开关操作停止供电时，文件不会损坏。
- NJ/NX 系列 CPU 单元在打开文件的状态下不操作供电停止开关即拔下 SD 存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡后，即使重新安装SD存储卡，也无法读写文件。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - SD存储卡空间不足时。
  - 超出可创建的目录数量时。
  - 已存在 “DirName”指定的目录时。
  - 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
  - “DirName”的值并非正确的目录名时。
  - “DirName”的值超出了可作为目录名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。
  - “FileName”指定的文件正在访问时。

示例程序

请参阅 “FileCopy指令(P.2-1346)”的示例程序。

DirRemove

删除SD存储卡内的指定目录。

指令	名称	FB/FUN	图形表现	ST表现
DirRemove	目录删除	FB		DirRemove_instance(Execute, DirName, All, Done, Busy, Error, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DirName	删除目录名	输入	待删除的目录名	最大66字节(65个半角英数字字符+结尾NULL字符)	–	''
All	所有指定		目录内存在文件/子目录时的指定 TRUE ：连同文件/子目录一起删除 FALSE：不删除	遵从数据类型		FALSE

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DirName																				○
All	○																			

功能

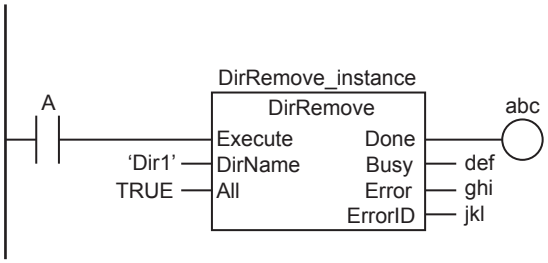
删除SD存储卡内删除目录名 “DirName”指定的目录。

指定目录内存在文件及子目录时，根据所有指定 “All”的值，执行下述处理。

“All”的值	处理
TRUE	连同文件及子目录一起，删除指定的目录。
FALSE	不删除指定的目录，从而导致异常。

示例如下所示。删除目录 ‘Dir1’。

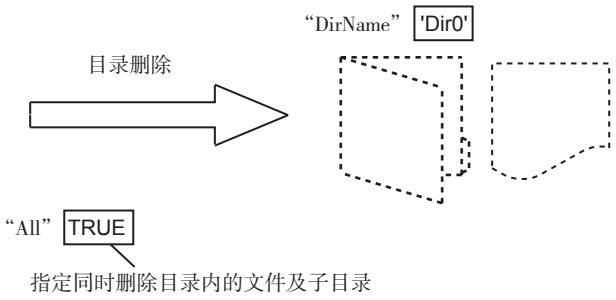
LD



ST

```
DirRemove_instance(A, ‘Dir1’, TRUE, abc,
                   def, ghi, jkl);
```

删除SD存储卡内名称由“DirName”指定的目录。
同时删除目录内的文件及子目录。



相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示以物理方式安装SD存储卡，安装处理正常结束后，是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：不可使用
_Card1Protect*2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err*2	SD存储卡异常标志	BOOL	表示安装规格外的SD存储卡(例：SDHC卡)时或格式是否异常(FAT16以外或文件系统损坏)的标志。 TRUE ：有异常 FALSE：无异常
_Card1Access*2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE ：访问中 FALSE：非访问中
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问SD存储卡*3的过程中有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

文件名的根目录指SD存储卡的正下方。

使用注意事项

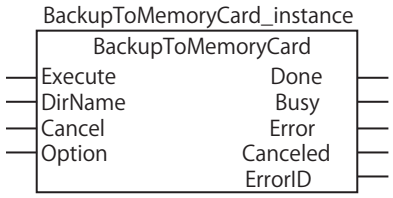
- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2-3)”。
- 在文件打开的状态下将CPU单元动作模式变更为程序或发生全部停止故障电平的控制器异常时，强制关闭该文件。此时，将处于执行过程中的数据读写操作进行到最后。
- NJ/NX系列CPU单元在打开文件的状态下通过供电停止开关操作停止供电时，文件不会损坏。
- NJ/NX系列CPU单元在打开文件的状态下不操作供电停止开关即拔下SD存储卡时，可能会损坏文件内容。拔下SD存储卡时，请务必停止供电。
- NJ/NX系列CPU单元在打开文件的状态下停止供电或拔下SD存储卡后，即使重新安装SD存储卡，也无法读写文件。
- “DirName”指定的目录禁止写入时会导致异常，不会删除该目录。但该目录内存在的非禁止写入文件、非禁止写入目录被删除。
- 请勿同时访问同一文件。请在用户程序内执行多个SD存储卡指令的排他性控制。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - “All”的值为TRUE且 “DirName”指定的目录正通过其它指令访问时。
  - “All”的值为FALSE且 “DirName”指定的目录内存在文件及目录时。
  - “DirName”指定的目录禁止写入时。
  - “DirName”指定的目录内存在禁止写入的文件、禁止写入的目录时。
  - 同时执行变量中不带 “FileID”的SD存储卡相关指令(FileWriteVar、FileReadVar、FileCopy、DirCreate、FileRemove、DirRemove、FileRename)5条以上时。
  - “DirName”指定的目录不存在时。
  - “DirName”的值超出了可作为目录名使用的字节数。
  - SD存储卡访问过程中发生某种异常导致无法访问时。

示例程序

请参阅 “FileCopy指令(P.2-1346)”的示例程序。

BackupToMemoryCard

执行SD存储卡备份。

指令	名称	FB/FUN	图形表现	ST表现
BackupTo MemoryCard	SD存储卡备份	FB		BackupToMemoryCard _instance(Execute, DirName, Cancel, Option, Done, Busy, Error, Canceled, ErrorID);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
DirName	保存目录名称	输入	保存备份数据的目录名称	最大64字节(63个半角英数字字符+结尾NULL字符)	–	''
Cancel	取消		备份的取消 TRUE ：执行取消 FALSE：不执行取消	遵从数据类型		FALSE
Option	将来扩展用		将来扩展用的变量 无需连接参数	–		–
Canceled	取消完成	输出	表示是否已取消完成的标志 TRUE ：取消完成 FALSE：取消失败	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
DirName																				○
Cancel	○																			
Option	将来扩展用的变量　无需连接参数																			
Canceled	○																			

功能

执行SD存储卡备份。
SD存储卡备份指令的处理内容与使用CPU单元正面开关、系统定义变量_Card1BkupCmd或Sysmac Studio的SD存储卡画面执行的SD存储卡备份相同。

保存备份数据的目录名称通过保存目录名称 “DirName”指定。
“DirName”的值为 ‘’(长度为0字符的字符串)时，备份数据将保存至SD存储卡的根目录。
“DirName”可省略。省略 “DirName”时的保存目录如下所述。

本指令的执行次数	保存目录
第1次	根目录
第2次以后	上一次指定的目录

SD存储卡内不存在 “DirName”指定的保存目录时，将在新建目录后保存备份数据。
“DirName”指定的保存目录中存在与备份文件名称相同的文件时，将被改写。

备份处理中，将取消 “Cancel”的值从FALSE变更为TRUE时，备份处理将被取消。备份处理被取消时，不会创建备份文件。“DirName”指定的保存目录中已存在备份文件时，该文件不会被改写，将继续保留。
唯有通过同一FB实例执行中的备份处理才可取消。
取消完成时，取消完成 “Canceled”的值将变为TRUE。根据将 “Cancel”的值设为TRUE的时间，可能会来不及受理取消，执行备份处理直至最后。来不及受理取消时，“Canceled”的值将变为FALSE，“Done”的值将变为TRUE。

“Cancel”的值为TRUE时，即使将 “Execute”的值设为TRUE，也不会执行备份处理。

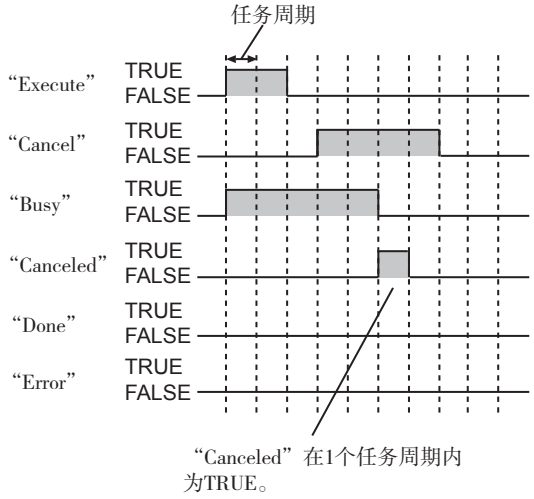
“Option”为将来扩展用的变量。请勿连接参数。

执行取消时的时序图

下面介绍执行了取消时各变量的时序图。

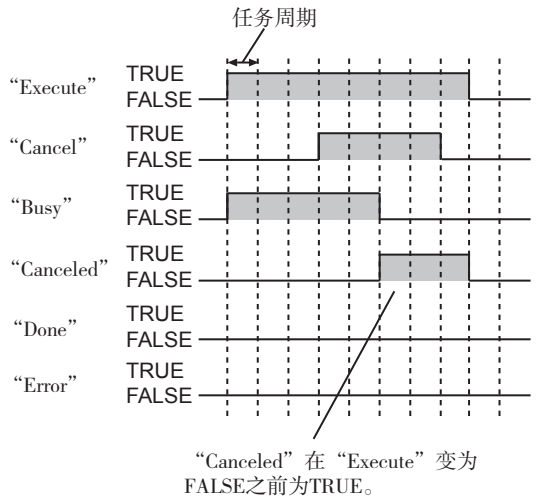
● 取消成功，在 “Canceled”变为TRUE之前将 “Execute”设为FALSE时

- 将 “Execute”的值设为TRUE时，将执行备份处理。“Busy”的值变为TRUE。
- 将 “Cancel”的值设为TRUE时，备份处理将被取消。
- 取消完成时，“Busy”的值将变为FALSE，“Canceled”的值将变为TRUE。
- “Canceled”的值变为TRUE之前，将 “Execute”的值设为FALSE。
- “Canceled”的值在1个任务周期后变为FALSE。
- 取消已完成，因此 “Done”的值将保持FALSE状态。



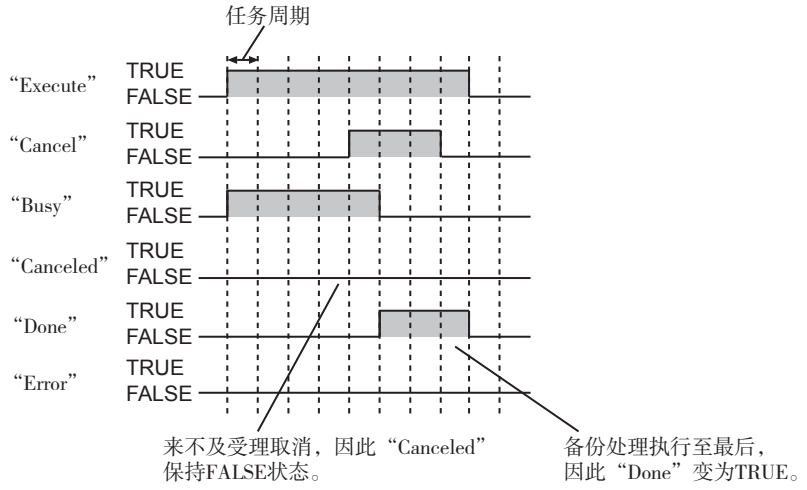
● 取消成功，在 “Canceled”变为TRUE之后将 “Execute”设为FALSE时

- 将 “Execute”的值设为TRUE时，将执行备份处理。“Busy”的值变为TRUE。
- 将 “Cancel”的值设为TRUE时，备份处理将被取消。
- 取消完成时，“Busy”的值将变为FALSE，“Canceled”的值将变为TRUE。
- “Canceled”的值变为TRUE之后，将 “Execute”的值设为FALSE。
- “Canceled”的值在 “Execute”的值变为FALSE之前为TRUE。
- 取消已完成，因此 “Done”的值将保持FALSE状态。

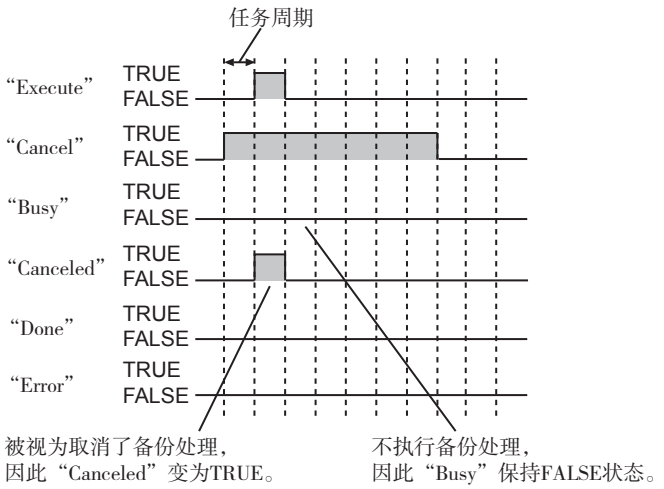


● 来不及受理取消时

- 将 “Execute”的值设为TRUE时，将执行备份处理。“Busy”的值变为TRUE。
- 将 “Cancel”的值设定为TRUE。但来不及受理取消，备份处理继续。
- 备份处理完成时，“Busy”的值变为FALSE。
- 备份处理执行至最后，因此 “Done”的值变为TRUE。
- 来不及受理取消，因此 “Canceled”的值保持FALSE状态。



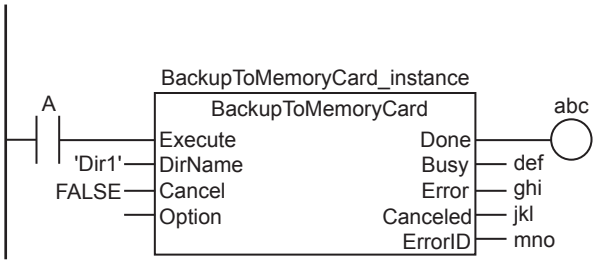
- **“Cancel”的值为TRUE，将 “Execute”的值设为TRUE时**
  - 将 “Cancel”的值设定为TRUE。
  - 即使将 “Execute”的值设为TRUE，也不会执行备份处理。因此 “Busy”的值保持FALSE状态。
  - 被视为取消了备份处理，因此 “Canceled”的值变为TRUE。
  - 将 “Execute”的值设为FALSE时，“Canceled”的值将变为FALSE。



记述示例

示例如下所示。备份文件保存至目录 ‘Dir1’中。

LD



ST

```
BackupToMemoryCard_instance(A, 'Dir1', FALSE,
                            , abc, def, ghi, jkl,
                            mno);
```

相关的系统定义变量

变量名称	名称	数据类型	内容
_Card1Ready	可使用SD存储卡标志	BOOL	表示是否可通过指令及通信指令进行访问的标志。*1 TRUE ：可使用 FALSE：无法使用
_Card1Protect *2	SD存储卡写保护标志	BOOL	表示安装SD存储卡后处于可使用状态且SD存储卡写保护是否启动的标志。 TRUE ：有写保护 FALSE：无写保护
_Card1Err *2	SD存储卡错误标志	BOOL	表示是否安装了规格外的SD存储卡(例：SDHC卡)或格式是否错误(FAT16以外或文件系统损坏)的标志。 TRUE ：有错误 FALSE：无错误
_Card1Access *2	SD存储卡访问中标志	BOOL	表示是否正在访问SD存储卡的标志。 TRUE: 访问中 FALSE: 非访问中
_Card1Deteriorated *2	SD存储卡寿命警告标志	BOOL	检测SD存储卡寿命的标志。 TRUE: 检测寿命 FALSE: 不检测
_Card1PowerFail	SD存储卡访问中断电标志	BOOL	表示访问过程中 *3有断电现象且执行过程中的处理异常结束的标志。本标志不会自动清除。 TRUE ：有异常 FALSE：无异常
_BackupBusy	备份相关功能执行中标志	BOOL	表示正在执行备份、恢复、核查的标志。 TRUE ：正在执行备份、恢复、核查 FALSE：未执行备份、恢复、核查

*1 NJ/NX系列以物理方式安装SD存储卡，且安装处理正常结束为前提条件。NY系列控制器以识别到共享文件夹为前提条件。
*2 NY系列控制器不使用。固定为FALSE。
*3 NJ/NX系列正在访问SD存储卡。NY系列控制器正在访问共享文件夹。

参考

- 备份功能的详情请参阅 🕮 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)” 或 🕮 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。
- 文件名的根目录指SD存储卡的正下方。

使用注意事项

- 本指令一旦执行，即使 “Execute”的值为FALSE或执行时间超过任务周期，仍将继续处理直至最后。请通过 “Done”的值是否变为TRUE确认处理是否正常结束。
- “Execute”、“Done”、“Busy”、“Error”的时序图请参阅 “本章说明(P.2–3)”。
- NJ/NX系列CPU单元在执行本指令的过程中，未操作SD存储卡供电停止按钮即拔下SD存储卡时，SD存储卡内的文件内容可能会损坏。拔下SD存储卡时，请务必停止供电。
- 即使设定了禁止向SD存储卡备份的情况下，执行本指令时也不会发生异常，仍将执行备份。
- 即使执行本指令，以下与备份相关的系统定义变量值仍不变。
  - SD存储卡备份指令 _CardBkupCmd
  - SD存储卡备份状态 _Card1BkupSta
- 执行本指令的过程中，请勿读写备份相关的文件。执行读取正在写入的文件等操作时，可能会进行意外处理。
- 执行本指令的过程中，即使变更控制器的动作模式，备份处理仍将继续。但，一旦按照运行模式→程序模式→运行模式进行变更时，即使在备份处理时 “Busy”的值也会变为 FALSE。此外，这种情况下也可执行取消，“Canceled”的值也会变为TRUE。
- 以下情况时会发生异常。“Error”变为TRUE。
  - SD存储卡并非可使用状态时。
  - SD存储卡处于写保护状态时。
  - SD存储卡空间不足时。
  - 超出可创建的文件数量、目录数量时。
  - 存在与 “DirName”指定的目录同名的文件。
  - “DirName”的值并非正确的目录名时。
  - SD存储卡访问过程中发生某种异常导致无法访问时。
  - 正在执行其他备份时。
  - 备份处理失败时。

版本相关信息

本指令可用于CPU单元 Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

每天，在日期改变时执行SD存储卡备份。备份相关的文件保存在SD存储卡内的目录/Backup/yyyy-mm-dd下。目录名称表示执行备份的年月日。yyyy表示年份，mm表示月份，dd表示日期。

触摸屏的规格

本程序以控制器连接触摸屏为前提。
触摸屏上有以下指示灯。

指示灯名称	含义
备份正常结束指示灯	备份处理正常结束时点亮。
备份取消指示灯	备份处理成功取消时点亮。
备份异常结束指示灯	备份处理异常结束时点亮。
SD存储卡寿命警告指示灯	检测到SD存储卡到达寿命时点亮。
SD存储卡断电指示灯	备份处理中，SD存储卡发生断电时点亮。

此外，触摸屏上有以下按钮。

按钮名称	按下时的动作
指示灯熄灭按钮	熄灭备份正常结束指示灯、备份取消指示灯、备份异常结束指示灯、SD存储卡断电指示灯。
取消按钮	取消执行中的备份。

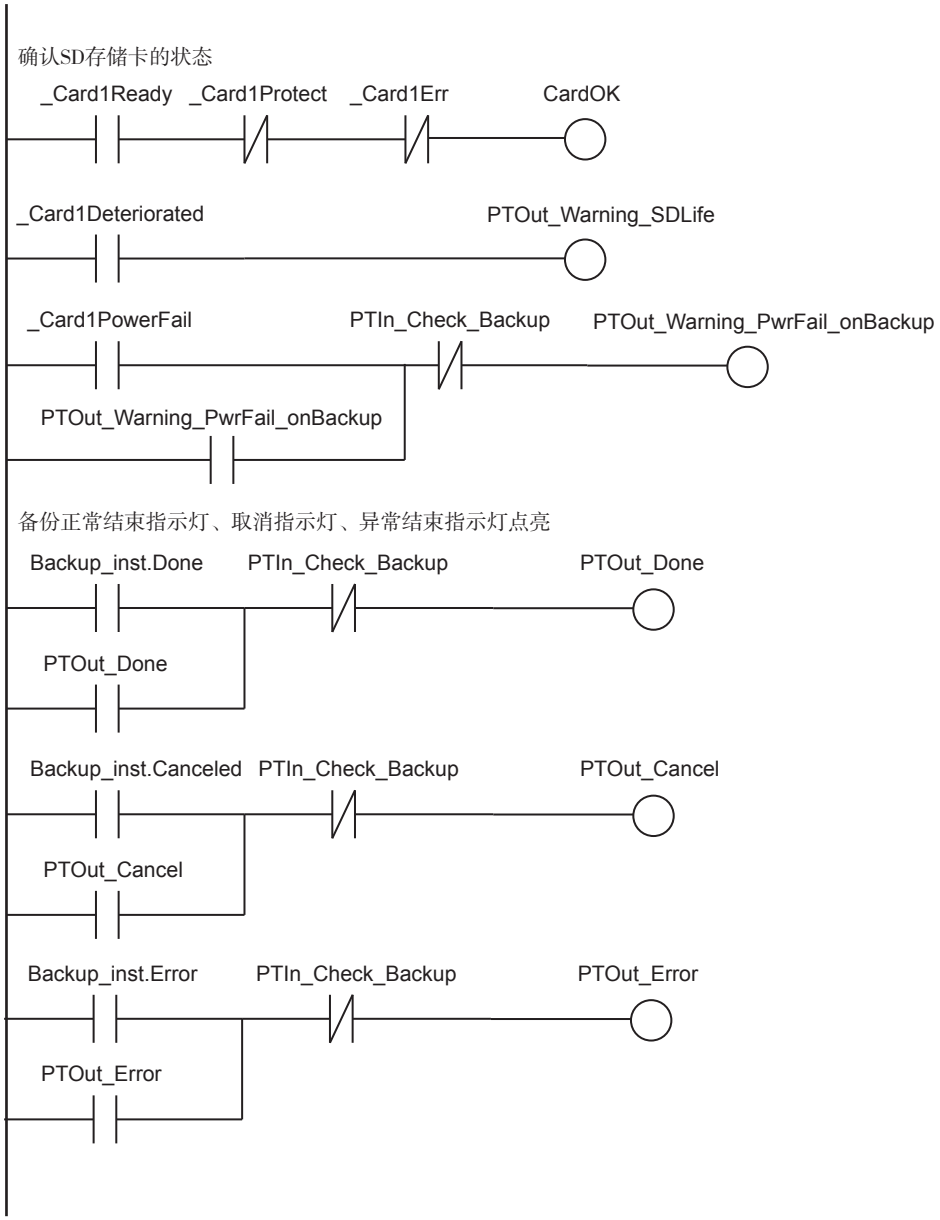
全局变量

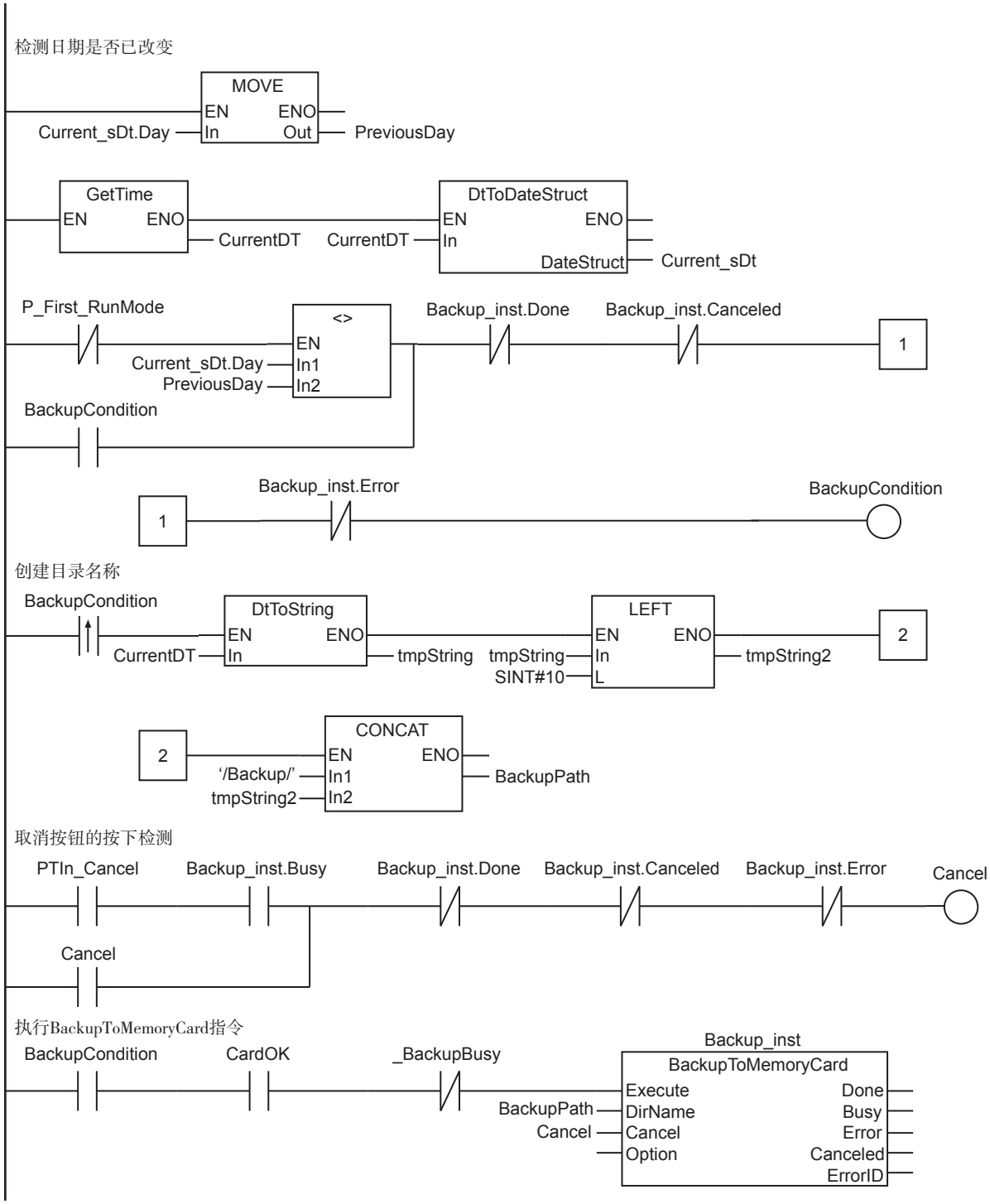
名称	数据类型	初始值	注释
PTOut_Warning_SDLife	BOOL	FALSE	SD存储卡寿命警告指示灯输出
PTOut_Warning_PwrFail_onBackup	BOOL	FALSE	SD存储卡断电指示灯输出
PTOut_Done	BOOL	FALSE	备份正常结束指示灯输出
PTOut_Cancel	BOOL	FALSE	备份取消指示灯输出
PTOut_Error	BOOL	FALSE	备份异常结束指示灯输出
PTIn_Check_Backup	BOOL	FALSE	指示灯熄灭按钮输入
PTIn_Cancel	BOOL	FALSE	取消按钮输入

LD

内部变量	名称	数据类型	初始值	注释
	CardOK	BOOL	FALSE	SD存储卡正常标志
	Backup_inst	BackupToMemory Card		BackupToMemoryCard指令的实例
	PreviousDay	USINT	0	上一次任务周期的日期
	CurrentDT	DATE_AND_TIME	ST#1970-01-01-00:00:00.000000000	当前的日期时刻
	Current_sDt	_sDT	(Year:=0, Month:=0, Day:=0, Hour:=0, Min:=0, Sec:=0, NSec:=0)	将当前的日期时刻分解为年、月、日、时、分、秒、纳秒
	BackupCondition	BOOL	FALSE	备份条件成立标志
	tmpString	STRING[256]	''	创建目录名称时的临时字符串
	tmpString2	STRING[256]	''	创建目录名称时的临时字符串
	BackupPath	STRING[64]	''	目录名称
	Cancel	BOOL	FALSE	取消条件成立标志

外部变量	名称	数据类型	常数	注释
	_Card1Ready	BOOL	☑	可使用SD存储卡标志
	_Card1Protect	BOOL	☑	SD存储卡写保护标志
	_Card1Err	BOOL	☑	SD存储卡错误标志
	_Card1Deteriorated	BOOL	☑	SD存储卡寿命警告标志
	_Card1PowerFail	BOOL	☐	SD存储卡访问中断电标志
	_BackupBusy	BOOL	☑	备份相关功能执行中标志
	PTOut_Warning_SDLife	BOOL	☐	SD存储卡寿命警告指示灯输出
	PTOut_Warning_PwrFail_on Backup	BOOL	☐	SD存储卡断电指示灯输出
	PTOut_Done	BOOL	☐	备份正常结束指示灯输出
	PTOut_Cancel	BOOL	☐	备份取消指示灯输出
	PTOut_Error	BOOL	☐	备份异常结束指示灯输出
	PTIn_Check_Backup	BOOL	☐	指示灯熄灭按钮输入
	PTIn_Cancel	BOOL	☐	取消按钮输入


确认SD存储卡的状态
_Card1Ready
_Card1Protect
_Card1Err
CardOK
_Card1Deteriorated
PTOut_Warning_SDLife
_Card1PowerFail
PTIn_Check_Backup
PTOut_Warning_PwrFail_onBackup
PTOut_Warning_PwrFail_onBackup
备份正常结束指示灯、取消指示灯、异常结束指示灯点亮
Backup_inst.Done
PTIn_Check_Backup
PTOut_Done
PTOut_Done
Backup_inst.Canceled
PTIn_Check_Backup
PTOut_Cancel
PTOut_Cancel
Backup_inst.Error
PTIn_Check_Backup
PTOut_Error
PTOut_Error


检测日期是否已改变
MOVE
EN
ENO
Current_sDt.Day
In
Out
PreviousDay
GetTime
EN
ENO
CurrentDT
CurrentDT
DtToDateStruct
EN
ENO
In
DateStruct
Current_sDt
P_First_RunMode
<>
EN
In1
In2
Current_sDt.Day
PreviousDay
Backup_inst.Done
Backup_inst.Canceled
1
BackupCondition
Backup_inst.Error
1
BackupCondition
创建目录名称
BackupCondition
DtToString
EN
ENO
In
CurrentDT
tmpString
LEFT
EN
ENO
In
L
tmpString
SINT#10
tmpString2
2
2
CONCAT
EN
ENO
In1
In2
'/Backup/'
tmpString2
BackupPath
取消按钮的按下检测
PTIn_Cancel
Backup_inst.Busy
Backup_inst.Done
Backup_inst.Canceled
Backup_inst.Error
Cancel
Cancel
执行BackupToMemoryCard指令
BackupCondition
CardOK
_BackupBusy
Backup_inst
BackupToMemoryCard
Execute
DirName
Cancel
Option
Done
Busy
Error
Canceled
ErrorID
BackupPath
Cancel

ST

内部变量	名称	数据类型	初始值	注释
	CardOK	BOOL	FALSE	SD存储卡正常标志
	Backup_inst	BackupToMemory Card		BackupToMemoryCard指令的实例
	PreviousDay	USINT	0	上一次任务周期的日期
	CurrentDT	DATE_AND_TIME	ST#1970-01-01-00:00:00.000000000	当前的日期时刻
	Current_sDt	_sDT	(Year:=0, Month:=0, Day:=0, Hour:=0, Min:=0, Sec:=0, NSec:=0)	将当前的日期时刻分解为年、月、日、时、分、秒、纳秒
	BackupCondition	BOOL	FALSE	备份条件成立标志
	tmpString	STRING[256]	''	创建目录名称时的临时字符串
	tmpString2	STRING[256]	''	创建目录名称时的临时字符串
	BackupPath	STRING[64]	''	目录名称
	Cancel	BOOL	FALSE	取消条件成立标志
	RS1	RS		复位优先保持指令的实例1
	RS2	RS		复位优先保持指令的实例2
	RS3	RS		复位优先保持指令的实例3
	RS4	RS		复位优先保持指令的实例4
	RS5	RS		复位优先保持指令的实例5
	RS6	RS		复位优先保持指令的实例6

外部变量	名称	数据类型	常数	注释
	_Card1Ready	BOOL	☑	可使用SD存储卡标志
	_Card1Protect	BOOL	☑	SD存储卡写保护标志
	_Card1Err	BOOL	☑	SD存储卡错误标志
	_Card1Deteriorated	BOOL	☑	SD存储卡寿命警告标志
	_Card1PowerFail	BOOL	☐	SD存储卡访问中断电标志
	_BackupBusy	BOOL	☑	备份相关功能执行中标志
	PTOut_Warning_SDLife	BOOL	☐	SD存储卡寿命警告指示灯输出
	PTOut_Warning_PwrFail_onBackup	BOOL	☐	SD存储卡断电指示灯输出
	PTOut_Done	BOOL	☐	备份正常结束指示灯输出
	PTOut_Cancel	BOOL	☐	备份取消指示灯输出
	PTOut_Error	BOOL	☐	备份异常结束指示灯输出
	PTIn_Check_Backup	BOOL	☐	指示灯熄灭按钮输入
	PTIn_Cancel	BOOL	☐	取消按钮输入

```
// 确认SD存储卡的状态
CardOK := _Card1Ready OR NOT(_Card1Protect) OR NOT(_Card1Err);
PTOut_Warning_SDCardLife := _Card1Deteriorated;
RS1(Set := _Card1PowerFail, Reset1 := PTIn_Check_Backup, Q1=>PTOut_Warning_PwrFail_onBackup);

// 备份正常结束指示灯、取消指示灯、异常结束指示灯点亮
RS2(Set    := Backup_inst.Done,
    Reset1 := PTIn_Check_Backup,
    Q1     => PTOut_Done);
RS3(Set    := Backup_inst.Canceled,
    Reset1 := PTIn_Check_Backup,
    Q1     => PTOut_Cancel);
RS4(Set    := Backup_inst.Error,
    Reset1 := PTIn_Check_Backup,
    Q1     => PTOut_Error);

// 检测日期是否已改变
PreviousDay := Current_sDT.Day;
CurrentDT:=GetTime();
DtToDateStruct(In := CurrentDT,DateStruct=>Current_sDT);
RS5(Set    := ( NOT (P_First_RunMode) & (Current_sDT.Day<>PreviousDay),
    Reset1 := (Backup_inst.Done OR Backup_inst.Canceled OR Backup_inst.Error),
    Q1     => BackupCondition);

// 创建目录名称
IF(BackupCondition) THEN
  BackupPath := CONCAT( '/Backup/' , Left(In:= DtToString(CurrentDT), L:=SINT#10));
END_IF;

// 取消按钮的按下检测
RS6(Set    := (PTIn_Cancel &Backup_inst.Busy),
    Reset1 := (Backup_inst.Done OR Backup_inst.Canceled OR Backup_inst.Error),
    Q1     => Cancel);

// 执行BackupToMemoryCard指令
Backup_inst(Execute := (BackupCondition & CardOK & NOT (_BackupBusy)),
            DirName := BackupPath,
            Cancel  := Cancel);
```

时间戳指令

指令	名称	页码
NX_DOutTimeStamp	时间戳数字输出写入	2–1388
NX_AryDOutTimeStamp	时间戳数字输出数组写入	2–1393

NX_DOutTimeStamp

在时间戳方式对应的数字输出单元的输出触点写入值。

指令	名称	FB/FUN	图形表现	ST表现
NX_DOut TimeStamp	时间戳数字输出写入	FB	NX_DOutTimeStamp_instance NX_DOutTimeStamp Enable SetDOut SetTimeStamp SyncOutTime DOut TimeStamp ENO	NX_DOutTimeStamp_instance(Enable, SetDOut, SetTimeStamp, SyncOutTime, DOut, TimeStamp);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	有效	输入	TRUE：输出 “SetDOut”的值 FALSE：下降时，将输出设为FALSE	遵从数据类型	–	FALSE
SetDOut	输出值		输出值			
SetTime Stamp	指定时间戳		输出时刻		ns	0
SyncOutTime	输出同步时刻信息		EtherCAT耦合器单元和CPU单元上的NX单元的设备变量 “Time Stamp of Synchronous Output”			(*)
DOut	DOut单元输出触点	输入输出	时间戳方式对应的数字输出单元的设备变量 “Output Bit **”	遵从数据类型	–	–
TimeStamp	时间戳		时间戳方式对应的数字输出单元的设备变量 “Output Bit ** Time Stamp”		ns	

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
SetDOut	○																			
SetTime Stamp									○											
SyncOutTime									○											
DOut	○																			
TimeStamp									○											

功能

有效 “Enable”的值为TRUE时，在指定时刻，在时间戳方式对应的数字输出单元的输出触点写入输出值“SetDOut”。
“Enable”的值从TRUE下降到FALSE时，从下一任务周期起，输出触点的值变为FALSE。
指定时刻和输出时刻的误差最大为 ± 1μs。

输出同步时刻信息 “SyncOutTime”是指，时间戳方式对应的数字输出单元连接的EtherCAT耦合器单元和CPU单元上的NX总线连接的NX单元的基准时刻信息。请指定连接的EtherCAT耦合器单元和CPU单元上的NX总线连接的NX单元的设备变量 “Time Stamp of Syncronous Output”。
但是，EtherCAT耦合器单元的I/O入口，必须事先追加0x200A:02(Time Stamp of Synchronous Output)。

请对 DOut 单元输出触点 “DOut”指定时间戳方式对应的数字输出单元的输出触点上分配的设备变量“Output Bit **”。

请对时间戳 “TimeStamp”指定时间戳方式对应的数字输出单元的输出触点的时刻信息中分配的设备变量“Output Bit ** Time Stamp”。

输出时刻的指定

输出时刻的指定按以下步骤进行。

1. **获取基准时刻单元的触点的时刻信息中分配的设备变量。**
2. **将获得的时刻信息和输出触点上数据写入时刻的差换算为ns单位的数值，叠加到1)中获取的变量中。**
3. **将叠加的结果传输至本指令的指定时间戳 “SetTimeStamp”。**

详情请参考本指令的示例程序。

使用注意事项

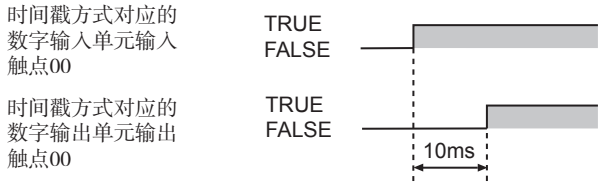
- 本指令仅可针对时间戳方式对应的数字输出单元执行。但是，即使在未连接时间戳方式对应的数字输出单元的状态下执行本指令，也不会发生异常。
- 发生 EtherCAT 通信异常或超出任务周期时，可能无法在指定时刻写入。此时，在下一任务周期以后输出。
- 其它指令或程序变更、查看本指令使用的设备变量时，请进行排他处理。
- 请对 “SyncOutTime” 指定时间戳方式对应的数字输出单元直接连接的EtherCAT耦合器单元和CPU单元上的NX总线连接的NX单元的设备变量 “Time Stamp of Synchronous Output”。但是，即使指定其它的变量，也不会发生异常。
- 请对 “DOut” 和 “TimeStamp” 指定待输出的时间戳方式对应的数字输出单元的设备变量。但是，即使指定其它的变量，也不会发生异常。
- 请对 “DOut” 和 “TimeStamp” 指定同一单元的同一通道编号的设备变量。但是，即使指定其它的变量，也不会发生异常。
- 如对 “SetTimeStamp” 指定过去的时刻，则在下一任务周期进行写入。此时，“TimeStamp” 的值变为0。

版本相关信息

本指令可用于CPU单元Ver.1.06以上且Sysmac Studio Ver.1.07以上。

示例程序

在时间戳方式对应的数字输入单元的输入触点00的值从FALSE变为TRUE的10ms后，将使时间戳方式对应的数字输出单元的输出触点00的值从FALSE变为TRUE。
输入触点00的值在比NX总线的I/O刷新时间还长的期间内为TRUE。本示例程序中，将输入触点00的上升沿设为输入触发。输入触点00的值为TRUE的时间段比NX总线的I/O刷新时间短时，可能无法检测到输入触点00的上升沿。此时，必须进行变更。比如将输入触点00变化时刻的变化设为输入触发等。在传感器输入变化后的一定时间内将输出设为 ON 的示例程序，请参阅 “NX 系列 数字 I/O 单元 用户手册 (SBCA-407)”。



网络构成

网络构成如下所示。在EtherCAT的节点地址1连接了下列构成的从站终端。分别分配以下设备名称。

单元编号	型号	单元	设备名称
0	NX-ECC201	EtherCAT耦合器单元	E001
1	NX-ID3344	时间戳方式对应的数字输入单元	N1
2	NX-OD2154	时间戳方式对应的数字输出单元	N2

单元动作设定

时间戳方式对应的数字输入单元的单元动作设定如下所示。

项目名	设定值	含义
时间戳(触发设定)/Input Bit 00 Trigger Setting	FALSE	获取变化时刻的边沿：上升沿
时间戳(模式设定)/Input Bit 00 Mode Setting	TRUE	获取变化时刻的动作模式：单次获取(最初变化的时刻)

I/O映射

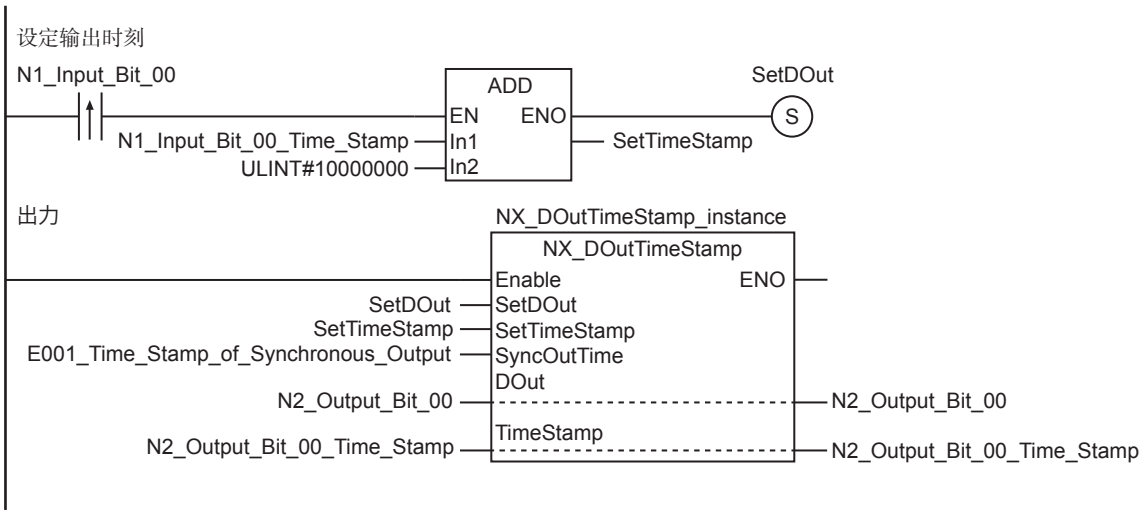
I/O映射设定如下所示。

位置	端口	说明	R/W	数据类型	变量	变量种类
Node1	Time Stamp of Synchronous Output	保存连接的NX单元同步输出时的时刻信息。(单位:ns)	R	ULINT	E001_Time_Stamp_of_Synchronous_Output	全局变量
Unit1	Input Bit 00	输入触点00	R	BOOL	N1_Input_Bit_00	全局变量
Unit1	Input Bit 00 Time Stamp	输入触点00的变化时刻	R	ULINT	N1_Input_Bit_00_Time_Stamp	全局变量
Unit2	Output Bit 00 Time Stamp	输出触点00的指定时刻	W	ULINT	N2_Output_Bit_00_Time_Stamp	全局变量
Unit2	Output Bit 00	输出触点00	W	BOOL	N2_Output_Bit_00	全局变量

LD

内部变量	名称	数据类型	初始值	注释
	SetTimeStamp	ULINT	0	指定时间戳
	SetDOut	BOOL	FALSE	输出值
	NX_DOutTimeStamp_instance	NX_DOutTimeStamp		

外部变量	名称	数据类型	常数	注释
	N1_Input_Bit_00	BOOL	☐	输入触点00
	N1_Input_Bit_00_Time_Stamp	ULINT	☐	输入触点00的变化时刻
	E001_Time_Stamp_of_Synchronous_Output	ULINT	☐	连接的NX单元同步输出时的时刻信息
	N2_Output_Bit_00	BOOL	☐	输出触点00
	N2_Output_Bit_00_Time_Stamp	ULINT	☐	输出触点00的指定时刻



ST

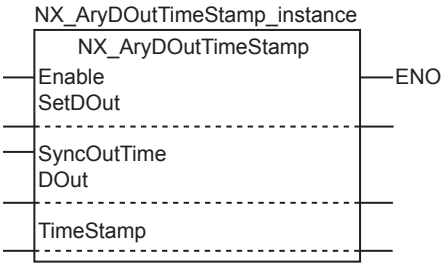
内部变量	名称	数据类型	初始值	注释
	SetEN	BOOL	FALSE	执行条件
	SetTimeStamp	ULINT	0	指定时间戳
	SetDOut	BOOL	FALSE	输出值
	R_TRIG_instance	R_TRIG		
	NX_DOutTimeStamp _instance	NX_DOutTimeStamp		

外部变量	名称	数据类型	常数	注释
	N1_Input_Bit_00	BOOL	☐	输入触点00
	N1_Input_Bit_00_Time_Stamp	ULINT	☐	输入触点00的变化时刻
	E001_Time_Stamp_of_Synchronous_Output	ULINT	☐	连接的NX单元同步输出时的时刻信息
	N2_Output_Bit_00	BOOL	☐	输出触点00
	N2_Output_Bit_00_Time_Stamp	ULINT	☐	输出触点00的指定时刻

```
// 执行触发输入
R_TRIG_instance( N1_Input_Bit_00, SetEN);

// 输出时刻设定
IF ( SetEN = TRUE ) THEN
  SetDOut      := TRUE;
  SetTimeStamp := N1_Input_Bit_00_Time_Stamp + ULINT#10000000;
END_IF;

// 输出
NX_DOutTimeStamp_instance(
  Enable       := TRUE,
  SetDOut      := SetDOut,
  SetTimeStamp := SetTimeStamp,
  SyncOutTime  := E001_Time_Stamp_of_Synchronous_Output,
  DOut         := N2_Output_Bit_00,
  TimeStamp    := N2_Output_Bit_00_Time_Stamp);
```

NX_AryDOutTimeStamp

从时间戳方式对应的数字输出单元输出脉冲。

指令	名称	FB/FUN	图形表现	ST表现
NX_AryDOut TimeStamp	时间戳数字输出数组写入	FB		NX_AryDOutTimeStamp _instance(Enable, SetDOut, SyncOutTime, DOut, TimeStamp);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Enable	有效	输入	TRUE：根据 “SetDOut” 的设定进行输出 FALSE：下降时，将输出设为FALSE	遵从数据类型	–	FALSE
SyncOutTime	输出同步时刻信息		EtherCAT耦合器单元和CPU单元上的NX单元的设备变量 “Time Stamp of Synchronous Output”		ns	(*)
SetDOut	输出脉冲	输入输出	输出脉冲	–	–	–
DOut	DOut单元输出触点		时间戳方式对应的数字输出单元的设备变量 “Output Bit **”	遵从数据类型		
TimeStamp	时间戳		时间戳方式对应的数字输出单元的设备变量 “Output Bit ** Time Stamp”		ns	

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Enable	○																			
SyncOutTime									○											
SetDOut	结构体_sOUTPUT_REF 详情参阅功能说明																			
DOut	○																			
TimeStamp									○											

功能

有效“Enable”的值为TRUE时，在指定时刻，从时间戳方式对应的数字输出单元输出输出脉冲“SetDOut”设定的脉冲。

“Enable”的值从TRUE下降到FALSE时，将向时间戳方式对应的数字输出单元的输出设为FALSE。
指定时刻和输出时刻的误差最大为 ± 1μs。

输出同步时刻信息 “SyncOutTime”是指，时间戳方式对应的数字输出单元连接的EtherCAT耦合器单元和CPU单元上的NX总线连接的NX单元的基准时刻信息。请指定连接的EtherCAT耦合器单元和CPU单元上的NX总线连接的NX单元的设备变量 “Time Stamp of Syncronous Output”。
但是，EtherCAT耦合器单元的I/O入口，必须事先追加0x200A:02(Time Stamp of Synchronous Output)。

请对 DOut 单元输出触点 “DOut”指定时间戳方式对应的数字输出单元的输出触点上分配的设备变量“Output Bit **”。

请对时间戳“TimeStamp”指定时间戳方式对应的数字输出单元的输出触点的时刻信息中分配的设备变量“Output Bit ** Time Stamp”。

输出时刻的指定

输出时刻的指定按以下步骤进行。

1. 获取基准时刻单元的触点的时刻信息中分配的设备变量。
2. 将获得的时刻信息和输出触点ON时刻的差换算为ns单位的数值，叠加到1)中获取的变量中。
3. 将叠加的结果传输至本指令的SetDOut.OnTime[]。
4. 同2)，将获得的时刻信息和输出触点OFF时刻的差换算为ns单位的数值，叠加到1)中获取的变量中。
5. 将叠加的结果传输至本指令的SetDOut.OffTime[]。

输出脉冲的指定

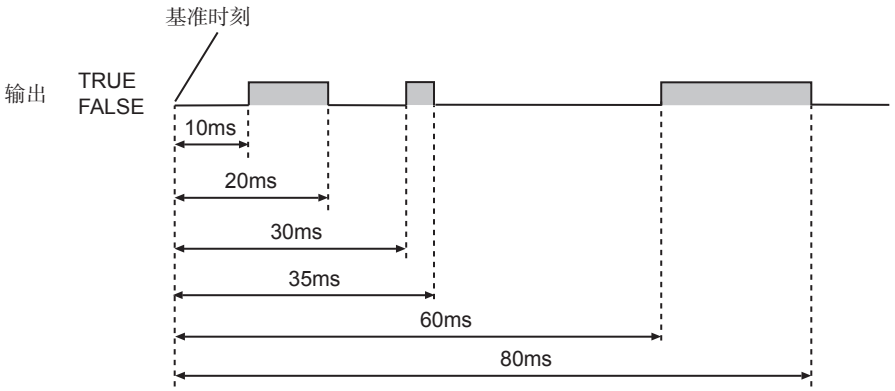
输出脉冲 “SetDOut”的数据类型为结构体_sOUTPUT_REF。规格如下所示。

变量	名称	内容	数据类型	有效范围	单位	初始值
SetDOut	输出脉冲	输出脉冲	_sOUTPUT_REF	–	–	–
EnableOut	输出有效	输出有效标志 TRUE：“OnTime”、“OffTime”的设定有效 FALSE：“OnTime”、“OffTime”的设定无效	BOOL	遵从数据类型	–	FALSE
OnTime[]数组	ON时刻	将输出触点设为ON的时刻	ARRAY[0..15] OF ULINT		ns	所有元素全为0
OffTime[]数组	OFF时刻	将输出触点设为OFF的时刻	ARRAY[0..15] OF ULINT			

ON时刻OnTime[]和OFF时刻OffTime[]分别是元素数为16的数组。2个数组的同一元素编号的值表示一个脉冲的ON时刻和OFF时刻。因此，最多可指定16个脉冲。2个数组的同一元素编号的值均为0时，该元素编号以后的值无效。

例如，OnTime[]和OffTime[]的元素值如下时，输出如下图所示。下表的值是以基准时刻几ms后表示指定时刻。

变量	元素编号				
	0	1	2	3	4
OnTime[]	10ms后	30ms后	60ms后	0	90ms后
OffTime[]	20ms后	35ms后	80ms后	0	100ms后



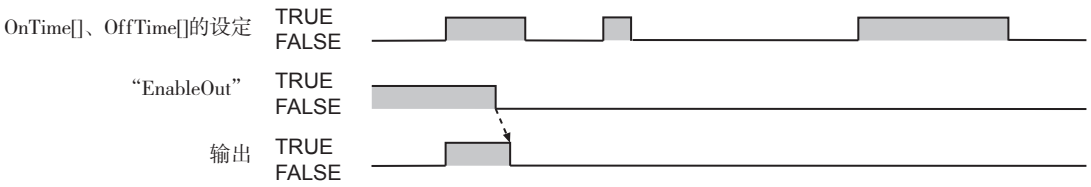
OnTime[]和OffTime[]的元素值不必按照时间的升序排列。因此，2个数组的元素值如下时，输出也与上图相同。

变量	元素编号				
	0	1	2	3	4
OnTime[]	30ms后	60ms后	10ms后	0	90ms后
OffTime[]	35ms后	80ms后	20ms后	0	100ms后

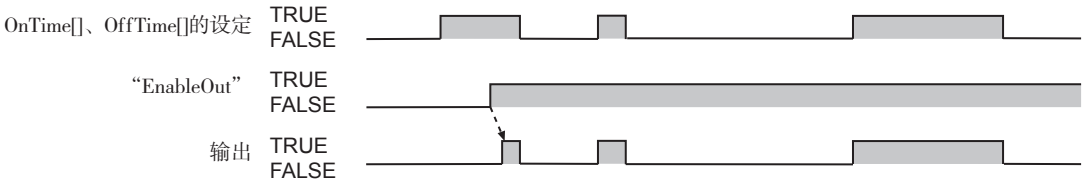
● 输出有效 “EnableOut”

输出有效 “EnableOut”是使OnTime[]、OffTime[]的设定生效的标志。“EnableOut”的值为FALSE时，无论OnTime[]、OffTime[]的值如何，输出的值均为FALSE。

在执行本指令的过程中，可变更 “EnableOut”的值。
将 “EnableOut”的值从TRUE变更为FALSE时，输出的值变为FALSE。

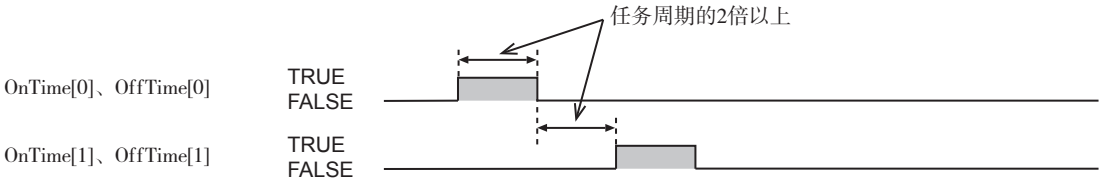


将 “EnableOut”的值从FALSE变更为TRUE时，OnTime[]、OffTime[]的值生效。



● 输出脉冲的最小宽度

要以1μs的时间精度输出输出脉冲时，请将OnTime[]和OffTime[]的间隔设为任务周期的2倍以上。如指定值小于2倍，可能不会输出指定脉冲，或比指定的ON/OFF时刻延迟1个任务周期进行输出。

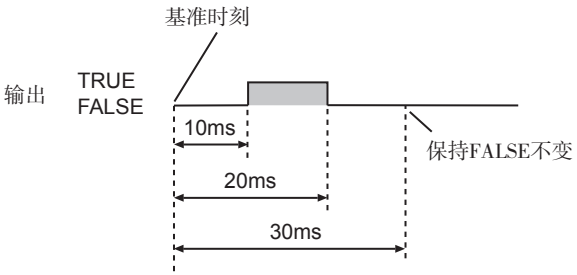


详细动作请参阅以后的说明。

● OnTime[]和OffTime[]的相同元素编号的值相同时

OnTime[]和OffTime[]的相同元素编号的值相同时，输出为FALSE。因此，2个数组的元素值如下时，输出如下图所示。

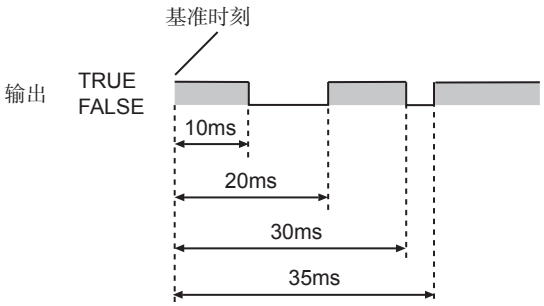
变量	元素编号		
	0	1	2
OnTime[]	10ms后	30ms后	0
OffTime[]	20ms后	30ms后	0



● OnTime[]的元素值大于同一元素编号的OffTime[]的元素值时

OnTime[]的元素值大于同一元素编号的OffTime[]的元素值时，输出值在先变为FALSE后变为TRUE。
OnTime[] 元素的最小值大于 OffTime[] 元素的最小值时，在执行本指令后，输出值立即变为 TRUE。
OnTime[]元素的最大值大于OffTime[]元素的最大值时，在本指令执行结束后，输出值保持TRUE。
因此，2个数组的元素值如下时，输出如下图所示。

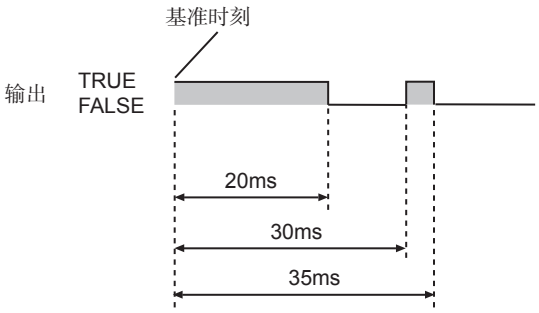
变量	元素编号		
	0	1	2
OnTime[]	20ms后	35ms后	0
OffTime[]	10ms后	30ms后	0



● OnTime[]和OffTime[]任一元素值为0时

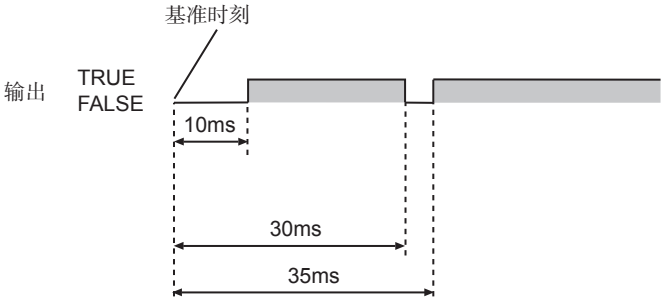
OnTime[]和OffTime[]任一元素值为0时，在执行本指令后，输出值立即变为TRUE或FALSE。
仅OnTime[]的元素值为0时，在执行本指令后，输出值立即变为TRUE。因此，2个数组的元素值如下时，输出如下图所示。

变量	元素编号		
	0	1	2
OnTime[]	0	30ms后	0
OffTime[]	20ms后	35ms后	0



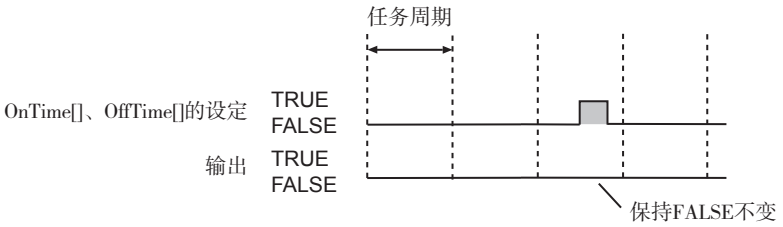
仅OffTime[]的元素值为0时，在执行本指令后，输出值立即变为FALSE。因此，2个数组的元素值如下时，输出如下图所示。

变量	元素编号		
	0	1	2
OnTime[]	10ms后	35ms后	0
OffTime[]	0	30ms后	0



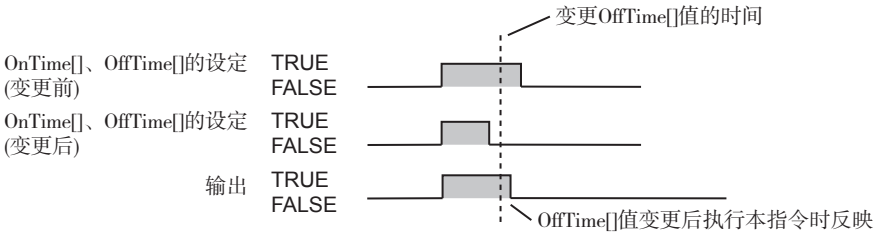
● 在同一任务周期内，将输出值设为TRUE的设定和设为FALSE的设定连续时

在同一任务周期内，将输出设为TRUE的设定和设为FALSE的设定连续时，输出值不变。



● 本指令生效中变更OnTime[]、OffTime[]的值时

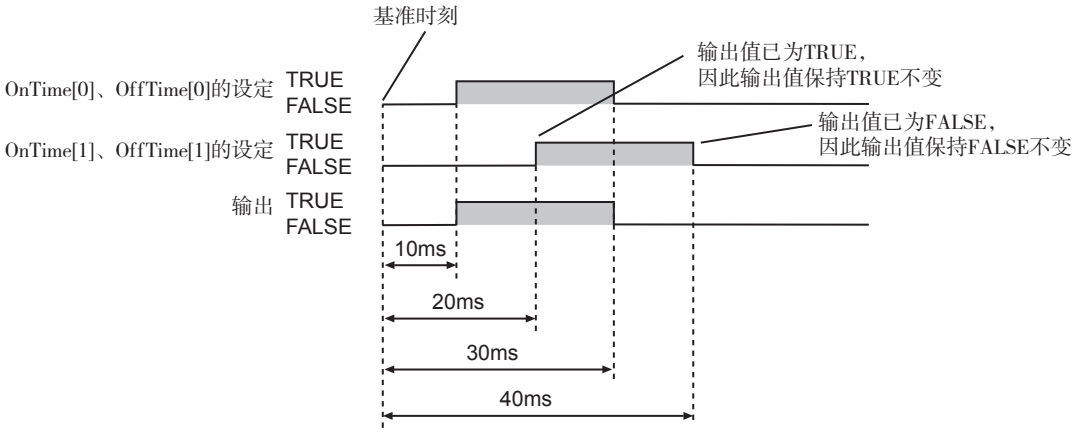
本指令生效中，可变更OnTime[]、OffTime[]的值。变更内容将在变更后执行本指令时反映。



● 将输出值设为TRUE的设定重叠时

将输出值设为TRUE的设定重叠时，不发生异常，输出值保持TRUE不变。将输出值设为FALSE的设定重叠时也一样。因此，2个数组的元素值如下时，输出如下图所示。

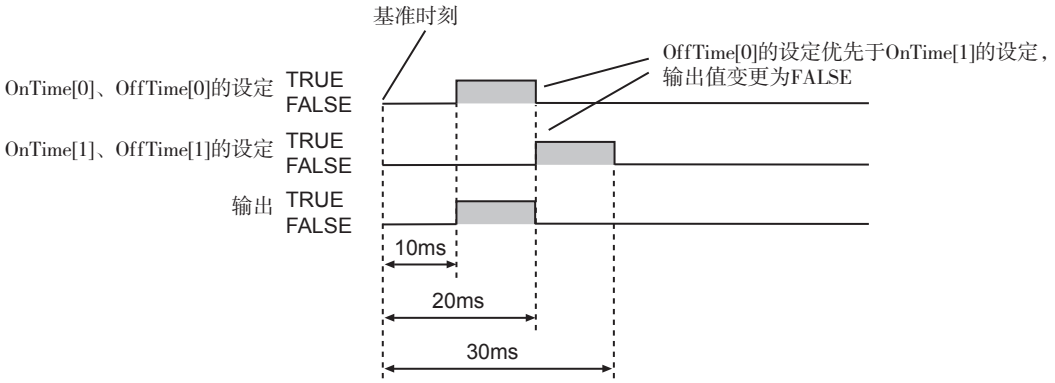
变量	元素编号		
	0	1	2
OnTime[]	10ms后	20ms后	0
OffTime[]	30ms后	40ms后	0



● 将输出值设为TRUE的设定和设为FALSE的设定在同一时刻重叠时

将输出值设为TRUE的设定和设为FALSE的设定在同一时刻重叠时，不发生异常，OnTime[]和OffTime[]的元素编号小的设定优先。因此，2个数组的元素值如下时，输出如下图所示。

变量	元素编号		
	0	1	2
OnTime[]	10ms后	20ms后	0
OffTime[]	20ms后	30ms后	0



参考

使用MC_DigitalCamSwitch指令时使用本指令。MC_DigitalCamSwitch指令的详情，请参阅 “NJ/NX系列指令基准手册 运动篇(SBCE-364)”。

使用注意事项

- 本指令仅可针对时间戳方式对应的数字输出单元执行。但是，即使在未连接时间戳方式对应的数字输出单元的状态下执行本指令，也不会发生异常。
- 发生 EtherCAT 通信异常或超出任务周期时，可能无法在指定时刻输出。此时，在下一任务周期以后输出。
- 其它指令或程序变更、查看本指令使用的设备变量时，请进行排他处理。
- 请对 “SyncOutTime” 指定时间戳方式对应的数字输出单元直接连接的EtherCAT耦合器单元和CPU单元上的NX总线连接的NX单元的设备变量 “Time Stamp of Synchronous Output”。但是，即使指定其它的变量，也不会发生异常。
- 请对 “DOut” 和 “TimeStamp” 指定待输出的时间戳方式对应的数字输出单元的设备变量。但是，即使指定其它的变量，也不会发生异常。
- 请对 “DOut” 和 “TimeStamp” 指定同一单元的同一通道编号的设备变量。但是，即使指定其它的变量，也不会发生异常。

版本相关信息

本指令可用于CPU单元Ver.1.06以上且Sysmac Studio Ver.1.07以上。

示例程序

示例程序请参阅 “NJ/NX系列 指令基准手册 运动篇(SBCE-364)” 的MC_DigitalCamSwitch指令。

其它指令

ReadNbit_**

读取位串内的多位。

指令	名称	FB/FUN	图形表现	ST表现
ReadNbit_**	N位读取组	FUN	(@)ReadNbit_** EN ENO In Out Pos Size **为位串的数据类型名称	Out:=ReadNbit_**(In, Pos, Size); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	读取源	输入	读取源的位串	遵从数据类型	–	0
Pos	读取位置		待读取的位位置	0～“In”的位数–1		
Size	读取大小		待读取的位数	0～“In”的位数		1
Out	读取结果	输出	读取结果	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Pos						○														
Size						○														
Out	与“In”相同的数据类型																			

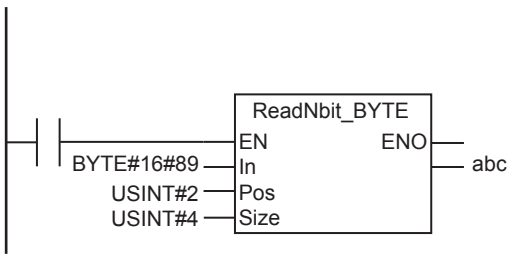
功能

将读取源位串 “In”的包含读取位置 “Pos”的高位 “Size”位的值代入读取结果 “Out”。

指令名称因 “In”、“Out”的数据类型而异。例如，“In”、“Out”为WORD型时，指令名称为ReadNbit_WORD。

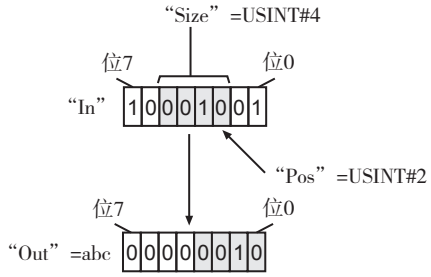
ReadNbit_BYTE指令下，“In”=BYTE#16#89，“Pos”=USINT#2，“Size”=USINT#4时的示例如下所示。

LD



ST

```
abc:=ReadNbit_BYTE(BYTE#16#89, USINT#2,
                   USINT#4);
```



参考

将多个位写入位串时，请使用WriteNbit_**指令。

使用注意事项

- 请将 “In”和 “Out”的数据类型设为相同。
- “Size”的值为0时，“Out”的值为16#0。
- 以下情况时会发生异常。ENO变为FALSE，“Out”不变。
  - “Size”的值超过有效范围时。
  - “Pos”的值超过有效范围时。
  - “In”从 “Pos”值所示位置起无 “Size”值所示数量的位串时。

WriteNbit_**

在位串内写入多位。

指令	名称	FB/FUN	图形表现	ST表现
WriteNbit_**	N位写入组	FUN	(@)WriteNbit_** EN ENO In Out InOut Pos Size **为位串的数据类型名称	WriteNbit_**(In, InOut, Pos, Size); **为位串的数据类型名称

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	读取源	输入	读取写入 “InOut” 的值的字符串	遵从数据类型	–	0
Pos	写入位置		待写入的位位置	0～ “InOut” 的位数–1		
Size	写入大小		待写入的位数	0～ “In” 的位数		1
InOut	写入对象	输入输出	读取结果	遵从数据类型	–	–
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

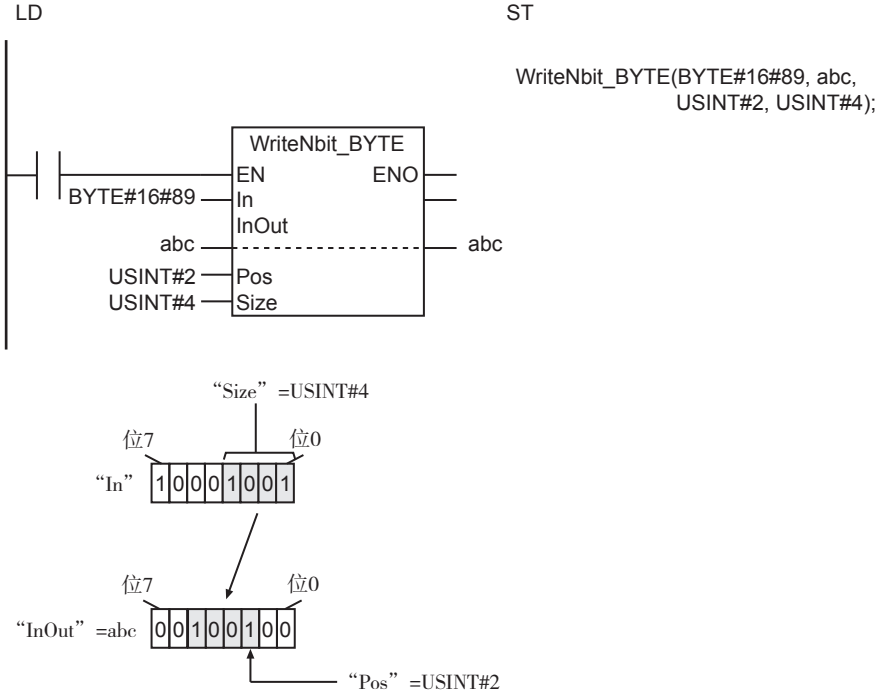
	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In		○	○	○	○															
Pos						○														
Size						○														
InOut	与 “In” 相同的数据类型																			
Out	○																			

功能

首先，读取读取源“In”的低位“Size”位。然后，将读取的值覆盖写入对象“InOut”的写入位置“Pos”。

指令名称因 “In”、“Out”的数据类型而异。例如，“In”、“Out”为WORD型时，指令名称为WriteNbit_WORD。

WriteNbit_BYTE指令下，“In”=BYTE#16#89，“Pos”=USINT#2，“Size”=USINT#4时的示例如下所示。



参考

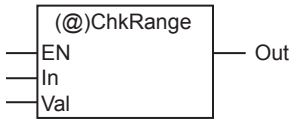
读取位串的多个位时，请使用ReadNbit_**指令。

使用注意事项

- 请将 “In”和 “InOut”的数据类型设为相同。
- “Size”的值为0时，“InOut”的值不变。
- 在ST程序中使用本指令时，不使用返回值 “Out”。
- 以下情况时会发生异常。ENO变为FALSE，“InOut”不变。
  - “Size”的值超过有效范围时。
  - “Pos”的值超过有效范围时。
  - “InOut”从 “Pos”值所示位置起无 “Size”值所示数量的位串时。

ChkRange

判定变量值是否在范围型指定的有效范围内。

指令	名称	FB/FUN	图形表现	ST表现
ChkRange	范围型变量检查	FUN		Out:=ChkRange(In, Val);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
In	检查对象	输入	检查对象	遵从数据类型	–	(*)
Val	范围指定变量		范围指定变量	遵从范围指定		
Out	检查结果	输出	检查结果	遵从数据类型	–	–

* 省略输入参数时，初始值不适用。编连时会发生异常。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
In						○	○	○	○	○	○	○	○							
Val	范围指定　源的基本数据类型与 “In” 相同																			
Out	○																			

功能

判定检查对象 “In” 的值是否在范围指定变量 “Val” 的有效范围内。如果在有效范围内，则检查结果 “Out” 为TRUE；否则，为FALSE。

参考

范围指定可对整数型(USINT，UINT，UDINT，ULINT，SINT，INT，DINT，LINT)的变量进行定义。

使用注意事项

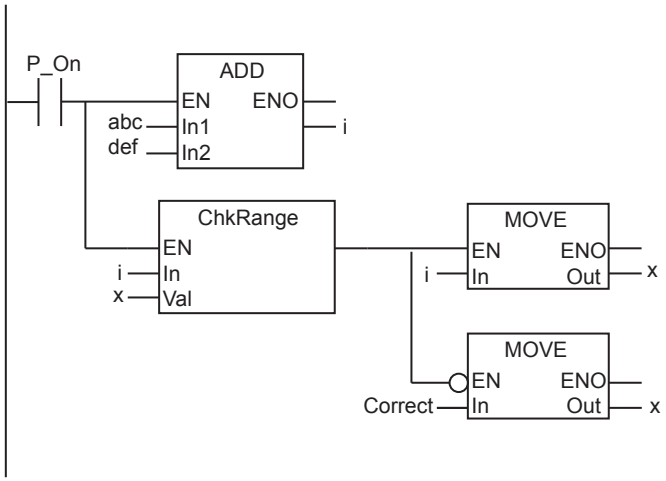
- “In” 不是范围指定变量时，“Out” 的值变为TRUE。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out” 的值为FALSE。

示例程序

判定相加结果i是否在范围指定变量x的有效范围(10 ~ 99)内。如果在有效范围内，则将i的值代入x。如果超过有效范围，则将变量Correct的值代入x。

LD

名称	数据类型	初始值
i	INT	0
abc	INT	0
def	INT	0
x	INT(10..99)	10
Correct	INT	0



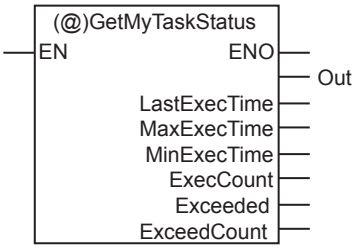
ST

名称	数据类型	初始值
i	INT	0
abc	INT	0
def	INT	0
Chk	BOOL	FALSE
x	INT(10..99)	10
Correct	INT	0

```
i := abc+def;
Chk:=ChkRange(i, x);        // 范围型变量检查

IF (Chk=TRUE) THEN
    x := i;                 // 如果i的值在范围内，则将i代入x
ELSE
    x := Correct;           // 如果i的值超过范围，则将Correct代入x
END_IF;
```

GetMyTaskStatus

读取我的任务状态。

指令	名称	FB/FUN	图形表现	ST表现
GetMyTaskStatus	我的任务状态读取	FUN		GetMyTaskStatus(LastExecTime, MaxExecTime, MinExecTime, ExecCount, Exceeded, ExceedCount);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	返回值	输出	始终为TRUE	仅TRUE	–	–
LastExecTime	上次任务执行时间		上次我的任务执行时间	遵从数据类型(*)	ns	
MaxExecTime	任务执行时间最大值		我的任务执行时间最大值			
MinExecTime	任务执行时间最小值		我的任务执行时间最小值			
ExecCount	任务执行次数		我的任务执行次数	遵从数据类型	–	
Exceeded	超过任务周期标志		TRUE：上次执行我的任务未在任务周期内完成。 FALSE：上次执行我的任务已在任务周期内完成。			
ExceedCount	超过任务周期次数		我的任务超过任务周期的次数			

* 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out						○														
LastExecTime																○				
MaxExecTime																○				
MinExecTime																○				
ExecCount								○												
Exceeded	○																			
ExceedCount								○												

功能

获取我的任务的任务状态。
任务状态是指上次任务执行时间 “LastExecTime”、任务执行时间最大值 “MaxExecTime”、任务执行时间最小值“MinExecTime”、任务执行次数“ExecCount”、超过任务周期标志 “Exceeded”、超过任务周期次数 “ExceedCount”。

参考

在以下时间将 “MaxExecTime”、“MinExecTime”、“ExecCount”、“ExceedCount”的值复位。

- 运行开始时
- Sysmac Studio的任务执行时间监控画面中执行复位时

使用注意事项

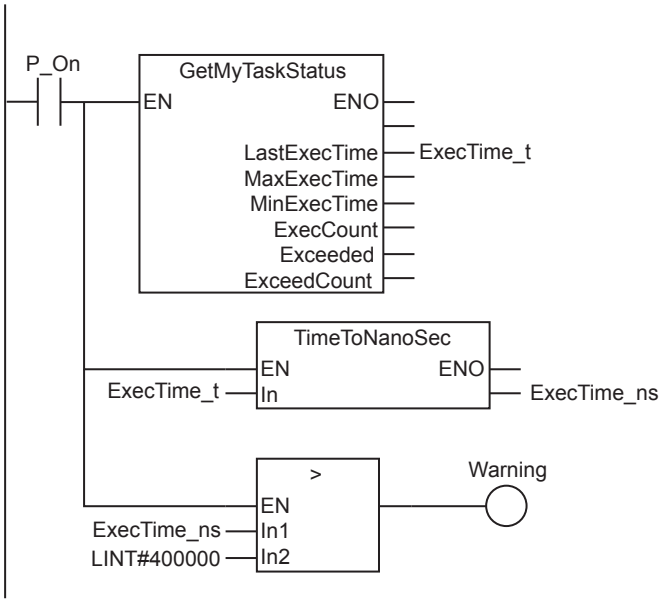
- 如果 “ExecCount”和 “ExceedCount”的值超过UDINT型的最大值(4,294,967,295)，则返回0。
- 在ST程序中使用本指令时，不使用返回值 “Out”。

示例程序

获取我的任务的任务状态。如果上次任务执行时间超过400us(400000ns)，则将Warning的值设为TRUE。

LD

名称	数据类型	初始值	注释
ExecTime_t	TIME	T#0s	上次任务执行时间(TIME型)
ExecTime_ns	LINT	0	上次任务执行时间(LINT型 纳秒数)
Warning	BOOL	FALSE	警告



ST

名称	数据类型	初始值	注释
ExecTime_t	TIME	T#0s	上次任务执行时间(TIME型)
ExecTime_ns	LINT	0	上次任务执行时间(LINT型 纳秒数)
Warning	BOOL	FALSE	警告

```
GetMyTaskStatus(LastExecTime=>ExecTime_t);     // 获取上次任务周期
ExecTime_ns:=TimeToNanoSec(ExecTime_t);        // 将上次任务周期从TIME型转换为纳秒数
IF (ExecTime_ns>DINT#400000) THEN              // 上次任务周期超过400,000ns时，
    Warning:=TRUE;                             //       将TRUE代入变量Warning。
ELSE
    Warning:=FALSE;
END_IF;
```

GetMyTaskInterval

读取我的任务的任务周期。

指令	名称	FB/FUN	图形表现	ST表现
GetMyTask Interval	我的任务设定周期读取	FUN	(@)GetMyTaskInterval EN ENO Out	Out:=GetMyTaskInterval();

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	任务周期	输出	我的任务的任务周期	遵从数据类型*1	ms	–

*1 不含负数。

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out																○				

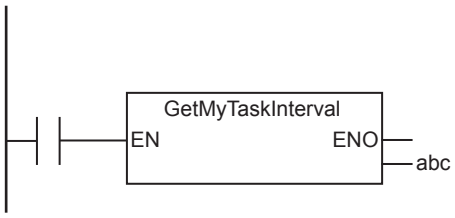
功能

执行本指令的任务为原始恒定周期任务或固定周期任务时，我的任务的任务周期保存在任务周期 “Out”中。

执行本指令的任务为事件任务时，“Out”的值为T#0s。

示例如下所示。我的任务的任务周期为1ms时，abc的值为T#1ms。

LD



ST

```
abc:=GetMyTaskInterval();
```

版本相关信息

本指令可用于CPU单元 Ver.1.08以上且Sysmac Studio Ver.1.09以上。

示例程序

开始运行后，最初执行本程序时，读取我的任务的任务周期。之后，将读取的任务周期从TIME型转换为LREAL型的ms单位。是为了计算各任务周期的轴的目标位置等而使用的示例程序。

从TIME型转换为LREAL型的ms单位的步骤如下所示。

1 使用GerMyTaskInterval指令，读取TIME型的任务周期。

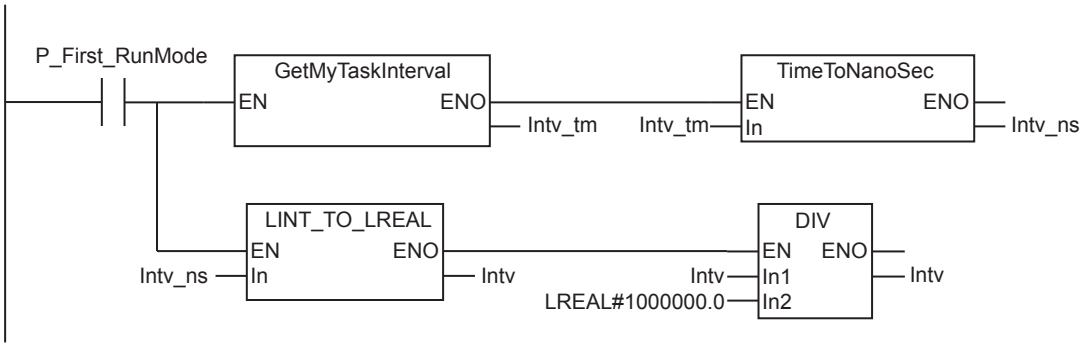
2 使用TimeToNanoSec指令，从TIME型转换为LINT型的ns单位。

3 使用LINT_TO_LREAL指令，从LINT型的ns单位转换为LREAL型的ns单位。

4 使用DIV指令，用3的结果除以1,000,000，转换为ms单位。

LD

名称	数据类型	初始值	注释
Intv_tm	TIME	T#0s	以TIME型表示的任务周期
Intv_ns	LINT	0	以LINT型的ns为单位表示的任务周期
Intv	LREAL	0	以LREAL型的ms为单位表示的任务周期



ST

名称	数据类型	初始值	注释
Intv	LREAL	0	以LREAL型的ms为单位表示的任务周期

```
IF P_First_RunMode = TRUE THEN
  Intv := LINT_TO_LREAL(TimeToNanoSec(GetMyTaskInterval()))/1000000;
END_IF;
```

Task_IsActive

判定指定任务是否正在执行。

指令	名称	FB/FUN	图形表现	ST表现
Task_IsActive	任务执行中判定	FUN	(@)Task_IsActive EN TaskName Out	Out:=Task_IsActive(TaskName);

变量

	名称	输入/输出	内容	有效范围	单位	初始值
TaskName	任务名称	输入	任务名称	最大64个字节(63个半角英数字字符+结尾NULL字符)	–	''
Out	执行中判定	输出	TRUE ：执行中或待机状态 FALSE：非执行中	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
TaskName																				○
Out	○																			

功能

判定任务名称 “TaskName”指定的任务是否处于执行中或待机状态。待机状态是指任务启动后，因更高优先顺序的任务启动而使处理中断的状态。
如果处于执行中或待机状态，则执行中判定 “Out”的值为 TRUE。如果处于非执行中，则 “Out”的值为FALSE。

使用注意事项

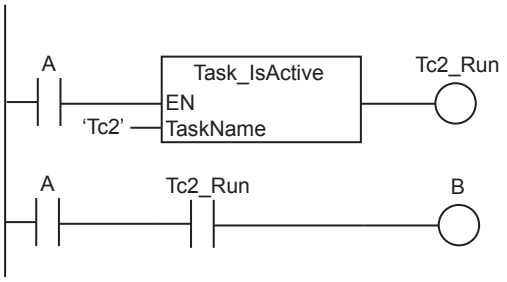
- 无法为 “TaskName”指定已代入字符串的变量。请直接指定字符串。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。
- 以下情况时会发生异常。“Out”不变。
  - “TaskName”指定的任务不存在时。

示例程序

变量A的值变为TRUE时，检查周期任务Tc2是否正在启动。如果正在启动，则将变量B的值设为TRUE。

LD

名称	数据类型	初始值	注释
A	BOOL	FALSE	
B	BOOL	FALSE	
Tc2_Run	BOOL	FALSE	任务Tc2的执行状态



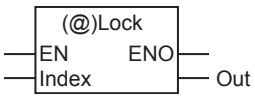
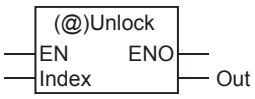
ST

名称	数据类型	初始值	注释
A	BOOL	FALSE	
B	BOOL	FALSE	
Tc2_Run	BOOL	FALSE	任务Tc2的执行状态

```
IF (A=TRUE) THEN
    // 任务执行中判定
    Tc2_Run:=Task_IsActive( 'Tc2' );
    // 如果Tc2处于执行中，则将TRUE代入变量B
    IF (Tc2_Run=TRUE) THEN
        B := TRUE;
    END_IF;

END_IF;
```

Lock/Unlock

Lock ：开启任务间的排他锁。与其它任务锁定编号相同的区间无法执行。
Unlock：解除任务间排他锁。

指令	名称	FB/FUN	图形表现	ST表现
Lock	任务间排他锁	FUN		Lock(Index);
Unlock	任务间排他锁解除	FUN		Unlock(Index);

变量

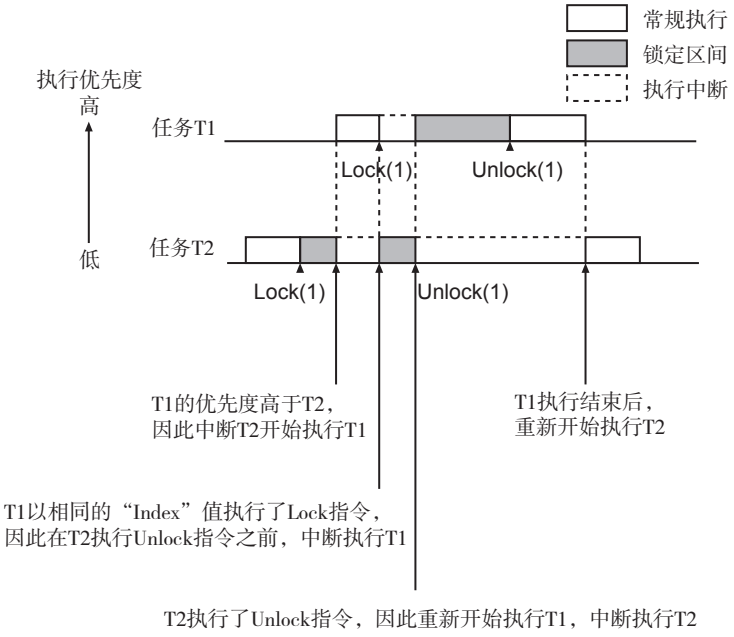
	名称	输入/输出	内容	有效范围	单位	初始值
Index	锁定编号	输入	锁定编号	遵从数据类型	–	0
Out	返回值	输出	始终为TRUE	仅TRUE	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Index						○														
Out	○																			

功能

正在执行从Lock指令到Unlock指令的区间(锁定区间)时，应使锁定编号与其他任务相同的锁定区间无法执行。
锁定编号由 “Index”指定。

动作示例如下所示。
假设任务T1、T2均具有 “Index”=1的锁定区间。如果先执行T2的Lock指令，则直至执行T2的Unlock指令之前，T1的锁定区间无法执行。



“Index”的值不同的锁定区间相互之间无影响。

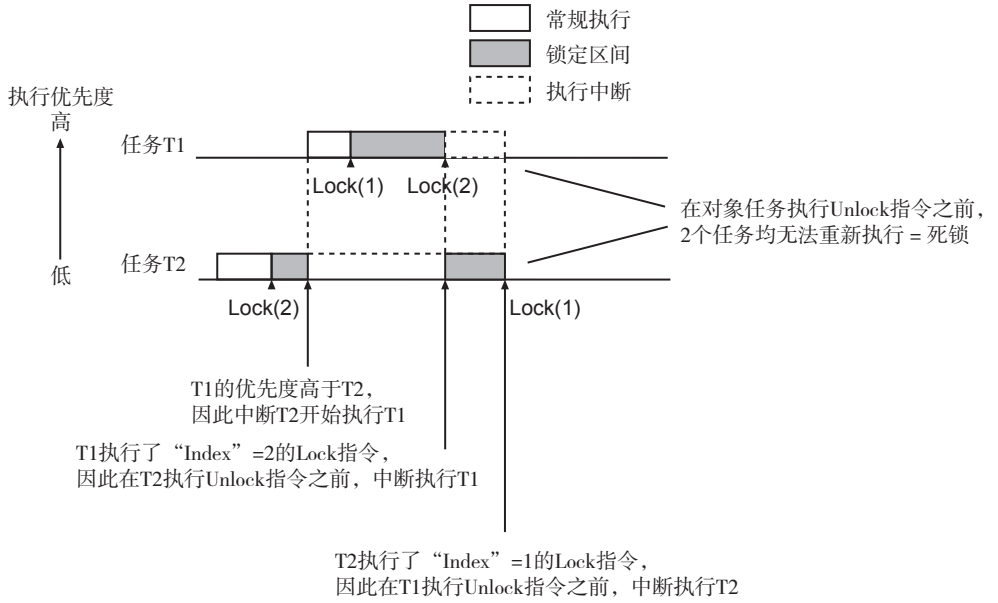
参考

- 本指令用于可能从多个任务读写相同数据的场合等。某一任务正在读写数据时，使其他任务无法读写该数据。
- 如果 “Index”的值不同，则多个Lock指令和Unlock指令也可同时存在于相同POU中。其配置也可为嵌套结构。

使用注意事项

- 锁定区间的长度请设为所需的最低限度。如果锁定区间较长，可能会超过任务的执行周期。
- 请务必在相同POU的相同段内成对使用Lock指令和Unlock指令。
- 最多可同时设定16777215个锁定区间。

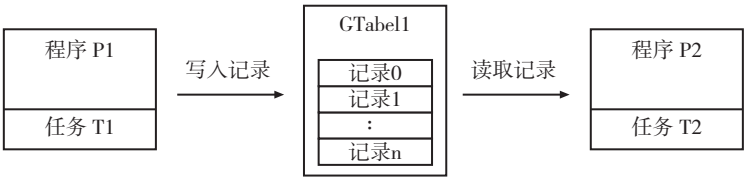
- 多个任务使用多个 Lock 指令时，根据其配置的不同，可能会发生死锁。死锁时会发生任务执行超时异常，从而全部停止。发生死锁时的示例如下所示。



- 以下情况时会发生异常。“Out”不变。
  - 试图同时设定超过16777215个锁定区间时。

示例程序

任务T1的程序P1和任务T2的程序P2访问相同全局变量GTable1。
写入请求WriteReq的值变为TRUE时，P1将1条记录写入记录数组GTable1.Record[ ]，对索引GTable1.Index进行增量。
读取请求ReadReq的值变为TRUE时，P2对GTable1.Index进行减量，从GTable1.Record[ ]读取1条记录。
使用Lock指令，以避免同时进行读取和写入。



全局变量GTable的定义

数据类型

名称	数据类型	注释
USERTABLE	STRUCT	记录保存结构体
Index	INT	索引
Record	ARRAY[0..99] OF LREAL	记录数组

全局变量

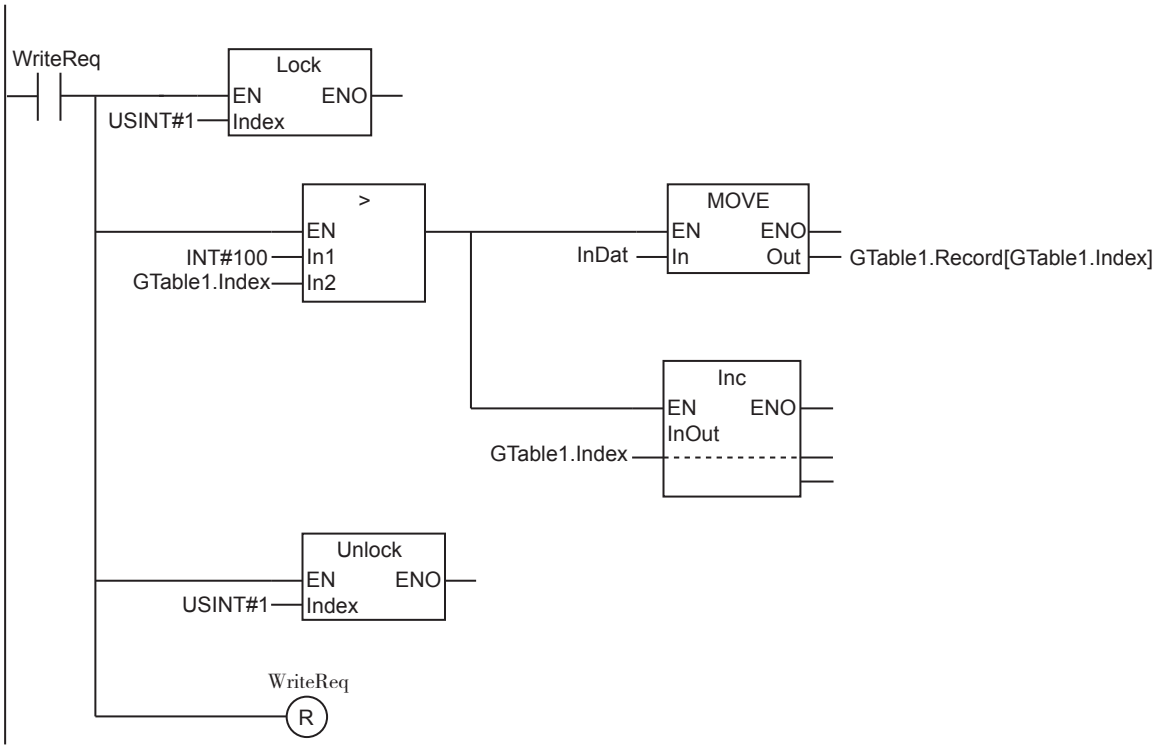
名称	数据类型	初始值	注释
GTable1	USERTABLE	(Index:=0, Record:=[100(0.0)])	记录保存结构体

程序P1

LD

内部变量	名称	数据类型	初始值	注释
	WriteReq	BOOL	FALSE	写入请求
	InDat	LREAL	0.0	写入数据

外部变量	名称	数据类型	注释
	GTable1	USERTABLE	记录保存结构体



ST

内部变量	名称	数据类型	初始值	注释
	WriteReq	BOOL	FALSE	写入请求
	InDat	LREAL	0.0	写入数据

外部变量	名称	数据类型	注释
	GTable1	USERTABLE	记录保存结构体

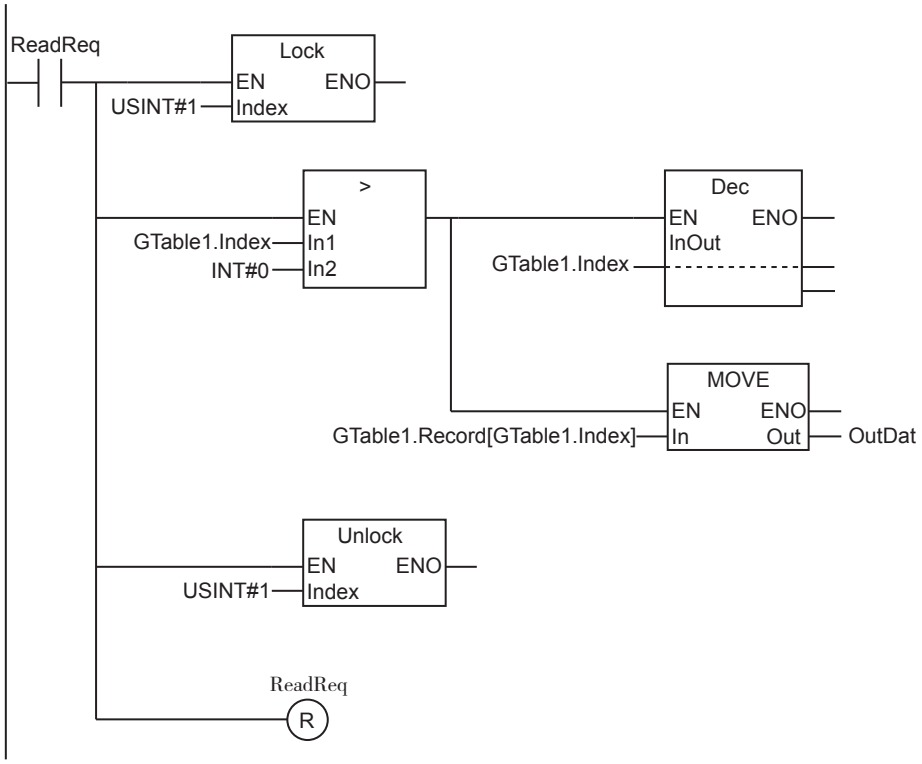
```
// 写入请求检测
IF (WriteReq=TRUE) THEN

    // 执行Lock指令
    Lock(USINT#1);

    IF (INT#100>GTable1.Index) THEN
        GTable1.Record[GTable1.Index] :=InDat;
        GTable1.Index                 :=GTable1.Index+INT#1;
    END_IF;

    // 执行Unlock指令
    Unlock(USINT#1);
    WriteReq:=FALSE;

END_IF;
```

程序P2

LD

内部变量	名称	数据类型	初始值	注释
	ReadReq	BOOL	FALSE	读取请求
	OutDat	LREAL	0.0	读取数据

外部变量	名称	数据类型	注释
	GTable1	USERTABLE	记录保存结构体



ST

内部变量	名称	数据类型	初始值	注释
	ReadReq	BOOL	FALSE	读取请求
	OutDat	LREAL	0.0	读取数据

外部变量	名称	数据类型	注释
	GTable1	USERTABLE	记录保存结构体

```
// 读取请求检测
IF (ReadReq=TRUE) THEN

    // 执行Lock指令
    Lock(USINT#1);

    IF (GTable1.Index>INT#0) THEN
        GTable1.Index :=GTable1.Index-INT#1;
        OutDat        :=GTable1.Record[GTable1.Index];
    END_IF;

    // 执行Unlock指令
    Unlock(USINT#1);
    ReadReq:=FALSE;

END_IF;
```

ActEventTask

启动事件任务。

指令	名称	FB/FUN	图形表现	ST表现
ActEventTask	事件任务启动	FUN	(@)ActEventTask EN ENO TaskName Out	ActEventTask(TaskName);

变量

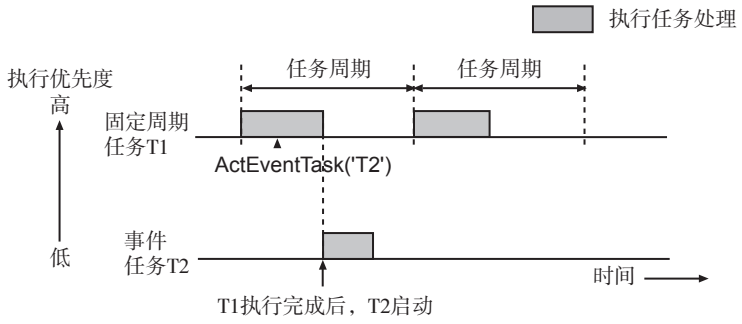
	名称	输入/输出	内容	有效范围	单位	初始值
TaskName	任务名称	输入	待启动的事件任务的任务名称	最大64字节(63个半角英数字字符+结尾NULL字符)	–	''
Out	返回值	输出	TRUE：执行指令且无异常 FALSE：不执行指令或有异常	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
TaskName																				○
Out	○																			

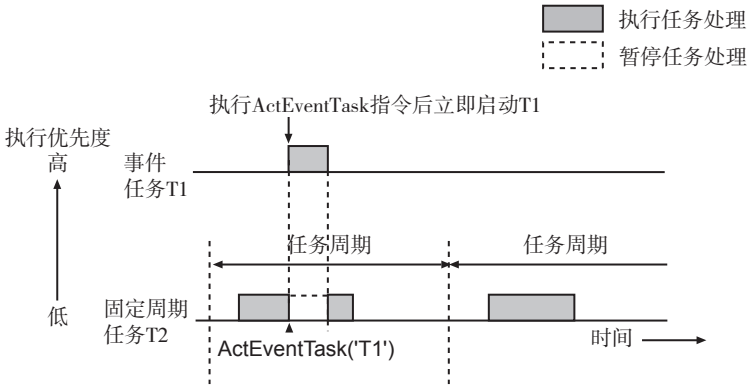
功能

启动1次任务名称 “TaskName”的事件任务。事件任务按照任务的执行优先度动作。

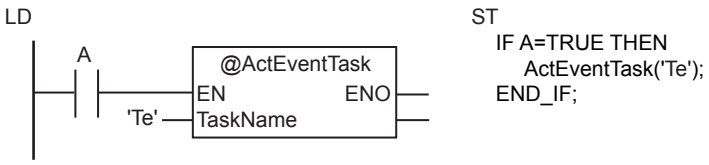
启动执行优先度低于执行本指令的任务的事件任务时，执行本指令的任务执行完成后，启动事件任务。例如，假设事件任务T2的执行优先度低于固定周期任务T1。T1指定T2，执行ActEventTask指令时，T1执行完成后启动T2。



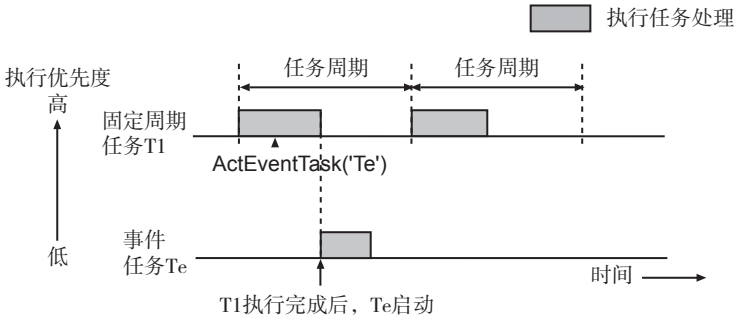
启动执行优先度高于执行本指令的任务的事件任务时，执行本指令的任务暂停执行后，启动事件任务。例如，假设固定周期任务T2的执行优先度低于事件任务T1。T2指定T1，执行ActEventTask指令时，暂停执行T2后启动T1。



示例如下所示。变量A的值为TRUE时，启动事件任务 ‘Te’。



假设已将含有该记述的程序分配至固定周期任务T1，Te的执行优先度低于T1.此时，如果通过T1执行本指令，T1执行完成后启动Te。



相关的系统定义变量

变量名称	名称	数据类型	内容
_**_Active*1	任务执行中标志	BOOL	任务的执行状态。*2 TRUE：执行中 FALSE：停止中

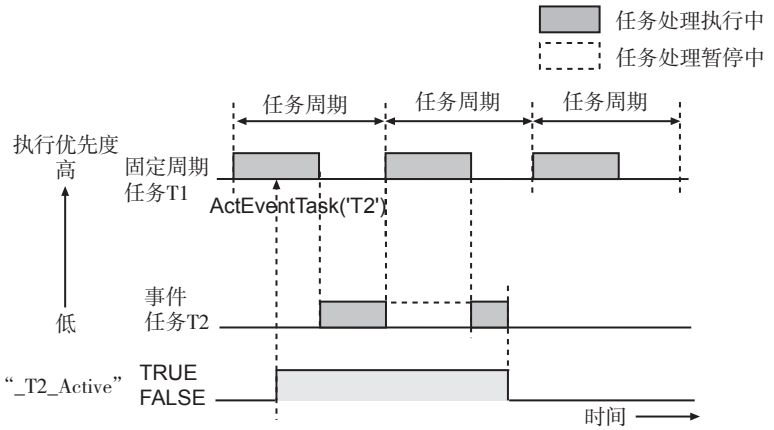
*1 **为任务名称。

*2 详情请参阅 “NJ/NX系列 CPU单元 用户手册 软件篇(SBCA-359)” 或 “NY系列 工业用平板电脑/工业用台式电脑 用户手册 软件篇(SBCA-436)”。

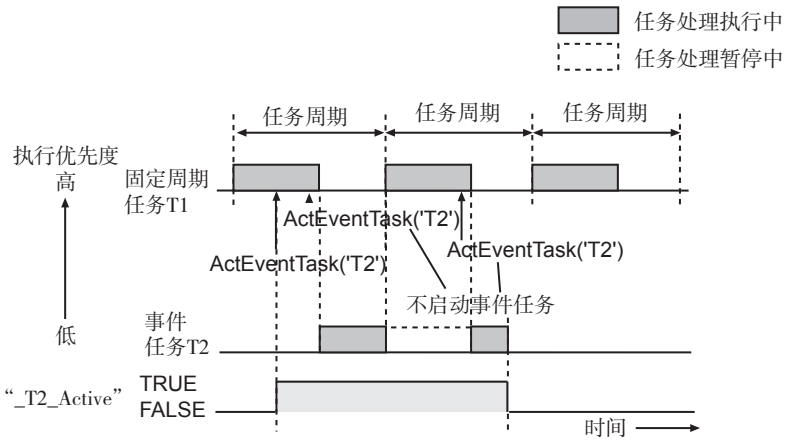
参考

系统定义变量 “_**_Active”的动作

- 执行本指令时，指定的事件任务的系统定义变量 “_**_Active”变为TRUE。事件任务执行完成时，变为FALSE。例如，假设事件任务T2的执行优先度低于固定周期任务T1。T1指定T2，执行ActEventTask指令时，系统定义变量 “_T2_Active”的变化如下图所示。



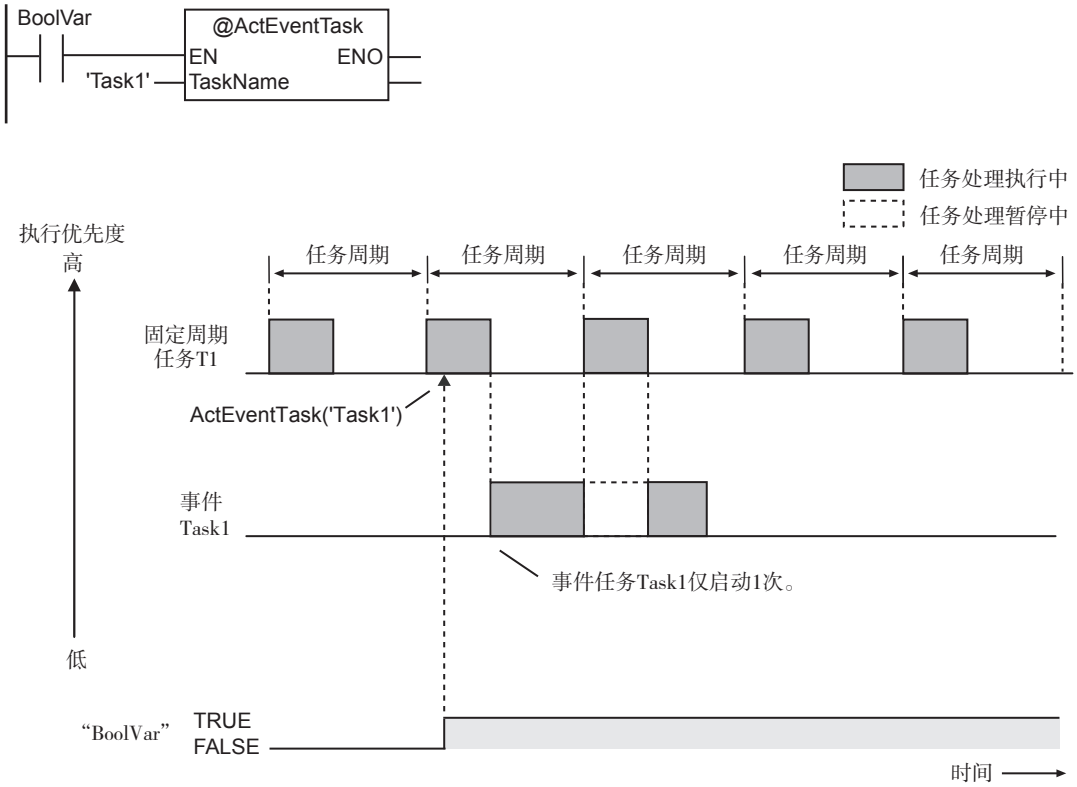
- 事件任务的系统定义变量 “_**_Active”为TRUE期间，即使执行本指令，事件任务也不会启动。



事件任务仅启动1次和反复启动

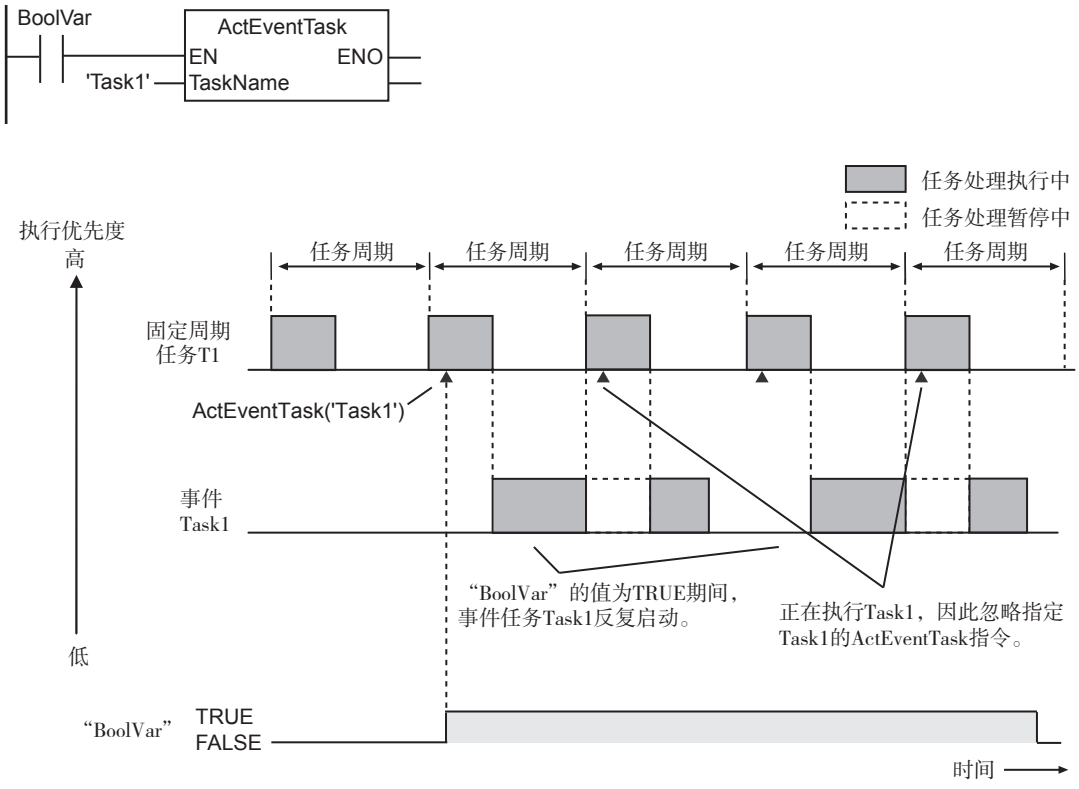
如下所示，如需在指定变量的值发生变化时仅启动 1 次事件任务或在变量为指定值期间反复启动事件任务，请改写用户程序。

例1)变量的值发生变化时，仅启动1次事件任务时
如下图所示，如果向本指令附加输入上升沿微分的动作选项，则BOOL型变量BoolVar的值从FALSE变为TRUE时，事件任务Task1仅启动1次。



例2)变量为指定值期间，反复启动事件任务时

如下图所示，如果不向本指令附加输入上升沿微分的动作选项，则 BOOL 型变量 BoolVar 的值为 TRUE 期间，会反复启动事件任务Task1。正在执行Task1时，即使执行已指定Task1的本指令，也会忽略。



使用注意事项

- 如需缩短指令执行时间，请仅在需要启动事件任务时执行本指令。系统定义变量“_**_Active”为TRUE期间执行本指令时，即使不启动事件任务，也需执行指令的时间。
- “TaskName”的事件任务不存在时，会发生异常。“ENO”为FALSE。

版本相关信息

本指令可用于Ver.1.03以上的CPU单元和Ver.1.04以上的Sysmac Studio。

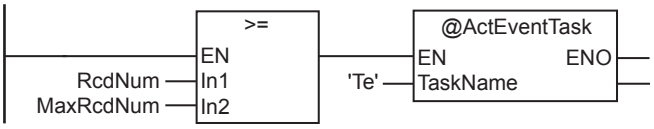
示例程序

● 变量值与指定条件一致时启动事件任务的示例

变量RcdNum从小于变量MaxRcdNum变为大于MaxRcdNum时，仅启动1次事件任务Te。

LD

名称	数据类型	初始值
RcdNum	INT	0
MaxRcdNum	INT	100



ST

名称	数据类型	初始值
RcdNum	INT	0
MaxRcdNum	INT	100
met	BOOL	FALSE

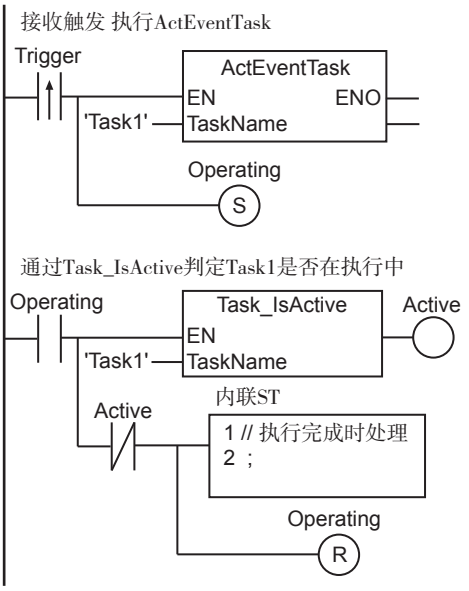
```
IF (RcdNum>=MaxRcdNum) THEN
    IF (met=FALSE) THEN
      ActEventTask('Te');
      met:=TRUE;
    END_IF;
ELSE
    met:=FALSE;
END_IF;
```

● 确认事件任务执行完成后，进行以下处理的示例

变量Trigger处于上升沿时，启动事件任务Task1。使用Task_IsActive指令，确认Task1的执行完成。

LD

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
Operating	BOOL	FALSE	事件任务执行确认中
Active	BOOL	FALSE	事件任务执行中



ST

名称	数据类型	初始值	注释
Trigger	BOOL	FALSE	执行条件
LastTrigger	BOOL	FALSE	上个任务周期的Trigger的值
Operating	BOOL	FALSE	事件任务执行确认中
Active	BOOL	FALSE	事件任务执行中

```
// 在Trigger的上升沿启动时序
IF ( (Trigger=TRUE) AND (LastTrigger=FALSE) ) THEN
    ActEventTask( ‘Task1’);     // 事件任务Task1启动
    Operating:=TRUE;
END_IF;
LastTrigger:=Trigger;

// 判定Task1是否在执行中
IF (Operating=TRUE) THEN
    Active:=Task_IsActive('Task1');

    IF (Active=FALSE) THEN       // Task1执行完成
        Operating:=FALSE;
    END_IF;

END_IF;
```

Get**Clk

输出指定周期的时钟脉冲。

指令	名称	FB/FUN	图形表现	ST表现
Get**Clk	时钟脉冲获取组	FUN	Get**Clk EN — Out **为100us、1ms、10ms、20ms、100ms、1s、1min中的任意一个	Out:=Get**Clk( ); **为100us、1ms、10ms、20ms、100ms、1s、1min的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	时钟脉冲	输出	时钟脉冲	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out	○																			

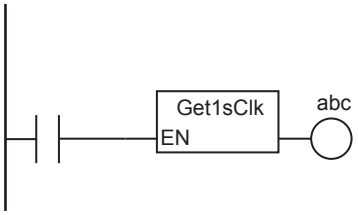
功能

输出指定周期的时钟脉冲。
时钟脉冲的周期为100us、1ms、10ms、20ms、100ms、1s、1min的任意一个。

指令名称因时钟脉冲的周期而异。例如，时钟脉冲的周期为10ms时，指令名称为Get10msClk。

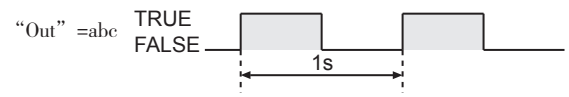
Get1sClk指令的示例如下所示。

LD



ST

```
abc:=Get1sClk();
```



使用注意事项

- 执行指令时开头的 “Out”值为TRUE还是为FALSE不确定。
- 通过梯形图程序使用本指令时，如果本指令的电路前段发生异常，则 “Out”的值为FALSE。

Get**Cnt

获取指定周期的自激计数器的值。

指令	名称	FB/FUN	图形表现	ST表现
Get**Cnt	自激加法计数器获取组	FUN	Get**Cnt EN ENO Out **为100ns、1us、1ms、10ms、100ms、1s中的任意一个	Out:=Get**Cnt( ); **为100ns、1us、1ms、10ms、100ms、1s的任意一个

变量

	名称	输入/输出	内容	有效范围	单位	初始值
Out	计数值	输出	自激计数器的值	遵从数据类型	–	–

	布尔	位串				整数								实数		时刻、持续时间、日期、字符串				
	BOOL	BYTE	WORD	DWORD	LWORD	USINT	UINT	UDINT	ULINT	SINT	INT	DINT	LINT	REAL	LREAL	TIME	DATE	TOD	DT	STRING
Out									○											

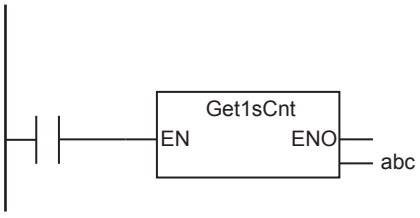
功能

获取指定周期的自激计数器的值。
自激计数器是以固定周期进行计数的计数器，“Out”为计数值。计数器的周期为100ns、1us、1ms、10ms、100ms、1s的任意一个。

指令名称因计数器的周期而异。例如，计数器的周期为10ms时，指令名称为Get10msCnt。

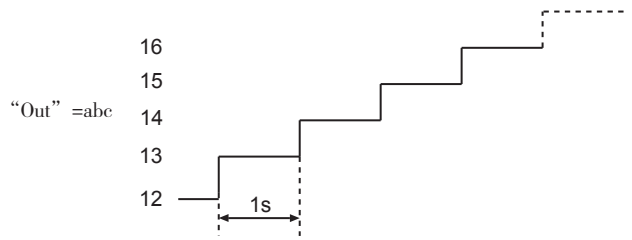
Get1sCnt指令的示例如下所示。

LD



ST

```
abc:=Get1sCnt();
```



使用注意事项

- 自激计数器在接通电源的同时开始计数。超过ULINT型的有效范围18、446、744、073、709、551、615时，返回0，继续计数。
- 本指令仅获取自激计数器的值，并非归零。
- “Out”的值从何时开始不确定。并非必定从0开始。

A

附录

A–1 可通过ErrorID确认的错误代码一览

执行指令时发生的异常根据其内容分配错误代码。使用带输出变量的错误代码 “ErrorID”的指令时，可使用错误代码进行异常处理的编程。

带“ErrorID”的指令和该指令可能导致的错误代码一览如下所示。错误代码的内容请参阅 “错误代码的详情(P.A–41)”。

参考

不带 “ErrorID”的指令的异常可通过事件日志功能作为事件进行确认。

种类	指令	名称	错误代码	错误名称
模拟控制指令	PIDAT	带自动调谐的PID运算	16#0400	超过输入值范围
			16#0401	数值关系错误
	PIDAT_HeatCool	带自动调谐的加热冷却PID运算	16#0400	超过输入值范围
			16#0401	数值关系错误
	AC_StepProgram	步程序	16#0400	超过输入值范围
系统控制指令	ResetPLCError	PLC异常解除	–	–
	ResetCJBError	I/O总线异常解除	16#0400	超过输入值范围
			16#040D	指定单元错误
	ResetMCError	运动控制异常解除	–	–
	ResetECError	EtherCAT异常解除	16#041A	多重启动指令
	GetNXUnitError	NX单元异常状态获取	16#0400	超过输入值范围
			16#041A	多重启动指令
			16#2C00	NX信息异常
			16#2C02	NX信息超时
	ResetUnit	单元重启	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#040F	单元重启失败
	RestartNXUnit	NX单元重启	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#2C00	NX信息异常
			16#2C01	NX信息资源超限
			16#2C02	NX信息超时
			16#2C05	NX信息网络异常(EtherCAT)
			16#2C06	指定单元外部重启已执行
			16#2C07	非指令对象单元指定
	NX_ChangeWriteMode	NX单元写入模式变更	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#2C00	NX信息异常
			16#2C01	NX信息资源超限
			16#2C02	NX信息超时
			16#2C05	NX信息网络异常(EtherCAT)
			16#2C07	非指令对象单元指定

种类	指令	名称	错误代码	错误名称
系统控制指令	NX_SaveParam	NX单元参数保存	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#2C00	NX信息异常
			16#2C01	NX信息资源超限
			16#2C02	NX信息超时
	NX_ReadTotalPowerOnTime	NX单元累计通电时间读取	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#2C00	NX信息异常
			16#2C01	NX信息资源超限
			16#2C02	NX信息超时
			16#2C08	累计通电时间记录错误
EtherCAT通信指令	EC_CoESDOWrite	CoE SDO写入	16#0400	超过输入值范围
			16#1800	EtherCAT通信错误
			16#1801	EtherCAT从站不存在
			16#1802	EtherCAT超时
			16#1804	SDO中止错误
			16#1808	通信资源超限
	EC_CoESDORead	CoE SDO读取	16#0400	超过输入值范围
			16#1800	EtherCAT通信错误
			16#1801	EtherCAT从站不存在
			16#1802	EtherCAT超时
			16#1803	接收缓存溢出
			16#1804	SDO中止错误
			16#1808	通信资源超限
	EC_StartMon	EtherCAT分组监控功能启动	16#1805	分组监控保存中
			16#1807	分组监控功能运行中
			16#1808	通信资源超限
	EC_StopMon	EtherCAT分组监控功能停止	16#1806	分组监控功能未启动
			16#1808	通信资源超限
	EC_SaveMon	EtherCAT分组保存	16#1805	分组监控保存中
			16#1807	分组监控功能运行中
			16#1808	通信资源超限
	EC_CopyMon	EtherCAT分组传送	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1403	指定文件错误
			16#1404	超过最大文件和目录数
			16#1405	文件访问中
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
			16#1808	通信资源超限
	EC_DisconnectSlave	EtherCAT从站脱离	16#1800	EtherCAT通信错误
			16#1801	EtherCAT从站不存在
			16#1808	通信资源超限
	EC_ConnectSlave	重新加入EtherCAT从站	16#1800	EtherCAT通信错误
			16#1801	EtherCAT从站不存在
			16#1808	通信资源超限

种类	指令	名称	错误代码	错误名称
EtherCAT通信指令	EC_ChangeEnableSetting	EtherCAT从站有效/无效切换	16#1800	EtherCAT通信错误
			16#1801	EtherCAT从站不存在
			16#1808	通信资源超限
	NX_WriteObj	NX对象写入	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#041B	数据容量超限
			16#2C00	NX信息异常
			16#2C01	NX信息资源超限
			16#2C02	NX信息超时
			16#2C03	NX信息长度错误
	NX_ReadObj	NX对象读取	16#0400	超过输入值范围
			16#0410	字符串格式异常
			16#0419	数据类型错误
			16#041C	数据大小不一致
			16#2C00	NX信息异常
			16#2C01	NX信息资源超限
			16#2C02	NX信息超时
IO-Link通信指令	IOL_ReadObj	IO-Link设备对象读取	16#0400	超过输入值范围
			16#0410	字符串格式异常
			16#0419	数据类型错误
			16#041C	数据大小不一致
			16#4800	接收到设备错误
			16#4801	指定单元不存在
			16#4802	信息处理数超限
			16#4803	指定单元状态异常
			16#4804	同时执行数超限
			16#4805	通信超时
			16#4806	模式错误
			16#4807	I/O电源OFF状态
			16#4808	发生核查异常
	IOL_WriteObj	IO-Link设备对象写入	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#041B	数据容量超限
			16#4800	接收到设备错误
			16#4801	指定单元不存在
			16#4802	信息处理数超限
			16#4803	指定单元状态异常
			16#4804	同时执行数超限
			16#4805	通信超时
			16#4806	模式错误
			16#4807	I/O电源OFF状态
			16#4808	发生核查异常

种类	指令	名称	错误代码	错误名称
EtherNet/IP通信指令	CIPOpen	CIPClass3(Large_Forward_Open)连接建立	16#0400	超过输入值范围
			16#1C00	Explicit异常
			16#1C01	根路径错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#1C05	Class3连接建立失败
			16#2000	本机IP地址设定错误
			16#2004	本机IP地址未确定
	CIPOpenWithDataSize	CIPClass3连接建立(带大小指定)	16#0400	超过输入值范围
			16#1C00	Explicit异常
			16#1C01	根路径错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#1C05	Class3连接建立失败
			16#2000	本机IP地址设定错误
			16#2004	本机IP地址未确定
	CIPRead	变量读取(Class3 Explicit信息)	16#0400	超过输入值范围
			16#0407	区域超限
			16#0419	数据类型错误
			16#1C00	Explicit异常
			16#1C02	CIP句柄错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#1C06	CIP通信数据大小超限
	CIPWrite	变量写入(Class3 Explicit信息)	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#0419	数据类型错误
			16#1C00	Explicit异常
			16#1C02	CIP句柄错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#1C06	CIP通信数据大小超限
	CIPSend	任意Explicit信息发送(Class3)	16#0400	超过输入值范围
			16#0401	数值关系错误
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#0419	数据类型错误
			16#1C00	Explicit异常
			16#1C02	CIP句柄错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#1C06	CIP通信数据大小超限
	CIPClose	CIP Class3连接的切断	16#1C02	CIP句柄错误
			16#1C03	CIP通信资源超限

种类	指令	名称	错误代码	错误名称
EtherNet/IP通信指令	CIPUCMMRead	变量读取(UCMM Explicit信息)	16#0400	超过输入值范围
			16#0407	区域超限
			16#0419	数据类型错误
			16#1C00	Explicit异常
			16#1C01	根路径错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#2000	本机IP地址设定错误
			16#2004	本机IP地址未确定
	CIPUCMMWrite	变量写入(UCMM Explicit信息)	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0419	数据类型错误
			16#1C00	Explicit异常
			16#1C01	根路径错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#2000	本机IP地址设定错误
			16#2004	本机IP地址未确定
	CIPUCMMSend	任意Explicit信息发送(UCMM)	16#0400	超过输入值范围
			16#0401	数值关系错误
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#0419	数据类型错误
			16#1C00	Explicit异常
			16#1C01	根路径错误
			16#1C03	CIP通信资源超限
			16#1C04	CIP超时
			16#2000	本机IP地址设定错误
			16#2004	本机IP地址未确定
	SktUDPCreate	UDP Socket建立	16#0400	超过输入值范围
			16#2000	本机IP地址设定错误
			16#2001	无法使用TCP/UDP端口
			16#2003	Socket状态异常
			16#2004	本机IP地址未确定
			16#2008	Socket通信资源超限
	SktUDPRcv	UDP Socket接收	16#0400	超过输入值范围
			16#0407	区域超限
			16#0419	数据类型错误
			16#2003	Socket状态异常
			16#2006	Socket超时
			16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	SktUDPSend	UDP Socket发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0419	数据类型错误
			16#2002	地址解决失败
			16#2003	Socket状态异常
			16#2007	Socket句柄错误
			16#2008	Socket通信资源超限

种类	指令	名称	错误代码	错误名称
EtherNet/IP通信指令	SktTCPAccept	TCP Socket接受	16#0400	超过输入值范围
			16#2000	本机IP地址设定错误
			16#2001	无法使用TCP/UDP端口
			16#2002	地址解决失败
			16#2003	Socket状态异常
			16#2004	本机IP地址未确定
			16#2006	Socket超时
			16#2008	Socket通信资源超限
	SktTCPConnect	TCP Socket连接	16#0400	超过输入值范围
			16#2000	本机IP地址设定错误
			16#2001	无法使用TCP/UDP端口
			16#2002	地址解决失败
			16#2003	Socket状态异常
			16#2004	本机IP地址未确定
			16#2006	Socket超时
			16#2008	Socket通信资源超限
	SktTCPRcv	TCP Socket接收	16#0400	超过输入值范围
			16#0407	区域超限
			16#0419	数据类型错误
			16#2003	Socket状态异常
			16#2006	Socket超时
			16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	SktTCPSend	TCP Socket发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0419	数据类型错误
			16#2003	Socket状态异常
			16#2006	Socket超时
			16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	SktGetTCPStatus	TCP Socket的状态读取	16#2003	Socket状态异常
			16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	SktClose	TCP/UDP Socket闭合	16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	SktClearBuf	TCP/UDP Socket接收缓存清除	16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	SktSetOption	TCP Socket选项设定	16#0400	超过输入值范围
			16#0419	数据类型错误
			16#2003	Socket状态异常
			16#2007	Socket句柄错误
			16#2008	Socket通信资源超限
	ChangeIPAdr	IP地址变更	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#2400	无执行权限
	ChangeFTPAccount	FTP账号变更	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#2400	无执行权限
	ChangeNTPServerAdr	NTP服务器地址变更	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#2400	无执行权限

种类	指令	名称	错误代码	错误名称
EtherNet/IP通信指令	FTPGetFileList	FTP服务器的文件列表获取	16#0400	超过输入值范围
			16#2403	FTP客户端执行数超限
			16#2405	指定目录错误(FTP)
			16#2406	FTP服务器连接失败
			16#2407	连接目标FTP服务器执行失败
	FTPGetFile	从FTP服务器下载文件	16#0400	超过输入值范围
			16#2403	FTP客户端执行数超限
			16#2404	文件数超限
			16#2405	指定目录错误(FTP)
			16#2406	FTP服务器连接失败
			16#2407	连接目标FTP服务器执行失败
			16#2408	SD存储卡访问失败(FTP)
			16#2409	指定文件不存在
			16#240A	禁止覆盖指定文件
			16#240C	指定文件访问失败
	FTPPutFile	文件上传至FTP服务器	16#0400	超过输入值范围
			16#2403	FTP客户端执行数超限
			16#2404	文件数超限
			16#2405	指定目录错误(FTP)
			16#2406	FTP服务器连接失败
			16#2407	连接目标FTP服务器执行失败
			16#2408	SD存储卡访问失败(FTP)
			16#2409	指定文件不存在
			16#240A	禁止覆盖指定文件
			16#240B	指定文件删除失败
			16#240C	指定文件访问失败
	FTPRemoveFile	FTP服务器的文件删除	16#0400	超过输入值范围
			16#2403	FTP客户端执行数超限
			16#2404	文件数超限
			16#2405	指定目录错误(FTP)
			16#2406	FTP服务器连接失败
			16#2407	连接目标FTP服务器执行失败
			16#2409	指定文件不存在
	FTPRemoveDir	FTP服务器的目录删除	16#0400	超过输入值范围
			16#2405	指定目录错误(FTP)
			16#2406	FTP服务器连接失败
			16#2407	连接目标FTP服务器执行失败

种类	指令	名称	错误代码	错误名称
串行通信指令	ExecPMCR	协议宏	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#040D	指定单元错误
			16#0413	存储器非地址指定
			16#0419	数据类型错误
			16#0C00	串行通信超时
			16#0800	FINS异常
			16#0801	无法使用FINS端口
	SerialSend	串行通信单元 串行端口输出	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#0C00	串行通信超时
			16#0800	FINS异常
			16#0801	无法使用FINS端口
	SerialRcv	串行通信单元 串行端口输入	16#0400	超过输入值范围
			16#0407	区域超限
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#0C00	串行通信超时
			16#0800	FINS异常
			16#0801	无法使用FINS端口
	SerialRcvNoClear	串行通信单元 不删除串行端口输入接收缓存	16#0400	超过输入值范围
			16#0407	区域超限
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#0C00	串行通信超时
			16#0800	FINS异常
			16#0801	无法使用FINS端口
	SendCmd	指令发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#0419	数据类型错误
			16#0800	FINS异常
			16#0801	无法使用FINS端口
	NX_SerialSend	无协议数据的发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0C	对非对象端口的指令执行

种类	指令	名称	错误代码	错误名称
串行通信指令	NX_SerialRcv	无协议数据的接收	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#041D	同时执行指令资源超限
			16#0C03	接收缓存已满
			16#0C04	端口多重启动
			16#0C05	奇偶校验错误
			16#0C06	结构错误
			16#0C07	超程错误
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
	NX_ModbusRtuCmd	Modbus-RTU 通用指令发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#0407	区域超限
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C05	奇偶校验错误
			16#0C06	结构错误
			16#0C07	超程错误
			16#0C08	CRC不一致
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
			16#0C10	Modbus例外响应
			16#0C11	Modbus响应错误
	NX_ModbusRtuRead	Modbus-RTU Read 指令发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C05	奇偶校验错误
			16#0C06	结构错误
			16#0C07	超程错误
			16#0C08	CRC不一致
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
			16#0C10	Modbus例外响应
			16#0C11	Modbus响应错误

种类	指令	名称	错误代码	错误名称
串行通信指令	NX_ModbusRtuWrite	Modbus-RTU Write 指令发送	16#0400	超过输入值范围
			16#0406	超过区域范围指定
			16#040D	指定单元错误
			16#0419	数据类型错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C05	奇偶校验错误
			16#0C06	结构错误
			16#0C07	超程错误
			16#0C08	CRC不一致
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
			16#0C10	Modbus例外响应
			16#0C11	Modbus响应错误
	NX_SerialSigCtl	串行控制信号的ON/OFF切换	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
	NX_SerialSigRead	串行控制信号的读取	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
	NX_SerialStatusRead	串行端口状态的读取	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
	NX_SerialBufClear	缓存清除	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行

种类	指令	名称	错误代码	错误名称
串行通信指令	NX_SerialStartMon	串行线路监控的开始	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
	NX_SerialStopMon	串行线路监控的停止	16#0400	超过输入值范围
			16#040D	指定单元错误
			16#041D	同时执行指令资源超限
			16#0C04	端口多重启动
			16#0C0B	串行通信超时
			16#0C0C	对非对象端口的指令执行
SD存储卡指令	FileWriteVar	变量文件写入	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1403	指定文件错误
			16#1404	超过最大文件和目录数
			16#1405	文件访问中
			16#1409	含有相同文件名
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
	FileReadVar	变量文件读取	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1403	指定文件错误
			16#1405	文件访问中
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
	FileOpen	文件打开	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1403	指定文件错误
			16#1404	超过最大文件和目录数
			16#1405	文件访问中
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
	FileClose	文件关闭	16#1400	无法使用SD存储卡
			16#1403	指定文件错误
			16#1405	文件访问中
			16#140E	SD存储卡访问失败

种类	指令	名称	错误代码	错误名称
SD存储卡指令	FileSeek	文件查找	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1403	指定文件错误
			16#1405	文件访问中
			16#1407	超过偏置范围
			16#140E	SD存储卡访问失败
	FileRead	文件读取	16#0406	超过区域范围指定
			16#0419	数据类型错误
			16#1400	无法使用SD存储卡
			16#1403	指定文件错误
			16#1405	文件访问中
			16#1406	打开模式不一致
			16#140E	SD存储卡访问失败
	FileWrite	文件写入	16#0406	超过区域范围指定
			16#0419	数据类型错误
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1403	指定文件错误
			16#1405	文件访问中
			16#1406	打开模式不一致
			16#140E	SD存储卡访问失败
	FileGets	字符串读取	16#1400	无法使用SD存储卡
			16#1403	指定文件错误
			16#1405	文件访问中
			16#1406	打开模式不一致
			16#140E	SD存储卡访问失败
	FilePuts	字符串写入	16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1403	指定文件错误
			16#1405	文件访问中
			16#1406	打开模式不一致
			16#140E	SD存储卡访问失败
	FileCopy	文件复制	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1403	指定文件错误
			16#1404	超过最大文件和目录数
			16#1405	文件访问中
			16#1409	含有相同文件名
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败

种类	指令	名称	错误代码	错误名称
SD存储卡指令	FileRemove	文件删除	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1403	指定文件错误
			16#1405	文件访问中
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
	FileRename	文件名变更	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1403	指定文件错误
			16#1404	超过最大文件和目录数
			16#1405	文件访问中
			16#1408	非空目录
			16#1409	含有相同文件名
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
	DirCreate	目录创建	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1404	超过最大文件和目录数
			16#1405	文件访问中
			16#1409	含有相同文件名
			16#140B	超过最大文件打开数
			16#140C	指定目录错误
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败
	DirRemove	目录删除	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1405	文件访问中
			16#1408	非空目录
			16#140A	禁止写入指定文件
			16#140B	超过最大文件打开数
			16#140C	指定目录错误
			16#140D	超过最大文件和目录名长度
			16#140E	SD存储卡访问失败

种类	指令	名称	错误代码	错误名称
SD存储卡指令	BackupToMemoryCard	SD存储卡备份	16#0400	超过输入值范围
			16#1400	无法使用SD存储卡
			16#1401	SD存储卡写保护
			16#1402	SD存储卡容量不足
			16#1404	超过最大文件和目录数
			16#1409	含有相同文件名
			16#140C	指定目录错误
			16#140E	SD存储卡访问失败
			16#140F	备份功能已执行
			16#1410	无法执行备份
			16#1411	单元/从站备份失败

A–2 错误代码的概要

表示指令导致的异常(事件)一览。事件代码的低4位表示指令的错误代码。各错误代码的说明请参阅对应的事件代码的说明。例如，指令的错误代码为16#0400时，请参阅54010400Hex的事件代码的说明。

表中的重要程度表示以下含义。

全：全部停止故障等级
部：部分停止故障等级
轻：轻度故障等级
监：监控信息
总：总体信息

事件代码栏( )内表示限定发生该事件的CPU单元时的CPU单元的单元版本。

如需了解各错误代码的详情，请进一步参阅 “错误代码的详情(P.A–41)”。

关于NJ/NX系列的异常(事件)的推测方法和所有的事件代码，请参阅 “NJ/NX系列故障诊断手册(SBCA–361)”。
指令的事件代码支持单元版本Ver.1.02以上的CPU单元。

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54010400Hex	超过输入值范围	指令的输入参数已超过输入变量的范围。 或已通过整数0进行除法/余数运算。	• 指令的输入参数已超过输入变量的范围。或已通过整数0进行除法/余数运算。				○		P.A–42
54010401Hex	数值关系错误	指令的输入参数值的关系未满足条件。或指令执行运算中/结果的数值未满足条件	• 输入参数值的关系未满足条件。 • 指令执行运算中/结果的数值未满足条件				○		P.A–42
54010402Hex	浮点数错误	向指令的浮点数输入参数输入了非数值	• 向指令的浮点数输入参数输入了非数值				○		P.A–43
54010403Hex	非BCD	向指令的BCD输入参数输入了非BCD数据的值	• 向指令的BCD输入参数输入了16进制的A、B、C、D、E、F				○		P.A–43
54010404Hex	带符号BCD错误	向指令的带符号BCD数据的输入参数的最高位输入了错误值	• 向指令的带符号BCD数据的输入参数的最高位数输入了错误值 •BCD格式已指定为 “_BCD0” 时，最高位为2 ~ F。 •BCD格式已指定为 “_BCD2” 时，最高位为A、B、C、D、E中的任意一个。 •BCD格式已指定为 “_BCD3” 时，最高位为B、C、D、E中的任意一个。				○		P.A–44
54010405Hex	指定位位置错误	指令指定的位位置错误	• 指令指定的位位置已超过指定数据的范围				○		P.A–44

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54010406Hex	超过区域范围指定	指令指定的数据的存储器地址和数据大小不恰当	• 指令指定的数据的存储器地址超过有效范围。或指令指定的数据大小已超过有效范围。可能是变量的数据类型和数据大小不相符				○		📖 P.A–45
54010407Hex	区域超限	指令的运算结果已超过输出参数数据区域的范围	• 数组元素数等指令的运算结果已超过输出参数数据区域的范围				○		📖 P.A–45
54010409Hex	控制器异常解除失败	在未发生控制器异常的状态下，执行了控制器异常解除指令	• 在未发生控制器异常的状态下，执行了控制器异常解除指令				○		📖 P.A–46
5401040BHex	用户异常解除失败	在未发生用户异常的状态下，执行了用户异常解除指令	• 在未发生用户异常的状态下，执行了用户异常解除指令				○		📖 P.A–46
5401040CHex	用户异常最大数量	用户异常指令下发生了超过最大数量的用户异常	• 用户异常指令下发生了超过最大数量的用户异常				○		📖 P.A–47
5401040DHex	指定单元错误	指令指定的单元不存在	• 指定了作为单元构成信息不存在的单元 • 指定了单元构成信息中存在但实体不存在的单元				○		📖 P.A–47
5401040FHex	单元重启失败	高功能单元重启失败	• 高功能单元处理中				○		📖 P.A–48
54010410Hex	字符串格式异常	指令中输入的字符串不是正确的字符串	• 将字符串转换为数值的指令的输入字符串不是表示数值或正数值的字符串 • 输入字符串未以NULL字符结尾				○		📖 P.A–48
54010411Hex	指定程序错误	指令指定的程序不存在	• 相应指令指定的程序不存在或被删除				○		📖 P.A–49
54010413Hex	存储器非地址指定	CJ单元用存储器指定的所需变量未在CJ单元用存储器中指定	• CJ单元用存储器指定的所需变量未在CJ单元用存储器中进行AT指定				○		📖 P.A–49
54010414Hex	堆栈下溢	堆栈中无数据	• 试图从无数据的堆栈读取数据				○		📖 P.A–50
54010416Hex	数组的元素数、维数错误	针对指令的数组的输入输出参数，元素数、维数超过了范围	• 针对指令的数组的输入输出参数，元素数、维数超过了范围				○		📖 P.A–50
54010417Hex	指令任务错误	指令指定的任务不存在	• 指定的任务不存在				○		📖 P.A–51
54010418Hex	不允许指定任务	不允许指定指令指定的任务	• 指定了我的任务、主要周期任务或周期任务				○		📖 P.A–51
54010419Hex	数据类型错误	已将指令无法使用的类型的数据指定至输入/输入输出	• 已将指令无法使用的类型的数据指定至输入/输入输出				○		📖 P.A–52
5401041AHex	多重启动指令	多重启动无法多重启动的指令	• 试图同时执行多个无法多重启动的指令				○		📖 P.A–52
5401041BHex (Ver.1.02以上)	数据容量超限	传输至指令的数据过大，无法处理	• 将大小超过可处理指令的容量的数据传输至指令				○		📖 P.A–53
5401041CHex (Ver.1.04以上)	数据大小不一致	指令的输入/输入输出指定的数据与对象参数的大小不一致	• 将大小与对象参数的大小不一致的数据指定至指令的输入/输入输出				○		📖 P.A–53

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
5401041DHex (Ver.1.05以上)	同时执行指令资源超限	超过可同时执行的相应指令组的资源执行了指令	• 超过可同时执行的个数执行了相应指令。				○		🕮 P.A-54
54010800Hex	FINS异常	收发FINS指令时发生了异常	• 收发FINS指令时发生了异常				○		🕮 P.A-54
54010801Hex	无法使用FINS端口	正在使用FINS端口	• 正在使用FINS端口				○		🕮 P.A-55
54010C00Hex	串行通信模式错误	串行通信单元未处于执行指令所需的串行通信模式	• 串行通信单元的串行通信端口未设定至指令所需的模式				○		🕮 P.A-55
54010C03Hex (Ver.1.11 以上)	接收缓存已满	进入了接收缓存已满状态	• 接收缓存已满				○		🕮 P.A-56
54010C04Hex (Ver.1.11以上)	端口多重启动	执行了不能同时执行的串行通信指令	• 在不能同时执行的指令执行过程中启动了相应指令				○		🕮 P.A-56
54010C05Hex (Ver.1.11以上)	奇偶校验错误	接收数据发生了奇偶校验错误	• 通信设定、传送速度的设定与配对设备不一致 • 干扰				○		🕮 P.A-57
54010C06Hex (Ver.1.11以上)	结构错误	接收数据发生了结构错误	• 通信设定、传送速度的设定与配对设备不一致 • 干扰				○		🕮 P.A-57
54010C07Hex (Ver.1.11以上)	超程错误	接收数据发生了超程错误	• 传送速度过快，因此在接收处理中接收了下一个数据				○		🕮 P.A-58
54010C08Hex (Ver.1.11以上)	CRC不一致	接收数据的CRC不一致	• 接收了错误的信息 • 干扰				○		🕮 P.A-58
54010C0BHex (Ver.1.11以上)	串行通信超时	串行通信发生了超时	• 与配对设备之间未进行接线 • 配对设备未接通电源 • 通信设定、传送速度的设定与配对设备不一致 • 干扰				○		🕮 P.A-59
54010C0CHex (Ver.1.11以上)	对非对象端口的指令执行	对非对象端口执行了指令	• 对非对象端口执行了指令				○		🕮 P.A-59
54010C0DHex (Ver.1.13以上)	CIF单元初始化	CIF单元被初始化，因此CIF单元中缓存的收发数据消失	• CIF单元被初始化				○		🕮 P.A-60
54010C10Hex (Ver.1.11以上)	Modbus例外响应	Modbus的从站返回了例外代码	• Modbus从站检测到异常				○		🕮 P.A-60
54010C11Hex (Ver.1.11以上)	Modbus响应错误	Modebus从站返回非预期响应	• Modebus从站返回的响应的Function Code或数据大小错误				○		🕮 P.A-61
54011400Hex	无法使用SD存储卡	执行指令时SD存储卡访问失败	• SD存储卡未安装或未正确插入 • SD存储卡损坏 • SD存储卡插槽损坏				○		🕮 P.A-61
54011401Hex	SD存储卡写保护	执行指令时试图向写保护的SD存储卡进行写入	• 试图向写保护的SD存储卡进行写入				○		🕮 P.A-62
54011402Hex	SD存储卡容量不足	向正在执行指令的SD存储卡进行写入时，发生了SD存储卡的容量不足	• 发生了SD存储卡的容量不足				○		🕮 P.A-62
54011403Hex	指定文件错误	指令指定的文件不存在。或指定的文件损坏。	• 指定的文件不存在 • 指定的文件损坏 • 接触不良等导致无法正常访问SD卡				○		🕮 P.A-63

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54011404Hex	超过最大文件和目录数	创建正在执行指令的文件/目录时，超过了最大文件/目录数	• 超过了最大文件和目录数				○		📖 P.A–63
54011405Hex	文件访问中	正在使用指令指定的文件，因此无法访问	• 试图以其他指令向正在通过其他指令访问的文件进行读写				○		📖 P.A–64
54011406Hex	打开模式不一致	指令下的文件操作与文件的打开模式不一致	• 文件打开指令指定的文件的打开模式与之后通过SD存储卡用指令进行的文件操作不匹配				○		📖 P.A–64
54011407Hex	超过偏置范围	无法访问指令指定的偏置指定的地址	• 试图超过文件大小进行访问				○		📖 P.A–65
54011408Hex	非空目录	试图执行目录删除指令或变更目录名，但目录内容不是空的	• 试图执行目录删除指令，但目录内容不是空的 • 试图变更目录名，但目录中存在目录				○		📖 P.A–65
54011409Hex	含有相同文件名	存在与指令指定的文件同名的文件，因此无法执行指令	• 已存在与指令指定的创建文件同名的文件				○		📖 P.A–66
5401140AHex	禁止写入指定文件	执行指令时试图向写保护的文件或目录进行写入	• 指令指定的写入文件或目录已设置写保护				○		📖 P.A–66
5401140BHex	超过最大文件打开数	打开正在执行指令的文件时，已打开的文件数超过了最大数量	• 打开正在执行指令的文件时，已打开的文件数超过了最大数量				○		📖 P.A–67
5401140CHex	指定目录错误	指令指定的目录不存在	• 指令指定的目录不存在				○		📖 P.A–67
5401140DHex	超过最大文件和目录名长度	指令指定的文件名或目录名过长	• 指令指定的创建文件名或创建目录名过长				○		📖 P.A–68
5401140EHex	SD存储卡访问失败	SD存储卡访问失败	• SD存储卡损坏 • SD存储卡插槽损坏				○		📖 P.A–68
5401140FHex (Ver.1.08以上)	备份功能已执行	其他备份功能正在动作	• 其他备份功能正在动作				○		📖 P.A–69
54011410Hex (Ver.1.08以上)	无法执行备份	正在执行其他功能，因此未能执行备份	• 在线编辑的执行过程中执行了指令 • 凸轮表保存指令的执行过程中执行了指令 • CPU单元名称更新的执行过程中执行了指令				○		📖 P.A–69
54011411Hex (Ver.1.08以上)	单元/从站备份失败	单元/从站备份失败	• 单元/从站备份失败				○		📖 P.A–70
54011800Hex	EtherCAT通信错误	执行指令时EtherCAT网络访问失败	• EtherCAT网络未处于可执行状态				○		📖 P.A–70
54011801Hex	EtherCAT从站不存在	执行指令时对象从站访问失败	• 对象从站不存在 • 对象从站未处于可执行状态				○		📖 P.A–71
54011802Hex	EtherCAT超时	执行指令时EtherCAT从站访问超时	• 与对象从站的通信超时				○		📖 P.A–71

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54011803Hex	接收缓存溢出	执行指令时，来自EtherCAT从站的接收数据超过接收缓存	• 来自从站的接收数据大小超过接收缓存				○		📖 P.A-72
54011804Hex	SDO中止错误	执行指令时，从EtherCAT从站接收了SDO中止错误	• 取决于从站规格				○		📖 P.A-72
54011805Hex	分组监控保存中	保存EtherCAT的分组监控文件过程中，执行了分组监控操作相关指令	• 保存EtherCAT的分组监控过程中，执行了分组监控操作相关指令				○		📖 P.A-73
54011806Hex	分组监控功能未启动	停止EtherCAT的分组监控过程中，执行了分组监控停止指令	• 停止EtherCAT的分组监控功能过程中，执行了分组监控停止指令				○		📖 P.A-73
54011807Hex	分组监控功能运行中	运行EtherCAT的分组监控过程中，执行了分组监控启动指令	• 运行EtherCAT的分组监控功能过程中，再次执行了分组监控启动指令				○		📖 P.A-74
54011808Hex	通信资源超限	同时执行了超过32个的EtherCAT通信指令	• 同时执行了超过32个的EtherCAT通信指令。EtherCAT通信指令如下所示 •EC_CoESDOWrite指令 •EC_CoESDORead指令 •EC_ConnectSlave指令 •EC_DisconnectSlave指令 •EC_StartMon指令 •EC_SaveMon指令 •EC_StopMon指令 •EC_CopyMon指令				○		📖 P.A-74
54011809Hex (Ver.1.01以上)	不支持分组监控功能	无法使用分组监控功能。	• 对未配备分组监控功能的CPU单元执行了分组监控功能的指令				○		📖 P.A-75
54011C00Hex	Explicit异常	通过CIP通信指令以Explicit信息返回了错误响应代码	• 取决于异常的内容				○		📖 P.A-75
54011C01Hex	根路径错误	CIP通信指令指定的根路径格式错误	• CIP通信指令指定的根路径格式错误				○		📖 P.A-76
54011C02Hex	CIP句柄错误	CIP通信指令指定的句柄错误。	• CIP通信指令指定的句柄错误				○		📖 P.A-76
54011C03Hex	CIP通信资源超限	超过可同时执行的CIP通信指令的资源执行了指令	• 同时执行了超过32个的CIP通信指令 • 试图同时使用超过32个的句柄				○		📖 P.A-77
54011C04Hex	CIP超时	执行CIP通信指令过程中发生了超时	• 指定IP地址的设备不存在 • 指定句柄的CIP连接超时，因此已关闭 • 配对设备的电源OFF • 配对设备的通信停止 • EtherNet/IP的Ethernet电缆连接器断开 • EtherNet/IP的Ethernet电缆断线 • 干扰				○		📖 P.A-77

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54011C05Hex (Ver.1.06以上)	Class3连接建立失败	使用CIP通信指令建立Class3连接失败	• 对不支持Class3(Large_Forward_Open)的设备执行了CIPOpen指令 • 对不支持Class3(Large_Forward_Open)的设备，将数据大小设定成510字节以上后执行了CIPOpenWithDataSize指令				○		P.A-78
54011C06Hex (Ver.1.06以上)	CIP通信数据大小超限	试图使用CIP通信指令发送超出可发送数据大小的Class3 Explicit信息	• CIPRead指令、CIPWrite指令、CIPSend指令的输入变量设定的数据大小超出了CIPOpenWithDataSize指令设定的数据大小				○		P.A-78
54012000Hex	本机IP地址设定错误	在本机IP地址已发生设定错误的状态下，执行了指令	• 在本机IP地址已发生设定错误的状态下，执行了指令				○		P.A-79
54012001Hex	无法使用TCP/UDP端口	执行指令时已使用UDP或TCP端口	• 已使用UDP或TCP端口				○		P.A-79
54012002Hex	地址解析失败	指令指定主机名称的对象节点的地址解析失败	• 相应指令指定的主机名称错误 • 控制器的hosts设定/DNS设定错误 • DNS服务器的设定错误				○		P.A-80

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54012003Hex	Socket状态异常	执行Socket服务指令时的状态不正确	• SktUDPCreate指令时 输入变量 “SrcUdpPort”指定的UDP端口为以下任意一种 •已打开 •关闭处理中 • SktUDPRcv指令时 •指定的Socket接收处理中 • 指定的Socket已关闭 • SktUDPSend指令时 •指定的Socket发送处理中 • 指定的Socket已关闭 • SktTCPAccept指令时 指定的TCP端口为以下任意一种 •打开处理中 •关闭处理中 •本指令已由相同IP地址、TCP端口建立连接 • SktTCPConnect指令时 •输入变量 “SrcTcpPort”指定的TCP端口已打开 •输入变量 “DstAdr”指定的对象节点不存在 •输入变量 “DstAdr”、“DstTcpPort”指定的对象节点未处于连接等待状态 • SktTCPRcv指令时 •指定的Socket接收处理中 •指定的Socket已关闭 • SktTCPSend指令时 •指定的Socket发送处理中 • 指定的Socket已关闭 •指定Socket的发送缓存已满(连接目标电源OFF、线路切断等) • SktSetOption指令时 •指定Socket已开始收发信息 •指定了指定Socket不支持的选项类型				○		📖 P.A-81
54012004Hex	本机IP地址未确定	执行Socket服务指令时本机IP地址未确定	• BOOTP服务器的设定异常 • BOOTP服务器不存在 • 刚启动后本机IP地址未确定				○		📖 P.A-82
54012006Hex	Socket超时	Socket服务指令发生超时	• SktTCPAccept指令： 用户指定的超时时间内，无来自对象节点的连接请求 • SktTCPRcv指令、SktUDPRcv指令： 用户指定的超时时间内，无法接收来自对象节点的数据				○		📖 P.A-82
54012007Hex	Socket句柄错误	Socket服务指令指定的句柄错误	• Socket服务指令指定的句柄错误				○		📖 P.A-83
54012008Hex	Socket通信资源超限	超过可同时执行的Socket服务指令的资源执行了指令。	• 同时执行了超过32个的Socket服务指令 • 试图同时使用超过30个(Ver.1.02以下的CPU单元时为16个)的Socket句柄				○		📖 P.A-83

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54012400Hex (Ver.1.02以上)	无执行权限	在无法执行的状态下执行了变更EtherNet/IP端口设定的指令	• 内置EtherNet/IP端口重启过程中，执行了变更内置EtherNet/IP端口或CJ系列EtherNet/IP单元设定的指令 • CJ系列EtherNet/IP单元重启过程中，执行了变更该单元设定的指令 • 通过指令或CIP信息变更内置EtherNet/IP端口设定过程中，执行了变更EtherNet/IP端口或CJ系列EtherNet/IP单元设定的指令 • 通过指令或CIP信息变更CJ系列EtherNet/IP单元设定过程中，执行了变更该单元设定的指令 • 指令指定的单元编号非内置EtherNet/IP端口、CJ系列EtherNet/IP单元				○		🕮 P.A–84
54012401Hex (Ver.1.02以上)	设定反映失败	CJ系列EtherNet/IP单元无法反映变更的设定	• 执行变更CJ系列EtherNet/IP单元设定的指令过程中，该单元或内置EtherNet/IP端口重启				○		🕮 P.A–85
54012402Hex (Ver.1.02以上)	同时执行指令数超限	超过可同时执行的数量执行了控制器通信设定的指令	• 同时执行了2个以上的控制器通信设定的指令				○		🕮 P.A–85
54012403Hex (Ver.1.08以上)	FTP 客户端执行数超限	FTP客户端通信指令超过可同时执行的数量进行了执行	• 同时执行了4个以上的FTP客户端通信指令				○		🕮 P.A–86
54012404Hex (Ver.1.08以上)	文件数超限	FTP客户端通信指令的通配符指定对象文件超过了1000个	• FTP客户端通信指令以通配符指定文件名时，对象文件超过了1000个				○		🕮 P.A–86
54012405Hex (Ver.1.08以上)	指定目录错误(FTP)	FTP客户端通信指令指定的目录在控制器中不存在或指定了错误的路径	• FTP客户端通信指令指定的目录在控制器中不存在或指定了错误的路径				○		🕮 P.A–87
54012406Hex (Ver.1.08以上)	FTP服务器连接失败	FTP客户端通信指令指定的连接目标FTP服务器在网络上不存在或已停止FTP服务	• FTP客户端通信指令指定的连接目标FTP服务器在网络上不存在 • FTP客户端通信指令指定的连接目标FTP服务器已停止FTP服务				○		🕮 P.A–87
54012407Hex (Ver.1.08以上)	连接目标FTP服务器执行失败	使用FTP客户端通信指令时，连接目标FTP服务器返回了错误	• 使用FTP客户端通信指令对连接目标FTP服务器请求的处理，在连接目标FTP服务器侧执行失败				○		🕮 P.A–88
54012408Hex (Ver.1.08以上)	SD存储卡访问失败(FTP)	FTP客户端对SD存储卡的访问失败	• 未安装SD存储卡 • 执行FTP客户端通信指令的过程中拔出了SD存储卡 • SD存储卡容量不足 • SD存储卡被写保护				○		🕮 P.A–88
54012409Hex (Ver.1.08以上)	指定文件不存在	FTP客户端通信指令指定的文件在控制器中不存在	• FTP客户端通信指令指定的文件在控制器中不存在				○		🕮 P.A–89
5401240AHex (Ver.1.08以上)	禁止覆盖指定文件	FTP客户端通信指令指定了不覆盖同名文件，因此未传送	• FTP客户端通信指令的覆盖指定为“不覆盖文件”，传送目标处存在与指定文件同名的文件，因此未传送				○		🕮 P.A–89

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
5401240BHex (Ver.1.08以上)	指定文件删除失败	FTP客户端通信指令未能删除已传送文件	• FTP客户端通信指令的传送后文件删除指定为 “删除已传送文件”，但指定文件的属性为只读，因此未能删除 • FTP客户端通信指令指定的文件正被其他应用程序使用，因此未能删除				○		🕮 P.A-90
5401240CHex (Ver.1.08以上)	指定文件访问失败	FTP客户端通信指令访问文件失败，因此FTP传送失败	• FTP客户端通信指令指定的文件正被其他应用程序使用 • FTP客户端通信指令指定的写入文件或目录已设置写保护				○		🕮 P.A-90
5401240DHex (Ver.1.10以上)	IP地址设定错误	指令指定端口的IP地址设定与其他端口设定之间存在设定错误，因此未能执行指令	• 指令指定端口的网络地址与其他端口的网络地址重复 • 指令指定端口与其他端口的设定均为未使用				○		🕮 P.A-91
54012C00Hex (Ver.1.05以上)	NX信息异常	以NX信息返回了错误响应	• 取决于异常的内容				○		🕮 P.A-91
54012C01Hex (Ver.1.05以上)	NX信息资源超限	超过可同时执行的NX信息指令的资源执行了指令	• 同时执行了超过32个的NX信息指令				○		🕮 P.A-92
54012C02Hex (Ver.1.05以上)	NX信息超时	NX信息执行过程中发生了超时	• 指定的NX单元不存在 • NX信息因超时而关闭 • 对象单元的电源OFF • 对象单元的通信停止 • 通信电缆的连接器断开 • 通信电缆断线 • 干扰				○		🕮 P.A-92
54012C03Hex (Ver.1.05以上)	NX信息长度错误	NX信息的长度错误	• WriteDat或Path指定的大小过长				○		🕮 P.A-93
54012C05Hex (Ver.1.05以上)	NX信息网络异常 (EtherCAT)	NX信息线路上的EtherCAT通信发生了异常	• NX信息线路上的EtherCAT通信发生了异常				○		🕮 P.A-93
54012C06Hex (Ver.1.05以上)	指定单元外部重启已执行	执行指令时，已通过Sysmac Studio执行了重启	• 执行指令时，已通过Sysmac Studio执行了重启				○		🕮 P.A-94
54012C07Hex (Ver.1.05以上)	非指令对象单元指定	指定单元的从站节点地址连接了非相应指令对象的从站	• 指定单元的从站节点地址连接了非相应指令对象的从站				○		🕮 P.A-94
54012C08Hex (Ver.1.10以上)	累计通电时间记录错误	累计通电时间的读取失败。	• 非易失性存储器故障				○		🕮 P.A-95
54013461Hex	过程数据对象设定不足	PDO映射不正确	• 未通过运动指令进行所需的PDO映射 • 对不带支持相应指令对象的对象设备执行了相应指令 • 针对映射了欧姆龙生产的EtherCAT编码器从站GX-EC02□□□的轴，作为触发条件指定Z相(_mcEncoderMark)，启动了运动指令				○		🕮 P.A-96
54014800Hex (Ver.1.12以上)	接收到设备错误	接收到设备发出的错误响应	• 接收到设备发出的错误响应				○		🕮 P.A-97
54014801Hex (Ver.1.12以上)	指定单元不存在	指定的单元不存在。	• 指定位置未连接/安装IO-Link主站单元				○		🕮 P.A-97

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54014802Hex (Ver.1.12以上)	信息处理数超限	IO–Link主站正在处理来自其它应用程序的信息，因此无法执行指令	• IO–Link主站正在处理来自其它应用程序(指令执行或工具连接)的信息，因此无法执行指令				○		P.A–98
54014803Hex (Ver.1.12以上)	指定单元状态异常	指定单元未处于可接收信息的状态	• 指定单元未处于可接收信息的状态				○		P.A–98
54014804Hex (Ver.1.12以上)	同时执行数超限	超过了可同时执行的指令数	• 同时执行的NX信息指令和EtherCAT通信指令的总数超过了32个				○		P.A–99
54014805Hex (Ver.1.12以上)	通信超时	通信过程中发生了超时	• 信息响应时间比通信超时时间长 • EtherCAT或IO–Link的电缆断线 • 干扰 • 设备故障				○		P.A–99
54014806Hex (Ver.1.12以上)	模式错误	指定IO–Link主站的端口非IO–Link模式	• 指定IO–Link主站的端口非IO–Link模式				○		P.A–100
54014807Hex (Ver.1.12以上)	I/O电源OFF状态	未对指定IO–Link主站的端口供给I/O电源	• 未对指定IO–Link主站的端口供给I/O电源				○		P.A–100
54014808Hex (Ver.1.12以上)	发生核查异常	指定IO–Link主站的端口发生了核查异常或通信异常	• 指定IO–Link主站的端口发生了核查异常或通信异常				○		P.A–101
54015420Hex	超过电子齿轮分子设定范围	运动控制指令的输入变量“RatioNumerator”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–101
54015421Hex	超过电子齿轮分母设定范围	运动控制指令的输入变量“RatioDenominator”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–102
54015422Hex	超过目标速度设定范围	运动控制指令的输入变量“Velocity”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–102
54015423Hex	超过加速度设定范围	运动控制指令的输入变量“Acceleration”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–103
54015424Hex	超过减速度设定范围	运动控制指令的输入变量“Deceleration”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–103
54015425Hex	超过跃度设定范围	运动控制指令的输入变量“Jerk”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–104
54015427Hex	超过扭矩倾斜设定范围	运动控制指令的输入变量“TorqueRamp”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–104
54015428Hex	超过主轴系数设定范围	运动控制指令的输入变量“MasterScaling”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–105

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015429Hex	超过从轴系数设定范围	运动控制指令的输入变量“SlaveScaling”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-105
5401542AHex	超过标准速度设定范围	运动控制指令的输入变量“FeedVelocity”指定的参数超过范围	• 标准速度(输入变量“FeedVelocity”)保持为初始值(0)				○		📖 P.A-106
5401542BHex	超过缓存模式选择范围	运动控制指令的输入变量“BufferMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-106
5401542CHex	超过坐标系选择范围	运动控制指令的输入变量“CoordSystem”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-107
5401542DHex	超过圆弧插补模式选择范围	运动控制指令的输入变量“CircMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-107
5401542EHex	超过方向选择范围	运动控制指令的输入变量“Direction”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-108
5401542FHex	超过路径选择范围	运动控制指令的输入变量“PathChoice”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-108
54015430Hex	超过位置类型选择范围	运动控制指令的输入变量“ReferenceType”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-109
54015431Hex	超过移动方法选择范围	运动控制指令的输入变量“MoveMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-109
54015432Hex	超过过渡模式选择范围	运动控制指令的输入变量“TransitionMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围 • “BufferMode”下指定“_mcAborting”、“_mcBuffered”且将“TransitionMode”指定为“_mcTMCornerSuperimposed”				○		📖 P.A-110
54015433Hex	超过持续方法选择范围	变更了运动控制指令的输入变量“Continuous (Reserved)”的值	• 变更了输入变量“Continuous (Reserved)”的值				○		📖 P.A-110
54015434Hex	超过加减法运算方法选择范围	运动控制指令的输入变量“CombineMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-111
54015435Hex	超过开始同步条件指定范围	运动控制指令的输入变量“LinkOption”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-111

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015436Hex	主轴从轴相同	运动控制指令的输入变量“Master”和“Slave”指定的轴相同	• 指令的输入变量“Master”和“Slave”的参数相同				○		P.A-112
54015437Hex	主轴辅轴相同	运动控制指令的输入变量“Master”和“Auxiliary”指定的轴相同	• 指令的输入变量“Master”和“Auxiliary”的参数相同				○		P.A-112
54015438Hex	主轴/从轴轴号非升序	运动控制指令的输入变量“Master”和“Slave”指定的轴号非升序	• 指令的输入变量“ReferenceType”指定“_mcLatestCommand”时，指令的输入变量“Master”和“Slave”的参数非升序				○		P.A-113
54015439Hex	凸轮表指定错误	运动控制指令的输入变量“CamTable”指定的参数超过范围	• 指令的输入变量“CamTable”指定了非凸轮数据变量				○		P.A-113
5401543AHex	同步停止中	执行了运动控制的同步控制指令，但并非可执行条件。	• 执行了MC_CamOut(凸轮动作解除)指令，但MC_CamIn(凸轮动作开始)指令未在运行 • 执行了MC_GearOut(齿轮动作解除)指令，但MC_GearIn(齿轮动作开始)指令、MC_GearInPos(位置指定齿轮动作)指令未在运行 • 执行了MC_Phasing(主轴相对值相位补偿)指令，但MC_CamIn(凸轮动作开始)指令、MC_GearIn(齿轮动作开始)指令、MC_GearInPos(位置指定齿轮动作)指令、MC_MoveLink(梯形模式凸轮)指令未在运行				○		P.A-114
5401543BHex	无法重启运动指令	重启了无法重启的运动控制指令	• 重启了无法重启的运动控制指令				○		P.A-115
5401543CHex	无法多重启动运动指令	对相同对象(MC通用/轴/轴组)执行了多个无法同时执行的功能	• 对相同对象(MC通用/轴)执行了多个无法同时执行的功能				○		P.A-115
5401543DHex	不符合轴类型	对编码器轴执行了动作指令	• 对编码器轴执行了动作指令				○		P.A-116
5401543EHex	无法启动多轴协调动作中的指令	• 对多轴协调动作中的轴或轴组执行了动作指令 • 执行了启用轴组时无法使用的机器人指令	• 对多轴协调动作中的轴或轴组执行了动作指令 • 对轴组有效状态下的轴组执行了MC_SetKinTransform指令				○		P.A-116
5401543FHex	启动轴组无效状态下的多轴协调指令	对轴组无效状态下的轴组启动了多轴协调指令	• 对轴组无效状态下的轴组启动了多轴协调指令 • 对轴组无效状态下的轴组启动了以下指令 •MC_MoveTimeAbsolute指令 •MC_SyncLinearConveyor指令 •MC_SyncOut指令 •MC_RobotJog指令				○		P.A-117

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015440Hex	无法启用轴组	MC_GroupEnable(启用轴组)指令执行失败	• 执行MC_GroupEnable(启用轴组)指令时，构成轴中存在未处于停止状态的轴 • 执行MC_GroupEnable(启用轴组)指令时，构成轴中存在正在执行MC_TouchProbe(启用外部锁定)指令的轴				○		P.A-117
54015441Hex	无法运行(伺服OFF)轴动作指示	对伺服OFF中的轴执行了动作指令	• 对伺服OFF中的轴执行了动作指令 • 对未建立EtherCAT的过程数据通信的轴执行了通过MC_Home(原点复位)指令或MC_HomeWithParameter(参数指定原点复位)指令进行的原点预设。				○		P.A-118
54015442Hex	构成轴强制停止中错误	针对构成轴，向已执行MC_Stop(强制停止)指令的轴组执行了动作指令	• 针对构成轴，向已执行MC_Stop(强制停止)指令的轴组执行了动作指令				○		P.A-118
54015443Hex	多重启动运动指令数超限	通过缓存模式Buffered、Blending缓存的运动控制指令的缓存数超限	• 正在执行的轴指令和缓存中的轴指令的总数超过了 “2” • 正在执行的轴组指令和缓存中的轴组指令的总数超过了 “8”				○		P.A-119
54015444Hex	移动量不足	多重启动/重启定位指令时，无法执行以指定减速度或加速度指定的动作	• 如果已将 “加减速超限”设定为 “作为异常停止”，则无法在多重启动/重启定位指令时，在指定减速度/加速度的条件下在目标位置停止				○		P.A-119
54015445Hex	用于达到混合中继速度的移动量不足	用于中继速度加减速的移动量不足	• 将 “加减速超限”设定为 “作为异常停止”时，用于将当前指令加减速至中继速度的移动量不足				○		P.A-120
54015446Hex	梯形模式凸轮等速移动量不足	主轴的等速移动量小于 “0”	• MC_MoveLink(梯形模式凸轮)指令下，主轴的等速移动量小于 “0”				○		P.A-120
54015447Hex	位置指定齿轮动作目标速度不足	MC_GearInPos(位置指定齿轮动作)指令下，从轴的 “目标速度”较小，因此达不到所需速度	• MC_GearInPos(位置指定齿轮动作)指令下，输入变量 “Velocity(目标速度)”的值小于(启动指令时的主轴速度 × 齿轮比)				○		P.A-121
54015448Hex	圆弧插补起点终点相同	MC_MoveCircular2D(2轴圆弧插补)指令下，指定半径指定方式时起点和终点的位置相同。或指定通过点指定方式时起点、终点及通过点的位置相同	• MC_MoveCircular2D(2轴圆弧插补)指令下，指定了半径指定方式，起点和终点的位置相同 • MC_MoveCircular2D(2轴圆弧插补)指令下，指定了通过点指定方式，起点、终点及通过点的位置相同				○		P.A-121
54015449Hex	超过圆弧插补中心点指定位置范围	MC_MoveCircular2D(2轴圆弧插补)指令下，指定中心点指定方式时，中心点的位置指定超过容许范围	• MC_MoveCircular2D(2轴圆弧插补)指令下，指定了中心点指定方式，起点与中心点的距离、终点与中心点的距离之差超过了轴组设定的 “中心点补偿容许率”指定的容许范围				○		P.A-122

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
5401544AHex	计数模式设定导致的指令启动异常	对计数模式已设定为旋转模式的轴，执行了旋转模式下无法使用的指令	• 以旋转模式下无法使用的指令使用了计数模式已设定为旋转模式的轴				○		🕮 P.A-122
5401544CHex	超过参数选择范围	运动控制指令的输入变量“ParameterNumber”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		🕮 P.A-123
5401544DHex	超过停止方法选择范围	运动控制指令的输入变量“StopMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		🕮 P.A-123
5401544EHex	超过触发输入条件的锁定ID选择范围	运动控制指令的输入变量“TriggerInput::LatchID”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		🕮 P.A-124
5401544FHex	超过MC设定写入的设定范围	运动控制指令的输入变量“SettingValue”指定的参数超过范围	• 指令的输入参数超过输入变量的范围 • 参数指定和设定值的数据类型不一致				○		🕮 P.A-124
54015450Hex	超过触发输入条件的模式选择范围	运动控制指令的输入变量“TriggerInput::Mode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		🕮 P.A-125
54015451Hex	超过触发输入条件的驱动触发输入信号选择范围	运动控制指令的输入变量“TriggerInput::InputDrive”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		🕮 P.A-125
54015453Hex	无法重启运动指令(轴指定)	变更了重启运动控制指令时无法变更的输入变量“Axis”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-126
54015454Hex	无法重启运动指令(缓存模式选择)	变更了重启运动控制指令时无法变更的输入变量“BufferMode”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-126
54015455Hex	无法重启运动指令(方向选择)	变更了重启运动控制指令时无法变更的输入变量“Direction”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-127
54015456Hex	无法重启运动指令(重复模式)	变更了重启运动控制指令时无法变更的输入变量“Periodic”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-127
54015457Hex	无法重启运动指令(轴组指定)	变更了重启运动控制指令时无法变更的输入变量“AxesGroup”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-128

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015458Hex	无法重启运动指令(跳动设定)	变更了重启运动控制指令时无法变更的输入变量“Jerk”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-128
54015459Hex	无法重启运动指令(主轴)	变更了重启运动控制指令时无法变更的输入变量“Master”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-129
5401545AHex	无法重启运动指令(Master Offset)	变更了重启运动控制指令时无法变更的输入变量“MasterOffset”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-129
5401545BHex	无法重启运动指令(Master Scaling)	变更了重启运动控制指令时无法变更的输入变量“MasterScaling”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-130
5401545CHex	无法重启运动指令(Master Start Distance)	变更了重启运动控制指令时无法变更的输入变量“MasterStart Distance”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-130
5401545DHex	无法重启运动指令(Continuous)	变更了重启运动控制指令时无法变更的输入变量“Continuous”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-131
5401545EHex	无法重启运动指令(Move Mode)	变更了重启运动控制指令时无法变更的输入变量“MoveMode”的参数	• 变更了重启时无法变更的输入变量的参数				○		🕮 P.A-131
5401545FHex	辅轴指定错误	运动控制指令的输入变量“Auxiliary”指定的轴不存在	• 指令的输入变量 “Auxiliary”指定的轴不存在的变量				○		🕮 P.A-132
54015460Hex	轴指定错误	运动控制指令的输入变量 “Axis”指定的轴不存在	• 指令的输入变量 “Axis”指定的轴不存在的变量				○		🕮 P.A-132
54015461Hex	轴组指定错误	运动控制指令的输入变量“AxesGroup”指定的轴组不存在或不是使用轴组	• 指令的 “AxesGroup”指定的轴组不存在的变量 • 指令的 “AxesGroup”指定的轴组未设定为使用轴组				○		🕮 P.A-133
54015462Hex	主轴指定错误	运动控制指令的输入变量 “Master”指定的轴错误	• 指令的输入变量 “Master”指定的轴不存在的变量 • MC_Phasing(主轴相对值相位补偿)指令时，输入变量 “Master”指定的轴不是同步主轴 • 分配主轴和从轴的任务不同				○		🕮 P.A-133

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015463Hex	无法重启运动指令(Slave Offset)	变更了重启运动控制指令时无法变更的输入变量“SlaveOffset”的参数	• 变更了重启时无法变更的输入变量的参数				○		📖 P.A–134
54015464Hex	无法重启运动指令(Slave Scaling)	变更了重启运动控制指令时无法变更的输入变量“SlaveScaling”的参数	• 变更了重启时无法变更的输入变量的参数				○		📖 P.A–134
54015465Hex	无法重启运动指令(Start Position)	变更了重启运动控制指令时无法变更的输入变量“StartPosition”的参数	• 变更了重启时无法变更的输入变量的参数				○		📖 P.A–135
54015466Hex	原点未确定状态下的指令启动异常	原点未确定状态下执行了高速原点复位或插补指令	• 原点未确定状态下执行了高速原点复位 • 向包含原点未确定状态的构成轴执行了插补指令 • 向包含原点未确定状态逻辑轴的轴组执行了以下机器人指令 •MC_SetKinTransform指令 •MC_MoveTimeAbsolute指令 •MC_SyncLinearConveyor指令 •MC_SyncOut指令 •MC_GroupMon指令 •MC_RobotJog指令				○		📖 P.A–135
54015467Hex	无法重启运动指令(位置类型)	变更了重启运动控制指令时无法变更的输入变量“ReferenceType”的参数	• 变更了重启时无法变更的输入变量的参数				○		📖 P.A–136
54015468Hex	未使用轴指定(主轴)	运动控制指令指定的主轴为未使用轴	• 运动控制指令指定的主轴为未使用轴				○		📖 P.A–136
54015469Hex	超过起始位置设定范围	运动控制指令的输入变量“FirstPosition”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A–137
5401546AHex	超过终止位置设定范围	运动控制指令的输入变量“LastPosition”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A–137
5401546BHex	起始位置/终止位置、大小关系错误(线性模式)	运动控制指令的输入变量“LastPosition”指定的参数值小于输入变量“FirstPosition”指定的参数值	• 计数模式为线性模式时，指令的输入参数 “LastPosition”的值小于 “FirstPosition”的值				○		📖 P.A–138
5401546CHex	超过主轴同步位置设定范围	运动控制指令的输入变量“MasterSync Position”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A–138

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
5401546DHex	超过从轴同步位置设定范围	运动控制指令的输入变量 “Slave SyncPosition”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-139
5401546EHex	触发输入条件的锁定ID重复	多个指令已重复运动控制指令指定的锁定ID	• MC_TouchProbe(启用外部锁定)指令、MC_MoveLink(梯形模式凸轮)指令、MC_MoveFeed(中断标准定位)指令已同时使用相同的锁定ID • 试图通过MC_AbortTrigger(不启用外部锁定)指令中止正在以非MC_TouchProbe(启用外部锁定)指令使用的锁定				○		📖 P.A-139
5401546FHex	超过跃度超调值范围	运动控制指令的输入变量 “JerkFactor”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-140
54015470Hex	超过加减速度超调值范围	运动控制指令的输入变量 “AccFactor”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-140
54015471Hex	超过起始位置方式指定范围	运动控制指令的输入变量 “StartMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-141
54015472Hex	无法重启运动指令(起始位置方式)	变更了重启运动控制指令时无法变更的输入变量 “StartMode”的参数	• 变更了重启时无法变更的输入变量的参数				○		📖 P.A-141
54015474Hex	未使用轴指定(辅轴)	运动控制指令的输入变量 “Auxiliary”指定的轴为未使用轴	• 指令的 “Auxiliary”指定的轴为未使用轴				○		📖 P.A-142
54015475Hex	位置指定齿轮指定值异常	无法以运动控制指令输入的速度/加速度/减速度进行同步动作	• 无法以指令输入的速度/加速度/减速度进行指定的同步动作				○		📖 P.A-142
54015476Hex	位置指定齿轮主轴零速	启动运动控制指令时主轴的速度为 “0”	• 启动指令时主轴的速度为 “0”				○		📖 P.A-143
54015478Hex	超过目标位置设定范围	运动控制指令的输入变量 “Position”指定的参数超过范围	• 指令的输入参数超过输入变量的范围 • 对于旋转模式的轴，目标位置超过环设定的范围				○		📖 P.A-143
54015479Hex	超过移动距离范围	运动控制指令的输入变量 “Distance”指定的参数超过范围或加上 “Distance”值后的目标位置超过范围	• 将指令输入参数的绝对值转换为脉冲单位时，已超过40位的范围 • 对于线性模式的轴，将加上移动距离后的目标位置转换为脉冲单位时超过带符号40位的范围				○		📖 P.A-144
5401547AHex	超过凸轮表起点位置设定范围	运动控制指令的输入变量 “StartPosition”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-144

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
5401547BHex	超过凸轮动作(主轴跟踪)开始位置设定范围	运动控制指令的输入变量“MasterStart Distance”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-145
5401547CHex	圆弧插补半径指定异常	MC_MoveCircular2D(2轴圆弧插补)指令下，指定半径指定方式时无法以指定的半径创建圆弧的轨迹	• MC_MoveCircular2D(2轴圆弧插补)指令下，指定了半径指定方式，无法以指定的半径创建圆弧的轨迹				○		P.A-145
5401547DHex	圆弧插补半径溢出	MC_MoveCircular2D(2轴圆弧插补)指令下，通过点指定方式/中心点指定方式时，圆弧的半径超过最大值	• MC_MoveCircular2D(2轴圆弧插补)指令下，通过点指定方式/中心点指定方式时，将圆弧的半径转换为脉冲单位时超过40位的范围				○		P.A-146
5401547EHex	超过圆弧轴指定范围	运动控制指令的输入变量“CircAxes”指定的参数超过范围	• 指令的输入参数超过输入变量的范围 • “CircAxes”指定的轴未包含在轴组设定的构成轴中 • “CircAxes”的2轴已指定相同轴				○		P.A-146
5401547FHex	辅轴/从轴、轴号非升序	运动控制指令指定的输入变量“Auxiliary”和“Slave”的参数值未以升序排列	• 指令的输入变量 “Auxiliary”和“Slave”的参数非升序				○		P.A-147
54015480Hex	凸轮表属性更新中数据升序异常	判定有效数据数计算中相位非升序。或判定计算后有效数据数为 “0”	• 判定有效数据数计算中相位非升序 • 判定计算后有效数据数为 “0”				○		P.A-147
54015481Hex	超过MC设定写入的对象范围	运动控制指令的输入变量 “Target”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-148
54015482Hex	超过主轴移动距离指定范围	运动控制指令的输入变量“MasterDistance”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-148
54015483Hex	超过主轴加速移动距离指定范围	运动控制指令的输入变量“MasterDistanceInACC”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-149
54015484Hex	超过主轴减速移动距离指定范围	运动控制指令的输入变量“MasterDistanceInDEC”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-149
54015487Hex	超过执行模式选择范围	运动控制指令的输入变量“ExecutionMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-150

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015488Hex	超过轴间偏差容许值范围	运动控制指令的输入变量“PermittedDeviation”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-150
54015489Hex	通过点位置/中心位置/超过半径指定范围	运动控制指令的输入变量“AuxPoint”指定的参数超过范围	• 通过点指定或中心指定时，将“AuxPoint”的值转换为脉冲单位时超过带符号40位的范围 • 半径指定时，将“AuxPoint[0]”的绝对值转换为脉冲单位时超过40位的范围				○		P.A-151
5401548AHex	超过终点指定范围	运动控制指令的输入变量“EndPoint”指定的参数超过范围	• 将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围				○		P.A-151
5401548BHex	超过从轴移动距离指定范围	运动控制指令的输入变量“SlaveDistance”指定的参数超过范围	• 将指令的输入参数转换为脉冲单位时，已超过40位的范围				○		P.A-152
5401548CHex	超过相位补偿量范围	运动控制指令的输入变量“PhaseShift”指定的参数超过范围	• 将指令输入参数的绝对值转换为脉冲单位时，已超过40位的范围				○		P.A-152
5401548DHex	超过标准距离范围	运动控制指令的输入变量“FeedDistance”指定的参数超过范围	• 将指令输入参数的绝对值转换为脉冲单位时，已超过40位的范围				○		P.A-153
5401548EHex	辅轴/从轴相同	运动控制指令的输入变量“Auxiliary”和“Slave”指定的轴相同	• 指令的输入变量“Auxiliary”和“Slave”的参数相同				○		P.A-153
5401548FHex	超过相对位置选择范围	运动控制指令的输入变量“Relative”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-154
54015490Hex	超过凸轮过渡指定选择范围	运动控制指令的输入变量“CamTransition”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-154
54015491Hex	超过同步控制解除模式选择范围	运动控制指令的输入变量“OutMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-155
54015492Hex	无法执行启用外部锁定指令	对于编码器轴，输入变量“StopMode”指定为“_mcImmediateStop(立即停止)”，执行了驱动模式的MC_TouchProbe(启用外部锁定)指令	• 对于编码器轴，输入变量“StopMode”指定为“_mcImmediateStop(立即停止)”，执行了驱动模式的MC_TouchProbe(启用外部锁定)指令				○		P.A-155

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015493Hex	超过主轴偏移设定范围	运动控制指令的输入变量“MasterOffset”指定的参数超过范围	• 将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围				○		P.A–156
54015494Hex	超过从轴偏移设定范围	运动控制指令的输入变量“SlaveOffset”指定的参数超过范围	• 将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围				○		P.A–156
54015495Hex	超过指令当前位置计数选择范围	运动控制指令的输入变量“CmdPosMode”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–157
54015496Hex	超过主轴齿轮比分子范围	运动控制指令的输入变量“RatioNumeratorMaster”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–157
54015497Hex	超过主轴齿轮比分母范围	运动控制指令的输入变量“RatioDenominatorMaster”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–158
54015498Hex	超过辅轴齿轮比分子范围	运动控制指令的输入变量“RatioNumeratorAuxiliary”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–158
54015499Hex	超过辅轴齿轮比分母范围	运动控制指令的输入变量“RatioDenominatorAuxiliary”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–159
5401549AHex	超过主轴位置类型选择范围	运动控制指令的输入变量“ReferenceTypeMaster”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–159
5401549BHex	超过辅轴位置类型选择范围	运动控制指令的输入变量“ReferenceTypeAuxiliary”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–160
5401549CHex	超过目标位置环计数器范围	已执行指令的目标位置超过环计数器的范围，因此无法动作	• 设定为环计数器范围内不含0时，执行了高速原点复位				○		P.A–160
5401549DHex (Ver.1.01以上)	超过轴组构成轴设定范围	运动控制指令的输入变量 “Axes”指定的参数超过范围	• 指令的输入参数超过输入变量的范围 • 分配轴组构成轴的任务不同				○		P.A–161
5401549EHex (Ver.1.04以上)	超过轴使用设定范围	运动控制指令的输入变量 “AxisUse”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A–161

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015700Hex (Ver.1.03以上)	超过原点复位参数设定范围	运动控制指令的输入变量“HomingParameter”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		📖 P.A-162
54015702Hex (Ver.1.04以上)	轴未使用切换异常	向非处于停止状态或处于指令速度饱和状态的轴执行了MC_ChangeAxisUse(轴使用变更)指令	• 向非处于停止状态或处于指令速度饱和状态的轴执行了MC_ChangeAxisUse(轴使用变更)指令				○		📖 P.A-162
54015703Hex (Ver.1.06以上)	无法进行轴使用切换	试图超出使用实轴最大数量或使用运动控制伺服轴最大数量执行MC_ChangeAxisUse(轴使用变更)指令	• 试图超出使用实轴最大数量执行MC_ChangeAxisUse(轴使用变更)指令 • 试图超出使用运动控制伺服轴最大数量执行MC_ChangeAxisUse(轴使用变更)指令				○		📖 P.A-163
54015720Hex (Ver.1.04以上)	轴使用切换时运动控制参数设定异常	切换为使用轴的轴的运动控制参数设定错误	• MC_ChangeAxisUse(轴使用变更)指令下，从未使用轴切换到使用轴的轴的运动控制参数设定错误 • 下载运动控制参数设定过程中电源断开 • 非易失性存储器的故障或非易失性存储器的寿命				○		📖 P.A-163
54015721Hex (Ver.1.04以上)	未设定轴使用切换时所需的过程数据对象	未设定切换为使用轴的轴类型的所需对象	• 切换为使用轴的轴类型的所需对象未设定PDO映射 • 下载运动控制参数设定过程中电源断开 • 非易失性存储器的故障或非易失性存储器的寿命 • 对[轴使用]设定为[未使用轴(无法切换为使用轴)]的轴执行了MC_ChangeAxisUse(轴使用变更)指令				○		📖 P.A-164
54015722Hex (Ver.1.06以上)	反馈位置正在发生溢出/下溢	反馈位置正在发生溢出/下溢时启动了无法执行的指令	• 反馈位置正在发生溢出或下溢时启动了无法执行的指令				○		📖 P.A-165
54015723Hex (Ver.1.06以上)	开关结构体的机架编号超过设定范围	运动控制指令的输入输出变量“Switches”指定的“TrackNumber”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		📖 P.A-165
54015724Hex (Ver.1.06以上)	开关结构体的ON开始位置超过设定范围	运动控制指令的输入输出变量“Switches”指定的“FirstOnPosition”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		📖 P.A-166
54015725Hex (Ver.1.06以上)	开关结构体的ON结束位置超过设定范围	运动控制指令的输入输出变量“Switches”指定的“LastOnPosition”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		📖 P.A-166

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015726Hex (Ver.1.06以上)	开关结构体的方向选择超过范围	运动控制指令的输入输出变量“Switches”指定的“AxisDirection”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		P.A-167
54015727Hex (Ver.1.06以上)	开关结构体的开关模式选择超过范围	运动控制指令的输入输出变量“Switches”指定的“CamSwitchMode”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		P.A-167
54015728Hex (Ver.1.06以上)	开关结构体的ON时间超过设定范围	运动控制指令的输入输出变量“Switches”指定的“Duration”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		P.A-168
54015729Hex (Ver.1.06以上)	机架选项结构体的ON时刻补偿超过设定范围	运动控制指令的输入输出变量“TrackOptions”指定的“OnCompensation”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		P.A-168
5401572AHex (Ver.1.06以上)	机架选项结构体的OFF时刻补偿超过设定范围	运动控制指令的输入输出变量“TrackOptions”指定的“OffCompensation”值超过范围	• 指令的输入输出变量指定的结构体型变量的结构要素值超过范围				○		P.A-169
5401572BHex (Ver.1.06以上)	开关结构体型变量的数组元素数超过范围	运动控制指令的输入输出变量“Switches”指定的结构体型变量的数组元素数超过范围	• 指令的输入输出变量指定的结构体型变量的数组元素数超过范围				○		P.A-169
5401572CHex (Ver.1.06以上)	输出信号结构体型变量的数组元素数超过范围	运动控制指令的输入输出变量“Outputs”指定的结构体型变量的数组元素数超过范围	• 指令的输入输出变量指定的结构体型变量的数组元素数超过范围				○		P.A-170
5401572DHex (Ver.1.06以上)	机架选项结构体型变量的数组元素数超过范围	运动控制指令的输入输出变量“TrackOptions”指定的结构体型变量的数组元素数超过范围	• 指令的输入输出变量指定的结构体型变量的数组元素数超过范围				○		P.A-170
5401572EHex (Ver.1.06以上)	输出信号与机架选项的数组元素数不一致	运动控制指令的输入输出变量“Outputs”与“TrackOptions”指定的结构体型变量的数组元素数不一致	• 指令的输入输出变量指定的输出信号结构体型变量与机架选项结构体型变量的数组元素数不一致				○		P.A-171
5401572FHex (Ver.1.06以上)	无法多重启动运动指令(主轴)	变更了多重启动指令时无法变更的输入输出变量“Master”	• 变更了多重启动指令时无法变更的输入输出变量 “Master”				○		P.A-171

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015730Hex (Ver.1.06以上)	无法多重启动运动指令(位置类型选择)	变更了多重启动指令时无法变更的输入输出变量“ReferenceType”	• 变更了多重启动指令时无法变更的输入输出变量“ReferenceType”				○		📖 P.A-172
54015731Hex (Ver.1.06以上)	开关结构体的同一机架指定数超过范围	运动控制指令的输入输出变量“Switches”指定的“TrackNumber”的同一机架编号数量超过范围	• 指令的输入输出变量指定的开关结构体型变量的“TrackNumber”指定的同一机架编号数量超过了1个机架可指定的个数范围				○		📖 P.A-172
5401573AHex (Ver.1.08以上)	轴参数无法写入	对未使用轴以外的轴启动了相应指令	• 对使用轴或未创建轴启动了相应指令				○		📖 P.A-173
5401573BHex (Ver.1.08以上)	轴参数超过设定范围	运动控制指令的输入变量“AxisParameter”指定的参数超过有效范围	• 指令的输入变量“AxisParameter”指定的参数超过输入变量的范围				○		📖 P.A-173
5401573CHex (Ver.1.08以上)	凸轮属性超过设定范围	运动控制指令的输入变量“CamProperty”指定的参数超过有效范围	• 指令的输入变量“CamProperty”指定的参数超过输入变量的范围				○		📖 P.A-174
5401573DHex (Ver.1.08以上)	凸轮节点超过设定范围	运动控制指令的输入变量“CamNodes”指定的参数超过有效范围	• 指令的输入变量“CamNodes”指定的参数超过输入变量的范围				○		📖 P.A-174
5401573EHex (Ver.1.08以上)	凸轮节点类型指定错误	运动控制指令的输入变量“CamNodes”指定的参数非_sMC_CAM_NODE型数组变量	• 指令的输入变量“CamNodes”指定的参数非_sMC_CAM_NODE型数组变量				○		📖 P.A-175
5401573FHex (Ver.1.08以上)	凸轮表生成节点数不足	运动控制指令的输入变量“CamNodes”指定的参数数组变量中元素编号0的Phase值为“0”	• 指令的输入变量“CamNodes”指定的参数数组变量中元素编号0的Phase(主轴相位)值为“0”				○		📖 P.A-175
54015740Hex (Ver.1.08以上)	凸轮节点主轴相位非升序	运动控制指令的输入变量“CamNodes”指定的参数数组变量中Phase的值未按元素编号顺序升序排列	• 指令的输入变量“CamNodes”指定的参数数组变量中Phase(主轴相位)的值未按元素编号顺序升序排列				○		📖 P.A-176
54015741Hex (Ver.1.08以上)	凸轮表生成数据点数过多	生成的凸轮数据数超过了运动控制指令的输入变量“CamTable”指定的凸轮数据变量的数组元素数	• 生成的凸轮表的凸轮数据数超过了指令的输入变量“CamTable”指定的凸轮数据变量的数组元素数				○		📖 P.A-176
54015742Hex (Ver.1.08以上)	凸轮表生成位移溢出	生成的凸轮表的Distance超过了REAL型可表示的范围	• 生成的凸轮表的Distance超过了REAL型可表示的范围				○		📖 P.A-177

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54015743Hex (Ver.1.08以上)	生成中止凸轮表的使用	相应指令的输入变量“CamTable”指定了被中止生成的凸轮数据变量	• 相应指令的输入变量“CamTable”指定了因MC_GenerateCamTable(凸轮表生成)指令异常而被中止生成的凸轮数据变量				○		P.A-177
54015749Hex (Ver.1.10以上)	执行ID超过设定范围	运动控制指令的输入变量“ExecID”指定的参数超过范围	• 指令的输入变量“ExecID”指定的参数超过输入变量的范围				○		P.A-178
5401574AHex (Ver.1.10以上)	位置偏置超过范围	运动控制指令的输入变量“OffsetPosition”指定的参数超过范围	• 将位置偏置转换为脉冲单位时，已超过带符号40位的范围				○		P.A-178
5401574BHex (Ver.1.10以上)	PDS状态变化指令选择超过范围	运动控制指令的输入变量“TransitionCmd”指定的参数超过范围	• 指令的输入参数超过输入变量的范围				○		P.A-179
5401574CHex (Ver.1.13以上)	无法启动单轴位置控制轴运动指令	执行了无法对单轴位置控制轴执行的动作指令	• 执行了无法对单轴位置控制轴执行的动作指令				○		P.A-179
54016440Hex	目标位置正方向软件超限	指定的位置超过正方向软件限制的范围	• 指令的输入变量“Position”指定的参数超过正方向软件限制的范围 • 起始位置超过正方向软件限制的范围，执行了指定与软件限制的范围相反方向的动作的指令 • 已指定通过点的MC_MoveCircular2D(2轴圆弧插补)指令的输入变量“AuxPoint”指定的参数超过正方向软件限制的范围				○		P.A-180
54016441Hex	目标位置负方向软件超限	指定的位置超过负方向软件限制的范围	• 指令的输入变量“Position”指定的参数超过负方向软件限制的范围 • 起始位置超过负方向软件限制的范围，执行了指定与软件限制的范围相反方向的动作的指令 • 已指定通过点的MC_MoveCircular2D(2轴圆弧插补)指令的输入变量“AuxPoint”指定的参数超过负方向软件限制的范围				○		P.A-181
54016442Hex	指令位置溢出/正在发生下溢	指令位置溢出/正在发生下溢时，执行了定位、溢出/下溢方向的指令或无法确定方向的指令	• 指令位置溢出/正在发生下溢时进行了以下操作 • 执行了定位的指令 • 执行了溢出/下溢方向的连续控制指令 • 执行了无法确定方向的指令(同步功能、扭矩控制)				○		P.A-181

事件代码	事件名称	内容	发生原因(推测原因)	重要程度					参考页
				全	部	轻	监	总	
54016443Hex	正方向限制输入中	正方向限制输入为“ON”的状态下执行了正方向动作的指令	• 正方向限制输入为 “ON”的状态下，执行了正方向动作的指令；或正方向限制输入为 “ON”的状态下，执行了不指定动作方向的指令。正方向限制输入为 “ON”的状态下，执行了轴组动作指令				○		P.A-182
54016444Hex	负方向限制输入中	负方向限制输入为“ON”的状态下，执行了负方向动作的指令	• 负方向限制输入为 “ON”的状态下，执行了负方向动作的指令；或负方向限制输入为 “ON”的状态下，执行了不指定动作方向的指令。负方向限制输入为 “ON”的状态下，执行了轴组动作指令				○		P.A-182
54017422Hex	伺服主电路电源OFF状态	伺服驱动器的主电路电源为 “OFF”的状态下，执行了伺服ON	• 伺服驱动器的主电路电源为 “OFF”状态下，执行了伺服ON				○		P.A-183

A–3 错误代码的详情

表示指令导致的异常(事件代码)的详细信息。事件代码的低4位表示指令的错误代码。各错误代码的说明请参阅对应的事件代码的说明。例如，指令的错误代码为16#0400时，请参阅54010400Hex的事件代码的说明。

表的说明

各异常的说明使用的表的各项目的含义在[ ]内表示。

事件名称	[异常(事件)的名称]			**事件代码**	[异常(事件)的代码]	
内容	[异常(事件)的内容]					
发生源	[异常(事件)发生的部位]		**发生源详情**	[发生源的详情]	**检测时间**	[异常检测的时间]
异常的属性	**重要程度**	[对控制产生影响的程度](*1)	**恢复方法**	[恢复方法](*2)	**日志类别**	[待保存的日志种类](*3)
发生后的影响	**用户程序**	[用户程序的执行状态](*4)	**动作**	[异常(事件)发生时的动作相关特别说明]		
LED	[内置EtherNet/IP端口用LED、内置EtherCAT端口用LED的显示状态。仅EtherCAT主站功能模块、EtherNet/IP功能模块时记载发生源]					
系统定义变量	**变量名称**		**数据类型**		**名称**	
	[检测异常的系统定义变量、受异常影响的系统定义变量、导致异常的系统定义变量的变量名称和数据类型、名称]					
发生原因及其处理	**发生原因(推测原因)**		**处理措施**		**防止再次发生**	
	[异常(事件)的发生原因、处理措施及防止再次发生的方法]					
附加信息	[Sysmac Studio/NS系列显示器中显示的附加信息内容](*5)					
注意事项/备注	[其他注意事项、限制事项、补充说明等]					

*1 以下中的任意一个
全部停止故障：全部停止故障电平
部分停止故障：部分停止故障电平
轻度故障：轻度故障电平
监控信息
总体信息

*2 以下中的任意一个
自动恢复：排除故障后自动恢复正常
异常解除：排除故障后通过执行异常解除恢复正常
重新接通电源：排除故障后通过重新接通控制器的电源恢复正常
控制器复位：排除故障后通过控制器复位恢复正常
基于发生原因：取决于发生原因

*3 以下中的任意一个
系统：系统事件日志
访问：访问事件日志

*4 以下中的任意一个
继续：继续执行用户程序
停止：停止执行用户程序
开始：开始执行用户程序

*5 以关于显示器故障诊断器的适用范围，请参阅 “NJ/NX系列 故障诊断手册(SBCA–361)”的附录。

事件名称	超过输入值范围		事件代码	54010400Hex		
内容	指令的输入参数已超过输入变量的范围。 或已通过整数0进行除法/余数运算。					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数已超过输入变量的范围。或已通过整数0进行除法/余数运算。		请确认相应指令的输入变量范围，避免输入参数超过范围或不以整数0进行除法/余数运算		请避免指令的输入参数值超过输入范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	数值关系错误		事件代码	54010401Hex		
内容	指令的输入参数值的关系未满足条件。 或指令执行运算中/结果的数值未满足条件					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	输入参数值的关系未满足条件。		请确认相应指令输入变量的含义和关系，进行修正以满足输入参数值的关系		请使指令的输入参数值满足输入变量的关系	
	指令执行运算中/结果的数值未满足条件		请确认相应指令的处理，避免输入参数值导致不恰当的运算结果		请确认指令的处理，避免输入参数在处理运算时导致本异常	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	浮点数错误			事件代码		54010402Hex	
内容	向指令的浮点数输入参数输入了非数值						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	向指令的浮点数输入参数输入了非数值		请进行修正，以将数值输入至相应指令的浮点数的输入参数			请将指令的浮点数的输入参数设为数值	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	非BCD			事件代码		54010403Hex	
内容	向指令的BCD输入参数输入了非BCD数据的值						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	向指令的BCD输入参数输入了16进制A、B、C、D、E、F		请进行修正，以将BCD数据输入至相应指令的BCD的输入参数			请将指令的BCD的输入参数设为BCD数据	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	带符号BCD错误			事件代码	54010404Hex	
内容	向指令的带符号BCD数据的输入参数的最高位输入了错误值					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	向指令的带符号BCD数据的输入参数的最高位数输入了错误值。 • BCD格式已指定为 “_BCD0” 时，最高位为2 ~ F。 • BCD格式已指定为 “_BCD2” 时，最高位为A、B、C、D、E中的任意一个。 • BCD格式已指定为 “_BCD3” 时，最高位为B、C、D、E中的任意一个。		请进行修正，以将带正确符号的BCD数据输入至相应指令的BCD的输入参数		请在指令的带符号BCD数据的输入参数的最高位指定正确的值	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	指定位位置错误			事件代码	54010405Hex	
内容	指令指定的位位置错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令指定的位位置已超过指定数据的范围		请进行修正，以将相应指令指定的位位置控制在指定的数据范围内		请将指令指定的位位置控制在指定的数据范围内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过区域范围指定			事件代码		54010406Hex
内容	指令指定的数据的存储器地址和数据大小不恰当					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令指定的数据的存储器地址超过有效范围。或指令指定的数据大小已超过有效范围。可能是变量的数据类型和数据大小不相符		请进行修正，以将相应指令指定的数据的存储器地址和数据大小控制在有效范围内		请将指令指定的数据的存储器地址和数据大小控制在有效范围内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	区域超限			事件代码		54010407Hex
内容	指令的运算结果已超过输出参数数据区域的范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	数组元素数等指令的运算结果已超过输出参数数据区域的范围		请修正输入参数，以将相应指令的处理结果控制在输出参数的数据区域范围内		请修正输入参数，以将指令的处理结果控制在输出参数的数据区域范围内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	控制器异常解除失败		事件代码		54010409Hex	
内容	在未发生控制器异常的状态下，执行了控制器异常解除指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对输出和单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	在未发生控制器异常的状态下，执行了控制器异常解除指令		请修正程序，以使发生控制器异常时执行相应指令		请创建程序，以使发生控制器异常时执行相应指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	用户异常解除失败		事件代码		5401040BHex	
内容	在未发生用户异常的状态下，执行了用户异常解除指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对输出和单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	在未发生用户异常的状态下，执行了用户异常解除指令		请修正程序，以使发生用户异常时执行相应指令		请创建程序，以使发生用户异常时执行相应指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	用户异常最大数量			事件代码		5401040CHex
内容	用户异常指令下发生了超过最大数量的用户异常					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对输出和单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	用户异常指令下发生了超过最大数量的用户异常		请执行用户异常解除指令。 确认用户异常数量时，监控系统定义变量的用户异常数量		请创建程序，以确保调出用户异常指令的条件为检查用户异常数量	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	指定单元错误			事件代码		5401040DHex
内容	指令指定的单元不存在					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对输出和单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定了作为单元构成信息不存在的单元		请进行修正，指定单元构成存在相应指令指定的单元No./单元编号且存在实体的单元		请指定单元构成中存在指令指定的单元No./单元编号且存在实体的单元	
	指定了单元构成信息中存在但实体不存在的单元					
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	单元重启失败			事件代码	5401040FHex	
内容	高功能单元重启失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对输出和单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	高功能单元处理中		请稍后再次重启高功能单元		请确认高功能单元未处于处理状态，再变更程序，以重启高功能单元	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	字符串格式异常			事件代码	54010410Hex	
内容	指令中输入的字符串不是正确的字符串					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将字符串转换为数值的指令的输入字符串不是表示数值或正数值的字符串		请进行修正，以使相应指令已输入的字符串符合该指令		将字符串转换为数值的指令时，请使待转换的输入字符串表示数值。含有正数条件时，请表示正数	
	输入字符串未以NULL字符结尾		请进行修正，以使相应指令已输入的字符串以NULL字符结尾		将字符串转换为数值的指令时，请使输入字符串以NULL字符结尾	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	指定程序错误			事件代码	54010411Hex	
内容	指令指定的程序不存在					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	相应指令指定的程序不存在或被删除		请进行修正，以使相应指令指定的程序为已存在的程序。 或追加相应指令已指定的程序		请将指令指定的程序设为已存在的程序。 或注意避免删除待使用的程序	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	存储器非地址指定			事件代码	54010413Hex	
内容	CJ单元用存储器指定的所需变量未在CJ单元用存储器中指定					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	CJ单元用存储器指定的所需变量未在CJ单元用存储器中进行AT指定		请进行修正，以通过AT指定将CJ单元用存储器指定的所需变量在CJ单元用存储器中指定		请创建程序，以通过AT指定将CJ单元用存储器指定的所需变量在CJ单元用存储器中指定	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	堆栈下溢			事件代码	54010414Hex	
内容	堆栈中无数据					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	试图从无数据的堆栈读取数据		请修正程序，以将数据保存至堆栈后提取数据		请创建程序，以将数据保存至堆栈后提取数据	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	数组的元素数、维数错误			事件代码	54010416Hex	
内容	针对指令的数组的输入输出参数，元素数、维数超过了范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	针对指令的数组的输入输出参数，元素数、维数超过了范围		请对指令的数组的输入输出参数进行修正，以将元素数、维数控制在范围内		请对指令的数组的输入输出参数进行修正，以将元素数、维数控制在范围内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	指令任务错误			事件代码		54010417Hex	
内容	指令指定的任务不存在						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	指定的任务不存在		请修正程序以指定存在的任务			请创建程序以指定存在的任务	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	不允许指定任务			事件代码		54010418Hex	
内容	不允许指定指令指定的任务						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	指定了我的任务、主要周期任务或周期任务		请修正程序以指定为非事件任务及我的任务			请创建程序以指定为非事件任务及我的任务	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	数据类型错误			事件代码	54010419Hex	
内容	已将指令无法使用的类型的数据指定至输入/输入输出					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	已将指令无法使用的类型的数据指定至输入/输入输出		请确认相应指令的输入/输入输出变量的类型，修正为可使用待输入数据的类型		请确认相应指令的输入/输入输出变量的类型，输入可使用类型的数据	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	多重启动指令			事件代码	5401041AHex	
内容	多重启动无法多重启动的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	试图同时执行多个无法多重启动的指令		请修正程序，对于无法多重启动的指令，在已执行的实例结束后执行其他实例。		请创建程序，对于无法多重启动的指令，在已执行的实例结束后执行其他实例。	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	数据容量超限		事件代码	5401041BHex*1		
内容	传输至指令的数据过大，无法处理					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将大小超过可处理指令的容量的数据传输至指令		请修正程序，以将小于传输至相应指令的数据容量的数据传输至指令		请确认指令可处理的数据容量，将小于该大小的数据传输至指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.02的CPU单元追加的错误代码(16#041B)。

事件名称	数据大小不一致		事件代码	5401041CHex*1		
内容	指令的输入/输入输出指定的数据与对象参数的大小不一致					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将大小与对象参数的大小不一致的数据指定至指令的输入/输入输出		请确认对象参数的大小，修正程序，以使待输入的数据大小一致		请确认对象参数的大小，创建程序，以使待输入的数据大小一致	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.04的CPU单元追加的错误代码(16#041C)。

事件名称	同时执行指令资源超限		事件代码	5401041DHex *1		
内容	超过可同时执行的相应指令组的资源执行了指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	超过可同时执行的个数执行了相应指令。		请修正程序，以将同时执行的相应指令控制在最多个数以内		请编写程序，以将同时执行的相应指令控制在最多个数以内	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#041D)。

事件名称	FINS异常		事件代码	54010800Hex		
内容	收发FINS指令时发生了异常					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	收发FINS指令时发生了异常		请确认相应指令的输出变量“ErrorIDEx”的值，参阅本手册中相应指令的扩展错误代码“ErrorIDEx”的说明		请事先参阅指令的 “ErrorIDEx”的说明，再创建程序	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法使用FINS端口			事件代码		54010801Hex
内容	正在使用FINS端口					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对通信输出和单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	正在使用FINS端口		请修正程序，以通过a接点插入“_Port_isAvailable”，作为输入条件		请创建程序，以通过a接点插入“_Port_isAvailable”，作为输入条件	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	串行通信模式错误			事件代码		54010C00Hex
内容	串行通信单元未处于执行指令所需的串行通信模式					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	串行通信单元的串行通信端口未设定至指令所需的模式		请变更执行指令所需的串行通信模式的设定。或修正程序，以确保设定模式下仅使用可执行的指令		请将串行通信单元设定为执行指令所需的串行通信模式。或创建程序，以确保设定模式下仅使用可执行的指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	接收缓存已满		事件代码	54010C03Hex *1	
内容	进入了接收缓存已满状态				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。即使因本异常而结束，仍将进行接收数据保存位置可保存大小的数据保存	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	以下原因导致接收缓存已满 • 配对设备的发送频率高 • 传送速度过快 • 缓存的接收处理频率低		请采取以下任意一种或所有措施，避免发生接收缓存满载的状态 • 降低配对设备的发送频率 • 降低传送速度 • 提高缓存的接收处理频率	使用时请考虑以下4个要素，避免发生接收缓存满载的状态 • 配对设备的发送频率 • 传送速度 • 缓存的接收处理频率 • 流程控制的采用	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息				

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C03)。

事件名称	端口多重启动		事件代码	54010C04Hex *1	
内容	执行了不能同时执行的串行通信指令				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	在不能同时执行的指令执行过程中启动了相应指令		请修正程序，以执行与无法同时执行指令的排他处理	请编写程序，以执行与无法同时执行指令的排他处理	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附属信息 • 无法同时执行的串行通信指令请参阅各指令的说明				

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C04)。

事件名称	奇偶校验错误			事件代码	54010C05Hex *1	
内容	接收数据发生了奇偶校验错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	通信设定、传送速度的设定与配对设备不一致		请使通信设定、传送速度的设定与配对设备一致		请使通信设定、传送速度的设定与配对设备一致	
	干扰		请采取防干扰措施		请采取防干扰措施	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C05)。

事件名称	结构错误			事件代码	54010C06Hex *1	
内容	接收数据发生了结构错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	通信设定、传送速度的设定与配对设备不一致		请使通信设定、传送速度的设定与配对设备一致		请使通信设定、传送速度的设定与配对设备一致	
	干扰		请采取防干扰措施		请采取防干扰措施	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C06)。

事件名称	超程错误		事件代码	54010C07Hex *1		
内容	接收数据发生了超程错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	传送速度过快，因此在接收处理中接收了下一个数据		请降低传送速度		请降低传送速度	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C07)。

事件名称	CRC不一致		事件代码	54010C08Hex *1		
内容	接收数据的CRC不一致					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	接收了错误的信息		请使配对设备侧的CRC生成方法与预期的一致		请确认配对设备侧的CRC生成方法是否与预期的一致	
	干扰		请重新接收。或者，请采取防干扰措施		请采取防干扰措施	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C08)。

事件名称	串行通信超时			事件代码	54010C0BHex *1	
内容	串行通信发生了超时					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	与配对设备之间未进行接线		请确认与配对设备之间的接线是否有问题，有问题时请修正接线		请确认与配对设备之间是否进行了接线	
	配对设备未接通电源		请接通配对设备的电源		请确认配对设备的电源是否已接通	
	通信设定、传送速度的设定与配对设备不一致		请使通信设定、传送速度的设定与配对设备一致		请使通信设定、传送速度的设定与配对设备一致	
	干扰		请采取防干扰措施		请采取防干扰措施	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C0B)。

事件名称	对非对象端口的指令执行			事件代码	54010C0CHex *1	
内容	指定非本指令的对象端口执行了指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定非本指令的对象端口执行了指令		请根据设备端口结构体指定本指令的对象端口后执行指令		请根据设备端口结构体指定本指令的对象端口后执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C0C)。

事件名称	CIF单元初始化			事件代码	54010C0DHex *1	
内容	CIF单元被初始化，因此CIF单元中缓存的收发数据消失					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	CIF单元被初始化		请根据需要重新发送或接收数据		在对CIF单元缓存收发数据的程序执行过程中，请勿重启CIF单元	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.13以上时发生的错误代码(16#0C0D)。

事件名称	Modbus例外响应			事件代码	54010C10Hex *1	
内容	Modbus的从站返回了例外代码					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	Modbus从站检测到异常		请确认ErrorIDEx的值16#0000_00xx的xx，参阅Modbus规定确定原因后采取相应措施。 Modbus规定的参考页请参阅相应指令的说明		包括配对设备在内，请按照Modbus规定编写用户程序	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C10)。

事件名称	Modbus响应错误			事件代码	54010C11Hex *1	
内容	Modebus从站返回非预期响应					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。通信输出遵从指令的规格。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	Modebus从站返回的响应的Function Code或数据大小错误		请调整选项的发送延时及接收监视时间等与配对设备之间收发数据的时序		请编写用户程序，以避免在返回响应前发送下一个指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.11以上时发生的错误代码(16#0C11)。

事件名称	无法使用SD存储卡			事件代码	54011400Hex	
内容	执行指令时SD存储卡访问失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	SD存储卡未安装或未正确插入		请切实安装SD存储卡		请确认SD存储卡已切实安装	
	SD存储卡损坏		请更换为已确认为正常的SD存储卡		无	
	SD存储卡插槽损坏		进行上述2项处理后仍再次发生本异常时，请更换CPU单元		无	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	SD存储卡写保护		事件代码	54011401Hex		
内容	执行指令时试图向写保护的SD存储卡进行写入					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	试图向写保护的SD存储卡进行写入		请解除SD存储卡的写保护。解除写保护时，将SD存储卡侧面的旋钮从锁定位置(LOCK)移动至可写入位置		对SD存储卡进行写入时，请使用未设置写保护的SD存储卡	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	SD存储卡容量不足		事件代码	54011402Hex		
内容	向正在执行指令的SD存储卡进行写入时，发生了SD存储卡的容量不足					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	发生了SD存储卡的容量不足		请更换为剩余容量充足的SD存储卡		对SD存储卡进行追加写入时，请使用剩余容量充足的SD存储卡	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附加信息 • 访问SD存储卡过程中，请勿取出SD存储卡。否则可能会损坏SD存储卡及其数据					

事件名称	指定文件错误		事件代码	54011403Hex		
内容	指令指定的文件不存在					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定的文件不存在		请进行修正，以使相应指令指定的文件名为存在的文件名。或将文件名修正为指令指定的名称		请将指令指定的文件名设为存在的文件名	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过最大文件和目录数		事件代码	54011404Hex		
内容	创建正在执行指令的文件/目录时，超过了最大文件/目录数					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	超过了最大文件和目录数		请删除无用文件/目录。或更换为文件/目录数相对于FAT16/FAT32的最大数余量充足的SD存储卡		请适当删除无用文件/目录，以防止文件/目录数过多。 如果使用过程中文件持续增加，则请定期更换SD存储卡	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	文件访问中			事件代码	54011405Hex	
内容	正在使用指令指定的文件，因此无法访问					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	试图以其他指令向正在通过其他指令访问的文件进行读写		请修正程序，以确保访问文件的其他指令的输出变量 “Busy” 为FALSE时执行相应指令		请创建程序，以确保在执行多个访问文件的指令时，将这些输出变量 “Busy” 用于指令的执行条件，不同时执行指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	打开模式不一致			事件代码	54011406Hex	
内容	指令下的文件操作与文件的打开模式不一致					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	文件打开指令指定的文件的打开模式与之后通过SD存储卡用指令进行的文件操作不匹配		请修正文件打开指令指定的打开模式，以使其变为与文件操作相符的模式		请确保文件打开指令指定的打开模式与之后的文件操作相符	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过偏置范围		事件代码	54011407Hex	
内容	无法访问指令指定的偏置指定的地址				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	试图超过文件大小进行访问		请减小相应指令指定的偏置	请变更程序，以将文件格式的识别信息等包含在文件内，对其进行检查并适当查找文件	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	非空目录		事件代码	54011408Hex	
内容	试图执行目录删除指令或变更目录名，但目录内容不是空的。				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	试图执行目录删除指令，但目录内容不是空的		请删除相应目录内的所有文件	使用删除目录的指令或变更目录的名称时，请事先确认目录的内容	
	试图变更目录名，但目录中存在目录		请删除相应目录内的所有目录		
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	含有相同文件名			事件代码	54011409Hex	
内容	存在与指令指定的文件同名的文件，因此无法执行指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	已存在与指令指定的创建文件同名的文件		请进行修正，以避免相应指令指定的文件名与已存在的文件名重复。或事先删除相应文件		使用创建文件的指令时，请避免指令指定的文件名与已存在的文件名重复	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附加信息 • 删除文件时，请先确认不是必要文件后再删除					

事件名称	禁止写入指定文件			事件代码	5401140AHex	
内容	执行指令时试图向写保护的文件或目录进行写入					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令指定的写入文件或目录已设置写保护		请解除相应指令已指定的写入文件的写保护。或变更写入文件名		请勿对写入文件设置写保护	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附加信息 • 解除文件的写保护时，请确认该文件为可覆盖的文件					

事件名称	超过最大文件打开数			事件代码		5401140BHex
内容	打开正在执行指令的文件时，已打开的文件数超过了最大数量					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	打开正在执行指令的文件时，已打开的文件数超过了最大数量		请修正程序，以减少同时打开的文件数		请减少待使用的文件数。或创建程序，以关闭无需打开的文件，避免同时打开的文件数过多	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	指定目录错误			事件代码		5401140CHex
内容	指令指定的目录不存在					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令指定的目录不存在		请进行修正，以使相应指令指定的目录为存在的目录。或事先创建相应目录		使用参照目录的指令时，请确保指令指定的目录为已存在的目录	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过最大文件和目录名长度			事件代码	5401140DHex	
内容	指令指定的文件名或目录名过长					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令指定的创建文件名或创建目录名过长		请修正程序，以将相应指令指定的文件名或目录名控制在FAT16/FAT32的限制内		请创建程序，以将指定的文件名或目录名控制在FAT16/FAT32的限制内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	SD存储卡访问失败			事件代码	5401140EHex	
内容	SD存储卡访问失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	SD存储卡损坏		请更换SD存储卡		无	
	SD存储卡插槽损坏		进行上述处理后仍发生本异常时，请更换CPU单元		无	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	备份功能已执行			事件代码	5401140FHex *1	
内容	其他备份功能正在动作					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	其他备份功能正在动作		请在其他备份功能执行完毕后再次执行指令		在其他备份功能执行过程中请勿进行备份	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#140F)。

事件名称	无法执行备份			事件代码	54011410Hex *1	
内容	正在执行其他功能，因此未能执行备份					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	在线编辑的执行过程中执行了指令		请在在线编辑完成后再次执行指令		在线编辑的执行过程中请勿进行备份	
	凸轮表保存指令的执行过程中执行了指令		请在凸轮表保存指令完成后再次执行指令		凸轮表保存指令的执行过程中请勿进行备份	
	CPU单元名称更新的执行过程中执行了指令		请在CPU单元名称更新完成后再次执行指令		CPU单元名称更新过程中请勿进行备份	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#1410)。

事件名称	单元/从站备份失败			事件代码	54011411Hex *1	
内容	单元/从站备份失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。对单元的动作无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	单元/从站备份失败		请参阅 “备份执行失败(CJ单元)”(102D0000Hex)或 “备份执行失败”(EtherCAT从站)”(102F0000Hex)的处理措施		请参阅 “备份执行失败(CJ单元)”(102D0000Hex)或 “备份执行失败”(EtherCAT从站)”(102F0000Hex)的再发防止对策	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#1411)。

事件名称	EtherCAT通信错误			事件代码	54011800Hex	
内容	执行指令时EtherCAT网络访问失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	EtherCAT网络未处于可执行状态		请确认EtherCAT主站状态下EtherCAT网络的动作状态，进行必要的处理，以排除故障		取决于异常的内容	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	EtherCAT从站不存在			事件代码		54011801Hex
内容	执行指令时对象从站访问失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对象从站不存在		请指定存在的节点地址		请为对象从站指定存在的节点地址	
	对象从站未处于可执行状态		请确认对象从站的状态，使对象从站处于可执行状态		请使对象从站处于可执行状态	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	EtherCAT超时			事件代码		54011802Hex
内容	执行指令时EtherCAT从站访问超时					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	与对象从站的通信超时		请确认对象从站的动作状态，进行必要的处理，以排除故障		取决于异常的内容	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	接收缓存溢出			事件代码		54011803Hex	
内容	执行指令时，来自EtherCAT从站的接收数据超过接收缓存						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。无法从从站接收数据			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	来自从站的接收数据大小超过接收缓存		请将接收缓存的大小设定为大于来自从站的接收缓存大小			请将接收缓存的大小设定为大于来自从站的接收缓存大小	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	SDO中止错误			事件代码		54011804Hex	
内容	执行指令时，从EtherCAT从站接收了SDO中止错误						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	取决于从站规格		请参阅从站的手册进行处理			请参阅从站的手册采取预防措施	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	分组监控保存中			事件代码		54011805Hex
内容	保存EtherCAT的分组监控文件过程中，执行了分组监控操作相关指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	保存EtherCAT的分组监控过程中，执行了分组监控操作相关指令		请在分组监控保存完成后执行分组监控操作相关指令。可通过分组监控保存中状态确认分组监控保存是否完成		请在分组监控保存完成后执行分组监控操作相关指令。可通过分组监控保存中状态确认分组监控保存是否完成	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	分组监控功能未启动			事件代码		54011806Hex
内容	停止EtherCAT的分组监控过程中，执行了分组监控停止指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	停止EtherCAT的分组监控功能过程中，执行了分组监控停止指令		请在分组监控功能启动后执行分组监控停止指令。可通过分组监控功能运行状态确认分组监控功能是否正在运行		请在分组监控功能启动后执行分组监控停止指令。可通过分组监控功能运行状态确认分组监控功能是否正在运行	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	分组监控功能运行中		事件代码	54011807Hex		
内容	运行EtherCAT的分组监控过程中，执行了分组监控启动指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	运行EtherCAT的分组监控功能过程中，再次执行了分组监控启动指令		请在分组监控功能停止后执行分组监控启动指令。可通过分组监控功能运行状态确认分组监控功能是否正在停止		请在分组监控功能停止后执行分组监控启动指令。可通过分组监控功能运行状态确认分组监控功能是否正在停止	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	通信资源超限		事件代码	54011808Hex		
内容	同时执行了超过32个的EtherCAT通信指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	同时执行了超过32个的EtherCAT通信指令。EtherCAT通信指令如下所示 • EC_CoESDOWrite指令 • EC_CoESDORead指令 • EC_ConnectSlave指令 • EC_DisconnectSlave指令 • EC_StartMon指令 • EC_SaveMon指令 • EC_StopMon指令 • EC_CopyMon指令		请修正程序，以将同时执行的EtherCAT指令控制在32个以下		请创建程序，以将同时执行的EtherCAT指令控制在32个以下	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	不支持分组监控功能		事件代码	54011809Hex*1	
内容	无法使用分组监控功能				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	对未配备分组监控功能的CPU单元执行了分组监控功能的指令		请勿执行EC_StartMon、EC_SaveMon、EC_StopMon、EC_CopyMon指令。 使用分组监控时，请使用配备分组监控功能的CPU单元单元	请勿对未配备分组监控功能的CPU单元的机型执行了分组监控功能的指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

*1 单元版本Ver.1.01的CPU单元追加的错误代码(16#1809)。

事件名称	Explicit异常		事件代码	54011C00Hex	
内容	通过CIP通信指令以Explicit信息返回了错误响应代码				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	取决于异常的内容		请确认相应指令的输出变量“ErrorIDEx”的值，参阅本手册的CIP信息异常代码的说明	取决于异常的内容。请参阅本手册的CIP信息异常代码的说明	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	根路径错误			事件代码	54011C01Hex	
内容	CIP通信指令指定的根路径格式错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	CIP通信指令指定的根路径格式错误		请修正程序，以改正相应指令指定的根路径		请创建程序，以改正指令指定的根路径	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	CIP句柄错误			事件代码	54011C02Hex	
内容	CIP通信指令指定的句柄错误。					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	CIP通信指令指定的句柄错误		请修正程序，以使相应指令指定的句柄为CIPOpen指令获取的句柄		请创建程序，以使指令指定的句柄为CIPOpen指令获取的句柄	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	CIP通信资源超限			事件代码		54011C03Hex
内容	超过可同时执行的CIP通信指令的资源执行了指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	同时执行了超过32个的CIP通信指令		请修正程序，以将同时执行的CIP通信指令控制在32个以下		请创建程序，以将同时执行的CIP通信指令控制在32个以下	
	试图同时使用超过32个的句柄		请修正程序，以将同时使用的句柄控制在32个以下		请创建程序，以将同时使用的句柄控制在32个以下	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	CIP超时			事件代码		54011C04Hex
内容	执行CIP通信指令过程中发生了超时					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定IP地址的设备不存在		请进行修正，以将指定的IP地址设为与配对设备的IP地址一致		请将指定的IP地址设为与发送对象设备的IP地址一致	
	指定句柄的CIP连接超时，因此已关闭		请在连接超时时间内执行相应指令。或延长连接超时时间		请在连接超时时间内执行相应指令	
	配对设备的电源OFF		请确认配对设备的状态，正常启动		请确认配对设备的状态，正常启动	
	配对设备的通信停止					
	EtherNet/IP的Ethernet电缆连接器断开		请确认连接器是否正确嵌合，重新连接		请切实连接连接器	
	EtherNet/IP的Ethernet电缆断线		请更换Ethernet电缆		无	
	干扰		干扰较多时，请采取防干扰措施		干扰较多时，请采取防干扰措施	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	Class3连接建立失败		事件代码	54011C05Hex *1		
内容	使用CIP通信指令建立Class3连接失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对不支持Class3(Large_Forward_Open)的设备执行了CIPOpen指令		对不支持Class3(Large_Forward_Open)的设备，请使用CIPOpenWithDataSize指令修正程序，使数据大小为509字节以下		对不支持Class3(Large_Forward_Open)的设备，请使用CIPOpenWithDataSize指令编写程序，使数据大小为509字节以下	
	对不支持Class3(Large_Forward_Open)的设备，将数据大小设定成510字节以上后执行了CIPOpenWithDataSize指令		对不支持Class3(Large_Forward_Open)的设备，请修正程序，使CIPOpenWithDataSize指令的数据大小为509字节以下		对不支持Class3(Large_Forward_Open)的设备，请编写程序，使CIPOpenWithDataSize指令的数据大小为509字节以下	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#1C05)。

事件名称	CIP通信数据大小超限		事件代码	54011C06Hex *1		
内容	试图使用CIP通信指令发送超出可发送数据大小的Class3 Explicit信息					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	CIPRead指令、CIPWrite指令、CIPSend指令的输入变量设定的数据大小超出了CIPOpenWithDataSize指令设定的数据大小		请修正程序，使相应指令的数据不超过CIPOpenWithDataSize指令设定的数据大小。 或者，请设定CIPOpenWithDataSize指令的数据大小，使其不低于相应指令的数据大小，并建立连接		请编写程序，使相应指令的数据不超过CIPOpenWithDataSize指令设定的数据大小。 或者，请设定CIPOpenWithDataSize指令的数据大小，使其不低于相应指令的数据大小，并建立连接	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#1C06)。

事件名称	本机IP地址设定错误			事件代码		54012000Hex
内容	在本机IP地址已发生设定错误的状态下，执行了指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	在本机IP地址已发生设定错误的状态下，执行了指令		执行相应指令时已发生 “TCP/IP基本设定错误”(IP地址设定错误)。请采取“TCP/IP基本设定错误”的处理措施，以排除异常		请正确进行IP地址的设定，以避免发生 “TCP/IP基本设定异常”(IP地址设定错误)	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法使用TCP/UDP端口			事件代码		54012001Hex
内容	执行指令时已使用UDP或TCP端口					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	已使用UDP或TCP端口		请修正程序，以使相应指令指定的端口编号为未使用的端口编号		请创建程序，以避免指令指定已使用的端口编号	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	地址解决失败			事件代码		54012002Hex	
内容	指令指定域名的对象节点的地址解析失败						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	相应指令指定的域名错误		请将相应指令指定的域名修正为正确的域名			请确认指令指定了正确的域名	
	控制器的hosts设定/DNS设定错误		请修正控制器的hosts设定/DNS设定			请确认控制器的hosts设定/DNS设定无误	
	DNS服务器的设定错误		请修正DNS服务器的设定			请确认DNS服务器的设定无误	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	Socket状态异常			事件代码	54012003Hex	
内容	执行指令时的状态不正确					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	• SktUDPCreate指令时 输入变量 “SrcUdpPort”指定的UDP端口为以下任意一种 •已打开 •关闭处理中 • SktUDPRcv指令时 •指定的Socket接收处理中 •指定的Socket已关闭 • SktUDPSend指令时 •指定的Socket发送处理中 •指定的Socket已关闭 • SktTCPAccept指令时 指定的TCP端口为以下任意一种 •打开处理中 •关闭处理中 •本指令已由相同IP地址、TCP端口建立连接 • SktTCPConnect指令时 •输入变量 “SrcTcpPort”指定的TCP端口已打开 •输入变量 “DstAdr”指定的对象节点不存在 •输入变量 “DstAdr”、“DstTcpPort”指定的对象节点未处于连接等待状态 • SktTCPRcv指令时 •指定的Socket接收处理中 •指定的Socket已关闭 • SktTCPSend指令时 •指定的Socket发送处理中 •指定的Socket已关闭 •指定Socket的发送缓存已满(连接目标电源OFF、线路切断等) • SktSetOption指令时 •指定Socket已开始收发信息 •指定了指定Socket不支持的选项类型		请排除相应指令对应的本异常的发生原因		请勿在故障发生的状态下执行指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	本机IP地址未确定		事件代码	54012004Hex	
内容	执行Socket服务指令时本机IP地址未确定				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	BOOTP服务器的设定异常		BOOTP服务器的设定有误时，请修正设定	请确认BOOTP服务器的设定无误	
	BOOTP服务器不存在		请确认BOOTP服务器是否已正常连接至网络及启动，确保可连接至BOOTP服务器	请确认BOOTP服务器已正常连接至网络及启动	
	刚启动后本机IP地址未确定		请等待本机IP地址确定后再执行Socket服务指令	请等待本机IP地址确定后再执行Socket服务指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	Socket超时		事件代码	54012006Hex	
内容	Socket服务指令发生超时				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	SktTCPAccept指令： 用户指定的超时时间内，无来自对象节点的连接请求		请修正系统或程序，以使在执行相应指令后到超时之间，从对象节点执行连接请求。或延长超时时间	请创建系统或程序，以使在执行相应指令后到超时之间，从对象节点执行连接请求	
	SktTCPRcv指令、SktUDPRcv指令： 用户指定的超时时间内，无法接收来自对象节点的数据		请修正系统或程序，以使在执行相应指令后到超时之间，从对象节点执行数据发送。或延长超时时间	请创建系统或程序，以使在执行指令后到超时之间，从对象节点执行数据发送	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	Socket句柄错误			事件代码		54012007Hex	
内容	Socket服务指令指定的句柄错误						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	Socket服务指令指定的句柄错误		请修正程序，以使相应指令指定的Socket句柄为以下任意一个指令获取的句柄 • SktUDPCreate指令 • SktTCPConnect指令 • SktTCPAccept指令			请创建程序，以使相应指令指定的Socket句柄为以下任意一个指令获取的句柄 • SktUDPCreate指令 • SktTCPConnect指令 • SktTCPAccept指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	Socket通信资源超限			事件代码		54012008Hex	
内容	超过可同时执行的Socket服务指令的资源执行了指令。						
发生源	PLC功能模块		发生源详情	指令		检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	无		–			–	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	同时执行了超过32个的Socket服务指令		请修正程序，以将同时执行的Socket服务指令控制在32个以下			请创建程序，以将同时执行的Socket服务指令控制在32个以下	
	试图同时使用超过30个(Ver.1.02以下的CPU单元时为16个)的Socket句柄		请修正程序，以将同时使用的Socket句柄控制在30个(Ver.1.02以下的CPU单元时为16个)以下			请创建程序，以将同时使用的Socket句柄控制在30个(Ver.1.02以下的CPU单元时为16个)以下	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

<table>
<tr><td>事件名称</td><td colspan="2">无执行权限</td><td>事件代码</td><td colspan="2">54012400Hex*1</td></tr>
<tr><td>内容</td><td colspan="5">在无法执行的状态下执行了变更EtherNet/IP端口设定的指令</td></tr>
<tr><td>发生源</td><td colspan="2">PLC功能模块</td><td>发生源详情</td><td>指令</td><td>检测时间</td><td>执行指令时</td></tr>
<tr><td>异常的属性</td><td>重要程度</td><td>监控信息</td><td>恢复方法</td><td>–</td><td>日志类别</td><td>系统</td></tr>
<tr><td>发生后的影响</td><td>用户程序</td><td>继续</td><td>动作</td><td colspan="3">无影响</td></tr>
<tr><td rowspan="2">系统
定义变量</td><td colspan="2">变量名称</td><td colspan="2">数据类型</td><td colspan="2">名称</td></tr>
<tr><td colspan="2">无</td><td colspan="2">–</td><td colspan="2">–</td></tr>
<tr><td rowspan="6">发生原因及
其处理</td><td colspan="2">发生原因(推测原因)</td><td colspan="2">处理措施</td><td colspan="2">防止再次发生</td></tr>
<tr><td colspan="2">内置EtherNet/IP端口重启过程中，执行了变更内置EtherNet/IP端口或CJ系列EtherNet/IP单元设定的指令</td><td colspan="2" rowspan="4">请在结束内置EtherNet/IP端口、CJ系列EtherNet/IP单元的重启和设定变更后再执行变更设定的指令</td><td colspan="2" rowspan="4">请在未执行内置EtherNet/IP端口、CJ系列EtherNet/IP单元的重启和设定变更时执行变更设定的指令</td></tr>
<tr><td colspan="2">CJ系列EtherNet/IP单元重启过程中，执行了变更该单元设定的指令</td></tr>
<tr><td colspan="2">通过指令或CIP信息变更内置EtherNet/IP端口设定过程中，执行了变更EtherNet/IP端口或CJ系列EtherNet/IP单元设定的指令</td></tr>
<tr><td colspan="2">通过指令或CIP信息变更CJ系列EtherNet/IP单元设定过程中，执行了变更该单元设定的指令</td></tr>
<tr><td colspan="2">指令指定的单元编号非内置EtherNet/IP端口、CJ系列EtherNet/IP单元</td><td colspan="2">请由指令指定内置EtherNet/IP端口或CJ系列EtherNet/IP单元的单元编号。单元构成错误时，请改正单元构成</td><td colspan="2">请由指令指定内置EtherNet/IP端口或CJ系列EtherNet/IP单元的单元编号</td></tr>
<tr><td>附加信息</td><td colspan="6">附加信息1：异常的发生部位
附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号
附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示
附加信息4：扩展错误代码(ErrorIDEx)</td></tr>
<tr><td>注意事项/
备注</td><td colspan="6">发生异常后变更程序时，可能不会正确显示附加信息</td></tr>
</table>

*1　单元版本Ver.1.02的CPU单元追加的错误代码(16#2400)。

事件名称	设定反映失败			事件代码	54012401Hex*1	
内容	CJ系列EtherNet/IP单元无法反映变更的设定					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	执行变更CJ系列EtherNet/IP单元设定的指令过程中，该单元或内置EtherNet/IP端口重启		CJ系列EtherNet/IP单元、内置EtherNet/IP端口重启结束后，请再次执行变更设定的指令		执行变更CJ系列EtherNet/IP单元设定的指令过程中，请勿重启该单元或内置EtherNet/IP端口	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.02的CPU单元追加的错误代码(16#2401)。

事件名称	同时执行指令数超限			事件代码	54012402Hex*1	
内容	超过可同时执行的数量执行了控制器通信设定的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	同时执行了2个以上的控制器通信设定的指令		请修正程序，以使同时执行的控制器通信设定的指令为1个		请创建程序，以使同时执行的控制器通信设定的指令为1个	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.02的CPU单元追加的错误代码(16#2402)。

事件名称	FTP客户端执行数超限		事件代码	54012403Hex *1		
内容	FTP客户端通信指令超过可同时执行的数量进行了执行					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	同时执行了4个以上的FTP客户端通信指令		请修正程序，以将同时执行的FTP客户端通信指令控制在3个以下		请编写程序，以将同时执行的FTP客户端通信指令控制在3个以下	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2403)。

事件名称	文件数超限		事件代码	54012404Hex *1		
内容	FTP客户端通信指令的通配符指定对象文件超过了1000个					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	FTP客户端通信指令以通配符指定文件名时，对象文件超过了1000个		请修正程序，使FTP客户端通信指令的通配符对象文件数为1000个以下		请编写程序，使FTP客户端通信指令的通配符对象文件数为1000个以下	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2404)。

事件名称	指定目录错误(FTP)			事件代码	54012405Hex *1	
内容	FTP客户端通信指令指定的目录在控制器中不存在或指定了错误的路径					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	FTP客户端通信指令指定的目录在控制器中不存在或指定了错误的路径		请修正程序，使FTP客户端通信指令指定SD存储卡中存在的目录		请编写程序，使FTP客户端通信指令指定SD存储卡中存在的目录	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2405)。

事件名称	FTP服务器连接失败			事件代码	54012406Hex *1	
内容	FTP客户端通信指令指定的连接目标FTP服务器在网络上不存在或已停止FTP服务					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	FTP客户端通信指令指定的连接目标FTP服务器在网络上不存在		请修正程序，使FTP客户端通信指令指定网络中存在的FTP服务器		请编写程序，使FTP客户端通信指令指定网络中存在的FTP服务器	
	FTP客户端通信指令指定的连接目标FTP服务器已停止FTP服务		请启动指定连接目标FTP服务器的FTP服务，再次执行指令		请确认指定连接目标FTP服务器的FTP服务未停止后再执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*2 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2406)。

事件名称	连接目标FTP服务器执行失败			事件代码	54012407Hex *1	
内容	使用FTP客户端通信指令时，连接目标FTP服务器返回了错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	使用FTP客户端通信指令对连接目标FTP服务器请求的处理，在连接目标FTP服务器侧执行失败		请根据相应指令的输出变量“ErrorIDEx”的值确认连接目标FTP服务器的响应代码，参阅本手册中相应指令的扩展错误代码“ErrorIDEx”的说明		请事先参阅指令的 “ErrorIDEx”的说明，再编写程序	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2407)。

事件名称	SD存储卡访问失败(FTP)			事件代码	54012408Hex *1	
内容	FTP客户端对SD存储卡的访问失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	未安装SD存储卡		请安装SD存储卡，再次执行指令		请安装SD存储卡	
	执行FTP客户端通信指令的过程中拔出了SD存储卡		请安装SD存储卡，再次执行指令		执行FTP客户端通信指令的过程中请勿拔出SD存储卡	
	SD存储卡容量不足		请更换为剩余容量充足的SD存储卡		请使用剩余容量充足的SD存储卡	
	SD存储卡被写保护		请解除SD存储卡的写保护		请在SD存储卡写保护解除的状态下使用	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2408)。

事件名称	指定文件不存在		事件代码	54012409Hex *1	
内容	FTP客户端通信指令指定的文件不存在				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	FTP客户端通信指令指定的文件不存在		请修正程序，使FTP客户端通信指令指定存在的文件	请编写程序，使FTP客户端通信指令指定存在的文件	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息				

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#2409)。

事件名称	禁止覆盖指定文件		事件代码	5401240AHex *1	
内容	FTP客户端通信指令指定了不覆盖同名文件，因此未传送				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响	
系统定义变量	变量名称		数据类型	名称	
	无		–	–	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	FTP客户端通信指令的覆盖指定为“不覆盖文件”，传送目标处存在与指定文件同名的文件，因此未传送		请使用FTP客户端通信指令将覆盖指定设定成 “覆盖文件”后，再次执行指令。 或者，请变更传送目标或传送源的文件名后，再次执行指令	请使用FTP客户端通信指令将覆盖指定设定成 “覆盖文件”。 或者，请将传送目标与传送源的文件设定成不同名	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息				

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#240A)。

事件名称	指定文件删除失败			事件代码	5401240BHex *1	
内容	FTP客户端通信指令未能删除已传送文件					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	FTP客户端通信指令的传送后文件删除指定为 “删除已传送文件”，但指定文件的属性为只读，因此未能删除		请使用FTP客户端通信指令将传送后文件删除指定设定成 “不删除已传送文件”后，再次执行指令。或者，请将传送源的文件属性变更成可写入后，再次执行指令		请使用FTP客户端通信指令将传送后文件删除指定设定成 “不删除已传送文件”。或者，请将传送源的文件属性设定成非只读	
	FTP客户端通信指令指定的文件正被其他应用程序使用，因此未能删除		请在FTP客户端通信指令指定的文件未被其他应用程序使用时执行指令		请勿在其他应用程序中使用FTP客户端通信指令指定的文件	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#240B)。

事件名称	指定文件访问失败			事件代码	5401240CHex *1	
内容	FTP客户端通信指令访问文件失败，因此FTP传送失败					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	FTP客户端通信指令指定的文件正被其他应用程序使用		请在FTP客户端通信指令指定的文件未被其他应用程序使用时执行指令		请勿在其他应用程序中使用FTP客户端通信指令指定的文件	
	FTP客户端通信指令指定的写入文件或目录已设置写保护		请解除FTP客户端通信指令指定的写入文件的写保护。或变更写入文件名		请勿对FTP客户端通信指令指定的写入文件进行写保护	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#240C)。

事件名称	IP地址设定错误			事件代码	5401240DHex *1	
内容	指令指定端口的IP地址设定与其他端口设定之间存在设定错误，因此未能执行指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	无影响		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令指定端口的网络地址与其他端口的网络地址重复		请修正以避免相应指令指定的网络地址与其他端口的网络地址重复。 或者，请事先变更其他端口的网络地址。		使用变更IP地址的指令时，请避免指令指定的网络地址与其他端口的网络地址重复	
	指令指定端口与其他端口的设定均为未使用		请将相应指令指定的端口设定修正为 “不使用”以外的设定。 或者，请事先变更其他端口的未使用设定。		使用变更IP地址的指令时，请确保指令指定的端口与其他端口设定均不为未使用	
附属信息	附属信息1: 异常的发生部位 附属信息2: 异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3: 发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4: 扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.10以上时发生的错误代码(16#240D)。

事件名称	NX信息异常			事件代码	54012C00Hex *1	
内容	以NX信息返回了错误响应代码					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	取决于异常的内容		请确认相应指令的输出变量“ErrorIDEx”的值，参阅本手册的NX信息异常代码的说明		取决于异常的内容。请参阅本手册的NX信息异常代码的说明	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C00)。

事件名称	NX信息资源超限		事件代码	54012C01Hex *1		
内容	超过可同时执行的NX信息指令的资源执行了指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	同时执行了超过32个的NX信息指令		请修正程序，以将同时执行的NX信息指令控制在32个以下		请编写程序，以将同时执行的NX信息指令控制在32个以下	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C01)。

事件名称	NX信息超时		事件代码	54012C02Hex *1		
内容	NX信息执行过程中发生了超时					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定的NX单元不存在		请进行修正，使单元指定与对象单元的构成等一致		请使单元指定与对象单元的构成等一致	
	NX信息因超时而关闭		请延长相应指令的输入变量“TimeOut”指定的响应监视时间		请在输入变量“TimeOut”中指定合适的响应监视时间后执行指令	
	对象单元的电源OFF		请确认对象单元的状态后正常启动		请确认对象单元的状态后正常启动	
	对象单元的通信停止					
	通信电缆连接器断开		请确认连接器是否正确嵌合，重新连接		请切实连接连接器	
	通信电缆断线		请更换通信电缆		无	
	干扰		干扰较多时，请采取防干扰措施		干扰较多时，请采取防干扰措施	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C02)。

事件名称	NX信息长度错误			事件代码	54012C03Hex *1	
内容	NX信息的长度错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	WriteDat或Path指定的大小过长		请修正程序，使WriteDat或Path指定的大小在限制范围内		请编写程序，使WriteDat或Path指定的大小在限制范围内	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C03)。

事件名称	NX信息网络异常(EtherCAT)			事件代码	54012C05Hex *1	
内容	NX信息线路上的EtherCAT通信发生了异常					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	NX信息线路上的EtherCAT通信发生了异常		请确认EtherCAT通信发生的异常，并在解除异常后执行相应指令		取决于异常的内容	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C05)。

事件名称	指定单元外部重启已执行			事件代码	54012C06Hex *1	
内容	执行指令时，已通过Sysmac Studio执行了重启					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	执行指令时，已通过Sysmac Studio执行了重启		已通过Sysmac Studio执行了重启时，则无需使用指令进行重启		运行时请勿通过Sysmac Studio进行重启	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C06)。

事件名称	非指令对象单元指定			事件代码	54012C07Hex *1	
内容	指定单元的从站节点地址连接了非相应指令对象的从站					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定单元的从站节点地址连接了非相应指令对象的从站		请连接网络构成信息指定的相应指令的对象单元		请勿将非相应指令对象的从站连接至指定单元的从站节点地址	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.05以上时发生的错误代码(16#2C07)。

事件名称	累计通电时间记录错误			事件代码	54012C08Hex *1	
内容	累计通电时间的读取失败。					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	非易失性存储器故障		请更换累计通电时间读取失败的单元		无	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.10以上时发生的错误代码(16#2C08)。

事件名称	过程数据对象设定不足			事件代码	54013461Hex	
内容	PDO映射不正确					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	未通过运动指令进行所需的PDO映射		请以相应指令进行必要的PDO映射。 必要的PDO映射请参阅相应指令的“功能说明”		请以需使用的指令进行必要的PDO映射。 以各指令进行的必要的PDO映射(伺服驱动器的设定)请参阅 “NJ/NX系列用户手册 运动控制篇(SBCE-363)”	
	对不带支持相应指令对象的对象设备执行了相应指令		存在不带支持相应指令功能的对象设备。 请参阅对象设备的手册，确认是否支持相应指令，修正程序以避免执行不支持的指令		请参阅对象设备的手册，编写程序以避免执行不支持的指令	
	针对映射了欧姆龙生产的EtherCAT编码器从站GX-EC02□□的轴，作为触发条件指定Z相(_mcEncoderMark)，启动了运动指令		请针对映射了欧姆龙生产的EtherCAT编码器从站GX-EC02□□的轴，将外部输入(_mcEXT)用于触发条件		请针对映射了欧姆龙生产的EtherCAT编码器从站GX-EC02□□的轴，将外部输入(_mcEXT)用于触发条件	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

事件名称	接收到设备错误			事件代码		54014800Hex *1
内容	接收到设备发出的错误响应					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	接收到设备发出的错误响应		设备返回的错误代码将输出至指令的输出变量ErrorType。请参阅对象设备的手册确认错误内容后，再采取措施		请参阅设备手册确认错误原因后，编写程序并执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4800)。

事件名称	指定单元不存在			事件代码		54014801Hex *1
内容	指定的单元不存在。					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定位置未连接/安装IO-Link主站单元		请在指定位置连接/安装IO-Link主站单元。或者，请指定已连接/安装IO-Link主站单元的位置		请在指定位置连接/安装IO-Link主站单元。或者，请指定已连接/安装IO-Link主站单元的位置	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4801)。

事件名称	信息处理数超限		事件代码	54014802Hex *1		
内容	IO-Link主站正在处理来自其它应用程序的信息，因此无法执行指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	IO-Link主站正在处理来自其它应用程序(指令执行或工具连接)的信息，因此无法执行指令		请重新执行指令		请通过应用程序(执行指令或连接工具)执行信息的排他性控制处理或增加重试次数	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4802)。

事件名称	指定单元状态异常		事件代码	54014803Hex *1		
内容	指定单元未处于可接收信息的状态					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定单元未处于可接收信息的状态		请重新执行指令		请确保在发生本异常时重新执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4803)。

事件名称	同时执行数超限			事件代码		54014804Hex *1
内容	超过了可同时执行的指令数					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	同时执行的NX信息指令和EtherCAT通信指令的总数超过了32个		请修正程序，以将同时执行的NX信息指令和EtherCAT通信指令的总数控制在32个以下		请编写程序，以将同时执行的NX信息指令和EtherCAT通信指令的总数控制在32个以下	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4804)。

事件名称	通信超时			事件代码		54014805Hex *1
内容	通信过程中发生了超时					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	信息响应时间比通信超时时间长		请计算信息响应时间，设定比信息响应时间长的通信超时时间		请计算信息响应时间，设定比信息响应时间长的通信超时时间	
	EtherCAT或IO–Link的电缆断线		请更换电缆		无	
	干扰		请采取防干扰措施		请采取防干扰措施	
	设备故障		请更换相应设备		无	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4805)。

事件名称	模式错误		事件代码	54014806Hex *1	
内容	指定IO-Link主站的端口非IO-Link模式				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	-	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。	
系统定义变量	变量名称		数据类型	名称	
	无		-	-	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指定IO-Link主站的端口非IO-Link模式		请将指定IO-Link主站的端口设定成IO-Link模式后，重新执行指令	请将指定IO-Link主站的端口设定成IO-Link模式后执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息				

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4806)。

事件名称	I/O电源OFF状态		事件代码	54014807Hex *1	
内容	未对指定IO-Link主站的端口供给I/O电源				
发生源	PLC功能模块	发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	-	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。	
系统定义变量	变量名称		数据类型	名称	
	无		-	-	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	未对指定IO-Link主站的端口供给I/O电源		请对指定IO-Link主站的端口供给I/O电源后执行指令	请确认已对指定IO-Link主站的端口供给I/O电源后，再执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息				

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4807)。

事件名称	发生核查异常			事件代码	54014808Hex *1	
内容	指定IO-Link主站的端口发生了核查异常或通信异常					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令。		
系统定义变量	变量名称		数据类型		名称	
	无		–		–	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指定IO-Link主站的端口发生了核查异常或通信异常		请解除异常后重新执行指令		请在未发生异常的状态下执行指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorType)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.12以上时发生的错误代码(16#4808)。

事件名称	超过电子齿轮分子设定范围			事件代码	54015420Hex	
内容	运动控制指令的输入变量 “RatioNumerator”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

事件名称	超过电子齿轮分母设定范围		事件代码		54015421Hex	
内容	运动控制指令的输入变量 “RatioDenominator”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过目标速度设定范围		事件代码		54015422Hex	
内容	运动控制指令的输入变量 “Velocity”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过加速度设定范围			事件代码	54015423Hex	
内容	运动控制指令的输入变量 “Acceleration”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过减速度设定范围			事件代码	54015424Hex	
内容	运动控制指令的输入变量 “Deceleration”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过跃度设定范围		事件代码	54015425Hex	
内容	运动控制指令的输入变量 “Jerk”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL	轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过扭矩倾斜设定范围		事件代码	54015427Hex	
内容	运动控制指令的输入变量 “TorqueRamp”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过主轴系数设定范围			事件代码		54015428Hex
内容	运动控制指令的输入变量 “MasterScaling”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过从轴系数设定范围			事件代码		54015429Hex
内容	运动控制指令的输入变量 “SlaveScaling”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过标准速度设定范围		事件代码	5401542AHex	
内容	运动控制指令的输入变量 “FeedVelocity”指定的参数超过范围				
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	标准速度(输入变量“FeedVelocity”)保持为初始值(0)		请为标准速度(输入变量“FeedVelocity”)指定正值		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过缓存模式选择范围		事件代码	5401542BHex	
内容	运动控制指令的输入变量 “BufferMode”指定的参数超过范围				
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过坐标系选择范围			事件代码		5401542CHex
内容	运动控制指令的输入变量 “CoordSystem”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过圆弧插补模式选择范围			事件代码		5401542DHex
内容	运动控制指令的输入变量 “CircMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过方向选择范围			事件代码	5401542EHex	
内容	运动控制指令的输入变量 “Direction”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过路径选择范围			事件代码	5401542FHex	
内容	运动控制指令的输入变量 “PathChoice”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过位置类型选择范围			事件代码	54015430Hex	
内容	运动控制指令的输入变量 “ReferenceType” 指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过移动方法选择范围			事件代码	54015431Hex	
内容	运动控制指令的输入变量 “MoveMode” 指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过过渡模式选择范围		事件代码	54015432Hex		
内容	运动控制指令的输入变量 “TransitionMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
	以 “BufferMode”指定“_mcAborting”、“_mcBuffered”，且为 “_mcTMCornerSuperimpose”指定 “TransitionMode”		以 “BufferMode”指定“_mcAborting”、“_mcBuffered”时，如需为 “_mcTMNone”指定“TransitionMode”，为“_mcTMCornerSuperimpose”指定“TransitionMode”，则请以“BufferMode”指定“_mcBlendingLow”、“_mcBlendingPrevious”、“_mcBlendingNext”、“_mcBlendingHigh”		以 “BufferMode”指定“_mcAborting”、“_mcBuffered”时，如需为 “_mcTMNone”指定“TransitionMode”，为“_mcTMCornerSuperimpose”指定“TransitionMode”，则请以“BufferMode”指定“_mcBlendingLow”、“_mcBlendingPrevious”、“_mcBlendingNext”、“_mcBlendingHigh”	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过持续方法选择范围		事件代码	54015433Hex		
内容	变更了运动控制指令的输入变量 “Continuous (Reserved)”的值					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了输入变量 “Continuous (Reserved)”的值		请修正程序，以不变更输入变量“Continuous (Reserved)”的值		请创建程序，以不变更输入变量“Continuous (Reserved)”的值	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过加减法运算方法选择范围			事件代码		54015434Hex
内容	运动控制指令的输入变量 “CombineMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过开始同步条件指定范围			事件代码		54015435Hex
内容	运动控制指令的输入变量 “LinkOption”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	主轴从轴相同		事件代码		54015436Hex	
内容	运动控制指令的输入变量 “Master”和 “Slave”指定的轴相同					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Master”和 “Slave”的参数相同		请修正参数，以避免相应指令的输入变量 “Master”和 “Slave”指定的轴相同		请为指令的输入变量 “Master”和 “Slave”指定不同的轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	主轴辅轴相同		事件代码		54015437Hex	
内容	运动控制指令的输入变量 “Master”和 “Auxiliary”指定的轴相同					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Master”和 “Auxiliary”的参数相同		请修正参数，以避免相应指令的输入变量 “Master”和 “Auxiliary”指定的轴相同		请为指令的输入变量 “Master”和 “Auxiliary”指定不同的轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	主轴/从轴 轴号非升序			事件代码		54015438Hex	
内容	运动控制指令的输入变量 “Master” 和 “Slave” 指定的轴号非升序						
发生源	PLC功能模块		发生源详情	指令		检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	_MC_AX[*].MFaultLvl.Active		BOOL			轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	指令的输入变量 “ReferenceType” 指定 “_mcLatestCommand” 时，指令的输入变量 “Master” 和 “Slave” 的参数非升序		相应指令的输入变量 “ReferenceType” 指定 “_mcLatestCommand” 时，请修正参数，以使相应指令的输入变量 “Master” 和 “Slave” 指定的轴号为升序。或以主轴位置类型选择指定 “_mcCommand”			变量 “ReferenceType” 指定 “_mcLatestCommand” 时，请将输入变量的主轴和从轴指定为升序	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	凸轮表指定错误			事件代码		54015439Hex	
内容	运动控制指令的输入变量 “CamTable” 指定的参数超过范围						
发生源	PLC功能模块		发生源详情	指令		检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	_MC_COM.MFaultLvl.Active		BOOL			MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL			轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	指令的输入变量 “CamTable” 指定了非凸轮数据变量		请修正参数，以使相应指令的输入变量 “CamTable” 指定的参数为凸轮数据变量			请为指令的输入变量 “CamTable” 指定凸轮数据变量	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

<table>
<tr><td>事件名称</td><td colspan="2">同步停止中</td><td>事件代码</td><td colspan="2">5401543AHex</td></tr>
<tr><td>内容</td><td colspan="5">执行了运动控制的同步控制指令，但并非可执行条件。</td></tr>
<tr><td>发生源</td><td colspan="2">PLC功能模块</td><td>发生源详情</td><td>指令</td><td>检测时间</td><td>启动指令时</td></tr>
<tr><td>异常的属性</td><td>重要程度</td><td>监控信息</td><td>恢复方法</td><td>–</td><td>日志类别</td><td>系统</td></tr>
<tr><td>发生后的影响</td><td>用户程序</td><td>继续</td><td>动作</td><td colspan="3">按照规格结束相应指令</td></tr>
<tr><td rowspan="2">系统
定义变量</td><td colspan="2">变量名称</td><td colspan="2">数据类型</td><td colspan="2">名称</td></tr>
<tr><td colspan="2">_MC_AX[*].MFaultLvl.Active</td><td colspan="2">BOOL</td><td colspan="2">轴正在发生轻度故障</td></tr>
<tr><td rowspan="2">发生原因及
其处理</td><td colspan="2">发生原因(推测原因)</td><td colspan="2">处理措施</td><td colspan="2">防止再次发生</td></tr>
<tr><td colspan="2">• 执行了MC_CamOut(凸轮动作解除)指令，但MC_CamIn(凸轮动作开始)指令未在运行
• 执行了MC_GearOut(齿轮动作解除)指令，但MC_GearIn(齿轮动作开始)指令、MC_GearInPos(位置指定齿轮动作)指令未在运行
• 执行了MC_Phasing(主轴相对值相位补偿)指令，但MC_CamIn(凸轮动作开始)指令、MC_GearIn(齿轮动作开始)指令、MC_GearInPos(位置指定齿轮动作)指令、MC_MoveLink(梯形模式凸轮)指令未在运行</td><td colspan="2">请修正程序，以确保执行相应指令时可执行</td><td colspan="2">请确保执行同步指令时可执行</td></tr>
<tr><td>附加信息</td><td colspan="5">附加信息1：异常的发生部位
附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号
附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示
附加信息4：扩展错误代码(ErrorIDEx)</td></tr>
<tr><td>注意事项/
备注</td><td colspan="5">发生异常后变更程序时，可能不会正确显示附加信息</td></tr>
</table>

事件名称	无法重启运动指令			事件代码	5401543BHex	
内容	重启了无法重启的运动控制指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	重启了无法重启的运动控制指令		请修正程序，以避免相应指令的输出变量 “Busy”变为FALSE前，输入变量 “Execute”处于上升沿		使用无法重启的指令时，请确保输入变量 “Execute”的启动条件中包含该指令的输出变量 “Busy”为FALSE。或确保再次启动指令前暂时停止指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法多重启动运动指令			事件代码	5401543CHex	
内容	对相同对象(MC通用/轴)执行了多个无法同时执行的功能					
发生源	PLC功能模块		发生源详情	指令	检测时间	开始多重启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对相同对象(MC通用/轴)执行了多个无法同时执行的功能		请确认相应指令的多重启动的规格，修正程序以确保不同时执行无法同时执行的指令		请确认待使用指令的多重启动的规格，确保不同时执行无法同时执行的指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	不符合轴类型			事件代码	5401543DHex	
内容	对编码器轴执行了动作指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对编码器轴执行了动作指令		请为相应指令的轴类型设定指定伺服轴或虚拟伺服轴，或修正程序以确保不对编码器轴执行相应动作指令		请确保对伺服轴或虚拟伺服轴执行动作指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法启动多轴协调动作中的指令			事件代码	5401543EHex	
内容	对多轴协调动作中的轴或轴组执行了动作指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	开始多重启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对多轴协调动作中的轴或轴组执行了动作指令		请修正程序，以确保在相应轴或相应轴组未进行多轴协调动作过程中执行相应指令		请向未进行多轴协调动作过程中的轴或轴组执行轴动作指令	
	对轴组有效状态下的轴组执行了MC_SetKinTransform(运动学转换设定)指令		请修正程序，以确保在禁用轴组时执行相应指令		禁用轴组时请勿执行相应指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	启动轴组无效状态下的多轴协调指令		事件代码	5401543FHex	
内容	对轴组无效状态下的轴组启动了多轴协调指令				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL	轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	对轴组无效状态下的轴组启动了多轴协调指令		请修正程序，以将轴组设为有效状态后执行相应指令。将轴组设为有效状态时，请执行MC_GroupEnable(启用轴组)指令	请将轴组设为有效状态后执行多轴协调动作指令。将轴组设为有效状态时，请执行MC_GroupEnable(启用轴组)指令	
	对轴组无效状态下的轴组启动了以下指令 • MC_MoveTimeAbsolute(时间指定绝对值位置指令)指令 • MC_SyncLinearConveyor(输送机同步动作开始)指令 • MC_SyncOut(同步动作解除)指令 • MC_RobotJog(轴组微动移动)指令				
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	无法启用轴组		事件代码	54015440Hex	
内容	MC_GroupEnable(启用轴组)指令执行失败				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL	轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	执行MC_GroupEnable(启用轴组)指令时，构成轴中存在未处于停止状态的轴		请修正程序，以确保在构成轴均处于停止状态下执行MC_GroupEnable(启用轴组)指令。轴变量的 “Status.Disabled” 或 “Status.Standstill” 为TRUE时，该轴处于停止状态	请创建程序，以确保在构成轴均处于停止状态下执行MC_GroupEnable(启用轴组)指令。轴变量的 “Status.Disabled” 或 “Status.Standstill” 为TRUE时，该轴处于停止状态	
	执行MC_GroupEnable(启用轴组)指令时，构成轴中存在正在执行MC_TouchProbe(启用外部锁定)指令的轴		请修正程序，以确保在所有构成轴未执行MC_TouchProbe(启用外部锁定)指令时执行MC_GroupEnable(启用轴组)指令	请创建程序，以确保在所有构成轴未执行MC_TouchProbe(启用外部锁定)指令时执行MC_GroupEnable(启用轴组)指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	无法运行(伺服OFF)轴动作指示			事件代码	54015441Hex	
内容	对伺服OFF中的轴执行了动作指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对伺服OFF中的轴执行了动作指令		请修正程序，以确保在伺服ON后执行相应指令		请确保在伺服ON后执行轴动作指令	
	对未建立EtherCAT的过程数据通信的轴执行了通过MC_Home(原点复位)指令或MC_HomeWithParameter(参数指定原点复位)指令进行的原点预设。		EtherCAT主站的系统定义变量“_EC_PDSlavTbl”(过程数据通信中从站表)变为FALSE时，请排除其原因，“_EC_PDSlavTbl”变为TRUE后，通过MC_Home(原点复位)指令或MC_HomeWithParameter(参数指定原点复位)指令执行原点预设		请创建程序，以确保在控制器的电源ON或下载后，抑或从站的通信异常解除、取消、再次添加、不启用、启用后，通过MC_Home(原点复位)指令或MC_HomeWithParameter(参数指定原点复位)指令执行原点预设时，确认EtherCAT主站的系统定义变量“_EC_PDSlavTbl”(过程数据通信中从站表)变为TRUE后再执行	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	构成轴强制停止中错误			事件代码	54015442Hex	
内容	针对构成轴，向已执行MC_Stop(强制停止)指令的轴组执行了动作指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	针对构成轴，向已执行MC_Stop(强制停止)指令的轴组执行了动作指令		请将构成轴的强制停止指令的输入变量 “Execute”设为FALSE并停止执行，执行异常解除后赋予动作指令		请将构成轴强制停止指令的输入变量 “Execute”设为FALSE并退出后，执行轴组的动作指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	多重启动运动指令数超限			事件代码		54015443Hex
内容	通过缓存模式Buffered、Blending缓存的运动控制指令的缓存数超限					
发生源	PLC功能模块		发生源详情	指令	检测时间	开始多重启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	正在执行的轴指令和缓存中的轴指令的总数超过了 “2”		请修正程序，以确保执行相应指令时缓存数不超过上限		请确保正在执行的轴组指令和缓存中的轴组指令的总数不超过 “2”	
	正在执行的轴组指令和缓存中的轴组指令的总数超过了 “8”				请确保正在执行的轴组指令和缓存中的轴组指令的总数不超过 “8”	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	移动量不足			事件代码		54015444Hex
内容	多重启动/重启定位指令时，无法执行以指定减速度或加速度指定的动作					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	如果已将 “加减速超限”设定为“作为异常停止”，则无法在多重启动/重启定位指令时，在指定减速度/加速度的条件下在目标位置停止		请根据相应指令的动作规格修正程序，以确保定位动作不会在多重启动/重启时指定的减速度或加速度下超过目标位置。 或将 “加减速超限”变更设定为非 “作为异常停止”		请先确认待使用指令的动作规格，再创建程序以避免发生本异常。 或将 “加减速超限”设定为非 “作为异常停止”	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	用于达到混合中继速度的移动量不足		事件代码	54015445Hex		
内容	用于中继速度加减速的移动量不足					
发生源	PLC功能模块		发生源详情	指令	检测时间	开始多重启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将 “加减速超限”设定为 “作为异常停止”时，用于将当前指令加减速至中继速度的移动量不足		请根据相应指令的动作规格修正程序，以避免移动量不足。 或将 “加减速超限”变更设定为非 “作为异常停止”		请先确认待使用指令的动作规格，再创建程序以避免发生本异常。 或将 “加减速超限”设定为非 “作为异常停止”	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	梯形模式凸轮等速移动量不足		事件代码	54015446Hex		
内容	主轴的等速移动量小于 “0”					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_MoveLink(梯形模式凸轮)指令下，主轴的等速移动量小于 “0”		请修正程序，以满足(主轴移动距离≥主轴加速移动距离 + 主轴减速移动距离)		请先确认待使用指令的动作规格，再创建程序以避免发生本异常	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	位置指定齿轮动作目标速度不足			事件代码		54015447Hex
内容	MC_GearInPos(位置指定齿轮动作)指令下，从轴的 “目标速度”较小，因此达不到所需速度					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_GearInPos(位置指定齿轮动作)指令下，输入变量 “Velocity(目标速度)”的值小于(启动指令时的主轴速度齿轮比)		请根据相应指令的动作规格设定输入变量 “Velocity(目标速度)”的值，使其大于(指令启动时的主轴速度齿轮比)		请先确认待使用指令的动作规格，再创建程序以避免发生本异常	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	圆弧插补起点终点相同			事件代码		54015448Hex
内容	MC_MoveCircular2D(2轴圆弧插补)指令下，指定半径指定方式时起点和终点的位置相同。或指定通过点指定方式时起点、终点及通过点的位置相同					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_MoveCircular2D(2轴圆弧插补)指令下，指定了半径指定方式，起点和终点的位置相同		请修正程序，以避免在相应指令的起点和终点位置相同的状态下指定半径指定方式		执行半径指定方式的圆弧插补时，请确保起点和终点不同	
	MC_MoveCircular2D(2轴圆弧插补)指令下，指定了通过点指定方式，起点、终点及通过点的位置相同		请修正程序，以避免在相应指令的起点和终点与通过点位置相同的状态下指定通过点指定方式		执行通过点指定方式的圆弧插补时，请确保起点和终点与通过点不同	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过圆弧插补中心点指定位置范围			事件代码	54015449Hex	
内容	MC_MoveCircular2D(2轴圆弧插补)指令下，指定中心点指定方式时，中心点的位置指定超过容许范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_MoveCircular2D(2轴圆弧插补)指令下，指定了中心点指定方式，起点与中心点的距离、终点与中心点的距离之差超过了轴组设定的“中心点补偿容许率”指定的容许范围		请进行修正以设定中心点，使其满足起点与中心点的距离、终点与中心点的距离之差小于轴组设定的“中心点补偿容许率”指定的容许范围		请满足起点与中心点的距离、终点与中心点的距离之差小于轴组设定的“中心点补偿容许率”指定的容许范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	计数模式设定导致的指令启动异常			事件代码	5401544AHex	
内容	对计数模式已设定为旋转模式的轴，执行了旋转模式下无法使用的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	以旋转模式下无法使用的指令使用了计数模式已设定为旋转模式的轴		请将相应指令已指定的相应轴的计数模式的设定变更为线性模式		请确认可执行待使用指令的计数模式，设定轴的计数模式。	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过参数选择范围			事件代码		5401544CHex
内容	运动控制指令的输入变量 “ParameterNumber”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过停止方法选择范围			事件代码		5401544DHex
内容	运动控制指令的输入变量 “StopMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过触发输入条件的锁定ID选择范围		事件代码	5401544EHex	
内容	运动控制指令的输入变量 “TriggerInput::LatchID”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过MC设定写入的设定范围		事件代码	5401544FHex	
内容	运动控制指令的输入变量 “SettingValue”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_COM.MFaultLvl.Active		BOOL	MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
	参数指定和设定值的数据类型不一致		请进行修正，以使参数指定和设定值的数据类型一致	请使参数指定和设定值的数据类型一致	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过触发输入条件的模式选择范围		事件代码	54015450Hex		
内容	运动控制指令的输入变量 “TriggerInput::Mode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过触发输入条件的驱动触发输入信号选择范围		事件代码	54015451Hex		
内容	运动控制指令的输入变量 “TriggerInput::InputDrive”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(轴指定)		事件代码		54015453Hex	
内容	变更了重启运动控制指令时无法变更的输入变量 “Axis”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(缓存模式选择)		事件代码		54015454Hex	
内容	变更了重启运动控制指令时无法变更的输入变量 “BufferMode”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(方向选择)			事件代码		54015455Hex
内容	变更了重启运动控制指令时无法变更的输入变量 “Direction”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(重复模式)			事件代码		54015456Hex
内容	变更了重启运动控制指令时无法变更的输入变量 “Periodic”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(轴组指定)		事件代码	54015457Hex		
内容	变更了重启运动控制指令时无法变更的输入变量 “AxesGroup” 的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(跳动设定)		事件代码	54015458Hex		
内容	变更了重启运动控制指令时无法变更的输入变量 “Jerk” 的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(主轴)			事件代码		54015459Hex
内容	变更了重启运动控制指令时无法变更的输入变量 “Master”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(MasterOffset)			事件代码		5401545AHex
内容	变更了重启运动控制指令时无法变更的输入变量 “MasterOffset”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(MasterScaling)		事件代码	5401545BHex		
内容	变更了重启运动控制指令时无法变更的输入变量 “MasterScaling”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(MasterStartDistance)		事件代码	5401545CHex		
内容	变更了重启运动控制指令时无法变更的输入变量 “MasterStartDistance”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(Continuous)			事件代码		5401545DHex
内容	变更了重启运动控制指令时无法变更的输入变量 “Continuous”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(MoveMode)			事件代码		5401545EHex
内容	变更了重启运动控制指令时无法变更的输入变量 “MoveMode”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	辅轴指定错误			事件代码	5401545FHex	
内容	运动控制指令的输入变量 “Auxiliary”指定的轴不存在					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Auxiliary”指定的轴不存在的变量		请将相应指令已指定的轴修正为存在的变量		为指令的输入参数指定变量时，请指定存在的变量	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	轴指定错误			事件代码	54015460Hex	
内容	运动控制指令的输入变量 “Axis”指定的轴不存在					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Axis”指定的轴不存在的变量		请将相应指令已指定的轴修正为存在的变量		为指令的输入参数指定变量时，请指定存在的变量	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	轴组指定错误			事件代码		54015461Hex
内容	运动控制指令的输入变量 “AxesGroup” 指定的轴组不存在或不是使用轴组					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “AxesGroup” 指定的轴组不存在的变量		请将相应指令已指定的轴组修正为存在的轴组变量		为指令的输入参数指定变量时，请指定存在的变量	
	指令的输入变量 “AxesGroup” 指定的轴组未设定为使用轴组		请将相应指令已指定的轴组设定为使用轴组		请将指定为输入变量 “AxesGroup” 的轴组设定为使用轴组	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	主轴指定错误			事件代码		54015462Hex
内容	运动控制指令的输入变量 “Master” 指定的轴不存在或不是同步主轴					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Master” 指定的轴不存在的变量		请将相应指令已指定的轴修正为存在的变量		为指令的输入参数指定变量时，请指定存在的变量	
	MC_Phasing(主轴相对值相位补偿)指令时，输入变量 “Master” 指定的轴不是同步主轴		请将MC_Phasing(主轴相对值相位补偿)指令的输入变量 “Master” 输入的变量修正为同步指令的主轴已指定的轴		请将MC_Phasing(主轴相对值相位补偿)指令的输入变量 “Master” 输入的变量设定为同步指令的主轴已指定的轴	
	分配主轴和从轴的任务不同		请将相应指令的输入变量 “Master” 和 “Slave” 中输入的轴，分配至同一任务		主轴、从轴请指定为分配在同一任务中的轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(SlaveOffset)		事件代码	54015463Hex		
内容	变更了重启运动控制指令时无法变更的输入变量 “SlaveOffset”					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(SlaveScaling)		事件代码	54015464Hex		
内容	变更了重启运动控制指令时无法变更的输入变量 “SlaveScaling”					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(StartPosition)			事件代码		54015465Hex
内容	变更了重启运动控制指令时无法变更的输入变量 “StartPosition”					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	原点未确定状态下的指令启动异常			事件代码		54015466Hex
内容	原点未确定状态下执行了高速原点复位或插补指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	原点未确定状态下执行了高速原点复位		请通过原点复位确定原点后执行相应的高速原点复位		请在通过原点复位确定原点的状态下执行高速原点复位指令	
	向包含原点未确定状态的构成轴执行了插补指令		请通过原点复位确定所有构成轴的原点后执行相应的插补指令		请通过原点复位确定所有构成轴的原点后执行插补指令	
	向包含原点未确定状态逻辑轴的轴组执行了以下机器人指令 • MC_SetKinTransform(运动学转换设定)指令 • MC_SetKinTransform(运动学转换设定)指令 • MC_MoveTimeAbsolute(时间指定绝对值位置指令)指令 • MC_SyncLinearConveyor(输送机同步动作开始)指令 • MC_SyncOut(同步动作解除)指令 • MC_GroupMon(轴组监控)指令 • MC_RobotJog(轴组微动移动)指令					
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附加信息 • 重新通过执行原点复位设为原点确定状态，以实现执行原点复位，执行 “当前位置变更”指令后进入原点未确定状态					

事件名称	无法重启运动指令(位置类型)			事件代码	54015467Hex	
内容	变更了重启运动控制指令时无法变更的输入变量 “ReferenceType”					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	未使用轴指定(主轴)			事件代码	54015468Hex	
内容	运动控制指令指定的主轴为未使用轴					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	运动控制指令指定的主轴为未使用轴		请将相应指令已指定的主轴修正为使用轴		请为运动控制指令指定的主轴指定使用轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过起始位置设定范围			事件代码		54015469Hex
内容	运动控制指令的输入变量 “FirstPosition”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过终止位置设定范围			事件代码		5401546AHex
内容	运动控制指令的输入变量 “LastPosition”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	起始位置/终止位置、大小关系错误(线性模式)		事件代码	5401546BHex		
内容	运动控制指令的输入变量 “LastPosition”指定的参数值小于输入变量 “FirstPosition”指定的参数值					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	计数模式为线性模式时，指令的输入参数 “LastPosition”的值小于“FirstPosition”的值		请进行修正，以满足相应指令指定的 “LastPosition”的值大于“FirstPosition”的值。 或将计数模式变更为旋转模式		请创建程序，以满足相应指令指定的 “LastPosition”的值大于“FirstPosition”的值。或确认相应轴的计数模式已变为旋转模式	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴同步位置设定范围		事件代码	5401546CHex		
内容	运动控制指令的输入变量 “MasterSyncPosition”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过从轴同步位置设定范围			事件代码	5401546DHex	
内容	运动控制指令的输入变量 “SlaveSyncPosition”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	触发输入条件的锁定ID重复			事件代码	5401546EHex	
内容	多个指令已重复运动控制指令指定的锁定ID					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_TouchProbe(启用外部锁定)指令、MC_MoveLink(梯形模式凸轮)指令、MC_MoveFeed(中断标准定位)指令已同时使用相同的锁定ID		请修正程序，以避免相应指令和其他指令同时使用相同的锁定ID。请确保使用不同的锁定ID或避免同时执行使用相同锁定ID的指令。 正在执行MC_Home(原点复位)指令或MC_HomeWithParameter(参数指定原点复位)指令即正在使用锁定功能1/2		请避免MC_TouchProbe(启用外部锁定)指令、MC_MoveLink(梯形模式凸轮)指令、MC_MoveFeed(中断标准定位)指令同时使用相同的锁定ID	
	试图通过MC_AbortTrigger(不启用外部锁定)指令中止正在以非MC_TouchProbe(启用外部锁定)指令使用的锁定		请避免通过外部锁定无效指令中止非外部锁定有效指令使用过程中的锁定		请避免非外部锁定有效指令使用过程中的锁定执行外部锁定无效指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附加信息 • 变更锁定ID时，请确认其他指令未同时使用该锁定ID					

事件名称	超过跃度超调值范围		事件代码	5401546FHex	
内容	运动控制指令的输入变量 “JerkFactor”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过加减速度超调值范围		事件代码	54015470Hex	
内容	运动控制指令的输入变量 “AccFactor”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过起始位置方式指定范围			事件代码		54015471Hex
内容	运动控制指令的输入变量 “StartMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法重启运动指令(起始位置方式)			事件代码		54015472Hex
内容	变更了重启运动控制指令时无法变更的输入变量 “StartMode”的参数					
发生源	PLC功能模块		发生源详情	指令	检测时间	重启指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了重启时无法变更的输入变量的参数		请修正程序，以避免重启相应指令时相应输入变量的参数发生变化		请手动确认待使用的运动控制指令的各输入变量在重启时可否变更，并创建程序，以避免无法变更的输入变量在重启时输入参数发生变化	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	未使用轴指定(辅轴)			事件代码		54015474Hex
内容	运动控制指令的输入变量 “Auxiliary”指定的轴为未使用轴					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Auxiliary”指定的轴为未使用轴		请将相应指令已指定的轴设定为使用轴。或修正参数，以指定使用轴		请将指令指定的轴设为使用轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	位置指定齿轮指定值异常			事件代码		54015475Hex
内容	无法以运动控制指令输入的速度/加速度/减速度进行同步动作					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	无法以指令输入的速度/加速度/减速度进行指定的同步动作		请根据MC_GearInPos(位置指定齿轮动作)指令的动作规格修正程序，以设定为可同步动作		请确认待使用指令的处理，以设定为可同步动作	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	位置指定齿轮主轴零速			事件代码		54015476Hex
内容	启动运动控制指令时主轴的速度为 “0”					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	启动指令时主轴的速度为 “0”		请修正程序，以避免启动相应指令时主轴的速度变为 “0”		请创建程序，以避免执行指令时主轴的速度变为 “0”	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过目标位置设定范围			事件代码		54015478Hex
内容	运动控制指令的输入变量 “Position”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
	对于旋转模式的轴，目标位置超过环设定的范围		对于旋转模式的轴，请修正目标位置，以将其控制在环设定的范围内		对于旋转模式的轴，请将目标位置控制在环设定的范围内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过移动距离范围		事件代码	54015479Hex		
内容	运动控制指令的输入变量 “Distance”指定的参数超过范围或加上 “Distance”值后的目标位置超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将指令输入参数的绝对值转换为脉冲单位时超过40位的范围		请修正相应指令的输入变量“Distance”指定的输入参数，以将移动距离和目标位置控制在范围内		请创建程序，已将指令执行的移动距离和目标位置控制在范围内	
	对于线性模式的轴，将加上移动距离后的目标位置转换为脉冲单位时超过带符号40位的范围					
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过凸轮表起点位置设定范围		事件代码	5401547AHex		
内容	运动控制指令的输入变量 “StartPosition”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过凸轮动作(主轴跟踪)起始位置设定范围		事件代码	5401547BHex		
内容	运动控制指令的输入变量 “MasterStartDistance”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	圆弧插补半径指定异常		事件代码	5401547CHex		
内容	MC_MoveCircular2D(2轴圆弧插补)指令下，指定半径指定方式时无法以指定的半径创建圆弧的轨迹					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_MoveCircular2D(2轴圆弧插补)指令下，指定了半径指定方式，无法以指定的半径创建圆弧的轨迹		请修正半径，以确保可生成圆弧的轨迹		请确认待使用指令的处理，并设定半径，以确保可生成圆弧的轨迹	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	圆弧插补半径溢出		事件代码	5401547DHex		
内容	MC_MoveCircular2D(2轴圆弧插补)指令下，通过点指定方式/中心点指定方式时，圆弧的半径超过最大值					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_MoveCircular2D(2轴圆弧插补)指令下，通过点指定方式/中心点指定方式时，将圆弧的半径转换为脉冲单位时超过40位的范围		请根据相应指令的动作规格修正输入参数，以避免将圆弧半径转换为脉冲单位时超过40位的范围。 指定通过点时：起点、通过点、终点 指定中心点时：起点、终点、中心点		请确认待使用指令的处理，设定输入参数，以避免将圆弧半径转换为脉冲单位时超过40位的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	• 发生异常后变更程序时，可能不会正确显示附加信息 • 半径指定方式时，如果半径超过最大值，则发生 “通过点位置/中心位置/超过半径指定范围”异常					

事件名称	超过圆弧轴指定范围		事件代码	5401547EHex		
内容	运动控制指令的输入变量 “CircAxes”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免相应指令输入的参数超过输入变量的范围	
	“CircAxes”指定的轴未包含在轴组设定的构成轴中		请变更设定，以使轴组构成中包含指定为 “CircAxes”的轴		请将指定为 “CircAxes”的轴包含在轴组构成中	
	“CircAxes”的2轴已指定相同轴		请进行修正，以避免指定为“CircAxes”的2轴为相同的轴		请创建程序，以避免指定为“CircAxes”的2轴相同	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	辅轴/从轴、轴号非升序			事件代码		5401547FHex
内容	运动控制指令指定的输入变量 “Auxiliary”和 “Slave”的参数值未以升序排列					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Auxiliary”和 “Slave”的参数非升序		请进行修正，以使相应指令的 “Auxiliary”和 “Slave”的输入参数指定的轴号为升序		请创建程序，以使 “Auxiliary”和 “Slave”指定的轴号为升序	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	凸轮表属性更新中数据升序异常			事件代码		54015480Hex
内容	判定有效数据数计算中相位非升序。或判定计算后有效数据数为 “0”					
发生源	PLC功能模块		发生源详情	指令	检测时间	执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	判定有效数据数计算中相位非升序		请修正凸轮表，以使相位为升序		请确保凸轮表数据的相位为升序	
	判定计算后有效数据数为 “0”		请修正凸轮表的数据，以包含非 “0”的相位		请设定凸轮表的数据，以包含非 “0”的相位	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过MC设定写入的对象范围		事件代码	54015481Hex		
内容	运动控制指令的输入变量 “Target”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴移动距离指定范围		事件代码	54015482Hex		
内容	运动控制指令的输入变量 “MasterDistance”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴加速移动距离指定范围			事件代码		54015483Hex	
内容	运动控制指令的输入变量 “MasterDistanceInACC”指定的参数超过范围						
发生源	PLC功能模块		发生源详情	指令		检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	_MC_AX[*].MFaultLvl.Active		BOOL			轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围			请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	超过主轴减速移动距离指定范围			事件代码		54015484Hex	
内容	运动控制指令的输入变量 “MasterDistanceInDEC”指定的参数超过范围						
发生源	PLC功能模块		发生源详情	指令		检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	_MC_AX[*].MFaultLvl.Active		BOOL			轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围			请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

事件名称	超过执行模式选择范围		事件代码		54015487Hex	
内容	运动控制指令的输入变量 “ExecutionMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过轴间偏差容许值范围		事件代码		54015488Hex	
内容	运动控制指令的输入变量 “PermittedDeviation”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	通过点位置/中心位置/超过半径指定范围		事件代码	54015489Hex		
内容	运动控制指令的输入变量 “AuxPoint”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	通过点指定或中心指定时，将“AuxPoint”的值转换为脉冲单位时超过带符号40位的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
	半径指定时，将 “AuxPoint[0]”的绝对值转换为脉冲单位时超过40位的范围					
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过终点指定范围		事件代码	5401548AHex		
内容	运动控制指令的输入变量 “EndPoint”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过从轴移动距离指定范围			事件代码		5401548BHex
内容	运动控制指令的输入变量 “SlaveDistance” 指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将指令的输入参数转换为脉冲单位时，已超过40位的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过相位补偿量范围			事件代码		5401548CHex
内容	运动控制指令的输入变量 “PhaseShift” 指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将指令输入参数的绝对值转换为脉冲单位时超过40位的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过标准距离范围			事件代码		5401548DHex
内容	运动控制指令的输入变量 “FeedDistance” 指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将指令输入参数的绝对值转换为脉冲单位时超过40位的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	辅轴/从轴相同			事件代码		5401548EHex
内容	运动控制指令的输入变量 “Auxiliary” 和 “Slave” 指定的轴相同					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Auxiliary” 和 “Slave” 的参数相同		请修正参数，以避免相应指令的输入变量 “Auxiliary” 和 “Slave” 指定的轴相同		请为指令输入变量的辅轴和从轴指定不同的轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过相对位置选择范围		事件代码	5401548FHex	
内容	运动控制指令的输入变量 “Relative” 指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过凸轮过渡指定选择范围		事件代码	54015490Hex	
内容	运动控制指令的输入变量 “CamTransition” 指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过同步控制解除模式选择范围		事件代码	54015491Hex		
内容	运动控制指令的输入变量 “OutMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	无法执行启用外部锁定指令		事件代码	54015492Hex		
内容	非CSP模式的控制模式时，输入变量 “StopMode”指定为 “_mcImmediateStop(立即停止)”，执行了驱动模式的MC_TouchProbe(启用外部锁定)指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	非CSP模式的控制模式或伺服OFF时，输入变量 “StopMode”指定为 “_mcImmediateStop(立即停止)”，执行了驱动模式的MC_TouchProbe(启用外部锁定)指令		指定 “_mcImmediateStop(立即停止)”，执行驱动模式的启用外部锁定指令时，请将控制模式设为CSP模式		指定 “_mcImmediateStop(立即停止)”，执行驱动模式的启用外部锁定指令时，请将控制模式设为CSP模式	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴偏移设定范围		事件代码	54015493Hex	
内容	运动控制指令的输入变量 “MasterOffset”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过从轴偏移设定范围		事件代码	54015494Hex	
内容	运动控制指令的输入变量 “SlaveOffset”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过指令当前位置计数选择范围			事件代码		54015495Hex
内容	运动控制指令的输入变量 “CmdPosMode”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴齿轮比分子范围			事件代码		54015496Hex
内容	运动控制指令的输入变量 “RatioNumeratorMaster”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴齿轮比分母范围			事件代码	54015497Hex	
内容	运动控制指令的输入变量 “RatioDenominatorMaster”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过辅轴齿轮比分子范围			事件代码	54015498Hex	
内容	运动控制指令的输入变量 “RatioNumeratorAuxiliary”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过辅轴齿轮比分母范围			事件代码		54015499Hex
内容	运动控制指令的输入变量 “RatioDenominatorAuxiliary”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过主轴位置类型选择范围			事件代码		5401549AHex
内容	运动控制指令的输入变量 “ReferenceTypeMaster”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	超过辅轴位置类型选择范围		事件代码	5401549BHex	
内容	运动控制指令的输入变量 “ReferenceTypeAuxiliary”指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过目标位置环计数器范围		事件代码	5401549CHex	
内容	已执行指令的目标位置超过环计数器的范围，因此无法动作				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	设定为环计数器范围内不含0时，执行了高速原点复位		设定为环计数器范围内不含0时，无法执行高速原点复位。请修正程序，以避免执行高速原点复位。或进行变更，以设定为相应轴的环计数器范围内不含0	设定为环计数器范围内不含0时，无法执行高速原点复位。请创建程序，以避免执行高速原点复位。或设定为相应轴的环计数器范围内不含0	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

事件名称	超过轴组构成轴设定范围		事件代码	5401549DHex *1		
内容	运动控制指令的输入变量 “Axes”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
	分配轴组构成轴的任务不同		请将相应指令的输入变量 “Axes”指定的轴全部分配至同一任务		轴组构成轴请指定为分配在同一任务中的轴	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.01的CPU单元追加的错误代码(16#549D)。

事件名称	超过轴使用设定范围		事件代码	5401549EHex *1		
内容	运动控制指令的输入变量 “AxisUse”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.04的CPU单元追加的错误代码(16#549E)。

事件名称	超过原点复位参数设定范围		事件代码	54015700Hex *1	
内容	运动控制指令的输入变量 “HomingParameter” 指定的参数超过范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_COM.MFaultLvl.Active		BOOL	MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围	请避免指令的输入参数超过输入变量的范围	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

*1 单元版本Ver.1.03的CPU单元追加的错误代码(16#5700)。

事件名称	轴未使用切换异常		事件代码	54015702Hex *1	
内容	向非处于停止状态或处于指令速度饱和状态的轴执行了MC_ChangeAxisUse(轴使用变更)指令				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_AX[*].MFaultLvl.Active		BOOL	轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	向非处于停止状态或处于指令速度饱和状态的轴执行了MC_ChangeAxisUse(轴使用变更)指令		执行异常解除，轴已停止时，请在指令速度处于非饱和的状态下执行MC_ChangeAxisUse(轴使用变更)指令。 轴变量的 “Status.Disabled” 或 “Status.Standstill” 为TRUE时，该轴处于停止状态。轴变量的 “Details.VelLimit” 为TRUE时，该轴处于指令速度饱和状态	轴已停止时，请在指令速度处于非饱和的状态下执行MC_ChangeAxisUse(轴使用变更)指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

*1 单元版本Ver.1.04的CPU单元追加的错误代码(16#5702)。

事件名称	无法进行轴使用切换		事件代码	54015703Hex *1		
内容	试图超出使用实轴最大数量或使用运动控制伺服轴最大数量执行MC_ChangeAxisUse(轴使用变更)指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	试图超出使用实轴最大数量执行MC_ChangeAxisUse(轴使用变更)指令		请修正程序，以免使用的CPU单元的使用实轴超过最大数量		请编写程序，以免使用的CPU单元的使用实轴超过最大数量	
	试图超出使用运动控制伺服轴最大数量执行MC_ChangeAxisUse(轴使用变更)指令		请修正程序，以免使用的CPU单元的使用运动控制伺服轴超过最大数量		请编写程序，以免使用的CPU单元的使用运动控制伺服轴超过最大数量	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5703)。

事件名称	轴使用切换时运动控制参数设定异常		事件代码	54015720Hex *1		
内容	切换为使用轴的轴的运动控制参数设定错误					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	MC_ChangeAxisUse(轴使用变更)指令下，从未使用轴切换到使用轴的轴的运动控制参数设定错误		请使用Sysmac Studio将发生异常的轴的[轴使用]变更为[使用轴]，确认异常部位并修正。如果未发生异常，请再次变更为[未使用轴]并下载		请确认 “使用轴”的设定正常动作后作为[未使用轴]下载	
	下载运动控制参数设定过程中电源断开		请从Sysmac Studio下载运动控制参数设定		保存参数设定过程中请勿使电源断开	
	非易失性存储器的故障或非易失性存储器的寿命		进行上述处理后仍未解除本异常时，请更换CPU单元		无	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

*1 单元版本Ver.1.04的CPU单元追加的错误代码(16#5720)。

事件名称	未设定轴使用切换时所需的过程数据对象		事件代码	54015721Hex *1	
内容	未设定切换为使用轴的轴类型的所需对象				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	
系统定义变量	变量名称		数据类型	名称	
	_MC_COM.MFaultLvl.Active		BOOL	MC通用 正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施	防止再次发生	
	切换为使用轴的轴类型的所需对象未设定PDO映射		请在Sysmac Studio的[编辑PDO映射设定]中设定发生异常的轴的所需对象。 所需对象请参阅 “NJ/NX系列 指令基准手册 运动篇 (SBCE-364)”。	请确认 “使用轴”的设定正常动作后作为[未使用轴]下载	
	下载运动控制参数设定过程中电源断开		请从Sysmac Studio下载运动控制参数设定	保存参数设定过程中请勿使电源断开	
	非易失性存储器的故障或非易失性存储器的寿命		进行上述处理后仍未解除本异常时，请更换CPU单元	无	
	对[轴使用]设定为[未使用轴(无法切换为使用轴)]的轴执行了MC_ChangeAxisUse(轴使用变更)指令		请修正程序，以免对[轴使用]设定为[未使用轴(无法切换为使用轴)]的轴执行MC_ChangeAxisUse(轴使用变更)指令	请编写程序，以免对[轴使用]设定为[未使用轴(无法切换为使用轴)]的轴执行MC_ChangeAxisUse(轴使用变更)指令	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)				
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息				

*1 单元版本Ver.1.04的CPU单元追加的错误代码(16#5721)。

事件名称	反馈位置正在发生溢出/下溢			事件代码		54015722Hex *1
内容	反馈位置正在发生溢出/下溢时启动了无法执行的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	反馈位置正在发生溢出或下溢时启动了无法执行的指令		请执行异常解除，通过 “当前位置变更” 或 “原点复位” 解除溢出或下溢状态		请编写程序，以避免发生溢出或下溢	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5722)。

事件名称	开关结构体的机架编号超过设定范围			事件代码		54015723Hex *1
内容	运动控制指令的输入输出变量 “Switches” 指定的 “TrackNumber” 值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5723)。

事件名称	开关结构体的ON开始位置超过设定范围		事件代码	54015724Hex *1		
内容	运动控制指令的输入输出变量 “Switches” 指定的 “FirstOnPosition” 值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5724)。

事件名称	开关结构体的ON结束位置超过设定范围		事件代码	54015725Hex *1		
内容	运动控制指令的输入输出变量 “Switches” 指定的 “LastOnPosition” 值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5725)。

事件名称	开关结构体的方向选择超过范围		事件代码	54015726Hex *1		
内容	运动控制指令的输入输出变量 “Switches”指定的 “AxisDirection”值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5726)。

事件名称	开关结构体的开关模式选择超过范围		事件代码	54015727Hex *1		
内容	运动控制指令的输入输出变量 “Switches”指定的 “CamSwitchMode”值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5727)。

事件名称	开关结构体的ON时间超过设定范围			事件代码	54015728Hex *1	
内容	运动控制指令的输入输出变量 “Switches” 指定的 “Duration” 值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5728)。

事件名称	机架选项结构体的ON时刻补偿超过设定范围			事件代码	54015729Hex *1	
内容	运动控制指令的输入输出变量 “TrackOptions” 指定的 “OnCompensation” 值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5729)。

事件名称	机架选项结构体的OFF时刻补偿超过设定范围		事件代码	5401572AHex *1		
内容	运动控制指令的输入输出变量 “TrackOptions”指定的 “OffCompensation”值超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的结构要素值超过范围		请修正数值，以避免相应指令的输入输出变量中指定的结构体型变量的结构要素超过有效范围		请避免指令的输入输出变量中指定的结构体型变量的结构要素值超过有效范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#572A)。

事件名称	开关结构体型变量的数组元素数超过范围		事件代码	5401572BHex *1		
内容	运动控制指令的输入输出变量 “Switches”指定的结构体型变量的数组元素数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的数组元素数超过范围		请进行修正，以避免相应指令的输入输出变量中指定的结构体型变量的数组元素数超过范围		请避免指令的输入输出变量中指定的结构体型变量的数组元素数超过范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#572B)。

事件名称	输出信号结构体型变量的数组元素数超过范围		事件代码	5401572CHex *1		
内容	运动控制指令的输入输出变量 “Outputs”指定的结构体型变量的数组元素数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的数组元素数超过范围		请进行修正，以避免相应指令的输入输出变量中指定的结构体型变量的数组元素数超过范围		请避免指令的输入输出变量中指定的结构体型变量的数组元素数超过范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#572C)。

事件名称	机架选项结构体型变量的数组元素数超过范围		事件代码	5401572DHex *1		
内容	运动控制指令的输入输出变量 “TrackOptions”指定的结构体型变量的数组元素数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的结构体型变量的数组元素数超过范围		请进行修正，以避免相应指令的输入输出变量中指定的结构体型变量的数组元素数超过范围		请避免指令的输入输出变量中指定的结构体型变量的数组元素数超过范围	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#572D)。

事件名称	输出信号与机架选项的数组元素数不一致		事件代码	5401572EHex *1		
内容	运动控制指令的输入输出变量 “Outputs”与 “TrackOptions”指定的结构体型变量的数组元素数不一致					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的输出信号结构体型变量与机架选项结构体型变量的数组元素数不一致		请进行修正，使相应指令的输入输出变量中指定的输出信号结构体型变量与机架选项结构体型变量的数组元素数一致		请进行设定，使指令的输入输出变量中指定的输出信号结构体型变量与机架选项结构体型变量的数组元素数一致	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#572E)。

事件名称	无法多重启动运动指令(主轴)		事件代码	5401572FHex *1		
内容	变更了多重启动指令时无法变更的输入输出变量 “Master”					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了多重启动指令时无法变更的输入输出变量 “Master”		请修正程序，以避免在相应指令多重启动时变更输入输出变量 “Master”的值		请编写程序，以避免在相应指令多重启动时变更输入输出变量 “Master”的值	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#572F)。

事件名称	无法多重启动运动指令(位置类型选择)		事件代码	54015730Hex *1		
内容	变更了多重启动指令时无法变更的输入输出变量 “ReferenceType”					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	变更了多重启动指令时无法变更的输入输出变量 “ReferenceType”		请修正程序，以避免在相应指令多重启动时变更输入输出变量 “ReferenceType”的值		请编写程序，以避免在相应指令多重启动时变更输入输出变量 “ReferenceType”的值	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5730)。

事件名称	开关结构体的同一机架指定数超过范围		事件代码	54015731Hex *1		
内容	运动控制指令的输入输出变量 “Switches”指定的 “TrackNumber”的同一机架编号数量超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].Obsr.Active		BOOL		正在产生轴监控信息	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入输出变量指定的开关结构体型变量的 “TrackNumber”指定的同一机架编号数量超过了1个机架可指定的个数范围		请修正数值，以避免 “TrackNumber”指定的同一机架编号数量超过1个机架可指定的个数		请避免 “TrackNumber”指定的同一机架编号数量超过1个机架可指定的个数	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.06以上时发生的错误代码(16#5731)。

事件名称	轴参数无法写入		事件代码	5401573AHex *1		
内容	对未使用轴以外的轴启动了相应指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	对使用轴或未创建轴启动了相应指令		请修正程序，以确保在使用MC_ChangeAxisUse(轴使用变更)指令将对象轴变更成未使用轴后再启动相应指令		请编写程序，以确保相应指令启动时对象轴为未使用轴	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#573A)。

事件名称	轴参数超过设定范围		事件代码	5401573BHex *1		
内容	运动控制指令的输入变量 “AxisParameter”指定的参数超过有效范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “AxisParameter”指定的参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围。 超过范围的参数或参数间的不匹配事项可通过附属信息进行确认		请避免指令的输入参数超过输入变量的范围。 输入变量的有效范围请参阅MC_WriteAxisParameter(轴参数写入)指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#573B)。

事件名称	凸轮属性超过设定范围		事件代码	5401573CHex *1		
内容	运动控制指令的输入变量 “CamProperty”指定的参数超过有效范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “CamProperty”指定的参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围。 超过范围的参数可通过附属信息进行确认		请避免指令的输入参数超过输入变量的范围。	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#573C)。

事件名称	凸轮节点超过设定范围		事件代码	5401573DHex *1		
内容	运动控制指令的输入变量 “CamNodes”指定的参数超过有效范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “CamNodes”指定的参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围。 超过范围的参数可通过附属信息进行确认		请避免指令的输入参数超过输入变量的范围。	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#573D)。

事件名称	凸轮节点类型指定错误			事件代码		5401573EHex *1
内容	运动控制指令的输入变量 “CamNodes”指定的参数非_sMC_CAM_NODE型数组变量					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “CamNodes”指定的参数非_sMC_CAM_NODE型数组变量		请修正程序，以确保在相应指令的输入变量中指定sMC_CAM_NODE型数组变量		请编写程序，以确保在相应指令的输入变量中指定sMC_CAM_NODE型数组变量	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#573E)。

事件名称	凸轮表生成节点数不足			事件代码		5401573FHex *1
内容	运动控制指令的输入变量 “CamNodes”指定的参数数组变量中元素编号0的Phase值为 “0”					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “CamNodes”指定的参数数组变量中元素编号0的Phase(主轴相位)值为 “0”		请修正程序，以确保输入变量 “CamNodes”指定的参数数组变量中元素编号0的Phase(主轴相位)值不为 “0”		请编写程序，以确保输入变量 “CamNodes”指定的参数数组变量中元素编号0的Phase(主轴相位)值不为 “0”	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#573F)。

事件名称	凸轮节点主轴相位非升序			事件代码	54015740Hex *1	
内容	运动控制指令的输入变量 “CamNodes”指定的参数数组变量中Phase的值未按元素编号顺序升序排列					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “CamNodes”指定的参数数组变量中Phase(主轴相位)的值未按元素编号顺序升序排列。或者，舍去有效位第8位后的结果为非升序状态		请修正程序，以确保输入变量 “CamNodes”指定的参数数组变量中Phase(主轴相位)的值按元素编号顺序升序排列		请编写程序，以确保输入变量 “CamNodes”指定的参数数组变量中Phase(主轴相位)的值按元素编号顺序升序排列	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#5740)。

事件名称	凸轮表生成数据点数过多			事件代码	54015741Hex *1	
内容	生成的凸轮数据数超过了运动控制指令的输入变量 “CamTable”指定的凸轮数据变量的数组元素数					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	生成的凸轮表的凸轮数据数超过了指令的输入变量 “CamTable”指定的凸轮数据变量的数组元素数		请修正程序，以避免生成的凸轮表的凸轮数据数超过指令的输入变量 “CamTable”指定的凸轮数据变量的数组元素数。 生成的凸轮表的凸轮数据数请参阅MC_GenerateCamTable(凸轮表生成)指令		请编写程序，以避免生成的凸轮表的凸轮数据数超过指令的输入变量 “CamTable”指定的凸轮数据变量的数组元素数 生成的凸轮表的凸轮数据数请参阅MC_GenerateCamTable(凸轮表生成)指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#5741)。

事件名称	凸轮表生成位移溢出			事件代码		54015742Hex *1
内容	生成的凸轮表的Distance超过了REAL型可表示的范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时/ 执行指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统 定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
发生原因及 其处理	发生原因(推测原因)		处理措施		防止再次发生	
	生成的凸轮表的Distance超过了REAL型可表示的范围		输入变量 “CamNodes”的Curve(曲线形状)指定3次曲线或5次曲线时，请修正InitVel(初始速度)、ConnectingVel(连接速度)、ConnectingAcc(连接加速度)的值，以避免Distance溢出。 Distance的计算方法请参阅MC_GenerateCamTable(凸轮表生成)指令		输入变量 “CamNodes”的Curve(曲线形状)指定3次曲线或5次曲线时，请指定InitVel(初始速度)、ConnectingVel(连接速度)、ConnectingAcc(连接加速度)的值，以避免Distance溢出。 Distance的计算方法请参阅MC_GenerateCamTable(凸轮表生成)指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/ 备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#5742)。

事件名称	生成中止凸轮表的使用			事件代码		54015743Hex *1
内容	相应指令的输入变量 “CamTable”指定了被中止生成的凸轮数据变量					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统 定义变量	变量名称		数据类型		名称	
	_MC_COM.MFaultLvl.Active		BOOL		正在发生MC通用轻度故障	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴 轻度故障	
发生原因及 其处理	发生原因(推测原因)		处理措施		防止再次发生	
	相应指令的输入变量 “CamTable”指定了因MC_GenerateCamTable(凸轮表生成)指令异常而被中止生成的凸轮数据变量		请确认MC_GenerateCamTable(凸轮表生成)指令的输出变量ErrorID(错误代码)、ErrorParameterCode(参数详细代码)、ErrorNodePointIndex(节点元素编号)后修正程序，以生成正常的凸轮数据变量		请编写程序，以确保MC_GenerateCamTable(凸轮表生成)指令生成正常的凸轮数据变量。或编写程序，仅在MC_GenerateCamTable(凸轮表生成)指令正常完成时启动相应指令	
附属信息	附属信息1：异常的发生部位 附属信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4：扩展错误代码(ErrorIDEx)					
注意事项/ 备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.08以上时发生的错误代码(16#5743)。

事件名称	执行ID超过设定范围		事件代码	54015749Hex *1		
内容	运动控制指令的输入变量 “ExecID”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “ExecID”指定的参数超过输入变量的范围		请修正程序，以确保指令的输入变量 “ExecID”指定的输入参数在范围内		请编写程序，以确保指令的输入变量 “ExecID”指定的输入参数在范围内	
附属信息	附属信息1: 异常的发生部位 附属信息2: 异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3: 发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4: 扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.10以上时发生的错误代码(16#5749)。

事件名称	位置偏置超过范围		事件代码	5401574AHex *1		
内容	运动控制指令的输入变量 “OffsetPosition”指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	将指令的输入参数转换为脉冲单位时，已超过带符号40位的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附属信息	附属信息1: 异常的发生部位 附属信息2: 异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3: 发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4: 扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.10以上时发生的错误代码(16#574A)。

事件名称	PDS状态变化指令选择超过范围		事件代码	5401574BHex *1		
内容	运动控制指令的输入变量 “TransitionCmd” 指定的参数超过范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入参数超过输入变量的范围		请修正参数，以避免超过相应指令输入变量的范围		请避免指令的输入参数超过输入变量的范围	
附属信息	附属信息1: 异常的发生部位 附属信息2: 异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3: 发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4: 扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.10以上时发生的错误代码(16#574B)。

事件名称	无法启动单轴位置控制轴运动指令		事件代码	5401574CHex *1		
内容	执行了无法对单轴位置控制轴执行的动作指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		正在发生轴轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	执行了无法对单轴位置控制轴执行的动作指令		请在指令指定轴的轴基本设定的控制功能中设定 “0:全部”。 或者，请确保指定轴基本设定的控制功能设定成 “0:全部” 的轴		同左	
附属信息	附属信息1: 异常的发生部位 附属信息2: 异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附属信息3: 发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附属信息4: 扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附属信息					

*1 CPU单元的单元版本为Ver.1.13以上时发生的错误代码(16#574C)。

事件名称	目标位置正方向软件超限		事件代码	54016440Hex	
内容	指定的位置超过正方向软件限制的范围				
发生源	PLC功能模块	发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令	

系统定义变量	变量名称	数据类型	名称
	_MC_AX[*].MFaultLvl.Active	BOOL	轴正在发生轻度故障
	_MC_GRP[*].MFaultLvl.Active	BOOL	轴组正在发生轻度故障

发生原因及其处理	发生原因(推测原因)	处理措施	防止再次发生
	指令的输入变量 “Position”指定的参数超过正方向软件限制的范围	请修正相应指令的输入变量“Position”指定的参数，以将其控制在正方向软件限制的范围内	请将指令的输入变量 “Position”指定的输入参数控制在正方向软件限制的范围内
	起始位置超过正方向软件限制的范围，执行了指定与软件限制的范围相反方向的动作的指令	请修正程序，以确保相应指令下朝正方向软件限制的范围内移动	请创建程序，以确保起始位置超过正方向软件限制的范围时，朝正方向软件限制的范围内移动
	已指定通过点的MC_MoveCircular2D(2轴圆弧插补)指令的输入变量 “AuxPoint”指定的参数超过正方向软件限制的范围	请修正相应指令的输入变量“AuxPoint”指定的参数，以将其控制在正方向软件限制的范围内	请将已指定通过点的MC_MoveCircular2D(2轴圆弧插补)指令的输入变量 “AuxPoint”指定的参数控制在正方向软件限制的范围内

附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息

事件名称	目标位置负方向软件超限			事件代码		54016441Hex
内容	指定的位置超过负方向软件限制的范围					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令的输入变量 “Position”指定的参数超过负方向软件限制的范围		请修正相应指令的输入变量“Position”指定的参数，以将其控制在负方向软件限制的范围内		请将指令的输入变量 “Position”指定的输入参数控制在负方向软件限制的范围内	
	起始位置超过负方向软件限制的范围，执行了指定与软件限制的范围相反方向的动作的指令		请修正程序，以确保相应指令下朝负方向软件限制的范围内移动		请创建程序，以确保起始位置超过负方向软件限制的范围时，朝负方向软件限制的范围内移动	
	已指定通过点的MC_MoveCircular2D(2轴圆弧插补)指令的输入变量 “AuxPoint”指定的参数超过负方向软件限制的范围		请修正相应指令的输入变量“AuxPoint”指定的参数，以将其控制在负方向软件限制的范围内		请将已指定通过点的MC_MoveCircular2D(2轴圆弧插补)指令的输入变量 “AuxPoint”指定的参数控制在负方向软件限制的范围内	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	指令位置溢出/正在发生下溢			事件代码		54016442Hex
内容	指令位置溢出/正在发生下溢时，执行了定位、溢出/下溢方向的指令或无法确定方向的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	指令位置溢出/正在发生下溢时进行了以下操作 • 执行了定位的指令 • 执行了溢出/下溢方向的连续控制指令 • 执行了无法确定方向的指令(同步功能、扭矩控制)		请执行异常解除，通过原点复位或当前位置预设解除溢出/下溢状态		请避免发生溢出/下溢	
附加信息	附加信息1：因发生源详情而异 轴时：“0” 轴组时：逻辑轴号					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	正方向限制输入中		事件代码		54016443Hex	
内容	正方向限制输入为 “ON”的状态下执行了正方向动作的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	正方向限制输入为 “ON”的状态下，执行了正方向动作的指令；或正方向限制输入为 “ON”的状态下，执行了不指定动作方向的指令。正方向限制输入为 “ON”的状态下，执行了轴组动作指令		请执行异常解除，朝负方向进行复位动作。执行轴组动作指令过程中发生异常时，请将发生异常的轴组设为无效，再进行上述复位动作。再次发生本异常时，请确认正方向限制信号的连接、正方向限制输入的逻辑设定、启动指令的执行条件，再修正错误		请确认正方向限制信号的连接、正方向限制输入的逻辑设定、启动指令的执行条件无误	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	负方向限制输入中		事件代码		54016444Hex	
内容	负方向限制输入为 “ON”的状态下，执行了负方向动作的指令					
发生源	PLC功能模块		发生源详情	指令	检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–	日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令		
系统定义变量	变量名称		数据类型		名称	
	_MC_AX[*].MFaultLvl.Active		BOOL		轴正在发生轻度故障	
	_MC_GRP[*].MFaultLvl.Active		BOOL		轴组正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施		防止再次发生	
	负方向限制输入为 “ON”的状态下，执行了负方向动作的指令；或负方向限制输入为 “ON”的状态下，执行了不指定动作方向的指令。负方向限制输入为 “ON”的状态下，执行了轴组动作指令		请执行异常解除，朝正方向进行复位动作。执行轴组动作指令过程中发生异常时，请将发生异常的轴组设为无效，再进行上述复位动作。再次发生本异常时，请确认负方向限制信号的连接、负方向限制输入的逻辑设定、启动指令的执行条件，再修正错误		请确认负方向限制信号的连接、负方向限制输入的逻辑设定、启动指令的执行条件无误	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)					
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息					

事件名称	伺服主电路电源OFF状态			事件代码		54017422Hex	
内容	伺服驱动器的主电路电源为 “OFF”的状态下，执行了伺服ON						
发生源	PLC功能模块		发生源详情	指令		检测时间	启动指令时
异常的属性	重要程度	监控信息	恢复方法	–		日志类别	系统
发生后的影响	用户程序	继续	动作	按照规格结束相应指令			
系统定义变量	变量名称		数据类型			名称	
	_MC_AX[*].MFaultLvl.Active		BOOL			轴正在发生轻度故障	
发生原因及其处理	发生原因(推测原因)		处理措施			防止再次发生	
	伺服驱动器的主电路电源为“OFF”状态下，执行了伺服ON		请接通发生本异常的轴的伺服驱动器主电路电源，再执行伺服ON			请接通伺服驱动器的主电路电源，再执行伺服ON	
附加信息	附加信息1：异常的发生部位 附加信息2：异常的发生部位详情 电路编号。程序段时为从段开头起的电路编号。ST时为行号 附加信息3：发生异常的指令的指令名称、指令实例名称。有多个备选项时显示多个。无法确定时无显示 附加信息4：扩展错误代码(ErrorIDEx)						
注意事项/备注	发生异常后变更程序时，可能不会正确显示附加信息						

A–4 事件任务无法使用的指令

事件任务是指仅在指定的执行条件成立时执行1次的任务。不是每个任务周期反复执行的任务。因此，无法将包含跨越多个任务周期执行的指令的用户程序分配至事件任务。
下表中的指令为跨越多个任务周期执行的指令。请勿在分配至事件任务的用户程序中使用这些指令。否则编连时会发生异常。

种类	指令	名称	页码
堆栈/表指令	RecSort	记录排序	2–510
模拟控制指令	PIDAT	带自动调谐的PID运算	2–656
	PIDAT_HeatCool	带自动调谐的加热冷却PID运算	2–682
	AC_StepProgram	步程序	2–759
系统控制指令	ResetPLCError	PLC异常解除	2–802
	ResetCJBError	I/O总线异常解除	2–807
	ResetMCError	运动控制异常解除	2–813
	ResetECError	EtherCAT异常解除	2–820
	ResetNXBError	NX总线异常解除	2–825
	GetNXUnitError	NX单元异常状态获取	2–829
	ResetUnit	单元重启	2–838
	RestartNXUnit	NX单元重启	2–844
	NX_ChangeWriteMode	NX单元写入模式变更	2–849
	NX_SaveParam	NX单元参数保存	2–854
	NX_ReadTotalPower OnTime	NX单元累计通电时间读取	2–859
EtherCAT通信指令	EC_CoESDOWrite	CoE SDO写入	2–902
	EC_CoESDORead	CoE SDO读取	2–905
	EC_StartMon	EtherCAT分组监控功能启动	2–910
	EC_StopMon	EtherCAT分组监控功能停止	2–916
	EC_SaveMon	EtherCAT分组保存	2–918
	EC_CopyMon	EtherCAT分组传送	2–920
	EC_DisconnectSlave	EtherCAT从站脱离	2–922
	EC_ConnectSlave	重新加入EtherCAT从站	2–930
	EC_ChangeEnable Setting	EtherCAT从站有效/无效切换	2–932
	NX_WriteObj	NX对象写入	2–951
	NX_ReadObj	NX对象读取	2–966
IO–Link通信指令	IOL_ReadObj	IO–Link设备对象读取	2–974
	IOL_WriteObj	IO–Link设备对象写入	2–982
EtherNet/IP通信指令	CIPOpen	CIPClass3(Large_Forward_Open)连接建立	2–992
	CIPOpenWithDataSize	CIPClass3连接建立(带大小指定)	2–1001
	CIPRead	变量读取(Class3 Explicit信息)	2–1004
	CIPWrite	变量写入(Class3 Explicit信息)	2–1009
	CIPSend	任意Explicit信息发送(Class3)	2–1015
	CIPClose	CIP Class3连接的切断	2–1020
	CIPUCMMRead	变量读取(UCMM Explicit信息)	2–1022
	CIPUCMMWrite	变量写入(UCMM Explicit信息)	2–1027
	CIPUCMMSend	任意Explicit信息发送(UCMM)	2–1034
	SktUDPCreate	UDP Socket建立	2–1045
	SktUDPRcv	UDP Socket接收	2–1052
	SktUDPSend	UDP Socket发送	2–1055
	SktTCPAccept	TCP Socket接受	2–1058
	SktTCPConnect	TCP Socket连接	2–1061
	SktTCPRcv	TCP Socket接收	2–1069

种类	指令	名称	页码
EtherNet/IP通信指令	SktTCPSend	TCP Socket发送	2–1072
	SktGetTCPStatus	TCP Socket的状态读取	2–1075
	SktClose	TCP/UDP Socket闭合	2–1078
	SktClearBuf	TCP/UDP Socket接收缓存清除	2–1081
	SktSetOption	TCP Socket选项设定	2–1083
	ChangeIPAdr	IP地址变更	2–1088
	ChangeFTPAccount	FTP账号变更	2–1097
	ChangeNTPServerAdr	NTP服务器地址变更	2–1101
	FTPGetFileList	FTP服务器的文件列表获取	2–1105
	FTPGetFile	从FTP服务器下载文件	2–1120
	FTPPutFile	文件上传至FTP服务器	2–1128
	FTPRemoveFile	FTP服务器的文件删除	2–1138
	FTPRemoveDir	FTP服务器的目录删除	2–1148
串行通信指令	ExecPMCR	协议宏	2–1154
	SerialSend	串行通信单元 串行端口输出	2–1166
	SerialRcv	串行通信单元 串行端口输入	2–1175
	SerialRcvNoClear	串行通信单元 不删除串行端口输入接收缓存	2–1175
	SendCmd	指令发送	2–1190
	NX_SerialSend	无协议数据的发送	2–1202
	NX_SerialRcv	无协议数据的接收	2–1215
	NX_ModbusRtuCmd	Modbus RTU通用指令发送	2–1229
	NX_ModbusRtuRead	Modbus RTU Read 指令发送	2–1239
	NX_ModbusRtuWrite	Modbus RTU Write 指令发送	2–1249
	NX_SerialSigCtl	串行控制信号的ON/OFF切换	2–1259
	NX_SerialSigRead	串行控制信号的读取	2–1267
	NX_SerialStatusRead	串行端口状态的读取	2–1271
	NX_SerialBufClear	缓存清除	2–1275
	NX_SerialStartMon	串行线路监控的开始	2–1285
	NX_SerialStopMon	串行线路监控的停止	2–1289
SD存储卡指令	FileWriteVar	变量文件写入	2–1294
	FileReadVar	变量文件读取	2–1299
	FileOpen	文件打开	2–1304
	FileClose	文件关闭	2–1308
	FileSeek	文件查找	2–1311
	FileRead	文件读取	2–1314
	FileWrite	文件写入	2–1322
	FileGets	字符串读取	2–1330
	FilePuts	字符串写入	2–1338
	FileCopy	文件复制	2–1346
	FileRemove	文件删除	2–1355
	FileRename	文件名变更	2–1360
	DirCreate	目录创建	2–1366
	DirRemove	目录删除	2–1369
	BackupToMemoryCard	SD存储卡备份	2–1373
时间戳指令	NX_DOutTimeStamp	时间戳数字输出写入	2–1388
	NX_AryDOutTimeStamp	时间戳数字输出数组写入	2–1393

A–5 与NX信息通信异常相关的指令

一次性执行以下多个指令时，可能会发生 “NX 信息通信异常”。发生 “NX 信息通信异常”时，请减少以下指令的执行数。发生 “NX信息通信异常”的条件因通信负载等而异。

种类	指令	名称	页码
系统控制指令	RestartNXUnit	NX单元重启	2–844
	NX_ChangeWriteMode	NX单元写入模式变更	2–849
	NX_SaveParam	NX单元参数保存	2–854
	NX_ReadTotalPower OnTime	NX单元累计通电时间读取	2–859
EtherCAT通信指令	EC_CoESDOWrite	CoE SDO写入	2–902
	EC_CoESDORead	CoE SDO读取	2–905
	EC_StartMon	EtherCAT分组监控功能启动	2–910
	EC_StopMon	EtherCAT分组监控功能停止	2–916
	EC_SaveMon	EtherCAT分组保存	2–918
	EC_CopyMon	EtherCAT分组传送	2–920
	EC_DisconnectSlave	EtherCAT从站脱离	2–922
	EC_ConnectSlave	重新加入EtherCAT从站	2–930
	EC_ChangeEnableSetting	EtherCAT从站有效/无效切换	2–932
	NX_WriteObj	NX对象写入	2–951
	NX_ReadObj	NX对象读取	2–966
IO–Link通信指令	IOL_ReadObj	IO–Link设备对象读取	2–974
	IOL_WriteObj	IO–Link设备对象写入	2–982

版本相关信息

NX信息通信异常是CPU单元Ver.1.05以上且Sysmac Studio Ver.1.06以上时会发生的异常。

A–6 SDO Abort代码一览

表示EtherCAT通信的SDO Abort代码，以供参考。实际通信使用的Abort代码由从站侧规定。编程时，请参阅从站侧的手册。

值	含义
16#05030000	toggle bit无变化
16#05040000	SDO协议超时
16#05040001	无效/不明客户端/伺服指令指定符
16#05040005	超过存储器范围
16#06010000	访问不支持的对象
16#06010001	读取访问只写对象
16#06010002	写入访问只读对象
16#06020000	对象字典中不存在的对象
16#06040041	无法将对象映射至PDO
16#06040042	映射的对象数/长度超过PDO长度
16#06040043	普通的参数不一致
16#06040047	设备的普通的内部不一致
16#06060000	硬件导致的访问失败
16#06070010	数据类型不一致、服务参数长度不一致
16#06070012	数据类型不一致、服务参数过长
16#06070013	数据类型不一致、服务参数过短
16#06090011	子索引不存在
16#06090030	参数值超过范围(仅写入访问)
16#06090031	写入的参数值过大
16#06090032	写入的参数值过小
16#06090036	最大值小于最小值
16#08000000	常见错误
16#08000020	数据无法传送/保存至应用程序
16#08000021	为本地控制，因此无法将数据传送/保存至应用程序
16#08000022	当前的设备状态下，无法将数据传送/保存至应用程序
16#08000023	对象字典动态生成失败或对象字典不存在

*1 从站固有的内部状态的问题

引用源 EtherCAT规格书 Part6 应用层协议规格
文件编号：ETG.1000.6 S(R) V1.0.2(J02)

A–7 版本相关信息

下表中表示因CPU单元和Sysmac Studio各版本更新而变更规格的指令及新指令一览。
还对V1.02的Sysmac Studio显示发生的错误信息 “指令可能引起意外动作”时的处理方法进行说明。

因版本更新而变更规格的指令及新指令一览

指令的适用情况和规格可能因CPU单元和Sysmac Studio的版本而异。该内容如下所示。
对于同时标有CPU单元和Sysmac Studio版本的指令，表示两者的版本均需在该版本以上。

种类	指令	名称	新建/规格变更	版本		页码
				CPU单元	Sysmac Studio	
ST语法指令	FOR	重复	规格变更	–	Ver.1.08	2–44
时序输入指令	R_TRIG	上升沿微分	规格变更	Ver.1.02	–	2–46
时序输出指令	RS	复位优先保持	规格变更	–	Ver.1.03	2–52
	SR	置位优先保持	规格变更	–	Ver.1.03	2–54
比较指令	EQ,=	比较EQ	规格变更	–	Ver.1.02	2–94
	NE,<>	比较NE	规格变更	–	Ver.1.02	2–96
	LT,<	比较LT	规格变更	–	Ver.1.02	2–98
	LE,<=	比较LE	规格变更	–	Ver.1.02	2–98
	GT,>	比较GT	规格变更	–	Ver.1.02	2–98
	GE,>=	比较GE	规格变更	–	Ver.1.02	2–98
	ZoneCmp	区域比较	规格变更	Ver.1.01	Ver.1.02	2–110
计数器指令	CTD	减法计数器	规格变更	–	Ver.1.03	2–146
	CTD_**	减法计数器组	规格变更	–	Ver.1.03	2–148
	CTU	加法计数器	规格变更	–	Ver.1.03	2–151
	CTU_**	加法计数器组	规格变更	–	Ver.1.03	2–153
	CTUD	可逆计数器	规格变更	–	Ver.1.03	2–156
	CTUD_**	可逆计数器组	规格变更	–	Ver.1.03	2–160
数据类型转换指令	EnumToNum	枚举体→整数转换	新建	Ver.1.02	Ver.1.03	2–308
	NumToEnum	整数→枚举体转换	新建	Ver.1.02	Ver.1.03	2–310
选择指令	SEL	位选择	规格变更	Ver.1.02	Ver.1.03	2–326
	MUX	多路复用器	规格变更	Ver.1.02	Ver.1.03	2–328
	AryMax	数组变量的最大值检索	规格变更	Ver.1.01	Ver.1.02	2–339
	AryMin	数组变量的最小值检索	规格变更	Ver.1.01	Ver.1.02	2–339
	ArySerach	数组检索	规格变更	Ver.1.01	Ver.1.02	2–342
移位指令	SHL	左移N位	规格变更	Ver.1.02	Ver.1.03	2–388
	SHR	右移N位	规格变更	Ver.1.02	Ver.1.03	2–388
	ROL	左旋转N位	规格变更	Ver.1.02	Ver.1.03	2–392
	ROR	右旋转N位	规格变更	Ver.1.02	Ver.1.03	2–392
数据转换指令	UTF8ToSJIS	字符代码转换(UTF–8→SJIS)	新建	Ver.1.01	Ver.1.02	2–414
	SJISToUTF8	字符代码转换(SJIS→UTF–8)	新建	Ver.1.01	Ver.1.02	2–416
	PWLApproxNoLineChk	折线近似转换(无折线数据检查)	新建	Ver.1.03	Ver.1.04	2–418
	PWLLineChk	折线数据检查	新建	Ver.1.03	Ver.1.04	2–423
	PackWord	合并2字节	新建	Ver.1.12	Ver.1.16	2–479
	PackDword	合并4字节	新建	Ver.1.12	Ver.1.16	2–481

种类	指令	名称	新建/规格变更	版本		页码
				CPU单元	Sysmac Studio	
堆栈/表指令	RecSearch	记录检索	规格变更	Ver.1.01	Ver.1.02	2–500
	RecRangeSearch	范围指定记录检索	规格变更	Ver.1.01	Ver.1.02	2–505
	RecSort	记录排序	规格变更	Ver.1.01	Ver.1.02	2–510
	RecNum	记录数获取	规格变更	Ver.1.01	Ver.1.02	2–515
				Ver.1.02	Ver.1.03	
	RecMax	记录最大值检索	规格变更	Ver.1.01	Ver.1.02	2–517
	RecMin	记录最小值检索	规格变更	Ver.1.01	Ver.1.02	2–517
字符串指令	AddDelimiter	带分隔符的字符串写入	新建	Ver.1.02	Ver.1.03	2–559
	SubDelimiter	字符串读取分隔符删除	新建	Ver.1.02	Ver.1.03	2–569
时间/时刻指令	TruncTime	时间舍去	新建	Ver.1.01	Ver.1.02	2–642
	TruncDt	日期时刻舍去	新建	Ver.1.01	Ver.1.02	2–646
	TruncTod	时刻舍去	新建	Ver.1.01	Ver.1.02	2–650
模拟控制指令	PIDAT_HeatCool	带自动调谐的加热冷却PID运算	新建	Ver.1.08	Ver.1.09	2–682
	Time ProportionalOut	分时比例输出	新建	Ver.1.02	Ver.1.03	2–719
	LimitAlarm_**	上下限警报组	新建	Ver.1.02	Ver.1.03	2–734
	LimitAlarmDv_**	上下限偏差警报组	新建	Ver.1.02	Ver.1.03	2–738
	LimitAlarmDv StbySeq_**	带待机时序的上下限偏差警报组	新建	Ver.1.02	–	2–742
	ScaleTrans	比例转换	新建	Ver.1.05	Ver.1.06	2–756
	AC_StepProgram	步程序	新建	Ver.1.06	Ver.1.07	2–759
系统控制指令	ResetMCError	运动控制异常解除	规格变更	Ver.1.02	Ver.1.03	2–813
			规格变更	Ver.1.10	Ver.1.12	
	GetECError	EtherCAT异常状态获取	规格变更	Ver.1.02	Ver.1.03	2–822
	ResetNXBError	NX总线异常解除	新建	Ver.1.13	Ver.1.17	2–825
	GetNXBError	NX总线异常状态获取	新建	Ver.1.13	Ver.1.17	2–827
	GetNXUnitError	NX单元异常状态获取	新建	Ver.1.13	Ver.1.17	2–829
	RestartNXUnit	NX单元重启	新建	Ver.1.05	Ver.1.06	2–844
			规格变更	Ver.1.07	Ver.1.08	
	NX_Change WriteMode	NX单元写入模式变更	新建	Ver.1.05	Ver.1.06	2–849
	NX_SaveParam	NX单元参数保存	新建	Ver.1.05	Ver.1.06	2–854
	NX_ReadTotal PowerOnTime	NX单元累计通电时间读取	新建	Ver.1.10	Ver.1.12	2–859
	PLC_ReadTotal PowerOnTime	PLC累计通电时间读取	新建	Ver.1.13	Ver.1.17	2–866
程序控制指令	PrgStart	程序执行指令	新建	Ver.1.08	Ver.1.09	2–870
	PrgStop	程序停止指令	新建	Ver.1.08	Ver.1.09	2–878
	PrgStatus	程序状态读取	新建	Ver.1.08	Ver.1.09	2–896

种类	指令	名称	新建/规格变更	版本		页码
				CPU单元	Sysmac Studio	
EtherCAT通信指令	EC_StartMon	EtherCAT分组监控功能启动	规格变更	Ver.1.10	Ver.1.12 *1	2–910
	EC_StopMon	EtherCAT分组监控功能停止	规格变更	Ver.1.10	Ver.1.12 *1	2–916
	EC_SaveMon	EtherCAT分组保存	规格变更	Ver.1.10	Ver.1.12 *1	2–918
	EC_CopyMon	EtherCAT分组传送	规格变更	Ver.1.10	Ver.1.12 *1	2–920
	EC_Change EnableSetting	EtherCAT从站有效/无效切换	新建	Ver.1.04	Ver.1.05	2–932
	NX_WriteObj	NX对象写入	新建	Ver.1.05	Ver.1.06	2–951
	NX_ReadObj	NX对象读取	新建	Ver.1.05	Ver.1.06	2–966
IO–Link通信指令	IOL_ReadObj	IO–Link设备对象读取	新建	Ver.1.12	Ver.1.16	2–974
	IOL_WriteObj	IO–Link设备对象写入	新建	Ver.1.12	Ver.1.16	2–982
EtherNet/IP通信指令	CIPOpenWith DataSize	CIPClass3连接建立(带大小变更)	新建	Ver.1.06	Ver.1.07	2–1001
	CIPSend	任意Explicit信息发送(Class3)	规格变更	Ver.1.11	Ver.1.15	2–1015
	CIPUCMMSend	任意Explicit信息发送(UCMM)	规格变更	Ver.1.11	Ver.1.15	2–1034
	SktUDPCreate	UDP Socket建立	规格变更	Ver.1.03	–	2–1045
			规格变更	Ver.1.10	Ver.1.13	
	SktTCPAccept	TCP Socket接受	规格变更	Ver.1.03	–	2–1058
	SktTCPConnect	TCP Socket连接	规格变更	Ver.1.03	–	2–1061
	SktSetOption	TCP Socket选项设定	新建	Ver.1.12 *2	Ver.1.16	2–1083
	ChangeIPAdr	IP地址变更	新建	Ver.1.02	Ver.1.03	2–1088
			规格变更	Ver.1.10	Ver.1.13	
	ChangeFTPAccount	FTP账号变更	新建	Ver.1.02	Ver.1.03	2–1097
			规格变更	Ver.1.10	Ver.1.13	
	ChangeNTP ServerAdr	NTP服务器地址变更	新建	Ver.1.02	Ver.1.03	2–1101
			规格变更	Ver.1.10	Ver.1.13	
	FTPGetFileList	FTP服务器的文件列表获取	新建	Ver.1.08	Ver.1.09	2–1105
			规格变更	Ver.1.09	Ver.1.10	
	FTPGetFile	从FTP服务器下载文件	新建	Ver.1.08	Ver.1.09	2–1120
			规格变更	Ver.1.09	Ver.1.10	
	FTPPutFile	文件上传至FTP服务器	新建	Ver.1.08	Ver.1.09	2–1128
			规格变更	Ver.1.09	Ver.1.10	
	FTPRemoveFile	FTP服务器的文件删除	新建	Ver.1.08	Ver.1.09	2–1138
			规格变更	Ver.1.09	Ver.1.10	
	FTPRemoveDir	FTP服务器的目录删除	新建	Ver.1.08	Ver.1.09	2–1148
			规格变更	Ver.1.09	Ver.1.10	
串行通信指令	SerialRcvNo Clear *3	不删除串行通信单元串行端口输入接收缓存	新建	Ver.1.03	Ver.1.04	2–1175
	NX_SerialSend	无协议数据的发送	新建	Ver.1.11	Ver.1.15	2–1202
	NX_SerialRcv	无协议数据的接收	新建	Ver.1.11	Ver.1.15	2–1215
	NX_ModbusRtu Cmd	Modbus RTU通用指令发送	新建	Ver.1.11	Ver.1.15	2–1229

种类	指令	名称	新建/规格变更	版本		页码
				CPU单元	Sysmac Studio	
串行通信指令	NX_ModbusRtu Read	Modbus RTU Read 指令发送	新建	Ver.1.11	Ver.1.15	2–1239
	NX_ModbusRtu Write	Modbus RTU Write 指令发送	新建	Ver.1.11	Ver.1.15	2–1249
	NX_SerialSigCtl	串行控制信号的ON/OFF切换	新建	Ver.1.11	Ver.1.15	2–1259
	NX_SerialSigRead	串行控制信号的读取	新建	Ver.1.13	Ver.1.17	2–1267
	NX_SerialStatus Read	串行端口状态的读取	新建	Ver.1.13	Ver.1.17	2–1271
	NX_SerialBufClear	缓存清除	新建	Ver.1.11	Ver.1.15	2–1275
	NX_SerialStartMon	串行线路监控的开始	新建	Ver.1.11	Ver.1.15	2–1285
	NX_SerialStopMon	串行线路监控的停止	新建	Ver.1.11	Ver.1.15	2–1289
SD存储卡指令	BackupTo MemoryCard	SD存储卡备份	新建	Ver.1.08	Ver.1.09	2–1373
时间戳指令	NX_DOut TimeStamp	时间戳数字输出写入	新建	Ver.1.06	Ver.1.07	2–1388
	NX_AryDOut TimeStamp	时间戳数字输出数组写入	新建	Ver.1.06	Ver.1.07	2–1393
其它指令	GetMyTask Interval	我的任务设定周期读取	新建	Ver.1.08	Ver.1.09	2–1411
	ActEventTask	事件任务启动	新建	Ver.1.03	Ver.1.04	2–1420

*1 NJ101 CPU单元时为Ver.1.13以上的组合。

*2 无法在NX1P2 CPU单元Ver.1.13中使用。

*3 SerialRcvNoClear指令可用于Ver.1.03以上版本的CPU单元、Ver.1.04以上版本的Sysmac Studio、Ver.2.1以上版本的串行通信单元的组合。

显示错误信息 “指令可能引起意外动作”时的处理方法

Sysmac Studio可能会显示如下的错误信息，即“指令可能引起意外动作”。详情请通过指令语手册(基本篇)进行确认。这是由用户程序的限制引起的，可通过修正用户程序进行处理。
下面对该错误信息的显示条件及其处理方法进行说明。

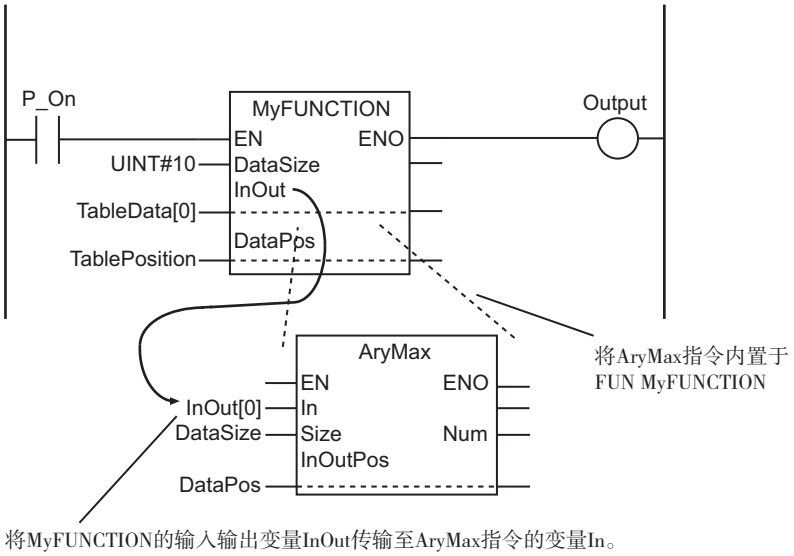
版本相关信息

仅Ver.1.02的Sysmac Studio会显示该错误信息。

错误信息显示条件

在FB和FUN内将该FB和FUN的输入输出变量传输至特定指令的特定变量时显示错误信息。
显示错误信息的指令和变量一览将在下文记载。

显示错误信息的用户程序的示例



显示错误信息的指令和变量

仅特定指令的特定变量会发生该异常。其一览如下所示。

种类	指令名称	名称	变量名称	页码
比较指令	TableCmp	表比较	Table,AryOut	2–113
	AryCmpEQ	数组整体比较EQ	In1,In2,AryOut	2–116
	AryCmpNE	数组整体比较NE	In1,In2,AryOut	2–116
	AryCmpLT	数组整体比较LT	In1,In2,AryOut	2–118
	AryCmpLE	数组整体比较LE	In1,In2,AryOut	2–118
	AryCmpGT	数组整体比较GT	In1,In2,AryOut	2–118
	AryCmpGE	数组整体比较GE	In1,In2,AryOut	2–118
	AryCmpEQV	数组元素比较EQ	In1,AryOut	2–121
	AryCmpNEV	数组元素比较NE	In1,AryOut	2–121
	AryCmpLTV	数组元素比较LT	In1,AryOut	2–123

种类	指令名称	名称	变量名称	页码
比较指令	AryCmpLEV	数组元素比较LE	In1,AryOut	2–123
	AryCmpGTV	数组元素比较GT	In1,AryOut	2–123
	AryCmpGEV	数组元素比较GE	In1,AryOut	2–123
算术指令	AryAdd	数组整体加法	In1,In2,AryOut	2–218
	AryAddV	数组元素加法	In1,AryOut	2–220
	ArySub	数组整体减法	In1,In2,AryOut	2–222
	ArySubV	数组元素减法	In1,AryOut	2–224
	AryMean	数组元素的平均值计算	In	2–226
	ArySD	数组元素的标准偏差	In	2–228
BCD转换指令	AryToBCD	数组整体BCD转换	In,AryOut	2–253
	AryToBin	数组整体无符号整数转换	In,AryOut	2–255
位串运算指令	AryAnd	数组逻辑积	In1,In2,AryOut	2–322
	AryOr	数组逻辑和	In1,In2,AryOut	2–322
	AryXor	数组异或	In1,In2,AryOut	2–322
	AryXorN	数组同或	In1,In2,AryOut	2–322
选择指令	AryMax	数组变量的最大值检索	In	2–339
	AryMin	数组变量的最小值检索	In	2–339
	ArySearch	数组检索	In	2–342
数据传送指令	TransBits	多位传送	InOut	2–353
	AryExchange	数组数据交换	InOut1,InOut2	2–361
	AryMove	数组的传送	In,AryOut	2–363
	Clear	初始化	InOut	2–365
移位指令	AryShiftReg	移位寄存器	InOut	2–380
	AryShiftRegLR	左右移位寄存器	InOut	2–382
	ArySHL	数组左移N个元素	InOut	2–385
	ArySHR	数组右移N个元素	InOut	2–385
	NSHLC	N位数据带CY左移N位	InOut	2–390
	NSHRC	N位数据带CY右移N位	InOut	2–390
数据转换指令	Decoder	位译码器	InOut	2–399
	Encoder	位编码器	In	2–402
	ColmToLine_**	位串→位行转换组	In	2–405
	LineToColm	位行→位串转换	InOut	2–407
	PWLApprox	折线近似转换	Line	2–418
	MovingAverage	移动平均	Buf	2–426
	StringToAry	字符串→数组转换	AryOut	2–455
	AryToString	数组→字符串转换	In	2–457
	DispartDigit	4位分离	AryOut	2–459
	UniteDigit_**	4位结合组	In	2–461
	Dispart8Bit	以字节为单位的数据分离	AryOut	2–463
	Unite8Bit_**	以字节为单位的数据组合组	In	2–465
	ToAryByte	转换为字节数组	In,AryOut	2–467
	AryByteTo	从字节数组转换	In,OutVal	2–472
	SizeOfAry	获取数组元素数	In	2–477
堆栈/表指令	StackPush	堆栈数据保存	InOut	2–484
	StackFIFO	先入先出	InOut,OutVal	2–493
	StackLIFO	后入先出	InOut,OutVal	2–493
	StackIns	堆栈数据插入	InOut	2–496
	StackDel	堆栈数据删除	InOut	2–498
	RecSearch	记录检索	In,Member, InOutPos	2–500
	RecRangeSearch	范围指定记录检索	In,Member, InOutPos	2–505
	RecSort	记录排序	InOut, Member	2–510
	RecNum	记录数获取	In,Member	2–515

种类	指令名称	名称	变量名称	页码
堆栈/表指令	RecMax	记录最大值检索	In,Member, InOutPos	2–517
	RecMin	记录最小值检索	In,Member, InOutPos	2–517
FCS指令	AryLRC_**	LRC值计算(数组)组	In	2–530
	AryCRCCCITT	CRC–CCITT值计算(数组)	In	2–532
	AryCRC16	CRC–16值计算(数组)	In	2–534
系统控制指令	SetAlarm	用户异常发生	Info1, Info2	2–793
	SetInfo	用户信息生成	Info1, Info2	2–836
EtherCAT通信指令	EC_CoESDOWrite	CoE SDO写入	WriteDat	2–902
	EC_CoESDORead	CoE SDO读取	ReadDat	2–905
EtherNet/IP通信指令	CIPRead	变量读取(Class3 Explicit信息)	DstDat	2–1004
	CIPWrite	变量写入(Class3 Explicit信息)	SrcDat	2–1009
	CIPSend	任意Explicit信息发送(Class3)	ServiceDat, RespServiceDat	2–1015
	CIPUCMMRead	变量读取(UCMM Explicit信息)	DstDat	2–1022
	CIPUCMMWrite	变量写入(UCMM Explicit信息)	SrcDat	2–1027
	CIPUCMMSend	任意Explicit信息发送(UCMM)	ServiceDat, RespServiceDat	2–1034
	SktUDPRcv	UDP Socket接收	RcvDat	2–1052
	SktUDPSend	UDP Socket发送	SendDat	2–1055
	SktTCPRcv	TCP Socket接收	RcvDat	2–1069
	SktTCPSend	TCP Socket发送	SendDat	2–1072
串行通信指令	ExecPMCR	协议宏	SrcDat, DstDat	2–1154
	SerialSend	串行通信单元 串行端口输出	SrcDat	2–1166
	SerialRcv	串行通信单元 串行端口输入	DstDat	2–1175
	SendCmd	指令发送	CmdDat, RespDat	2–1190
SD存储卡指令	FileWriteVar	变量文件写入	WriteVar	2–1294
	FileReadVar	变量文件读取	ReadVar	2–1299
	FileRead	文件读取	ReadBuf	2–1314
	FileWrite	文件写入	WriteBuf	2–1322
运动控制指令	MC_SetCamTable Property	更新凸轮表属性	CamTable	(*)
	MC_SaveCamTable	保存凸轮表	CamTable	(*)
	MC_Write	写入MC设定	Target, SettingValue	(*)
	MC_CamIn	凸轮动作开始	CamTable	(*)
	MC_ChangeAxes InGroup	写入轴组构成轴	Axes	(*)
其它指令	ChkRange	范围型变量检查	Val	2–1406

* 详情请参阅 “NJ/NX系列 指令基准手册 运动篇(SBCE–364)”。

处理方法

需修正用户程序以避免发生错误信息。其方法有以下2种。

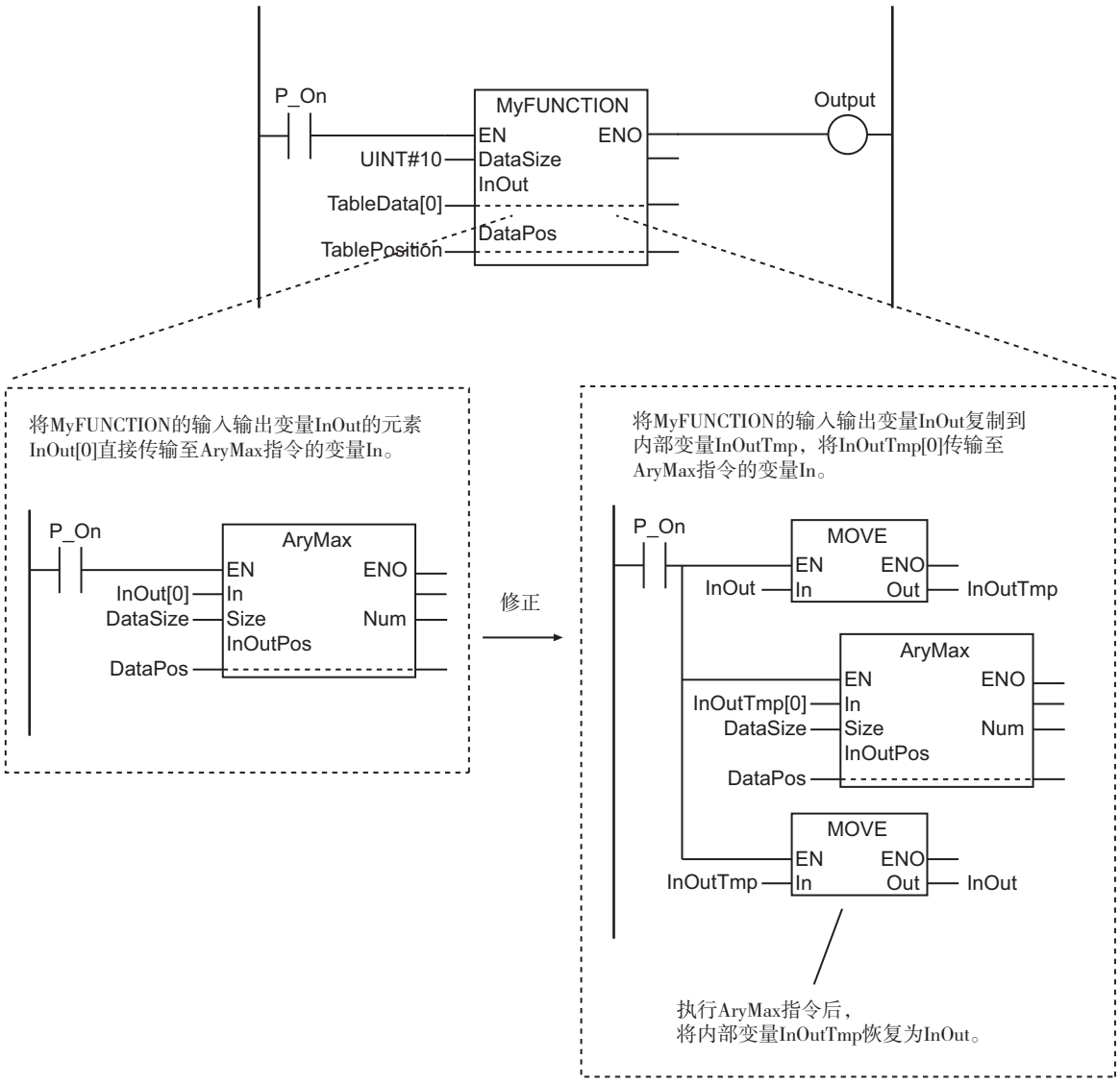
- 将输入输出变量复制到FB、FUN的内部变量后传输至指令。
- 将指令移出FB、FUN。

下面分别对这2种方法进行说明。

● 将输入输出变量复制到FB、FUN的内部变量后传输至指令的方法

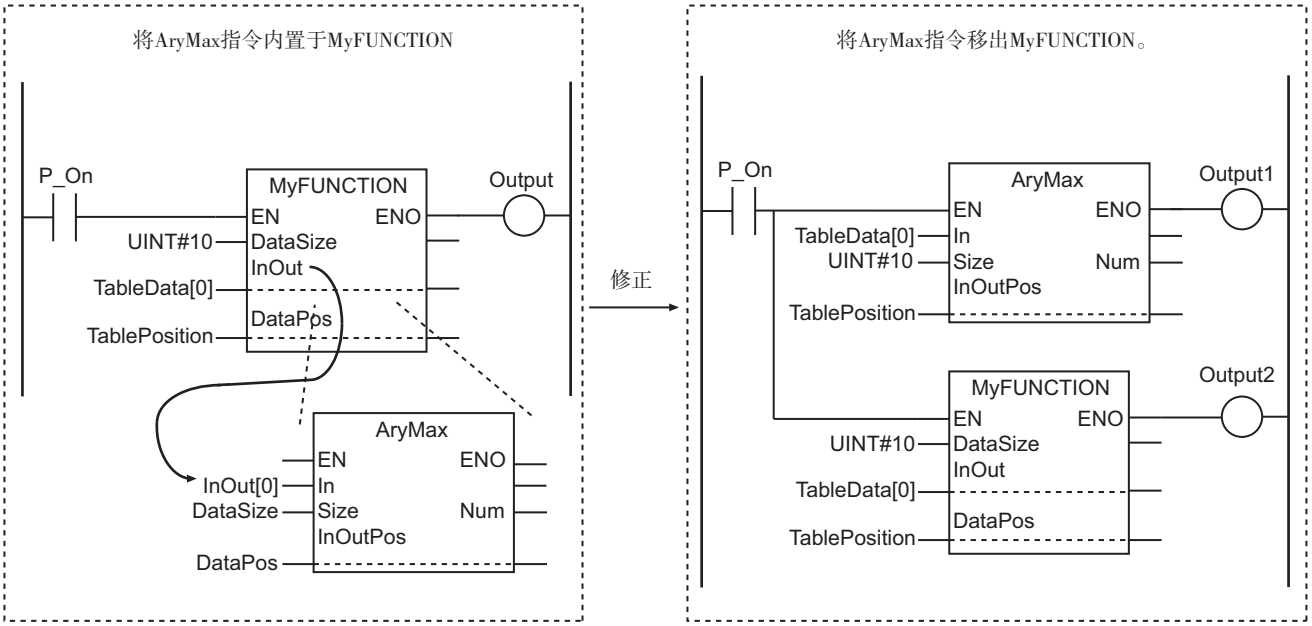
不直接将FB、FUN的输入输出变量传输至指令，而是复制到内部变量后传输。
如果该内部变量的值因执行指令而发生变化，则执行指令后，重新将内部变量复制到原来的输入输出变量。

Clear指令无法使用该方法。如果已使用Clear指令，则请采用将指令移出FB、FUN的方法。



● **将指令移出FB、FUN的方法**

不将指令内置于FB、FUN，而是移出。



即使显示错误信息也无问题时

即使显示错误信息，有时也可照常使用指令。可否使用指令将取决于传输至FB、FUN的输入输出变量的参数。

其条件如下所示。

可否使用指令	传输至FB、FUN的输入输出变量的参数
可使用	基本数据类型、枚举体、整个数组、整个结构体、整个联合体
无法使用	数组的1个元素、结构体的1个结构要素、联合体的1个结构要素

● 可使用指令的示例

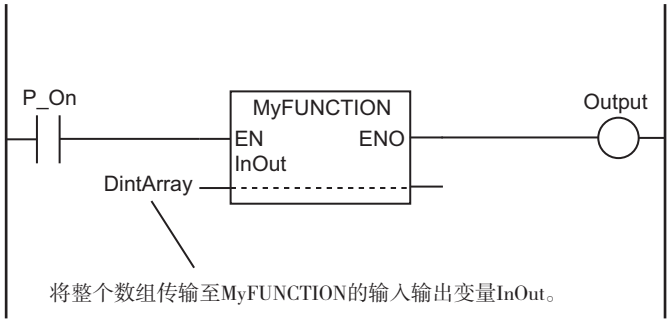
以下示例中，将整个数组传输至FB、FUN的输入输出变量，因此可使用已内置的指令。

MyFUNCTION的调用源的变量表

名称	数据类型
DintArray	ARRAY[0..9] OF DINT

MyFUNCTION的变量表

名称	数据类型
InOut	ARRAY[0..9] OF DINT



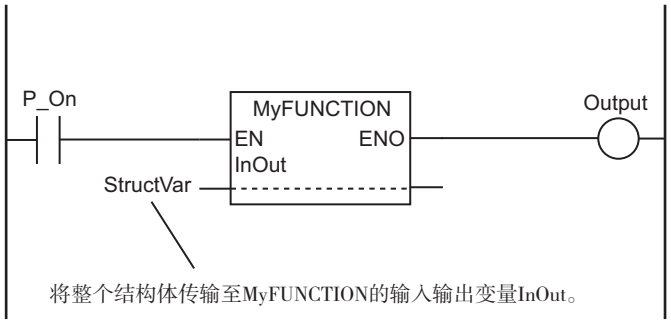
以下示例中，将整个结构体传输至FB、FUN的输入输出变量，因此可使用已内置的指令。

MyFUNCTION的调用源的变量表

名称	数据类型
StructVar	STRUCT

MyFUNCTION的变量表

名称	数据类型
InOut	STRUCT



● 无法使用指令的示例

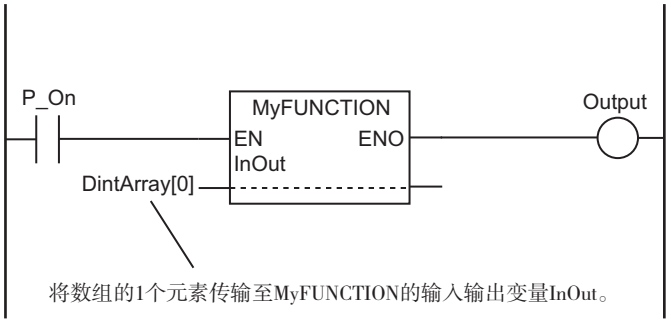
以下示例中，将数组的1个元素传输至FB、FUN的输入输出变量，因此无法使用已内置的指令。

MyFUNCTION的调用源的变量表

名称	数据类型
DintArray	ARRAY[0..9] OF DINT

MyFUNCTION的变量表

名称	数据类型
InOut	DINT



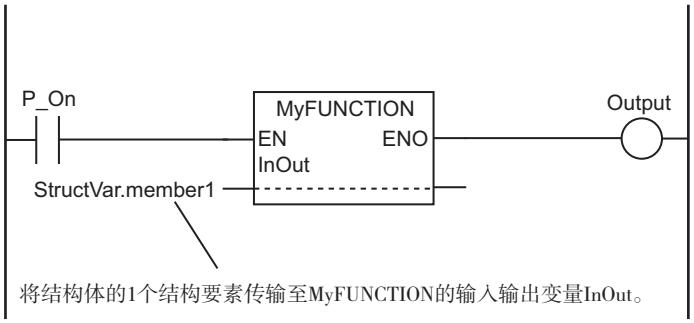
以下示例中，将结构体的1个结构要传输至FB、FUN的输入输出变量，因此无法使用已内置的指令。

MyFUNCTION的调用源的变量表

名称	数据类型
StructVar	STRUCT

MyFUNCTION的变量表

名称	数据类型
InOut	DINT



异常检测的选项设定

如上文所述，即使显示错误信息，有时也可使用指令。Sysmac Studio中可选择是否检测该异常。
请在Sysmac Studio的选项设定画面中取消勾选可选项目“程序检查”的“将输入输出变量传输至特定指令的特定自变量时出错”，以避免检测该异常。Sysmac Studio的操作详情请参阅 🕮《Sysmac Studio Version 1 操作手册(SBCA-362C之后)》。
取消勾选前，请务必确认用户程序中的所有指令均处于可使用的状态。即使取消勾选，也会显示该信息作为警告。

⚠ 注 意

取消勾选该选项时，指令可能引起意外动作，并对装置造成影响。
请务必确认与 🕮“即使显示错误信息也无问题时(P.A-196)”中记载的可使用的条件相符。

版本相关信息

仅Sysmac Studio Ver.1.02会显示错误信息，可进行上述的选项设定。

I

索引

索引

Symbols

Numerics

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

W

X

Y

Z

购买欧姆龙产品的客户须知

承诺事项

承蒙对欧姆龙株式会社(以下简称“本公司”)产品的一贯厚爱和支持，藉此机会再次深表谢意。
如果未特别约定，无论贵司从何处购买的产品，都将适用本承诺事项中记载的事项。
请在充分了解这些注意事项基础上订购。

1. 定义

本承诺事项中的术语定义如下。

(1) “本公司产品”：是指“本公司”的FA系统机器、通用控制器、传感器、电子/结构部件。
(2) “产品目录等”：是指与“本公司产品”有关的欧姆龙综合产品目录、FA系统设备综合产品目录、安全组件综合产品目录、电子/机构部件综合产品目录以及其他产品目录、规格书、使用说明书、操作指南等，包括以电子数据方式提供的资料。
(3) “使用条件等”：是指在“产品目录等”资料中记载的“本公司产品”的使用条件、额定值、性能、运行环境、操作使用方法、使用时的注意事项、禁止事项以及其他事项。
(4) “客户用途”：是指客户使用“本公司产品”的方法，包括将“本公司产品”组装或运用到客户生产的部件、电子电路板、机器、设备或系统等产品中。
(5) “适用性等”：是指在“客户用途”中“本公司产品”的(a)适用性、(b)动作、(c)不侵害第三方知识产权、(d)法规法令的遵守以及(e)满足各种规格标准。

2. 关于记载事项的注意事项

对“产品目录等”中的记载内容，请理解如下要点。

(1) 额定值及性能值是在单项试验中分别在各条件下获得的值，并不构成对各额定值及性能值的综合条件下获得值的承诺。
(2) 提供的参考数据仅作为参考，并非可在该范围内一直正常运行的保证。
(3) 应用示例仅作参考，不构成对“适用性等”的保证。
(4) 如果因技术改进等原因，“本公司”可能会停止“本公司产品”的生产或变更“本公司产品”的规格。

3. 使用时的注意事项

选用及使用本公司产品时请理解如下要点。

(1) 除了额定值、性能指标外，使用时还必须遵守“使用条件等”。
(2) 客户应事先确认“适用性等”，进而再判断是否选用“本公司产品”。“本公司”对“适用性等”不做任何保证。
(3) 对于“本公司产品”在客户的整个系统中的设计用途，客户应负责事先确认是否已进行了适当配电、安装等事项。
(4) 使用“本公司产品”时，客户必须采取如下措施：(i)相对额定值及性能指标，必须在留有余量的前提下使用“本公司产品”，并采用冗余设计等安全设计(ii)所采用的安全设计必须确保即使“本公司产品”发生故障时也可将“客户用途”中的危险降到最小程度、(iii)构建随时提示使用者危险的完整安全体系、(iv)针对“本公司产品”及“客户用途”定期实施各项维护保养。
(5) “本公司产品”是作为应用于一般工业产品的通用产品而设计生产的。如果客户将“本公司产品”用于以下所列用途，则本公司对产品不作任何保证。但“本公司”已表明可用于特殊用途，或已与客户有特殊约定时，另行处理。
  (a) 必须具备很高安全性的用途(例：核能控制设备、燃烧设备、航空/宇宙设备、铁路设备、升降设备、娱乐设备、医疗设备、安全装置、其他可能危及生命及人身安全的用途)
  (b) 必须具备很高可靠性的用途(例：燃气、自来水、电力等供应系统、24小时连续运行系统、结算系统、以及其他处理权利、财产的用途等)
  (c) 具有苛刻条件或严酷环境的用途(例：安装在室外的设备、会受到化学污染的设备、会受到电磁波影响的设备、会受到振动或冲击的设备等)
  (d) “产品目录等”资料中未记载的条件或环境下的用途
(6) 除了不适用于上述3.(5)(a)至(d)中记载的用途外，“本产品目录等资料中记载的产品”也不适用于汽车(含二轮车，以下同)。请勿配置到汽车上使用。关于汽车配置用产品，请咨询本公司销售人员。

4. 保修条件

“本公司产品”的保修条件如下。

(1) 保修期限　自购买之日起1年。(但是，“产品目录等”资料中有明确说明时除外。)
(2) 保修内容　对于发生故障的“本公司产品”，由“本公司”判断并可选择以下其中之一方式进行保修。
  (a) 在本公司的维修保养服务点对发生故障的“本公司产品”进行免费修理(但是对于电子、结构部件不提供修理服务。)
  (b) 对发生故障的“本公司产品”免费提供同等数量的替代品
(3) 当故障因以下任何一种情形引起时，不属于保修的范围。
  (a) 将“本公司产品”用于原本设计用途以外的用途
  (b) 超过“使用条件等”范围的使用
  (c) 违反本注意事项“3. 使用时的注意事项”的使用
  (d) 非因“本公司”进行的改装、修理导致故障时
  (e) 非因“本公司”出品的软件导致故障时
  (f) “本公司”生产时的科学、技术水平无法预见的原因
  (g) 除上述情形外的其它原因，如“本公司”或“本公司产品”以外的原因(包括天灾等不可抗力)

5. 责任限制

本承诺事项中记载的保修是关于“本公司产品”的全部保证。对于因“本公司产品”而发生的其他损害，“本公司”及“本公司产品”的经销商不负任何责任。

6. 出口管理

客户若将“本公司产品”或技术资料出口或向境外提供时，请遵守中国及各国关于安全保障进出口管理方面的法律、法规。否则，“本公司”有权不予提供“本公司产品”或技术资料。

欧姆龙自动化（中国）有限公司

欧姆龙自动化（中国）有限公司北京分公司
欧姆龙自动化（中国）有限公司天津分公司
欧姆龙自动化（中国）有限公司广州分公司

欢迎关注
欧姆龙自动化微信

技术咨询
网　址：http://www.fa.omron.com.cn
400咨询热线：400-820-4535

上海总公司	021-50372222	太原事务所	0351-5229870	中山事务所	0760-88224545	汕头事务所	0754-88706001
南京事务所	025-83240556	天津分公司	022-83191580	福州事务所	0591-88088551	香港事务所	00852-23753827
徐州事务所	0516-83736516	沈阳事务所	024-22815131	南宁事务所	0771-5531371		
武汉事务所	027-82282145	西安事务所	029-88851505				
苏州事务所	0512-68669277	银川联络处	0951-5670076				
昆山事务所	0512-50110866	成都事务所	028-86765345				
杭州事务所	0571-87652855	绵阳联络处	0816-2687423				
宁波事务所	0574-27888220	自贡联络处	0813-8255616				
温州事务所	0577-88919195	重庆事务所	023-68796406				
合肥事务所	0551-63639629	大连事务所	0411-39948181				
长沙事务所	0731-84585551	哈尔滨事务所	0451-53009917				
无锡事务所	0510-85169303	昆明事务所	0871-63527224				
张家港事务所	0512-56313157	兰州事务所	0931-8720101				
南昌事务所	0791-86304711	长春事务所	0431-81928301				
郑州事务所	0371-65585192	乌鲁木齐事务所	0991-5198587				
北京分公司	010-57395399	贵阳事务所	0851-4812320				
唐山事务所	0315-6328518	广州分公司	020-87557798				
石家庄事务所	0311-86918122	深圳事务所	0755-26948238				
济南事务所	0531-82929795	厦门事务所	0592-2686709				
青岛事务所	0532-66775819	东莞事务所	0769-22423200				
烟台事务所	0535-6865018	佛山事务所	0757-83305268				

特约店

注：规格如有变更，恕不另行通知。请以最新产品说明书为准。

Cat. No. SBCA-CN5-360W

上海印刷
201706